Project Two: Security Policy Presentation

CS-405: Secure Coding

Sarah Brady

12/18/2022

YouTube link: <https://youtu.be/kMXtCZammEQ>

| **Slide Number** | **Narrative** |
| --- | --- |
| **1** | The following presentation is a Security Policy Presentation, compiled and narrated by myself, Sarah Brady |
| **2** | Defense in depth is the practice and study of compiling security, meaning it acts as multiple layers of protection. If the first layer of protection is breached, there is another layer of defense waiting. If an attacker manages to make it through the first line of defense, they will have to jump more hurdles before reaching the reason for the attack, whether that be for financial gain or simply a challenge they were presented with. However, It is very important to not take it too far because you still want to make sure your code is simple and readable. |
| **3** | When looking at the threats matrix, it is important to understand how they fall on a list of priorities. A risk with a high severity and high probability of occurrence needs to be taken care of immediately and would be categorized as “priority”. A risk with a lessened probability, but with a higher severity would be categorized as “likely”, and so on down to unlikely where the risk poses the least threat. |
| **4** | The 10 Core Security Principals set a framework for best practices in the successful creation of secure code. Each standard correlates to one or more principles which emphasizes the alignment of between our standards and principles. While each standard applies to one or two principles, many of our standards apply directly to many of our principles. |
| **5** | The coding standards are as follows:   * Do not cast to an out-of-range enumeration value * Use valid references, pointers, and iterators to reference elements of a container * Do not attempt to create a std::string from a null pointer * Do not store already-owned pointer value in an unrelated smart   pointer   * Properly deallocate dynamically allocated resources * Use a static assertion to test the value of a constant expression * Handle all exceptions thrown before main() begins executing * Do not alternately input and output from a file stream without an   intervening positioning call   * Do not invoke virtual functions from constructors or destructors * Value returning functions must return a value from all exit paths |
| **6** | The encryption polices include Encryption in rest, Encryption at flight and Encryption in use.  Encryption in rest means that stored passwords will be stored in a secure manner. All data must be encrypted to prevent unauthorized access. Encryption will change and stay up to date with technology changes. If an attacker gains access to a system with encrypted data but not the encryption keys, the attacker must decipher the encryption in order to read the data.  Encryption at flight means data communications will be done in a secure manner along with identification of the data recipient to prevent interception of data.  Encryption in use means data that is in use will be muddled at any point that is practical. Heavily used data will also be encrypted when in rest then unencrypted when being used. |
| **7** | The Triple A Polices include: authentication, authorization and accounting.  Authentication Identifies a user in order to administer a security policy. This occurs during the login stage. Some updated methods use multilayer step authentication.  Authorization Identifies the user and determines the user’s authorization. Also determined on a case-by-case basis along with device and location. It is a level of access that user has in a system, whether the user is reading, modifying ot deleting files in the database.  And Accounting logs each transaction which may need to be investigated for malicious activity if an attack or breach occurs. Accounting is very important as this determines where security threats, attacks or breaches initiated from and the cause. Every system will need to be logged in order for this be effective |
| **8** | Unit testing is the ability to test blocks of code for functionality based on parameters. As shown in the slides, the code blocks uses assert to ensure that the collection is not greater than or equal to the set max size. This could cause errors when running the code if the sizes do not meet the requirements. |
| **9** | Here we see the max size test. Here we conduct a unit test to verify that the max size test is larger than or equal to the given values, 0,1,5 and 10. |
| **10** | Here we see a collection capacity test. This tests for a new collection that is greater than or equal to 0. Then add the number of entries to ensure that the number capacities can keep up with the number of entries. |
| **11** | Finally, we see a resizing increases collection test which verifies that the resizing increases the collection. |
| **12** | In this slide we see an automation summary of pre-production and production, which leads us into our next slide. |
| **13** | The DevSecOps pipeline is a method of secure coding that enforces a security policy.  The pipeline is as follows:  Plan, Design and development  Code and build  Test for vulnerabilities  Vulnerability Scanning  Release  Log and audit  Automated Penetration Testing  Monitor and Detect  Automated Response  And Recover |
| **14** | Risks are always a potential concern when building code, but continuous security practices will keep those risks at bay. Following policies and continuing training is vital to a successful and secure system.  The risk of Automation means it can be prone to errors  However, there are Actions to take.  Always practice and be up to date with static analysis tools and static testing,  and follow and practice the coding standards and principles. |
| **15** | Here are my recommendations:  Continue to keep up with current security trends to maintain the proper level of security.  Follow security standards  Automate, automate, automate  Proper code testing to maintain that line defense |
| **16** | Throughout this presentation, we touched on the principles and standards of secure coding and have concluded that practicing and maintaining secure coding is vital for the protection of sensitive data and information. Understanding multilevel security as well as the tools of the DevSecOps pipeline along with the risks and benefits will lead to the protection that every organization strives for. |