

WhiteHat Website Security Statistic Report

Spring 2009, 7th Edition

7th edition

There is a difference between what is possible and what is probable, something we often lose sight of in the world of information security. For example, a vulnerability represents a possible way for an attacker to exploit an asset, but remember not all vulnerabilities are created equal. Obviously we must also keep in mind that just because a vulnerability exists does not necessarily mean it will be exploited, or indicate by whom or to what extent. Clearly, many vulnerabilities are very serious leaving the door open to compromise of sensitive information, financial loss, brand damage, violation of industry regulations, and downtime. Some vulnerabilities are more difficult to exploit than others and therefore attract different attackers. Autonomous worms & viruses may attack one type of issue, while a sentient targeted attacker may prefer another path. Better understanding of these factors enables us to make informed business decisions about website risk management and what is probable.

In relation to website vulnerabilities, this report represents what is possible. For those unfamiliar with the methods commonly employed to break into websites, they are not buffer overflows, format string issues, or unsigned integers. Those avenues are most often applied to commercial and open source software. Instead custom Web applications, those employed by e-commerce, financial and business sites the world over, are exploited via SQL Injection, Cross-Site Scripting (XSS), and various forms of business logic flaws – the very same issues represented in our Top Ten list (see below) and a leading cause of data loss according to Verizon's 2009 Data Breach Investigations Report (DBIR)¹. The Verizon report contains the data that helps correlate what is possible to what is probable. Tying back a particular event to a source and attack vector is a crucial step toward making better decisions down the road.

According to Verizon's DBIR, most incidents stemmed from external sources, linked to organized crime, and exploited Web-based flaws. This means our adversaries do not typically have access to the source code or binaries of the custom Web applications they exploit (not that they need it). This threat profile is also different from someone analyzing a piece of operation system software to uncover a 0-day who may test locally 24x7 without raising alarms. It also differs from scanning a network for unpatched issues open to a ready made exploit. Verizon goes further, organizing the apparent attackers into three types based upon how they chose their targets². They are "Random opportunistic," "Directed opportunistic," and "Fully targeted." To put these attacker types into a Web security context we found it necessary to describe their basic actions as we have experienced the same trends in our world.

The WhiteHat Website Security Statistics Report provides a one-of-a-kind perspective on the state of website security and the issues that organizations must address to avert attack. WhiteHat has been publishing the report, which highlights the top ten vulnerabilities, vertical market trends and new attack vectors, since 2006.

The WhiteHat report presents a statistical picture of current website vulnerabilities, accompanied by WhiteHat expert analysis and recommendations. WhiteHat's report is the only one in the industry to focus solely on unknown vulnerabilities in custom Web applications, code unique to an organization, within real-world websites.

WhiteHat issues continued installments of the Website Security Statistics Report on a quarterly basis. To ensure the report remains useful and relevant, WhiteHat incorporates feedback and ideas from leading industry thought leaders and influencers. Based on feedback already received, the latest report includes: comparing vulnerability prevalence by severity, top ten vulnerability classes sorted by percentage likelihood and an outline of the types of technology typically encountered during WhiteHat vulnerability assessments mapped with the associated vulnerability percentage breakdown.

Random opportunistic – victim randomly selected

Attacks are completely automated, noisy, unauthenticated, and exploit well-known unpatched issues and some customized Web application vulnerabilities. Targets are chosen indiscriminately through wide scans and tend to impact the most vulnerable. Typical motivation is to infect Web pages with malware or subtle defacement.

Example: With Mass SQL Injection automated worms insert malicious JavaScript IFRAMEs (pointing to malware servers) into back-end databases and used the capability to exploit unpatched Web browsers³.

Directed opportunistic – victim selected, but only because they were known to have a particular exploitable weakness

Attacker has professional or open source scanning tools. May register accounts, authenticate, and customize exploits for custom Web application flaws found easily by automation. Targets are those with valuable data that can be monetized, have a tarnishable brand, and are penetrable with a few days of effort.

Example: XSS vulnerabilities used to create very convincing Phishing scams that appear on the real website as opposed to a fake. JavaScript malware steals victims session cookies and passwords⁴.

Fully targeted – victim was chosen and then attack planned

Highly motivated attacker with professional, open source, and purpose-built scanning tools. May register accounts, authenticate, and customize exploits for custom Web application vulnerabilities, and capitalize on business logic flaws. Victims may be defrauded, extorted, and targeted for up to a year or more.

Example: 'The Analyzer' allegedly hacked into multiple financial institutions using SQL Injection to steal credit and debit card numbers that were then used by thieves in several countries to withdraw more than \$1 million from ATMs⁵.

By aligning an attacker's capabilities against a particular security control, it becomes much easier to predict and/or justify that control's effectiveness. We have also observed that the majority of attacks are opportunistic rather than fully targeted. The well-worn adage about outrunning the bear can be applied. "When being chased by a bear you don't have to be the fastest deer, you just have to be faster than the one next to you." Given the large number of websites WhiteHat Security currently assesses, we have been able to provide this intelligence to our customers, showing them how the security of their websites ranks globally or within their particular industry vertical (Figure 1). The more decision making information we possess the more effective the website risk management strategy.

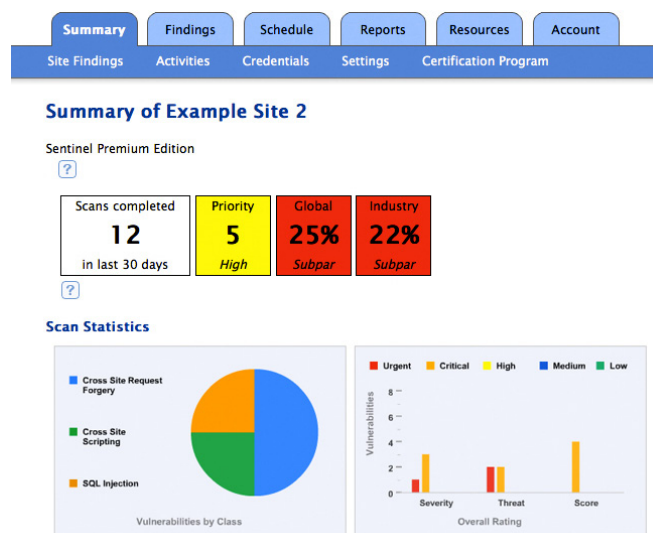


Figure 1. WhiteHat Sentinel summary screen displaying global ranking

Web security is a moving target and enterprises need timely information about the latest attack trends, how they can best defend their websites, and visibility into their vulnerability life-cycle. Through its Software-as-a-Service (SaaS) offering, WhiteHat Sentinel⁶, WhiteHat Security is uniquely positioned to deliver the knowledge and solutions that organizations need to protect their brands, attain PCI compliance and avert costly breaches.

WhiteHat customers use Sentinel, an annual subscription service, to assess and manage their website vulnerability exposure. Each week, WhiteHat Sentinel assesses thousands of public-facing and pre-production websites for vulnerabilities using a consistent and repeatable three-phase process:

- *Proprietary scanning technology identifies technical vulnerabilities such as Cross-Site Scripting, SQL Injection, some forms of Cross-Site Request Forgery and many others.*
- *Experts create customized tests for each website to uncover business logic flaws including Insufficient Authorization, Insufficient Authentication, Abuse of Functionality, etc.*
- *Results are verified to remove false-positives and assign an appropriate level of severity in order for data to be accurate and actionable.*

It is important to note that the websites WhiteHat Sentinel manages likely represent the most “important” and “secure” websites on the Web, owned by enterprises that are very serious about their security. With access to a vast sampling of vulnerabilities in custom Web applications we are able to publish the most prevalent issues on an aggregate basis.

WhiteHat Sentinel captured the data contained within this report by focusing solely on previously unknown vulnerabilities in custom Web applications – code unique to an organization, on real-world websites (Figure 2). WhiteHat Sentinel offers three different levels of service (Premium, Standard, and Baseline) to match the level of security assurance required by the organization.

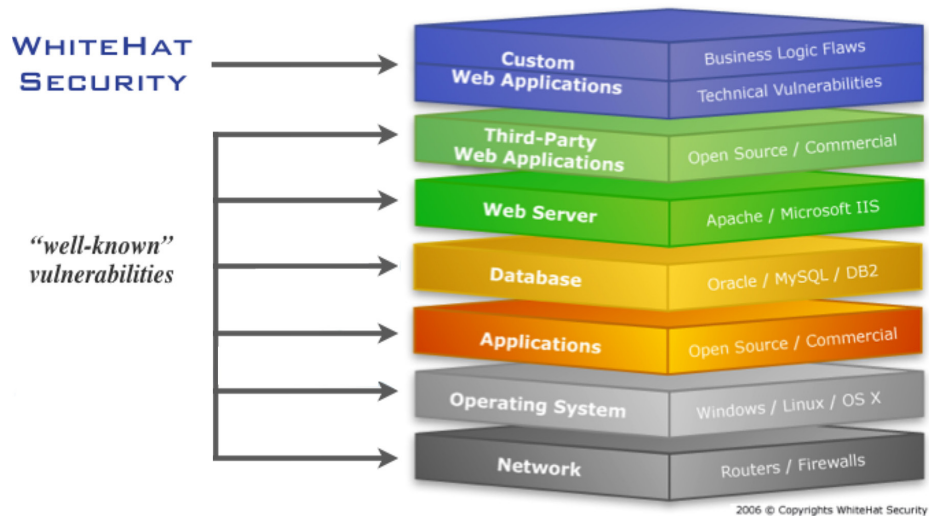


Figure 2. Software vulnerability stack

Data Overview

- 1,031 total websites
- 17,888 verified vulnerabilities
- Data collected from January 1, 2006 to March 31, 2009
- Vast majority of websites assessed for vulnerabilities weekly
- Vulnerabilities classified according to WASC Threat Classification⁷
- Vulnerability severity naming convention aligns with PCI-DSS

Q1 2009 Key Findings

- 82% of websites have had a **HIGH**, **CRITICAL**, or **URGENT** issue
- 63% of websites currently have a **HIGH**, **CRITICAL**, or **URGENT** issue
- 60% vulnerability resolution rate among sample with 7,157 (out of 17,888 historical vulnerabilities) unresolved issues remaining as of 3/31/09
- Vulnerability time-to-fix metrics are not changing substantively, typically requiring weeks to months to achieve resolution.
- Average # of **HIGH**, **CRITICAL**, or **URGENT** severity vulnerabilities per website during the vulnerability assessment lifetime: 17
- Average number of serious unresolved vulnerabilities per website: 7
- Average number of inputs (attack surface) per website: 227
- Average ratio of vulnerability count / number of inputs: 2.58%

When interpreting the results there are several factors that should be considered:

- The mix of websites ranges from highly complex and interactive sites with a large attack surface to some static brochureware sites.
- Vulnerabilities are organized and counted based upon a unique Web application and class of attack. For example, if there are five possible parameters in a single Web application (*/foo/webapp.cgi*), three of which are vulnerable to SQL Injection, it is counted as one vulnerability (not three).
- Vulnerabilities do not include "best practice" findings. For example, if a website mixes SSL content with non-SSL on the same Web page, while this may be considered a business policy violation, it must be taken on a case-by-case basis. As an example, the lack of encrypted passwords or data storage on the system are not considered vulnerabilities for the purpose of this report. Only issues that can be directly exploited remotely are included.
- Vulnerability assessment processes are incremental and ongoing, the frequency of which is customer-driven and as such should not automatically be considered "complete." However, the vast majority of WhiteHat Sentinel customers do assess their sites on a weekly basis. When interpreting the data it is best to keep in mind new attack vectors are always being researched by attackers, making it best to view the data as a best-case scenario based on the most up-to-date information available.
- Websites may be covered by different WhiteHat Sentinel service levels (Premium (PE), Standard (SE), Baseline (BE)) offering varying degrees of testing comprehensiveness. PE covers all technical vulnerabilities and business logic flaws identified by the WASC 24 (and some beyond). SE focuses primarily on the technical vulnerabilities. BE bundles critical technical security checks into a safe, fully-automated service. All WhiteHat Sentinel services include verification of all vulnerabilities.

Vulnerability Prevalence by Severity

In order for organizations to take appropriate action, each website vulnerability must be independently evaluated for business criticality. For example, not all Cross-Site Scripting or SQL Injection vulnerabilities are equal, making it necessary to consider its true “severity” for an individual organization. Using the Payment Card Industry Data Security Standard⁸ (PCI-DSS) severity system (Urgent, Critical, High, Medium, Low) as a baseline, WhiteHat Security rates vulnerability severity by the potential business impact if the issue were to be exploited and does not rely solely on default settings. It should also be noted that according to the PCI-DSS, any websites with **Urgent**, **Critical**, or **High** severity issues cannot be considered compliant.

While historical and current vulnerability count averages are holding at 17 and 7 respectively, unfortunately the likelihood of websites having at least one issue of a specific severity has also remained constant. It can be reasonably argued that having several **Urgent**, **Critical**, or **High** severity issues makes it easier for an attacker to achieve a successful data compromise, but finding and exploiting a single issue is all that is required. Organizations are finding it very challenging to remediate 100% of their flaws of a specific type or severity.

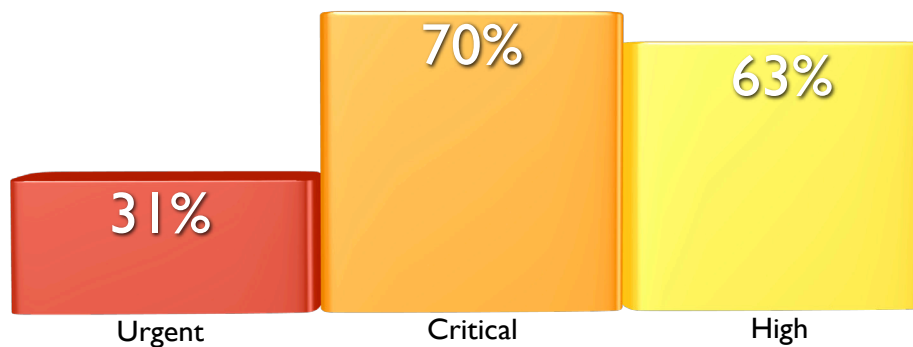


Figure 3. Percentage likelihood of websites having at least one vulnerability (sorted by severity)

The Top Ten

WhiteHat Security determines the most prevalent issues by calculating the percentage likelihood of a particular vulnerability class occurring within websites (Figure 4). This approach minimizes data skewing in website edge cases that are either highly secure or extremely risk-prone.

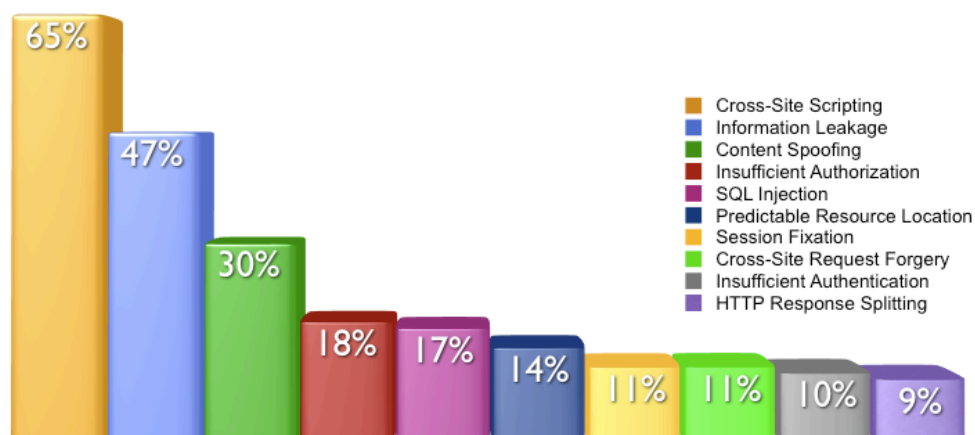


Figure 4. Top 10 vulnerability classes (sorted by percentage likelihood)

To supplement vulnerability likelihood statistics, the following graph (Figure 5) illustrates prevalence by class in the overall vulnerability population. Notice how greatly it differs from the Top Ten graph. The reason is that one website may possess hundreds of unique issues of a specific class, such as Cross-Site Scripting, Information Leakage, or Content Spoofing, while another website may not contain any.

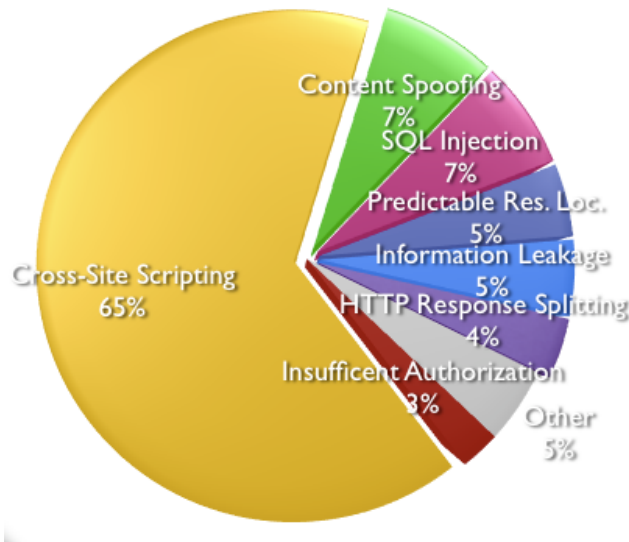


Figure 5. Vulnerability classes (sorted by class population)

Development Technology and Vulnerabilities

Table 1 provides insight into the types of technologies encountered during WhiteHat Sentinel vulnerability assessments and the associated vulnerability percentage breakdown. The statistics are not meant to establish which technology is more secure. For example, the under-representation of PHP likely means that this technology is not being utilized by those in the sample set relative to others. The large set of “unknown” are those without file extension. In future reports, we plan to offer likelihood of vulnerability numbers related to specific technologies.

URL Extension	% of websites	% of vulnerabilities
unknown	59%	40%
asp	24%	25%
aspx	23%	9%
xml	10%	2%
jsp	9%	8%
do	7%	3%
php	6%	3%
html	4%	2%
old	4%	1%
dll	4%	1%
cfm	3%	4%

Table 1.

Time-to-Fix

When website vulnerabilities are identified there is a certain amount of time required for the issue to be resolved. Resolution could take the form of a software update, configuration change, Web application firewall rule, etc. Ideally the time to fix should be as short as possible because an open vulnerability represents an opportunity for hackers to exploit the website, but no remedy is instantaneous. To perform this analysis we focused on vulnerabilities identified and resolved within the last twelve months between March 31, 2008 and 2009. The data was then sorted by the most common **Urgent**, **Critical**, or **High** severity issues. There are several aspects worth noting that may bias the sample:

- *Should a vulnerability be resolved it could take up to 7 days before it is retested and confirmed closed by WhiteHat Sentinel, depending upon the customer's scan schedule. However, a customer can proactively use the auto-retest function to get real-time confirmation of a fix.*
- *Not all vulnerabilities identified within this period have been resolved, which means the time to fix measurements are likely to grow (See Table 2).*
- *Since June 2008, WhiteHat Sentinel has offered integration capabilities with the F5 Application Security Manager and the Breach ModSecurity Web Application Firewalls⁹ which allow safe, automated "virtual patching" of vulnerabilities without development resources. We expect other similar integrations to follow.*
- *Business logic flaws, including Insufficient Authentication and Authorization, tend to take longer to fix than technical vulnerabilities including such as Cross-Site Scripting and SQL Injection. Anecdotally we believe the reason is that syntax (implementation) bugs are largely oversights and can often be remedied through the existing development frameworks with little additional code. Semantic issues conversely require careful checking with authentication/ authorization layers or perhaps unfortunately a substantial rewrite of the application. What we do know is organizations are finding it difficult to fix custom Web application vulnerabilities quickly or at all.*

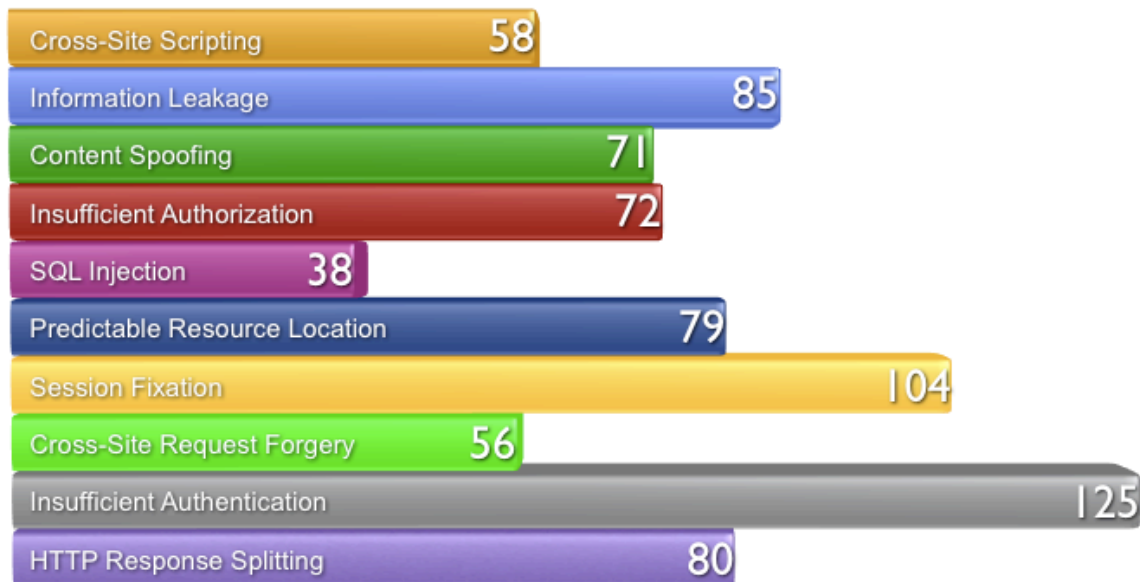


Figure 6. Average number of days for vulnerabilities to be resolved (sorted by class)

Once vulnerabilities are identified it does not necessarily mean they are fixed quickly, or ever. It is interesting to analyze the types and severity of the vulnerabilities that do get fixed (or not) and in what volumes. Some organizations target the easier issues first to demonstrate their progress by vulnerability reduction. Others prioritize the high severity issues to reduce overall risk. Still, resources and security interest are not infinite so some issues will remain unresolved for extended periods of time. The reasons for this are diverse, but may include:

- *No one at the organization understands or is responsible for maintaining the code.*
- *Feature enhancements are prioritized ahead of security fixes.*
- *Affected code is owned by an unresponsive third-party vendor.*
- *Website will be decommissioned or replaced "soon."*
- *Risk of exploitation is accepted.*
- *Solution conflicts with business use case.*
- *Compliance does not require it*
- *No one at the organization knows about, understands, or respects the issue.*
- *Lack of budget to fix the holes*

Class of Attack	% resolved	severity
Cross Site Scripting	20%	urgent
Insufficient Authorization	19%	urgent
SQL Injection	30%	urgent
HTTP Response Splitting	75%	urgent
Directory Traversal	53%	urgent
Insufficient Authentication	38%	critical
Cross-Site Scripting	39%	critical
Abuse of Functionality	28%	critical
Cross-Site Request Forgery	45%	critical
Session Fixation	21%	critical
Brute Force	11%	high
Content Spoofing	25%	high
HTTP Response Splitting	30%	high
Information Leakage	29%	high
Predictable Resource Location	26%	high

Table 2. Percentage of vulnerabilities resolved (sorted by class & severity)

Comparing Industry Verticals

Figure 7 shows the percentage of websites with at least one **Urgent**, **Critical**, or **High** severity issue sorted by industry vertical. The majority of websites have these types of issues, which also most likely precludes them from being classified as PCI-DSS compliant. Clearly no vertical is performing exceptionally well, but some are improving (highlighted in green) and achieving better results than others. It is difficult to show causation, but we have some correlation ideas. The first is compulsory battlefield testing, which are those types of sites where the bulk of the functionality is ahead of the login screen (ie. Retail). Meaning, the bad guys are able to scan these sites deeper and more often, which forces security improvement. Other factors at work could be the development technology in use and the application's date of deployment. In our opinion organizations, and by extension developers, do not code against attacks they are not aware of and do not respect.

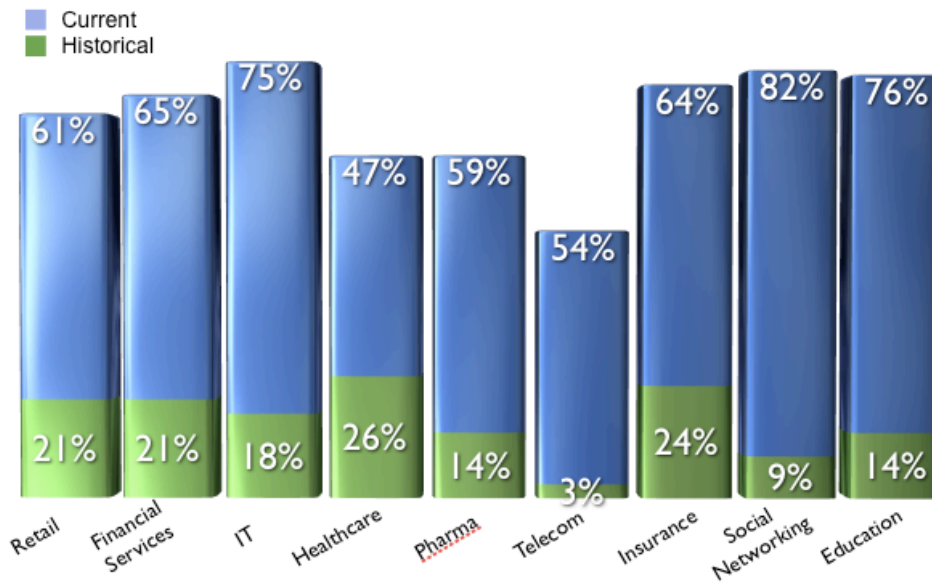


Figure 7. Percentage of websites with an **URGENT, CRITICAL** or **HIGH** severity vulnerability sorted by industry vertical

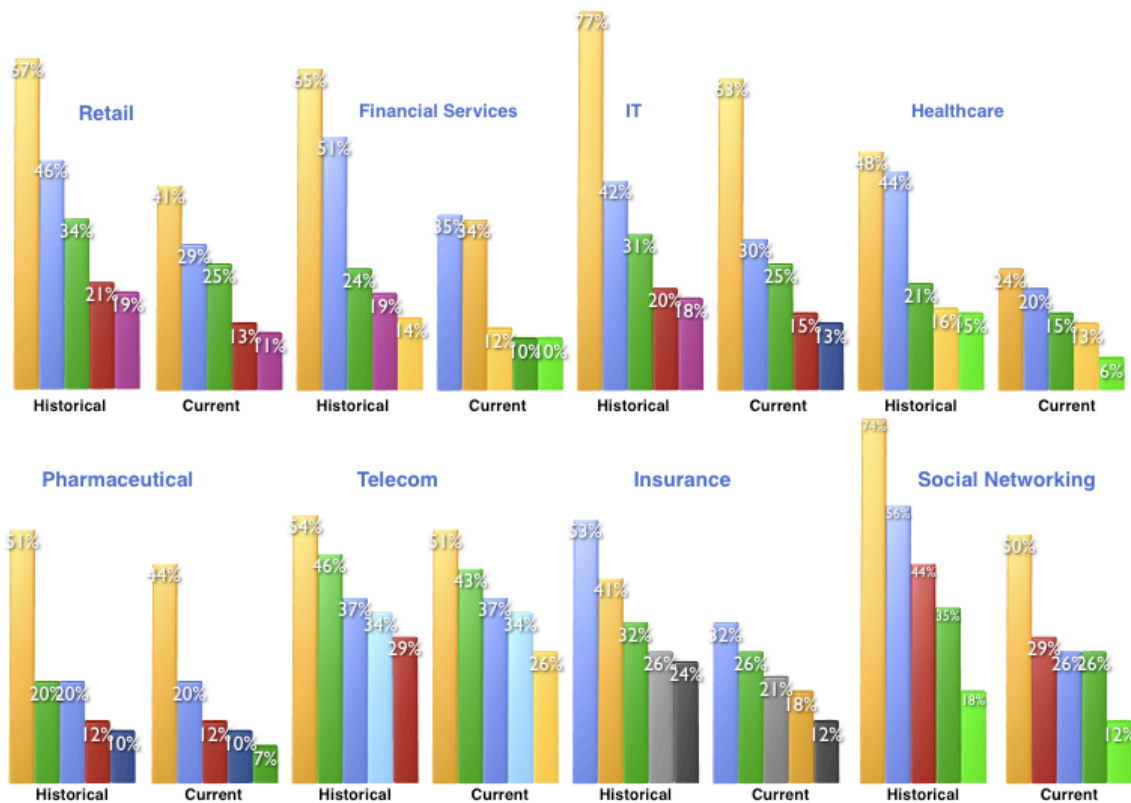


Figure 8. Top 5 percentage likelihood of a website having at least one **HIGH, CRITICAL, or URGENT** issue (sorted by industry vertical)

Conclusion

Based upon everything we've seen thus far, Web security requires a two-pronged strategy: one for websites that currently exist and another for those yet to be developed. Doing so allows organizations to maximize their resources in areas that impact the business most. The reality is there are an extraordinary number of websites to protect, most of which are vulnerable, and at the same time, the attackers' capabilities are outpacing our ability to defend those sites. Additionally, we are generating tremendous volumes of new production Web code with the hope that the next 200+ million websites¹⁰ will be much more secure than those which currently exist. To achieve this we as an industry, must continue to integrate security into the software development lifecycle and operationalize.

For existing websites, operationalizing begins with an organization identifying all their websites (Web assets) because you can't secure what you don't know you own. Next, we must rank these assets based upon their relative business criticality. Then determine which adversaries (Random Opportunistic, Directed Opportunistic, Fully Targeted) are most likely to attack and therefore need to be defended against. These data points influence the level of comprehensiveness and assessment frequency required to determine the current security posture. In aggregate, this information allows security controls (SDL, virtual patch, configuration change, decommission, outsource, version roll-back, etc.) to be implemented in a way that their effects can be measured. Not only does operationalizing provide more effective security, but it also delivers budget justification for management by demonstrating real risks and the most effective way to mitigate them. Operationalized security gives you visibility into the assets you must protect, the flexibility to match the solution to the risk and control against any type of attack, and an easy way to track progress.

Glossary: The Top Ten Defined

1. Cross-Site Scripting (65% of websites)

Cross-site Scripting¹¹ (XSS) is easily the most prevalent website vulnerability. XSS has proven to be extremely hazardous to businesses and consumers in the form of either Web Worms¹², "Phishing with Superbait¹³" scams, Javascript malware-laced defacements, and malicious Web Widgets. The evolution of JavaScript malware, finding its way into more and more attackers toolboxes, has made finding and fixing this vulnerability more vital than ever.

2. Information Leakage (46% of websites)

Information Leakage¹⁴ occurs when a website knowingly or unknowingly reveals sensitive information such as developer comments, user information, internal IP addresses, source code, software versions numbers, error messages/codes, etc., which may all aid in a targeted attack. While most of the time rated MEDIUM or LOW severity, several Information Leakage issues could be used in combination to compromise a website.

3. Content Spoofing (30% of websites)

Content Spoofing¹⁵ is often used in phishing scams (or intelligence gathering) as a method of forcing a legitimate website to deliver or redirect users to bogus content. For example, users often receive a suspicious link that instructs them to confirm their user name and password information. Typically, phishing websites are hosted on look-alike domain names mimicking the content of the real site. In the case of Content spoofing phishing scams fake content is injected into the real website, making it very difficult, if not impossible, for users to detect the difference and therefore protect themselves.

4. Insufficient Authorization (18% of websites)

Insufficient Authorization¹⁶ flaws are also typically found within the business logic of an application. Successful exploitation leads to an attacker being able to escalate his or her privileges, exercise unauthorized access, and potentially defraud the systems. For example, while logged-in as a normal user, an attacker could gain access to another user's data while still being logged-in under their current account.

5. SQL Injection (17% of websites)

SQL Injection¹⁷ has been at the center of some of the largest credit card, identity theft incidents, and mass scale website compromises. Today's backend website databases store highly sensitive information, making them a natural, attractive target for malicious hackers. Names, addresses, phone numbers, passwords, birth dates, intellectual property, trade secrets, encryption keys and often much more could be vulnerable to theft. With a few well-placed quotes, semi-colons and commands entered into a standard Web browser entire databases could fall into the wrong hands.

6. Predictable Resource Location¹⁸ (PRL) (14% of websites)

Over time, many pages on a website become unlinked, orphaned, and forgotten--especially on websites experiencing a high rate of content and/or code updates. These Web pages sometimes contain payment logs, software backups, post dated press releases, debug messages, source code – nothing or everything. Normally the only mechanism protecting the sensitive information within is the predictability of the URL. Automated scanners have become adept at uncovering these files by generating thousands of guesses.

7. Session Fixation (11% of websites)

Session Fixation is an attack technique that forces a user's session ID to an explicit value. Depending on the functionality of the target web site, a number of techniques can be utilized to “fix” the session ID value. Once the victim user authenticates in with the fixed session value, the attacker can then leverage it because of the knowledge of the value.

8. Cross-Site Request Forgery (11% of websites)

Cross-Site Request Forgery¹⁹ (aka Session Riding, Web Trojan, Confused Deputy, etc.) allow an attacker to force an unsuspecting user's browser to make a Web request they didn't intend. For example, the attacker could force a user to compromise their own banking, eCommerce or other website accounts invisibly without their knowledge. Since the forged request is coming from the legitimate user, even when they are logged-in, the website will accept it as being the intent of that user.

9. Insufficient Authentication (10% of websites)

Insufficient Authentication²⁰ flaws are typically found within the business logic of an application. Successful exploitation leads to an attacker gaining unauthorized access to protected sections of a website. For example, while logged-in as a normal user, an attacker could impersonate another user on the system. These types of issues are common in financial, healthcare systems, and general content management systems where there is a high concentration of complex business logic functionality.

10. HTTP Response Splitting (9% of websites)

HTTP Response Splitting is an attack technique in which a single request is sent to the website in such a way that the response may appear to look like two. Depending on the network architecture of the website or the behavior of a user's Web browser, the “second” HTTP response that's under the control of the attacker can be used to poison cache servers, deface Web pages, perform session fixation, etc.

References

- ¹ Verizon 2009 Data Breach Investigations Report – <http://securityblog.verizonbusiness.com/2009/04/15/2009-dbir/>
- ² Verizon 2009 DBIR: Attack targeting – <http://securityblog.verizonbusiness.com/2009/04/15/2009-dbir-attack-targeting/>
- ³ Over 1.5 million pages affected by the recent SQL injection attacks – <http://blogs.zdnet.com/security/?p=1150>
- ⁴ New Phishing Attacks Combine Wildcard DNS and XSS – http://news.netcraft.com/archives/2009/02/17/new_phishing_attacks_combine_wildcard_dns_and_xss.html
- ⁵ 'The Analyzer' Hack Probe Widens; \$10 Million Allegedly Stolen From U.S. Banks – <http://www.wired.com/threatlevel/2009/03/the-analyzer-ha/>
- ⁶ WhiteHat Sentinel – <http://www.whitehatsec.com/home/services/services.html>
- ⁷ Web Security Threat Classification – <http://www.webappsec.org/projects/threat/>
- ⁸ PCI Data Security Standard – <https://www.pcisecuritystandards.org/>
- ⁹ WhiteHat Sentinel Web Application Firewall (WAF) Integration – <http://www.whitehatsec.com/home/services/waf.html>
- ¹⁰ Netcraft April 2009 Web Server Survey – http://news.netcraft.com/archives/2009/04/06/april_2009_web_server_survey.html
- ¹¹ Cross-Site Scripting – http://www.webappsec.org/projects/threat/classes/cross-site_scripting.shtml
- ¹² Cross Site Scripting Worms and Viruses – <http://www.whitehatsec.com/home/assets/WP5CSS0607.pdf>
- ¹³ Phishing with Superbait – http://www.whitehatsec.com/home/assets/presentations/phishing_superbait.pdf
- ¹⁴ Information Leakage – http://www.webappsec.org/projects/threat/classes/information_leakage.shtml
- ¹⁵ Content Spoofing – http://www.webappsec.org/projects/threat/classes/content_spoofing.shtml
- ¹⁶ Insufficient Authorization – http://www.webappsec.org/projects/threat/classes/insufficient_authorization.shtml
- ¹⁷ SQL Injection – http://www.webappsec.org/projects/threat/classes/sql_injection.shtml
- ¹⁸ Predictable Resource Location – http://www.webappsec.org/projects/threat/classes/predictable_resource_location.shtml
- ¹⁹ Cross-Site Request Forgery – http://en.wikipedia.org/wiki/Cross-site_request_forgery
- ²⁰ Insufficient Authentication – http://www.webappsec.org/projects/threat/classes/insufficient_authentication.shtml
- ²¹ HTTP Response Splitting – http://www.webappsec.org/projects/threat/classes/http_response_splitting.shtml

The WhiteHat Sentinel Service – Website Risk Management

Find and Fix Vulnerabilities, Protect Your Website – The WhiteHat Sentinel Service delivers the most accurate and comprehensive website vulnerability coverage available. Worried about the OWASP Top Ten vulnerabilities or the WASC Threat Classification? Scanners alone cannot identify all the vulnerabilities defined by these standards. WhiteHat Sentinel can. Many of the most dangerous vulnerabilities reside in the business logic of an application and are only uncovered through expert human analysis.

Virtually Eliminate False Positives – No busy security team has time to deal with false positives. That's why the WhiteHat Sentinel Security Operations Team verifies the results of all scans. Customers see only real, actionable vulnerabilities, saving time and money.

Virtual Patching is Now a Reality – WhiteHat Sentinel can directly configure policies on a WAF to protect against vulnerability exploits (e.g., cross-site scripting, SQL injection) that were found during the scanning process. Normally, this will be a two step process: (1) identify vulnerabilities using WhiteHat Sentinel and (2) create highly-targeted policies on WAF. This makes the process simpler for the end user – find the problem, then fix the problem with the click of a button.

Dynamic Improvement and Refinement – WhiteHat Sentinel stays one step ahead of the latest website attack vectors with persistent updates and refinements to its service. Updates are dynamic – as often as one day to several weeks, versus up to six months or longer for traditional software tools. And, Sentinel uses its unique "Inspector" technology to apply identified vulnerabilities across every website it evaluates. Ultimately, each site benefits from the protection of others.

Total Control – WhiteHat Sentinel runs on the customer's schedule, not ours. Scans can be manually or automatically scheduled to run daily, weekly, and as often as websites change. Whenever required, WhiteHat Sentinel provides a comprehensive assessment, plus prioritization recommendations based on threat and severity levels, to better arm security professionals with the knowledge needed to secure them.

Unlimited Assessments, Anytime Websites Change – With WhiteHat Sentinel, customers pay a single annual fee, with unlimited assessments per year. And, the more applications under management with WhiteHat Sentinel, the lower the annual cost per application. High volume e-commerce sites may have weekly code changes, while others change monthly. WhiteHat Sentinel offers the flexibility to assess sites as frequent as necessary.

Simplified Management – There is no cumbersome software installation and configuration. Initial vulnerability assessments can often be up-and-running in a matter of hours. With WhiteHat Sentinel's Web interface, vulnerability data can be easily accessed, scans or print reports can be scheduled at any time from any location. No outlays for software, hardware or an engineer to run the scanner and interpret results. With the WhiteHat Sentinel Service, website vulnerability management is simplified and under control.

About WhiteHat Security, Inc.

Headquartered in Santa Clara, California, WhiteHat Security is the leading provider of website security solutions that protect critical data, ensure compliance and narrow the window of risk. WhiteHat Sentinel, the company's flagship product family, is the most accurate, complete and cost-effective website vulnerability management solution available. It delivers the flexibility, simplicity and manageability that organizations need to take control of website security and prevent Web attacks. Furthermore, WhiteHat Sentinel enables automated mitigation of website vulnerabilities via integration with Web application firewalls. To learn more about WhiteHat Security, please visit our website at www.whitehatsec.com.

