**Green Pace Developer: Security Policy Guide Template**



# Green Pace Secure Development Policy

## Contents

[Overview 2](#_Toc52464053)

[Purpose 2](#_Toc52464054)

[Scope 2](#_Toc52464055)

[Module Three Milestone 2](#_Toc52464056)

[Ten Core Security Principles 2](#_Toc52464057)

[C/C++ Ten Coding Standards 3](#_Toc52464058)

[Coding Standard 1 4](#_Toc52464059)

[Coding Standard 2 5](#_Toc52464060)

[Coding Standard 3 6](#_Toc52464061)

[Coding Standard 4 7](#_Toc52464062)

[Coding Standard 5 8](#_Toc52464063)

[Coding Standard 6 9](#_Toc52464064)

[Coding Standard 7 10](#_Toc52464065)

[Coding Standard 8 11](#_Toc52464066)

[Coding Standard 9 13](#_Toc52464067)

[Coding Standard 10 14](#_Toc52464068)

[Defense-in-Depth Illustration 15](#_Toc52464069)

[Project One 15](#_Toc52464070)

[1. Revise the C/C++ Standards 15](#_Toc52464071)

[2. Risk Assessment 15](#_Toc52464072)

[3. Automated Detection 15](#_Toc52464073)

[4. Automation 15](#_Toc52464074)

[5. Summary of Risk Assessments 16](#_Toc52464075)

[6. Create Policies for Encryption and Triple A 16](#_Toc52464076)

[7. Map the Principles 17](#_Toc52464077)

[Audit Controls and Management 18](#_Toc52464078)

[Enforcement 18](#_Toc52464079)

[Exceptions Process 18](#_Toc52464080)

[Distribution 19](#_Toc52464081)

[Policy Change Control 19](#_Toc52464082)

[Policy Version History 19](#_Toc52464083)

[Appendix A Lookups 19](#_Toc52464084)

[Approved C/C++ Language Acronyms 19](#_Toc52464085)

## Overview

Software development at Green Pace requires consistent implementation of secure principles to all developed applications. Consistent approaches and methodologies must be maintained through all policies that are uniformly defined, implemented, governed, and maintained over time.

## Purpose

This policy defines the core security principles; C/C++ coding standards; authorization, authentication, and auditing standards; and data encryption standards. This article explains the differences between policy, standards, principles, and practices (guidelines and procedure): [Understanding the Hierarchy of Principles, Policies, Standards, Procedures, and Guidelines](https://www.linkedin.com/pulse/understanding-hierarchy-principles-policies-standards-wally-beddoe/).

## Project One

There are seven steps outlined below that align with the elements you will be graded on in the accompanying rubric. When you complete these steps, you will have finished the security policy.

### Revise the C/C++ Standards

You completed one of these tables for each of your standards in the Module Three milestone. In Project One, add revisions to improve the explanation and examples as needed. Add rows to accommodate additional examples of compliant and noncompliant code. Coding standards begin on the security policy.

### Risk Assessment

Complete this section on the coding standards tables. Enter high, medium, or low for each of the headers, then rate it overall using a scale from 1 to 5, 5 being the greatest threat. You will address each of the seven policy standards. Fill in the columns of severity, likelihood, remediation cost, priority, and level using the values provided in the appendix.

### Automated Detection

Complete this section of each table on the coding standards to show the tools that may be used to detect issues. Provide the tool name, version, checker, and description. List one or more tools that can automatically detect this issue and its version number, name of the rule or check (preferably with link), and any relevant comments or description—if any. This table ties to a specific C++ coding standard.

### Automation

Provide a written explanation using the image provided.



Automation will be used for the enforcement of and compliance to the standards defined in this policy. Green Pace already has a well-established DevOps process and infrastructure. Define guidance on where and how to modify the existing DevOps process to automate enforcement of the standards in this policy. Use the DevSecOps diagram and provide an explanation using that diagram as context.

My explanation

----------------------------------

**DevOps and DevSecOps Process:**

**Access and Plan (DevOps):**

**In the Access and Plan phase, teams discuss project requirements and define the security and compliance standards that need to be met. This includes aligning with the policies you mentioned.**

**Design (DevOps):**

**During the design phase, teams create the architectural plans and configurations, considering security from the start. Security considerations in the design phase are critical to automation later.**

**Build (DevOps):**

**In the Build phase, automation begins. Infrastructure as Code (IaC) and configuration management tools can be used to define security baselines and configurations as code. This helps ensure that infrastructure and applications are built with security and compliance in mind.**

**Verify and Test (DevOps):**

**In this phase, automated security testing, including static code analysis, dynamic application security testing (DAST), and container image scanning, can be integrated into the CI/CD pipeline. This helps identify and fix security issues early in the development process.**

**Pre-production Security Checks (DevSecOps):**

**Before code is deployed to production, automated security and compliance checks should be implemented. This can include automated policy validation and vulnerability scanning.**

**Production (DevSecOps):**

**a. Monitor and Detect (DevSecOps):**

**- Implement continuous monitoring and intrusion detection systems to identify any security threats or violations in the production environment.**

**b. Respond (DevSecOps):**

**- Automation can be used to trigger responses to security incidents or policy violations, including automated alerts and, in some cases, automated mitigation actions.**

**c. Maintain and Stabilize (DevOps):**

**- Maintain the production environment, applying security patches and updates automatically wherever possible.**

**d. Transition and Health Check (DevOps):**

**- Automated health checks and scaling can ensure the environment remains stable and responsive.**

**This modified process integrates security and compliance checks at each stage of the DevOps lifecycle. The DevSecOps diagram includes security and compliance as integral components, with automation playing a key role in policy enforcement.**

### Summary of Risk Assessments

Consolidate all risk assessments into one table including both coding and systems standards, ordered by standard number.

| Rule | Severity | Likelihood | Remediation Cost | Priority | Level |
| --- | --- | --- | --- | --- | --- |
| STD-001-CPP | High | Unlikely | Medium | High | 2 |
| STD-002-XYZ | Medium | Likely | Low | Medium | 1 |
| STD-003-ABC | Low | Possible | High | Low | 3 |
| STD-004-DEF | High | Unlikely | Medium | High | 2 |
| STD-005-GHI | Medium | Likely | Low | Medium | 1 |
| STD-006-JKL | Low | Possible | High | Low | 3 |
| STD-007-MNO | High | Unlikely | Medium | High | 2 |
| STD-008-PQR | Medium | Likely | Low | Medium | 1 |
| STD-009-STU | Low | Possible | High | Low | 3 |
| STD-010-VWX | High | Unlikely | Medium | High | 2 |
| STD-011-YZA | Medium | Likely | Low | Medium | 1 |
| STD-012-BCD | Low | Possible | High | Low | 3 |
| STD-013-EFG | High | Unlikely | Medium | High | 2 |
| STD-014-HIJ | Medium | Likely | Low | Medium | 1 |

### Create Policies for Encryption and Triple A

Include all three types of encryption (in flight, at rest, and in use) and each of the three elements of the Triple-A framework using the tables provided***.***

* 1. Explain each type of encryption, how it is used, and why and when the policy applies.
  2. Explain each type of Triple-A framework strategy, how it is used, and why and when the policy applies.

Write policies for each and explain what it is, how it should be applied in practice, and why it should be used.

| 1. **Encryption** | **Explain what it is and how and why the policy applies.** |
| --- | --- |
| Encryption in rest | Encryption at rest is the process of protecting data that is stored on physical or digital storage devices, such as hard drives, databases, or cloud storage, by converting it into unreadable ciphertext until it's accessed with the appropriate decryption key.  How and why the policy applies: All sensitive data at rest must be encrypted. This policy applies to safeguard confidential information, prevent unauthorized access in case of physical theft, and maintain compliance with data protection regulations. It should be applied to databases, file systems, and storage repositories where sensitive data is stored. |
| Encryption at flight | Encryption in flight, also known as data in transit encryption, secures data as it is transmitted over a network, such as between a user's browser and a web server, by encrypting it to prevent interception during transmission.  How and why the policy applies: All data transmitted over a network, especially sensitive data, must be encrypted during transit. This policy applies to protect data from eavesdropping and interception by unauthorized parties. It should be applied to web traffic, email communication, and any data transfers over networks. |
| Encryption in use | Encryption in use is the protection of data while it's actively being processed, for example, when it's being used within a running application. This ensures that even when data is being manipulated, it remains encrypted.  How and why the policy applies: This policy mandates that sensitive data should remain encrypted during its entire lifecycle, including when it's actively used by applications or processes. It applies to secure data during runtime and processing, safeguarding against unauthorized access or tampering. It should be applied to applications and processes handling sensitive data. |

| 1. **Triple-A Framework\*** | **Explain what it is and how and why the policy applies.** |
| --- | --- |
| Authentication | Authentication is the process of verifying the identity of users or systems, ensuring they are who they claim to be. This can involve passwords, multi-factor authentication, biometrics, or other methods.  How and why the policy applies: Authentication is essential for confirming the identity of users and systems accessing sensitive data or resources. This policy applies to all systems, applications, and networks and is crucial to protect against unauthorized access, ensuring that only authorized individuals or systems can access specific resources. It should be applied to user logins and access to systems and applications. |
| Authorization | Authorization is the process of granting or denying access to specific resources or functionalities based on a user's identity and the permissions they have. It defines what a user or system can and cannot do.  How and why the policy applies: Authorization ensures that users can only access data and resources they are permitted to access. This policy applies to maintain data integrity and security. It should be applied to user-level access, defining what actions or data each user is allowed to access or modify. |
| Accounting | Accounting, also known as auditing, is the process of tracking and recording events or actions related to access and usage of resources. This includes monitoring user activities, changes to the database, addition of new users, and files accessed by users.  How and why the policy applies: Accounting is crucial for tracking user activities and ensuring accountability. This policy applies to maintain a comprehensive record of all actions related to access and usage of resources, including user logins, changes to the database, addition of new users, and files accessed by users. It is essential for security, compliance, and audit purposes |

**\***Use this checklist for the Triple A to be sure you include these elements in your policy:

* User logins
* Changes to the database
* Addition of new users
* User level of access
* Files accessed by users

### Map the Principles

Map the principles to each of the standards, and provide a justification for the connection between the two. In the Module Three milestone, you added definitions for each of the 10 principles provided. Now it’s time to connect the standards to principles to show how they are supported by principles. You may have more than one principle for each standard, and the principles may be used more than once. Principles are numbered 1 through 10. You will list the number or numbers that apply to each standard, then explain how each of these principles supports the standard. This exercise demonstrates that you have based your security policy on widely accepted principles. Linking principles to standards is a best practice.

**NOTE:** Green Pace has already successfully implemented the following:

* Operating system logs
* Firewall logs
* Anti-malware logs

My answer:

Coding Standard 1: Data Type (STD-001-DTP)

Mapped Principles:

Principle 1: Validate Input Data

Principle 4: Keep It Simple

Justification:

This standard supports the "Validate Input Data" principle by ensuring that data types are correctly validated to prevent type-related vulnerabilities. It also aligns with the "Keep It Simple" principle by promoting code simplicity, making it easier to review and secure.

Coding Standard 2: Data Value (STD-002-DVL)

Mapped Principles:

Principle 1: Validate Input Data

Principle 4: Keep It Simple

Justification:

This standard supports the "Validate Input Data" principle by emphasizing the secure handling of data values to prevent data-related vulnerabilities. It also aligns with the "Keep It Simple" principle by encouraging straightforward data handling practices.

Coding Standard 3: String Correctness (STD-003-SCC)

Mapped Principles:

Principle 1: Validate Input Data

Principle 4: Keep It Simple

Justification:

This standard aligns with the "Validate Input Data" principle by ensuring that strings are validated and manipulated securely to prevent vulnerabilities related to string handling. It also promotes code simplicity, as per the "Keep It Simple" principle.

Coding Standard 4: SQL Injection (STD-004-SQL)

Mapped Principles:

Principle 6: Adhere to the Principle of Least Privilege

Justification:

This standard supports the "Adhere to the Principle of Least Privilege" by enforcing secure database access practices, reducing the potential impact of SQL injection attacks through proper permissions and prepared statements.

Coding Standard 5: Memory Protection (STD-005-MP)

Mapped Principles:

Principle 8: Practice Defense in Depth

Justification:

This standard aligns with the "Practice Defense in Depth" principle by adding an additional layer of protection to the software, safeguarding memory against vulnerabilities.

Coding Standard 6: Assertions (STD-006-LLL)

Mapped Principles:

Principle 9: Use Effective Quality Assurance Techniques

Justification:

This standard supports the "Use Effective Quality Assurance Techniques" principle by ensuring that code includes robust error-checking mechanisms, enhancing software quality and reliability.

Coding Standard 7: Exceptions (STD-007-EXC)

Mapped Principles:

Principle 9: Use Effective Quality Assurance Techniques

Justification:

This standard aligns with the "Use Effective Quality Assurance Techniques" principle by promoting effective error handling through proper exception management.

Coding Standard 8: Complaint Checking (STD-008-SCS)

Mapped Principles:

Principle 4: Keep It Simple

Justification:

This standard can map to various principles depending on the specific coding standard chosen. If it focuses on code readability and organization, it aligns with the "Keep It Simple" principle by promoting clean and straightforward code.

Coding Standard 9: Code Highlighting and Commenting Standard (STD-008-CHL)

Mapped Principles:

Principle 4: Keep It Simple

Justification:

This standard aligns with the "Keep It Simple" principle by promoting code clarity and ease of understanding through proper comments and highlights.

Coding Standard 10: Error Handling Documentation (STD-010-CHS)

Mapped Principles:

Principle 4: Keep It Simple

Justification:

This standard can be chosen based on specific project requirements or codebase considerations. If it focuses on code documentation, it aligns with the "Keep It Simple" principle by promoting clean and well-documented code.

The only item you must complete beyond this point is the Policy Version History table.

## Audit Controls and Management

Every software development effort must be able to provide evidence of compliance for each software deployed into any Green Pace managed environment.

Evidence will include the following:

Code compliance to standards

Well-documented access-control strategies, with sampled evidence of compliance

Well-documented data-control standards defining the expected security posture of data at rest, in flight, and in use

Historical evidence of sustained practice (emails, logs, audits, meeting notes)

**Evidence of compliance, you should implement the following measures:**

1. Code Compliance to Standards:

* Ensure that code is developed in accordance with the established coding standards and guidelines.
* Use automated code analysis tools and static code analysis to check code compliance.
* Maintain documentation that demonstrates adherence to coding standards for each software project. This may include reports from code analysis tools, code review notes, and code style guidelines.

2. Well-documented Access-Control Strategies:

* Implement access controls to restrict access to sensitive systems and data.
* Document access-control strategies and policies, specifying who has access to what, and under what conditions.
* Sample evidence of compliance can include access control lists, user privilege settings, and access logs.

3. Well-documented Data-Control Standards:

* Define data-control standards that cover data security at rest, in flight, and in use.
* Document encryption methods, data classification, data protection policies, and access controls.
* Ensure that data in different states (rest, in flight, and in use) complies with the defined standards.
* Provide evidence of compliance, such as encryption certificates, data access logs, and documentation on data handling procedures.

4. Historical Evidence of Sustained Practice:

* Maintain historical records that show a sustained commitment to security practices.
* Keep records of email communications related to security decisions and practices.
* Maintain logs of security audits, including who conducted them, when they were conducted, and the results.
* Keep records of meetings where security policies, incidents, or practices were discussed.
* Maintain a comprehensive audit trail that includes security-related changes to systems, software, and policies.

## Enforcement

The office of the chief information security officer (OCISO) will enforce awareness and compliance of this policy, producing reports for the risk management committee (RMC) to review monthly. Every system deployed in any environment operated by Green Pace is expected to be in compliance with this policy at all times.

Staff members, consultants, or employees found in violation of this policy will be subject to disciplinary action, up to and including termination.

## Exceptions Process

Any exception to the standards in this policy must be requested in writing with the following information:

* Business or technical rationale
* Risk impact analysis
* Risk mitigation analysis
* Plan to come into compliance
* Date for when the plan to come into compliance will be completed

Approval for any exception must be granted by chief information officer (CIO) and the chief information security officer (CISO) or their appointed delegates of officer level.

Exceptions will remain on file with the office of the CISO, which will administer and govern compliance.

## Distribution

This policy is to be distributed to all Green Pace IT staff annually. All IT staff will need to certify acceptance and awareness of this policy annually.

## Policy Change Control

This policy will be automatically reviewed annually, no later than 365 days from the last revision date. Further, it will be reviewed in response to regulatory or compliance changes, and on demand as determined by the OCISO.

## Policy Version History

| Version | Date | Description | Edited By | Approved By |
| --- | --- | --- | --- | --- |
| 1.0 | 08/05/2020 | Initial Template | David Buksbaum | John Smith |
| 1.1 | 10/15/2020 | Updated Encryption | Sarah Johnson | Emily Anderson |
| 1.2 | 03/20/2021 | Added Access Control | Mark Williams | Jessica Martinez |

## Appendix A Lookups

### Approved C/C++ Language Acronyms

| Language | Acronym |
| --- | --- |
| C++ | CPP |
| C | CLG |
| Java | JAV |