# CS 405 Project Two Script

By: Gary Clark

| **Slide Number** | **Narrative** |
| --- | --- |
| **1** | Hello and welcome, my name is Gary Clark and I will be presenting an overview of my evaluation of the security policies at Green Pace as well as some recommendations for the future. I will touch on what the goals are, why we need to care, and what we can do about an improved data security policy. |
| **2** | To begin we must identify what makes a system secure. This diagram showcases one of the most well known and recommended strategies in the world of data security. Defense in depth layers multiple security features to prevent any single failure from compromising the system. The more layers, the harder a system is to breach. The key is striking a balance between dedicating resources to more layers while keeping the projects time restraints and goals in mind. |
| **3** | With this practice in mind, lets examine some common metrics for defining potential risks to security. This matrix places standards, which I will go over more in depth in the coming slides, into categories based on the likelihood of occurrence and the severity of a failure to follow procedure. The most severe issues and the top focus are those with a high likelihood of occurrence and high consequences from the occurrence. Conversely, the lowest priority are issues that are unlikely to occur, with little to no consequences of such an occurrence. |
| **4** | When categorizing the standards most applicable to Green Pace, there are 10 guiding principles for secure coding that must be considered. For example, best practice states that systems should deny entry by default and only grant access under very specific conditions like a valid username and password. Details of each of the principles can be found in the full security policy document. The chart on the screen maps the coding standards onto their applicable guiding principles and in the next slide we will discuss exactly what these standards are. |
| **5** | Here we have the ten coding standards for Green Pace to follow during the development process. They are listed in order of highest priority as defined by the threats matrix. We must sanitize data passed to complex subsystems, design functions should validate their parameters, handle all exceptions thrown before main() begins executing, guarantee that storage for strings has sufficient space for character data and the null terminator, avoid large stack allocations, never call a function with a mismatched language linkage, ensure that integer conversions do not result in lost or misinterpreted data, incorporate diagnostic tests using assertions, never attempt to create a std::string from a null pointer, and never invoke virtual functions from constructors or destructors. Again more details and examples of eac h standard can be found in the full report. |
| **6** | As an important tool in the protection of sensitive data, encryption must be appropriately utilized. This is especially important in systems like Green Pace where financial data and other protected classes of data are stored. Data in flight applies to the category of data being exchanged or transported to or from our system. This data must be protected from interception and modification, and many systems will use TLS Encryption to prevent this. Data at rest refers to any data stored within our system. It also applies to data which is entrusted to third party vendors. Client data must be protected from unauthorized access to comply with the Gramm-Leach-Bliley Act. The most common method to encrypt this data is using the AES protocol. The final type of data is that which is currently being used. It can be new data that is created, existing data that is modified, or data being viewed by stakeholders. While real-time encryption of the data can prevent external entities from hijacking a session, often the most effective measure is to ensure employees have the proper training and are following robust policies surrounding the usage of internal data. |
| **7** | Along with encryption, the Triple-A framework provides a structure consisting of three categories which must be implemented for a secure system. The three A’s are Authentication, Authorization, and Accounting. Before a user can access the system, they must authenticate their credentials which should be done through Multi-Factor Authentication and an internal VPN network. Authorization refers to the practice of providing levels of access to employees based on the needs of their job. When structuring these levels, Green Pace must follow the principle of Least Privilege, or in other words only grant the bare minimum access required for someone to do their work. Finally Accounting, is the practice of ensuring the first two A’s are being following properly and that security standards are upheld. Records of permissions and access should be stored offsite as to avoid internal tampering and third-party reviews should be periodically conducted to avoid bias. There are other practices which must be following in a secure system that overlap with one of more of the categories. The creation of a Security Log will help aid in the implementation of all three categories and is highly recommended by security experts. |
| **8** | To follow a policy, it is essential to have a method to validate the product meets the standards of the company. One method to do just that is by using Unit Tests. These are tests designed to verify functionality and behavior within a system. There are two types: positive and negative. Positive tests will verify that the program behaves as intended when receiving expected inputs or data. In contrast, a negative unit test verifies that the program can properly handle unexpected or unintended inputs and data. To showcase this procedure, we will look at a few examples from a basic program which can create and add values to a collection. First we will consider some positive tests to validate functionality. This test asserts that adding five values to the collection correctly increases the size of the collection by five. The next test verifies that the capacity of the container is always large enough to house the size of the collection. Moving on to some negative unit tests, the next block of code asserts that the program handles a request for a value which does not exist by throwing an exception. Similarly, the next example ensures that an exception is thrown when an invalid parameter is passed to the function. When designing these tests, it is important to implement both positive and negative tests that focus specifically on a single function of the program. There are several frameworks which exist to accomplish this goal such as Google Test and JUnit Test. |
| **9** | In the DevSecOps pipeline, security is a focus at all stages. To maintain efficiency, many of the methods to validate functionality can be automated. These automations vary from static tests to compiler errors to dynamic testing, but regardless of the tool, these automations will be mainly utilized after some amount of prototype code has been created. |
| **10** | With the knowledge that DevSecOps is a continuous cycle of development, proper development of testing and monitoring tools is a must. System must be agile and ready to respond to new threats or information, and the development team must be ready to switch focus at a moment’s notice. Once again, automation can greatly reduce the required resources for such a robust monitoring program. Some popular tools to identify vulnerabilities, breaches, and security issues are listed here. They can follow predefined rules and integrate with a variety of platforms which allows for smooth implementation into the existing systems at Green Pace. |
| **11** | So why should security be integrated into the existing DevOps framework? Let’s consider the following pros and cons of changing now versus waiting. Implementing these policies now will lead to a more secure product and an increase in consumer trust. It will reduce the need for reactive maintenance in the face of security issues and as a result, reduce the long-term costs associated with business. Finally, Green Pace has a chance to be at the forefront of the market as more companies begin to shift to this revised framework. However, change does not come without cost. Adapting to a new methodology will require added time for training as well as the additional cost of modifying the current systems. Both creates a more logistically complex situation in the short term. Now if Green Pace is to wait to implement these changes, short term cost will be cheaper, and products can be pushed to the market quicker. But waiting also creates an increased risk for security breaches which would cause both legal and financial liabilities. These security risks would breed consumer distrust and put Green Pace at a disadvantage in the market. Employees can also become dissatisfied it they are pushing unsecure products to the market. |
| **12** | Change is not easy and secure development requires a constant effort. Even with these policies security is never a 100% guarantee, but a failure to act now will only lead to negative long-term repercussions for the business. Secure Development reduces risk which leads to user satisfaction and business growth. |
| **13** | With considerations to the current operating model, these are the immediate steps necessary to achieve a prosperous business through secure development. Green Pace must foster a culture around security throughout all levels of the organization. To aid with this, creating a new position dedicated to reviewing security policies and enforcement should be created. At the development level, periodic meetings discussing the latest threats and steps to mitigate the risk should be held. This combined with a robust and unified coding standard for the team to follow, a policy of least privilege, and skepticism for any external data will provide the initial steps to a successful implementation of the DevSecOps framework. |
| **14** | To recap, Green Pace must build upon its current implementation of DevOps and add a renewed focus on Data Security. A culture shift is necessary for the business to grow consumer trust and satisfaction. The following tips can help the team as it undergoes this transition. Security must begin from the start, resources should be dedicated solely towards data security, a DevSecOps manifesto should be drafted, and a healthy paranoia regarding system security are necessary. Change starts small, but constant focus and practice will build Green Pace into an institution based on trust. There is one final example I will leave you with. In 2018, Cisco began a transition to a DevSecOps framework and when recapping their journey to implement this change; their Senior Director of Security Engineering noted that 72% if accounts serviced by their cloud hosting adopted their security tools within the first 9 months. Of these accounts, 97% of them now report a security health score of A or B correlating to a healthy level of security. Not only were their accounts eager to adopt new security measures, but these measures were greatly successful in mitigating the risks to these accounts. Trust is the basis for any business and to quote Ruba Borno, Former Senior Vice President at Cisco CX, “It starts with security as the foundation.” Thank you for you time. |