Web Application Vulnerability Assessment Essentials: Your First Step to a Highly Secure Web Site

If an organization isn't taking a systematic and proactive approach to web security, and to running a web application vulnerability assessment in particular, then that organization isn't defended against the most rapidly increasing class of attacks. Web-based attacks can lead to lost revenue, the theft of customers' personally identifiable financial information, and falling out of regulatory compliance with a multitude of government and industry mandates: the Payment Card Industry Data Security Standard (PCI) for merchants, HIPAA for health care organizations or Sarbanes-Oxley for publicly traded companies. In fact, the research firm Gartner estimates that 75 percent of attacks on web security today are aimed straight at the application layer.

While they're described with such obscure names as Cross-Site Scripting, SQL Injection, or directory transversal, mitigating the risks associated with web application vulnerabilities and the attack methods that exploit them needn't be beyond the reach of any organization. This article, the first in a three-part series, will provide an overview of what you need to know to perform a vulnerability assessment to check for web security risks. It'll show you what you can reasonably expect a web application security scanner to accomplish, and what types of assessments still require expert eyes. The following two articles will show you how to remedy the web security risks a vulnerability assessment will uncover (and there'll be plenty to do), and the final segment will explain how to instill the proper levels of awareness, policies, and technologies required to keep web application security flaws to a minimum - from an application's conception, design, and coding, to its life in production.

Just What Is a Web Application Vulnerability Assessment?

A web application vulnerability assessment is the way you go about identifying the mistakes in application logic, configurations, and software coding that jeopardize the *availability* (things like poor input validation errors that can make it possible for an attacker to inflict costly system and application crashes, or worse), *confidentiality* (SQL Injection attacks, among many other types of attacks that make it possible for attackers to gain access to confidential information), and *integrity* of your data (certain attacks make it possible for attackers to change pricing information, for example).

The only way to be as certain as you can be that you're not at risk for these types of vulnerabilities in web security is to run a vulnerability assessment on your applications and infrastructure. And to do the job as efficiently, accurately, and comprehensively as possible requires the use of a web application vulnerability scanner, plus an expert savvy in application vulnerabilities and how attackers exploit them.

Web application vulnerability scanners are very good at what they do: identifying technical programming mistakes and oversights that create holes in web security. These are coding errors, such as not checking input strings, or failure to properly filter database queries, that let attackers slip on in, access confidential information, and even crash your applications. Vulnerability scanners automate the process of finding these types of web security issues; they can tirelessly crawl through an application performing a vulnerability assessment, throwing countless variables into input fields in a matter of hours, a process that could take a person weeks to do manually.

Unfortunately, technical errors aren't the only problems you need to address. There is another class of web security vulnerabilities, those that lay within the business logic of application and system flow that still require human eyes and experience to identify successfully. Whether called an ethical hacker or a web security

consultant, there are times (especially with newly developed and deployed applications and systems) that you need someone who has the expertise to run a vulnerability assessment in much the way a hacker will.

Just as is the case with technical errors, business logic errors can cause serious problems and weaknesses in web security. Business logic errors can make it possible for shoppers to insert multiple coupons in a shopping cart - when this shouldn't be allowed - or for site visitors to actually guess the usernames of other customers (such as directly in the browser address bar) and bypass authentication processes to access others' accounts. With business logic errors, your business may be losing money, or customer information may be stolen, and you'll find it tough to figure out why; these transactions would appear legitimately conducted to you.

Since business logic errors aren't strict syntactical slip-ups, they often require some creative thought to spot. That's why scanners aren't highly effective at finding such problems, so these problems need to be identified by a knowledgeable expert performing a vulnerability assessment. This can be an in-house web security specialist (someone fully detached from the development process), but an outside consultant would be preferable. You'll want a professional who has been doing this for a while. And every company can benefit from a third-party audit of its web security. Fresh eyes will find problems your internal team may have overlooked, and since they'll have helped hundreds of other companies, they'll be able to run a vulnerability assessment and quickly identify problems that need to be addressed.

Conducting Your Vulnerability Assessment: The First Steps

There are a number of reasons your organization may need to conduct a vulnerability assessment. It could be simply to conduct a checkup regarding your overall web security risk posture. But if your organization has more than a handful of applications and a number of servers, a vulnerability assessment of such a large scope could be overwhelming. The first thing you need to decide is what applications need to be assessed, and why. It could be part of your PCI DSS requirements, or to meet HIPAA requirements. Or the scope could be the web security of a single, ready-to-be-deployed application.

Once you've figured out the scope, you need to prioritize the applications that need to be assessed. If you're accessing a single, new application, that decision is easy. But if you're on the precipice of accessing every web application in your architecture, you have some decisions to make. Whether you're looking at the web security of applications your own, or only those that take part in online sales transactions, you need to inventory and prioritize the applications to be assessed.

Depending on the scope and purpose of your vulnerability assessment, it makes sense to start looking at the web security of your crucial applications first - for instance, those that conduct the most transactions or dollar volume - and work down from there. Or it could be starting with all applications that touch those that process and store sales transactions.

No matter your scope, or the purpose of your vulnerability assessment, other aspects of your architecture always need to be considered when listing and prioritizing your applications. For instance, any externally facing applications - even those that don't contain sensitive information - need to be given high priority. The same is true for externally hosted applications, whether they are Internet-facing or directly connected to back-end systems. Any applications that are accessible by the Internet, or hosted by others, should be subject to a vulnerability assessment. You can't assume that an application is secure just because it is hosted by a third-party, just as you can't assume that just there is no risk just because a web application, form, or entire site doesn't handle sensitive information. In both cases, any web security vulnerabilities could very likely lead an attacker directly to your most critical network segments and applications.

The Vulnerability Assessment

Now you're ready for the vulnerability assessment. Believe it or not, much of the hard work is already done: deciding the scope, and then classifying and prioritizing your applications. Now, assuming you've already acquired a web security scanner and have identified who will conduct the manual scan for business logic errors, you're ready to take a whack at your application.

The resulting report, based on the security health of the application, will provide you a list of high, medium, and low priority vulnerabilities. At this point, you'll need someone to vet the automated vulnerability assessment results to find any false positives, or vulnerabilities identified by the scanner, but don't actually exist. If it seems overwhelming, don't fret; we'll delve into how to prioritize and remedy these web security vulnerabilities in the next installment. About the same time as your automated vulnerability assessment, the manual assessment will be underway. During the manual assessment, the expert will look for logic errors in the application: Is it possible for users to conduct transactions in ways the developers hadn't anticipated? Such as the ability of someone to tamper with application values that are being passed from the client to the server to alter the price of an item. The manual vulnerability assessment will end with a list of all vulnerabilities to web security found, and the assessor should prioritize the risks posed by each problem - based on the ease of exploiting the vulnerability, and the potential harm that could result if an attacker is successful.

Now you have your list of web security vulnerabilities, both technical and logic. And, if your organization is like most others, you have some remedying work to do. The challenge now is to prioritize what needs to be fixed, so that your existing applications can be hardened, and those being built can be remedied and safely placed into production.

While the list of web security issues may be long, you've completed the first major phase on the road to a highly secure application. Take comfort in the fact that your vulnerability assessment has identified problems in your applications before they were attacked by competitors, lone-hackers, or organized crime. In the next article, **Effective Web Application Vulnerability Remediation Strategies**, we'll show you how to prioritize your remediation work so that development time isn't prolonged, and existing applications at risk are remedied before they can be attacked.