WhiteHat SECURITY

WEBSITE SECURITY STATISTICS REPORT

MAY 2013

INTRODUCTION

WhiteHat Security's Website Security Statistics Report provides a one-of-a-kind perspective on the state of website security and the issues that organizations must address in order to conduct business online safely.

Website security is an ever-moving target. New website launches are common, new code is released constantly, new Web technologies are created and adopted every day; as a result, new attack techniques are frequently disclosed that can put every online business at risk. In order to stay protected, enterprises must receive timely information about how they can most efficiently defend their websites, gain visibility into the performance of their security programs, and learn how they compare with their industry peers. Obtaining these insights is crucial in order to stay ahead and truly improve enterprise website security.

To help, WhiteHat Security has been publishing its Website Security Statistics Report since 2006. This report is the only one that focuses exclusively on unknown vulnerabilities in custom Web applications, code that is unique to an organization, and found in real-world websites. The underlying data is hundreds of terabytes in size, comprises vulnerability assessment results from tens of thousands of websites across hundreds of the most well-known organizations, and collectively represents the largest and most accurate picture of website security available. Inside this report is information about the most prevalent vulnerabilities, how many get fixed, how long the fixes can take on average, and how every application security program may measurably improve. The report is organized by industry, and is accompanied by WhiteHat Security's expert analysis and recommendations.

Through its Software-as-a-Service (SaaS) offering, WhiteHat Sentinel, WhiteHat Security is uniquely positioned to deliver the depth of knowledge that organizations require to protect their brands, attain compliance, and avert costly breaches.

ABOUT WHITEHAT SECURITY

Founded in 2001 and headquartered in Santa Clara, California, WhiteHat Security provides end-to-end solutions for Web security. The company's cloud website vulnerability management platform and leading security engineers turn verified security intelligence into actionable insights for customers. Through a combination of core products and strategic partnerships, WhiteHat Security provides complete Web security at a scale and accuracy unmatched in the industry. WhiteHat Sentinel, the company's flagship product line, currently manages more than 15,000 websites – including sites in the most regulated industries, such as top e-commerce, financial services and healthcare companies.

EXECUTIVE SUMMARY

Whether you read the Verizon Data Breach Incidents Report, the Trustwave Global Security Report, the Symantec Internet Security Threat Report, or essentially all other reports throughout the industry, the story is the same -- websites and web applications are one of, if not the leading target of cyber-attack. This has been the case for years. Website breaches lead directly to financial fraud, identity theft, regulatory fines, brand damage, lawsuits, downtime, malware propagation, and loss of customers. Given modern society's ever-increasing reliance on the web, the impact of a breach and the associated costs are going up, and fast. While an organization may ultimately survive a cyber-crime incident, the business disruption is often severe. It is far preferable to do something now to avert and minimize harm before disaster strikes.

Adding more "robust" firewalls to the network or allocating more budget to anti-virus software is not the answer. These controls provide nearly zero protection against today's web-based attacks. So while protecting the network and the host layers is still important, forward-thinking professionals are now seeing the bigger picture of computer security for what it really is: a software security problem.

What's needed is more secure software, NOT more security software.

Understanding this subtle distinction is key. Organizations must demand that software be designed in a way that makes it resilient against attack and does not require additional security products to protect it. The question that organizations should be asking themselves is: how do we integrate security throughout the software development life-cycle (SDLC)? How do we procure this type of software?

As simple as these questions sound, the answers have proven elusive. Most responses by the so-called experts are based purely on personal anecdote and devoid of any statistically compelling evidence, such as the data presented in this report. Many of these experts will cite various "best-practices," such as software security training for developers, security testing during QA, static code analysis, centralized controls, Web Application Firewalls, penetration-testing, and more; however, the term "best-practices" implies the activity is valuable in every organization at all times. The reality, though, is that just because a certain practice works well for one organization does not mean it will work at another. Unfortunately, this hasn't prevented many from boisterously and carelessly advocating a litany of best-practices with little regard for true efficacy and important operational considerations.

The net result: *websites no less hackable today than they were yesterday.*

Organizations need to better understand how various parts of the SDLC affect the introduction of vulnerabilities, which leave the door open to breaches. For example, we would like to

say, “organizations that provide software security training for their developers experience 25% fewer serious vulnerabilities annually than those who do not.” Or, “organizations that perform application security testing prior to each major production release not only have fewer vulnerabilities year-over-year, but exhibit a 35% faster time-to-fix.” We cannot make such statement today because the supporting data does not exist -- at least, not yet. If we had these insights, supported by empirical evidence, it would be nothing less than a game changer.

To move in this direction we asked WhiteHat Security customers to assist us by answering roughly a dozen very specific survey questions about their SDLC and application security program. Questions such as: how often do you perform security tests on your code during QA? What is your typical rate of production code change? Do you perform static code analysis? Have you deployed a Web Application Firewall? Who in your organization is accountable in the event of a breach? We even asked: has your website been breached?

We received responses to this survey from 76 organizations, and then correlated those responses with WhiteHat Sentinel website vulnerability data. The results were both stunning and deeply head scratching. The connections from various software security controls and SDLC behaviors to vulnerability outcomes and breaches is far more complicated than we ever imagined.

• 86% of all websites tested by WhiteHat Sentinel had at least one serious* vulnerability, and most of the time far more than one -- 56 to be precise.

• On average, 61% of these vulnerabilities were resolved, but doing so required an average of 193 days from first customer notification.

*(*Serious vulnerabilities are defined as those in which an attacker could take control over all, or some part, of the website, compromise user accounts on the system, access sensitive data, violate compliance requirements, and possibly make headline news. In short, serious vulnerabilities are those that should really be fixed.)*

As has been the case since our first report in 2006, the two most prevalent vulnerability classes remain Information Leakage and Cross-Site Scripting, identified in 55% and 53% of websites respectively. SQL Injection continued its downward slide from 11% in 2011 to 7% of websites in 2012, no longer making the Top 10. While there is much work left to be done, in many respects overall website security continues to show steady signs of improvement despite the stream of news headlines.

Next, this report will detail various software security controls (or "best-practices") our customers said were in place, and will directly correlate those responses with WhiteHat Sentinel vulnerability data:

• 57% of organizations said they provide some amount of instructor-led or computer-based software security training for their programmers. These organizations experienced 40% fewer vulnerabilities, resolved them 59% faster, but exhibited a 12% lower remediation rate.

• 53% of organizations said their software projects contain an application library or framework that centralizes and enforces security controls. These organizations experienced 64% more vulnerabilities, resolved them 27% slower, but demonstrated a 9% higher remediation rate.

• 39% of organizations said they perform some amount of Static Code Analysis on their website(s) underlying applications. These organizations experienced 15% more vulnerabilities, resolved them 26% slower, and had a 4% lower remediation rate.

• 55% of organizations said they have a Web Application Firewall (WAF) in some state of deployment. These organizations experienced 11% more vulnerabilities, resolved them 8% slower, and had a 7% lower remediation rate.

•23% of organizations website(s) said they experienced a data or system breach as a result of an application layer vulnerability. These organizations experienced 51% fewer vulnerabilities, resolved them 18% faster, and had a 4% higher remediation rate.

Much of the data above seems reasonable, even logical, while other bits seem completely counterintuitive. For instance, organizations that do perform Static Code Analysis or have a Web Application Firewall appear to have notably worse performance metrics than those who did neither.

One explanation may be that these metrics are precisely WHY these organizations [recently] acquired the technologies and they will eventually reap the benefits down the road (e.g. have fewer vulnerabilities). This remains to be seen. It could also be that they are misusing or overconfident in a particular solution, which only comes to light when WhiteHat Sentinel is engaged. What we know for sure is there are customers for whom these solutions absolutely make a measurable positive impact -- we see it in the data -- while others receive no discernible benefit. The average is somewhere in the middle. We believe that the reason for this difference is that there are in fact few, if any, truly universal application best-practices.

For example, if developers are not fixing reported vulnerabilities, or if they are taking a long time to do so, it could be because they don't understand the issues well enough. If this is the case, this is a good indication that providing training is a good idea. It could also easily be that a long time-to-fix or a lagging remediation rate is the result of management opting to task developers towards feature creation rather than vulnerability fixing. If so, additional work in business

prioritization and risk management is recommended, as is considering a Web Application Firewall with virtual patching capabilities. Either way this is why it is important for organizations to know their metrics and know what application security problems they really have. The alternative is blind adoption of "best-practices" that may only serve to disrupt the software development process for no tangible gain. Too often this is exactly what happens.

We were also curious about business drivers and the impact of compliance on website security. By a slim margin, organizations said their #1 driver for resolving vulnerabilities was compliance, narrowly ahead of risk reduction. At the same time, when we asked the same organizations to rank the reasons why their vulnerabilities go unresolved, compliance was cited as the #1 reason.

Proponents of compliance often suggest that mandatory regulatory controls be treated as a "security baseline," a platform to raise the floor, and not represent the ceiling. While this is a nice concept in casual conversation, this is not the enterprise reality we see. Keep in mind that WhiteHat Sentinel reports are often used to satisfy a plethora of auditors, but WhiteHat Security is not a PCI-DSS ASV nor a QSA vendor. When organizations are required to allocate funds toward compliance, which may or may not enhance security, there are often no resources left or tolerance by the business to do anything more effective.

Finally, we also wanted to know what part(s) of the organization are held accountable in the event of a website(s) data or system breach: we found that 79% said the Security Department would be accountable. Additionally, 74% said Executive Management, 66% Software Development, and 22% Board of Directors. By analyzing the data in this report, we see evidence of a direct correlation between increased accountability and decreased breaches, and of the efficacy of "best-practices" and security controls.

For example, if developers are required by compliance to attend security training, they'll view it as a checkbox activity and not internalize much of anything they've learned in training. However, if the organization places accountability on developers should a breach occur, all of a sudden training effectiveness increases, because now there is an obvious incentive to learn. When you empower those who are also accountable, whatever best-practices are then put into place have a higher likelihood of being effective. For security to really improve, some part of the organization must be held accountable. This is our working theory, the underlying narrative of our report.

Here is the biggest lesson and the common theme we're seeing: software security has not yet percolated into the collective consciousness as something organizations actually need to do something about proactively. While much lip service may be paid, we must address the issue that application security professionals are essentially selling preventative medicine, while much of the buying population still behaves with a wait-for-a-visit-to-the-emergency-room attitude before kicking into gear. This is a dangerous policy and in stark contrast to their pious rhetoric, which attempts to obfuscate that reality.

KEY FINDINGS

From a career perspective buying another firewall or AV product is safe, while investing in application security is not – unless, of course, the buyer is the person who will be accountable for a breach. On the bright side there is a unique opportunity for a new generation of corporate leaders to emerge, security executives who are proficient in software security and who aspire to distinguish themselves. They seek to address the real business challenges and base their decisions on data. The Web security world is moving in this direction.

High-Level

• 86% of all websites had at least one serious* vulnerability.

• The average number of serious* vulnerabilities identified per website was 56, continuing the downward trend from 79 in 2011 and 230 in 2010.

• Serious* vulnerabilities were resolved in an average of 193 days from first notification.

• 61% of all serious* vulnerabilities were resolved, slightly less than the 63% during from 2011, but still up from 53% in 2010 and far better than 2007 when it was just 35%.

Vulnerability Classes

• After reclaiming the top spot in 2011, Cross-Site Scripting lost its title as the most prevalent vulnerability to Content Spoofing, identified in 55% and 53% of websites respectively.

• SQL Injection fell out of the Top Ten most prevalent vulnerabilities, currently residing in 14th at 7% of websites, down from 11% in 2011.

• Of the total population of vulnerabilities identified, Cross-Site Scripting, Information Leakage, and Content Spoofing took the top three spots at 43%, 11%, and 13% respectively. This is a near repeat of 2011 where the percentages were 50%, 14%, and 9%.

• 57% of SQL Injection vulnerabilities identified were resolved, up slightly from 55% in 2011, and to do so required an average of 196 days.

• 50% of Cross-Site Scripting vulnerabilities were resolved, up from 48% in 2011, and to do so required an average of 227 days.

Industry

• Retail websites improved slightly, yet remain the industry possessing the second most security issues with an average of 106 serious* vulnerabilities identified per website, which narrowly beat IT at 114.

• Every single Manufacturing, Education, Energy, Government, and Food & Beverage website had at least one serious* vulnerability identified.

• 33% of all websites had at least one serious* vulnerability exposed every single day of 2012. Conversely, 18% of websites were vulnerable less than 30 days of the year.

• The industries that fixed their serious* vulnerabilities the fastest on average were Entertainment & Media (33 days), Government (48 days), and Gaming (67 days).

• The industries that fixed their serious* vulnerabilities the slowest on average were Education (342 days), Healthcare (276 days), and Insurance (274 days) websites.

• The industries that remediated the largest percentage of their serious* vulnerabilities on average were Entertainment & Media (81%), Telecommunications (74%), and Energy (71%) websites.

• The industries that remediated the fewest percentage of their serious* vulnerabilities on average were Non-Profit (41%), followed by Social Networking, Gaming, and Food & Beverage, all at 46%.

Survey: Application Security in the SDLC (Basic)

• 57% of organizations said they provide some amount of instructor led or computer-based software security training for their programmers.

• 53% of organizations said their software projects contain an application library or framework that centralizes and enforces security controls.

• 85% of organizations said they perform some amount of application security testing in pre-production website environments (i.e. staging, QA or other pre-prod systems).

• 39% of organization said they perform some amount of Static Code Analysis on their website(s) underlying applications.

• 55% of organizations said they have a Web Application Firewall (WAF) in some state of deployment

• Organizations said their #1 driver for resolving vulnerabilities was “Compliance,” narrowly ahead of “Risk Reduction.” When the same organizations were asked why vulnerabilities go unresolved, “Compliance” was also cited as the #1 reason.

• In the event an organization experiences a website(s) data or system breach, 79% said the Security Department would be accountable. Additionally, 74% said Executive Management, 66% Software Development, and 22% Board of Directors.

Survey: Application Security In The SDLC (Sentinel Correlation)

• Organizations that provided instructor led or computer-based software security training for their programmers had 40% fewer vulnerabilities, resolved them 59% faster, but exhibited a 12% lower remediation rate.

• Organizations with software projects containing an application library or framework that centralizes and enforces security controls had 64% more vulnerabilities, resolved them 27% slower, but demonstrated a 9% higher remediation rate.

• Organizations that performed Static Code Analysis on their website(s) underlying applications had 15% more vulnerabilities, resolved them 26% slower, and had a 4% lower remediation rate.

• Organizations with a Web Application Firewall deployment had 11% more vulnerabilities, resolved them 8% slower, and had a 7% lower remediation rate.

• Organizations whose website(s) experienced a data or system breach as a result of an application layer vulnerability had 51% fewer vulnerabilities, resolved them 18% faster, and had a 4% higher remediation rate.

AT A GLANCE: THE CURRENT STATE OF WEBSITE SECURITY

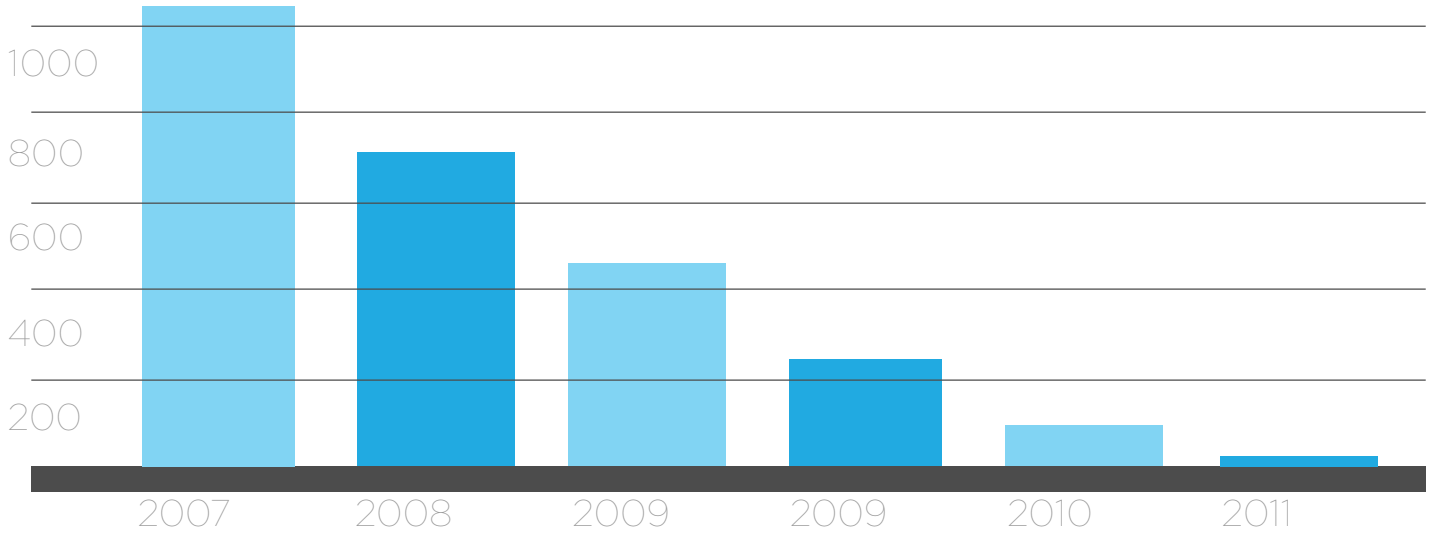


Figure 1. Vulnerability Historical Trend
The annual average number of serious* vulnerabilities discovered per website per year

Over the last year we saw continued reduction in the average number of serious* vulnerabilities found per website per year: from 230 identified in 2010 to 79 in 2011 to 56 in 2012. While this vulnerability reduction trend is welcome news, there are several possible explanations that must be taken into consideration, as the "real" numbers may not be as pretty. This is why we always remind readers that this report illustrates a best-case scenario: websites are, at a minimum, THIS vulnerable. The same is true for any industry report of this kind. That said, this reduction in vulnerabilities is likely a combination of several factors:

1) Despite the constant flood of website-related breach headlines, which would lead us to believe otherwise, websites generally may in fact be getting more "secure" — that is to say, markedly less vulnerable. All the media coverage raises awareness and puts pressure on corporate executives to do something about the web security problem. This awareness leads to increased investment, and increased investment eventually leads to vulnerability reduction (provided the investment is applied in the right way, of course).

2) While we are sure awareness makes a positive impact – at least within the WhiteHat Sentinel customer-base – there are still many, many websites with hundreds of serious* vulnerabilities. We also know website security is greatly influenced by compliance obligations and customer security requirements. As we found in this data, when WhiteHat customers require independent attestation of security readiness before business relationships move forward, things tend to improve. WhiteHat Sentinel is a trusted vehicle used for this website security proof point.

3) Another possible explanation is that organizations are choosing a less comprehensive form of vulnerability assessment, such as WhiteHat Sentinel Standard or Baseline over Premium Edition, which includes testing for business logic flaws. Some websites only need to be resilient against automated attacks rather than sentient adversaries because of their value to the business and/or their level of application complexity. Obviously when we look for fewer vulnerabilities, as defined by the assessment methodology in that service line, we will find fewer vulnerabilities. This does affect our report's statistics to some degree, particularly the vulnerability totals.

4) Another factor that may be contributing to the overall vulnerability reduction is that our sampling of websites, especially early on, was not representative of the Web at large. As of April 2013, Netcraft says there are roughly 649 million websites and that number fluctuates in the millions per month. While we are assessing far more websites, far more often than any other vendor, it is still a tiny fraction of the whole. It could also be that historically, WhiteHat customers only provided us with their most insecure websites first. While there may be some truth to this, we have seen reports released by our peers and their numbers are not far off from our own.

5) The move to modern development frameworks, such as the migration from the legacy Microsoft ASP Classic to .Net, which provides automatic security value to software developers. .Net, and frameworks with similar designs, are more forgiving for developers by making it easier

for them to produce code that's resilient to issues such as Cross-Site Scripting and SQL Injection without requiring them to know much about the technical details.

While overall vulnerability reduction is no doubt positive, the sheer number of serious* vulnerabilities in the wild is quite stunning. Consider for a moment that there are 1.8 million websites that have SSL certificates, an indication that they transact data worthy of keeping private. At 56 vulnerabilities per website, we can estimate that there are over 100 million serious vulnerabilities undiscovered on the Web.

To put these numbers in context, understanding how we count is essential. If a particular URL has 5 total parameters, 3 of which are vulnerable to SQL Injection, we count that as 3 vulnerabilities – not 5 or 1. Next consider all the discrete Web applications that make up a website's attack surface, each of which may have several input parameters, and the many ways each parameter may be exploited by dozens of vulnerability classes; then multiply that over a year with constant application code updates. Within that frame of reference, the volume of vulnerabilities may not appear so unreasonable.

Vulnerability counts alone do not provide a clear picture of the current state of website security. For this, it is necessary to consider the average number of days it takes to fix a serious* vulnerability (Time-to-Fix), the percentage of reported vulnerabilities that are no longer exploitable (Remediation Rate), and the average number of days a website is exposed to at least one serious* vulnerability (Window-of-Exposure). As these metrics are tallied at each organization, specific operational and software development lifecycle deficiencies can be isolated and improved.

Industry	Avg Vuln Sites	Annual Avg Vulns	Remediation Rate	Avg. Time-to-Fix (Days)
All	86%	56	61%	193
Entertainment & Media	91%	12	81%	33
Financial Services	81%	50	67%	226
Retail	91%	106	54%	224
Technology	85%	18	61%	71
IT	85%	114	54%	185
Healthcare	90%	22	53%	276
Banking	81%	11	54%	107
Manufacturing	100%	27	55%	197
Social Networking	86%	20	46%	175
Telecommunications	89%	20	74%	163
Education	100%	47	58%	342
Energy	100%	59	71%	144
Insurance	78%	39	55%	274
Government	100%	8	65%	48
Non Profit	95%	28	41%	236
Food & Beverage	100%	18	46%	36
Gaming	92%	17	46%	67

Figure 2. At a Glance: The Current State of Website Security (2012) (Sorted by industry)

Beyond just vulnerability volume, the high percentage of websites being vulnerable to at least one serious* vulnerability during 2012 is alarming. Alarming not only because the figures are 80% or greater – in some industries even at 100% – but that these numbers have been largely unchanged over the years. This suggests that building web applications is inherently a task that lends itself to vulnerabilities. We are not inclined to simply blame developers – that's too easy, and unfair – as applications increasingly become more complex and attack surface of applications grows with each newly added feature.

Once website vulnerabilities are identified, verified, and reported to customers by WhiteHat Sentinel, a certain amount of time transpires before the issue is resolved and confirmed as such. As no remedy can be instantaneous, it is important to measure the amount of time,required to resolve certain vulnerabilities (Time-to-Fix). Resolution could take the form of a software update, configuration change, Web application firewall rule, etc. Open vulnerabilities represent a window of opportunity for malicious hackers to exploit the website.

The approximately 6 months (193 days) to fix a serious* vulnerability is not only unacceptable; it proves that we are recording and socializing – both upwards and downwards – the wrong metrics to effect change. Fortunately some industries are doing quite well, but overall, a tracked statistic flattening points to the disappearance of a change agent. If the goal is to attain an acceptable level then flattening may be a positive sign. Absent the evidence of a change agent disappearing, this statistic may require correlation with another data point to be instructive. Perhaps that's the declining number of serious vulnerabilities.

The most common question we receive regarding the Average Time-to-Fix (Days) and Average Remediation Rate statistics is: why does it take so long for organizations to remediate vulnerabilities they've been informed of? To the uninitiated, the choice would seem to be a no-brainer. The fact is, again, these vulnerabilities are not resolved with a simple vendor supplied patch. This is almost always customer code we're dealing with and, as such, it requires the creation of a custom patch. So while many contributing factors may play into the eventual decision to fix now or to fix later, the fundamental business choice is almost always the same: security is a trade-off.

As everyone is well aware, software development resources are not infinite. Therefore, a development manager faces a choice: sacrifice a revenue-generating feature that, if it ships late, will for certain cost the organization money? Or resolve a vulnerability that might be exploited and might cost the company money? The challenge is this decision must be made every day with incomplete information. There is no guarantee that any single vulnerability will get exploited today, tomorrow, or ever – and if so, what the costs will be to the organization.

In this context it should be no surprise to anyone that often features are prioritized ahead of vulnerability fixes. Of course there are several other scenarios to explain why vulnerabilities go under resolved:

Factors inhibiting organizations from remediating vulnerabilities:

- No one at the organization understands, or is responsible for, maintaining the code.
- No one at the organization knows about, understands, or respects the vulnerability.
- Feature enhancements are prioritized ahead of security fixes.
- Lack of budget to fix the issues.
- Affected code is owned by an unresponsive third-party vendor.
- Website will be decommissioned or replaced "soon." To note, we have experienced deprecated websites under the Sentinel Service still in active use for over two years.
- Risk of exploitation is accepted.
- Solution conflicts with business use case.
- Compliance does not require fixing the issue.

Later in the "Survey: Application Security In The SDLC" section of this report, we asked customers to rank the prevalence of these issue. The results were highly illuminating.

This is a good opportunity to point out that while an individual programmer can be in control of what net-new vulnerabilities they produce with each release, they often don't have much control over remediation rates. The bugs they fix – and when they are fixed – are largely dictated by development managers.

To ease this difficult vulnerability remediation decision, as an industry, we must invest resources that improve true risk-decision making capabilities. With better data we can help development managers decide which vulnerabilities should be fixed, how, and when. They can more accurately decide which issues can wait till later and be placed under the watchful eye of the operational security team.

Additionally, more strategies and options would be welcome that reduce the cost to fix code, such as centralized security controls, or temporarily mitigate issues that will land in production systems no matter what is done in the SDLC. A virtual patch using a Web Application Firewall is a good example of this. Of course we must do whatever we reasonably can to decrease the number of vulnerabilities so that these tough choices are faced far less often than they are today.

Websites are an ongoing business concern and security must be ensured all the time, not just at a point in time. That's also why it must never be forgotten that an attacker only needs to exploit a single vulnerability, on any given day, to win. That's why the true Key Performance Indicator (KPI) of website security is Window-of-Exposure.

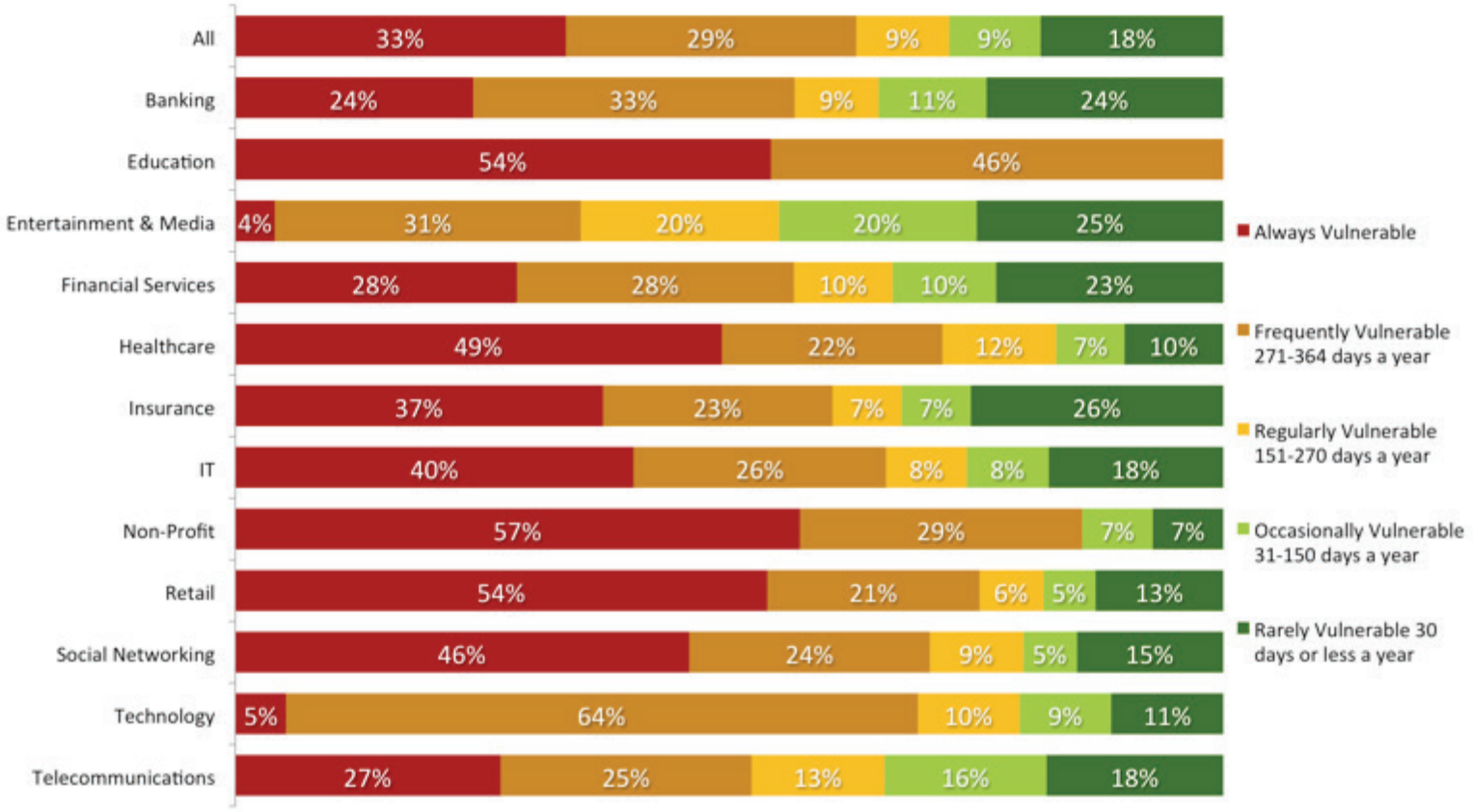


Figure 3. Overall Window of Exposure to Serious* Vulnerabilities (2012)
The percentage of websites that fall within a particular Window of Exposure zone (Sorted by industry)

Window-of-Exposure is the number of days in a year a website is exposed to at least one serious* vulnerability. As such, Window-of-Exposure is an informative combination of the total number of vulnerabilities, time-to-fix, and remediation rates – taken over time. Any one of these metrics, or a combination thereof, may be the area that has the greatest impact on a given organization's Window-of-Exposure outcome. To provide context, let's consider two identical websites, SiteA and SiteB.

1) SiteA had 10 serious* vulnerabilities identified during the last year and for 365 of those days it had at least one of those issues publicly exposed.

2) SiteB had 100 serious* vulnerabilities identified during the last year and for 10 of those days it had at least one of those issues publicly exposed.

Despite having 10 times the number of vulnerabilities, we would argue that during the last year SiteB had a substantially better security posture than SiteA as measured by the Window-of-Exposure. It is revealing to see how various industries perform in the area of Window-of-Exposure. Figure 3 illustrates 2012 performance by industry.

It is difficult to conclusively state why certain industries perform better than others, or even why a particular development group in a single organization seemingly performs better than the rest. All told, the data shows that the industry in which a particular website falls seems to, at best, only slightly correlate to an expected security posture. Previous reports have also explored potential correlations that may exist between organization size and development framework/programming language in use, but only small performance variations emerge. Clearly, some organizations have something figured out, as their websites are far less vulnerable than others.

MOST COMMON VULNERABILITIES

Now that we have an understanding of the average total number of serious* vulnerabilities, Time-to-Fix, Remediation Rates, and Window of Exposure across industry verticals, we'll look at the distribution of vulnerability classes. In Figure 4, the most prevalent vulnerabilities classes are calculated based upon their percentage likelihood of at least one instance being found within any given website. This approach minimizes data skewing in websites that are either highly secure or extremely risk-prone.

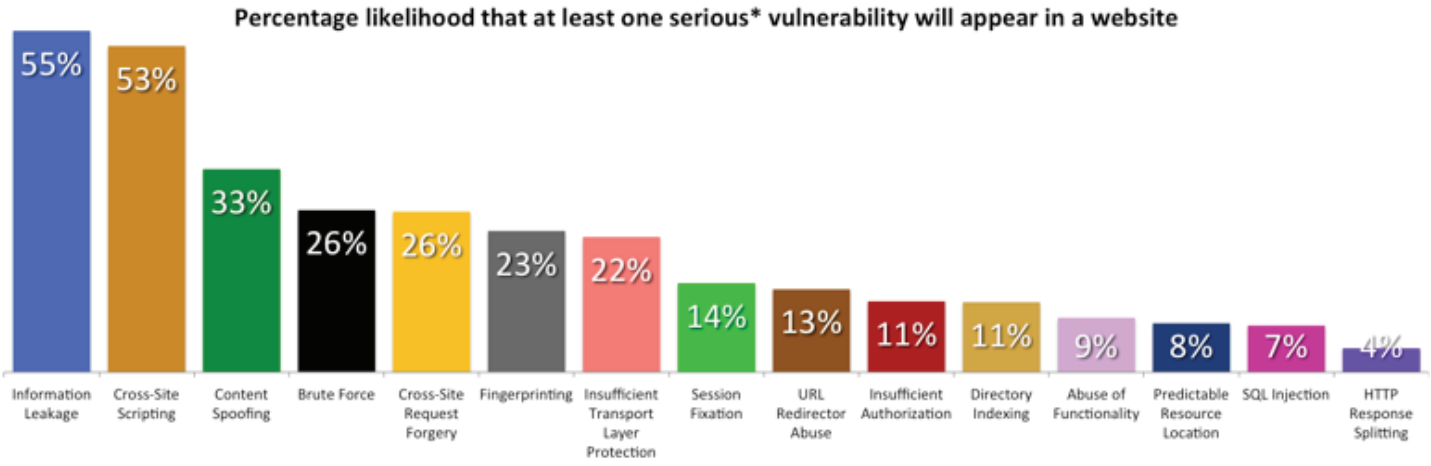


Figure 4. Top 15 Vulnerability Classes (2012)
(Sorted by vulnerability class)

Before getting into explaining specifics, it is important to state that these vulnerabilities represent possible ways attackers can penetrate websites and/or cause harm to their users. Alone, these vulnerabilities do not in any way represent the probability, or likelihood, that they will actually be exploited. To achieve that level of clarity it is necessary to compare our data against reports such Verizon's Data Breach Investigation Report, Trustwave Global Security Report, and others that describe actual incident data. This type of analysis goes beyond our current scope. For now, it's fair to say these vulnerabilities only represent ways that incidents may take place in the future.

(1 & 2) In 2012, Information Leakage overtook Cross- Site Scripting (XSS) once again and regained its crown as the most prevalent website vulnerability, identified in 55% and 53% of websites respectively. Last year, coincidentally, those percentages were exactly opposite. While in recent years both classes of attack were on the steady decline, now substantive improvement seems to have stalled. A potential reason for this is the sheer number of them in the average website, the ease with which coding mistakes can lead to vulnerabilities, and new ways in which they are discovered / exploited.

Note: For those unfamiliar, Information Leakage is largely a catch-all term used to describe a vulnerability where a website reveals sensitive data, such as technical details of the Web application, environment, or user-specific data: sensitive data that's completely unnecessary for a typical visitor, but may be used by an attacker to exploit the system, its hosting network, or users. Common examples are a failure to scrub out HTML/ JavaScript comments containing sensitive information (database passwords), improper application or server configurations, or differences in page responses for valid versus invalid data.

3) Also on a downward trend, yet remaining at #3, is Content Spoofing at 33% of websites. This is a slight drop over 2011 where it was at 36% of websites and 43% in 2010. Content Spoofing is a very similar vulnerability to Cross-Site Scripting, only without the "script" (HTML, JavaScript). This vulnerability class is most often used to force a website to display content unauthorized to another user. As such, Content Spoofing is useful in performing phishing scams and other malicious brand attacks.

4) Brute Force moved up to fourth from sixth place in the last year and increased 10 percentage points over last year to 26% of websites. A large number of these Brute Force vulnerabilities occur because a website login field reveals which entry of the username / password combination is incorrect. Due to spammers mining for valid email addresses, which double as usernames on a variety of websites, enterprises have an increased awareness and appreciation for the problem. In these cases we adjust the severity of Brute Force vulnerability accordingly.

5) Cross-Site Request Forgery (CSRF) maintained its placement at #5 during 2012, while increasing seven percentages points since the year prior, and now resides in 26% of websites. CSRF is widely considered to be the sleeping giant of Web security with the industry consensus asserting that nearly every website has at least one vulnerability, particularly those with login capability. The challenge has been that due to CSRF simplicity, many organizations historically did not perceive CSRF as a vulnerability – or at least, not as one they were held responsible

to fix. As a result, organizations effectively did not want these issues reported. Fortunately this mindset has been changing. As such, several years back WhiteHat Security updated our testing methodology and predicted the numbers would rise, which for years they did -- a self-fulfilling prophecy of sorts.

6) Fingerprinting, a new addition to our top ten, presently resides at #6 overall and is found in 23% of the websites we assessed in 2012. An extremely common methodology, attackers first footprint their target's web presence and enumerate as much information as possible. Pieces of information may include the target's platform distribution and version, web application software language and technology, backend database type and version, configuration details and possibly even their network architecture/topology. Collectively this creates an instructive profile of the prospective victim. The attacker may then develop an attack chain that will more effectively exploit one or more vulnerabilities in the target website.

Note: While the presence of Fingerprinting as a class is new, we must point out that these particular vulnerabilities are not technically new. We've been reporting them all along, just placing them under the label of Information Leakage. In 2012 a change was made to break them out in their own class in order to be consistent with the WASC Threat Classification v2.

7) Insufficient Transport Layer Protection comes in at seventh on the list and is found in 22% of websites in 2012. This class allows communication to be exposed to untrusted third parties, providing an attack vector to compromise websites and/or steal sensitive information. Websites typically use Secure Sockets Layer/Transport Layer Security (SSL/TLS) to provide encryption at the transport layer. However, unless the website is configured to use SSL/TLS and configured to use SSL/TLS properly, the website may be vulnerable to traffic interception and modification. When a website uses SSL incorrectly, such as making use of weak or null ciphers, that also gets flagged as Insufficient Transport Layer Protection. The same is true if the website is using older, outdated versions of SSL. The largest set of vulnerabilities in this set is simply the lack of SSL use entirely on the given website.

8) Session Fixation moved from #9 in 2011 to #8 in 2012 and increased by four percentage points to 14% of websites. Session Fixation occurs when a website's session credential, such as a cookie, can be forcibly pre-set to a value that the bad guy knows, which then becomes valid after the user successfully authenticates. Fortunately, because of the nature of this issue, it doesn't show up in large numbers. Usually one fix in the underlying framework causes Session Fixation problems to go away from that particular website entirely.

9) URL Redirector Abuse, a newcomer to our top vulnerabilities list encompassing 13% of websites and a vulnerability once largely under-appreciated, has proved itself to be an effective tool in a phisher's arsenal. Many websites possess functionality where a visitor is redirected from one page to another on the same website, or sent off to a completely different website. Often the destination URL is set via a parameter value in the current URL (ie http://website/redirect?destination=http://anywhere/), which can be modified to any arbitrary value. A phisher

may take advantage of this functionality by sharing a URL, which goes to a completely legitimate website, but actually lands a visitor on a lookalike page under the attacker's control. The victim may then be asked to change their password by presenting their current one, and is then effectively compromised.

Note: It is important to understand that websites did not suddenly become vulnerable to this issue during 2012. The vulnerability was always there and we've been reporting it, only under the 'Insufficient Authorization' banner. Today, each new instance we find goes under its own label, so we expect these numbers to climb over time because the issue is indeed pervasive.

10) Insufficient Authorization, which was at 21% of websites and in the fourth spot in 2011, experienced a large decline during 2012, and now resides in tenth place on the list and is found in just 11% of websites. The typical Insufficient Authorization vulnerability is when an authenticated or non-authenticated user can access data or functionality that their privilege level should not allow. For example, User A can surreptitiously obtain data from User B's account, or perform actions reserved only for Admins, perhaps simply by changing a single number in a URL.

Provided the website's underlying application framework has a notion of 'authorization' built-in, typically these vulnerabilities are not overly voluminous; they are simple oversights and easy enough to fix. This could explain the large drop. Other times Insufficient Authorization vulnerabilities are indicative of a need for a huge platform upgrade, often put off for as long as possible due to the work required, potentially explaining why one in 10 websites still have at least one of them. The most probable explanation of the drop is the separating out of URL Redirector Abuse vulnerabilities from this class.

Another very extremely notable change from 2011 to 2012 is that SQL Injection no longer appears on what would technically be a Top 10 list. SQL Injection landed at #14 and was identified in just 7% of websites assessed. This is a continuation of its steady decline since 2006 when we first started reporting. While overall prevalence is now low, as most readers are acutely aware, this has not stopped attackers from leveraging SQL Injection as a leading attack vector to compromise websites and steal the data they possess. This is yet another proof point that vulnerability prevalence does not automatically equate to probability of exploitation.

Overall the 2012 WhiteHat Top 15 contains many common features with our Top 10 lists from 2011 and 2010. While progress is being made, wiping out particular vulnerability classes globally – even the most well-known – is proving to be a monstrously difficult task. The backlog of work required to repair up to 20 years of vulnerable Web code will take a while.

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 WhiteHat Top 15. The reason is that one website may possess hundreds of unique issues of a specific class, while another website may not contain any. Vulnerability classes such as Cross-

Site Scripting, Content Spoofing, and Information Leakage take the top three spots as they represent 43%, 11%, and 13% of the total population respectively. Also noticeable on the list are Cross-Site Request Forgery (7%), SQL Injection, and Brute Force (4%).

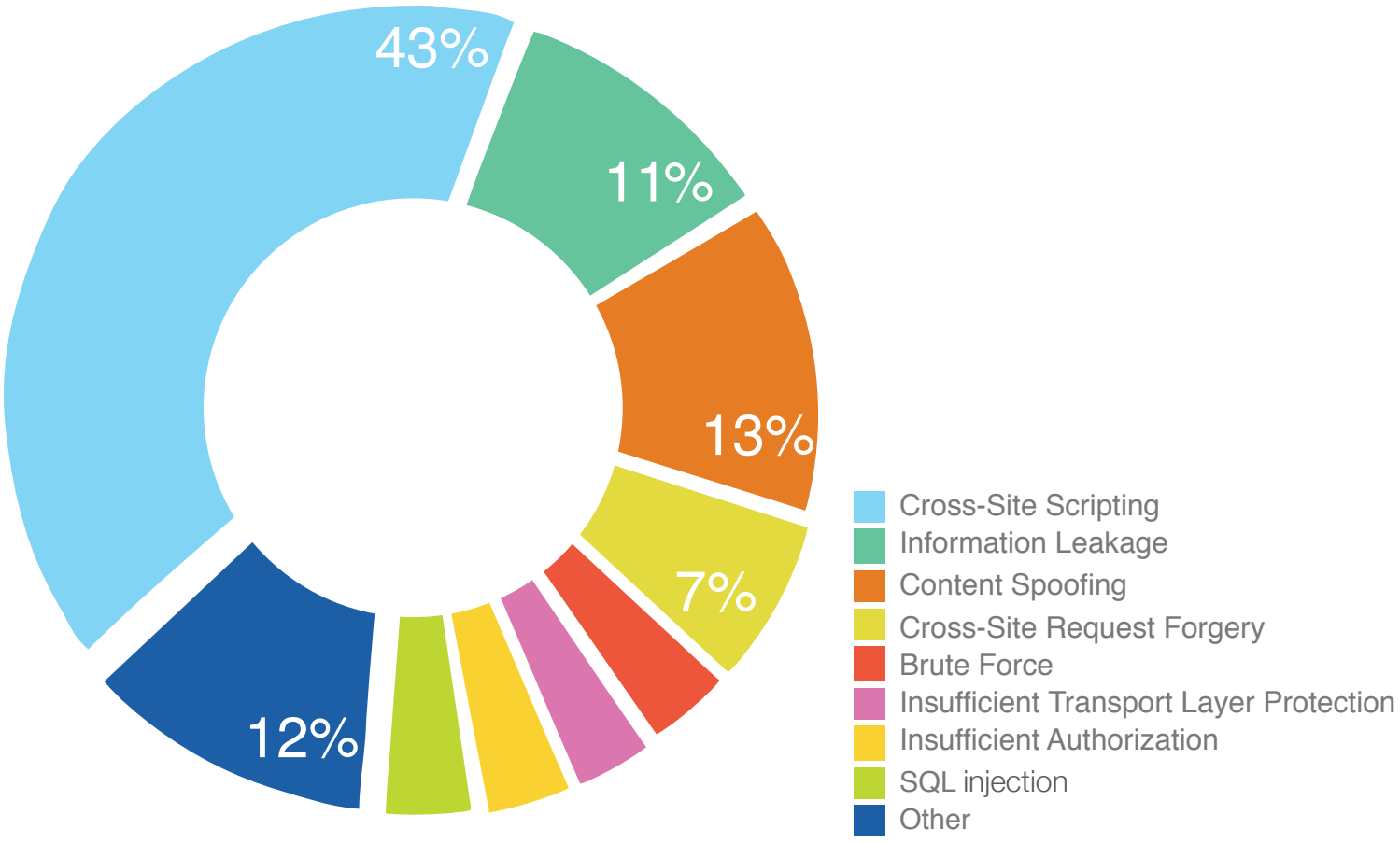


Figure 5. Overall Vulnerability Population (2012)
Percentage breakdown of all the serious* vulnerabilities discovered
(Sorted by vulnerability class)

INDUSTRY SCORECARDS

C-level executives, managers, and software developers often ask their security teams, "How are we doing? Are we safe, are we secure?" The real thing they may be asking for is a sense of how the organization's current security posture compares to their peers or competitors. They want to know if the organization is leading, falling way behind, or is somewhere in between with respect to their security posture. The answers to that question are extremely helpful for progress tracking and goal setting.

What many do not first consider is that some organizations (or particular websites) are 'targets of opportunity,' while others are 'targets of choice.' Targets of opportunity are breached when their security posture is weaker than the average organization (in their industry) – and they get unlucky in the total pool of potential victims. Targets of choice possess some type of unique and valuable information, or perhaps a reputation or brand that is particularly attractive to a motivated attacker. The attackers know precisely whom – or what – they want to penetrate.

Here's the thing: since '100% security' is an unrealistic goal – mostly because it is flatly impossible, and the attempt is prohibitively expensive and for many completely unnecessary – it is imperative for every organization to determine if they most likely represent a target of opportunity or choice. In doing so an organization may establish and measure against a "secure enough" bar.

If an organization is a target of opportunity, a goal of being just above average with respect to website security among peers is reasonable. The bad guy will generally prefer to attack weaker, and therefore easier to breach, targets. On the other hand, if an organization is a target of choice, that organization must elevate its website security posture to a point where an attacker's efforts are detectable, preventable, and in case of a compromise, survivable. This is due to the fact that an adversary will spend whatever time is necessary looking for gaps in the defenses to exploit.

Whether an organization is a target of choice or a target of opportunity, the following Industry Scorecards have been prepared to help organizations to visualize how its security posture compares to its peers (provided they know their own internal metrics, of course).

WhiteHat SECURITY

Banking Industry Scorecard

April 2013

AT A GLANCE

THE CURRENT STATE OF WEBSITE SECURITY

PERCENT OF ANALYZED SITES WITH A SERIOUS* VULNERABILITY

81%

AVERAGE NUMBER OF SERIOUS* VULNERABILITIES PER SITE PER YEAR

11

PERCENT OF SERIOUS* VULNERABILITIES THAT HAVE BEEN FIXED

54%

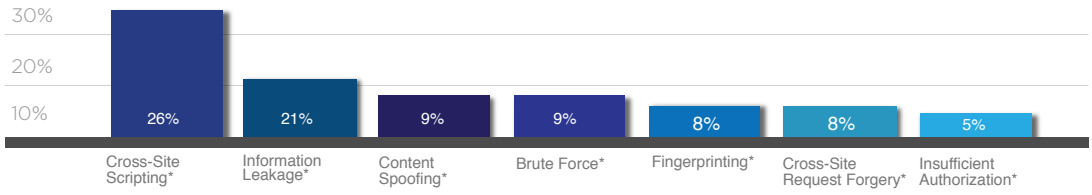
AVERAGE TIME TO FIX

107 DAYS

*Serious vulnerabilities are defined as those in which an attacker could take control over all, or a part, of a website, compromise user accounts, access sensitive data or violate compliance requirements.

MOST COMMON VULNERABILITIES

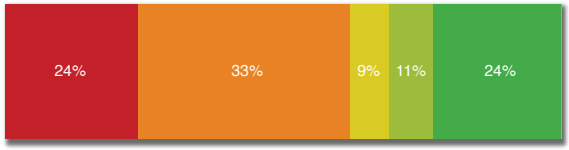
TOP SEVEN VULNERABILITY CLASSES



*The percent of sites that had at least one example of...

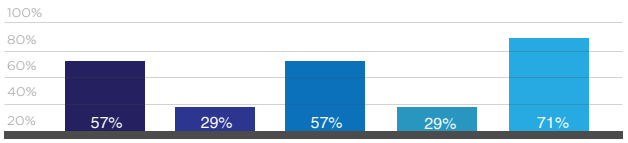
EXPOSURE AND CURRENT DEFENSE

DAYS OVER A YEAR THAT A SITE IS EXPOSED TO SERIOUS* VULNERABILITIES



- 24% Always Vulnerable
- 33% Frequently Vulnerable 271-364 days a year
- 9% Regularly Vulnerable 151-270 days a year
- 11% Occasionally Vulnerable 31-150 days a year
- Rarely Vulnerable 30 days or less a year

CURRENT APPLICATION SECURITY BEHAVIORS AND CONTROLS USED BY ORGANIZATIONS



- Programmers receive instructor led or computer-based software security training
- Applications contain a library or framework that centralizes and enforces security controls
- Perform Static Code Analysis on their website(s) underlying applications
- Web Application Firewall Deployed
- Transactional / Anti-Fraud Monitoring System Deployed

WhiteHat SECURITY

Financial Services Industry Scorecard

AT A GLANCE

THE CURRENT STATE OF WEBSITE SECURITY

PERCENT OF ANALYZED SITES WITH A SERIOUS* VULNERABILITY
81%

AVERAGE NUMBER OF SERIOUS* VULNERABILITIES PER SITE PER YEAR
50

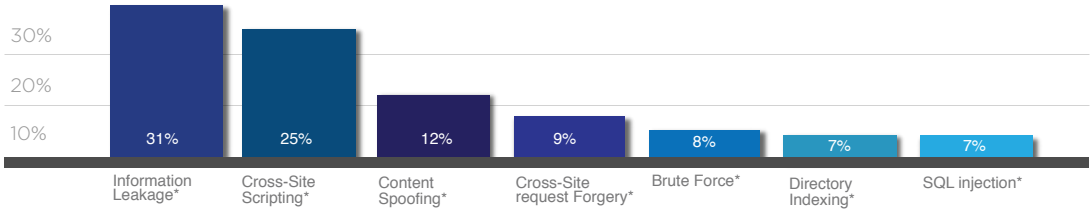
PERCENT OF SERIOUS* VULNERABILITIES THAT HAVE BEEN FIXED
67%

AVERAGE TIME TO FIX
226 DAYS

*Serious vulnerabilities are defined as those in which an attacker could take control over all, or a part, of a website, compromise user accounts, access sensitive data or violate compliance requirements.

MOST COMMON VULNERABILITIES

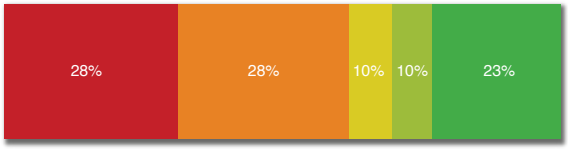
TOP SEVEN VULNERABILITY CLASSES



*The percent of sites that had at least one example of...

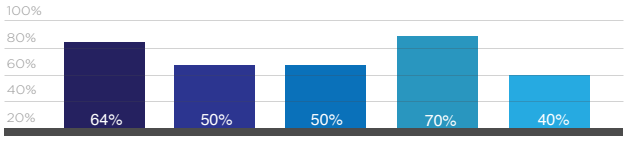
EXPOSURE AND CURRENT DEFENSE

DAYS OVER A YEAR THAT A SITE IS EXPOSED TO SERIOUS* VULNERABILITIES



- 28% Always Vulnerable
- 38% Frequently Vulnerable 271-364 days a year
- 10% Regularly Vulnerable 151-270 days a year
- 10% Occasionally Vulnerable 31-150 days a year
- 23% Rarely Vulnerable 30 days or less a year

CURRENT APPLICATION SECURITY BEHAVIORS AND CONTROLS USED BY ORGANIZATIONS



- Programmers receive instructor led or computer-based software security training
- Applications contain a library or framework that centralizes and enforces security controls
- Perform Static Code Analysis on their website(s) underlying applications
- Web Application Firewall Deployed
- Transactional / Anti-Fraud Monitoring System Deployed

WhiteHat SECURITY Healthcare Industry Scorecard

April 2013

AT A GLANCE

THE CURRENT STATE OF WEBSITE SECURITY

PERCENT OF ANALYZED SITES WITH A SERIOUS* VULNERABILITY
90%

AVERAGE NUMBER OF SERIOUS* VULNERABILITIES PER SITE PER YEAR
22

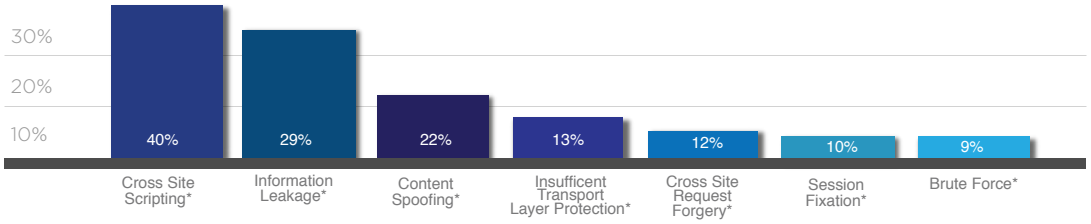
PERCENT OF SERIOUS* VULNERABILITIES THAT HAVE BEEN FIXED
53%

AVERAGE TIME TO FIX
276 DAYS

*Serious vulnerabilities are defined as those in which an attacker could take control over all, or a part, of a website, compromise user accounts, access sensitive data or violate compliance requirements.

MOST COMMON VULNERABILITIES

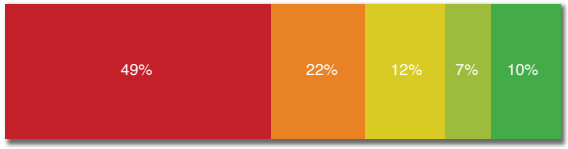
TOP SEVEN VULNERABILITY CLASSES



*The percent of sites that had at least one example of...

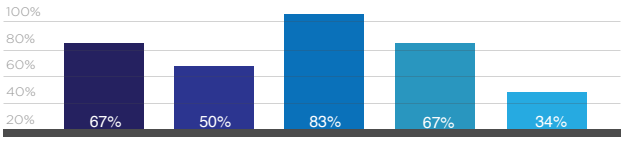
EXPOSURE AND CURRENT DEFENSE

DAYS OVER A YEAR THAT A SITE IS EXPOSED TO SERIOUS* VULNERABILITIES



- 48% Always Vulnerable
- 22% Frequently Vulnerable 271-364 days a year
- 12% Regularly Vulnerable 151-270 days a year
- 7% Occasionally Vulnerable 31-150 days a year
- 10% Rarely Vulnerable 30 days or less a year

CURRENT APPLICATION SECURITY BEHAVIORS AND CONTROLS USED BY ORGANIZATIONS



- Programmers receive instructor led or computer-based software security training
- Applications contain a library or framework that centralizes and enforces security controls
- Perform Static Code Analysis on their website(s) underlying applications
- Web Application Firewall Deployed
- Transactional / Anti-Fraud Monitoring System Deployed

WhiteHat SECURITY Retail Industry Scorecard

April 2013

AT A GLANCE

THE CURRENT STATE OF WEBSITE SECURITY

PERCENT OF ANALYZED SITES WITH A SERIOUS* VULNERABILITY

91%

AVERAGE NUMBER OF SERIOUS* VULNERABILITIES PER SITE PER YEAR

106

PERCENT OF SERIOUS* VULNERABILITIES THAT HAVE BEEN FIXED

54%

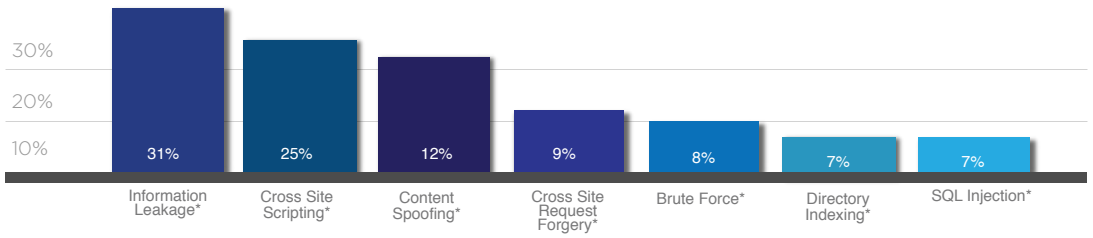
AVERAGE TIME TO FIX

224 DAYS

*Serious vulnerabilities are defined as those in which an attacker could take control over all, or a part, of a website, compromise user accounts, access sensitive data or violate compliance requirements.

MOST COMMON VULNERABILITIES

TOP SEVEN VULNERABILITY CLASSES



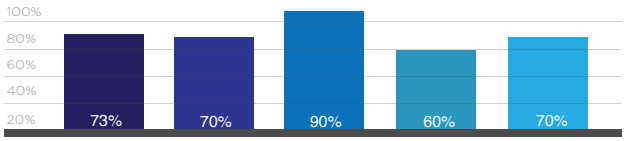
*The percent of sites that had at least one example of...

EXPOSURE AND CURRENT DEFENSE

DAYS OVER A YEAR THAT A SITE IS EXPOSED TO SERIOUS* VULNERABILITIES

- 54% Always Vulnerable
- 21% Frequently Vulnerable 271-364 days a year
- 6% Regularly Vulnerable 151-270 days a year
- 5% Occasionally Vulnerable 31-150 days a year
- 13% Rarely Vulnerable 30 days or less a year

CURRENT APPLICATION SECURITY BEHAVIORS AND CONTROLS USED BY ORGANIZATIONS



- Programmers receive instructor led or computer-based software security training
- Applications contain a library or framework that centralizes and enforces security controls
- Perform Static Code Analysis on their website(s) underlying applications
- Web Application Firewall Deployed
- Transactional / Anti-Fraud Monitoring System Deployed

WhiteHat SECURITY

Technology Industry Scorecard

April 2013

AT A GLANCE

THE CURRENT STATE OF WEBSITE SECURITY

PERCENT OF ANALYZED SITES WITH A SERIOUS* VULNERABILITY

85%

AVERAGE NUMBER OF SERIOUS* VULNERABILITIES PER SITE PER YEAR

18

PERCENT OF SERIOUS* VULNERABILITIES THAT HAVE BEEN FIXED

61%

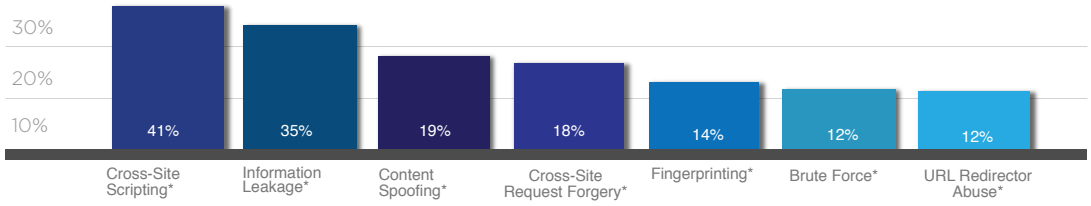
AVERAGE TIME TO FIX

71 DAYS

*Serious vulnerabilities are defined as those in which an attacker could take control over all, or a part, of a website, compromise user accounts, access sensitive data or violate compliance requirements.

MOST COMMON VULNERABILITIES

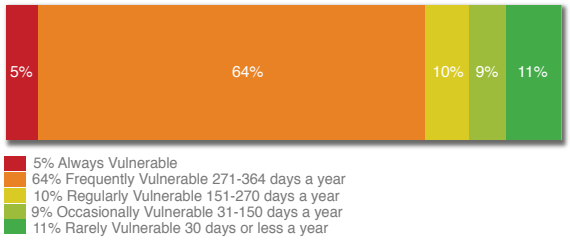
TOP SEVEN VULNERABILITY CLASSES



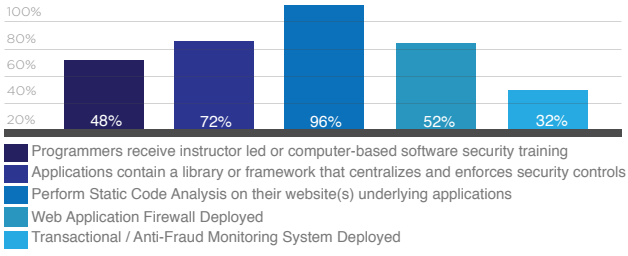
*The percent of sites that had at least one example of...

EXPOSURE AND CURRENT DEFENSE

DAYS OVER A YEAR THAT A SITE IS EXPOSED TO SERIOUS* VULNERABILITIES



CURRENT APPLICATION SECURITY BEHAVIORS AND CONTROLS USED BY ORGANIZATIONS



SURVEY

APPLICATION SECURITY IN THE SDLC

More secure software, NOT *more security software.*

How do we integrate security throughout the software development lifecycle (SDLC)? How do we measurably improve the security of the software we produce? How do we ensure we are procuring secure software?

When attempting to address these questions, it is very easy for organizations to throw away precious time and money. Corporate security teams looking for guidance will often borrow from various industry standards listing out "best-practices" that are usually just copy-pasted from a disjointed chain of "experts." These so-called best-practices might include software security training for developers, security testing during QA, static code analysis, centralized controls, Web Application Firewalls, penetration-testing, and others. The term "best-practices" implies the activity is valuable in *every* organization at *all times*. We think most would agree that just because a certain practice works well for one organization does not mean it will automatically work well at another.

Of course, we all would like to be able to say, "organizations that provide software security training for their developers experience 25% fewer serious vulnerabilities annually than those who do not." Or, "organizations that perform application security testing prior to each major production release not only have fewer vulnerabilities year-over-year, but exhibit a 35% faster time-to-fix."

Unfortunately, the commonly held assumption is that the listed best-practices above have somehow been statistically demonstrated to cost-effectively increase software and website security within any organization – that there is some supporting data-backed evidence, that there are studies showing vulnerability volumes and breaches go down when these activities are implemented. The fact is there isn't any study or data repository to this effect, at least not when it comes to anything related to website security.

What comes the closest is BSIMM (Building Security In Maturity Model) , a study of 51 real-world software security initiatives – the majority of which are from major corporations. This study provides keen insight into what these organizations are doing. Unfortunately the observed data provides zero indication that the activities actually work. By *work* we mean do they reduce vulnerability volume, speed up time-to-fix, improve remediation rates, and of course decrease the occurrence and severity of website breaches. The observations in the study are not paired with outcomes. As such, BSIMM could actually represent firms that organizations should *NOT* be like if they want to keep from getting hacked.

We contend that security experts cannot afford to make such careless assumptions about supposed best-practices, even those performed by large and reputable organizations. With so much at risk every day online, there must be no tolerance for tradition and sacred cows. Organizations need to better understand how various parts of *their* SDLC affect the introduction of vulnerabilities, which leave the door open to breaches. That said, we used BSIMM as a source of inspiration, a jumping off point to further the collective understanding of what really works in software and website security.

To move in this direction, during February 2013, we asked WhiteHat Security customers to assist us by answering roughly a dozen very specific survey questions about their 2012 SDLC and application security program. As you'll see below, these included questions such as: how often do you preform security tests on your code during QA? What is your typical rate of production code change? Do you perform static code analysis? Have you deployed a Web Application Firewall? Who in your organization is accountable in the event of a breach? We even asked "has your website been breached?"

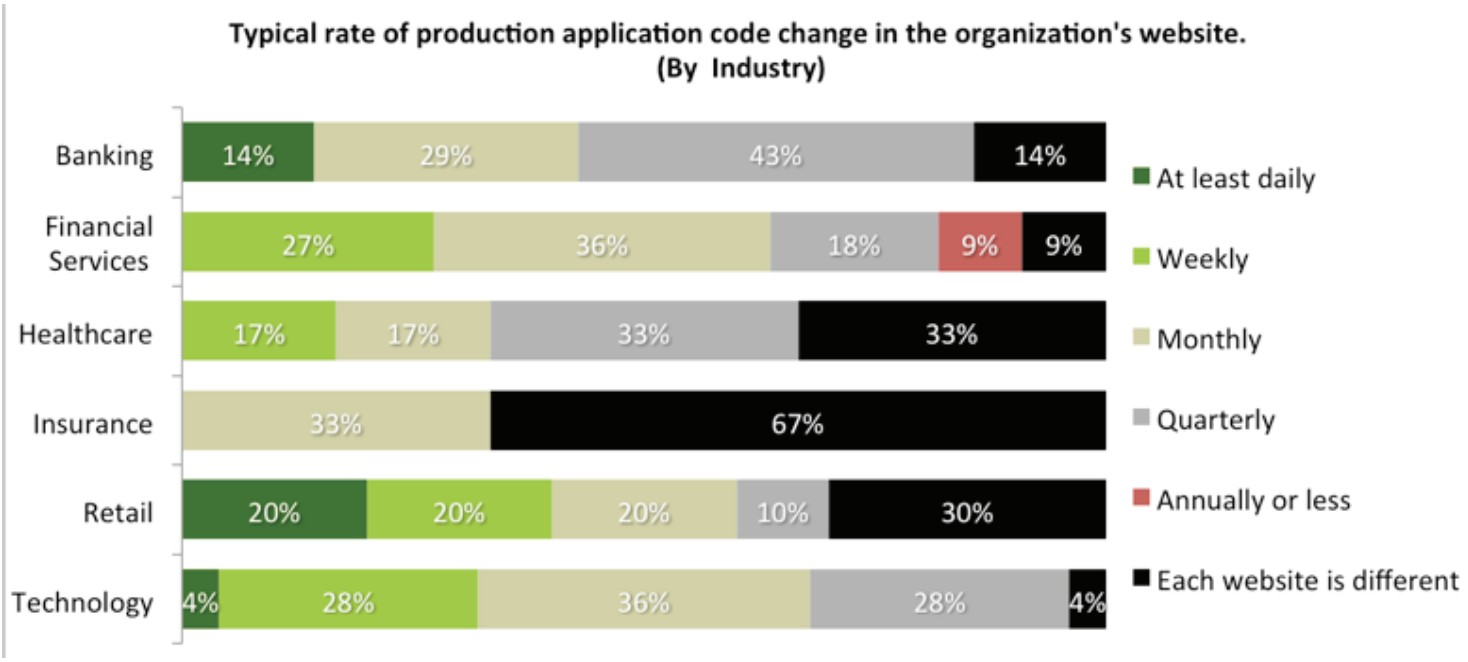
We received survey responses from 76 organizations, surpassing that of BSIMM, and then correlated those responses with company demographics and WhiteHat Sentinel website vulnerability data. The results were both stunning and deeply puzzling. The connections between various software security controls and SDLC behaviors and the vulnerability outcomes and breaches is far more complicated than we ever imagined.

Note: While we are able to correlate security performance relative to the size of an organization by revenue and numbers of employees, we did not perform that analysis in this report. This may be an avenue of additional study later on.

SDLC ACTIVITIES BY INDUSTRY

1) Rate of Production Application Code Change

A reasonable theory is that the rate at which an organization updates production code greatly affects their overall website security posture. Each code update increases the possibility for new vulnerabilities to be introduced. At the same time, each change also provides opportunities to push fixes to existing issues. To get a sense for rate of code change across our customer-base we asked, "Please select the answer that best describes the typical rate of production application code change in your organization's website(s)" and then sorted the results by industry.

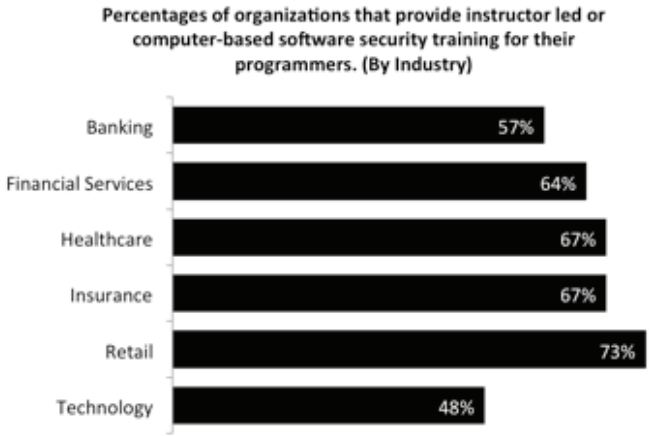


It is rare when anyone gets to see such figures laid out. As we might have expected up front, nothing really stood out to us. Clearly certain industries tend to update production more often than others, such as Retailers and Technology, more so than Banking and Insurance. Most probably could have predicted that result up front, given the type of business they are in. Interestingly, only Financial Services had some number of websites that only saw change annually (9%).

2) Software Security Training

Next we wanted to learn how many developers are receiving software security training and how much training they receive. Again, a reasonable theory is that educated programmers should generally produce more secure code. (Figures 7 & 8)

(Figure 7)



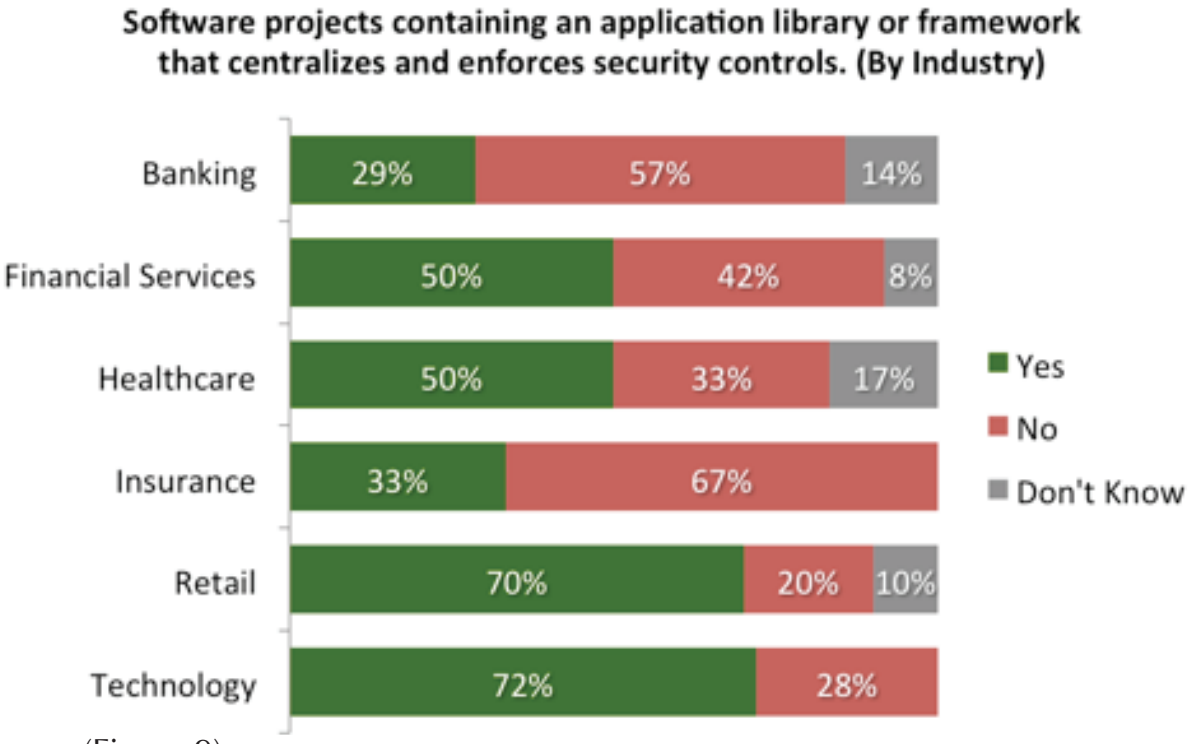
(Figure 8)

Our results show that most developers (57%) working for WhiteHat Security customers receive some amount of software security training. We cannot attest to the quality or format of the training received, but the subject at least comes up. We take this as a positive indicator.

Across industries, we see a rather even distribution of training. No industry had near universal coverage or complete absence of training. Retailers were at the top of the list with roughly three-quarters saying their developers had received training. Perhaps this performance is stimulated by PCI-DSS compliance obligations; we're unable to say with any certainty. What is also interesting is Technology had the lowest score by percentage, a little less than half. A cynical theory might be that this sector feels they already know what they're doing and does not need security training. We leave it to the reader to theorize what motivating factors might be at play.

3) Centralized Software Security Controls

Another component of an SDLC is understanding how security is technologically integrated into an application. For example, to make things easier on developers, libraries and frameworks can be made available to centralize and enforce security controls. Centralized security controls may include, but are not limited to authentication, authorization, database access, input validation, output filtering, and so on. We asked, "Do your organization's software projects contain an application library or framework that centralizes and enforces security controls?"

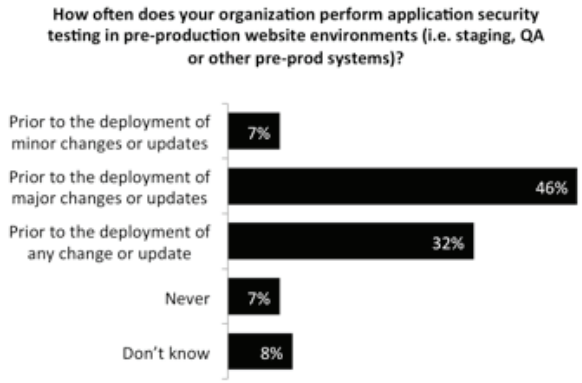


(Figure 9)

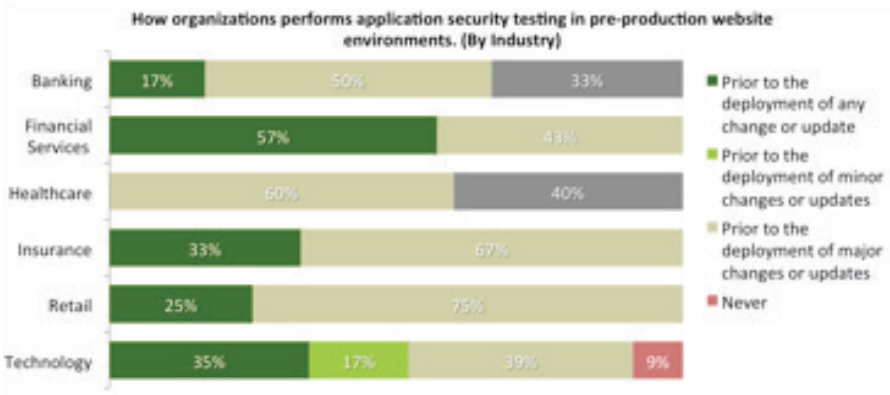
Overall, 53% of organizations said their software projects contain an application library or framework that centralizes and enforces security controls. When results are broken down by industry, we see all industries making at least some use of them. In Technology and Retail we see a heavy use of centralized security controls, nearly three-quarters; in Insurance and Banking, just over one-quarter, and half of Healthcare and Financial firms.

4) Pre-Production Application Security Testing

As anyone would agree, finding and fixing vulnerabilities during the QA process, before production release, is both safer and easier than finding and fixing them in production. Many organizations have a pre-production process that tests code for security. (Figures 10 & 11)



(Figure 10)

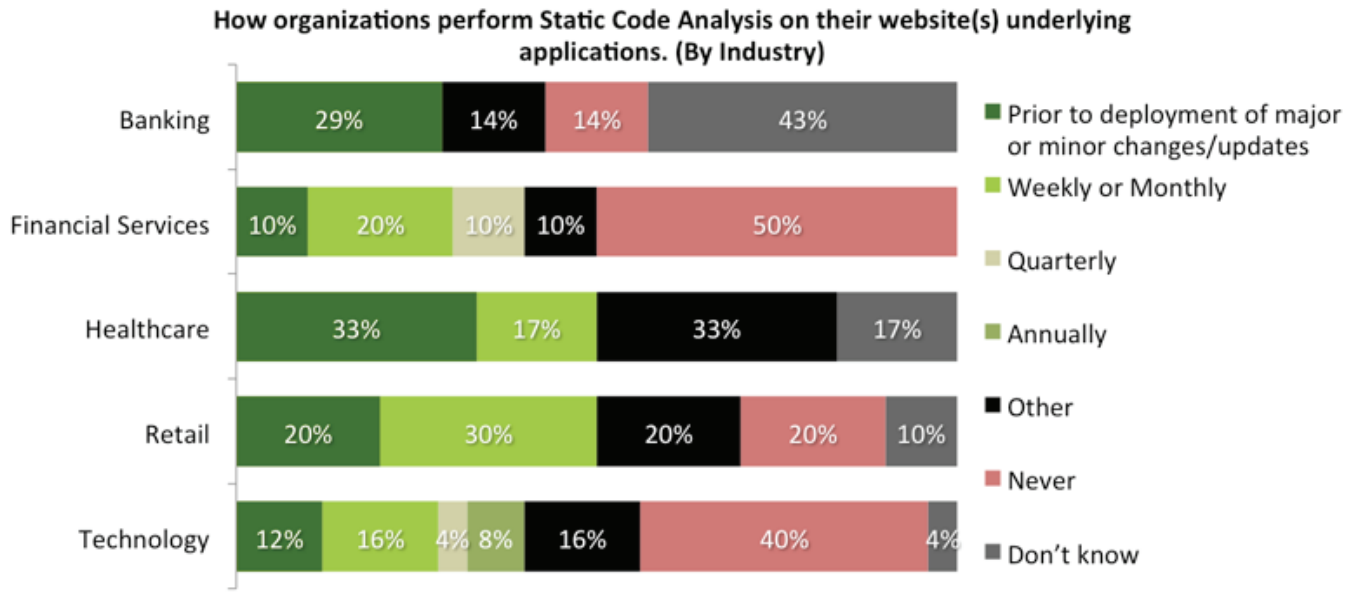


(Figure 11).

85% of organizations said they perform some amount of application security testing in pre-production website environments (i.e. staging, QA or other pre-prod systems). That's very nearly universal coverage. We do see some differences between industries. Quite clearly we're seeing Financial Services do the most testing— half before each and every change. Oddly, only a small amount of Technology firms are *not* doing some form of security QA. Could this be the result of a DevOps-Agile-esque development process? We're unsure.

5) Static Code Analysis

Static Code Analysis (SCA), the automated process of analyzing source code for issues, has been given a lot of attention over the last few years. When code is tested for security in QA, this is a common method used. Code can be tested for security issues as it is being written. We wanted to better understand how often this activity is employed so we asked, "How often does your organization perform Static Code Analysis on your website(s) underlying applications?"

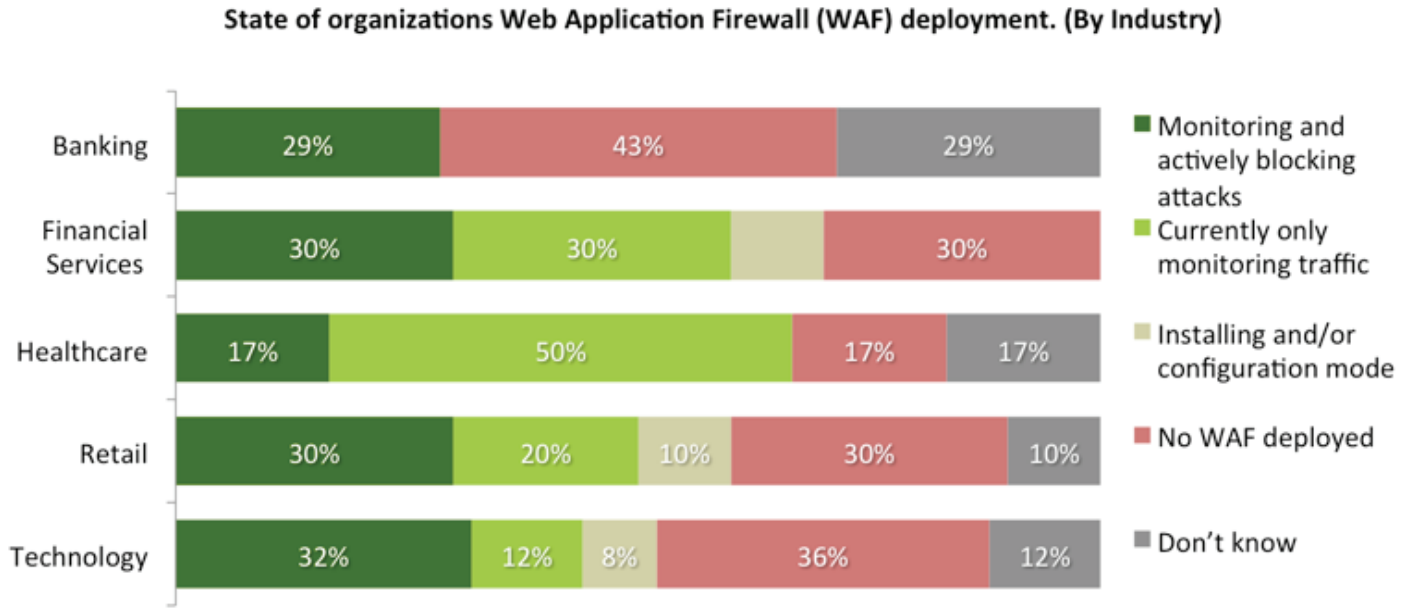


(Figure 12)

Overall, 39% of organizations said they perform some amount of Static Code Analysis on their websites' underlying applications. However, when the data is sliced by industry, we see wide variation of adoption and use of SCA. Financial Services and Retail appear to be making the most use. Interestingly, 40% of Technology companies and 50% of Healthcare companies are going without any form of SCA.

6) Web Application Firewalls

We know many firms have adopted the use of Web Application Firewalls (WAFs). WAFs have many uses, including gaining visibility into incoming website attacks and the ability block them. Exposed vulnerabilities, which might otherwise be externally exploitable, may be protected with a WAF. In our experience, WAFs in production can be in a number of deployment states – not all of which are in active blocking mode. We asked, "Please describe the state of your organizations Web Application Firewall (WAF) deployment?" (Figure 13)

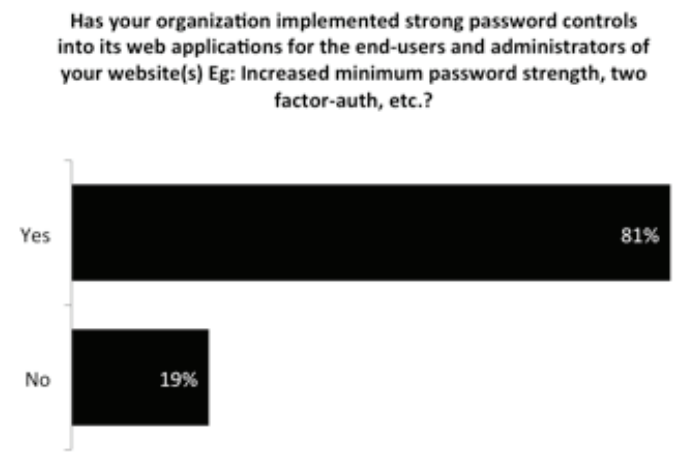


(Figure 13).

Overall, 55% of organizations said they have a Web Application Firewall (WAF) in some state of deployment. As it stands, WAFs are in "monitoring and actively blocking attacks" mode in nearly one-third of organizations across all industries we have data for. The only exception was Healthcare, where 17% are actively blocking, but 50% more have WAFs at least monitoring traffic. And notably, only in Banking do we see less than one-third of the organizations have a WAF deployed in some mode. We also know that PCI-DSS regulation has provisions that may have stimulated WAF adoption across Retail. Perhaps it has, but it is difficult to make out from this chart.

7) Strong Password Controls, Anti-Fraud Monitoring, and Audit Log Monitoring

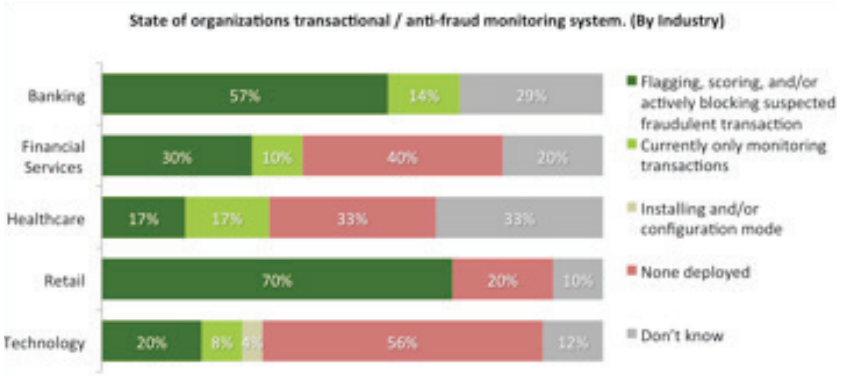
There are several other controls commonly used as part of an overall website security program. These controls might include strong password controls, anti-fraud monitoring, and audit log monitoring, in which we were curious about the level of their deployment. It is important to point out that these controls, in and of themselves, may not directly affect the vulnerability metrics in a website. However, they can be highly valuable in driving up the cost of compromising a site and in understanding what happened after the fact. (Figures 14 - 18)



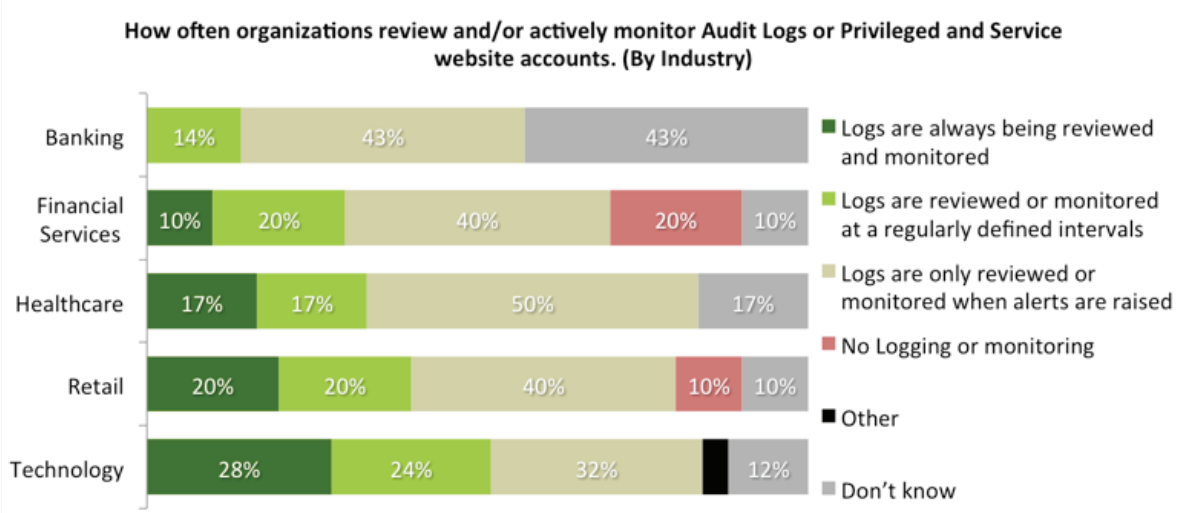
(Figure 14)

(Figure 15)

(Figure 16)



(Figure 17)



(Figure 18)

DRIVERS AND FRICTION

There are many possible reasons why organizations may choose to fix, or not fix, website vulnerabilities they are well aware of. We wanted to better understand these dynamics to get a sense for how they might impact security performance. So, we asked WhiteHat Sentinel customers to force rank their drivers to resolve websites vulnerabilities and also force rank the reasons why they go unresolved. (Figures 19 & 20)

(Figure 19)

(Figure 20)

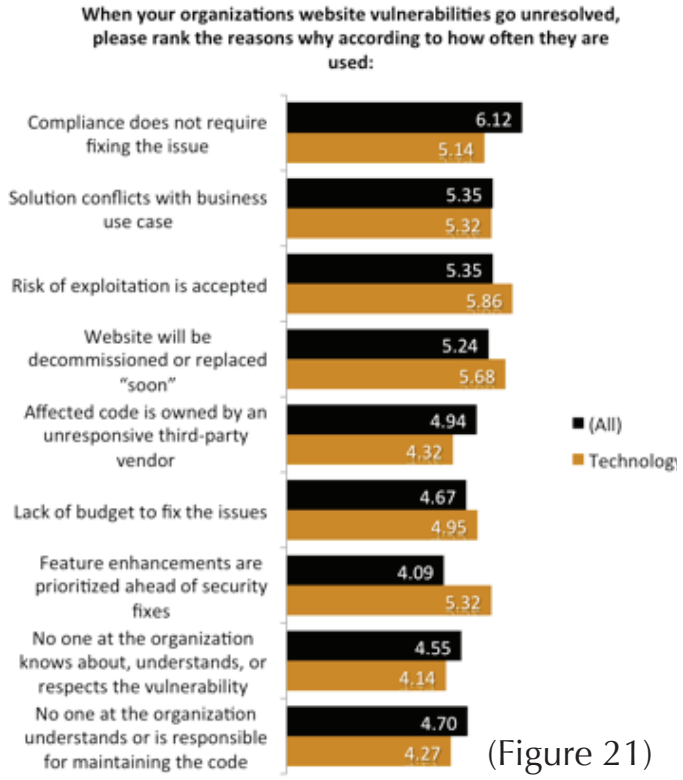
By an extremely slim margin, not perceivable in the chart, organizations said their #1 driver for resolving vulnerabilities was "Compliance," just ahead of "Risk Reduction." At the same time, Compliance was cited as the #1 reason why their vulnerabilities go unresolved. At first this sounded completely counter-intuitive, then logically the results began to make perfect sense: When security is driven by something such as compliance, security teams will do what is required and no more.

Proponents of compliance often suggest that mandatory regulatory controls be treated as a "security baseline," a platform to raise the floor, and not represent the ceiling. While this is a nice concept, it is not the enterprise reality we see. Keep in mind that WhiteHat Sentinel reports are often used to satisfy a plethora of auditors, but WhiteHat Security is not a PCI-DSS ASV nor a QSA vendor. When organizations are required to allocate funds toward compliance, which may or may not enhance security, there are often no resources left or tolerance by the business to do anything more effective.

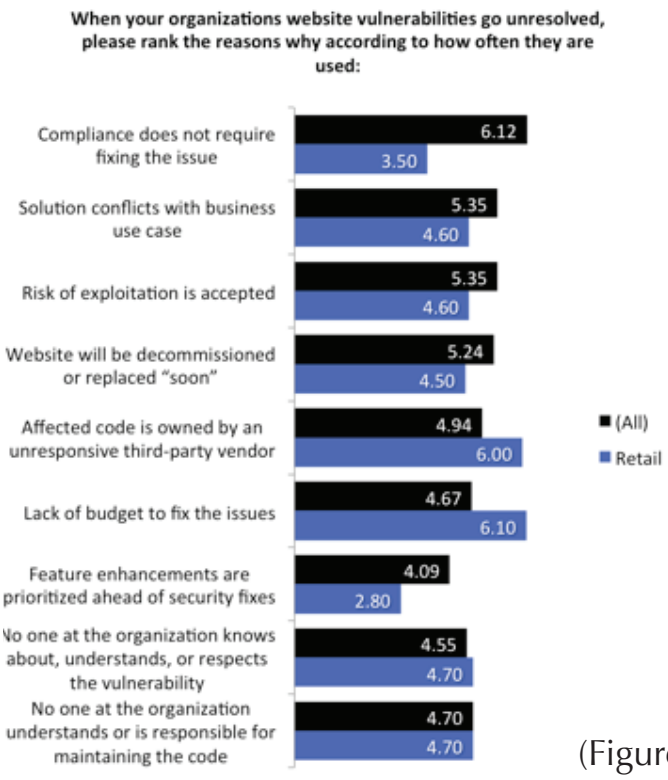
We also sliced the drivers for fixing vulnerabilities and reasons why they go unresolved by industry to see if anything stood out. Here we see Healthcare is heavily motivated by Compliance and Customer / Partner Demand, an even split, to fix issues. Then we see the biggest reason why vulnerabilities in Healthcare websites go unresolved is "No one at the organization understands or is responsible for maintaining the code." It seems this sector is challenged by large amounts of legacy code and systems. Perhaps this explains why we see a large deployment of WAFs in this industry.

In Financial Services, the leading driver to fix vulnerabilities was cited as Risk Reduction and the number one reason not to fix was Compliance. In Banking, Compliance was also the primary reason why they didn't fix vulnerabilities, but "Other" was the primary driver to fix (the data unfortunately does not lend us insights into what "Other" might have been exactly. This is another place to keep looking for answers).

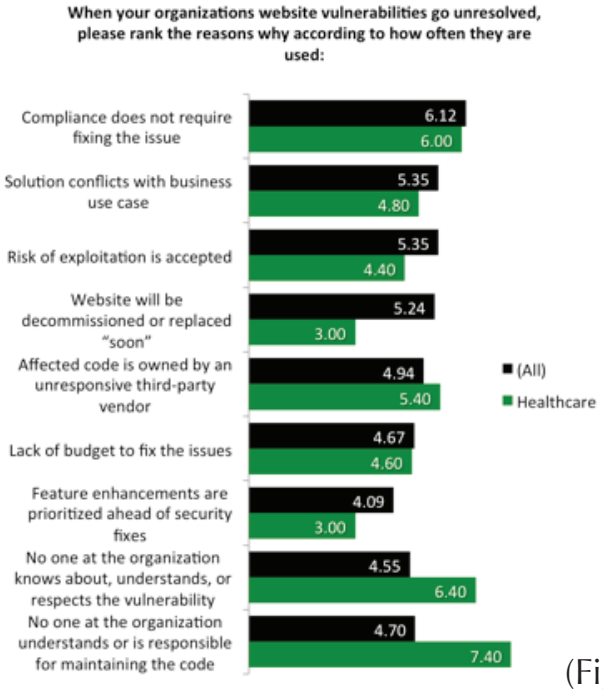
In the Technology sector, vulnerability resolution is driven by Customer / Partner Demand, while "Risk of exploitation is accepted" is the biggest reason not to fix issues. And finally in Retail, Risk Reduction is why this industry primarily fixes vulnerabilities, but "Lack of budget" is the leading reason why they are not. (Figures 21 - 26)



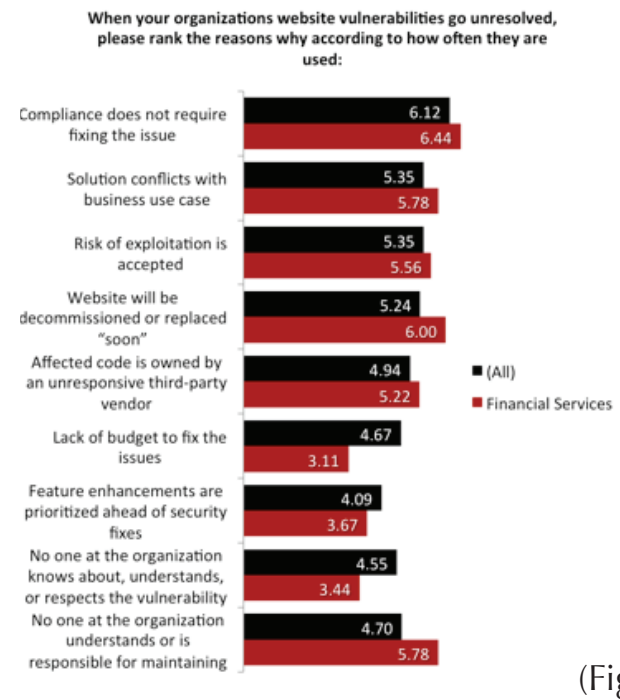
(Figure 21)



(Figure 22)



(Figure 23)

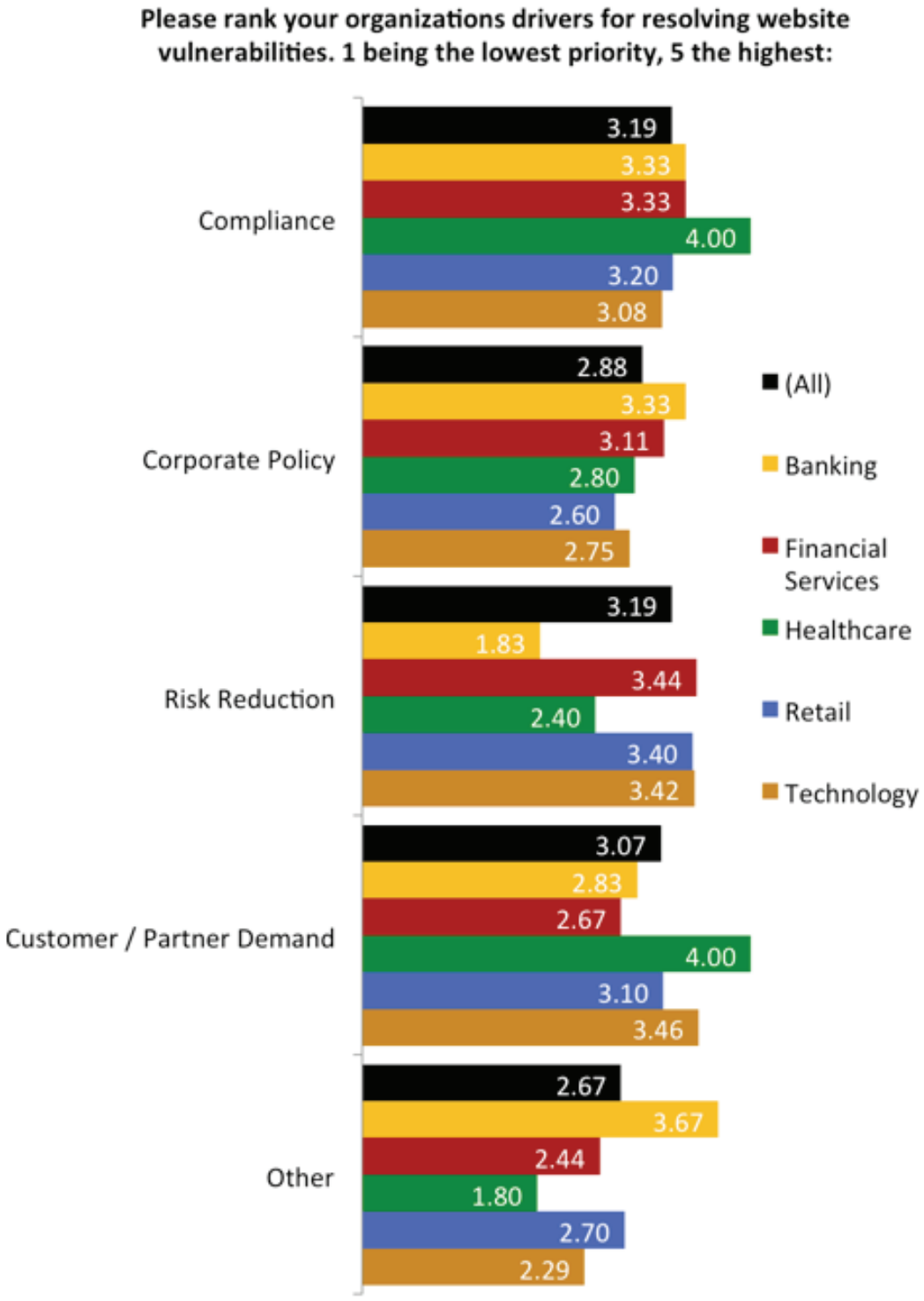


(Figure 24)


When your organizations website vulnerabilities go unresolved, please rank the reasons why according to how often they are used:
Compliance does not require fixing the issue
6.12
6.33
Solution conflicts with business use case
5.35
5.67
Risk of exploitation is accepted
5.35
3.83
Website will be decommissioned or replaced "soon"
5.24
5.33
Affected code is owned by an unresponsive third-party vendor
4.94
5.33
Lack of budget to fix the issues
4.67
3.67
Feature enhancements are prioritized ahead of security fixes
4.09
4.33
No one at the organization knows about, understands, or respects the vulnerability
4.55
6.17
No one at the organization understands or is responsible for maintaining the code
4.70
4.33
(All)
Banking

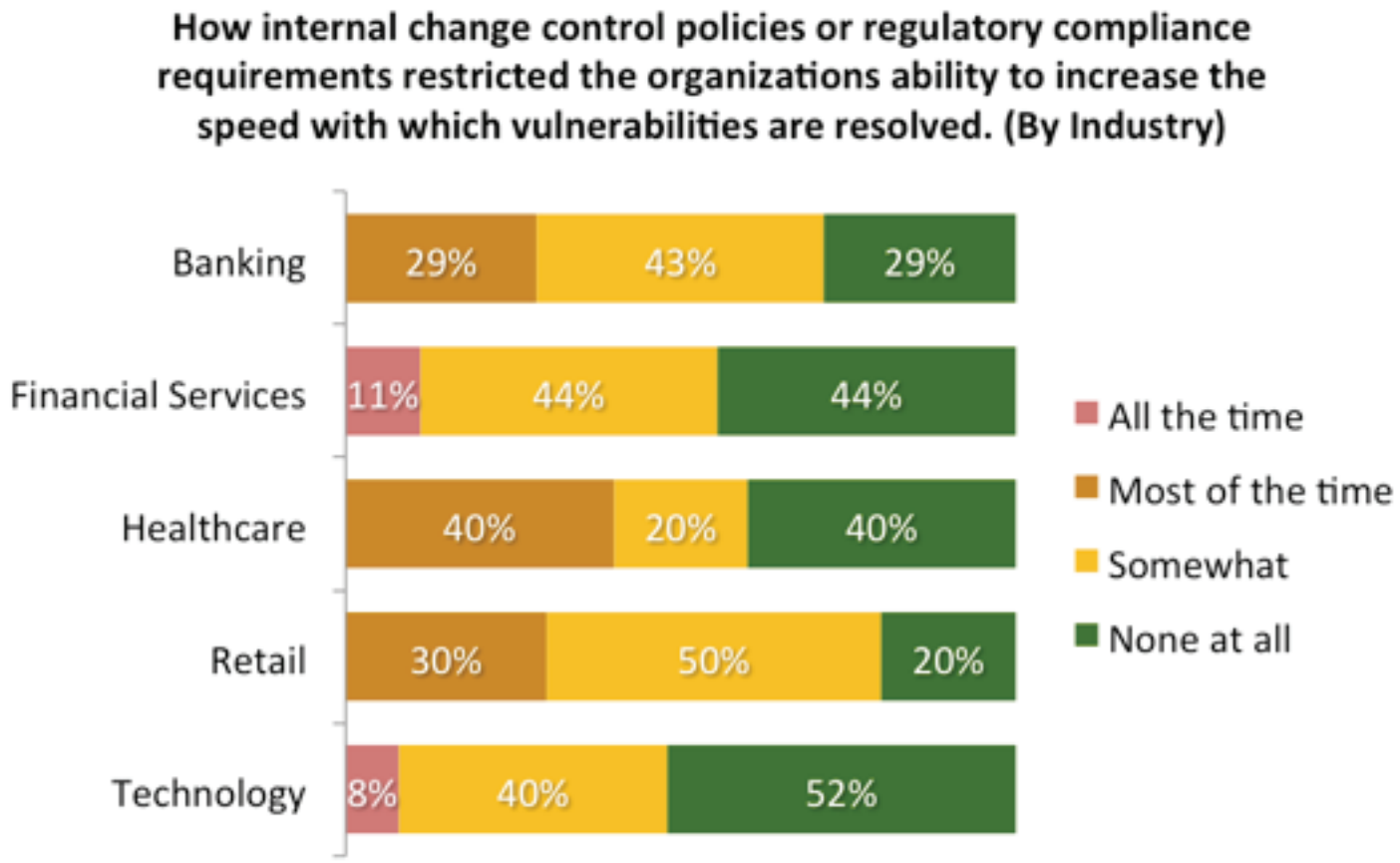
Figure 25).

.



We also wanted to get a general sense for how organizations perceived internal change control policies and regulatory requirements as an inhibitor to vulnerability resolution (see Figure 27 below). Across roughly half of all industries, it seems change control policies and regulations reported do "somewhat" slow down the fixing of vulnerabilities. In Financial Services and Technology, roughly one in 10 organizations said both have a negative impact "all the time." On the other hand, a good number of organizations across all industries said the impact was

"None at all." The question we were left with is: Why do we see such widely disparate answers in the exact same industries? How do some organizations effectively manage their change control policies and regulatory obligations so as not to be slowed down while others are severely challenged? Clearly this is an area in need of further study.



(Figure 27).

How do drivers for fixing vulnerabilities impact security performance relative to other drivers? As we can see from Figure 28 below, if an organization's #1 driver for resolving vulnerabilities is Corporate Policy, they perform the best when it comes to Average Vulnerabilities per Site, second in Average Time-to-Fix, and fourth in Average Number of Vulnerabilities Fixed (Remediation Rate). If Compliance is the leading driver, they have the speediest Time-to-Fix. When an organizations' #1 driver for fixing vulnerabilities is Customer / Partner Demand, we found the greatest percentage of fixes for their website vulnerabilities. There are a lot of potential reasons for these results, but we'll leave that speculation as an exercise for our readers.

Question: What is your #1 driver for resolving vulnerabilities?

	Average Vulnerabilities per Site Ranking	Average Time to Fix a Vulnerability Ranking	Average Number of Vulnerabilities Fixed Ranking
Answer: COMPLIANCE	2nd	1st	5th
Answer: CORPORATE POLICY	1st	2nd	4th
Answer: RISK REDUCTION	5th	5th	2nd
Answer: CUSTOMER/PARTNER DEMAND	3rd	4th	1st
Answer: OTHER	4th	3rd	3rd

(Figure 28).

Question: If an organization experiences a website(s) data or system breach, which part of the organization is held accountable and and what is its performance?

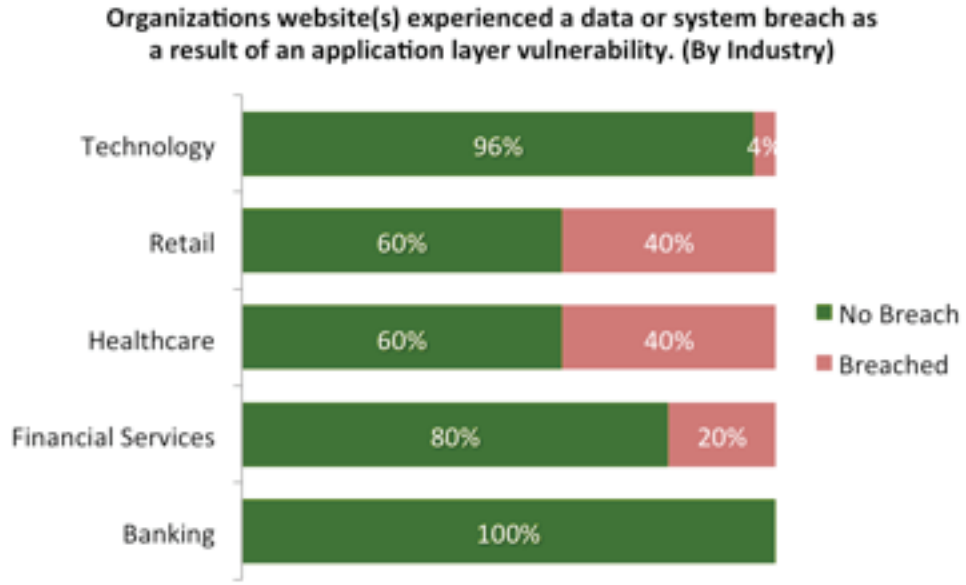
	Average Vulnerabilities per Site Ranking	Average Time to Fix a Vulnerability Ranking	Average Number of Vulnerabilities Fixed Ranking
Answer: BOARD OF DIRECTORS	4th	3rd	1st
Answer: EXECUTIVE MANAGEMENT	1st	4th	3rd
Answer: SOFTWARE DEVELOPMENT	2nd	2nd	3rd
Answer: SECURITY DEPARTMENT	3rd	1st	2nd

(Figure 29).

SURVEY ANSWERS: BREACH CORRELATION

As shown in the Figures 30 and 31 below, nearly one-quarter of our respondents said at least one of their websites suffered a data or system breach due to an application layer vulnerability. While our sample size is nowhere near that of Verizon's DBIR, the bulk of our breach data set occurred in the Retail and Healthcare industries.

(Figure 30)



(Figure 31).

What we wanted to know next was how the security metrics of those who had been breached compared to those who said they had not been. **We found:**

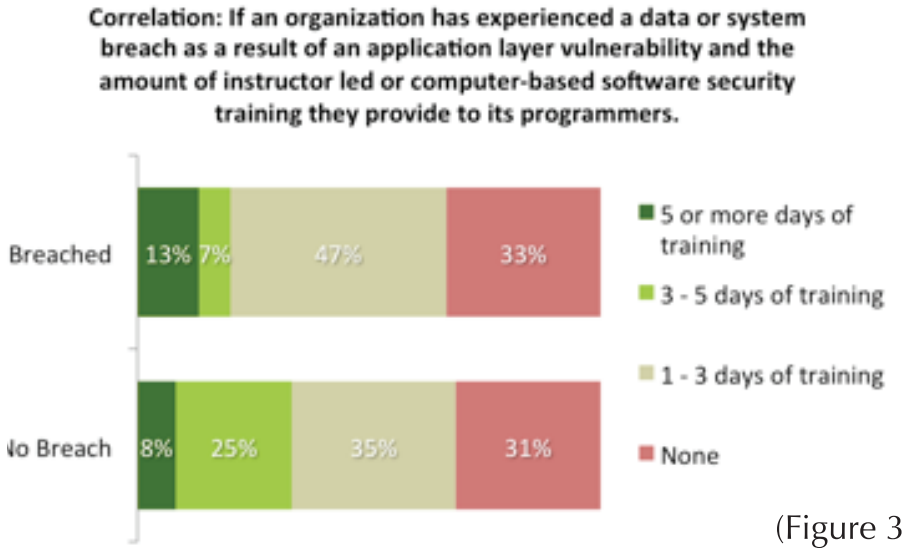
Organizations whose website(s) experienced a data or system breach as a result of an application layer vulnerability had 51% fewer vulnerabilities, resolved them 18% faster, and had a 4% higher remediation rate.

By a clear margin, those who were breached perform better than those who had not. Analysis here could go in a number of directions: Did the breach stimulate an increased security performance to ensure a breach would not happen again? Were these organizations just unlucky? Perhaps they were specifically targeted and would have been breached eventually regardless of what they did in their SDLC or how they performed. The answer could be some combination of these, or something outside our data set may be influencing the outcome. It is important not to draw any hard conclusions, but our intuition tells us that when a breach occurs, the business really starts to get serious about security.

As anyone working in the industry will tell you, providing software security training for developers is a long-standing best-practice. The question is: Does it work? Does it measurably improve an organization's security metrics and reduce breaches?

Organizations that provided instructor-led or computer-based software security training for their programmers had 40% fewer vulnerabilities, resolved them 59% faster, but exhibited a 12% lower remediation rate.

When it comes to security performance metrics, whatever type of training our customers' programmers are currently receiving, it seems to be working. Clear improvements in vulnerability volume and time-to-fix are seen. The only exception is in the area of remediation rate. Our theory is that while developers have a level of control over the number of vulnerabilities they produce with each release, they often have little organizational influence as to when existing vulnerabilities get fixed. That work is assigned to them by a development manager. If a developer is told to create a feature, they will. If they are told to fix a vulnerability, they will. In each case, however, they only do so when they are tasked to do so.



(Figure 32).

The larger issue is that training developers on software security *does not* seem to be correlated with breaches. According to Figure 32 above, among those that said they were breached, 13% had their developers attend 5 or more days of training, while those organizations in the "no-breach" category were at a very close at 8%. If there is a single thing that helps prevent website breaches – there probably is not – software security training for developers is not it. So, we have to continue looking.

Centralized software security control holds the promise of improving security metrics by making things easier on developers. Each developer working on a project no longer has to create their own custom code for dealing with authentication, authorization, database access, input validation, output filtering and so on. Instead, one or more of those controls is made available to all programmers, and the use of it may even be required by corporate policy.

Organizations with software projects containing an application library or framework that centralizes and enforces security controls had 64% more vulnerabilities, resolved them 27% slower, but demonstrated a 9% higher remediation rate.

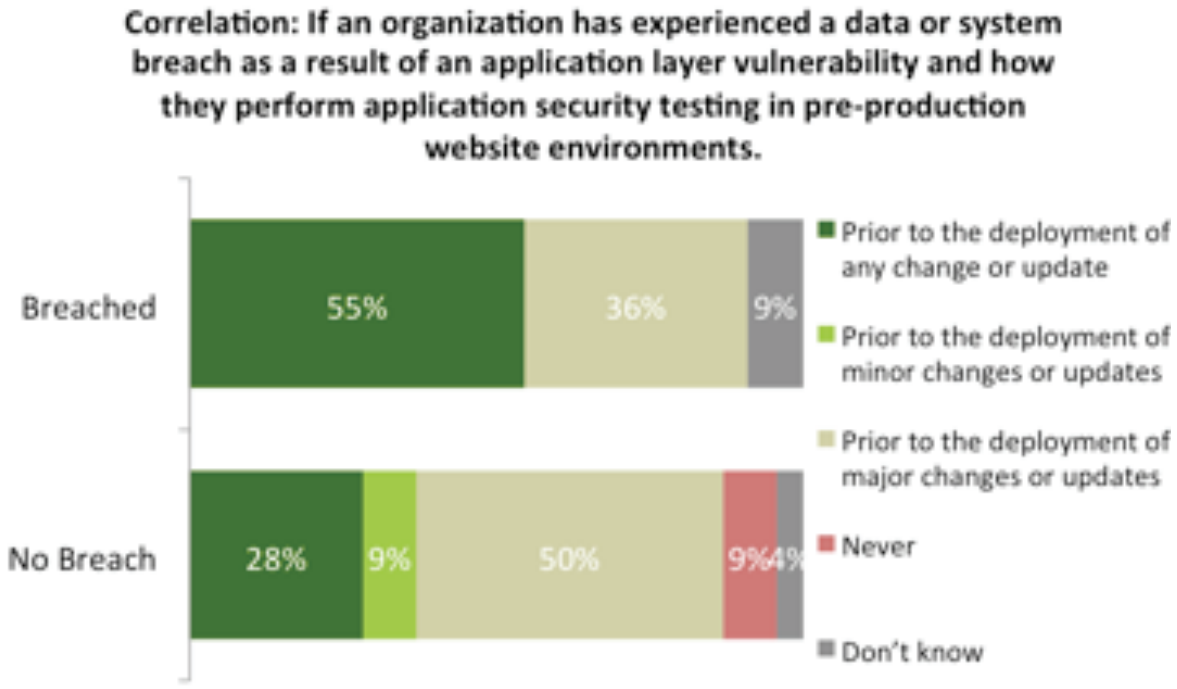
First and foremost, it's important to point out that an organization may not have a comprehensive centralized control system in place. It is possible they only have one or two controls centralized, and if so, they would still have answered, "Yes." Also, organizations may have different definitions of what constitutes centralized controls (Note: We also did not define "security controls" or what it means to "centralize" them ahead of asking the question). That variation cannot be discounted. What we can say is that these firms did at least *think* they had centralized security controls, and from that standpoint, it is enlightening.

Even with this in mind, we were quite stunned to see that those who supposedly had at least some centralized security control in place performed significantly more poorly than those who didn't – with the exception of the remediation rate. Perhaps what we are dealing with is an over-confidence problem with respect to these controls. Too much confidence that they are perfect and enabled, when in fact they are not. If we assume we find a way to bypass these controls, then we identify a large number of vulnerabilities in each place they are used. And since the controls are relied upon everywhere, any updates will require extensive QA or run the risk of breaking a large number of dependent applications.

We want to make sure that we do not discourage the use of centralized security controls. We believe the concept has a tremendous amount of value. However, proper care must be taken so that they are comprehensive, effective, tested thoroughly, and measured over time.

(Figure 33).

Do those utilizing centralized security controls suffer fewer (or more) website breaches over those who do not? As with developer training, the data doesn't indicate that centralized controls make much of a difference in either direction. Again, perhaps what works is a combination of factors. Perhaps that factor is the amount of pre-production security testing.

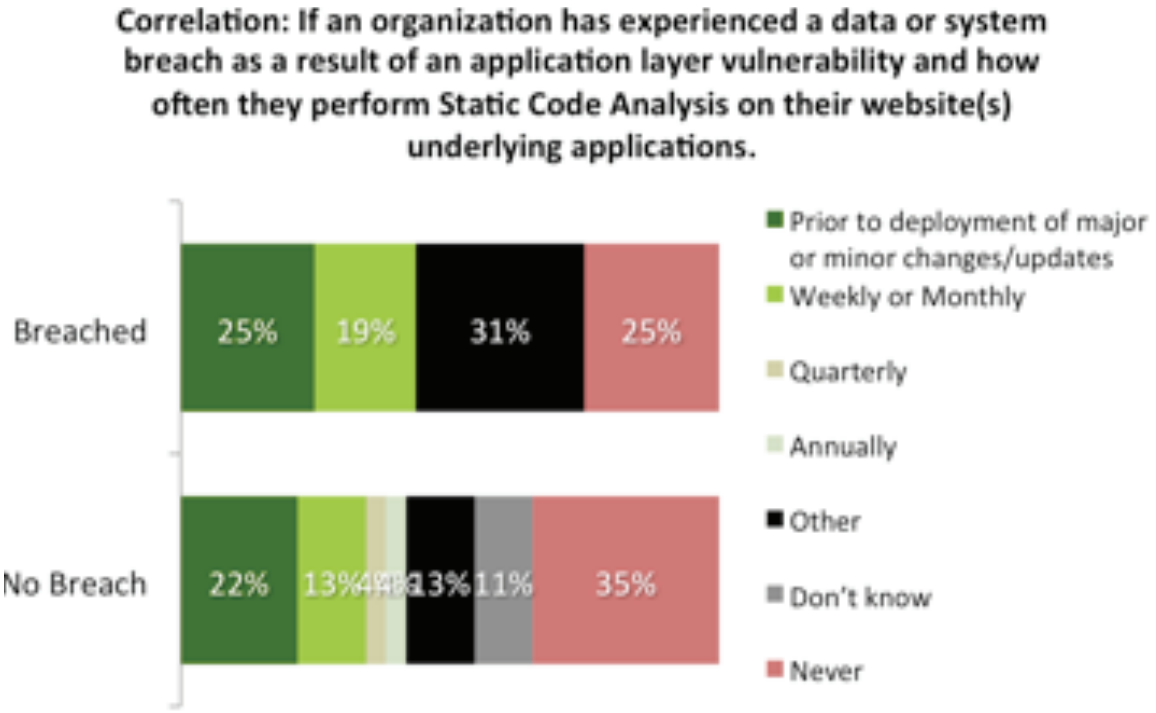


(Figure 34)

In Figure 34 we get a sense for how many WhiteHat Security customers are doing pre-production application security testing and how much they are doing. Here we see that, as with other factors already examined, pre-production testing does not appear to make any impact on breach volume. Next, we'll have a look at Static Code Analysis (SCA).

Organizations that performed Static Code Analysis on their website(s) underlying applications had 15% more vulnerabilities, resolved them 26% slower, and had a 4% lower remediation rate.

As we saw in the results from centralized security controls and pre-production testing, those organizations saying they perform some amount of SCA actually show worse security metrics in every category. This measurement was extremely surprising – even shocking. Could it be an indictment of this class of testing? It could be, but we really don't think so. More likely the chosen solution, commercial or open source, is limited in what it finds. Perhaps the vulnerabilities the SCA solution is identifying are not the same vulnerabilities that lead to website exploitation. This is a far more likely outcome. And one other possibility is that the solution was purchased, but not fully utilized and integrated in the SDLC.



(Figure 35)

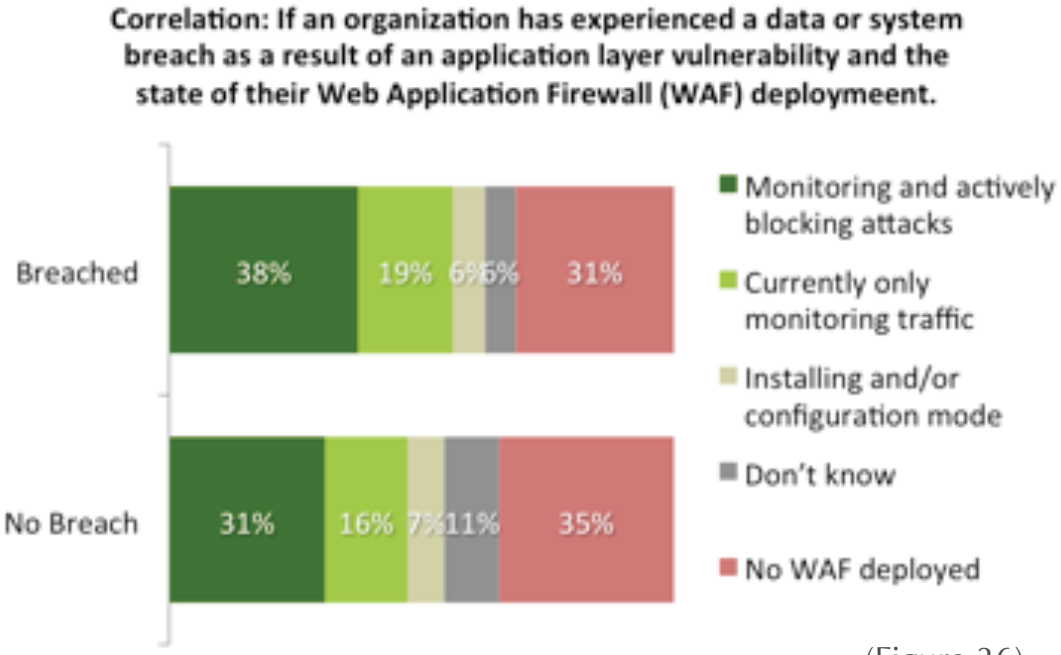
Whatever the case may be for why SCA exhibits worse security metrics, the breach metrics correlation shows no difference – just like every other best-practice we've correlated so far.

The Web Application Firewall (WAF) is the last place we'll look for answers as to what might help improve an organizations website security metrics. WAFs have the ability to block incoming attacks, protecting against the exploitation of vulnerabilities that already exist. This could definitely

positively impact website security performance.

Organizations with a Web Application Firewall deployment had 11% more vulnerabilities, resolved them 8% slower, and had a 7% lower remediation rate.

The percentages are small, but it does not appear WAF deployments are making websites less vulnerable. It could be that more of these devices need to move from configuration to blocking mode. Or perhaps what they are blocking needs to be more precise and actually block vulnerabilities the website has. It could also be that when a vulnerability is found, the organization needs to assign more resources to managing them to get the most out of a WAF.



(Figure 36).

And we should be used to it by now, but like everything else we've looked at, those with a WAF deployment fare no better or worse than those without.

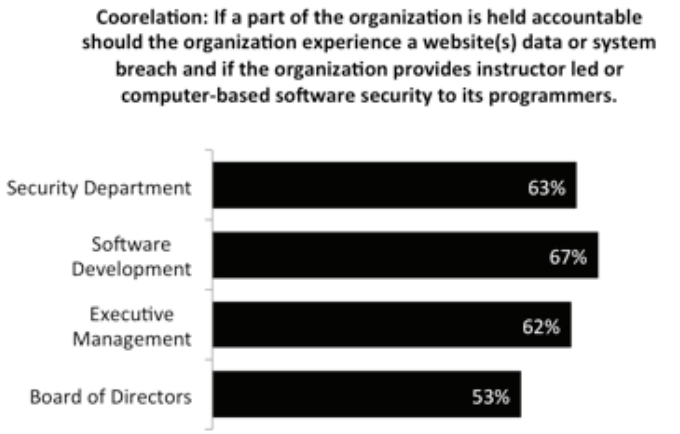
Overall, the breach correlation analysis was certainly enlightening, but also disappointing. We were really hoping to find something that actually worked rather universally, something that statistically improves security metrics AND decreases breaches. The closest we got was software security training for developers. Maybe that's the only "best-practice" that really did seem to work everywhere. However, no matter what you do, it doesn't seem to protect you from a security breach.

We don't want to leave it here, though. What we did see in the data is that many organizations are clearly deriving value from other things such as centralized controls, pre-production testing, SCA, and WAFs – while for others, there is clearly no value. The average was somewhere in-between. What this tells us is that these solutions can work, certainly they can help, but only when implemented in the right way, at the right time, targeted at the right problem. In the next section, we hope to get at some of these answers.

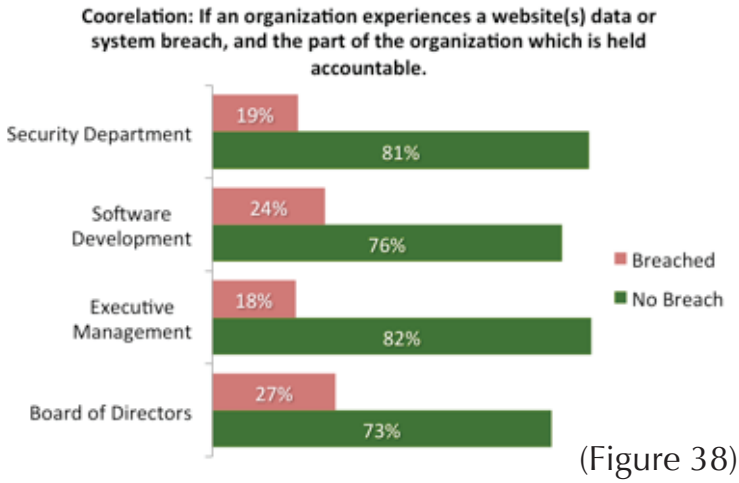
SURVEY ANSWERS: ACCOUNTABILITY CORRELATION

If there is one chart in the entire report that really opened our eyes, it is Figure 38. There is a huge delta between accountability of those who said they were breached and those who haven't been. Of those who said their security department was accountable (first row), 81% of them said there was no breach (green) while 19% said their had been (red). In software development the accountability results were nearly the same. 71% had no breach, 24% said there was. Each part of the organization, including the executive management and board of directors, showed similar results. Combine this with the other charts in the section and the evidence is strong: **Accountability matters a great deal.**

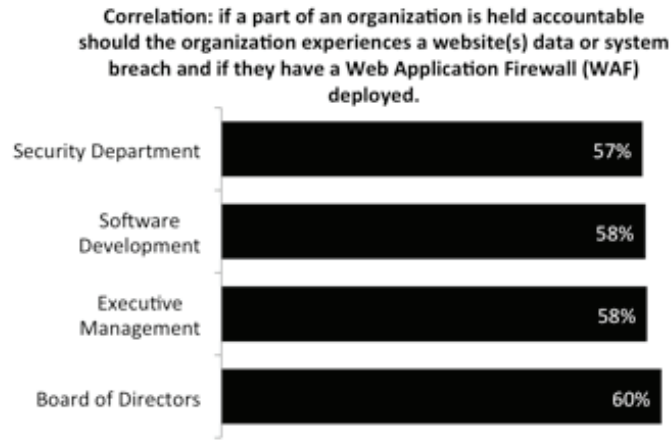
Accountability could be the underlying factor that allows SDLC activities such as training for developers, centralized controls, pre-production testing, SCA, WAFs, etc. to improve security performance metrics and decrease breaches. Accountability may be the difference between truly adopting and internalizing the activity and simply treating it as a time-wasting checkbox. The answer to what works in application and website security may certainly contain a technique and process level answer, but must include the support of an organizational mandate. The theory certainly sounds plausible.



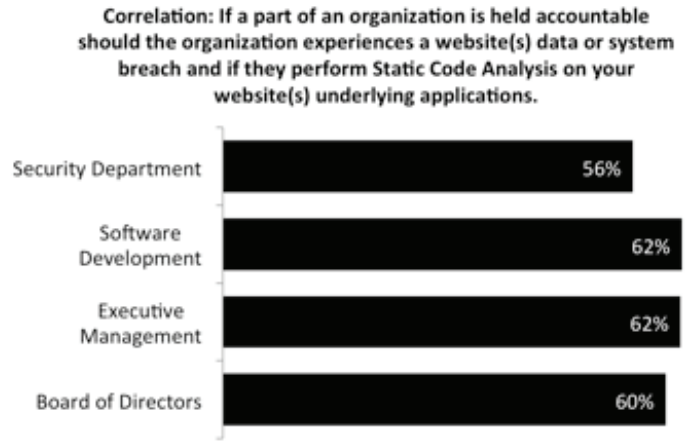
(Figure 37).



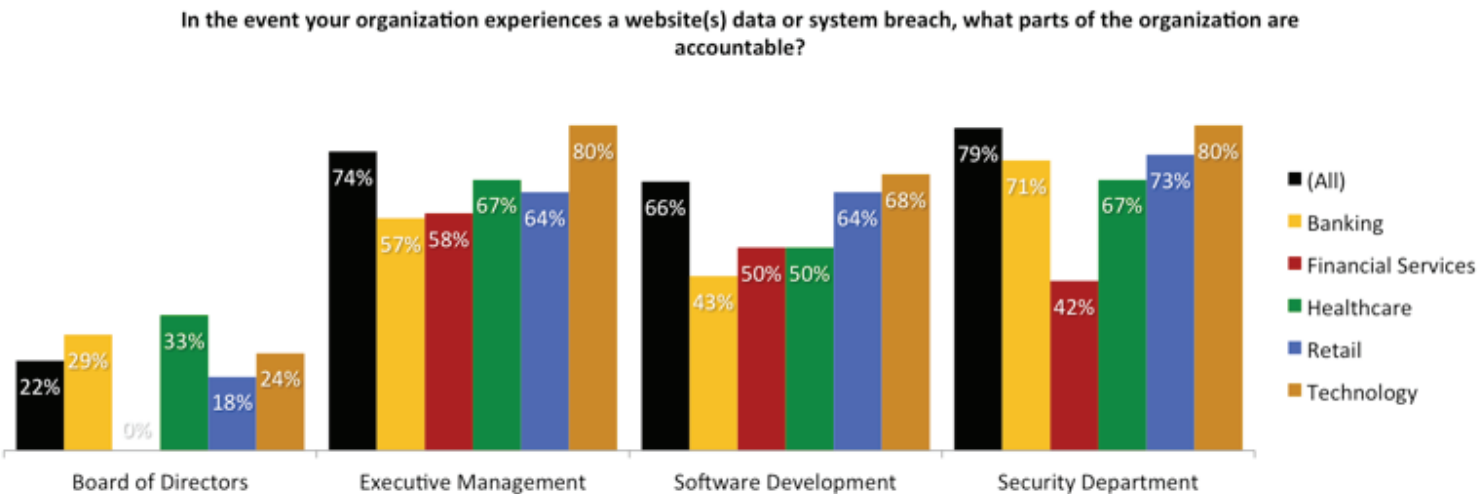
(Figure 38).



(Figure 39).



(Figure 40).



(Figure 41).

RECOMMENDATIONS

First and foremost, we believe the data contained in this report provides supportive evidence of something most information security professionals already instinctively know: accountability is an absolute necessity for an effective security program. If web and software security are truly important to an organization, then someone in the organization must not only be held accountable for security performance, but they must also be empowered to affect change. Warning: assigning responsibility without providing adequate authority is a surefire way to frustrate individuals and encourage talent to walk out the door.

Accountability may begin at as high a level as the board of directors, which is often the case in the aftermath of a breach. From there, accountability will naturally filter down throughout the organization: through executive management to security teams to software developers, and even through to the software procurement department. Without a foundation of accountability, even the best-laid plans will fail. The data leads us in this direction. With accountability, there will be a tangible return on investment regardless of what best-practices an organization might implement. Furthermore, high-end security talent appreciates this kind of responsibility and views it as a career advancement opportunity.

Secondly, there are no best-practices. Our data and experience strongly suggest that security "best-practices" in the software development lifecycle (SDLC) are not universally effective, but are situation-dependent. Whether referring to source code reviews, security testing during QA, Web Application Firewalls, or other methods, each has an appropriate time and place. The challenge in increasing the organization's security posture cost-effectively is determining what to recommend, what to implement, what to purchase, and when. Security teams must get this right.

The tactical key to improving a web security program is having a comprehensive metrics program in place – a system capable of performing ongoing measurement of the security posture of production systems, exactly where the proverbial rubber meets the road. Doing so

provides direct visibility into which areas of the SDLC program are doing well and which ones need improvement. Failure to measure and understand where an SDLC program is deficient before taking action is a guaranteed way to waste time and money – both of which are always extremely limited.

Below is a step-by-step strategy for building out a website security program that yields results:

0) Assign an individual or group that is accountable for website security: These individuals or groups may include the board of directors, executive management, security teams, and software developers. They should be commissioned and authorized to establish a culturally consistent incentives program that will help move the organization in a positive direction with respect to security.

1) Find your websites – all of them – and prioritize: Prioritization can be based on business criticality, data sensitivity, revenue generation, traffic volume, number of users, or other criteria the organization deems important. Knowing what systems need to be defended and what value they have to the organization provides a barometer for an appropriate level of security investment.

2) Measure your current security posture from an attacker's perspective: This step is not just about identifying vulnerabilities; while that is a byproduct of the exercise, it's about understanding what classes of adversaries you need to defend against and what your current exposure to them is. Just finding vulnerabilities is not enough. Measure your security posture the same way a bad guy would before they exploit the system – fixing those vulnerabilities first is what's important.

3) Trend and track the lifecycle of vulnerabilities: At a minimum, measure how many vulnerabilities are introduced per production code release, what vulnerability classes are most prevalent, the average number of days it takes to remediate them, and the overall remediation rate. The result provides a way to track the organization's progress over time, and serves as a guide for which of the SDLC-related activities are likely to make the most impact. Anything measured tends to improve.

4) Fast detection and response: Historically it has been prudent to operate under the assumption that [all] networks are compromised, or are at least hostile. This is the case especially since everyone is only one zero-day away from a break-in. Borrowing from that frame of reference, application security professionals are well advised to take a similar approach and focus on the impact of that assumption. Start by asking the question: "If my application is already vulnerable what action(s) should I begin taking?" If an organization is breached, the real damage happens when the adversary is in the system for days, weeks, or months. If an intruder can be successfully identified and kicked off the system within hours, the business impact of a breach can be substantially minimized.

APPENDIX A: ABOUT THE DATASET

WhiteHat Sentinel is the most accurate, complete and cost-effective website vulnerability management solution available. It delivers the flexibility, simplicity and manageability that organizations need to control their website security and prevent Web attacks. WhiteHat Sentinel is built on a Software-as-a-Service (SaaS) platform that scales massively, supports the largest enterprises, and offers the most compelling business efficiencies to lower your overall cost for website security.

Unlike traditional website scanning software or consultants, only WhiteHat Sentinel combines proprietary scanning technology with custom testing by the industry's only Threat Research Center (TRC) . The WhiteHat Security TRC is a 100+ team of website security experts who act as a critical and integral component of the WhiteHat Sentinel family. The TRC is as an extension of your website security team – actively managing business website risk posture so you can focus on technology and business goals.

DATA, PLATFORM, AND METHODOLOGY

• 15,000+ production and pre-production websites across 650+ organizations representing many of the world's most recognizable brands, across industries including Banking, Education, Energy, Financial Services, Healthcare, Information Technology, Insurance, Manufacturing, Non-Profit, Retail, Social Networking, Telecommunications, etc.

• The websites WhiteHat Security assesses generally represent the more important and secure websites on the Web, owned by organizations that care about the security of their websites.

• Each WhiteHat Sentinel customer self-selects the most appropriate industry label for each website under service.

• While WhiteHat Sentinel customers maintain complete control over their vulnerability assessment schedule, the majority of websites are assessed for vulnerabilities multiple times per month.

• The WhiteHat Sentinel platform currently peaks at over 5,000 simultaneous scans, collectively generating well over a billion HTTP requests per month, consuming many terabytes of HTTP-related data per week, and necessitating 100 Mbps of bandwidth 24x7.

• Vulnerabilities are classified according to WASC Threat Classification v2

• Severity naming convention aligns with PCI-DSS

WhiteHat Sentinel Service
https://www.whitehatsec.com/sentinel_services/sentinel_services.html
WhiteHat Security Threat Research Center (TRC)
https://www.whitehatsec.com/sentinel_services/threat_research.html
WASC Threat Classification (v2)
http://projects.webappsec.org/Threat-Classification
PCI Data Security Standard (PCI DSS)
https://www.pcisecuritystandards.org/security_standards/pci_dss.shtml

IMPORTANT FACTORS INFLUENCING THE DATA

1. Websites range from highly complex and interactive with large attack surfaces to static brochureware. Brochureware websites, because of their relatively limited attack surface, tend to have a lower number of "custom" Web application vulnerabilities.
2. The customer-base is largely, but not exclusively, US-based, as are their websites. However, a significant number of websites are localized in a variety of languages.
3. Vulnerabilities are counted by unique Web application and vulnerability class. If three of the five parameters of a single Web application (/foo/webapp.cgi) are vulnerable to SQL Injection, this is counted as 3 individual vulnerabilities (e.g. attack vectors). Secondly, if a single parameter can be exploited in more than one way, each of those are counted as well. We count this way because a vulnerability in each parameter may actually lead to a different problem in a different part of the code.
4. Only serious* vulnerabilities that can be directly and remotely exploitable and that may lead to data loss or account compromise are included.
5. "Best practice" findings are not included in the report. For example, if a website mixes SSL content with non-SSL on the same Web page, while this may be considered a policy violation, it must be taken on a case-by-case basis.
6. Our vulnerability assessment processes are incremental and ongoing. The frequency of assessments, which is customer-driven, should not automatically be considered "complete."
7. It is best to view this report as a best-case scenario, and there are always more vulnerabilities to be found. New attack techniques are constantly being researched to uncover previously unknown vulnerabilities, including in previously tested and unchanged code. Likewise assessments may be conducted in different forms of authenticated state (i.e. user, admin, etc.).
8. Websites may be covered by different WhiteHat Sentinel Service. Premium (PE), Standard (SE), Baseline (BE), and PreLaunch (PL) offer varying degrees of testing criteria, but all include verification. PE covers all technical vulnerabilities and business logic flaws identified by the WASC Threat Classification (and some beyond). SE focuses primarily on the technical vulnerabilities. BE bundles critical technical security checks into a production-safe, fully-automated service.
9. There is some amount of double website / vulnerability counting. Customers sometimes deploy WhiteHat Sentinel simultaneously on production websites and their internal QA/Staging mirrors. As standard practice, our process makes no assumption that any website is identical to another. It is also important to note that it is statistically rare for production and QA/Staging to have identical vulnerabilities.

The Platform

Built as a Software-as-a-Service (SaaS) technology platform, WhiteHat Sentinel combines proprietary scanning technology with analysis by security experts in its Threat Research Center to enable customers to identify, prioritize, manage and remediate website vulnerabilities. WhiteHat Sentinel focuses solely on previously unknown vulnerabilities in custom Web applications – code unique to an organization. Every vulnerability discovered by any WhiteHat Sentinel Service is verified for accuracy and prioritized by severity and threat.

The Methodology

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 Standard (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 scanner settings.

WhiteHat Sentinel offers four different levels of service (Premium, Standard, Baseline, and PreLaunch) to match the level of security assurance required by the organization. Additionally, WhiteHat Sentinel exceeds PCI 6.6 and 11.3.2 requirements for Web application scanning.

Scanning Technology

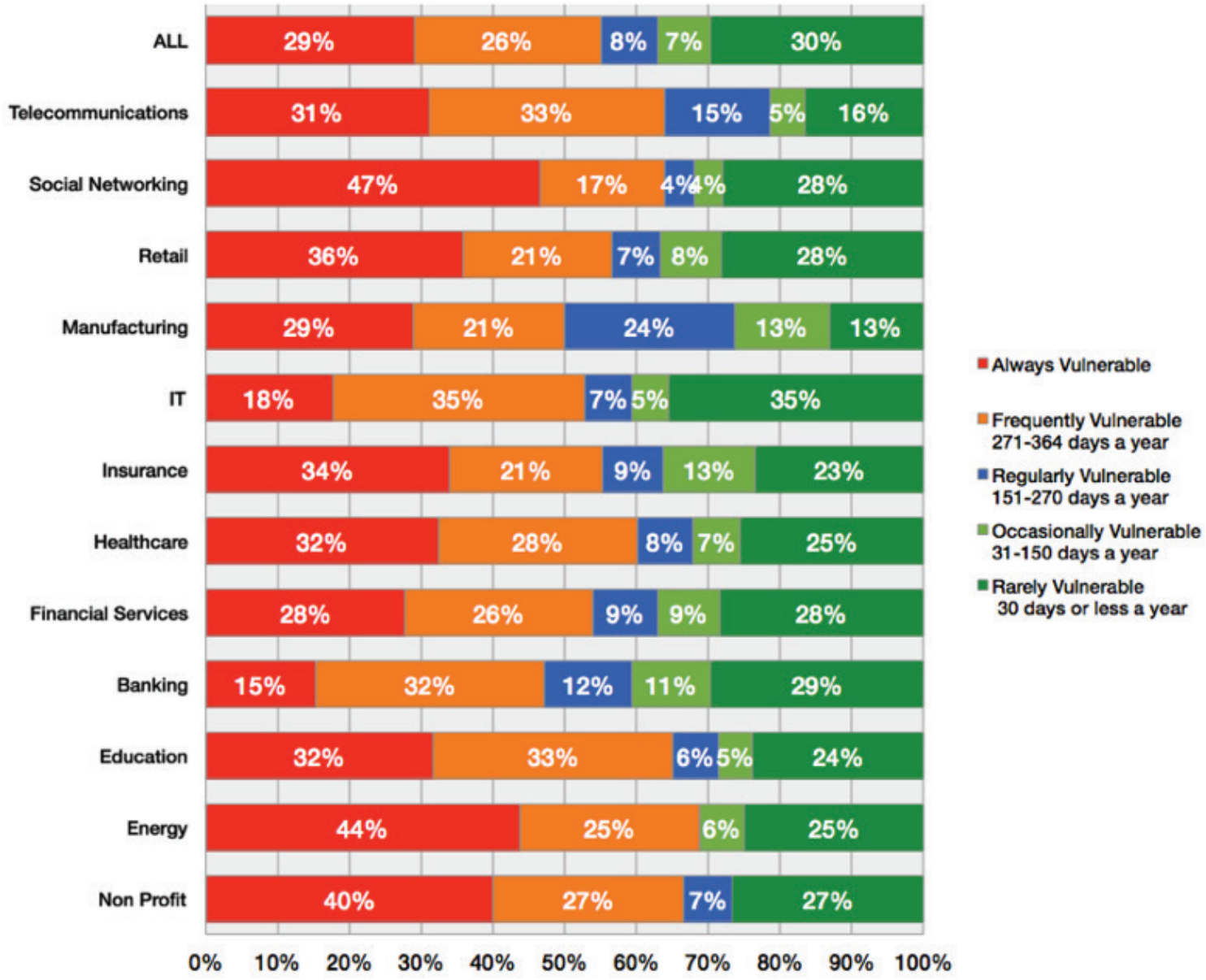
- Production Safe (PE, SE, BE): Non-invasive testing with less performance impact than a single user.
- False-positives: Every vulnerability is verified for accuracy by WhiteHat Security's Threat Research Center.
- Web 2.0 Support: JavaScript, Flash, AJAX, Java Applets, and ActiveX are handled seamlessly.
- Authenticated Scans: Patented automated login and session-state management for complete website coverage.
- Business Logic: customized tests analyze every form, business process, and authentication / authorization component.

APPENDIX B

Industry	Annual Avg. Vulnerabilities	Std. Dev	Avg. Time-to-Fix (Days)	Average Remediation	Std. Dev	Window of Exposure (Days)	Std. Dev
ALL	79	670	38	63%	36	231	159
Banking	17	554	45	74%	37	185	147
Education	53	885	30	46%	37	261	153
Financial Services	67	853	80	63%	35	227	157
Healthcare	48	461	35	63%	36	239	155
Insurance	92	171	40	58%	32	211	154
IT	85	36	35	57%	31	208	159
Manufacturing	30	56	17	50%	33	252	125
Retail	121	125	27	66%	36	238	160
Social Networking	31	431	41	62%	43	264	162
Telecom	52	82	50	69%	31	271	136
Non-Profit	37	56	94	56%	40	320	168
Energy	31	62	4	40%	35	250	154

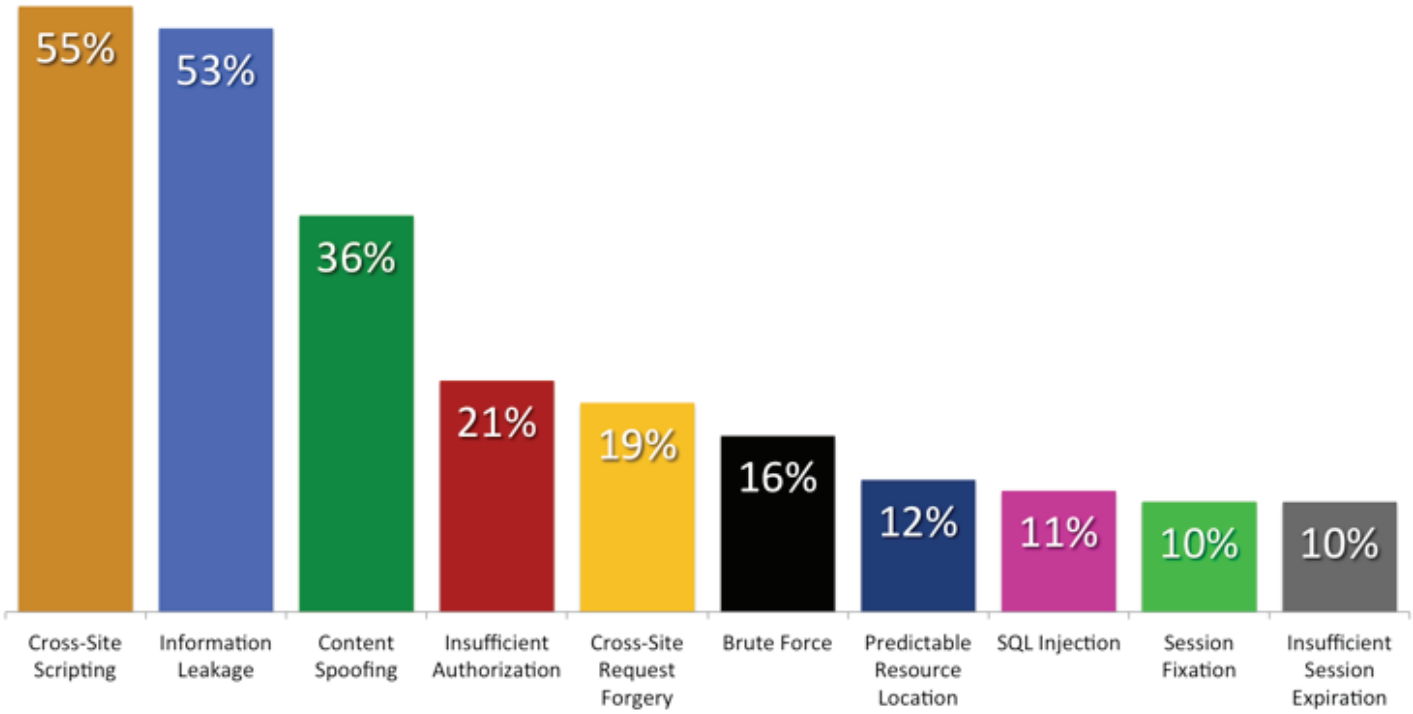
At a Glance: The Current State of Website Security (2011)
(Sorted by Industry)

APPENDIX C



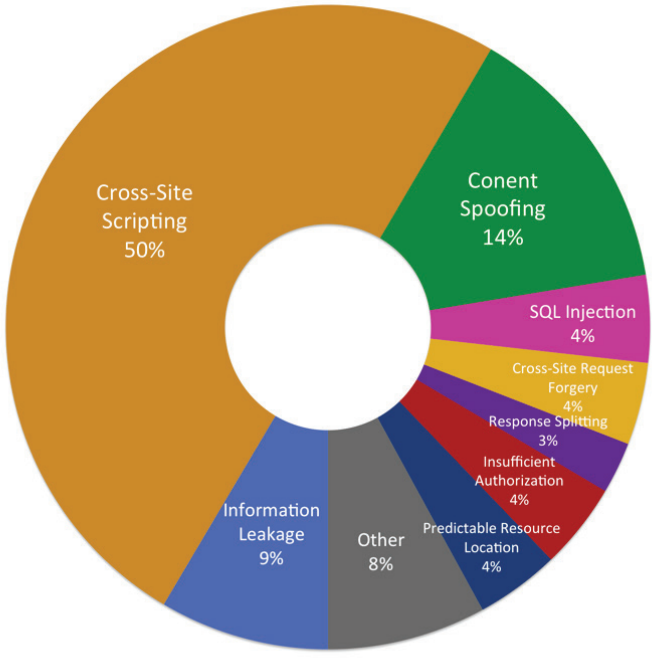
Overall Window of Exposure to Serious* Vulnerabilities (2011)
The percentage of websites that fall within a particular Window of Exposure zone (Sorted by industry)

APPENDIX D



Top 10 Vulnerability Classes (2011)
(Sorted by vulnerability class)

APPENDIX E



Overall Vulnerability Population (2011)
Percentage breakdown of all the serious* vulnerabilities discovered
(Sorted by vulnerability class)