Security Policy Presentation

CS-405-T5469 Secure Coding 21EW5

Michael Iyaomolere

Southern New Hampshire University

June 20, 2021

https://youtu.be/_p6IuM5HYiE

CS 405 Project Two Script

Complete this template by replacing the bracketed text with the relevant information.

Slide Number	**Narrative**
1	Hi Everyone, my name is Michael I am presenting the use of external testing methods to identify potential vulnerabilities by presenting the Green Pace security policy guide and to provide implementation guidelines and recommendations for maintaining it in the future
2	Here we see the Defense in depth and the topic I am identifying is Multi-factor authentication. Multi-factor authentication (MFA; encompassing Two-factor authentication or 2FA, along with similar terms) is an electronic authentication method in which a user is granted access to a website or application only after successfully presenting two or more pieces of evidence (or factors) to an authentication mechanism: knowledge (something only the user knows), possession (something only the user has), and inherence (something only the user is). MFA protects the user from an unknown person trying to access their data such as personal ID details or financial assets.
3	Security threat matrix can be divided into four as we see in this matrix. We have the likely, priority, low priority and unlikely. The like is is the loss of confidentiality, things like important documents and information, the priority falls under things like when the software is down. That should be one of the first things to bring it up, Low priority is in cases where we there are inadequate exception handling and loss of integrity might fall under unlikely, this might still happen.
4	In this slide, we will identify the 10 different types of security principles. • Validate Input Data - Obey the one-definition rule • Heed Compiler Warnings - Do not cast to an out-of-range enumeration value • Architect and Design for Security Policies - Do not attempt to create a std::string from a null pointer • Keep It Simple - Be careful using functions that use file names for

Slide Number	Narrative
	identification • Default Deny - Use a static assertion to test the value of a constant expression • Adhere to the Principle of Least Privilege • Sanitize Data Sent to Other Systems • Practice Defense in Depth • Use Effective Quality Assurance Techniques • Adopt a Secure Coding Standard - Handle all exceptions
5	We have different coding standards. Below are the 10 coding standards. • Data Type - Obey the one-definition rule • Data Value - Do not cast to an out-of-range enumeration value • String Correctness - Do not attempt to create a std::string from a null pointer • SQL Injection • Memory Protection - Do not call a deallocation function on anything other than nullptr • Assertions - Use a static assertion to test the value of a constant expression • Exceptions - Handle all exceptions • Input Output - Be careful using functions that use file names for identification • Concurrency - Do not destroy a mutex while it is locked • Containers - Use valid iterator ranges
6	I am now going to define different types of encryption policies. Encryption in rest - Encryption at rest is designed to prevent the attacker from accessing the unencrypted data by ensuring the data is encrypted when on disk. Encryption at flight - The process of encrypting data while the data is being transmitted. In some applications, such as remote replication, data may be unencrypted while it is at rest on drive arrays, but encrypted while it is being

Slide Number	Narrative
	transmitted to provide protection. Encryption in use - In-Use encryption takes a new approach that ensures that sensitive data is never left unsecured, regardless of or lifecycle stage (at rest, in transit, or in use) source, or location (on premise, cloud, or hybrid).
7	Now we will summarize the policies that support authentication, authorization and accounting. Authentication - Authentication is the act of validating that user are whom they claim to be. This is the first step in any security process. Authorization - Authorization in system security is the process of giving the user permission to access a specific resource or function. This term is often used interchangeably with access control or client privilege. Accounting - Accounting is the process of taking into consideration all things are require in the implementation of a secure application.
8	The diagram here shows some unit test which includes assertions.
9	Here we have a diagram of the automation summary
10	An effective DevSecOps pipeline ensures that security is baked in throughout the software development life cycle. Here, we explain each DevSecOps phase and suggest beneficial DevSecOps tools that can help safeguard and secure your software. A traditional DevOps pipeline has several distinct phases: Plan, Code, Build, Test, Release, Deploy, Operate, and Monitor. With DevSecOps, there are distinct security steps that happen during each of those phases. DevSecOps tools ensure that your code is free from coding errors and safeguarded against software security vulnerabilities at each phase of the software development life cycle. There are two commonly used DevSecOps tools: **DAST** A DAST tool “looks inside” an application and dynamically analyzes execution logic and live data. In addition, these types of tools provide DevSecOps pipelines

Slide Number	Narrative
	with the following benefits: • Analyzes the whole application as it runs, within the full system environment. • Attempts to break encryption algorithms from outside. • Verifies permissions to ensure the isolation of privilege levels.
11	The problem is enhancing the loop holes and gaps in a software security infrastructure The solution is to implement all the securities and policies to safeguard the application. The risk involved is exposure to hackers and the benefit is a secure software.
12	**Unpreparedness** With the increase in frequency and complexity of cyber incidents, organisations cannot afford to be unprepared anymore. Organisations must test their defenses before a breach occurs, and be ready to respond when required. Failing to expect and respond to breaches will come at a high cost as organisations struggle to resume business. **Unknown threats** To be prepared, organisations must know what the threats are. Knowing the enemy and assets available is key. Besides keeping abreast of the latest developments, organisations can also get intel from the Dark Web to know where their threats are. **Is it too late?** Attackers may have infiltrated an organisation's network and are just waiting for the right opportunity to strike. It is recommended that organizations conduct active threat hunting to intercept these attempts and stop attacks before they happen. Active threat hunting can be done if proper monitoring systems are in place, or via searches on the Dark Web to identify any weak links or exploited areas within the organisation.

Slide Number	Narrative
	Lack of monitoring In order to ensure that threats are identified early on, organisations have to ensure that they have the right monitoring solutions in place. Anomalous behavior on the network and endpoints must be flagged at the onset to minimize the organisation's vulnerability to attacks or fraud. **Incident handling** When incidents do occur, organisations must ensure that they manage the crisis properly. A detailed crisis response plan should be in place, and well-rehearsed during "peaceful times" to ensure that everyone is aware of their roles and responsibilities. Mishandling of incidents can result in much higher costs and reputational damage, from which it may be challenging to recover. **People risk** Employees can be an organisation's weakest link, but also its greatest defense. A malicious staff may sell confidential information, or even allow attackers entry into the organisation's network. An ignorant employee may even unknowingly leave an "open door" for attackers. However, an employee who is aware of the risks and educated about signs to look out for in a breach, is an organisation's first line of defense. Ensure that employees are familiar with the risks and responses. Recommendation: **Threat Modeling** Threat modeling provides a summary of possible attack scenarios, outlines the flow of sensitive data, and identifies vulnerabilities and offers potential mitigation options. This phase helps to address security vulnerabilities and improves the security knowledge of everyone on the team. **Scan** Scanning is the process of analyzing code to ensure that it is safeguarded from security vulnerabilities. This includes both manual and automated code review. AppSec tools — such as SAST and DAST — are used during this phase. This phase enables developers to address security vulnerabilities and bugs earlier in the software development life cycle.

Slide Number	Narrative
	Analyze During the Analyze phase, all of the collected data and metrics from the previous phases is reviewed to identify all of the security risks. Then, those risks are compiled into a list ranging from most to least severe. (Note: Some SAST tools — like Klocwork— are able to do this process automatically.) **Remediate** After identifying and organizing security vulnerabilities in previous phases, they are finally dealt with in the Remediation phase. Some DevSecOps tools — like SAST — can recommend solutions for the vulnerabilities, errors, and bugs that it has identified. This makes it easier to address security issues as they arise. **Monitor** Monitor refers to the process of tracking the identified vulnerabilities, the steps taken to mitigate and/or eliminate those vulnerabilities, and the overall status of the application's security. In addition, it may be beneficial to also track and manage the differences between the actual and target metric values. This helps to make informed data-driven decisions during the software development lifecycle.
13	All the coding standards and security policies should be adopted to prevent future problems.
14	Thank you so much.