Project Two: Security Policy Presentation

Tyler Hollingsworth

Computer Science Department, Southern New Hampshire University

CS-405: Secure Coding

Professor Aaron Demory

April 16, 2023

|  |  |
| --- | --- |
| **Slide #** | **Narrative** |
| 1 | Hello Everyone,  welcome to my security policy presentation for Green Pace. |
| 2 | A security policy is needed to maintain uniformity in all the development processes that a company uses to ensure security is correctly implemented within all the companies' systems. I would like to begin by explaining the practice of using Defense in Depth and its importance in a security policy.   * Our security policy is based on the defense in depth practice which is illustrated below * This is to ensure that there are multiple layers of overlapping security measures in place * By doing this we ensure that there are no overlapping holes or vulnerabilities in the system to which an attacker could exploit. |
| 3 | * Secure coding standards always come with a range of different severities, likelihoods, remediation costs, priorities, and levels. These aspects will determine the threat that each individual standard poses toward the system if they are not followed. This illustration shows how the security policy standards can be grouped into a threat matrix which determines what risks the developer is adding to the system by ignoring the policy standard. We can also use tools to automate checks which will find non-compliant code. |
| 4 | Listed on this slide we can see the 10 core security principles and where each of the coding standards fit into those principles. These principles are: validate input data, heed compiler warnings, architect and design for security policies, keep it simple, default deny, adhere to the principle of leash privilege, sanitize data sent to other systems, practice defense in depth, use effective quality assurance techniques, and finally to adopt a secure coding standard. As you can see, most of the security policy standards fall into the adopt a secure coding standard principle which shows the importance of having a standard which is the entire basis for a security policy to begin with. |
| 5 | This next slide shows the standards listed in order of risk priority according to the severity, likelihood, and remediation costs of that standard. Performing a risk analysis and putting the standards in order from the most at risk to the least at risk shows what the priority level for each of the policy standards are. |
| 6 | Hackers have multiple motivations for exploiting a vulnerability to attack a system and stealing data is among the top motivators. To prevent this, we need to secure the data using encryption techniques.   * Encryption at rest - Encryption at rest is the practice of encrypting stored data such as that held within a server. By encrypting stored data, it is protected from both attackers who gain unauthorized access to the servers and from the system administrators that would be able to read the raw data if it were unencrypted. * Encryption in flight - Encryption in flight is the practice of encrypting all the data that is being transferred. This protects the data from being intercepted and read by a man in the middle attack. * Encryption in use - Encryption in use is the practice ensuring data is never left unsecured regardless of stage or location. This protects all the data of the system from attackers and their exploits no matter if the data is being used, stored, or transferred. |
| 7 | The triple-A policy is another important practice which is used to authenticate a user, authorize what that user is able to do, and account for that users’ actions within the system.   * Authentication - This is the practice of verifying a user is who they say they are through login methods such as using a user id and password or through 2-step verification. * Authorization – This is the practice of using deny by default and least privilege access for all the system users. This means that all the new users added to the system will start with no access to any resources of the system and an administrator will then give them access to only the minimal resources that the user will need to be able to complete their tasks. * Accounting – This is the process of logging users' session statistics and usage information so that all system resource accesses and changes are carefully monitored. |
| 8 | Unit Tests is yet another important practice to implement in creating code to verify that the code meets the requirements and that it functions correctly. This unit test is an example of a positive one which checks to ensure that an empty vector can be added. |
| 9 | This is another example of a positive unit test that ensures that resizing the collection will in fact increase the collection. |
| 10 | It is just as important to add negative unit tests to the project such as testing whether an error condition works appropriately. In this example we are testing to ensure that a bad index throws an index out of bounds error. In this instance we are calling the 75th object in a collection when the collection only goes up to 50. |
| 11 | Here is another example of a negative unit test that ensures an exception is thrown when trying to resize the collection with a negative number. |
| 12 | In this slide we can see the development, security, and operations pipeline or the DevSecOps pipeline that is used to implement security into the development lifecycle of a project. This shows what the development process should look like throughout the pre-production and production stages. |
| 13 | * + The DevSecOps pipeline is an iterative process that begins with a pre-production phase consisting of assessing and planning the system; designing the system; building the system; and verifying and testing the system. Then there is the production phase which consists of transitioning and performing a health check; monitoring for and detecting event alerts or intrusions; responding to those event alerts and intrusions; and lastly maintaining and stabilizing the system.   + The DevSecOps pipeline builds security into the development lifecycle of a project and applies the principle we spoke about earlier of don’t leave security until the end.   + Some tools that can be used to help ensure that the policy is implemented in the DevSecOps pipeline are Clang, CodeSonar, Parasoft c/c++ test, Polyspace Bug Finder, Astree, Helix QAC, Coverity, LDRA tool suite, Axivion Bauhaus suite, Eclaire, and RuleChecker |
| 14 | There are always risks involved when not utilizing the practice of don’t save security until the end which will result in an increase of exploits due to the increase of vulnerabilities in the system. The benefits of implementing security at every stage of the development process is that it protects all the stakeholders of the system through means such as money, time, reputation, legal action, and most importantly safety.   * One of the main risks of waiting to implement security into a project is that it is exponentially more difficult to correct a problem once it is found later in the development process. Sometimes this results in a complete rework of the code that goes into the system. It is better to find and correct the error early before other functionalities and architectures are built on top of the erroneous code. * The benefit of not leaving security until the end means that the system will be more secure, it will have less vulnerabilities, and most errors will be caught sooner rather than later, saving time, money, and effort. |
| 15 | Some advice and recommendations that are not included in this security policy that I would give to new developers is to:   * + - Always communicate with stakeholders in an effective, efficient, and transparent manner.     - Always have a support network available for second opinions and to answer any questions     - Always ask probing questions to get more details and clarification if you are unsure of a requirement,     - Have a plan when exploits do occur and an action team to address the exploit     - Never cut corners for any reason |
| 16 | In closing:   * It is highly important to adopt and adhere to a company security policy so that everyone is on the same page * Pay particular attention to the ten principles and coding standards set forth by your company * Identify threats to the system and then rank them in order from the most critical to the least critical * Always use defense in depth, encryption, and Triple-A practices in development of your code * Use unit tests to automate and verify that the code you write behaves appropriately * Always follow the DevSecOps pipeline to integrate security into the development lifecycle * Lastly adhere to the tools that the company has approved for use   Thank You |