FEATURE

Bridging the Gap Between Policies and Execution in an Agile Environment

Do you have something to say about this article?

Visit the *Journal* pages of the ISACA® website *(www.isaca.org/journal)*, find the article and click on the Comments link to share your thoughts.

https://bit.ly/2JJcOKs

With the perpetual occurrence of high-profile attacks and data breaches caused by software vulnerabilities, a new trend known as secure by design ("shifting left") has gradually shaped the software world.[1, 2, 3] It is easier today to convince strategic governance teams that nonfunctional requirements such as security requirements, are equally as important as functional requirements. After-the-fact security activities such as patching and integrations have proven to be much more expensive and less effective than incorporating security requirements into the early stages of design.[4]

However, designing applications with security at the forefront raises new challenges. Organizations must comply with an array of regulations and standards based on factors such as their sector, location, whether they deal with personal data and more. The cost of noncompliance can be much higher than the cost of a proactive approach to integrating standards and regulatory requirements into design and development processes.[5, 6]

At the same time, development life cycles are becoming shorter, and software releases are becoming more frequent. Traditional and linear software development processes (e.g., waterfall models) are being replaced by Agile processes. Moreover, with the advent of DevOps practices where traditionally separate business units now work closely together using Agile methods, the boundary between development and operations has become blurred. Practices such as continuous integration/continuous development have grown popular among software development teams. Exacerbating this complex environment is the push toward automation to minimize the latency of releasing new software features.

Mina Miri
Is a security researcher at SD Elements/Security Compass. She is particularly attuned to the need for applications to have well-developed security characteristics. In her current position, she researches various security and privacy contexts for securing software all throughout its life cycle. Miri has published articles in the *ISACA® Journal* and IAPP *Privacy Tech*, and she has presented at the Open Web Application Security Project (OWASP) AppSec conference.

Amir Pourafshar, CISSP
Is a senior security researcher at SD Elements/Security Compass. He has more than seven years of information security research experience ranging from malware behavioral analysis to formulating secure software development life cycle practices. Pourafshar contributes to global forums such as PCI Security Standards Council (PCI SSC) and SAFECode and presents at conferences such as OWASP AppSec.

Pooya Mehregan, Ph.D.
Is a security researcher at SD Elements/Security Compass. He has more than a decade of academic and industry-related experience in software and information security. His main areas of research are in application security, access control and privacy. He has published and presented at academic venues such as the ACM Conference on Computer and Communications Security (CCS), the ACM Symposium on Access Control Models and Technologies (SACMAT), and the IFIP WG 11.3. Conference on Data and Applications Security and Privacy (DBSec).

Nathanael Mohammed
Is a technical writer at SD Elements/Security Compass. He specializes in communicating about technology, with a focus on security and privacy. He has been involved with projects concerning EU General Data Protection Regulation requirements in Agile software development, and he has published an article on a tagging approach to privacy impact assessments in IAPP *Privacy Tech*.

There is a dilemma that forms between these security by design and Agile software development and deployment phenomena: Security requirements are considered disruptive to Agile practices by the vast majority of the software community.[7] Therefore, there needs to be a system of injecting actionable security requirements into the short development cycles of Agile processes. In addition to this, this system needs to bridge the gap between the policy space and the execution space. This gap is created when the requirements of policies, regulations and standards are too high level and abstract, which causes the process of extracting actionable tasks from them arduous, if not impossible. Systems that provide these translations are known in the security community as policy to execution (P2E) platforms.[8, 9, 10] At the moment, only a handful of these platforms have been developed, but their numbers are growing rapidly.

> “THERE NEEDS TO BE A SYSTEM OF INJECTING ACTIONABLE SECURITY REQUIREMENTS INTO THE SHORT DEVELOPMENT CYCLES OF AGILE PROCESSES.”

Regulatory bodies continue to publish new security and privacy regulations and standards, such as the EU General Data Protection Regulation (GDPR) and the recently published Payment Card Industry Software Security Framework (PCI SSF), in an attempt to accommodate the ever-changing software landscape. The gap between policy and execution is widening because policies mandated by regulations and standards have become more abstract by requiring secure processes around the software development life cycle (SDLC), rather than providing straightforward, actionable tasks. This could be an attempt to render regulations and standards persistent in the face of a changing environment. Testing this hypothesis, however, requires a separate study and falls outside the scope of this work. Instead, the following is an approach for staying compliant with new and ever-changing regulations in an Agile SDLC environment with minimal business disruption or slowdown.

Research Methodologies for Consolidating Security Controls

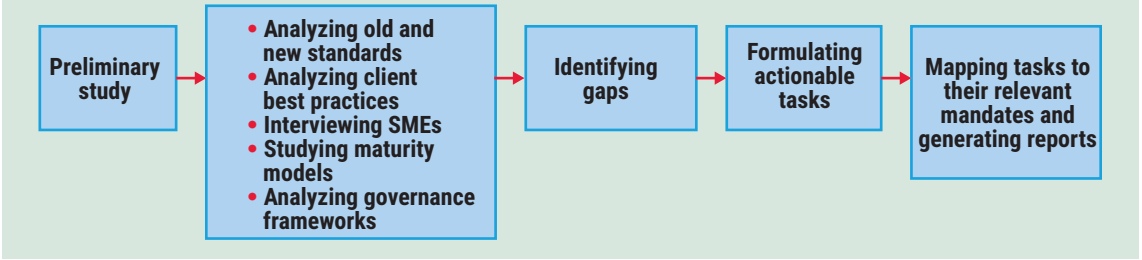
Figure 1 provides a high-level overview of the approach used in the study presented herein. After a preliminary examination of sources and required information, an extensive literature review was conducted to gather information about existing workflows for addressing the governance of technologies in organizations.

Existing standards from external (e.g., compliance regulation) and internal (e.g., internal policies) sources and industry best practices were analyzed to gain insight about current controls in tactical governance. Subsequently, a set of interviews was conducted with subject matter experts (SMEs) to extract information about their experiences in completed projects. The interviews consisted largely of open-ended questions about existing processes. The results from SME interviews were then collected to verify consolidated controls from the aforementioned sources. In the next step, actionable controls were formulated from analyzing all collected resources.

Principles for Addressing Business and Security Needs

This study is based on three fundamental principles for success. One is that the proposed process can

Figure 1—Overview of the Studied Approach



be repeated for existing and future standards without slowing down the production pipeline. The second is that the process should be minimally disruptive to developer workflows and easily integrated into their day-to-day activities. The third is that the process should be seamlessly adaptable to organizations of any maturity level and size without changing the structure of production to make the process functional. Taking these three principles into account allows for a unique solution in addressing the policy-to-execution gap.

Use Cases

In this study, PCI SSF is analyzed as a recently published compliance regulation.[11] In January 2019, the PCI Security Standards Council (PCI SSC) released two new PCI Software Security Standards as part of the new PCI SSF. These standards are the PCI SSC's efforts to better address the integrity of payment transactions and the confidentiality of all sensitive data as new technologies and software development practices emerge.

The Secure Software Life Cycle (Secure SLC or SSLC) Requirements and Assessment Procedures is a standard in the PCI SSF that offers security assessment guidance for both the development and operations life cycles. Secure SLC compliance aligns with Agile and continuous deployment methodologies to develop software faster and without requiring an assessment from a qualified assessor for each release.[12]

Historically, many organizations that handle credit cards, such as payment software vendors and payment providers, have been subject to PCI Data Security Standard (PCI DSS) compliance. PCI DSS requirements revolve around a product's technical security features and configurations for maintaining data integrity and confidentiality such as cardholder data encryption, strict access control and firewall configuration. However, PCI SSLC focuses on securing the software life cycle and building a secure software production and maintenance ecosystem regardless of the technology stack. This new need for compliance encourages strategic governance teams to recognize that, in addition to data security, application life cycle security is a business requirement. This business requirement mandates that product teams at the tactical governance level look for a new framework to build secure software life cycle capabilities. However, due to the widening gap between new requirements and traditional technical practices, it is not trivial to translate high-level strategic governance mandates and policies into step-by-step guidelines for building new practical processes for technical product teams. Moreover, adding a variety of emergent tools, frameworks and software development techniques to the endeavor renders it overwhelming.

In the first step of the proposed framework, PCI SSLC guidelines are analyzed and compared to existing best practices. This analysis helps identify gaps in the currently implemented controls. For example, section 4.1 of PCI SSLC requires a mature process for security testing that aims to determine the existence and emergence of vulnerabilities. While existing best practices aligned with traditional standards and business requirements advise utilizing static application security testing (SAST), they do not require a proper process for identifying the appropriate tool, the practical integration of those tools into the application development and deployment pipelines, or the proper management of vulnerabilities.

Although the new standard mandates the addition and governance of more processes and activities to an application's life cycle, it does not provide a code of practice or set of guidelines for implementation.

Therefore, tactical governance teams may choose any of the decisions from **figure 2** to address this gap. Each of these decisions leads to detrimental consequences.

Next, SMEs such as a development manager, application security (AppSec) advisor and security verification engineer are interviewed. Each of them provides their experience and perspective of the secure SDLC in Agile environments. Having all information from both the compliance and execution sides, an extensive analysis is completed to improve existing practices and is organized into actionable processes. At one end of the spectrum, these processes are aligned with high-level compliance requirements and, on the other, they are compatible with production technology stacks. The outcome is imported to an existing P2E platform in the form of tasks that prescribe actionable steps to relevant roles for establishing and executing secure SDLC processes. The existing P2E platform, shown with an example in **figure 3**, facilitates mapping policy-level requirements to execution-level tasks. It also compiles a compliance report that lists actionable steps required to comply with particular sections.[13]

For example, PCI-SSLC section 4.2 requires establishing a mature process for identifying and fixing software vulnerabilities.[14] The proposed approach is used to add a new execution-level task to the existing P2E platform. **Figure 4** shows the content of this task.

As shown in **figure 4**, actionable steps are identified to establish and execute a process for finding vulnerabilities and fixing them using SAST tools. This new task, as well as others, can now be mapped to the sections of PCI SSLC that have been added to the P2E platform as a new compliance report. **Figure 5** shows an excerpt of the compliance report.

Enjoying this article?

- Read *Reasonable Software Security Engineering. www.isaca.org/reasonable-software-security-engineering*
- Learn more about, discuss and collaborate on information and cybersecurity in ISACA's Online Forums. *https://engage.isaca.org/onlineforums*

Figure 2—Unsystematic Approaches to Filling the Gap Between SSLC Policies and Executable Processes

Decision	Consequence
Assuming existing practices are sufficient	• Failing compliance audits • Risking security incidents
Building new practices internally	• Missing gaps in certain areas • Incurring huge expenses • Having a bias toward internal practices • Engaging in conflicts of interest
Adapting practices developed by other teams	• Dealing with inconsistencies • Managing inapplicable controls

Figure 3—P2E Platform

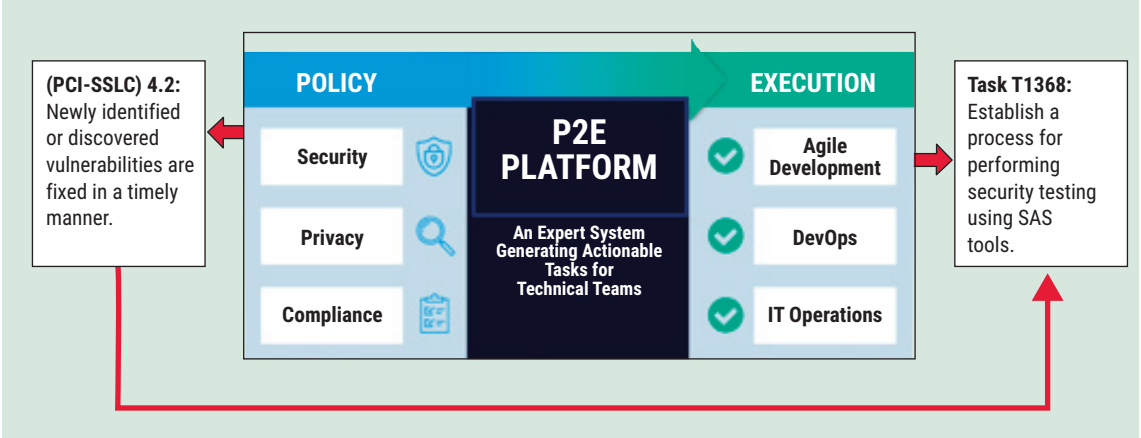


Figure 4—Section 4.2 of the PCI SSLC in the Form of a Task (Screenshot of the Task in the P2E Platform)

T1368: Perform security testing using SAST tools

Follow these guidelines for proper integration of a SAST tool into your SDLC:

- Choose a SAST tool appropriate for your software architecture (such as monolithic, service-oriented, micro-services, and so on), programming language, and development frameworks.
  - For example, configure and use OS/cloud configuration scanning, such as Microsoft Baseline Security Analyzer or Evident.IO, for cloud environments.
- Configure the SAST tool to include:
  - The entire code base
  - Configuration files
  - Third-party and open-source components
  - Shared components and libraries
- Execute SAST routinely at least in one of the following phases based on the maturity of the existing security controls:
  - Where applicable, add the SAST tool's plug-in to the developer IDE (development team should be highly mature)
  - Code commit
  - Unit, integration and regression testing
  - After staging release to scan static files and configurations of different components (development team is not very security aware but a security team handles the scan)
- Triage results and update the scanner profiles to reduce the number of false positives of the next scans. The following strategies help with reducing the false positives:
  - Interviewing the development team to figure out how mature they are from security perspective.
  - Suppressing certain low-level vulnerability categories if there are other means of proof. For example, if a team uses certain framework or library to cover that category.
  - Suppressing a false positive in a file as long as its hash value has not changed since the last scan.
  - Communicating the true findings with developers.
- Properly document and maintain an inventory of the scanning results and the corresponding actions taken to address the findings.
- Identify proper controls to permanently fix discovered vulnerabilities (true positives).

Figure 5—PCI SSLC Compliance Report for a Sample Payment Software

Section	Regulation Description		
	Task ID	Task Title	Status
Section 4.2	**Newly identified or discovered vulnerabilities are fixed in a timely manner. The reintroduction of previously resolved vulnerabilities is prevented.**		
	T1368	Establish a process for performing security testing using SAST tools	Incomplete
	T1369	Establish a process for performing security testing using DAST tools	Complete
Section 5.1	**All changes to payment software are identified, assessed, approved, and tracked.**		
	T1372	Establish and follow a software change management process	Complete
…	…	…	…

Conclusion and Next Steps

This approach seeks to bridge the gap between complying with requirements outlined in a regulation and determining actionable tasks using a policy-to-execution platform. This systematic approach can be repeated in similar situations where requirements in regulations are too high level and do not provide sufficient guidance for implementation. Though this example used the newly published PCI SSF, a similar approach can be adopted with other compliance regulations. The utility of an existing policy-to-execution platform was leveraged to mitigate the perceived disruption of security requirements for Agile and DevOps environments. Next, best practices with respect to each requirement given in the PCI SSF were compiled from the experience of organizations of varying maturity levels. Then, SMEs were interviewed to evaluate and augment the collected best practices in the previous step. The entire process led to a set of actionable tasks that correspond to the original requirements of a compliance regulation, which can be adapted to an organization of a given maturity level.

An ongoing project designed to automate the process of listing required tasks to reduce as much manual work as possible is underway. A pattern for designing such a method has already been created and is also under development. In this method, the P2E platform used in this study collects necessary information from different sources such as code scanner results and its own activity logs to assign a task to the responsible role within the required time frame. Once the undertaking is complete, the process will be tested on a live project for a better evaluation of the approach.

Endnotes

1 Lietz, S.; "<— Shifting Security to the Left," DevSecOps, 5 June 2016, *https://www.devsecops.org/blog/2016/5/20/-security*

2 The Open Web Application Security Project (OWASP) Foundation, Security by Design Principles, *https://www.owasp.org/index.php/Security_by_Design_Principles*

3 Schneier, B.; "Patching Is Failing as a Security Paradigm," *Motherboard*, 16 November 2018, *https://motherboard.vice.com/en_us/article/439wbw/patching-is-failing-as-a-security-paradigm*

4 Muresan, R.; "Costs of Non-Compliance Are Getting Higher," Bitdefender, 2 April 2018, *https://businessinsights.bitdefender.com/costs-of-non-compliance-getting-higher*

5 Merkulov, P.; "The Staggering Costs of Non-Compliance," *SC Magazine*, 1 May 2018, *https://www.scmagazine.com/home/opinion/executive-insight/the-staggering-costs-of-non-compliance/*

6 Johansson, D.; "Agile vs. Security: Resolving the Culture Clash," Synopsys, 5 May 2016, *https://www.synopsys.com/blogs/software-security/agile-vs-security/*

7 National Institute of Standards and Technology, Special Publication (SP) 800-55 Rev. 1, Performance Measurement Guide for Information Security, USA, July 2008, *https://csrc.nist.gov/publications/detail/sp/800-55/rev-1/final*

8 Mohamed, S. I.; "DevOps Maturity Calculator DMOC—Value Oriented Approach," *International Journal of Engineering Research & Science*, vol. 2, iss. 2, February 2016, *https://www.academia.edu/32117015/DevOps_maturity_calculator_Value_oriented_approach*

9 Johnson, M. E.; E. Goetz; "Embedding Information Security Into the Organization," *IEEE Security & Privacy*, May/June 2007, *www.ists.dartmouth.edu/library/352.pdf*

10 Sultan, K.; A. En-Nouaary; A. Hamou-Lhadj; "Catalog of Metrics for Assessing Security Risks of Software Throughout the Software Development Life Cycle," 2008 International Conference on Information Security and Assurance, Busan, South Korea, April 2008, *https://ieeexplore.ieee.org/document/4511611/*

11 Gray, L. K.; "Just Published: New PCI Software Security Standards," PCI Security Standards Council, 16 January 2019, *https://blog.pcisecuritystandards.org/just-published-new-pci-software-security-standards*

12 Security Compass, "How You Can Comply With The New PCI Software Security Framework," 17 January 2019, *https://blog.securitycompass.com/how-you-can-comply-with-the-new-pci-software-security-framework-f1013d4df0b7*

13 Miri, M.; F. H. Foomany; N. Mohammed; "Complying With GDPR: An Agile Case Study," *ISACA® Journal*, vol. 2, 2018, *www.isaca.org/archives*

14 Payment Card Industry Security Standards Council, "Secure Software Lifecycle (Secure SLC) Requirements and Assessment Procedures," January 2019, *https://www.pcisecuritystandards.org/documents/PCI-Secure-SLC-Standard-v1_0.pdf*