What **Every Security Leader Needs to Know** Handbook

hackerone

Table of Contents

PART 1

The First 90 Days

You've been told that you're the next director of security for your organization. You must start and scale security practices at a high-growth company. What happens next? Where do you get started?

Your company may have basic security processes in place, such as security scanning tools and a penetration test every year. But what is the best foundation for security at scale? What steps will build a strong security program that grows along with your company? Take our plan and use it to start making progress from day one.

CHAPTER 1

Getting Your Bearings

Congratulations! You've been made the next Director of Security (or other similar title) of your organization. Taking over such an important role can seem daunting. New projects, those already in progress, upgrades, vulnerabilities, what security will look like in five years for your company. All of this is swirling around in your head from day one.

Veteran security leaders we've talked to suggest starting your tenure by getting your bearings. Take the time to understand what the current state of your organization is and where it should go. This will take time. But this time will prepare you to lead your security efforts for years to come.

Understand the Business Reason for Your Appointment

Getting your bearings begins with understanding the circumstances surrounding your appointment to be the Director of Security. This doesn't mean what your professional credentials are, but what purpose your position serves within the organization in the greater scheme of things.

Our experience working with security leaders at start-ups, Fortune 500 companies like GM, and everything in between reveals 3 primary reasons they appoint a new security leader.

Growth of the organization. As organizations grow, their security footprint grows. Small startups need security but are mainly focused on delivering functionality quickly to the market to test if it's viable. As you gain users, the need for security increases, and the damage from a security vulnerability is more far-reaching.

Application security practices and procedures, infrastructure, network, and cloud security policies are needed. You'll be the liaison between security and other key departments within the company. Your job will be part captain, part ambassador.

Acquiring another company. When a company is acquired, its systems must be integrated safely with yours. Its applications may have vulnerabilities you have to manage. Safe integration is now your responsibility.

Expansion into a new region. When conducting business in new regions of the world you're exposed to new laws and regulations, such as GDPR. You must understand what new laws apply and build a plan for compliance.

Understanding the reasons for your role to exist plays a large part in deciding where to start. You can build a better plan when you understand the events leading to your appointment. Once that is done, you can move on to understanding what security at your company looks like right now.

Understand the Baseline of Today's Security

One of the first things you'll need to do when beginning your journey as Director of Security is to create a baseline of current security within your organization. It's important to understand the current state before trying to decide what the future should be.

Speak to the leaders and managers from several departments, including HR, legal, finance, IT, and security. They can tell you what risks and threats they see for your organization as a whole. You can properly rate risks and decide which to tackle first when you understand the entire picture, including business and technology risks.

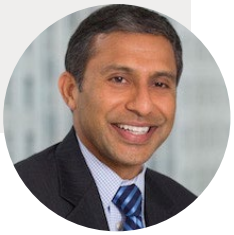
Get a feel for recent incidents or major issues experienced by your organization. These could point to systemic flaws that need to be repaired right away.

Ask each department about the current state of risk management. Are risks being handled properly? How long does it take for mitigation to be put into place? Are there any process improvements that would make risk management more efficient for them?

Engaging with other teams gives you two benefits. First, you'll better understand how your organization is handling risk right now. Your future decisions will be based on a complete picture of your company. Second, you're showing yourself as a partner to these groups instead of an adversary. You're not there to place blame but to help reduce risk and make the company a safer place for employees and customers alike.

With your bearings and a complete picture of the current state of risk management in hand, you'll be prepared to set the foundation for security moving forward.

In a recent interview with Security Roundtable, Kal Bittianda, an executive recruiter who helped Equifax find a new CISO after their data breach, spoke about the important skill of listening—a skill every security leader needs to develop. "Listening is huge—trying to understand the problems versus being perceived as the person who is adding more problems to their plate. It's important to build those relationships so they believe you've got their back and are willing to help them get things done.



CHAPTER 2

Setting the Foundation for Security

Not all foundations are the same. Some types of soil don't have great strength or have troublesome properties. You must investigate the soil before building the foundation.

At this point, you've met with several departments to get a clear picture of the risks facing your organization. You've created a picture of risk management. This is important for a good foundation.

But don't forget that security cannot happen without the developers being on board. The relationship between the security and development teams is foundational to good security.

Approach those relationships with tender loving care.. Take time to assess its health. Determine what changes may be required to strengthen it.

It's important for development teams to understand that you're there to help them find solutions. You aren't the "code cops" coming to break down their door. You aren't trying to get in the way of delivering functionality. Instead, you're helping them to find solutions to fundamental security problems. You're working with them, with an equal say on both sides, to create solutions that please both sides.

Coming from a place of mutual trust lays a solid foundation. It may take time if developers and security have a history of mistrust. It's worth the effort.

Understand Security from a User's Standpoint

An important piece of a sound security foundation is your viewpoint of the customers using your product. Some products are used by employees, others by paying customers. Both viewpoints are key to a successful security strategy.

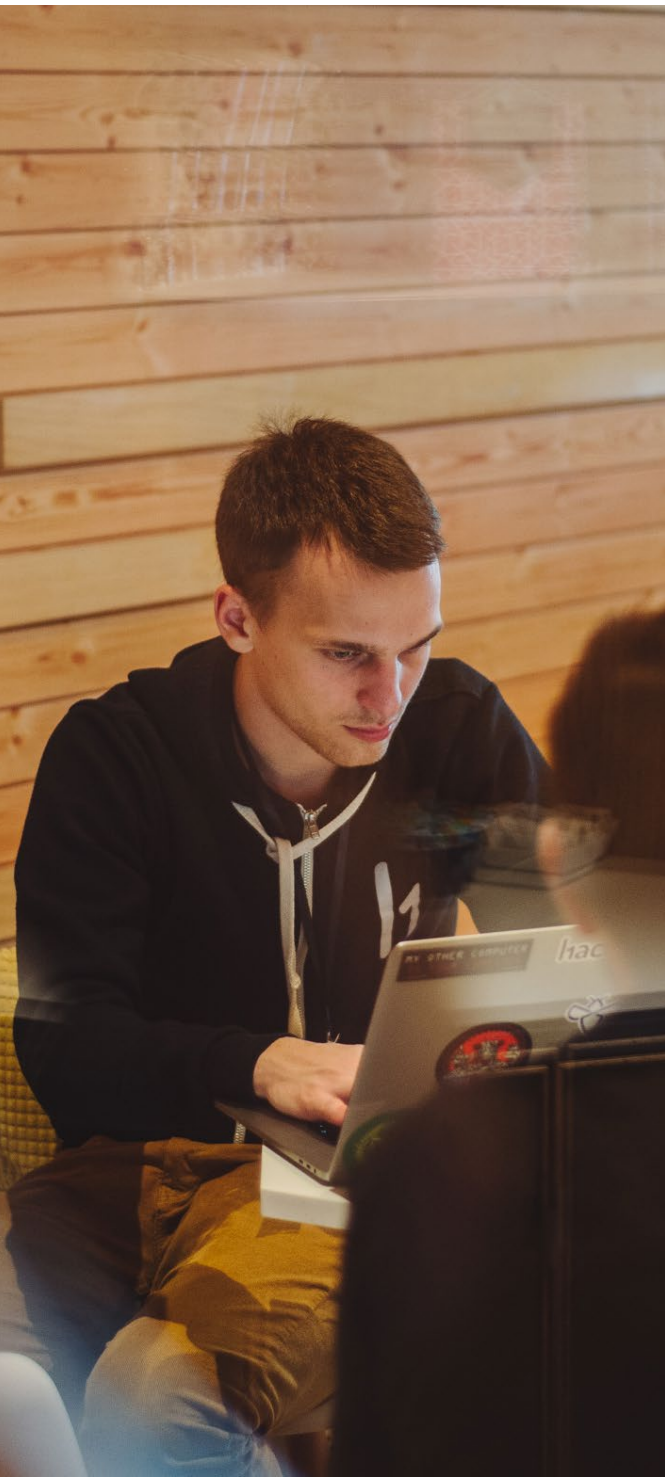
Good hackers get familiar with the product and try to put themselves in the role of a user. They try to break the system by doing unexpected things. They ask themselves: "What can a user do to make the system do something it shouldn't?"

You should be doing the same. Start by walking through the employee onboarding process from start to finish. Pay attention to the processes used to create accounts, provision hardware, and create passwords. Don't search for vulnerabilities, but pay attention to inefficiencies or glaring problem areas.

After an internal look, walk through a customer's experience in buying your product. Start with marketing and move through account access, entering credit card information, or however your particular flow takes people from awareness to paying customer.

Understanding these flows can better inform your strategy moving forward. You'll view your systems through the user's eyes.

You can't secure a system you don't understand. Taking the time now to understand it will pay off later.



Understanding Threats Against Your Systems

Threat modeling is the practice of reviewing the design of a system to find threats against that system. These threats are recorded and mitigated to the extent possible.

Threat modeling allows you to anticipate problems. Use it to identify threats against your systems and possible weaknesses attackers could exploit. When done early enough in the development life cycle, you'll be able to fix major problems before they can be exploited.

Threat modeling results can change your plan. Use it to focus your security budget and penetration testing efforts.

Setting a secure foundation helps you remain stable as you begin to implement your security strategy. Build trust with development teams. Learn your systems from a user's viewpoint. Threat model your systems to find weaknesses your strategy can help resolve.

Once a foundation is laid, it's time to start building.

CHAPTER 3

Prioritize and Execute

At this point, you've identified:

1. *What projects are currently in progress*
2. *Which projects are needed to improve your security*
3. *Improvements based on threat modeling results*

Your job now is to figure out what to do first. There is no silver bullet to effective planning. It's difficult. But there are ways to break down the necessary pieces of your plan and arrange them in a way that makes sense.

Congratulations! You've made it through your first 90 days. Here's what you've accomplished:

- *You've taken a baseline of the risks and threats facing your organization*
- *You've learned your systems in and out, from both a technical and user-focused angle*
- *You've decided what projects are the most important to complete, and are ready to execute*

Leading a security organization is more than planning. You must now decide on a mode of operation. How will you operate your business day-to-day? What principles will guide your decisions?

PRIORITIZATION CHECKLIST

- ☐ First, understand the reasons for the existing projects. Are they based on a real need? Is that need based on data? You may find it necessary to shut down projects that are no longer important or projects that aren't succeeding.
- ☐ Second, find out what new initiatives are coming down the line. Try to anticipate future needs. These could be regulatory changes, new marketing initiatives, or new technology initiatives. Understand what is absolutely necessary for the upcoming year.
- ☐ Once you have this list of projects, the hard part comes. You have finite resources. You have to choose projects that are absolutely necessary right now.
- ☐ Walk through the upcoming year in quarters. For the first quarter, pick one or two projects that must be done. These are the projects you'll start with because there is absolutely no wiggle room.
- ☐ For each quarter after that, imagine yourself at the beginning of the quarter. Your first two projects have been successfully delivered. What project has to be done right now? That's the next project on your priority list. Move through the year, slotting each project where it belongs. All of the projects will be finished this year, but not all should be in progress at the same time.
- ☐ Planning in this manner assures that your team can focus on completing the most important project. Spreading people across too many projects can lead to making small progress on all of them but not finishing any of them. Focusing on two, maybe three projects with all of your resources will ensure their completion before moving on.

PART 2

You're Settled In, Now What?

You've gotten your bearings, built strategic relationships with other departments, and gained a firm handle on what you need to do next.

Now comes the fun part: executing your plan. Let's look at how to manage the business of security on a daily basis.

CHAPTER 4

Don't Eliminate Risk, Manage It

The act of planning projects based on the threats found against your systems can have a detrimental effect. A project has a beginning and an end. Once it's finished, you move on to the next project.

Risk management doesn't work that way. **Eliminating risk is impossible. There is no neat bow to tie on risks when you're done. Risk must be managed.**

Risk management begins with understanding your assets—where and how valuable they are. A vulnerability in an access database used by 20 employees is not as important as a vulnerability found in your publicly exposed API.

Any risk found goes through a cost/benefit analysis. The possible exposure is compared to the cost to mitigate the risk. If a realized risk will cost you $10,000 but it'll cost $50,000 to fix, then it is a low priority. It simply doesn't make sense to fix it right away.

Risk ratings are a necessary part of any risk management strategy. What rating system works best to help determine priority?

Simple rating systems help the relationship between security and development teams. Developers need to understand what needs to be fixed, and what doesn't. Introducing intermediary levels, such as Medium or scales from one to ten can make it harder for developers to plan upcoming work. Medium risks are the ones most likely to sit in limbo because no one knows how important they are. Be clear in your priorities to avoid confusion.

Any risks rated highly should undergo root cause analysis. Find out what led to such a critical risk and how to prevent the same thing from happening again. Over time, the emergency, "hair on fire" vulnerabilities should begin to disappear.

Andrew Dunbar, VP of Security Engineering and IT at Shopify, stated in a recent webinar on effective application security testing his preference for simplicity. He determines the potential exposure of a risk "in a way that is as simple as it can be to achieve the right outcomes." Don't over-complicate your risk rating system to the point where ratings don't mean anything. Dunbar explains that Shopify has two ratings, high or low. High ratings need to be fixed right away. Low risks can be fixed during the next sprint or two.

Patch Management

Dependencies are inevitable when building complex software systems. Your business operations likely depend on a mixture of in-house applications, open source components, and third-party vendor applications. All require good patch management to stay secure.

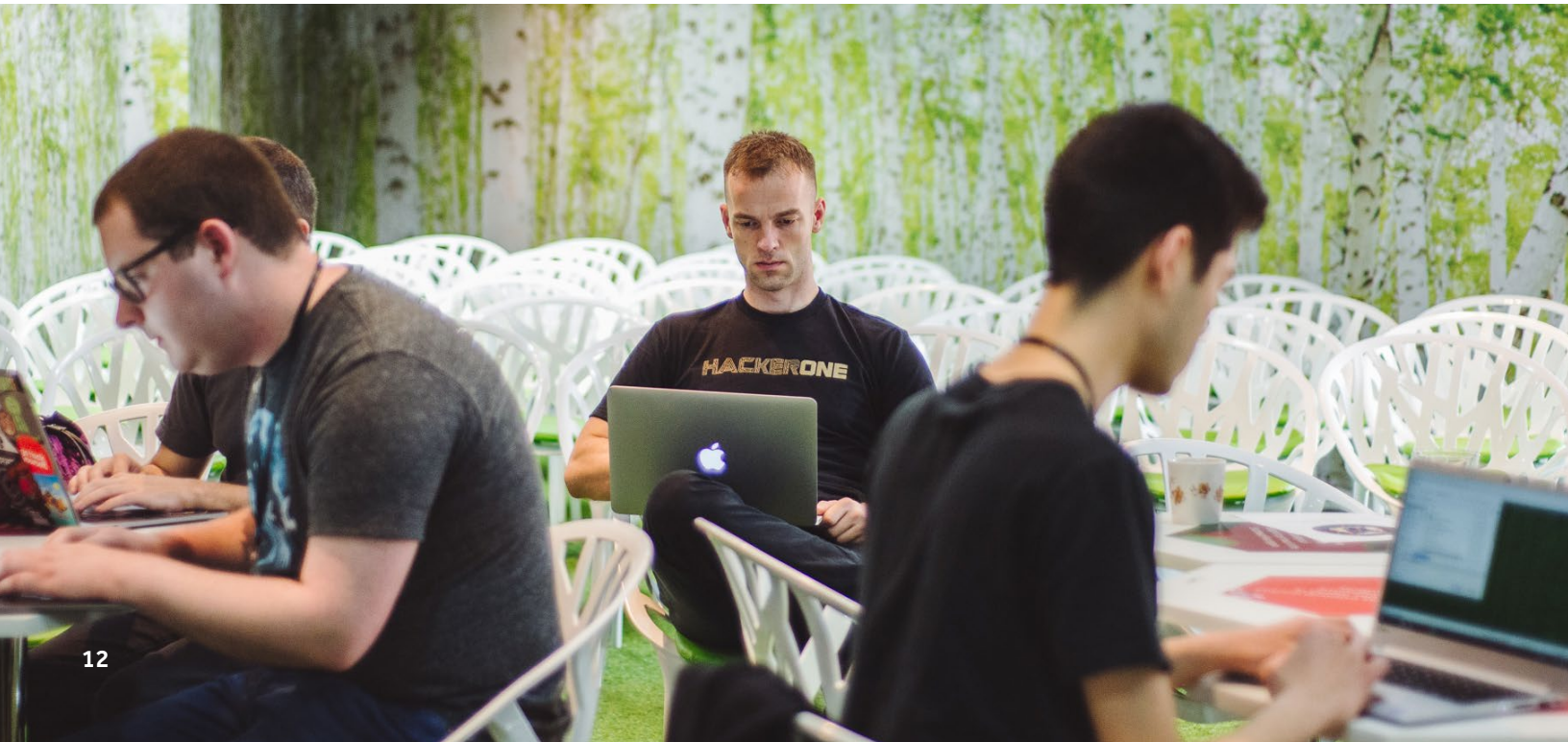
The code you write depends on open source libraries and frameworks. When vulnerabilities are found within these frameworks, all applications that use them are also vulnerable. It's essential to update libraries and frameworks quickly so your applications aren't getting exploited. The code you write might be bug-free, but what about the code your code depends on?

Third-party vendor products undergo the same testing as yours. This testing could find vulnerabilities that put your users at risk. Patch management processes ensure that patches released by third-party vendors don't sit for months before being applied to your environment.

Once a vulnerability is announced to the public, attackers will try to exploit it. The longer you wait to patch your systems, the longer attackers have to find your application and take advantage of your sloth.

Patch management processes can be difficult to create, but they don't have to be complicated. Whenever possible, work within the confines of your developers' daily workflow.

For example, if an open source library needs to be patched, create a pull request within the repo of the application so all developers will see it. Explain in the pull request what you're updating and that all they have to do is merge and their job is done. You'll not only help your application stay secure, but you'll also generate plenty of goodwill with development teams.



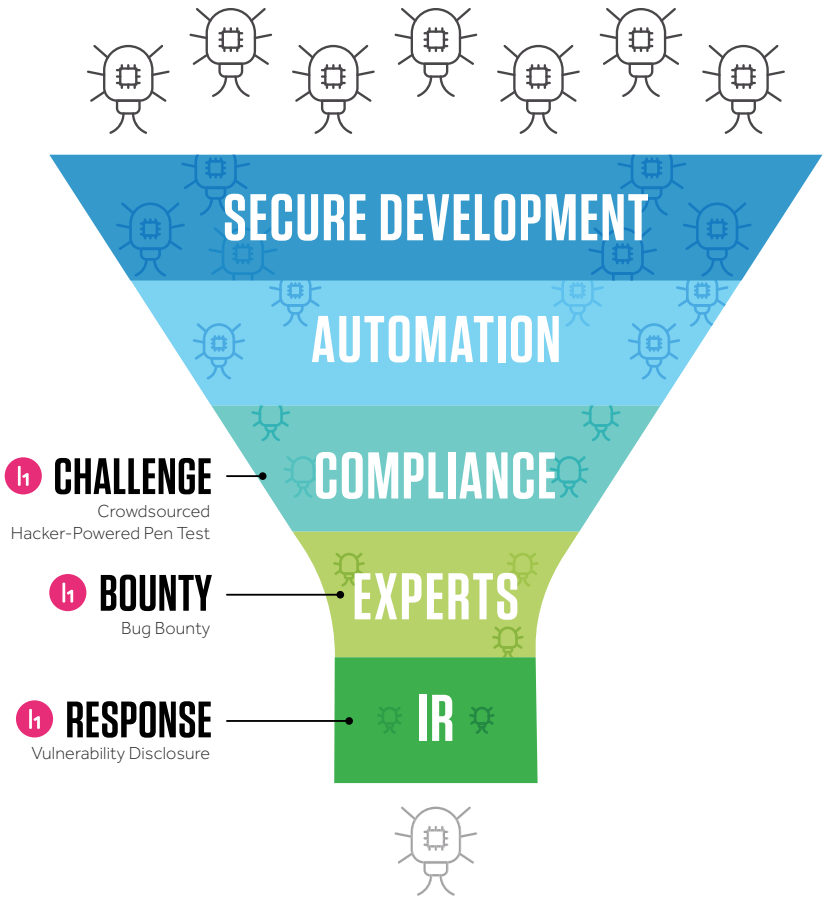
CHAPTER 5

The Wide World of Security Assessments

Security assessments are a necessary part of every security leader's job. In smaller organizations, you may be directly meeting with outside partners to complete these assessments. In larger organizations, there may be a team in charge of handling these kinds of relationships, but you'll still have to sign off on what is happening.

Before jumping right into scheduling security assessments for your applications, take the time to understand the specific purpose of each type of assessment.

Defense in Depth



Black-box and White-box Testing

Security assessments can either be white-box or black-box assessments. Understanding the difference will help you to use security assessments effectively.

White-box testing refers to allowing the tester to see into the inner workings of an application. The tester can see code or system diagrams, allowing them to find problems in the implementation of an application.

Black-box testing refers to testing an application with no knowledge of how it works. These tests simulate the vantage point of an attacker who has to learn by exercising the system. This type of testing better illustrates what an attacker would have to accomplish to successfully attack a system.

A good example of the difference would be in the use of encryption. A black-box test blindly attacks the cryptographic system and tries to break it. If successful, the tester knows that the encryption is inferior, but doesn't know exactly why.

A white-box tester would read the code and see exactly which encryption algorithm is being used. The tester will note that the algorithm is inferior and suggest a better one. In this way, access to the inner workings of an application can result in more specific guidance on what vulnerabilities exist and how to fix them.

Vulnerability Assessments

Vulnerability assessments are white-box tests designed to reveal as many vulnerabilities as possible within an environment, along with guidance on remediation and priority.

Vulnerability assessments are not the same as penetration tests, which we'll discuss next. They are a comprehensive look at all of your systems with access to the inner workings via code or diagrams.

Vulnerability assessments are the best choice if your organization has a low to medium security maturity. The goal is to find as many vulnerabilities within your environment as possible so you can secure the most critical pieces quickly. Once your environment has been hardened by several of these assessments, you can better take advantage of other assessment types.

Penetration Tests

Penetrations tests and vulnerability assessments are sometimes confused, but they are not the same. Vulnerability assessments are white-box examinations of all vulnerabilities within a system. The point is to fix as many problems as possible within a short period of time.

Penetration tests are tightly focused, black-box tests aimed at specific functionality within a system. Penetration tests should be done after you've cleaned up the vulnerabilities found and have a reasonable level of confidence in your application. Penetration tests have a specific goal in mind, such as exfiltrating data or gaining admin rights to a server.

For instance, you may perform a vulnerability assessment (or multiple assessments) against your shopping cart functionality to find common configuration and coding errors. Once all of the findings from that first assessment are fixed, you run a penetration test against the shopping cart system to make sure it's been sufficiently protected. This order of testing ensures that penetration tests are used effectively—finding difficult to detect errors that code scanners won't find.

Red Team Services

A red team is a permanent team used to improve the information security posture of a company. Red teams are not a one-time assessment but are continually testing applications to find vulnerabilities.

Red teams focus on using real-world tactics to attack an organization's assets. They are made up of highly trained and experienced professionals who think like attackers.

Red teams serve a unique purpose. They aren't just finding vulnerabilities and reporting them, but are built to make corporate defenders, or the "blue team," better. The result is an improved ability to detect and contain attacks as they happen, instead of finding the evidence long after the damage is done.

Audit

Audits are a necessary concern for most companies, especially those in highly-regulated industries.

An audit is not a true security assessment. It measures how well your systems match up to a chosen standard. Even if some vulnerabilities are found during an audit, that isn't the main purpose.

A standard, such as PCI-DSS, may state that you must use strong encryption to protect customer data. It may have a list of acceptable encryption algorithms and you may use one of those algorithms. That's as far as the audit goes. A penetration test may find that you store your encryption keys alongside the data in the database, allowing an attacker to easily decrypt the data.

You can be compliant with a standard and be insecure at the same time. Audits don't verify security but verify conformance with an interpretation of what security should be. This is an important distinction.

Organizations with good security practices are very likely to be compliant. But compliance, while necessary, should never be confused with security.

Juggling various security assessments and audits is no easy feat—you can do it when you understand the purpose behind each assessment and when to use it.

HACKERONE COMPLIANCE PASSES ALL TESTS

Audits measure compliance. Many standards, such as SOC2 and PCI-DSS, require companies to test their applications' security on a regular basis. Annual penetration tests are a common approach to achieving compliance.

Traditional penetration tests have limitations. There are only a handful of security researchers available. You have to fit into their schedule. Their expertise may not be enough to find complex vulnerability chains, or several low-severity bugs that combine to have a large impact. You may find that you've spent large amounts of money for little benefit.

HackerOne Compliance offers a hybrid model that combines incentive-driven vulnerability testing with targeted testing for specific categories of vulnerabilities, such as the OWASP (Open Web Application Security Project) Top 10. Access to potentially hundreds of security researchers during your test exposes your applications to a diverse skill set. Talented researchers test your applications using real-world attacks and modern testing tools and techniques.

Many growing companies have experienced reduced friction and saved time and money using HackerOne Compliance.

GRAND ROUNDS®

"As a rapidly growing enterprise, Grand Rounds needed to get more agile and implement more automation when it came to cybersecurity. That said, with some automation comes predictability. We were worried that we were testing the same things the same way. We turned to HackerOne for scalable real-time testing that would look in the places we weren't looking, not a simulation or templated test, for SOC2 compliance. The findings through the HackerOne Challenge programs have helped us grow our security team's capabilities and better secure the organizations and individuals that turn to us for healthcare guidance."
— Steve Shead, Grand Rounds

TRUELOGIC

"We had received reports from annual pen tests that, while valid, did not demonstrate any real, practical risk to truelogic. Within the first 6 hours of running a HackerOne Challenge, hackers found real issues, which could actually be exploited."
— Mike Burgh, truelogic

yext

"We've had mixed results with traditional penetration testing firms in the past. With HackerOne, our Challenge was immensely successful. Beyond my expectations."
— Sean MacIsaac, Yext

sumologic

"This is value that we never got from a pen test," he added. "Traditional pen tests are not enough for modern day security."
— George Gerchow, Chief Security Officer at Sumo Logic

CHAPTER 6

Securing the Shifting Sands

Any experienced security leader will tell you that continuous learning is a requirement of the job. The ever-changing risk landscape exposes your company to new and dangerous risks every day. Let's look at some general principles that'll help you keep ahead of these risks.

Keep an Eye on Third Parties and Vendors

Third parties have practically become a requirement in today's connected world. There's nothing wrong with using third parties to perform tasks your business wasn't created to do. Why create your own Human Resources system or shopping cart functionality if that's not what brings you revenue?

Using third parties, however, brings risks to your business. There are three ways risk increases when using third parties:

1. *The risk of data being misused by a third party*
2. *The risk of poor security practices leaking your data without your knowledge*
3. *The increased attack surface if the third party application contains vulnerabilities*

In today's environment, you're only as secure as your weakest vendor. Vet your vendors carefully and make sure you're comfortable with their security policies before signing a contract with them. Once you hand over your data, it's too late to worry about security.

Your Attack Surface Changes

Companies are adopting DevOps practices more as time moves on. These practices encourage dev teams to put code out and get fast feedback wherever possible.

Don't sacrifice security in exchange for "better, faster, cheaper." Fast-moving development environments increase the risk of services and applications being deployed without the security team's knowledge. This is especially true of cloud environments, where developers could create virtual machines and deploy code to them at any time.

Strong policies are needed so admins know what they are allowed to do. Policies preventing the abuse of cloud resources are a good idea, but you have to enforce them. AWS, Azure, and Google Cloud aren't going to police your admins and developers for you.

The good news is that automation can be used to help enforce policies. For example, AWS Lambda can be used to scan file uploads to S3 buckets in AWS. Policies can prevent developers from creating new virtual machines using their accounts. A good rule of thumb is to use a build pipeline to build any infrastructure your applications need, preventing humans from deciding how to build VMs and deploy them.

The security team should be made aware of any new assets and what their purpose is. This can be daunting in a microservice environment but is absolutely necessary. Know when new services are created and released. Understand what cloud service accounts exist and have a clear process on how to create new ones so they can be monitored. You can't be reactive—proactively look at what is currently running in your environment and respond to anything weird.

The key to keeping up with the security of your systems is to reduce the element of surprise. Build processes that enable security teams to stay up-to-date with new assets. These assets must be tested and deployed according to well-enforced policies. Hold your vendors to the same standards you'd hold yourself. Don't trust your data with just anyone.

Making sure third parties handle your data right and keeping up with your changing environment can be a challenge. But successfully tackling this challenge is possible and enables a mature security program that'll be invaluable as your company grows.

PART 3

Maintaining a Successful Security Program

As a Director of Security, it's your responsibility to create an environment that encourages security, making the day-to-day measures we discussed in Part 2 much easier.

How do you create programs designed to deliver security at DevOps speeds? How do you stay ahead of coding errors that can cause large amounts of damage? Do you share what you've learned with others or keep it secret?

CHAPTER 7

The Art of Continuous Security

"Continuous security" may seem like a strange phrase. Nothing is 100% secure. No one silver bullet exists that keeps all systems everywhere impenetrable. But that's not the main goal with continuous security.

Continuous security is a defined process that allows you to know what is happening in your environment and react quickly to it. It uses smart automation to make security the default. You make security an intrinsic part of your applications without stopping development teams from delivering quickly.

The use of the term "art" in the chapter title is deliberate. Security in a DevOps environment is often more an art than a science. There are concrete aspects, such as metrics to measure test coverage or policies to prevent rogue servers or buckets. But how much test coverage is enough? 70%? 80%? And who should have authority to create servers, all admins or a select few?

These are decisions that have to be made. You can get advice from hundreds of articles on the internet, but the final decision is yours. You make it and you have to live with it.

The best guideline to use is your customers. What will it take to make sure your software is trustworthy? Your goal should be to build software your customers will trust. Often, vanity metrics or minimum thresholds only deliver minimum security. Being trustworthy takes much more than just meeting the minimum.

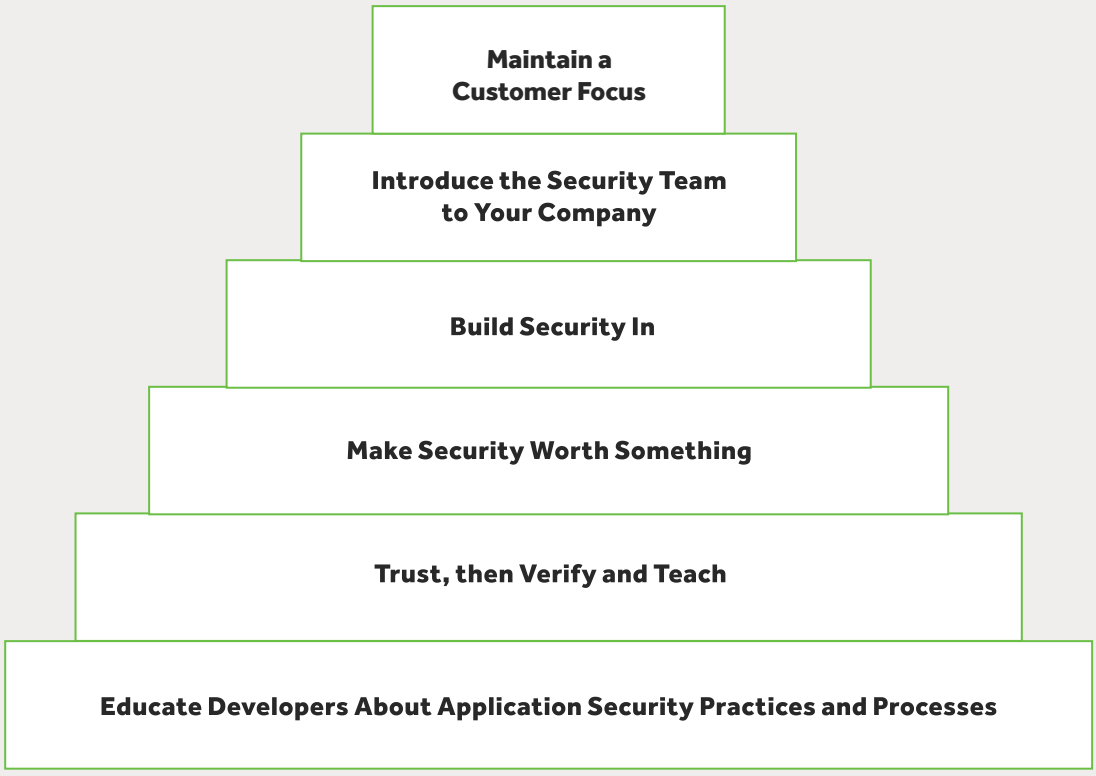
Build a Culture of Security

Culture is like the personality of a company. It's the operating environment of a company. The values, mission, and attitude of a company and its employees.

Security has often been a background process, like scanning for vulnerabilities or performing a vulnerability assessment before deploying to production. That's not enough for continuous security.

ACHIEVING A CULTURE OF SECURITY

Let's look at some key steps to building a culture of security.



Your developers should understand basic application security principles. They should be trained to understand exactly what processes exist and why. Allow them to spend time with the security team, learning what to look for and what applications look like through the security team's eyes. Allow the security team to spend time with the developers. Learn what security processes get in the way and eliminate them.

Give developers the freedom to experiment. Trust that they want to do the right thing, then verify. When mistakes happen, help solve the problem without placing blame or punishing whoever made the mistake. Instead, fix your systems so the same mistake can't be made again.

An excellent way to show trust while educating your developers was discussed in a BSides San Francisco conference talk by Arkadiy Tetelman. His favorite way to build a security culture is to send out "fireside bug bounty" write-ups to developers and other interested parties when a legitimate vulnerability is found via bug bounty. Every communication outlines what the vulnerability was, how to exploit it, who fixed it and how, and how to eliminate that entire class of vulnerability across the system. Fireside bug bounties encourage development teams to think about security regularly and to be helpful when security teams make requests for fixes or updates.

Make security worth something. Give $200 to the developer who reports a strange VM running in the cloud or fixes a nasty vulnerability without the security team having to ask first. Reward the marketing employee when she reports a phishing email, even when no tests are ongoing.

Build security in as much as possible. Common security features, like authentication and authorization, should be built into reusable development frameworks. Build servers with automated scripts based on a known secure template. Make security easy. Someone should have to work hard to build an insecure system.

Introduce the entire company to what your security team does and why it's important. Fun events like security expos give you the chance to demonstrate what attackers can do if they succeed in breaching the company. Show a day in the life of a security engineer or incident response engineer. Tell the security team's story, entertain your visitors. If it's memorable, you'll have less friction when you need to introduce new policies or standards.

Above all, make your customer the focus. Build your culture around delivering the best service to your customers—not just keeping the lights on, **but becoming trustworthy stewards of their data**.

Building a culture takes time, but is well worth the effort. Security has become everyone's job. Make security easy. Make it fun. Make it worth something.

So far, we've focused on planning and directing internal resources. Many security leaders recognize the value of opening certain security activities to external experts. In addition to the assessments described earlier, bug bounties are becoming increasingly popular. When should bug bounties be brought into your strategy? And how can you integrate them with existing security activities?

CHAPTER 8

Bug Bounties: When and How?

Bug bounties take advantage of the large hacker community to find vulnerabilities you don't have the resources to find yourself. Hackers submit bugs they find and are rewarded by you based on the severity and impact of the bug.

Bug bounties bring many benefits. First, you have access to more security researchers than you could ever afford to hire. Most security teams are a fraction of the size of development teams. Bug bounties allow you to scale your security efforts much more easily than hiring.

While not a silver bullet, bug bounties are a strong addition to the security program at your company. When your goal is continuous security, bug bounties help you reach that goal by providing continuous security testing by thousands of hackers all around the world.

WHAT SECURITY LEADERS SAY ABOUT BUG BOUNTIES

flickr

"Bug bounty is a crucial element to our larger strategy. While we train and encourage our teams to think about security as being paramount, when things slip through the cracks we're glad we have bug bounty hackers researching the site and keeping our users safe."
— Alex Seville, Senior Engineer Manager, Flickr

Quantopian

"We started receiving reports from hackers of vulnerabilities which the consulting firm should have found but didn't. We decided we needed to try something different."
— Jonathan Kamens, CISO, Quantopian

grammarly

"Our HackerOne bug bounty program is one part of our enhanced security strategy. It helps us identify systemic issues, which we can then work to resolve. This can mean anything from taking a different approach to building our technology to having more robust review processes.
— Joe Xavier, VP of Engineering, Grammarly

How To Build Toward a Bug Bounty Program

Like any large initiative, it's best to start small with bug bounties and scale them up as you learn how to manage them and mature. There are concrete steps that'll take you from bug bounty newbie to master in no time.

A vulnerability disclosure policy (VDP) is the first step organizations need to take before creating a bug bounty program. VDPs formalize the method of submitting vulnerabilities to a company. They typically give guidelines to hackers on what applications they can test, how to submit vulnerabilities, and how the company will handle submissions. VDPs typically include language promising not to prosecute hackers as long as they stick to the guidelines.

5 CRITICAL COMPONENTS IN EVERY VDP

1. **Promise:** Demonstrate a clear, good faith commitment to customers and other stakeholders potentially impacted by security vulnerabilities.

2. **Scope:** Indicate what properties, products, and vulnerability types are covered.

3. **"Safe Harbor":** Assures that reporters of good faith will not be unduly penalized.

4. **Process:** The process finders use to report vulnerabilities.

5. **Preferences:** A living document that sets expectations for preferences and priorities regarding how reports will be evaluated.

VDPs are essential to establish guidelines and make hackers feel comfortable submitting findings. Once a VDP is in place, you'll need a mechanism to accept vulnerabilities. To keep things simple, use an email address for vulnerability submissions. These typically take the form of "security@<yourcompanynamehere>.com."

You can experience the bug bounty world without starting a permanent program. Hacker-powered penetration tests offer the best of both worlds—the limited scope of penetration tests with the hackers and bounties you see in a traditional bug bounty program.

These 4-week engagements demonstrate the power of bug bounties while giving you complete control over what is tested. Some companies elect to hold one-day challenges to see who can find the most bugs. These options give you the chance to meet exceptional hackers who you could later invite to a private bug bounty.

A hacker-powered penetration test is also a great way to remain compliant with standards and regulations while receiving high-quality vulnerabilities at the same time.

Once word gets out about your bug bounty, it's easy to become overwhelmed with bug submissions. Private bug bounties invite a few hackers to participate in the program. These hackers submit vulnerabilities and help iron out any inefficiencies in the process while still helping your systems improve security. Start with a private bug bounty program to help your triage team get used to communicating with hackers.

Once you have submission, triage and resolution processes moving like a well-oiled machine, you can open the bug bounty up to the public. Public bug bounties have many more bug reports to work through, but can help find creative vulnerabilities other testing methods will miss. Public bug bounties maximize your exposure to skills and testing methods and provide your company with some good PR.

Bug bounties have helped many companies supplement and scale their security teams in ways they never thought possible. Take the time to evaluate your current processes and see where a bug bounty program may fit.

CHAPTER 9

Share Your Knowledge

Security has long been associated with secrecy. If no one knows how something works, they can't attack it. If you share what you do to protect yourself, you're opening yourself to attacks.

The security industry is changing. Secrecy is no longer the name of the game. Instead, openness and collaboration between security professionals is becoming the norm. When security leaders share their knowledge, it helps the entire security community become better.

The Internet doesn't have to be the wild west with an "everyone for themselves" attitude. The Internet becomes safer as a whole when leaders work together to share information. If your security program develops into a world-class operation, share with others how that happened. If everyone has a world-class security team, everyone is that much safer.

Keep track of what you've learned and looked for opportunities to share with others. Professional organizations, such as ISC2 and SANS exist to help share security knowledge. Become an instructor or contribute research reports and case studies. Conferences also present a great opportunity to share knowledge. Apply to speak at a security conference so you can share what you've learned in a relaxed setting to hundreds of people. If they apply what you've taught them, you've made an impact on the security industry as a whole.

Security leaders lead not only their organization but possibly others as well. Collaboration and sharing is the way to a more secure future. Take advantage of opportunities to teach and learn from others. Together, we'll make the Internet a safer place for everyone.

Appendix: Application Security Checklist

- ☐ Collect baseline of current state security
- ☐ Understand security from a user's standpoint
- ☐ Threat model your applications
- ☐ Don't try to eliminate risk, manage it
- ☐ Establish clear patch management procedures
- ☐ Use security assessments to find vulnerabilities from different angles
  - ☐ Threat assessments to review the entire system for vulnerabilities
  - ☐ Penetration tests to test one specific piece of your system once hardened
  - ☐ Create a red team to constantly attack your applications with the mind of an attacker
- ☐ Create security assessment policies for third-party vendors—vet your vendors
- ☐ Create processes to monitor new services and applications that are created
- ☐ Watch AWS accounts and assets for rogue VMs and services
- ☐ Create "trustworthy" thresholds—the thresholds for test coverage and other metrics that create the most "trustworthy" software
- ☐ Promote a culture of security
  - ☐ Teach application security practices and processes
  - ☐ Trust, then verify and teach
  - ☐ Make security worth something
  - ☐ Build security in
  - ☐ Introduce the company to your security team
  - ☐ Maintain a customer focus
- ☐ Build toward a bug bounty program
  - ☐ Create a VDP
  - ☐ Perform a hacker-powered pen test
  - ☐ Create a private bug bounty
  - ☐ Create a public bug bounty program
- ☐ Share your knowledge through professional organizations and conferences

hackerone

ABOUT US

HackerOne is the #1 hacker-powered security platform, helping organizations find and fix critical vulnerabilities before they can be exploited. More Fortune 500 and Forbes Global 1000 companies trust HackerOne than any other hacker-powered security alternative. The U.S. Department of Defense, General Motors, Google, Twitter, GitHub, Nintendo, Lufthansa, Microsoft, MINDEF Singapore, Panasonic Avionics, Qualcomm, Starbucks, Dropbox, Intel, the CERT Coordination Center and over 1,300 other organizations have partnered with HackerOne to find over 120,000 vulnerabilities and award over $51M in bug bounties. HackerOne is headquartered in San Francisco with offices in London, New York, the Netherlands, and Singapore.

Contact us **to get started.**