**Corey Hamilton**

**July 31, 2024**

**Project Two: Security Policy Presentation**

Video link https://www.youtube.com/watch?v=Q8EHtUC5lCk

**Title Slide: Green Pace Secure Development Policy**

Welcome, everyone. My name is Corey Hamilton, and today I'll be presenting our new security policy for Green Pace.

**Slide 2: Defense in Depth**

Our security policy is crucial for ensuring that all developed applications at Green Pace follow secure principles consistently. This policy is part of an overarching defense-in-depth strategy, ensuring multiple layers of security measures to protect our applications and systems.

**Slide 3: Threats Matrix**

We have identified several key coding standards to address potential security vulnerabilities.

**Likely Risks:**

**Data Type Consistency (STD-001-DAT):** Inconsistent data types can lead to unexpected behavior and vulnerabilities.

**Prevent SQL Injection (STD-004-SQL):** SQL injection allows attackers to manipulate database queries.

**Priority Risks:**

**String Handling (STD-003-STR):** Incorrect string handling can lead to buffer overflows.

**Safe Memory Management (STD-005-MEM):** Ensuring safe memory management to prevent remote code execution.

**Low Priority Risks:**

**Data Type Consistency (STD-001-DAT):** In non-critical logging functions, inconsistencies are less severe.

**Prevent SQL Injection (STD-004-SQL):** In internal reporting systems, the impact of SQL injection is less critical.

**Unlikely Risks:**

**Proper Exception Handling (STD-007-EXC):** In non-critical utility applications, poor handling has minimal impact.

**Ensure Proper Input Validation (STD-008-INV):** In internal tools, the risk is minimal due to trusted sources.

**Slide 4: 10 Principles**

We have established 10 principles, each supported by specific coding standards:

**Validate Input Data:**

**Standards:** Valid Data Values (STD-002-DAT), Ensure Proper Input Validation (STD-008-INV)

**Heed Compiler Warnings:**

**Standards:** Data Type Consistency (STD-001-DAT), String Handling (STD-003-STR)

Architect and Design for Security Policies**:**

**Standards:** Safe Memory Management (STD-005-MEM), Prevent SQL Injection (STD-004-SQL)

**Keep It Simple:**

**Standards:** Data Type Consistency (STD-001-DAT), Efficient Resource Management (STD-009-RSM)

**Default Deny:**

**Standards:** Secure Logging Practices (STD-010-LOG), Ensure Proper Input Validation (STD-008-INV)

**Adhere to the Principle of Least Privilege:**

**Standards:** Prevent SQL Injection (STD-004-SQL), Efficient Resource Management (STD-009-RSM)

**Sanitize Data Sent to Other Systems:**

**Standards:** Ensure Proper Input Validation (STD-008-INV), String Handling (STD-003-STR)

**Practice Defense in Depth:**

**Standards:** Safe Memory Management (STD-005-MEM), Prevent SQL Injection (STD-004-SQL)

**Use Effective Quality Assurance Techniques:**

**Standards:** Effective Use of Assertions (STD-006-ASS), Proper Exception Handling (STD-007-EXC)

**Adopt a Secure Coding Standard:**

**Standards:** All previously mentioned coding standards.

**Slide 5: Coding Standards**

Our 10 coding standards are prioritized as follows:

**. Data Type Consistency (STD-001-DAT)**

•**Explanation:** Ensuring data type consistency prevents errors and vulnerabilities arising from improper type usage. Consistent data types maintain predictable behavior and prevent type-related security issues such as buffer overflows.

**2. Valid Data Values (STD-002-DAT)**

•**Explanation:** Ensuring data values are valid and within expected ranges prevents incorrect behavior and security vulnerabilities. This standard ensures data integrity and application reliability.

**3. String Handling (STD-003-STR)**

•**Explanation:** Proper string handling prevents common security vulnerabilities such as buffer overflows and injection attacks. Ensuring correct string manipulation is essential for maintaining security and application stability.

**4. Prevent SQL Injection (STD-004-SQL)**

•**Explanation:** SQL injection is a common and dangerous vulnerability that can lead to unauthorized access to database information. This standard ensures that queries are constructed safely to prevent injection attacks.

**5. Safe Memory Management (STD-005-MEM)**

•**Explanation:** Proper memory management is crucial to prevent vulnerabilities such as buffer overflows, memory leaks, and use-after-free errors which can lead to security breaches and application crashes.

**6. Effective Use of Assertions (STD-006-ASS)**

•**Explanation:** Assertions help in identifying logical errors during development. They are a useful debugging tool that ensures assumptions made in the code hold true, preventing incorrect behavior.

**7. Proper Exception Handling (STD-007-EXC)**

•**Explanation:** Handling exceptions correctly prevents unexpected crashes and ensures the application can gracefully recover from errors. This standard ensures robust and maintainable code.

**8. Ensure Proper Input Validation (STD-008-INV)**

•**Explanation:** Validating input is crucial to prevent malicious data from entering the system which can cause various security issues including injection attacks, data corruption, and application crashes.

**9. Efficient Resource Management (STD-009-RSM)**

•**Explanation:** Proper management of resources such as memory, files, and network connections is essential to prevent resource leaks, exhaustion, and security issues.

**10. Secure Logging Practices (STD-010-LOG)**

•**Explanation:** Logging is essential for monitoring and debugging, but logs must be handled securely to prevent leakage of sensitive information and ensure integrity.

•

•

We prioritize these based on severity, likelihood, remediation cost, and overall impact on security.

**Slide 6: Encryption Policies**

**Encryption in Flight:**

**Policy:** All data transmitted over networks must be encrypted using secure protocols such as TLS 1.2 or higher.

**Encryption at Rest:**

**Policy:** All sensitive information must be encrypted when stored using robust methods like AES-256.

**Encryption in Use:**

**Policy:** Data being processed must be handled securely to prevent exposure.

**Slide 7: Triple-A Policies**

**Authentication:**

**Policy:** Verify the identity of all users and devices using multi-factor authentication (MFA).

**Authorization:**

**Policy:** Grant access based on roles and permissions using a role-based access control system.

**Accounting:**

**Policy:** Keep detailed logs of all activities, including user actions, system changes, and data access.

**Slides 8-11: Unit Testing**

**Coding Vulnerability Identification**

**Title: SQL Injection Vulnerability**

**Explanation:** SQL Injection vulnerabilities allow attackers to manipulate database queries, leading to unauthorized data access and manipulation. This test will ensure that our application correctly handles input validation to prevent SQL injection.

### **Test 1 - Positive Result**

**Title: Checking Safe Login with Regular User Input**

**What We Tested:**

We checked if the application correctly handles a normal login attempt using a username and password that do not contain any harmful SQL commands.

**Outcome:**

The application passed this test. It processed the login without any issues and didn't let any SQL injection attempts through. This shows that our application correctly handles regular user input safely.

### **Test 2: Positive Result**

**Title: Checking Resistance to SQL Injection Attempts**

**Here's What We Tested:**

We tested the system’s response to a login attempt where someone tries to exploit a security loophole. Instead of a regular password, the input used was a cleverly crafted statement intended to bypass the normal security checks. This statement is a common trick used by attackers, which basically tells the system to ignore the actual password and grant access if any part of the statement is true, which it always will be due to the way it's written.

**Outcome:**

Our system successfully recognized and blocked this attempt to trick it. By detecting and stopping such manipulation, our application effectively prevents unauthorized access that could otherwise bypass our usual login process. This safeguards our application from such attacks, ensuring that access is granted only to users who legitimately authenticate, thereby preserving the security and integrity of our systems.

### **Test 3: Negative Result**

**Title: Assessing Vulnerability to Direct SQL Query Manipulations**

**Here's What We Tested:**

We simulated a scenario where user input was directly used in constructing SQL queries without any form of sanitization or parameterization. This kind of approach can expose the system to SQL injection vulnerabilities.

**Outcome:**

Unfortunately, the application failed this test by allowing unauthorized access through a manipulated SQL query. This result highlights a significant security flaw where direct inclusion of user input in SQL commands compromises the system's security.

### **Test 4: Negative Result**

**Title: Testing for SQL Command Injection in User Input**

**Here's What We Tested:**

This test involved entering a user input that included malicious SQL commands designed to disrupt database operations, such as "' OR 'DROP TABLE users;". This tests the application's ability to segregate and neutralize potentially harmful SQL commands embedded within user inputs.

**Outcome:**

The application failed to prevent the execution of the malicious SQL command embedded in the user input, leading to potentially severe consequences like data loss. This demonstrates a critical vulnerability where the application does not adequately differentiate between data and SQL commands, posing a major risk to data integrity.

**Slide 12: Automation Summary**

**DevSecOps Diagram:**

**Explanation:**

**Planning:**

**Security Tools:** None

**Explanation:** Security requirements and policies are defined.

**Coding:**

**Security Tools:** Static Code Analysis Tools (e.g., SonarQube)

**Explanation:** Analyze the source code for security vulnerabilities.

**Building:**

**Security Tools:** Dependency Scanners (e.g., OWASP Dependency-Check)

**Explanation:** Check for vulnerabilities in third-party libraries.

**Testing:**

**Security Tools:** DAST, SAST, IAST tools

**Explanation:** Automated security tests for vulnerabilities.

**Releasing:**

**Security Tools:** Configuration Management Tools (e.g., Ansible, Chef)

**Explanation:** Ensure deployment configurations adhere to security policies.

**Deploying:**

**Security Tools:** Container Security Tools (e.g., Docker Security Scanning, Kubernetes Security)

**Explanation:** Scan container images for vulnerabilities.

**Operating:**

**Security Tools:** SIEM Systems, IDS

**Explanation:** Monitor application performance and detect security incidents.

**Monitoring:**

**Security Tools:** Continuous Monitoring Tools (e.g., Nagios, Prometheus)

**Explanation:** Continuously monitor for security vulnerabilities.

### **Slide: Risks and Benefits**

**Title: Risks and Benefits of Implementing Security Policies**

### **Slide 1: Problems and Solutions**

**Problems:**

**Security Vulnerabilities:**

**Description:** Existing applications may have unaddressed security vulnerabilities, leading to potential breaches.

**Impact:** Data theft, unauthorized access, and system compromise.

**Lack of Consistency:**

**Description:** Inconsistent security practices across different teams and projects.

**Impact:** Increased risk of security incidents and difficulty in managing security.

**Solutions:**

**Implementing Security Policies:**

**Description:** Establish and enforce comprehensive security policies.

**Benefit:** Standardizes security practices across the organization.

**Automated Security Tools:**

**Description:** Integrate automated security tools into the DevSecOps pipeline.

**Benefit:** Early detection and remediation of security vulnerabilities.

### **Slide 2: Risks and Benefits of Acting Now**

**Risks of Acting Now:**

**Resource Allocation:**

**Description:** Implementing new security policies and tools requires resources (time, personnel, budget).

**Impact:** Temporary reallocation of resources from other projects.

**Learning Curve:**

**Description:** Teams may need time to adapt to new security practices and tools.

**Impact:** Initial slowdown in productivity.

**Benefits of Acting Now:**

**Enhanced Security:**

**Description:** Immediate improvement in the security posture of applications.

**Impact:** Reduced risk of data breaches and security incidents.

**Regulatory Compliance:**

**Description:** Aligning with industry standards and regulations.

**Impact:** Avoiding potential fines and legal issues.

**Customer Trust:**

**Description:** Demonstrating a commitment to security.

**Impact:** Increased customer confidence and potential for new business opportunities.

### **Slide 3: Risks and Benefits of Waiting**

**Risks of Waiting:**

**Increased Vulnerability:**

**Description:** Continued exposure to existing security vulnerabilities.

**Impact:** Higher likelihood of data breaches and security incidents.

**Regulatory Non-Compliance:**

**Description:** Failure to meet industry security standards and regulations.

**Impact:** Potential for fines, legal issues, and reputational damage.

**Benefits of Waiting:**

**Resource Availability:**

**Description:** More time to allocate resources and plan the implementation.

**Impact:** Better preparedness for a comprehensive rollout.

**Thorough Planning:**

**Description:** More time for thorough planning and testing.

**Impact:** Potential for a smoother implementation process.

### **Summary**

**Acting Now:** Provides immediate security benefits, regulatory compliance, and enhanced customer trust but requires resource allocation and adaptation.

**Waiting:** Allows for better resource planning and thorough preparation but increases the risk of security incidents and non-compliance issues.

### **Identify Gaps in the Security Policy**

**Title: Identifying Gaps in the Security Policy**

### **Current Gaps in the Security Policy**

**Emerging Threats and New Vulnerabilities**

**Description:** The current policy may not cover newly discovered security threats and vulnerabilities.

**Impact:** Increased risk of attacks exploiting these new vulnerabilities.

**Solution:** Regularly update the security policy to include measures against emerging threats and ensure continuous monitoring for new vulnerabilities.

**Lack of Comprehensive Employee Training**

**Description:** Insufficient training programs for employees on security best practices and policy compliance.

**Impact:** Increased likelihood of human error leading to security breaches.

**Solution:** Implement regular, mandatory security training sessions for all employees and conduct periodic assessments to ensure understanding and adherence.

**Insufficient Coverage of Third-Party Integrations**

**Description:** The policy does not adequately address the security implications of integrating third-party services and APIs.

**Impact:** Potential vulnerabilities introduced through third-party components.

**Solution:** Develop and enforce strict security guidelines for evaluating and integrating third-party services, including regular security assessments and audits.

**Limited Scope of Encryption Policies**

**Description:** The current encryption policies may not cover all sensitive data types and communication channels.

**Impact:** Risk of data exposure in unprotected areas.

**Solution:** Expand the encryption policies to ensure all sensitive data, including backups and internal communications, are adequately protected.

**Inadequate Incident Response Plan**

**Description:** The existing incident response plan may not be comprehensive or regularly tested.

**Impact:** Delayed or ineffective response to security incidents, leading to greater damage.

**Solution:** Develop a detailed incident response plan, conduct regular drills, and continuously improve the plan based on the outcomes of these exercises.

**Gaps in Continuous Monitoring and Auditing**

**Description:** The policy might lack comprehensive guidelines for continuous security monitoring and auditing.

**Impact:** Delayed detection of security breaches and non-compliance issues.

**Solution:** Implement continuous monitoring tools and practices, and schedule regular security audits to ensure compliance and detect anomalies promptly.

**Limited Focus on Application Security**

**Description:** The policy may primarily focus on infrastructure security, neglecting application-level security.

**Impact:** Increased risk of application-specific vulnerabilities.

**Solution:** Integrate secure coding practices, application security testing, and regular vulnerability assessments into the policy.

**Insufficient User Access Management**

**Description:** Inadequate policies for managing user access and permissions.

**Impact:** Unauthorized access to sensitive information.

**Solution:** Implement strict access control measures, regular reviews of user permissions, and enforce the principle of least privilege.

### **Recommendations for Improvement**

**Regular Updates and Reviews:** Schedule regular reviews and updates of the security policy to address emerging threats and ensure ongoing relevance.

**Enhanced Training Programs:** Develop comprehensive training programs tailored to different roles within the organization.

**Strengthened Third-Party Policies:** Enforce stricter guidelines and assessments for third-party integrations.

**Comprehensive Encryption Coverage:** Ensure all sensitive data and communication channels are encrypted.

**Robust Incident Response:** Create a detailed and tested incident response plan.

**Continuous Monitoring and Auditing:** Implement continuous monitoring and regular auditing practices.

**Application Security Focus:** Emphasize application security in the policy.

**User Access Management:** Enforce strict user access controls and regular permission reviews.

### **Standards to Adopt for Future Security**

**Title: Standards to Adopt for Future Security**

### **1. NIST Cybersecurity Framework (CSF)**

**Description:** Provides a policy framework of computer security guidance for how private sector organizations can assess and improve their ability to prevent, detect, and respond to cyber attacks.

**Benefits:** Offers a comprehensive approach to managing cybersecurity risks, including best practices for identifying, protecting, detecting, responding, and recovering from cybersecurity events.

**Implementation:** Adopt NIST CSF to develop a robust cybersecurity program and conduct regular assessments to ensure compliance.

### **2. ISO/IEC 27001:2013**

**Description:** Specifies the requirements for establishing, implementing, maintaining, and continually improving an information security management system (ISMS).

**Benefits:** **Helps organizations keep information assets secure** by implementing a systematic approach to managing sensitive company information.

**Implementation:** Achieve ISO/IEC 27001:2013 certification to demonstrate commitment to information security management and regularly review and update policies to maintain compliance.

### **3. OWASP (Open Web Application Security Project) Top Ten**

**Description:** A standard awareness document for developers and web application security, representing a broad consensus about the most critical security risks to web applications.

**Benefits:** Helps organizations understand and improve the security of their web applications by addressing the most critical security risks.

**Implementation:** Integrate OWASP Top Ten into the development process to ensure web applications are built with security best practices from the outset.

### **4. CIS Controls (Center for Internet Security)**

**Description:** A set of best practices for securing IT systems and data against cyber attacks.

**Benefits:** Provides prioritized and focused sets of actions that organizations can implement to protect themselves from cyber threats.

**Implementation:** Adopt CIS Controls to establish a baseline of security measures and regularly update the controls to adapt to new threats.

### **5. GDPR (General Data Protection Regulation)**

**Description:** A regulation in EU law on data protection and privacy in the European Union and the European Economic Area.

**Benefits:** Ensures that organizations protect personal data and privacy of individuals, providing rights to individuals regarding their personal data.

**Implementation:** Comply with GDPR requirements by implementing strong data protection measures, conducting data protection impact assessments, and ensuring transparency in data processing activities.

### **Summary**

**Adopting these standards will help prevent future security problems by providing comprehensive frameworks and best practices for managing cybersecurity risks.**

**Regular assessments and updates will ensure ongoing compliance and adaptation to emerging threats.**

### **References**

**Title: References**

National Institute of Standards and Technology. (2018). Framework for Improving Critical Infrastructure Cybersecurity, Version 1.1. Retrieved from <https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf>

International Organization for Standardization. (2013). ISO/IEC 27001:2013 Information technology — Security techniques — Information security management systems — Requirements. Retrieved from <https://www.iso.org/standard/54534.html>

Open Web Application Security Project. (2021). OWASP Top Ten 2021. Retrieved from [https://owasp.org/Top10](https://owasp.org/Top10/)/

Center for Internet Security. (2020). CIS Controls Version 8. Retrieved from [https://www.cisecurity.org/controls/v8](https://www.cisecurity.org/controls/v8/)/

European Union. (2018). General Data Protection Regulation (GDPR) – Regulation (EU) 2016/679. Retrieved from [https://gdpr.eu](https://gdpr.eu/)/