ICIT | Institute for Critical Infrastructure Technology

The Cybersecurity Think Tank

Integrating Cybersecurity into the Application Development Lifecycle

AN ICIT FELLOW PERSPECTIVE
MARCH 2021

Darren Death

ICIT Fellow and Vice President, Information Security and CISO, ASRC Federal

Contents

Introduction

Developing resilient code for business applications is critically important to ensure continued mission success by defending against the exploitation of new software vulnerabilities that can be leveraged by attackers to disrupt organizational stability (Sinha, 2019). Ensuring that newly developed code is free from exploitable vulnerabilities contributes to mission resilience by reducing an organization's overall attack surface (Petratos & Faccia, 2019). By reducing the total attack surface across an enterprise's digital inventory, an organization is effectively making it more difficult for an attacker to exploit and retrieve sensitive organizational assets (Karanja, 2017). (Karanja, 2017). This makes ensuring that cybersecurity requirements, principles, and tools are implemented across the application's System Development Life Cycle (SDLC), a top priority for development teams that are looking to defend their organization from malicious adversaries (Josang, Odegaard, & Oftedal, 2015). A developer can implement mechanisms to ensure that their application code is free from exploitable vulnerabilities (Thomas, 2018). These mechanisms include the integration of cybersecurity throughout the SDLC, application of best practices and standards, static code analysis, and dynamic code analysis (Reagin & Gentry, 2018).

The general business problem affecting organizations is that development teams are developing insecure applications that pose serious risks to the mission of business. The specific problem is that cybersecurity requirements and best practices are not integrated and operationalized as part of the System Development Life Cycle (SDLC) for software application projects (Josang, Odegaard, & Oftedal, 2015). Additionally, many development teams do not have personnel trained in cybersecurity best practices and security is too often the last stage or an afterthought in the development process. By adequately integrating and operationalizing cybersecurity requirements throughout the SDLC for an application development project, the organization can securely develop software with a more reasonable level of risk at the end of the development cycle.

A critical concern for software security is ensuring that the organization's business and mission requirements related to cybersecurity and resiliency are adequately accounted for as part of the functional and technical specifications for a new software development project (Dombora, 2016). Without adequately considering the organization's cybersecurity and resiliency needs, enterprise software will expose the organization to a level of risk that is outside of the company's risk appetite (Ogutu, Bennett, & Olawoyin, 2018). By not including the critical cybersecurity and resiliency requirements into a software development project, enterprise software will be developed that severely impacts the organization's ability to operate. Enterprise software plays a critical role in the performance of a company due to the value that information technology brings to the organization related to execution success and mission support (Mithas & Rust, 2016). Additionally, information technology is a crucial support

function that enables the organization's delivery of value to its customers (Porter, 2001). Software that has inherent material defects related to cybersecurity vulnerabilities because of inadequate requirements gathering and execution could result in a loss of confidence from constituents and a severe impact to an organization's ability to deliver value to its customers (Golgeci & Ponomarov, 2013).

An effective countermeasure to this risk is ensuring that cybersecurity requirements are integrated into any new software development activity at project initiation (Yayla & Hu, 2012). Integrating and executing security requirements early in the software development life cycle helps to ensure that any new software is developed and implemented in accordance with the organization's risk tolerance (Petruzzi & Loyear, 2016). Also, by adopting cybersecurity requirements early in the development process, the organization defend against the accumulation of technical debt in newly developed software (Granneman, 2018). If an organization waits to implement cybersecurity controls, the cost to implement those controls will increase over the life of the software package. The cost to implement a security control is more expensive later in a project as a software package will require rework to accept required cybersecurity configuration changes (Brown et al., 2010).

Cybersecurity and Application Development Integration

Integrating cybersecurity into all phases of the SDLC serves to ensure that cybersecurity capabilities are built into a new application development effort. Historically, ensuring that the cybersecurity team was included as part of technology projects has been challenging across many organizations (Kulkarni, 2018). Many reasons exist related to why individuals find it difficult to include cybersecurity as part of projects. In some cases, an organization's cybersecurity program may have been perceived as the group that denies access to technology rather than being an enabler of innovation. In other cases, working to implement cybersecurity requirements may have been seen as slowing down projects resulting in reduced protracted delivery and reduced performance. As a result of these historical views, cybersecurity leadership must ensure a positive relationship between the cybersecurity, application development, and other technical teams exists to safeguard effective integration of cybersecurity into the application development SDLC. Cybersecurity leadership must work closely with the technical teams to provide services and products that support the developer community related to guidance, tools, and resources that can be used to develop code in a secure manner (Kulkarni, 2018). The application developer must make use of effective best practices and standards that are specifically targeted at ensuring that application code is developed in a secure manner (Sołtysik-Piorunkiewicz & Krysiak, 2020). Utilizing industry best practices like the Open Web Application Security Project (OWASP) top 10, the developer can retrieve specific information on known vulnerabilities that are commonly coded into a business application that is, in turn, exploited by attackers for malicious purposes (OWASP, 2017).

Application Development Cybersecurity Tools

Technical tools related to application code analysis should be utilized by the development teams to ensure that developed code is secure and free from vulnerabilities. Two types of application code vulnerability tools should be employed by an organization to look for security issues that may reside in an application. The two types of code analysis that should be conducted are static and dynamic. Static code analysis looks at the source code of an application and inspects the code for security vulnerabilities (Thomas, 2018). Static code analysis should be conducted as part of the testing process associated with other quality assurance related testing functions within the application's SDLC (Williams, 2018). This serves to increase the efficiency of conducting a security review of application code by embedding the security process as part of the already existing development application review processes (Morales, Yasar, & Volkmann, 2018). Dynamic code analysis differs from static code analysis in that it operates against a live, production environment (Rahul, Kharvi, & Manu, 2019). Dynamic code analysis tools mimic the actions of a user attempting to access a business application. However, in this case, the dynamic scanning tool makes malicious access attempts to find potential security vulnerabilities that may exist in the application. Dynamic code analysis is appropriate to be implemented as part of the quality assurance process, as mentioned above, like application static code analysis (Williams, 2018). Unlike static code analysis, dynamic code analysis is well suited for the operations phase of the SDLC (Josang, Odegaard, & Oftedal, 2015). Dynamic code analysis can be integrated into the continuous monitoring plan of an organization and conducted as part of regularly planned penetration testing exercises (Abdullah, 2020).

Application Host Server Cybersecurity Tools

Part of protecting an application environment is ensuring that the underlying operating system that the application resides on is secure (Ahmed & Al-Shaer, 2019). The guidance above that relates to the integration of cybersecurity throughout the application SDLC, and the implementation of best practices and standards is entirely applicable to the implementation of IT systems. For operating systems rather than utilizing best practices like those from OWASP, a better source of information would be the Center for Internet Security (CIS) Consensus Security Controls (CSC) (CIS, 2019). This set of best practices provides the information technology team with a wide variety of operating system types and technologies to choose from with a focus on delivering the recommended configuration needed to operate those technologies securely. From an SDLC perspective, cybersecurity should be embedded from project initiation - through disposition for all new server environments that will host business applications to ensure that required best practices are applied to the new hosts (Josang, Odegaard, & Oftedal, 2015). The tools that are used to verify and test operating system security control effectiveness are different than the tools that are used to test business application code. Where business applications utilized static and dynamic code analysis to verify security control effectiveness,

host operating systems utilize automated vulnerability assessment tools to analyze and report on the vulnerabilities that may be present on the target system (Patel, 2019). An automated vulnerability assessment tool can be loaded with the organization's cybersecurity policies, which can be used to assess the host's systems security posture. The automated vulnerability scanner will report to the IT operations and cybersecurity teams both the security controls that are compliant with organizational policies and the configurations that fall out of the expected norms that have been established as part of corporate policy. An automated vulnerability scanner also interrogates the host operating system for patch compliance. Organizations' failure to implement required security patches is a well-known source of entry for a malicious attacker looking to gain unauthorized access to sensitive organizational information (FBI, 2018). By effectively utilizing an automated vulnerability assessment tool, the threat surface can be reduced for a host operating system serving to improve mission resiliency for an organization.

Managing Discovered Application and Host Vulnerabilities

It is essential to look at all digital systems across an organization's environment, including ones internal to the organization or outsourced to a third party (Patel, 2019). Any internal vulnerability should be assessed across all deployed platforms to determine their severity (Ogutu, Bennett, & Olawoyin, 2018). Examples of these deployed platforms include servers and server software (i.e., web server application software), workstations, and desktop applications (i.e., Application Integrated Development Environment (IDE)), networking equipment, Internet of Things (IoT) technologies, and web-based applications. Special care must be taken to implement tools and techniques that allow for the identification of vulnerabilities across all these platforms. Vulnerability identification responsibility is not limited to on-premise technologies. Cloud technologies must also be assessed for weaknesses that may be present within those implementations (Yimam & Fernandez, 2016). Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) implementations must be reviewed to ensure that all configurations initiated by the customer are applied correctly and do not expose the organization to unexpected risk.

An organization will find vulnerabilities throughout its collection of digital assets. As a result, it may be unreasonable to expect that vulnerability can be closed all at once. This is where the concept of triage comes into play. Vulnerability mitigation will typically follow two planning strategies as part of triage (Ogutu, Bennett, & Olawoyin, 2018). The first strategy focuses on vulnerabilities that have known exploits currently in the wild and can be leveraged over a network connection - these must be immediately mitigated. The second strategy focuses on implementing mitigation activities queued for remediation. A more methodical and planned approach can be used for an asset where physical access is required to exploit a vulnerability, and robust segmentation exists on the network to protect the asset (Granneman, 2018).

Continuous monitoring of the enterprise ensuring that previously discovered vulnerabilities are appropriately mitigated, and new vulnerabilities are removed promptly based on risk are critical activities to ensure a safe and resilient operating environment. Vulnerability and patch compliance tools must be implemented by the organization to address the diverse technologies deployed throughout the enterprise (Cain, Edwards, & Still, 2018). These tools can provide automated analysis for server and workstation operating systems, server software applications (ex: database and web server software), desktop applications (ex: Application IDE), network devices (ex: switches, routers, and firewalls), applications and databases (ex: dynamic and static code analysis), and cloud-based Assets (IaaS, PasS, and SaaS).

Authentication Considerations for Applications

Many models and frameworks exist to protect the authentication process used by applications. One of those frameworks is Multi-Factor Authentication. Multi-Factor Authentication is a security scheme that utilizes multiple methods to gather information about a user to authenticate the user to an application in a secure fashion. (Ibrokhimov et al., 2019). Factors that are utilized as a part of a Multi-Factor Authentication system include Possession, Inherence, Time, and Knowledge (Nguyen & Memon, 2018).

Possession relates to something that an individual has with them. In this case, it could be a physical authentication token or a user's smartphone. The item that is in the user's possession provides signaling that can be used by an application to identify the user (Ometov et al., 2018). An example of this signaling includes bidirectional communication as part of an app on a smartphone. In the case of a smartphone application, the application on the user's phone can be programmatically tied to a business application. When the user attempts to logon to the application, the smartphone application will ask the user to verify that it is them accessing the application. Once the user has accomplished this activity, the application can then permit the user to access the system.

Inherence is a quality about the user that is inherent to their physical being (Ometov et al., 2018). This includes biometric characteristics like the retina of the eye or a fingerprint. This system is similar to the above factor in technical execution. When a user attempts to login to an application the user submits biometric data to the application through a terminal that is configured to accept and relay the information. The MFA system then compares a stored digitized sample of the biometric data with the data that was just submitted by the user for login purposes. If a match exists, the user is granted access to the application.

Time is a specific window where a user is allowed to use a particular sign in code that is generated by a time-based one-time password generator (Chang-Seop, 2018). Time-based one-time passwords can be implemented via hardware tokens and applications that reside on mobile and desktop operating systems. In the case of time-based one-time passwords, a

randomly generated code on a physical token or application is created based on the time of day, and an algorithm that uses data that is tied to the user. A user is challenged by a business application for the current code that is displayed on the token or application upon login. The user would then enter the randomly generated code into a predefined input field in the application interface. The application would then verify that the code generated by the token is the one that is expected and the one that is assigned to the user. If both criteria are met the user will be successfully authenticated to the system (Agrawal et al., 2019).

Knowledge is a multi-factor authentication concept that includes information like an individual's username and password and is the most commonly used item that is paired with the above-mentioned factors. By combining factors like username and password and time-based one-time passwords a multi factor authentication scheme has been introduced that greatly increases the security of an underlying business application's authentications scheme (Nguyen & Memon, 2018).

Application Security Management Considerations

To effectively integrate the cybersecurity capabilities described above in support of application and host operating system security, management support is essential (Rothrock et al., 2018). A formal cybersecurity program must be established within an organization that is aligned with the organization's risk appetite in support of business-aligned security control implementations (Ogutu, Bennett, & Olawoyin, 2018). The executive leadership for an organization must ensure that cybersecurity is prioritized within IT implementations and that cybersecurity is integrated into all digital projects at project initiation. In this way, a new software application or digital technology can be evaluated for its security ramifications and appropriateness for the organization. Implementing the above-mentioned technical countermeasures with strong support from executive leadership through an appropriately resourced, and empowered cybersecurity program will serve to protect company assets from potential exploits related to software development (Karanja & Rosso, 2017).

Conclusion

Developing secure code for business applications is critically important to ensure continued mission success by defending against the exploitation of new software vulnerabilities that can be leveraged by attackers to disrupt organizational stability (Sinha, 2019). Ensuring that newly developed code is free of exploitable vulnerabilities contributes to mission resilience by reducing an organization's overall attack surface (Petratos & Faccia, 2019). By reducing the total attack surface across an enterprise's digital inventory, an organization is effectively making it more difficult for an attacker to exploit and retrieve sensitive organizational assets. This makes ensuring that cybersecurity requirements, principles, and tools are implemented across the

application development System Development Life Cycle (SDLC), a top priority for development teams that are looking to defend their organization's from malicious adversaries (Josang, Odegaard, & Oftedal, 2015). A developer can implement mechanisms to ensure that their application code is free from vulnerabilities. These mechanisms include the integration of cybersecurity throughout the SDLC, application of best practices and standards, static code analysis, and dynamic code analysis.

For a new software project to be implemented successfully, the cybersecurity and software development teams must be well integrated, serving to ensure that requirements are communicated, implementable, and deployed into production (Death, 2017). To ensure that cybersecurity requirements and risk management are a priority, an organization's leadership must be fully engaged to guard against the risk of insecurely developed software applications. This serves to ensure that all team members throughout the company understand that cybersecurity requirements are, in fact, business requirements that are supported by the uppermost leaders of the organization (Rothrock, Kaplan, & Oord, 2018).

About the Author

Dr. Darren Death is a proven technology leader with over 20 years of experience deploying enterprise systems for large private and public organizations. Death has led, designed, and implemented large-scale, organizational-wide enterprise IT systems with far-reaching impact.

Dr. Death currently serves on the EC-Council International Advisory Board for TVM (Threat and Vulnerability Management) and as the CISO and serves as the InfraGard Maryland – Cyber Threat Special Interest Group Chief and American Council for Technology / Industry Advisory Council (ACT-IAC) – Cyber Security Community of Interest Program Chair. He serves on the Board of Advisors and as faculty for the Cyber Intelligence Initiative at the Institute of World Politics. Death holds a master's degree in Cybersecurity and Information Assurance and a Doctorate in Information Technology - Information Assurance and Cybersecurity.

About ICIT

The Institute for Critical Infrastructure Technology (ICIT) is a 501c(3) cybersecurity Think Tank with a mission to cultivate a cybersecurity renaissance that will improve the resiliency of our Nation's 16 critical infrastructure sectors, defend our democratic institutions, and empower generations of cybersecurity leaders.

References

Abdullah, H. S. (2020). Evaluation of open source web application vulnerability scanners. *Academic Journal of Nawroz University, 9*(1), 47-52.

Agrawal, V., Paliwal, R. K., Sharma, P., & Shrivastava, A. (2019). *Web security using user authentication methodologies: CAPTCHA, OTP and user behavior authentication.* OTP and User Behavior Authentication (2019).

Ahmed, M., & Al-Shaer, E. (2019). Measures and metrics for the enforcement of critical security controls: a case study of boundary defense. *In Proceedings of the 6th Annual Symposium on Hot Topics in the Science of Security* (pp. 1-3).

Brown, N., Cai, Y., Guo, Y., Kazman, R., Kim, M., Kruchten, P., & Sangwan, R. (2010). Managing technical debt in software-reliant systems. *In Proceedings of the FSE/SDP workshop on Future of software engineering research (pp. 47-52).*

Cain, A. A., Edwards, M. E., & Still, J. D. (2018). An exploratory study of cyber hygiene behaviors and knowledge. *Journal of Information Security and Applications, 42*, 36-45. doi:10.1016/j.jisa.2018.08.002

Center for Internet Security (CIS) (2019). *CIS controls.* Retrieved from https://www.cisecurity.org/controls/.

Chang-Seop, P. (2018). One-time password based on hash chain without shared secret and re-registration. *Computers & Security, 75*, 138–146.

Death, D. (2017). *Information security handbook: develop a threat model and incident response strategy to build a strong information security framework.* Birmingham, UK: Packt Publishing.

Dombora, S. (2016). Characteristics of information security implementation methods. Management, *Enterprise and Benchmarking in the 21st Century, 57*.

FBI (2018, May 07). *2017 internet crime report.* Retrieved from https://www.fbi.gov/news/stories/2017-internet-crime-report-released-050718

Golgeci, I., & Y. Ponomarov, S. (2013). Does firm innovativeness enable effective responses to supply chain disruptions? an empirical study. *Supply Chain Management: An International Journal, 18*(6), 604-617. doi:10.1108/SCM-10-2012-0331

Granneman, J. (2018). The business guide to improving information security. *The Journal of Equipment Lease Financing (Online), 36*(3), 1-9.

Ibrokhimov, S., Hui, K. L., Al-Absi, A. A., & Sain, M. (2019). Multi-Factor Authentication in Cyber Physical System: A State of Art Survey. In *2019 21st International Conference on Advanced Communication Technology (ICACT)* (pp. 279-284). IEEE.

Josang, A., Odegaard, M., & Oftedal, E. (2015). Cybersecurity Through Secure Software Development. *IFIP World Conference on Information Security Education (pp. 53-63).* Springer, Cham.

Karanja, E. (2017). The role of the chief information security officer in the management of IT security. *Information and Computer Security, 25*(3), 300-329. doi:10.1108/ICS-02-2016-0013.

Karanja, E., & Rosso, M. A. (2017). The chief information security officer: An exploratory study. *Journal of International Technology and Information Management, 26*(2), 23-47.

Kulkarni, M. (2018). IT governance in global banks: Emerging models. *Vinimaya, 39*(1), 11-21.

Mithas, S., Rust, R. T., & University of Maryland. (2016). How information technology strategy and investments influence firm performance: conjecture and empirical evidence. *MIS Quarterly, 40*(1), 223-245. doi:10.25300/MISQ/2016/40.1.10

Morales, J. A., Yasar, H., & Volkmann, A. (2018). Weaving security into devops practices in highly regulated environments. *International Journal of Systems and Software Security and Protection (IJSSSP), 9*(1), 18-46.

Nguyen, T., & Memon, N. (2018). Tap-Based user authentication for smartwatches. *Computers & Security, 78*(2018), 174–186.

National Institutes of Standards and Technology (NIST) (2001). Security Requirements for Cryptographic Modules Retrieved From https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-2.pdf.

Ogutu, J., Bennett, M. R., & Olawoyin, R. (2018). Closing the gap: Between traditional & enterprise risk management systems. *Professional Safety, 63*(4), 42-47.

Ometov, A., Bezzateev, S., Mäkitalo, N., Andreev, S., Mikkonen, T., & Koucheryavy, Y. (2018). Multi-factor authentication: A survey. *Cryptography, 2*(1), 1. doi:10.3390/cryptography2010001

Open Web Application Security Project (OWASP) (2017). *OWASP top ten.* Retrieved from https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/

Patel, K. (2019). A Survey on Vulnerability assessment & penetration testing for secure communication. *In 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)* (pp. 320-325). IEEE.

Petratos, P., & Faccia, A. (2019). Accounting information systems and system of systems: assessing security with attack surface methodology. *In Proceedings of the 2019 3rd International Conference on Cloud and Big Data Computing* (pp. 100-105).

Petruzzi, J., & Loyear, R. (2016). Improving organisational resilience through enterprise security risk management. *Journal of Business Continuity & Emergency Planning, 10*(1), 44-56.

Porter, M. E. (2001). *The value chain and competitive advantage*. Understanding Business Processes, 50-66.

Rahul, B. S., Kharvi, P., & Manu, M. N. (2019). *Implementation of devsecops using open-source tools.*

Reagin, M. J., & Gentry, M. V. (2018). Enterprise cybersecurity: Building a successful defense program. *Frontiers of Health Services Management, 35*(1), 13-22. doi:10.1097/HAP.0000000000000037.

Rothrock, R. A., Kaplan, J., & Oord, F. (2018). The board's role in managing cybersecurity risks. *MIT Sloan Management Review, 59*(2), 12-15.

Sinha, S. (2019). *Bug bounty hunting for web security* (pp. 57-78). Apress, Berkeley, CA.

Sołtysik-Piorunkiewicz, A., & Krysiak, M. (2020). *Towards industry 4.0—current challenges in information systems* (pp. 127-141). Springer, Cham.

Thomas, T. W. (2018). *Security code review with static analysis techniques for the detection and remediation of security vulnerabilities.*

Williams, L. (2018). Continuously integrating security. *In Proceedings of the 1st International Workshop on Security Awareness from Design to Deployment* (pp. 1-2).

Yayla, A. A., & Hu, Q. (2012). The impact of IT-business strategic alignment on firm performance in a developing country setting: Exploring moderating roles of environmental uncertainty and strategic orientation. *European Journal of Information Systems: Special Issue: Information Systems Research, Education & Policy in the Mediterranean Region (MCIS), 21*(4), 373-387. doi:10.1057/ejis.2011.52

Yimam, D., & Fernandez, E. B. (2016). A survey of compliance issues in cloud computing. Journal of Internet Services and Applications, 7(1), 1-12. doi:10.1186/s13174-016-0046-8