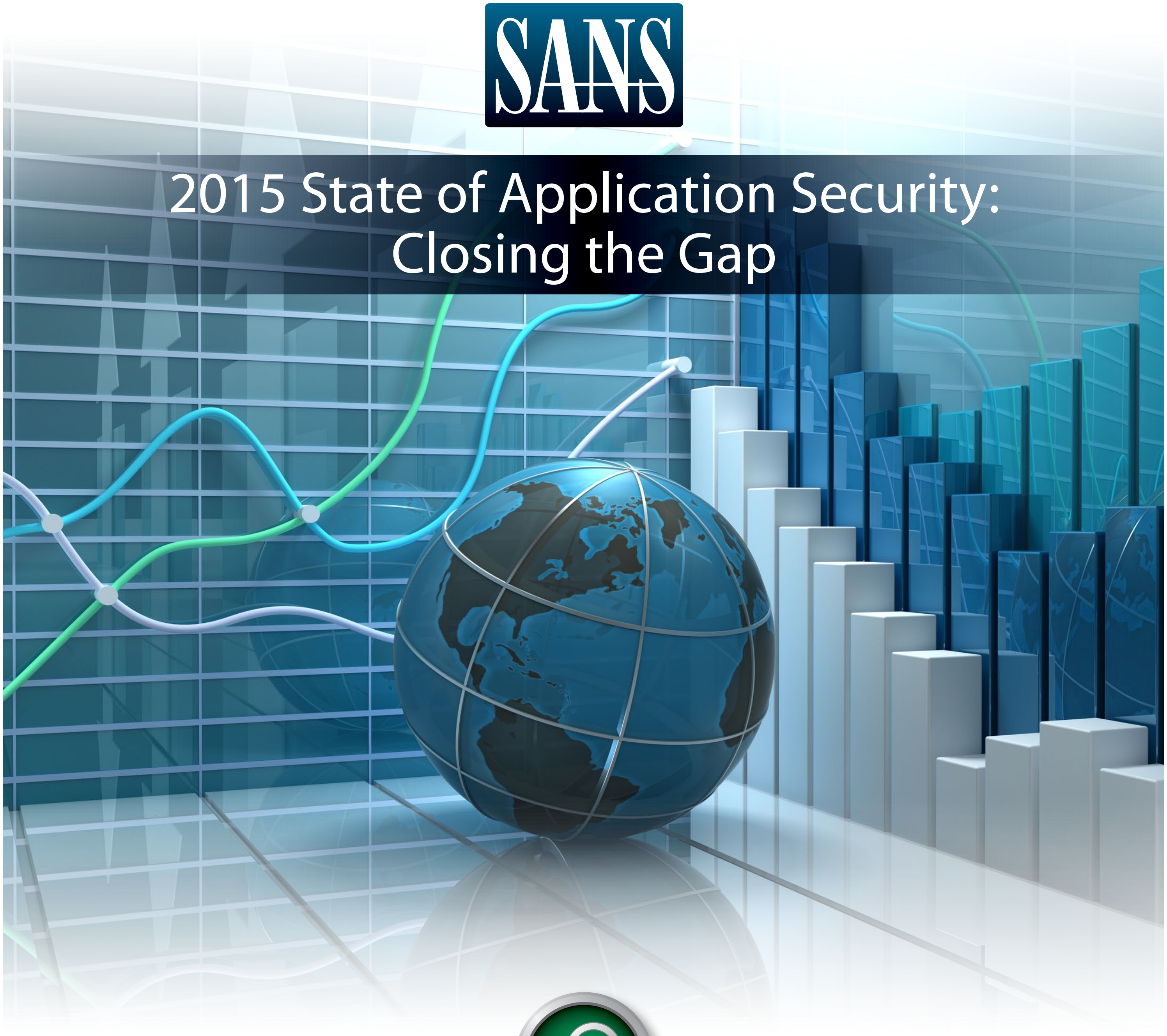

2015 State of Application Security: Closing the Gap

A SANS Survey

Written by Jim Bird, Eric Johnson and Frank Kim

May 2015

Sponsored by
Hewlett-Packard, Qualys, Veracode, Waratek, and WhiteHat Security

Executive Summary

The gap between developers and protectors of applications is closing slightly, according to the SANS 2015 State of Application Security Survey. In this year's survey, 435 qualified respondents answered application security questions from two different perspectives:[1]

- **Builders—**Developers and development organizations—who represent 35% of qualified respondents
- **Defenders—**Security and operations teams responsible for securing applications and running secure systems—who account for 65% of qualified respondents

Security Risk Management Aligned with Development

Defenders *and* builders are focused on where the greatest security risks are today:

apply security resources to public-facing web applications

spend resources on mobile applications

apply resources to applications in private or public clouds

SANS and other institutions have long recognized that these two groups need to climb out of their silos and work more closely together if we're going to build better, more reliable and more secure systems. Thankfully, this change is already occurring.

Because the industry is experiencing so many high-profile application security breaches that result in the compromise of personally identifiable information (PII), builders and their managers are becoming more aware of how important—and how hard—it is to write secure software. Today, application security experts are reaching out to builders and speaking at their conferences. As a result, builders are more aware of risks inherent in the same applications that defenders are concerned with. The most popular application development languages (including Java and .NET) are also recognized as the highest sources of security risk among both groups.

While a closer alignment bodes well for the future of applications, results also show continued gaps between the groups, such as builders putting security off on "someone else" and defenders trying to force security through compliance reviews and penetration testing rather than working with builders to design and build in security from the start.

[1] Taken from the Open Web Application Project's (OWASP's) Builder (developers), Breaker (pen testers) and Defender (infosec) community model: www.owasp.org/index.php/Defenders

The top three challenges for defender teams directly reflect problems that IT security professionals have in engaging with builders:

- Identifying all of the applications in the application portfolio—information that builders could easily provide
- Fear of modifying production code and potentially breaking an app
- Organizational and communications silos between security, application development and the rest of the organization

The top challenges for builders are completely different, and so are their goals and priorities:

- Need to focus on delivering features and on time to market
- Lack of skills or knowledge to build secure software
- Lack of management buy-in or funding

This paper discusses these challenges and how they are made more complicated by the rapidly accelerating pace of development and lack of control over applications hosted in the cloud.

Application Builders and Information Security Defenders

65%

Percentage of respondents who are categorized as defenders based on their work group's role

OWASP (Open Web Application Security Project) has defined communities that bring together experts with the common goal of advancing the state of application security.[2] This approach allows similar groups of professionals and experts to tackle security problems with the involvement of the most relevant stakeholders. SANS decided to look at the respondents to the 2015 survey in light of these communities—specifically defenders (roles that involve security management, compliance, evaluation or operations) and builders (architecture, development or design).

We compared respondents' primary roles in the organization with whether their organization or work group primarily develops applications or manages/secures applications in production. Figure 1 sorts the respondents by their roles and reflects the expectation as to which OWASP community the respondent would belong.

What is your primary role in the organization?

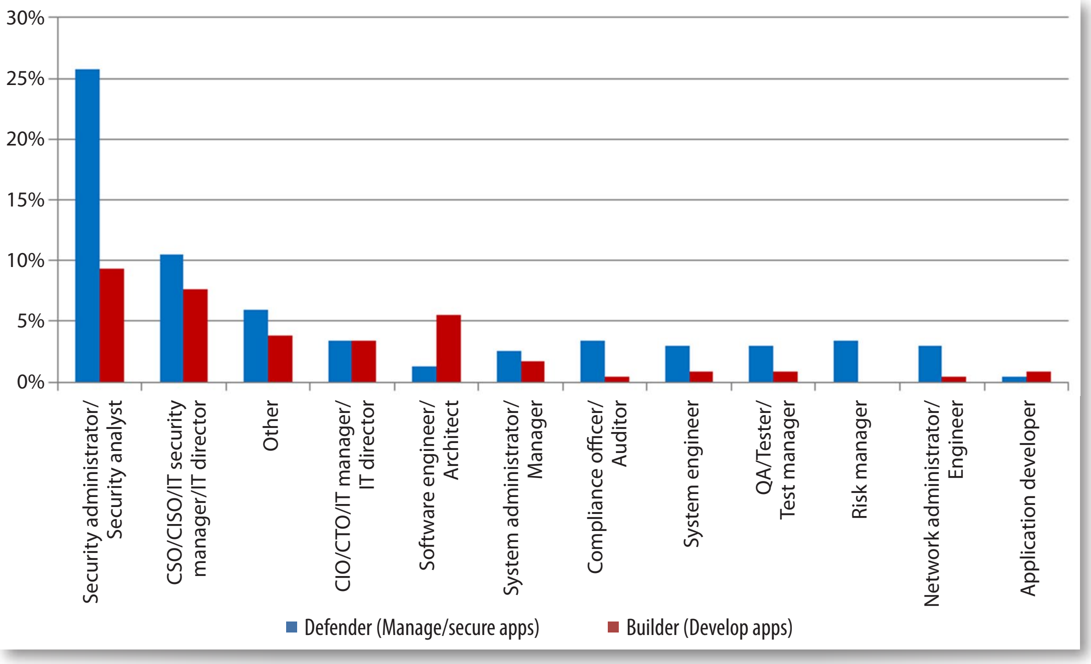

Figure 1. Respondent Roles

[2] www.owasp.org/index.php/Defenders

The 435 respondents who participated in this survey represented a wide range of industries. As in the SANS 2012 and 2014 surveys on this topic,[3, 4] financial services/banking and government led the way (see Figure 2).

What is your organization's primary industry?

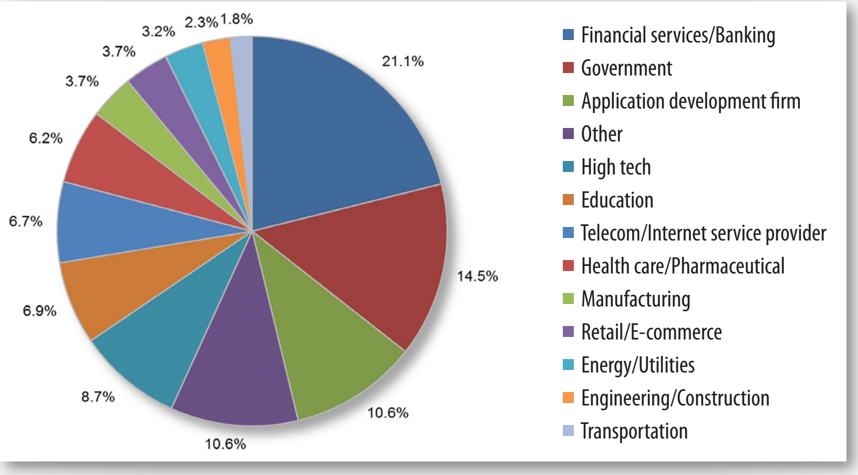

Figure 2. Industry Representation

It is interesting to note that 11% of respondents come from application development houses, up from 6% in 2014, showing the growing need for and awareness of security at the application development level.

The size of respondent organizations followed much the same distribution as in previous surveys, with 28% working in very large organizations of more than 15,000 people and 34% coming from organizations with 1,000 or fewer people, again lending a representative sampling of organizational size to the survey results.

[3] www.sans.org/reading-room/whitepapers/analyst/survey-application-security-programs-practices-35150

[4] www.sans.org/reading-room/whitepapers/analyst/survey-application-security-programs-practices-34765

Challenges Different, Yet the Same

Although results indicate defenders and builders of applications are moving closer, it's clear that these communities and their members aren't always on the same page. Many information security engineers don't understand software development—and most software developers don't understand security. Builders and defenders have fundamentally different drivers. Builders and their managers are focused on delivering features and meeting time-to-market expectations, rather than on making sure that software is secure. So to them, security is "someone else's job." Based on responses to our survey, only a small amount of security testing is done by developers or quality assurance personnel (builders), as noted in Table 1.

Many information security engineers don't understand software development—and most software developers don't understand security.

Table 1. Who tests application security?

Answer Options	Response Percent
Internal security team	83.2%
External security consultants	29.6%
Quality assurance	22.4%
Development team	21.6%
Security-as-a-service providers	15.2%
Business unit owner	11.2%
Our commercial application vendors	5.6%
Other	3.2%

On the other hand, fear of breaking the app and making it unavailable for business use are the top challenges for defenders. See Table 2.

Table 2. Top Challenges for Builders and Defenders

Top Challenges for Builders	Top Challenges for Defenders
Time to market/Deliver features first	Fear of breaking the app when fixing security vulnerabilities
Lack of AppSec skills and tools	Identifying all apps in the portfolio
Lack of management buy-in and funding	Silos between development, security and the rest of the organization

These divergent challenges reveal the training gap on the builders' side, while defenders are challenged with just knowing what apps they have in production. Because defenders are also doing most of the training and evangelizing, it follows that silos would be a concern for them rather than for builders, who still think of security as someone else's job.

The top challenges highlight the problems that builders and defenders have in working together effectively:

- The groups have different priorities.
- Understanding what applications are being used and what the risk profiles are is a critical first step in securing any system. We first identified this problem in our 2012 survey: More than one-quarter of respondents didn't know how many applications their organization used or managed—information that builders could easily provide to defenders and management.[5]
- Defenders and builders, together, don't have confidence in their ability to patch vulnerabilities correctly, test and re-deploy the system without making mistakes. Because builders don't understand security well enough and defenders don't understand software and how it is built well enough, neither group is able to make fixes correctly.
- Organizational and communications silos between security, development and the rest of the organization make communication of risks and threats, training and secure application development more difficult to achieve.

TAKEAWAY:

To break down the communications walls and organizational silos, a number of organizations are adopting collaborative DevOps[6] (and SecDevOps[7]) practices to bring builders, operations and information security together. Groups should be sharing tools and ideas as well as responsibility for building and running systems, while ensuring the availability, performance and security of these systems.

[5] www.sans.org/reading-room/whitepapers/analyst/survey-application-security-programs-practices-35150

[6] To learn more about DevOps, read "The Phoenix Project," a best-selling novel about IT operations http://itrevolution.com/books/phoenix-project-devops-book

[7] To learn more about SecDevOps, check out #SecDevOps on Twitter

Shared Focus: Web, Mobile and Cloud

The emphasis in application security—driven by changing market/consumer demands, escalating threats and evolving ways to manage them—is changing rapidly, so defenders and builders need to be flexible in their approaches to secure development and the application life cycle as application uses and delivery change.

Defenders

In our 2014 survey,[8] most organizations focused their application security programs on security risks in web apps (80%), business-critical apps (72%), mobile apps (35%) and legacy software (24%). Because most business-critical apps are web or legacy apps, that option was not included in the 2015 survey. Today, 79% of defenders still see public-facing web applications as the key focal point for their application security programs, but mobile and cloud applications have increased in importance, based on where respondents are applying their AppSec program resources, as shown in Figure 3.

Where are your application security management resources being applied?
Select all that apply.

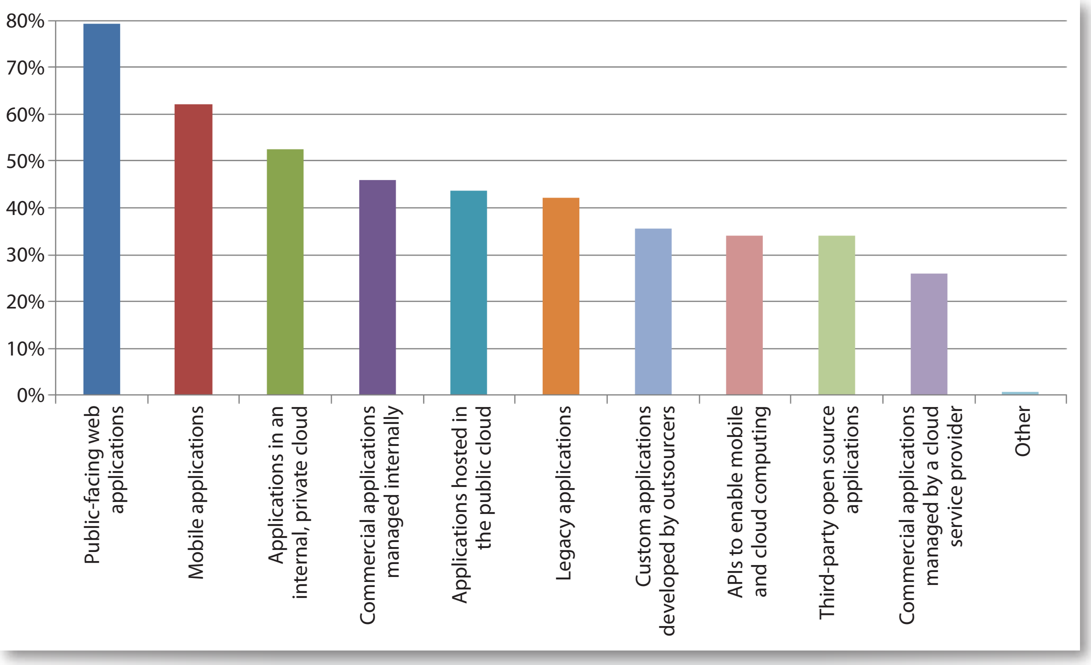

Figure 3. Defenders' Emphasis for Application Security Management Resources

[8] www.sans.org/reading-room/whitepapers/analyst/survey-application-security-programs-practices-34765

This emphasis directly correlates with the growth in the entire web/mobile/cloud ecosystem and its inherent risks. In 2014, web applications were the leading concern (38%); in 2015, public-facing web applications are rated as the major concern by 74% of respondents. Concern over mobile and cloud-based applications both increased from less than 10% in 2014 to dominate the next top spots in 2015. Defenders' concerns about risks are shown in Figure 4.

Which of the following are you most concerned about from a risk and/or compliance perspective?
Select the top three.

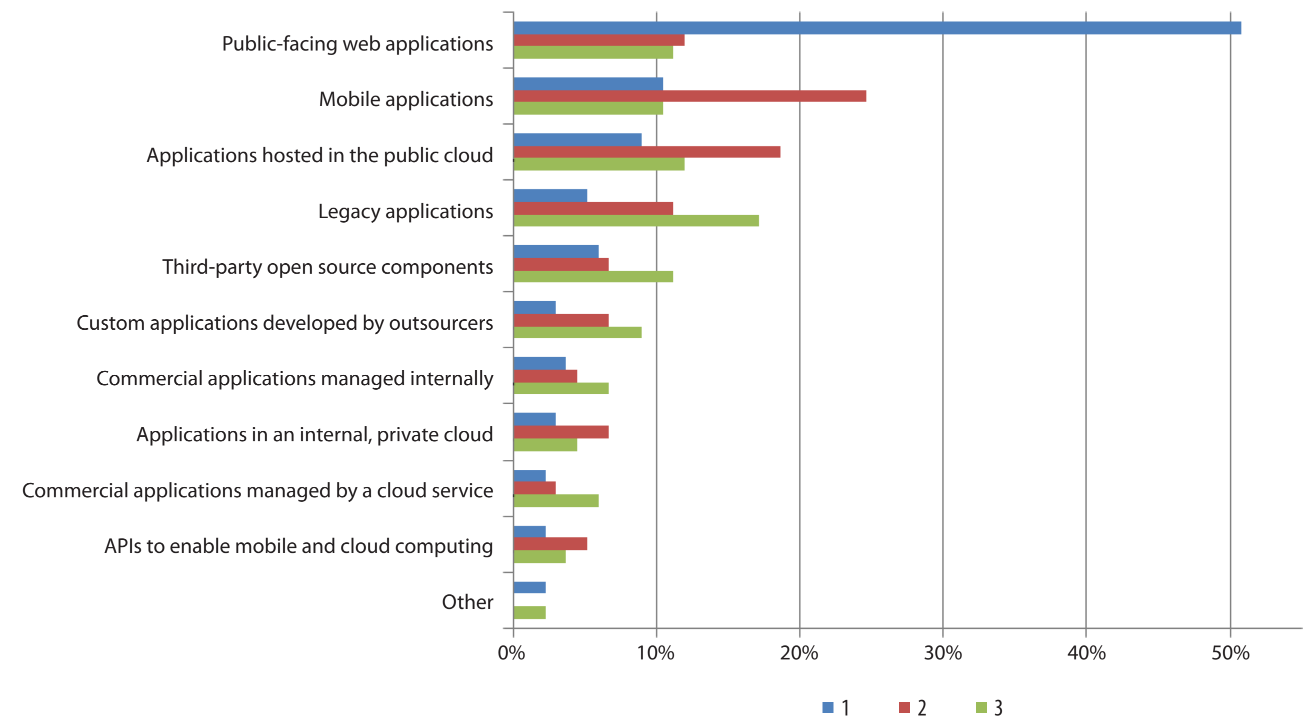

Figure 4. Defender Community Ranking of Application Risks

Builders

Today's builder community is also primarily concerned about the same types of applications the defender community is concerned with: public-facing web apps, mobile apps and cloud-based services. Figure 5 shows that concern over security risk and compliance directly tracks the number of organizations developing those categories of applications. For example, more organizations are developing public-facing web applications, and this category also carries the most concern about development risk.

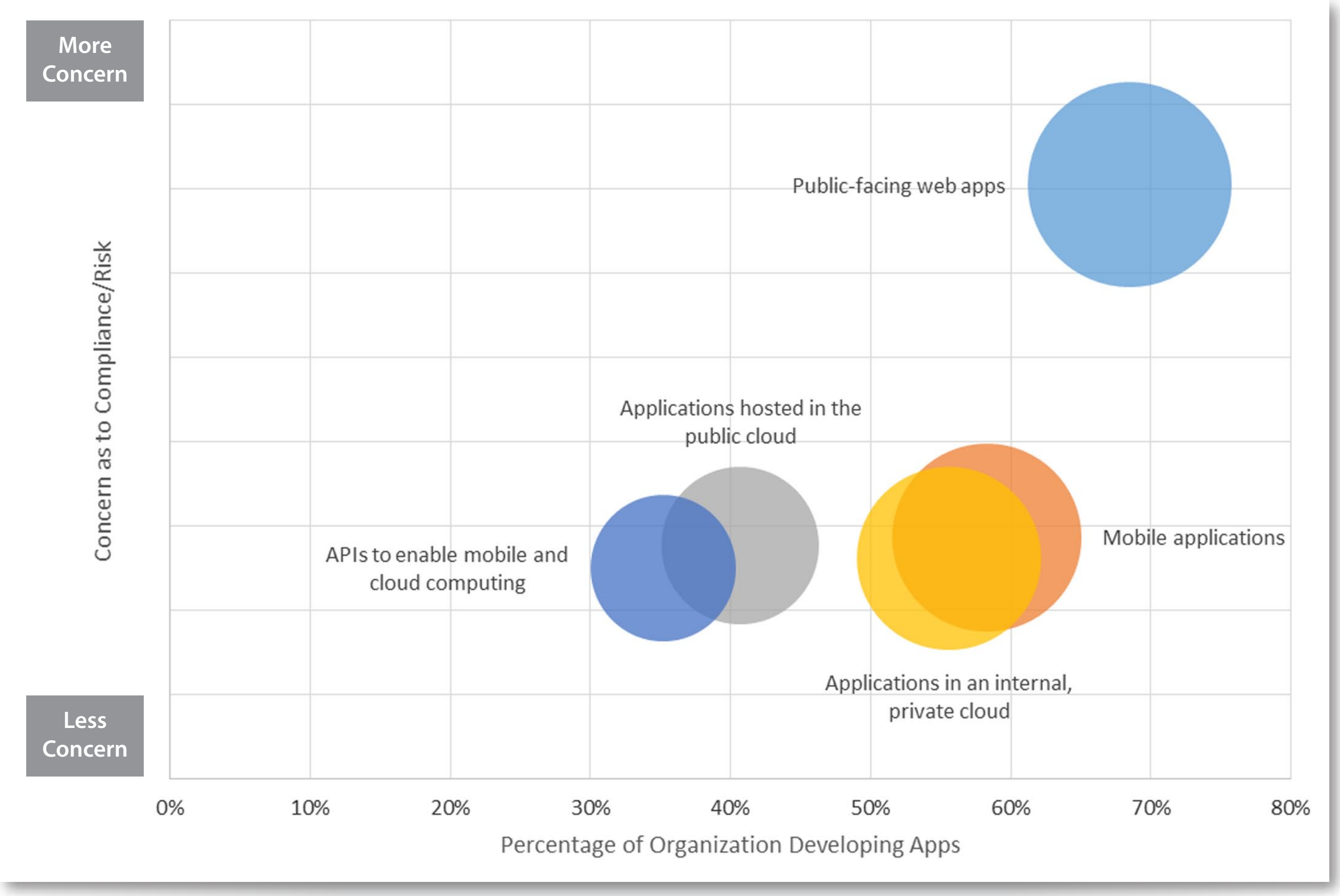

Figure 5. Overlap Between Development and Security Focus

Web, mobile and cloud-based apps are introducing new challenges for builders and defenders: continuously changing requirements, technologies and threats. The rate of change is driving builders to adopt lightweight Agile, Lean and DevOps approaches to deliver software capabilities faster and more frequently. This approach challenges defenders to keep up and change how they work and think.

Languages and Risk

As with application types, the most popular languages are also perceived to have the most security risk. Figure 6 shows that the more-popular programming languages—Java and .NET—are perceived to carry the most risk, even though (and probably because) they are also the most heavily used languages.

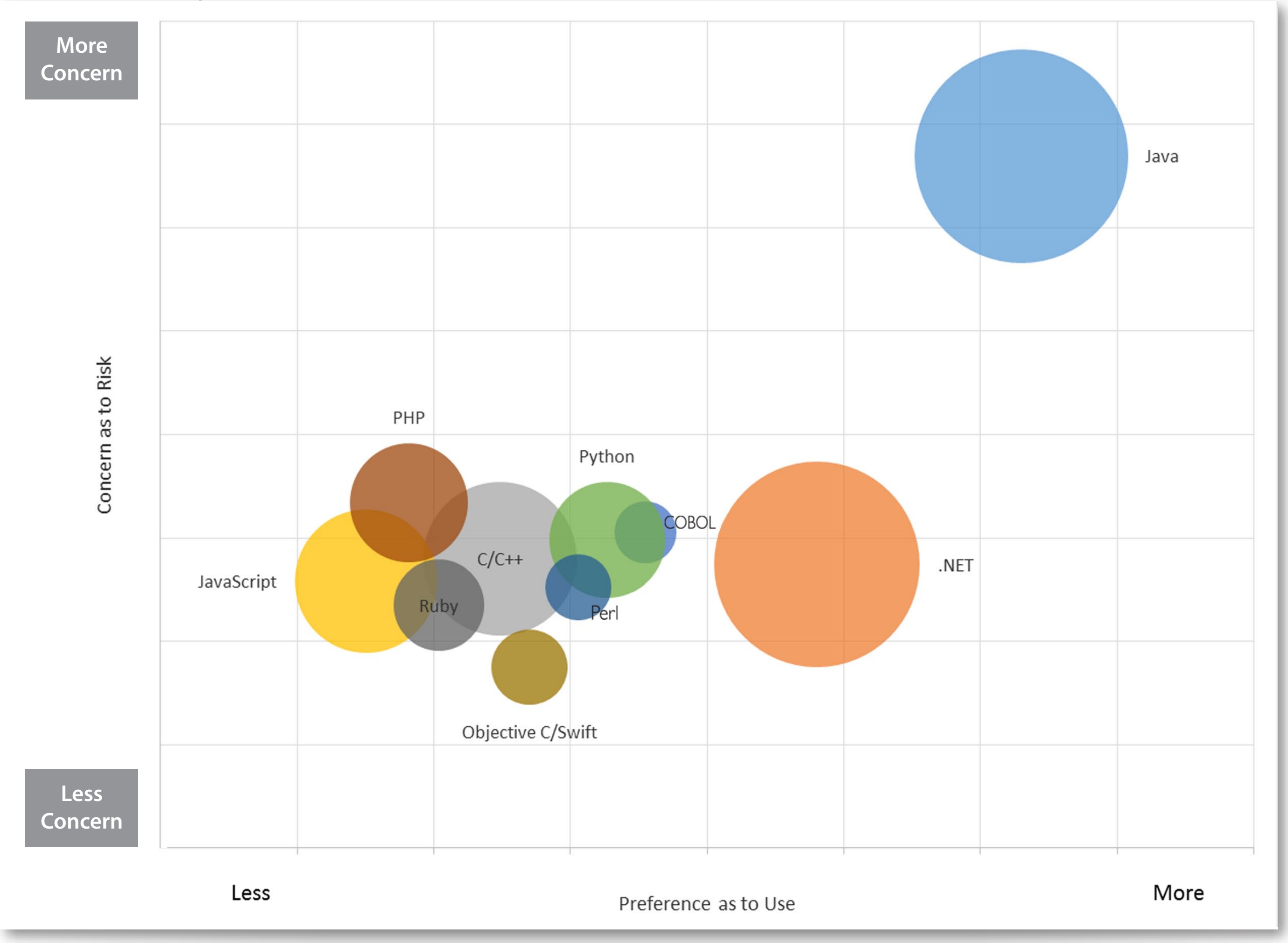

Figure 6. Popular Languages and Their Perceived Risks[9]

Java and .NET both protect developers from some serious security problems (like buffer overflows). Common frameworks (such as .NET MVC; Java application frameworks such as Spring, Hibernate, and Play; and security frameworks such as Spring Security and Apache Shiro) provide additional security protections. The risks arise because these languages are the ones commonly used to build big, feature-rich, business-critical applications with a lot of valuable code, especially legacy code written by developers who didn't understand secure development—code that is exposed to attack.

[9] Note: The size of the circle represents the number of respondents utilizing the language. Java is used by large numbers of respondents who consider it a security concern, whereas COBOL is used by a significantly smaller number who consider the security concerns to be somewhat less.

Application Security Programs

To be effective, application security has to be included throughout the complete development life cycle:

- **Design and build.** Consider compliance and privacy requirements; design security features; develop use cases and abuse cases; complete attack surface analysis; conduct threat modeling; follow secure coding standards; use secure libraries and use the security features of application frameworks and languages.

- **Test.** Use dynamic analysis (DAST), static analysis (SAST), interactive application security testing (IAST), fuzzing, code reviews, pen testing, bug bounty programs and secure component life-cycle management.

- **Fix.** Conduct vulnerability remediation, root cause analysis, web application firewalls (WAF) and virtual patching and runtime application self-protection (RASP).

- **Govern.** Insist on oversight and risk management; secure SDLC practices, metrics and reporting; vulnerability management; secure coding training; and managing third-party software risk.

Highly structured, heavyweight AppSec programs that are oriented toward sequential development (planning ⟶ requirements ⟶ design ⟶ coding ⟶ testing ⟶ deployment) and rely on stage gate approvals must be adapted to the way builders work today: simpler, faster, more agile, more iterative and incremental.

Standards

The OWASP Top 10[10] (a community-driven, consensus-based list of top 10 application security risks, with lists available for web and mobile applications) is by far the leading application security standard or guideline followed by builders who took this survey (see Figure 7).

What application security standards or models do you follow?
Select all that apply.

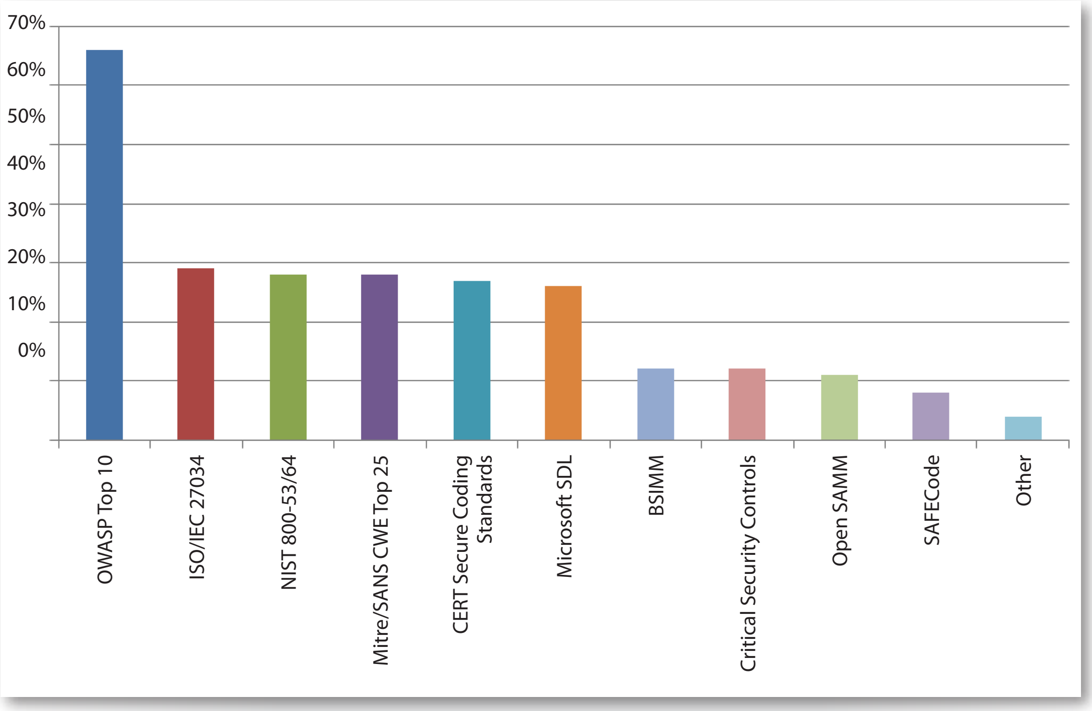

Figure 7. Application Security Standards in Use

There are a few reasons for the overwhelming reliance on OWASP:

- The Top 10 is the shortest and simplest of the software security guidelines to understand (there are only 10 different areas of concern).
- Most SAST and DAST tools report vulnerabilities in OWASP Top 10 risk categories, making it easy to show compliance.
- The OWASP Top 10 (like the Mitre/SANS Top 25[11]) is referenced in regulatory standards such as PCI DSS.

After the OWASP Top 10 comes reliance on much more comprehensive standards, such as ISO/IEC 27034 and NIST 800-53/64 (which are often required in government work), and then the more general coding guidelines and process frameworks such as Microsoft's SDL.

[10] www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

[11] http://cwe.mitre.org/top25

Effectiveness

Unfortunately, when asked, 47% of respondents (representing the majority) felt that the effectiveness of their AppSec programs needed improvement, whether evaluated internally (47%) or in comparison to other organizations (36%). See Figure 8.

How would you rate your organization's current application security program from an internal perspective? Compared to other similar organizations?

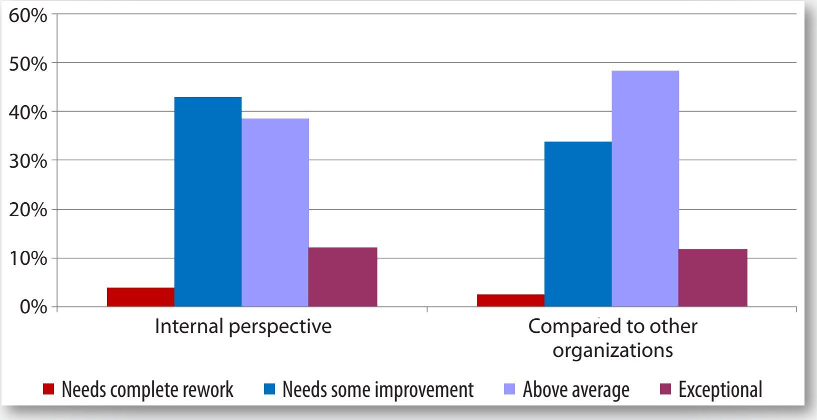

Figure 8. AppSec Programs Need Improvement

When comparing their AppSec program to other organizations, an additional 10% of respondents rated their programs as above average or exceptional. This may be due to raised awareness of publicly reported security breaches, giving some organizations false confidence that at least they are less vulnerable than their neighbors or competitors.

Drivers

Compliance is the major driver for spending on AppSec programs. This is not surprising, given that more than 50% of the respondents are in highly regulated industries including financial services/banking, government, telecommunications, energy and health care. See Table 3.

Table 3. Drivers for AppSec Programs

Drivers for AppSec Programs	Response Percent
Compliance, internal audit or direct response to audit findings	71.5%
Risk-based approach driven by estimating economic impact of breaches	69.6%
Wrapping security in as a standard support item	39.9%
Direct response to a security incident	36.7%
Comparing security spending/ROI/TCO to industry benchmarks	33.5%
Couching security as a direct "enablement" for new applications	30.4%
Other	1.3%

Compliance is closely followed by economic risk considerations: Executive decision makers are coming to understand the high cost of major security breaches caused by insecure software. Other organizations justify application security spending mostly on a reactive basis: as a direct response to security incidents, or by wrapping security into support service levels in response to customer demands. Instead of acting strategically, taking thoughtful and proactive steps to minimize the costs and risks of insecure software in a planned way, some organizations are still waiting to be forced to act, often only after damage has already been done.

Maximizing the AppSec Investment?

When asked, 48% of respondents did not feel that their funding was adequate, 18% had no opinion, 31% felt that their spending was adequate and 3% felt their funding was more than adequate. Figure 9 shows the percentage of their overall IT budgets respondents spend on AppSec.

What percent of your overall IT budget is spent on AppSec?

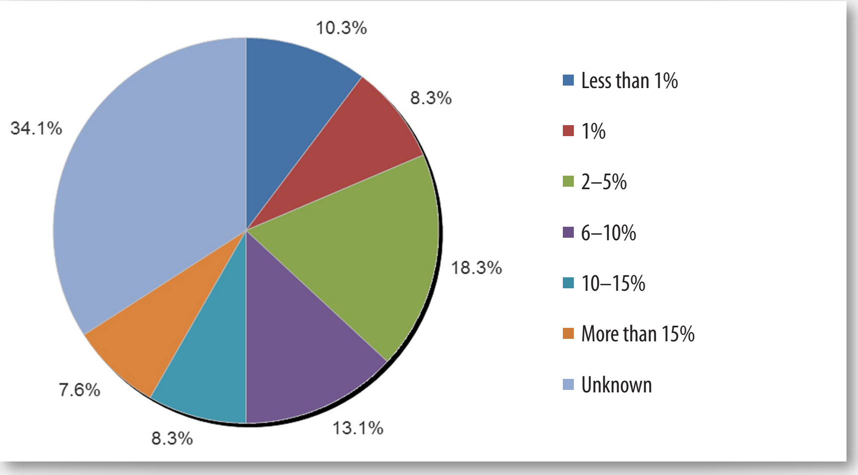

Figure 9. Percentage of Overall IT Budget Spent on AppSec

Tools and training are important and necessary; but improved application security will also require deeper organizational changes—changes in values and responsibilities.

Looking into some of the individual responses in more detail, we uncovered a number of contradictions indicating confusion about where the funding for AppSec comes from and how to measure it, including:

- How people define spending on application security isn't consistent, nor is the understanding of where the work involved in securing applications begins and ends.
- Measuring spending on application security isn't easy. Costs can be spread across IT, information security, software development, quality assurance, risk management, compliance and operations. So, who's really paying for AppSec? How much of general IT security costs or development and QA costs should be included in application security?
- We asked people to report application security spending as a percentage of "your overall IT budget." Some people understood "your overall IT budget" as the budget for the entire enterprise, while others looked at spending only within their department.

Future studies of spending on application security should take these factors into account.

What is clear is that there are no easy answers as to how much organizations should be spending on application security. However, they should probably be spending more than they are today. It's important for managers to understand that spending more money on tools and consulting is not enough. Tools and training are important and necessary, but improved application security will also require deeper organizational changes—changes in values and responsibilities.

Improving AppSec Across the Life Cycle

The majority of builders (59%) are following lightweight Agile or Lean methods (mostly Scrum). Another 14% still use Waterfall, and fewer still use other heavyweight, structured development approaches such as Capability Maturity Model Integration (CMMI).

Many organizations are looking at DevOps (and SecDevOps) practices and approaches to share the responsibilities for making systems secure and functional among builders, IT operations and defenders. This is a radically different way of thinking about and doing application security. Rather than trying to force security externally through pen testing, compliance reviews and stage gates, defenders need to work collaboratively with builders and operations teams to embed iterative security checks throughout software design, development and deployment.

TAKEAWAY:

Wire security into development tools and Continuous Integration and Continuous Delivery pipelines. Build feedback loops between builders, operations and defenders. Work together to continuously review and improve how application security is done.

Results indicate that these approaches are working: At least half of builders start thinking about security early in the development life cycle, during requirements definition and planning. Less than 10% of developers leave security to the end, before release of the system to production. See Table 4.

Table 4.
Security Planning Time Frames

Phase of Development	Response Percent
Planning/requirements phase	53.4%
Design phase	16.5%
During development	14.6%
Before code check in	4.9%
Before release to production	8.7%
Other	1.9%

It is far better to build in security during the development process. Tens or hundreds or thousands of builders and analysts can do a lot more to build secure applications than a much smaller number of defenders can do after an app is in production—as long as builders have the training, tools and time they need to do the job.

On the other hand, defenders need to continue to learn more about how software development is being done today and about how builders think and work. They need to learn about Scrum, BDD, TDD, Kanban, Continuous Flow, DevOps, Continuous Delivery and Continuous Deployment—and where security fits. They need to consider what security means for the cloud and mobile application development—and for the next new focus: the "Internet of Things."

Application Practices

Defenders take a more holistic view of application security than builders, who tend to be more focused on code development. We asked both defenders and builders to rank their AppSec practices. Defenders focus on the most useful practices in securing their production applications, whereas the builders focus on the practices they use in their development operations.

Defenders

For application defenders, penetration testing ranks high on the effectiveness of security controls and practices for web and mobile apps, while security training ranked highest for internal apps, as shown in Table 5.

Table 5. Useful Security Practices for Application Defenders

Most useful security practices	Internal Apps	Web Apps	Mobile Apps	Cloud Services
Penetration testing	54.2%	67.2%	42.7%	29.0%
Application security training	61.8%	47.3%	22.9%	20.6%
Identity/Access controls	56.5%	47.3%	32.8%	32.8%
Dynamic analysis (vulnerability scanning)	45.8%	54.2%	27.5%	19.8%
Application firewalls/Virtual patching	35.1%	48.1%	18.3%	16.0%
Compliance reviews or audits	47.3%	48.9%	26.0%	26.7%
Code review	43.5%	42.7%	25.2%	11.5%
Threat modeling	31.3%	34.4%	21.4%	16.0%
Static analysis (source or binary)	28.2%	30.5%	23.7%	6.9%
Other	3.8%	6.9%	4.6%	4.6%

Although we would like to think that cloud and mobility technologies would drive the need for more training and for better identity and access controls built into applications, responses show that only 33% overall consider identity/access (IAM) controls most useful for those purposes. Even fewer consider training most useful.

Builders

Risk assessment is also a leading AppSec practice for builders of applications. In fact, it is the top practice for all applications except web applications, where it closely follows penetration testing. See Table 6.

TAKEAWAY:

Builders and defenders need to work together to evaluate new and innovative technologies that could potentially solve multiple challenges in their AppSec programs.

Table 6. Builders' AppSec Practices

AppSec Practice	Internal Apps	Web Apps	Mobile Apps	Cloud Services
Risk and threat assessment	70.0%	64.0%	41.0%	28.0%
Penetration testing	50.0%	67.0%	32.0%	26.0%
Secure deployment standards and review	44.0%	54.0%	25.0%	19.0%
Dynamic analysis (vulnerability scanning)	45.0%	50.0%	24.0%	17.0%
Submit deployment processes for pen testing	36.0%	45.0%	25.0%	10.0%
Static analysis (source or binary)	37.0%	42.0%	24.0%	17.0%
Secure libraries/Frameworks	38.0%	45.0%	19.0%	11.0%
Security assessment of third-party components	26.0%	36.0%	19.0%	13.0%
Application integrity/Binary hardening	23.0%	27.0%	18.0%	6.0%
Virtual patching	14.0%	15.0%	11.0%	4.0%
Other	2.0%	4.0%	2.0%	1.0%

Third-Party Risks

Survey results indicate that IT security is not consistently engaged in procuring and contracting for new applications, and it is often not performing risk assessments ahead of procuring new apps. Almost 32% (the largest group) are involving IT in assessment of those apps "most of the time" based on the criticality of the app, while 26% are doing so all the time, and 17% do so frequently but not regularly. The rest are testing either infrequently, rarely or never.

The problem is: Even those who involve IT aren't doing thorough assessments. According to the survey, the primary method organizations are using to manage the security risks of their commercial apps is vendor attestation (see Figure 10).

Which of the following describes how your organization manages security risks from third-party applications purchased from software vendors?
Select the most appropriate.

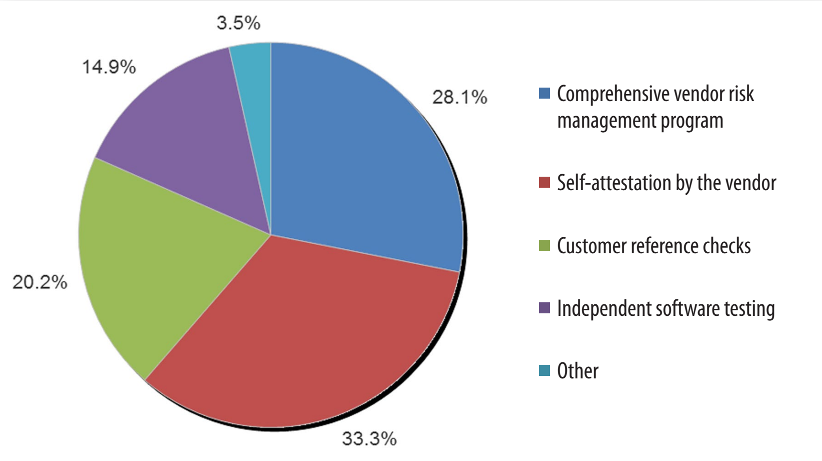

Figure 10. Managing Security Risks in Third-Party Applications

This is a serious concern, given the extent to which organizations depend on the components that builders use in their apps. Most (79%) of responding builders rely on open source or third-party software libraries in their applications.

A 2012 study of open source software use in Global 500 organizations found that a majority of organizations regularly download software components and frameworks with known security vulnerabilities, even if newer, patched versions of the components or frameworks were available.[12] HeartBleed, ShellShock, POODLE[13] and now FREAK, as well as the long, continuing string of serious vulnerabilities in the Java runtime, show the dark side of using open source components.

[12] www.cio.com/article/2397662/governance/do-insecure-open-source-components-threaten-your-apps-.html

[13] www.scmagazine.com/heartbleed-shellshock-and-poodle-the-sky-is-not-falling/article/379301

Testing and Development

The speed and frequency of security testing continues to increase year over year. In our 2012 survey, 23% of organizations tested their production apps on a continuous or near-continuous basis. This increased to 36% in 2014, and then to 40% this year. And, less than 10% leave testing until initial launch. See Table 7.

Table 7. Frequency of Security Assessment/Testing in Business-Critical Applications in Production *(2012–Present)*

Frequency	2015	2014	2012
Ongoing/Continuous	40.0%	35.6%	23.3%
Once a month	8.0%	8.1%	9.5%
Every three months	14.4%	12.1%	18.0%
Every year	13.6%	19.5%	14.3%
Only before systems are initially launched	7.2%	4.0%	N/A
Only when applications are updated, patched or changed	7.2%	10.1%	21.3%
Based on compliance or internal audit cycles	5.6%	N/A	N/A
When we sense or know there's a problem with the applications	1.6%	3.4%	N/A
We don't assess our applications	0.0%	2.7%	13.5%
Other	2.4%	2.7%	N/A
Whenever we remember to check them	N/A	2.0%	N/A

DYNAMIC ANALYSIS SECURITY TESTING (DAST)

A leading practice for web applications, where a wide selection of well-established, mature testing tools is available. Dynamic analysis testing is becoming a key practice for mobile apps, reflecting improvements in testing technology in the past couple of years.

Pen testing is still seen by organizations (both builders and defenders) as one of the most useful security practices. Usually done as part of a security stage gate check or mandated by regulatory compliance, pen testing and compliance reviews are ways for defenders to force security onto builders from the outside. But pen testing and compliance reviews are expensive and inefficient, and can't be scaled, especially as builder teams adopt Agile or DevOps methods for rapid development and deployment.

Builders are also using automated testing tools or platforms to conduct dynamic analysis and static analysis testing (especially for web applications). Application integrity checking is commonly used as an operational control to check for unauthorized code or configuration changes and is key to regulations such as the Sarbanes-Oxley Act (SOX).

Binary hardening, "tamper proofing" and obfuscating code to make it harder for attackers to decompile and analyze, is also used in some cases. While a determined attacker can still figure the code out, it raises the bar. Binary hardening applies especially to mobile and other remote clients as well as to code deployed to external clouds.[14]

[14] Application hardening is included in the OWASP Mobile Top 10. www.owasp.org/index.php/Mobile_Top_10_2014-M10

Vulnerability Repairs

Vulnerability management is a key area where defenders and builders must work together to identify and repair serious security vulnerabilities as quickly as possible. Builders need to know enough about security to understand the vulnerability and fix the code properly, test for regressions, and build and deploy the fix quickly. Even more importantly, they need to be able to go back and address the root cause. Otherwise, they will be stuck in a vicious and dangerous cycle, finding and fixing vulnerability after vulnerability, without improving or learning, never knowing when or if this cycle will end.

Defenders and builders must work together to identify and repair serious security vulnerabilities as quickly as possible.

In the survey, 26% of defenders took two to seven days to deploy patches to critical apps in use, while another 22% took eight to 30 days, and 14% needed 31 days to three months to deploy patches satisfactorily. See Figure 11.

On average, how long does it take for your organization to fix and deploy a patch to a critical application security vulnerability for systems already in use?

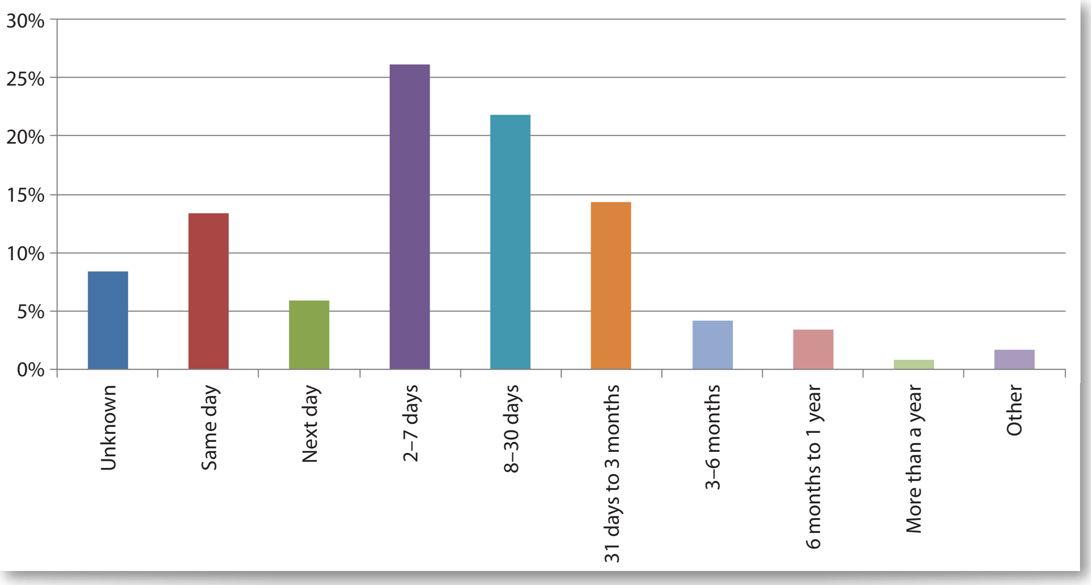

Figure 11. Finding and Fixing Vulnerabilities Takes Time

As to how discovered vulnerabilities are repaired, 63% of defenders report making root cause repairs using secure development life-cycle practices, while 51% update the environment to compensate. The third largest group, at 49%, is troubling: In many cases, vulnerabilities in production apps are patched through quick-and-dirty fixes or other short-term workarounds, such as disabling a feature or function in the app (see Table 8).

Table 8. How Vulnerabilities Are Fixed

Fixed at root cause through secure SDLC practices	63.0%
Update operating environment, network architecture, other protection mechanisms	51.3%
Fixed with a quick-and-dirty software patch	48.7%
Upgrade third-party or open source software component	44.5%
Disable feature or function of application	36.1%
WAF/Virtual patching	31.9%
Rely on container security through RASP (Runtime Application Self-Protection)	12.6%

Some organizations (32%) are also relying on a compensating control such as a WAF with virtual patching, or on container security through RASP.[15] These controls can be useful for repairing vulnerabilities when the code is not available, for protecting apps until patches can be worked into the repair cycle, and for automatically protecting apps from certain types of vulnerabilities at runtime.

Continuous Delivery—building automated testing and deployment pipelines from development to testing and through production—is one way for organizations to address the challenges in fixing vulnerabilities quickly and with confidence.[16]

[15] RASP is an emerging technology that adds protection for some kinds of attacks in the run-time container (currently, the Java JVM and/or .NET CLR).

[16] Refer to Nick Galbreath's presentation "Fixing Security by Fixing Software Development using Continuous Deployment," www.client9.com/post/2013-05-14-fixing-security

Training

Training builders on how to develop secure software is a fundamental part of any AppSec program—and is key to helping bridge the gap between defenders and builders. Custom, specialized knowledge on how the system works must be shared effectively within teams. A recent study by the Software Engineering Institute explains why:[17] Builders can't prevent or fix security problems if they don't understand application security concepts, tools, and language-specific and platform-specific security concerns.

- They need to know what to look out for when writing code and when changing or reviewing other people's code.
- Once they understand application security problems, builders can act on this knowledge to change how they develop applications and take responsibility for adding security into the development life cycle.

Training Effectiveness

The majority of respondents (70%) are offering AppSec training in the form of secure code training to their developers. The level of training, however, varies widely, with 73% relying on informal on-the-job mentoring and 62% on short, computer-based awareness training. Only 39% of respondents are using instructor-led, in-person training, followed by 32% with hands-on but virtual training and 28% with in-depth prerecorded instruction. See Table 9.

Table 9. Effectiveness of AppSec Training Methods

Training Approach	Very Effective	Effective	Not Effective
General AppSec training courses and materials	19.4%	67.2%	11.9%
Optional role-specific application security training	14.9%	68.7%	9.0%
Mandatory role-specific security certification	23.9%	47.8%	14.9%
Expert coaching	19.4%	47.8%	17.9%
Rely on third-party developers to keep up to date	9.0%	37.3%	35.8%

Informal and inexpensive general-purpose training and on-the-job mentoring are useful in building security awareness. However, these approaches are not as effective as targeted role-specific training with formal certification, rated by 24% as ″Very Effective" and by another 48% as "Effective." Indeed, all of the best rankings involve some sort of in-person component. Relying on third-party developers to keep up to date on their own ranked as the least effective approach. It's not enough to let builders—especially third-party developers—try to keep up to date on their own or through informal processes that are neither scheduled nor embedded into their processes.

[17] "Predicting Software Assurance Using Quality and Reliability Measures," page 11. http://resources.sei.cmu.edu/asset_files/TechnicalNote/2014_004_001_428597.pdf

TAKEAWAY:

The only practical way that application security can scale and security teams can succeed is by making sure that everyone involved in application development understands software security and can contribute to building secure software.

Training the Builders

Training builders in secure development is important to bridging the gap between builders and defenders. But don't stop at developers. Testers, business analysts, product owners, project managers and Scrum Masters all need training as well. They don't necessarily all have to understand all of the technical details of securing software, but everyone who is involved in developing software should, at a minimum, understand the fundamental security risks and issues in application development and what their roles and responsibilities are.

For example, with Adobe's internal karate belt approach to certification in secure software development,[18] everyone on the team holds at least a white or green belt, showing that they have attended basic training in secure development. In addition, every project has a brown belt- or a black belt-certified developer who provides security leadership and mentors and supports the rest of the team.

[18] www.adobe.com/content/dam/Adobe/en/security/pdfs/adobe-security-training-wp-web.pdf

Conclusion: What Does the Future Hold?

Organizations that develop and manage applications plan to invest more in mobile, cloud and virtualization projects over the next two years, according to write-in responses to the last survey question. The technology in these areas is new and rapidly changing—and so is the threat landscape.

Based on their future spending trends, the majority of organizations seem committed to making security more of a priority. More than half of respondents expect spending on AppSec programs to increase over the next year (and more than a quarter expect spending to increase significantly). Only 3% expect to spend less on application security in the future, as illustrated in Figure 12.

How do you expect your application security spending to change in the next year?

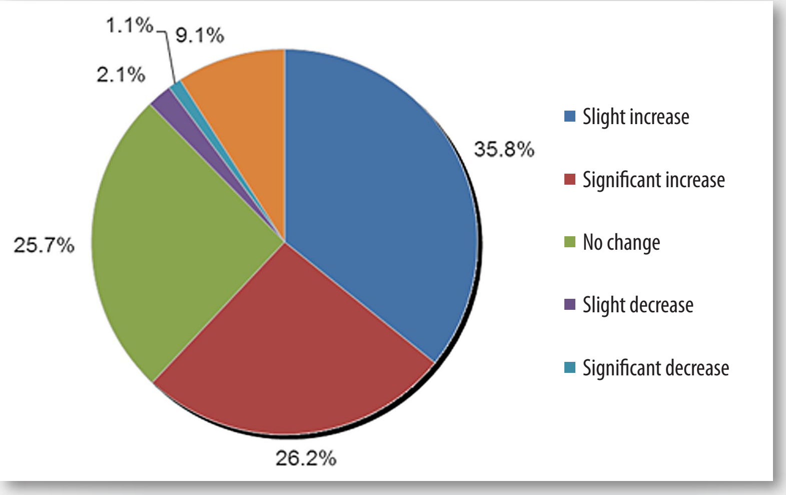

Figure 12. Future Application Security Spending

The technology organizations are investing in is new and rapidly changing—and so is the threat landscape. Both builders and defenders of apps are well aware that these new types of applications—and the languages and frameworks they are developed in—pose substantial, complex risks that cannot be improved immediately.

The overall results of this survey are promising: The gap between defenders and builders is closing, and they share a common goal of eliminating risk from their processes. However, there is still much work to be done at many levels. Management needs to walk the talk and provide developers with the time, tools and training to do a proper job of building secure systems. Builders need to understand that they share important responsibilities for security—it isn't just someone else's job. And, defenders need to understand and adapt to the ways development is changing and accelerating.

About the Authors

Jim Bird, SANS Analyst and lead author of the SANS Application Security Survey series, is an active contributor to OWASP, and a popular blogger on Agile development, DevOps and software security at "Building Real Software." Currently the co-founder and CTO of a major US-based institutional trading service, where he is responsible for managing the company's technology organization and information security program, Jim is an experienced software development professional and IT manager, having worked with high-integrity and high-reliability systems at stock exchanges and banks in more than 30 countries. He holds PMP, PMI-ACP, CSM, SCPM and ITIL certifications.

Eric Johnson, the Application Security Curriculum product manager at SANS, is the lead author and instructor for DEV544 Secure Coding in .NET, as well as an instructor for DEV541 Secure Coding in Java/JEE. A senior security consultant at Cypress Data Defense, Eric's experience includes web and mobile application penetration testing, secure code review, risk assessment, static source code analysis, security research and developing security tools. He currently holds the CISSP, GWAPT, GSSP-.NET and GSSP-Java certifications.

Frank Kim is CISO at the SANS Institute, where he leads the security risk function for the most trusted source of computer security training, certification and research in the world. He is a SANS certified instructor who helps shape, develop and support the next generation of security leaders through teaching, developing courseware, and leading the management and software security curricula. Frank previously served as executive director of Cyber Security at Kaiser Permanente, the nation's largest not-for-profit health plan and an integrated health care provider. Frank was a two-time recipient of the CIO Achievement Award for business-enabling thought leadership.

Sponsor

SANS would like to thank this paper's sponsors:

hp

QUALYS®
CONTINUOUS SECURITY

VERACODE

waratek

WhiteHat
SECURITY

Last Updated: October 29th, 2015

Upcoming SANS Training

Click Here for a full list of all Upcoming SANS Events by Location

SEC567: Social Engineering	**Herndon, VAUS**	**Nov 09, 2015 - Nov 10, 2015**	**Live Event**
SANS South Florida 2015	**Fort Lauderdale, FLUS**	**Nov 09, 2015 - Nov 14, 2015**	**Live Event**
SANS London 2015	**London, GB**	**Nov 14, 2015 - Nov 23, 2015**	**Live Event**
Pen Test Hackfest Summit & Training	**Alexandria, VAUS**	**Nov 16, 2015 - Nov 23, 2015**	**Live Event**
SANS Hyderabad 2015	**Hyderabad, IN**	**Nov 24, 2015 - Dec 04, 2015**	**Live Event**
SANS Cape Town 2015	**Cape Town, ZA**	**Nov 30, 2015 - Dec 05, 2015**	**Live Event**
SANS San Francisco 2015	**San Francisco, CAUS**	**Nov 30, 2015 - Dec 05, 2015**	**Live Event**
HIMSS	**Boston, MAUS**	**Dec 01, 2015 - Dec 02, 2015**	**Live Event**
Security Leadership Summit & Training	**Dallas, TXUS**	**Dec 03, 2015 - Dec 10, 2015**	**Live Event**
SANS Cyber Defense Initiative 2015	**Washington, DCUS**	**Dec 12, 2015 - Dec 19, 2015**	**Live Event**
SANS Las Vegas 2016	**Las Vegas, NVUS**	**Jan 09, 2016 - Jan 14, 2016**	**Live Event**
SANS Dubai 2016	**Dubai, AE**	**Jan 09, 2016 - Jan 14, 2016**	**Live Event**
Cyber Defence Delhi 2016	**Delhi, IN**	**Jan 11, 2016 - Jan 22, 2016**	**Live Event**
SANS Brussels Winter 2016	**Brussels, BE**	**Jan 18, 2016 - Jan 23, 2016**	**Live Event**
SANS Security East 2016	**New Orleans, LAUS**	**Jan 25, 2016 - Jan 30, 2016**	**Live Event**
SANS Sydney 2015	**OnlineAU**	**Nov 09, 2015 - Nov 21, 2015**	**Live Event**
SANS OnDemand	**Books & MP3s OnlyUS**	**Anytime**	**Self Paced**