# Jacob Weidner

4/20/2022

<https://www.youtube.com/watch?v=eF4oRueIG0E>

# CS 405 Project Two Script

| **Slide Number** | **Narrative** |
| --- | --- |
| **1** | We will be going over the basics of the Green Pace Security Policy.  This policy is based around standards that will ensure that the software that is developed is reliable and secure. By ensuring that these two areas are met we can help ensure that are taking a posture of Defense in Depth. By ensuring that the program is secure, and that the data is appropriate for the inputs that we are requiring, we will ensure that we are keeping an initial line of defense to the systems that the software is operating on. |
| **2** | As we look at the Threat matrix there is four areas that threats can be categorized as. If we have a threat that falls under the “Likely” category it will normally be input errors that occur when a user inputs data. Though when the product is used by an ordinary user this is not a major issue. But this can be avoided by appropriate checking functions to ensure that the data is accurate. Under the Unlikely category would be issues that require more advanced knowledge of the software to exploit. Even with these two categories we can also break down further into either low priority or Priority categories. If a threat is a low priority, it won’t necessarily cause any issues with the system but could cause inconvenience and still needs to be fixed. As for Priority threats are threats that can threaten the integrity of the program or could allow access to systems that are not supposed to be access by unauthorized users or programs. |
| **3** | There are ten basic Principles  1. Validate Input data  Always check for appropriate data types during input  2.Heed compiler Warnings  Complier warnings can help to ensure data types and proper data type size  3. Architect and Design for Security Policies  If program is a search engine that searches a SQL database and a search for a “or” statements is done prior to sending the query it can prevent running into SQL injection  4.Keep it Simple  Handling the exceptions prior to the main functions allows for the main function to remain simple.  5.Default Deny  By denying incorrect data types by default will ensure no chance of inappropriate data to follow on systems  6. Adhere to Principle of Least Privilege  Do not access Freed Memory. By ensuring that freed memory isn’t accessed, and it is closed properly will help prevent the freed-up memory from allow malicious code to be put in its place  7.Sanitize Data sent to other systems  Similar to the checking search engine check. Checking for SQL injection commands will sanitize search data that is being sent to the data server  8.Practice Defense in Depth  By detecting errors when converting will allows us to add a layer of defense when data or objects is moved through functions in the program.  9. Use Effective Quality Assurance Techniques  A basic quality assure practice is using Asserts Appropriately. If Asserts are used with out being fully thought out it could lead to unexpected termination of the program.  10. Adopt Secure Coding Standard. |
| **4** | 1. Correct Data type for the intended use and data type has appropriate space required    * If data type is not checked for type and size program could terminate causing other unknown issues 2. Do not access freed memory Ensure    * Accessing freed memory could lead to an opening for malicious code 3. Check for “or” statements prior to running a query to prevent injections    * Not checking for query statements inside of a search could lead to malicious code being injected into statement 4. Properly use shared pointers    * By not properly using shared pointers access to stored memory could occur and undesired code could be place there. 5. Check that the values being put into data do not exceed data type limit    * Could lead to errors during program operations 6. Handling exceptions prior to main    * Allows for errors to be handled prior to main function operation. Ensuring that user input is not the cause of error 7. Detect errors when converting    * Ensure that proper conversions are performed. 8. Exceptions must be handled    * Un handled exceptions could lead to termination of the program 9. Do not Attempt to modify String literals    * By modifying string literals could cause runtime errors leading to early termination of the program 10. Use assert appropriately     * Misuse of the assert function could cause unintended termination points to be generated in code. |
| **5** | Data should be encrypted in all three states. This will ensure that there is always appropriate levels of security around secure data. Data siting on the system and not being used should be encrypted to ensure that incase of unintended access data is still in a secure state. Data in flight should be secure to ensure that if data is sent to an unsecure system that the data still retains a level of security so that no data is lost or damaged. Data in use should be encrypted to ensure that malicious software on a system isn’t able to access being used in programs running proper security protocols. |
| **6** | Authentication ensures that the users or software accessing a system or network has the appropriate permissions to get access. This ensures that there is a way to prevent the system from being access prior to Authorization occurring.  Authorization ensures that the user or software has the appropriate level of privilege to perform the actions they are trying to make.  Accounting ensures that there is a method to log and track actions that occurred on a system or in a program. |
| **7** | Not checking for proper data input could lead to wrong information being placed into a database. Such as if an appointment program were to allow for incorrected dates from being taken in. |
| **8** | If we set a limit to a variable, we need to check to ensure that the data being input will not go out of range, causing an issue with the operation of the program. The following unit test shows the checking length of a phone number |
| **9** | Ensuring data is properly checked during input will ensure that the data structures are properly populated. |
| **10** | Checking if the function works properly and takes a data input. The following unit test performs the data population. |
| **11** | The DevSecOps pipeline is a visual representation of the development life cycle. Starting at the planning to the maintaining of the software that was developed. One external tool we can use is the automate check performed by the compiler. These can be done during the build process and through the testing phase. Though most issues should have been discovered during the build phase of the pipeline any errors or messages can still come up and be handled during these processes. We can also use penetration testing during the transmission and health check phase to ensure that the software does not allow for malicious code to work or make it through the program on to other systems. |
| **12** | If a proper security policy is not implemented and not used during the entirety of the development process could lead to issues during the testing or release stages. If tests are designed during the initial design process and during the development process, proper unit tests can be developed to ensure all input data is proper and all functions are working properly. Then during the testing process the security of the data being brought into the system will be verified. If it is not done during the design or during the development process these functional tests could be missed and lead to added work during the testing phase of development. This would cause for added time needed for more testing and then lead to a delay in release of the program. One risk that could cause from developing tests early on is that it could lead to longer design and development time. But if those are accounted for there should be no delay in meeting release dates. |
| **13** | The policy does not provide specific criteria for the creation of unit tests or system security requirement checks. Both should be either developed or designated during the planning and design phase so they can be implemented during the build phase of the development. |
| **14** | By ensuring that proper secure coding practices are followed. We will ensure data entries are checked. If the data being enter whether it is set up by the program itself or is based on user input. It should always be verified that it meets the requirements of the program so that this data does not cause issues with the program or lead to issues in the connecting systems such as what could occur if a search program is not verified that it does not have a SQL Injection passing through to the database. |