Jacob Cannamela

CSD 430

7/25/2025

**Module 11 Assignment**

In software development, coding standards are essential to a project’s success since they help to ensure the sustained maintenance, readability, and the dependability of the code throughout the diverse development teams. These standards offer a definition to a certain set of predefined rules, which is accepted by the relevant teams, for a specific language. Those rules include, but are not limited to, documentation, naming conventions, code file and folder organization, code indentation, and other relevant factors. In a corporate environment with a big team that collaborates on a single codebase, there is a need for cohesion, which makes the use of universal standards extremely important. Each developer has distinctive preferences and project with an ensemble of numerous developers require clear rules.

Enforcing coding standards has been an issue since the dawn of the computing industry. As with any discipline, companies involved in software engineering appreciated the importance of consistency in coding practices. For any software engineering company, the reasoning is quite clear. Increased consistency makes reading, understanding, and modifying code easier in the long run. As companies grow and take on more projects, the need for consistency in coding improves overall productivity, collaboration, and reduces errors and cumulative maintenance costs. The coding standards of such industry leaders as Google, Microsoft, and Oracle speaks volumes for this. Take Google for instance, their Java Style Guides dictates a wide range of details and extensiveness such as naming rules, line and comment wrapping, and comment formatting. These standards, as rigid as they seem, were developed for a reason and are the result of decades of software engineering complexities.

To enforce coding standards, a combination of documentation, automated systems, and peer reviews are used in workflows. In my current position in the cybersecurity and threat detection field, coding standards are of particular importance when creating detection logic in Sigma format and writing queries in Splunk’s Search Processing Language (SPL). While these may not be "traditional" programming languages, elements such as consistency in documentation and formatting are critical. As an example, we observe certain conventions when writing correlation searches in Splunk, prefixing with “THREAT -” for confirmed malicious activity and “Risk Score Alert -” for uncertain behaviors. This detection naming strategy enhances readability and allows a user to understand the purpose just by looking at the title. Furthermore, our version control system Git integrates with our workflow such that pre commit hooks run checks for violation of Sigma standards (syntax and style) or other syntactical errors.

Looking through an industry lens, one of the major considerations why coding standards are so important is due to the fact that software development is hardly ever an individual endeavor. Most corporate software projects are maintained over time by large groups of software engineers, sometimes numbering in the dozens or even hundreds. It would be impossible to develop and maintain an orderly system in a scenario where every engineer is free to apply their unique coding style. Consider the scenario of trying to debug a piece of software that is at least five years old and was patched by an employee that the organization no longer has contact with. If the code is formatted poorly, uses generic names for its variables, and has no comments, the process will be far more tedious than it has to. To solve this problem, enforcing coding standards helps by providing developers with a style manual that documents rules and defines policies that programmers must adhere to.

Another benefit of having coding standards in place is enhanced quality assurance and testing processes. When code adherence is maintained, automated testing and verification of expected functionalities becomes less complex. Additionally, many IDEs feature static code analysis tools, for instance ESLint for JavaScript or SonarQube for Java. These tools flag breaches of coding standards before deployment. These tools not only enforce consistency but can also catch potential bugs or security vulnerabilities early in the development lifecycle.

Striking a balance between a robust Cyber Security strategy and imposing coding standards creates a resilient structure. Lack of focus on code consistency and structure results into loopholes. Some of the common vulnerabilities include forgotten logic checks within user verification and missed proper escaping of user submissions. Controlling and enforcing best practices is critical to ensure optimal safety when users provide data, when conditions are checked and when crucial parts of the application are documented. Cyber threat detection reveals time and again that hackers scan for unprotected installation instructions or automated setup within applications designed to configure applications. Well defined coding policies make the development process to infuse security into the standard practices at the beginning.

As we have seen, the importance of coding guidelines cannot be overstated when it comes to professional software development practices. Guidelines guarantee uniformity, enforcement of cohesion, elevating the quality of the code, and easing collaboration on intricate tasks. In addition, they provide fortified protection and improved levels of maintainability in situations that have high risk of error. Regardless of whether these standards are imposed via written documents, automated systems, or through audits, they are far from being optional. In fact, without these practices, it becomes exceedingly difficult to build software that is dependable, can be scaled, and is secured.

Here is an example of a Sigma rule we would write for my job. We must be strict with how we structure this code because our CI/CD pipleline will break if any of the structure is missing when we deploy the new alert.

title: Cleartext Protocol Usage

id: d7fb8f0e-bd5f-45c2-b467-19571c490d7e

status: stable

description: |

Ensure that all account usernames and authentication credentials are transmitted across networks using encrypted channels.

Ensure that an encryption is used for all sensitive information in transit. Ensure that an encrypted channels is used for all administrative account access.

references:

- https://www.cisecurity.org/controls/cis-controls-list/

- https://www.pcisecuritystandards.org/documents/PCI\_DSS\_v3-2-1.pdf

- https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf

author: Alexandr Yampolskyi, SOC Prime, Tim Shelton

date: 2019-03-26

modified: 2022-10-10

tags:

- attack.credential-access

# - CSC4

# - CSC4.5

# - CSC14

# - CSC14.4

# - CSC16

# - CSC16.5

# - NIST CSF 1.1 PR.AT-2

# - NIST CSF 1.1 PR.MA-2

# - NIST CSF 1.1 PR.PT-3

# - NIST CSF 1.1 PR.AC-1

# - NIST CSF 1.1 PR.AC-4

# - NIST CSF 1.1 PR.AC-5

# - NIST CSF 1.1 PR.AC-6

# - NIST CSF 1.1 PR.AC-7

# - NIST CSF 1.1 PR.DS-1

# - NIST CSF 1.1 PR.DS-2

# - ISO 27002-2013 A.9.2.1

# - ISO 27002-2013 A.9.2.2

# - ISO 27002-2013 A.9.2.3

# - ISO 27002-2013 A.9.2.4

# - ISO 27002-2013 A.9.2.5

# - ISO 27002-2013 A.9.2.6

# - ISO 27002-2013 A.9.3.1

# - ISO 27002-2013 A.9.4.1

# - ISO 27002-2013 A.9.4.2

# - ISO 27002-2013 A.9.4.3

# - ISO 27002-2013 A.9.4.4

# - ISO 27002-2013 A.8.3.1

# - ISO 27002-2013 A.9.1.1

# - ISO 27002-2013 A.10.1.1

# - PCI DSS 3.2 2.1

# - PCI DSS 3.2 8.1

# - PCI DSS 3.2 8.2

# - PCI DSS 3.2 8.3

# - PCI DSS 3.2 8.7

# - PCI DSS 3.2 8.8

# - PCI DSS 3.2 1.3

# - PCI DSS 3.2 1.4

# - PCI DSS 3.2 4.3

# - PCI DSS 3.2 7.1

# - PCI DSS 3.2 7.2

# - PCI DSS 3.2 7.3

logsource:

category: firewall

detection:

selection:

dst\_port:

- 8080

- 21

- 80

- 23

- 50000

- 1521

- 27017

- 3306

- 1433

- 11211

- 15672

- 5900

- 5901

- 5902

- 5903

- 5904

selection\_allow1:

action:

- forward

- accept

- 2

selection\_allow2:

blocked: "false" # not all fws set action value, but are set to mark as blocked or allowed or not

condition: selection and 1 of selection\_allow\*

falsepositives:

- Unknown

level: low

**References:**

Patel, R. (2024, November 24). Understanding Coding Standards: Why They matter - Ravi Patel - Medium. *Medium*. https://medium.com/@ravipatel.it/understanding-coding-standards-why-they-matter-cc75adcba245

Codewave. (2023, September 29). *The crucial role of coding standards in software development services - Codewave Insights*. Codewave Insights. https://codewave.com/insights/the-crucial-role-of-coding-standards-in-software-development-services/

Manked, R. (2024, November 13). *Importance of coding standard and code quality in software development*. Multidots. http://multidots.com/blog/importance-of-coding-standard-and-code-quality-in-software-development/