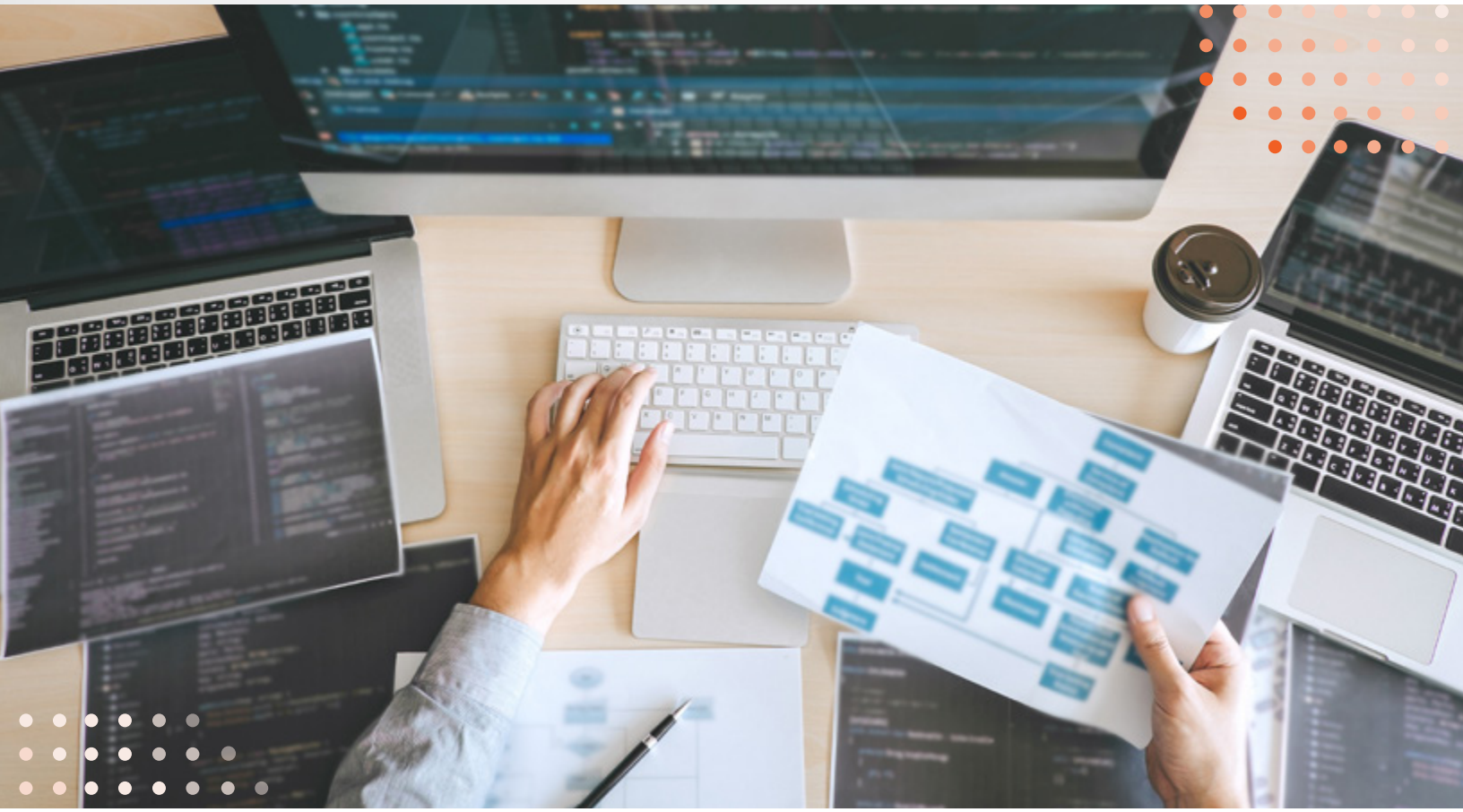
WHITEPAPER

Speed and Security: Recent Innovations in Secure Design and Development



SecurityCompass

Preface

On behalf of the Security Compass team, it has been a privilege to produce this whitepaper for you. This is the culmination of great team effort, and we hope you find this research valuable as you seek to improve the security of your software through secure design and development practices. Ultimately, we want to help build a world where our technology can be trusted to be secure and safe.

I want to take a moment to thank all the security community members around the world who work tirelessly across a number of different areas ranging from Agile to Zero Trust. This is an acknowledgement and an immense thank you to all. Please keep up the good work.

In order to provide clear value in this whitepaper, we had to carefully select our audience. The target groups for this whitepaper are Security Engineers and DevOps Engineers. We want to help bridge the gap between these two groups.

So why did we write a single whitepaper that addresses both secure design and development? Why not create two separate whitepapers? We feel there is a great deal of overlap between these two crucial security activities. Rather than repeating much of the same information across two whitepapers, it makes sense to produce a single whitepaper.

As with any whitepaper, there are those who both agree and disagree with us. We openly welcome your comments and invite you to join us as we push forward with integrating security practices into our DevOps delivery pipelines. Let us know what you think. Your feedback helps inform the narrative that we, as security professionals, are trying to push: that software can be produced both rapidly and securely.

Please visit us at securitycompass.com or drop us a line at info@securitycompass.com

If you'd like to join our research efforts, contact our research team by email at research@securitycompass.com

We look forward to hearing from you, and wish you great success.

Sincerely,

Altaz Valani
Director, Insights Research
Security Compass

Credits

Executive Sponsor: Bruce Warren

Community Outreach:
Janet Khoshaba, Raquel Rodrigues

Editor: Megan Barker

Designer: Morgan Dunbar

Introduction

Every activity within your software development life cycle should ultimately contribute some value to the business. Typically, this means a focus on revenue generation or cost reduction. This type of alignment helps your project teams focus on essential activities that help drive forward the business strategy and operating model.

As part of software development, security is no different. Your security activities must contribute value towards enabling the business priorities. While security in software development can include dozens of activities, in this paper, we will focus specifically on secure design and development activities. Both of these activities are sufficiently early in the software development life cycle to make a meaningful contribution toward proactive security and minimizing downstream noise that often slows down our lead time.

When we talk about secure design and development, it is often very difficult to speak about value creation in the same breath. In many cases, secure design practices (more specifically, threat modeling) are perceived as blockers to fast-moving DevOps pipelines. Secure development (more specifically, secure coding) suffers from lack of developer knowledge on how to code with security in mind.

In this whitepaper, we will drill down on these two security activities: threat modeling and secure coding. We conducted a review of current literature regarding these topics and identified a number of problems and solutions to overcome these challenges.

In our research, we focused on specific questions related to value creation. In our opinion, value creation is about achieving both speed and security. Often speed takes precedence over security. This offers very little value from a security perspective. The rationale for pursuing specific questions came out of our own experience working with customers and interacting with a broad range of communities in industry. The questions we chose were:

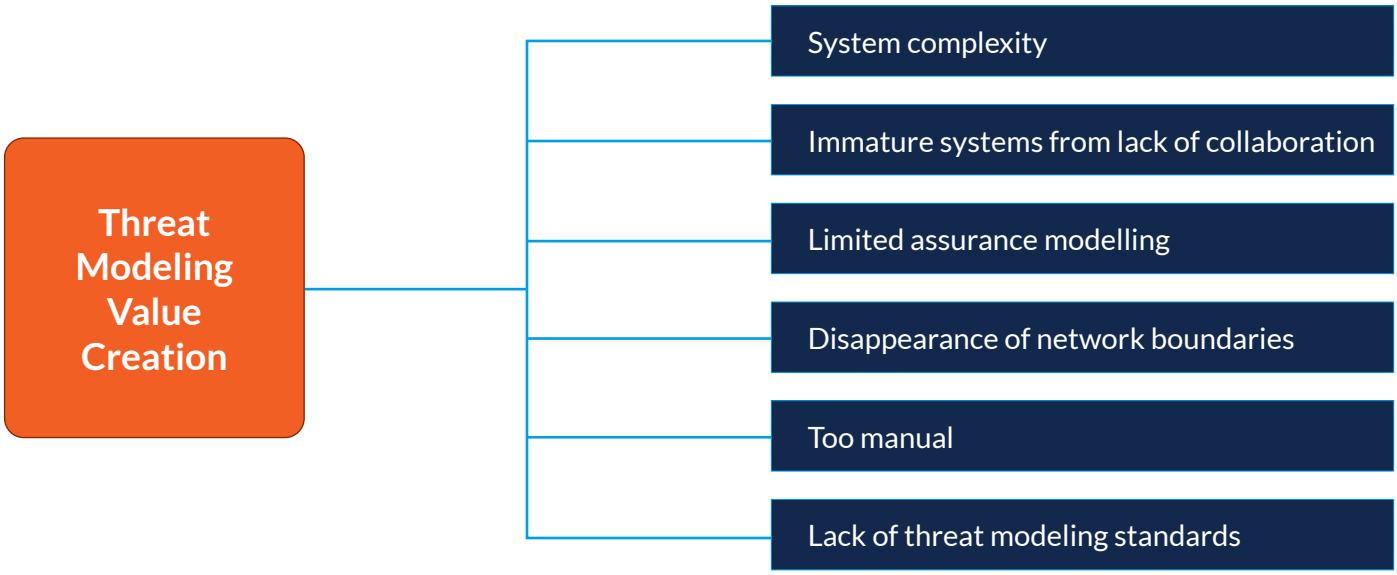
1. What hinders or promotes speed and security during secure coding?

2. What hinders or promotes speed and security during threat modeling?

This whitepaper expands upon these questions. It helps us, as a security community, better understand what we can do to help our DevOps delivery teams. Our objective is to enable them, not get in their way.

Threat Modeling

Secure design represents the essential foundation against which your secure coding activities can be performed. It provides constraints for your code and, ultimately, your solution. Threat modeling ensures that a particular design is as secure as possible; however, there are many challenges when you consider how threat modeling facilitates value creation as it balances security with speed. The following diagram represents what we found through our research.



System sprawl and complexity

In today's world, virtualization and containerization is normative. Containers, for example, are a great way to achieve scalability and process isolation; however, because of the ease with which containers can be spawned, the complexity of architecture continues to grow. Easing the burden of the monolith and subsequent release from development into production has led to a more complex operational environment. You may, for example, have orphaned containers that were used in the past, but no one has shut them down. Consider also the case of ephemeral components. Having short-lived microservices introduces greater architectural complexity, and that puts a burden on having to manage security in such contexts.

This complexity makes it difficult to fully understand the security vulnerabilities across all of your systems, particularly those that are developed outside your organization by other communities or vendors.

Immature systems caused by lack of collaboration

Security is best performed in a collaborative manner. Limiting threat modeling to a small group of experts creates bias which might miss some critical weaknesses in your architecture. Systems are best threat modeled when multiple people are permitted to contribute and trade-offs are made appropriately. If this collaboration is weak or non-existent, a small group will develop mitigations based on their own understanding of the "right way." This leads to architectures with weaker security. **[Simone Curzi et al, 2021]** explicitly writes, "Lack of diversity in POVs. By not involving a broad group of stakeholders, there is a reliance on personal bias and assumptions about components or libraries based on previous experience."

If your systems are immature based on lack of collaboration in the early stages of design, that puts additional pressure on your security operations team to detect the vulnerabilities and quickly provide feedback to your developers for mitigation.

Limited assurance from modeling

Lack of assurance is one of the biggest challenges of threat modeling. Assurance stems from consistency and traceability. Every threat modeler, however, seems to have a slightly different recommendation for mitigating attacks. This lack of consistency among your threat modelers gives rise to the question about whether the mitigations being proposed will help secure your systems. From a value perspective, this erodes credibility of the advice.

At the present time, there is a movement within the DevSecOps community, to help developers conduct threat modeling. This, in our opinion, will only amplify the problem. Developers necessarily have a small view into their own code. Attempting to threat model in this context can lead to a lot of confusion across teams.

Disappearance of network boundaries

Historically, many solutions were developed around the idea that a network boundary provides the necessary security protection. This led to an assumption of implicit trust. These days, however, this assumption is being called into question. Complex supply chains and contractor partnerships make our network boundaries very fluid. **[The Open Group, 2021]** states, "Traditional, perimeter-based approaches built on legacy models of identity, authentication, and authorization do not meet the needs of a digital business environment. In this modern digital world with ever-evolving threats – such as phishing, social engineering, and particularly insider-threats – organizations must abandon the flawed assumption that networks, both internal and external, are secure."

This has a profound impact on how you perform threat modeling on your systems since we can no longer rely on a network centric security model. The lack of a threat modeling standard already makes the exchange of threat models difficult. Coupling this with an evolving security paradigm that shifts emphasis from the network into assets like data, applications, and APIs compounds the problem. You need to threat model with finer granularity. But making this scalable presents a problem.

Too manual

Threat modeling is typically based on creation of diagrams that try to assess potential attacks. This activity is largely manual which interferes and slows down the match faster-moving developer and operations processes. **[Simone Curzi et al, 2021]** states, "Updating these diagrams is relatively slow since it is largely a manual task. Revisiting a diagram requires re-contextualizing the system and key participants in the original threat modeling effort may already be on other projects." You need to address manual intervention if speed is part of the value proposition of threat modeling.

Lack of threat modeling standards

One of the biggest complaints we hear about threat modeling is that it lacks a common standard. Sure, there are standards, shared practices, and frameworks that are proponents of threat modeling. But specific standards related to the design, analysis, and output of threat modeling do not exist. This makes quality assessments difficult. It means you are reliant on qualitative expert opinions from a few individuals who, hopefully, have a strong understanding of the entire solution.

Without an objective standard, this can be difficult since the output from your threat modeling can still vary widely.

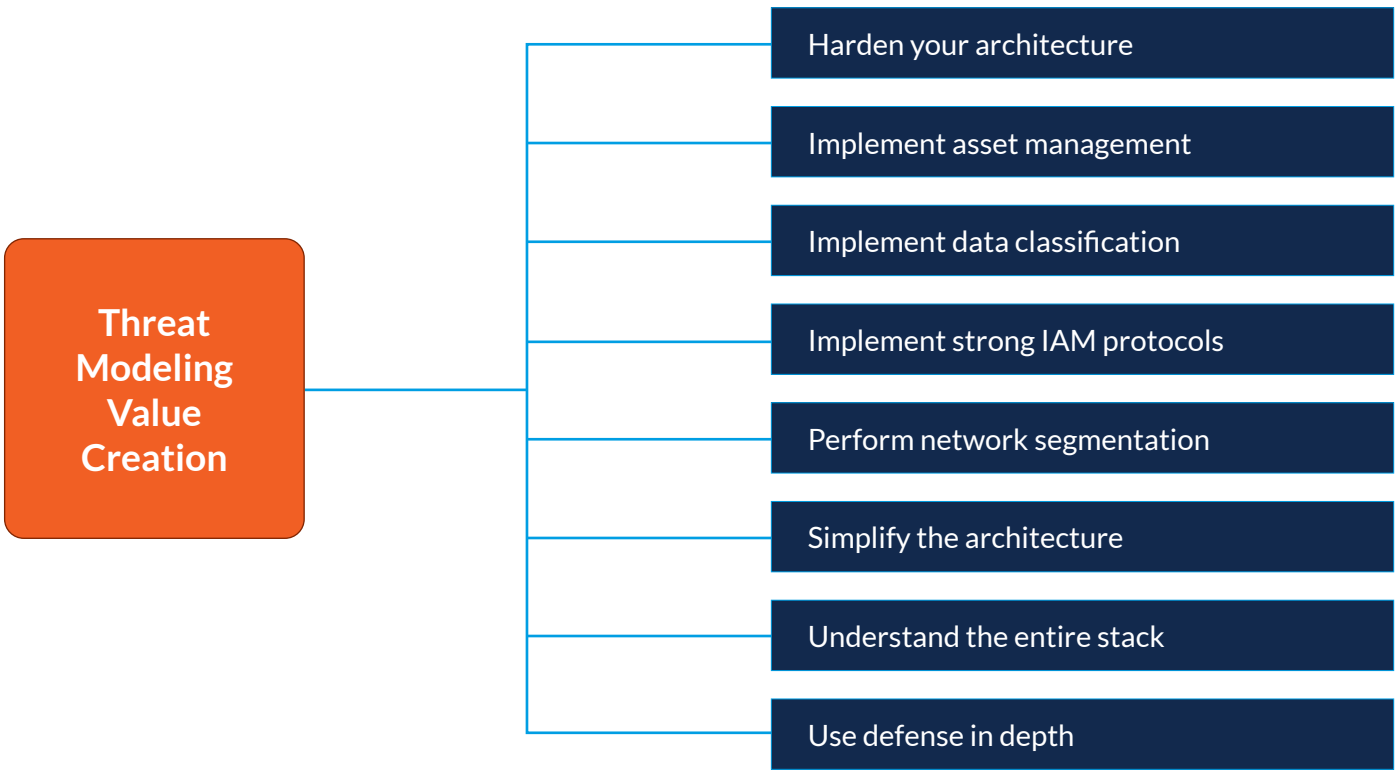
Despite these challenges, there are actions you can take in order to add value to your threat modeling process. Once again, the emphasis must be on balancing both speed and security. Our research revealed eight areas for improvement.

Harden your architecture

This implies using secure components in your architecture. Securing the components will be contextual. Depending on the level of security required, some components will require more security rigor than others. Having these secure components enables your DevOps teams to move much quicker because the hardened components have already been verified for security. But in order to secure those components, you need to know which components exist in the first place. And for that, you need asset management.

Implement asset management

As threat modelers understand various assets, they can assist DevOps teams by creating appropriate guardrails that secure coding can work within. At the present time, there is significant emphasis on Bill of Materials and Dependency Checking. There is some fine work being done in these areas (you can search for SPDX, CycloneDX, and dependency-check to find out more). By threat modeling your assets, you provide your downstream configuration management teams with an opportunity to update your threat model as they discover new vulnerabilities which may have been missed earlier.



Implement data classification

Protecting your data is the ultimate objective. Prioritizing what data to protect provides the necessary scope for threat modeling without wasting time on data that is not deemed as critical. Threat modeling with data classification in mind will help to reduce the load on your security teams and accelerate the threat modeling activity by focusing their efforts. This will also improve speed by reducing testing time against low priority data assets.

Implement strong IAM protocols

Controlling who has access to specific resources is essential. Reliance on an identity that is outside of the code assists developers by focussing less on security issues and more on coding. Threat modelers can recommend the security policies needed in order to achieve least privilege access within the system being developed.

Perform network segmentation

For security models that must be based on a network perimeter (like legacy systems), partitioning the network and trusted boundaries helps to minimize the attack surface. Reducing the network size also reduces the amount of code that needs to be tested for regression, thereby decreasing the lead time. It also provides the essential business continuity without resorting to a complete overhaul of the legacy system.

Simplify the architecture

Simplifying and rationalizing your architecture makes it easier to understand and identify security vulnerabilities through threat modeling. A simplified architecture also makes it easier for developers to understand the bigger picture and how they can develop various components that fit into that simplified architecture. The natural tendency of our architecture is for entropy to grow. Rationalizing the architecture on a regular basis ensures that security debt remains manageable over time.

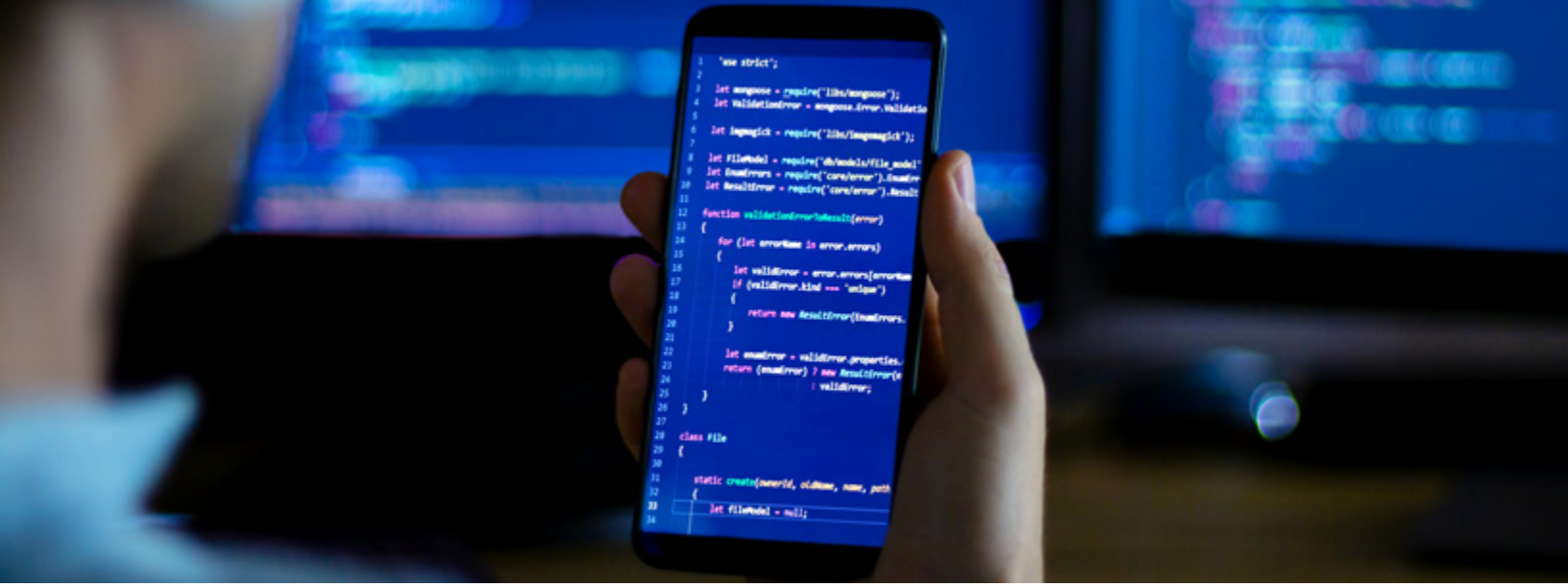
Understand the entire stack

If any part of your architecture is a black box, you will never be certain about the security of that component. Threat modeling will be limited to the periphery. Using trusted and verified components within your stack provides confidence in being able to generate the right controls to assist DevOps teams. Because developers will have guidance across the entire stack, they can proceed with greater confidence that security is being addressed.

Use defense in depth

Create security controls at various levels of your architecture with threat modeling. Some higher level controls may even reduce the need for lower level controls. This is good news because it can accelerate development time by allowing the lower, code level controls to be manageable and well-understood by your developers. This does not propose that developers do not need to interact with threat modelers or architects. On the contrary: with an understanding of how security is managed, a shared responsibility emerges.

The value matrix demonstrates how value against security and speed is created based on the suggested improvements.



Threat Modelling Value Creation	Threat Modeling Challenges					
	System Complexity	Immature systems from lack of collaboration	Limited assurance modeling	Disappearance of network boundaries	Too manual	Lack of threat modeling standards
Harden your architecture		SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY	SECURITY
Implement asset management	SECURITY SPEED	SPEED	SECURITY SPEED	SECURITY SPEED	SPEED	SECURITY SPEED
Implement data classification	SECURITY SPEED	SPEED	SECURITY SPEED	SECURITY SPEED	SPEED	SECURITY SPEED
Implement strong IAM controls	SECURITY	SPEED	SECURITY		SECURITY	SECURITY
Perform network segmentation	SECURITY	SECURITY	SECURITY		SECURITY	
Simplify the architecture	SECURITY SPEED	SECURITY	SECURITY SPEED	SECURITY SPEED	SECURITY	SECURITY SPEED
Understand the entire stack		SPEED	SPEED	SECURITY SPEED	SPEED	SECURITY SPEED
Use defense in depth	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED

Where SD Elements Can Help You Deliver Value in Threat Modeling

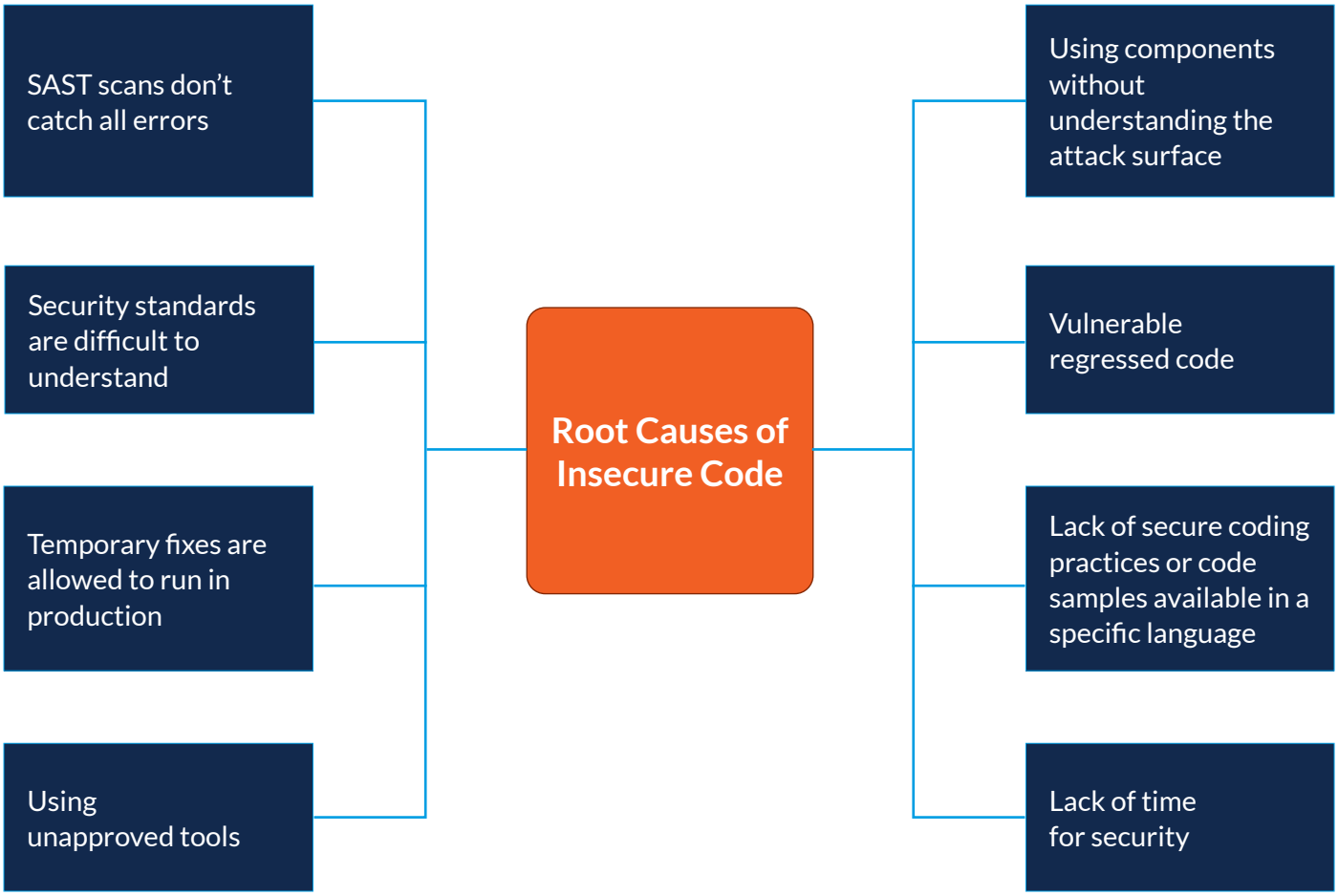
SD Elements can help your threat modeling deliver value in the following ways:
Challenges with Secure Development

SECURITY	SPEED
Hardening your architecture	
• Built in guidance on hardening your cloud infrastructure • A project-centric approach that decouples controls from network centricity • Reduces the time to create common mitigations for your applications and infrastructure	• Providing greater traceability and assurance through consistency of controls • Bidirectional integrations with DevOps pipelines thereby increasing the level of collaboration between teams
Implement asset management	
• Providing the security posture of your current application portfolio	• Reducing effort by pre-selecting controls based on given architectural components
Simplify the architecture	
• Provide a defensible argument for rationalizing your architecture against standards and regulations	• Pre-existing mappings between standards and recommended mitigations
Understand the entire stack	
• Using a project breakdown structure that allows you to examine the security gaps for each level	
Use defense in depth	
	• Supports defense in depth through mapping with standards and frameworks that help identify well known architecture weaknesses

Challenges with Secure Development

It behooves software developers to create code that is secure. Vulnerable code represents an entry point that can take a while to remediate while giving attackers an opportunity to exploit. As **[Adkond Rahman et al, 2019]** states, "We observe security smells can have a long lifetime, e.g., a hard-coded secret can persist for as long as 98 months, with a median lifetime of 20 months." Many teams focus on security once software has entered into a production environment. By then it is too late. **[Rod Cope, 2020]** agrees saying, "While there is – rightly – a big focus on securing software that is already deployed, the reality is that many future vulnerabilities stem from the creation of that software. Insecure applications give hackers a back door." As **[Rod Cope, 2020]** also states elsewhere: "The problem is that securing development is a tough challenge, due to the increasing complexity of software, the volume of code and other digital assets, multiple contributors, distributed teams and the pressure to deliver to tight deadlines."

Through our analysis, we found the following root causes for insecure development:



SAST scans don't catch all errors

Because code scanners don't catch all errors, downstream testing reveals code level bugs which have to be resolved much later in the software development life cycle. In our interviews, we discovered that developers feel pressured to make security decisions which they know little about. In such cases, they typically move on to another activity where the output is more predictable. This reduces the delivery speed because of the ambiguity of output from the tools being used.

Security standards are difficult to understand

For a software developer, trying to interpret high-level security guidance or policies can be problematic. There may be several ways to implement such policies. Resolving this ambiguity takes up precious time and slows down the development process. Despite the fact that many organizations rely on standards to guide their security programs, it needs to be recognized that developers need much more granular guidance around code level implementations. As **[Fabiola Moyón et al, 2020]** states, "...developers, quality engineers, and product owners face difficulties to identify security-relevant process artifacts as required by standards."

Temporary fixes are allowed to run in production

Because of the pressure to release quickly, code is modified in production environments as a way to quickly mitigate security concerns. The thought is that the production level changes will make their way back into the development environment; however, this leads to a discrepancy between production and development environments which could lead to vulnerabilities leaking as a result of regressed code.

Using unapproved tools

The pressure to deliver quickly often leads to the use of tools that improve developer productivity; however, these tools may not be best suitable from a security perspective. Therefore, even though developers end up moving fast, changes then need to be made later because of security concerns. This slows down the release.

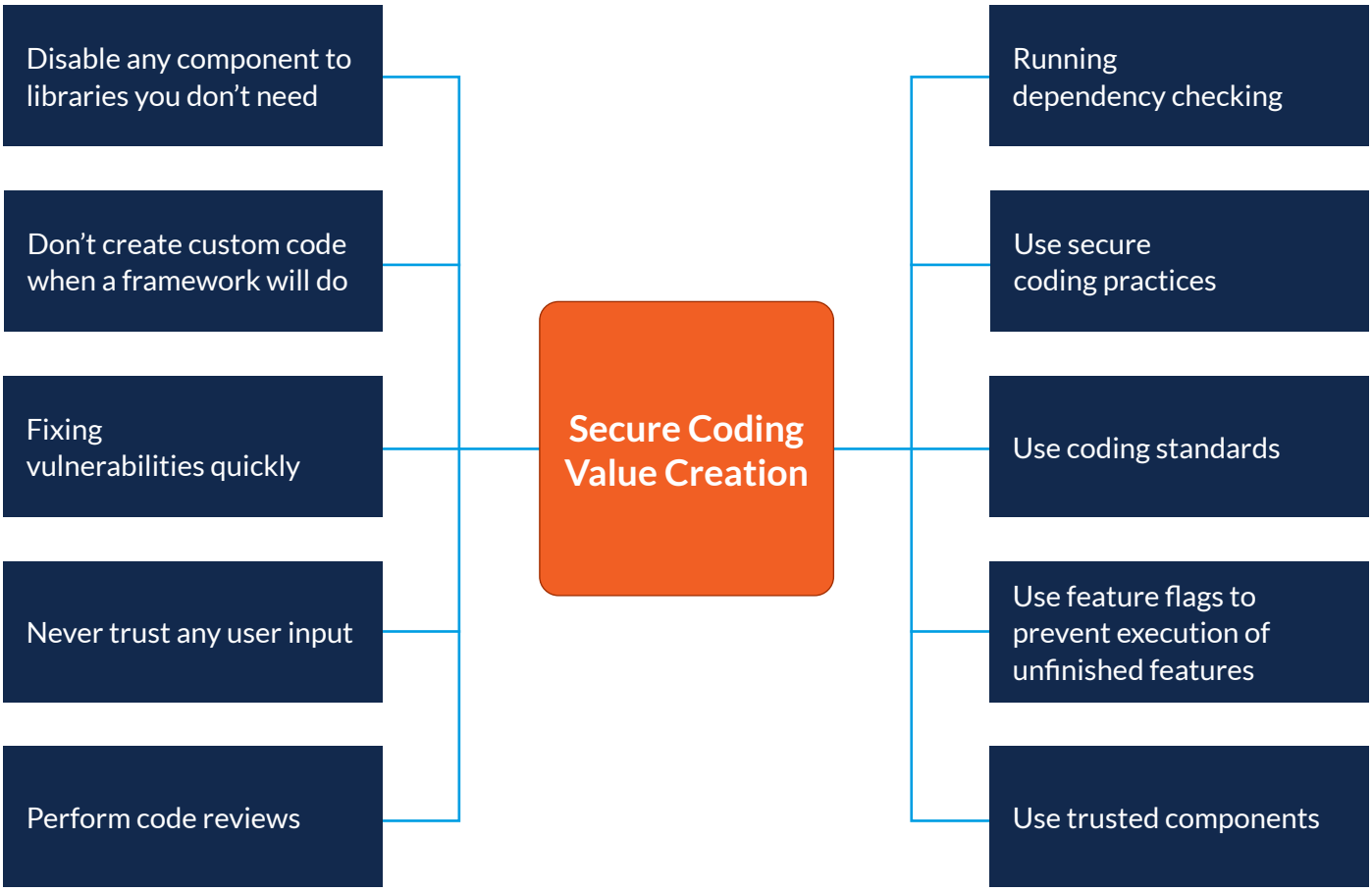
Using components without understanding the attack surface

Software development today is largely based on integrating various components. These components help us deliver quickly; however, using developer components blindly without understanding their security posture opens up your system to attacks. Developers rarely have the time to examine security databases like CWE or CVE to find out the vulnerabilities of a component.

Lack of secure coding practices or code samples available in a specific language

Because developers understand coding much better than security, they often seek out code samples to help them improve the security of their code. If this code is not available, the developer must figure out how to do it on their own.

To resolve this, focus on value creators.



Disabling any component to libraries you don't need

Removing or disabling any components you do not need reduces the potential attack surface. As **[National Cybersecurity Centre]** states, we need "...clearly defined and tightly constrained communication between components". By constraining the interaction between components, we can remove the libraries that are no longer required.

Don't create custom code on a framework will do work

Resist the temptation to create your own custom security framework when there are so many other frameworks that do a much better job. As **[National Cybersecurity Centre]** states, "Building something bespoke, when there are a variety of commodity options you could leverage, is not something you should do lightly."

Fixing vulnerabilities quickly

Many times, when a vulnerability has been discovered in a production environment, it takes a long time before that vulnerability is remediated. The feedback loop from production environments into developer workflows must be fast so that these vulnerabilities can be addressed quickly.

Perform code reviews

Engaging in code reviews provides the necessary quality checks as the code is being developed. Not only that, but it provides an opportunity to train your developers to think from a security perspective.

Running dependency checking

Running a dependency check ensures that the frameworks and components which your developers use can be tested and patched.

Use secure coding practices and standards

When your developers code, they should be thinking about security. This might mean constraining the use of the coding language in order to prevent vulnerable code from being deployed.

Use trusted components

While it may be tempting to use a component that improves developer productivity, resist the temptation. Instead, use trusted components from suppliers that are well-known and trusted.

Develop misuse cases

Developers often think in terms of use cases and functional requirements. They should extend this line of thinking to include misuse cases where they consider how a system might be compromised.

The value matrix demonstrates how value against security and speed is created based on the suggested improvements.

		Secure Coding Challenges							
		SAST scans don't catch all errors	Security standards are difficult to understand	Temporary fixes are allowed to run in production	Using unapproved tools	Using components without understanding the attack surface	Vulnerable regressed code	Lack of secure coding practices or code samples available in a specific language	Lack of time for security
Secure Coding Value Creation	Disable any component to libraries you don't need	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY	SECURITY SPEED
	Don't create custom code when a framework will do	SECURITY	SECURITY SPEED	SECURITY SPEED	SPEED	SECURITY SPEED	SECURITY	SECURITY SPEED	SECURITY SPEED
	Fixing vulnerabilities quickly			SECURITY			SECURITY SPEED		
	Never trust any user input		SPEED			SPEED			
	Perform code reviews	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY	SECURITY SPEED	SECURITY SPEED	SPEED	
	Run dependency checking	SECURITY	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY
	Use secure coding practices	SECURITY SPEED	SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED		
	Use coding standards	SECURITY SPEED	SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED		SECURITY
	Use trusted components	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED	SECURITY	SECURITY SPEED
	Develop misuse cases	SECURITY SPEED	SECURITY SPEED	SECURITY SPEED		SECURITY SPEED		SECURITY SPEED	SECURITY SPEED

Where SD Elements Can Help Your Secure Coding

SD Elements can help your secure coding deliver value in the following ways:

SECURITY	SPEED
Use secure coding practices	
• Reduce the amount of noise from code scanners because of proactive guidance	• In-situ secure coding training with full integration into the DevOps toolchain
Use coding standards	
• Knowledge base of well-known coding standards	• Mapping between coding standards and code examples
Develop misuse cases	
• Expertly curated advice that already takes into account common misuse cases	

Conclusion

[Rajavi Desai et al, 2021] states, "The most common challenges that any organization could encounter while implementing DevSecOps can be categorized into these three areas: speed, collaboration, and integration." We tend to agree. If we are going to create secure software, we must focus on automation for speed, and integrating secure design and coding for better collaboration. Threat modeling represents the longer term perspective focused on the design of your software, while secure coding represents the immediate, short-term view around fast delivery. You need both. Achieving this effectively means you need to understand the blockers in achieving the potential value of these activities. In this paper, we have shared our own experience of identifying the challenges and mitigations to resolve them.

By introducing collaboration between threat modeling and secure coding activities, you start to capture potential vulnerabilities at a much earlier stage. This gives you an opportunity to address these vulnerabilities long before they enter into production. Tools like SD Elements can help you in achieving value creation in your threat modeling and secure coding. In fact, as **[Rakesh Kumar et al, 2020]** mentions, "The preconceived notion that security implementation delays the development and delivery time can be addressed through automation of security requirements fulfillment in the adopted processes and practices." The reality, in fact, is quite the opposite. Tools like SD Elements automate the fulfillment of security requirements and facilitate the change needed with integrating security and DevOps teams.

Bibliography

- Akond Rahman et al. “The Seven Sins: Security Smells in Infrastructure as Code Scripts”, 2019.
- Anders Wallgren. “DevSecOps: 9 ways DevOps and automation bolster security, compliance”, 2021.
- Anna Koskinen. “DevSecOps: Building Security into the Core of DevOps”, 2020.
- Chris Romeo. “The 3 most crucial security behaviors in DevSecOps”, 2021.
- Cindy Blake. “Reducing risk with end-to-end application security automation”, 2020.
- Simone Curzi et al. “Evolving Threat Modeling for Agility and Business Value”, 2021.
- DORA, “State of Devops 2019”, 2019.
- Ericka Chickowski, “Seven Winning DevSecOps Metrics Security Should Track, 2018.
- Fabiola Moyón et al. “Using Process Models to understand Security Standards”, 2021.
- Fabiola Moyón et al. “Integration of Security Standards in DevOps Pipelines: An Industry Case Study”, 2021
- Fabiola Moyón et al. Integration of Security Standards in DevOps Pipelines: An Industry Case Study”, 2021.
- Fabiola Moyón et al. “How to Integrate Security Compliance Requirements with Agile Software Engineering at Scale?”, 2021.
- Fabiola Moyón et al. “A Light-weight Tool for the Self-Assessment of Security Compliance in Software Development – An Industry Case”, 2020.
- Faheem Ullah et al. “Security Support in Continuous Deployment Pipeline”, 2017.
- Hasan Yasar. “Implementing Secure DevOps assessment for highly regulated environments”, 2017.
- Jeff Schilling. “Diving into DevSecOps: Measuring Effectiveness & Success, 2018.
- Julie Peterson. “DevSecOps – All You Need To Know”, 2021.
- Kim Carter. “Francois Raynaud on DevSecOps”, 2017.
- Laurie Williams. “Continuously Integrating Security”, 2018
- Mary Sánchez-Gordón et al. “Security as Culture”, 2020.
- Meera Rao. “Building your DevSecOps pipeline: 5 essential activities”, 2017.
- Nataliya Shevchenko. “Threat Modeling: 12 Available Methods”, 2018.

- National Cybersecurity Centre. “Secure design principles”, https://www.ncsc.gov.uk/collection/cyber-security-design-principles
- Rajavi Desai et al. “Best Practices for Ensuring Security in DevOps: A Case Study Approach”, 2021.
- Rakesh Kumar et al. “Modeling continuous security: A conceptual model for automated DevSecOps using open-source software over cloud (ADOC), 2020.
- Rod Cope. “Strong security starts with software Development”, 2020.
- Saima Rafi et al. “Prioritization Based Taxonomy of DevOps Security Challenges Using PROMETHEE”, 2020.
- Teemu Laukkarinen et al. “Regulated Software Meets DevOps”, 2018.
- The Open Group. “Zero Trust Core Principles”, 2021.
- Wenjun Xiong et al. “Threat modeling – A systematic literature review”, 2019.

Security Compass

Go Fast. Stay Safe.

Security Compass, a leading provider of cybersecurity solutions and advisory services, enables organizations to build more secure software faster. With their flagship product, SD Elements, the company helps automate significant portions of proactive manual processes for security and compliance that improves time to market for new technology. In addition, they offer advisory services on how organizations can embrace emerging technologies like cloud to strengthen their security posture. Security Compass is the trusted solution provider to leading financial organizations, technology enablers, and renowned global brands. The company is headquartered in Toronto, with offices in the U.S. and India. Follow Security Compass on Twitter @securitycompass or visit them at securitycompass.com to learn more

1.888.777.2211
info@securitycompass.com
www.securitycompass.com

@SECURITYCOMPASS
SECURITY COMPASS

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

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

CALIFORNIA
600 California Street
San Francisco, California
USA 94108

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