A GUIDE TO

APPLICATION SECURITY

Security Compass

Table of Contents

A Guide to Application Security

Now that every business is a digital business, enterprises that turn application development into a core competency will gain a significant strategic advantage. This isn't news: even companies in the most conservative sectors, such as finance and hospitality, are staffing up to launch their own digital transformations. If last year had a business motto, it would be Develop or Die.

Speed is everything. Hiring and equipping an application development team is a big expensive effort, and businesses want to see a return on that investment as soon as possible. In that push to the next peak, security is often left till last and if there are delays in the development phase, the security team may not have enough time to do their work rigorously before the new features are launched.

In a business environment where the only headlines that make the general news are those about breaches and ransoms, that security-last approach to application development is hard to understand. Despite the clear danger, decision-makers tend to think current development and security practices meet the status quo for accountability and, in most cases, compliance (although that's changing).

The very structure of most organizations increases the security gap: security and development are separated inside large organizations and don't have an effective means of communicating with each other; in practice, that means the security team can make all the policies it wants, but there's no way to tell if development teams have procedures in place to follow those policies.

WHAT THE BOARDROOM NEEDS TO KNOW

92% of reported vulnerabilities are in applications, not insecure networks

There's a lot of confusion in corner offices and boardrooms about security. That's not an insult: security is hard, as every CISO would agree. A lot of very smart people think that security is an impenetrable mystery and, besides, isn't the current security environment good enough already?

Many decision-makers think cybersecurity is synonymous with network security. Those non-technical executives need to understand that the costliest, most up-to-date network security can't stop a bad guy from coming through a door left wide open by insecure code in a proprietary application. Just ask Equifax.

Best practice is to build it in

Most businesses build applications and then test them for vulnerabilities. This creates a lot of problems. First, it takes more time, which costs more money. It delays release cycles, so it erodes competitive advantage. And it leaves numerous vulnerabilities undetected, so it's a wasted effort.

The reason vulnerabilities go undetected is because code scanners are not 100% effective. In fact, we believe that code scanners only detect about 50% of vulnerabilities. Code scanners perform based on the assumption that code is executed predictably from start to finish. But that doesn't always happen. For example, when an array out of bounds error occurs, the test would have to extend beyond the buffer to prove true. Therefore, the program will actually not halt but will continue to run. Code scanners are also optimized for certain types of vulnerabilities, so they can't find what they're not looking for. The result of these and other limitations is that code scanners, whether static or dynamic, produce a lot of false positives and false negatives.

Level up your dev teams with tools and training

Application security needs to be integrated into the development process itself. This can be a hard case to make. Development teams are optimized to deliver value to their business, and asking them to change their process for the sake of security can be a battle.

Developers don't understand security all that well; it's not what they're trained to do. If they don't understand why they have to perform security-related tasks, they will perceive them as bottlenecks that can cause them to miss release dates.

Developers need help understanding that security processes slow down progress because they're not built into the process and because the developers aren't provided with the training they need to incorporate security into their work seamlessly. For an educated developer, security is just a normal part of producing good code.

The security team should be responsible for giving developers everything they need to build security controls and mitigation right into the SDLC workflow. Look for a tool that solves the collaboration problem. The security team should be able to communicate with the development team throughout the day in a way that doesn't cause friction. Training that is easy to consume should be provided, such as short focused video tutorials. A positive incentive program to encourage developers to produce rugged code will help build enthusiasm for the new processes.

YOUR QA TEAM IS A SECURITY RESOURCE

Your QA team is already writing scripts to test if the system is working as intended. They can support application security by writing scripts to see what happens if the system is used in ways not intended.

The security team should provide QA with an analysis of the application's possible vulnerabilities so they can be targeted directly.

Map security requirements to business requirements

The process of defining requirements is a chance for a CISO to take on a stronger and more strategic role. A CISO that manages risks strategically, works well with department heads across the organization, and promotes a culture of shared risk ownership will be seen as an enabler rather than a policeman.

Application security goals must be mapped to the priorities. For instance, if speeding development time is a priority, the security team will have to work with the development managers to make training available, tailor the workflow to include security activities, and add time for security testing into the schedule.

Defining security requirements will require a look beyond application security to the entire security environment. Controls that support application security, such as the proper use of security headers, should be in place where they make sense. Tools and services that were purchased in bundles may include forgotten application security tools that could be put to use in the new program.

TYPICAL PRIORITIES

- Understanding application risks
- Lowering application security risks
- Speeding up development time
- Lowering cost of risk remediation
- Improving compliance
- Increasing security of third-party software

See what is prioritized in the security budget as a whole. If application security is getting a tiny sliver of the budget but the enterprise's long-term strategy is to push out a lot of applications, it's time to rebalance.

Know which assets are most frequently attacked. Some will require more hardening than others. The results can be surprising; sometimes assets of the lowest priority will be the most frequently targeted, since attackers expect them to be the least protected.

The IT team must provide an inventory of its own assets, including those hosted elsewhere or owned by third parties. There needs to be a verified point of contact for all assets so that the security team can perform due diligence.

Prioritize the inventory by starting with web-based assets. Look for those that, if compromised, would allow an attacker to pivot and get into other systems. Check credentials, make sure all default admin credentials are changed, and remove excess privileges from users that don't need them.

Test the production environment at least as often as the application itself is updated. Hackers are going after the tools developers use as a way to get into other systems. Actually, it's not just hackers who are using this tactic: in 2016, the CIA allegedly tried to compromise Apple's development software, Xcode, in order to insert backdoors for surveillance into any devices using the tools.

Automate to build security into the SDLC

As digital business becomes the norm, old models for implementing security controls are far too slow to keep up with modern demands. Businesses that want to increase their development velocity without sacrificing code quality are fading out Agile and embracing DevOps.

Security teams working in companies with Agile environments do not usually get to see the output of a sprint until the sprint is over. Businesses that make security a separate last step do so because they lack security skills on their dev team, which is the norm, and their security teams do not have the time or ability to become involved during the development stage. By the time the security team is allowed to work with the code, time to launch is short and the pressure is intense to approve the code before a deadline is missed. Fixing vulnerabilities after the fact is a reactionary posture, and that's the opposite of a best practice.

WHAT COMPANIES BELIEVE

25% **of companies say applications are the source of their breaches**

WHAT COMPANIES (DON'T) DO

51% **spend >1% of their IT budgets on app security**

Source: Forbes

DevOps produces greater efficiency by reducing handoffs between the developers, operations team, security team, and customers. Automation is the chain that links the groups together. When security processes are automated, security becomes an inherent component of the practice of continuous integration (CI), so code can be regularly tested not only for quality assurance, but for vulnerabilities as well. Automation can also be used to build security tools and training into the development process early in the SDLC. These proactive stances make it possible for organizations to prevent attacks, rather than just defend against them.

Rules for risk

Security teams tend to focus on defenses such as firewalls and ATP solutions, but can lose sight of the underlying reason these things were purchased: security is important, but profitability is supreme.

Company leaders bring knowledge of business drivers to the table, and security executives bring an understanding of the current threat profile. There also needs to be a seat for the legal and compliance SMEs.

The end result should be a holistic view of each application's security status, including associated risks, countermeasures to mitigate those risks, a plan for downtime and emergency shutdowns, and a plan to improve at least one or more of these elements.

BASE RISKS ON DEFINED KPIS

- Number of vulnerabilities present in an application
- Time to fix vulnerabilities
- Remediation rate of vulnerabilities
- Time vulnerabilities remain open

Practice threat modeling

Know your enemy is a security fundamental. Threat modeling is a methodical way of looking at an application from an attacker's perspective. The results can guide security decisions based on which threats are considered most critical. With that awareness, an acceptable level of risk, which balances security and spend, can guide decisions about how much security is "enough" security.

Most organizations understand their threat profile at an intuitive level. That's a nice way of saying they don't document it. But attackers come and go, and motives and goals are always in flux. Turning threat modeling into a semi-regular activity with a formal process is a good idea.

Plan to change the plan

Security is not a one-and-done proposition. That's something the security and development teams have in common: they both need to continually improve their skills, processes, and tools.

Business priorities and software technologies change all the time, so last year's application security program may not be fully effective next year. Making big changes to any program is hard to do; everyone may agree the update is important, but it's never as important as the crisis that's looming at the moment.

The security team can mitigate this to a point by taking a leaf from the developers' book and practicing iterative improvements. If a big strategic shift occurs at the organizational level might require a good deal of change, but the hardest pieces of an application security program—getting the developers onboard and giving them training and tools, establishing security goals and requirements, and creating open lines of communication between security and development—are static. Focus iterative improvements on these areas and the strategic shifts later will be much easier to implement.

Test early, fix on the fly, and release with confidence

The later in the development cycle that a problem is discovered, the longer and harder it is to fix it. Most companies don't find vulnerabilities until the end of the cycle, when it's too late to have many options: the menu of choices is to delay the launch, launch a vulnerable application, or pay overtime for the developers to work into the night. Sometimes, the choice is to do all three.

When organizations have a process in place to expose problems earlier in the cycle, they can do more testing. Issues are discovered in time to fix them before the clock runs out. Vulnerabilities can be managed in the same way bugs are, which is something every development team is comfortable with already.

When organizations don't build security into the development process, they rely on firewalls and scanners to protect their assets. If they find issues, they fix them. That's a reactive approach. If they don't find issues, that doesn't mean there aren't any issues to worry about—it's just means they didn't find them. They don't really have an application security program in place at all.

By making an application secure as possible in the development stage, an organization gains the benefit of a holistic approach. Applications are built faster with fewer security vulnerabilities, communication between development and security is more open and complete, and levels of risk are known and accommodated before code is deployed.

An efficient application security program can save thousands of dollars in remediation and significantly reduce the risk of costly breaches. An enterprise that wants to remain competitive in a business environment where every company is a software company, application security programs are not in the nice-to-have bucket: they are mandatory.

SecurityCompass

Security Compass believes in a world where people can trust technology, and it is our mission to help customers proactively manage cybersecurity risk, without slowing down business. Our holistic program offerings are tailored to your organization's needs. Through advisory services, training, and SD Elements, our award-winning policy to procedure platform for security and compliance, we set up your company with all of the resources and tools it needs to develop secure software. Security Compass serves some of the world's largest enterprises, including 16 of the largest financial institutions in North America, South America, and Asia, as well as 4 of the largest tech companies in the world.

OFFICES

GLOBAL HEADQUARTERS
1 Yonge Street
Suite 1801
Toronto, Ontario
Canada M5E 1W7

TORONTO
390 Queens Quay W
2nd Floor
Toronto, Ontario
Canada M5V 3A6

CALIFORNIA
1001 Bayhill Drive
2nd Floor
San Bruno, California
USA 94066

NEW JERSEY
621 Shrewsbury Avenue
Suite 215
Shrewsbury, New Jersey
USA 07702

INDIA
#4.07
4th Floor, Statesman House
Barakhamba Road, New Delhi
India 110001

1.888.777.2211
info@securitycompass.com
www.securitycompass.com

@SECURITYCOMPASS
SECURITY COMPASS