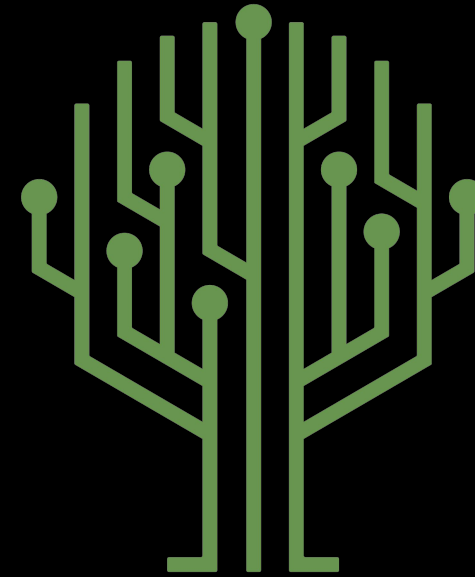


Green Pace

Security Policy Presentation

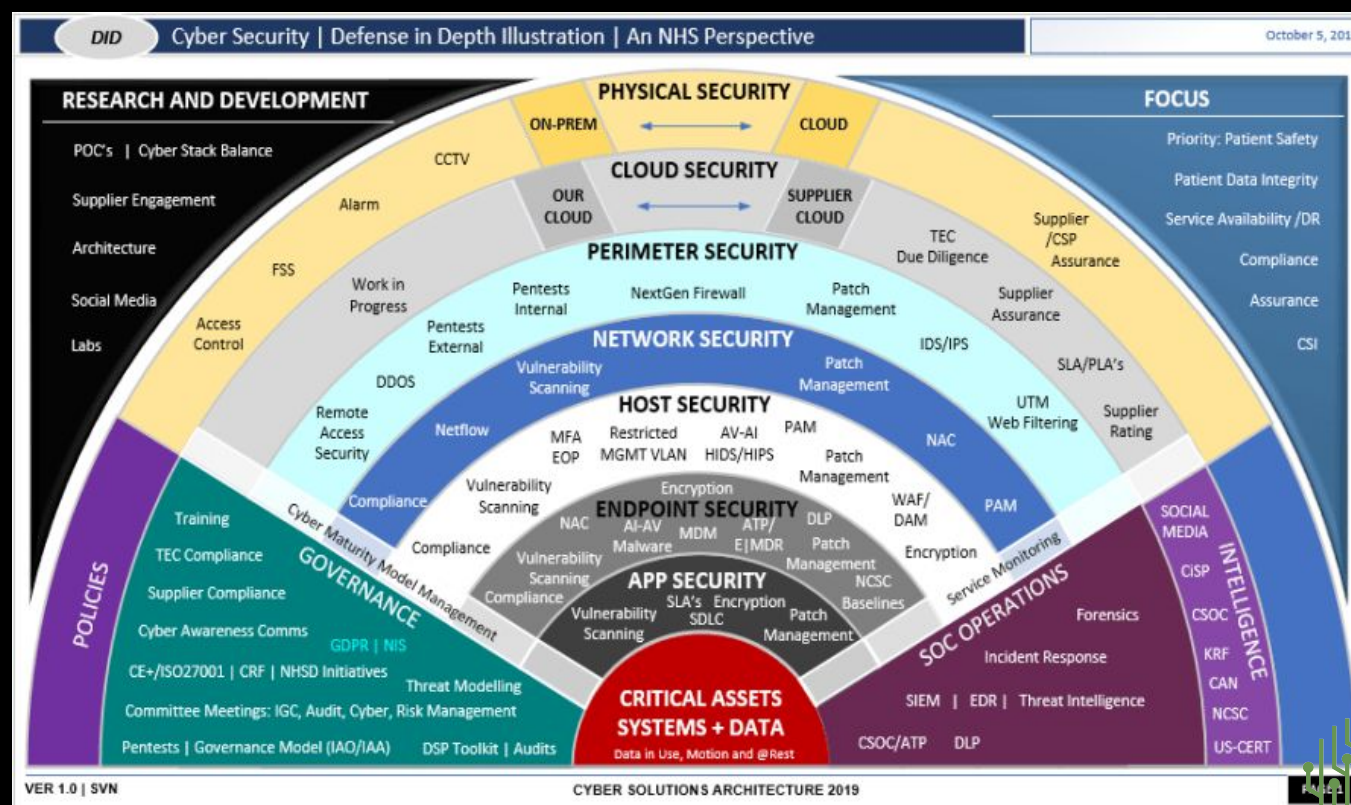
Developer: *Daniel Olson*



Green Pace

OVERVIEW: DEFENSE IN DEPTH

Security policies are required to set standards and processes for preventing unauthorized access and use of sensitive data and resources. These rules are used to promote the best practice of defense-in-depth by providing many levels of security via technological, administrative, and physical restrictions.



THREATS MATRIX

A security risk threat matrix is a tool for assessing and prioritizing security risks based on their probability of occurrence and possible impact on the company. The matrix assists in identifying high-risk areas that require immediate attention and allocating resources to alleviate such risks.

Likely Threats that are likely to happen	Priority Highly likely and almost certain to occur
Low priority Standard with low relevancy	Unlikely Relatively low chance of occurring

10 PRINCIPLES

1. Validate Input Data
2. Heed Compiler Warnings
3. Architect and Design for Security Policies
4. Keep it Simple
5. Default Deny
6. Adhere to the Principle of Least Privilege
7. Sanitize Data Sent to Other Systems
8. Practice Defence in Depth
9. Use Effective Quality Assurance Techniques
10. Adopt a Secure Coding Standard

CODING STANDARDS

1. Do not cast to an out-of range enumeration value
2. Do not declare or define a reserved identifier
3. Never qualify a reference type with const or volatile
4. Do not write syntactically ambiguous declaration
5. Overload allocation and deallocation functions as a pair in the same scope
6. Avoid information leakage when applying a class object across a trst boundary
7. Avoid cycles during initialization of static objects
8. Do not let exceptions escape from destructors or deallocation functions
9. Do not modify the standard namespaces
10. Do not define an unnamed namespace in a header file

ENCRYPTION POLICIES

Encryption in rest - The encryption of data while it is at rest or held on a device or server is referred to as encryption in rest. All data saved on devices, servers, or other storage media should be protected with a recognized encryption technique. Encryption keys should be handled and safeguarded against unwanted access using a dedicated key management system. Encryption at rest is critical for preventing unwanted access to sensitive data if the data storage medium is stolen, lost, or compromised. Encryption guarantees that even if the data is obtained, it is incomprehensible without the encryption key, adding an extra degree of protection.

Encryption at flight - The encryption of data while it is in transit between two devices or servers is referred to as encryption at flight. All data sent across a network, internal or external, should be encrypted with an authorized encryption technique. Encryption keys should be handled and safeguarded against unwanted access using a dedicated key management system. Encryption in flight is critical for safeguarding sensitive data from interception and unwanted access while it is being sent over a network. Encryption guarantees that even if the data is intercepted, it is incomprehensible without the encryption key, adding an extra degree of protection.

Encryption in use - The encryption of data while it is being utilized or processed by a device or application is referred to as encryption in use. All data that a device or application uses or processes should be encrypted with an authorized encryption technique. Encryption keys should be handled and safeguarded against unwanted access using a dedicated key management system. Encryption in use is critical for preventing unwanted access to sensitive data if the device or application is hacked or compromised. Encryption guarantees that even if the data is obtained, it is incomprehensible without the encryption key, adding an extra degree of protection.

TRIPLE-A POLICIES

Authentication - The process of authenticating the identity of a person or device seeking to access sensitive information is known as authentication. Verifying a login and password, biometric information, or a security token are all examples of this. Authentication is used to guarantee that sensitive information is only accessible to authorized users or devices. This is often accomplished through a login procedure in which users submit their credentials or use a security token. MFA (multi-factor authentication) can also be used to increase security. Any system or network that contains or processes sensitive information is subject to authentication procedures. It is vital to ensure that only authorized individuals or devices have access to sensitive information, since this can avoid data breaches and unauthorized access.

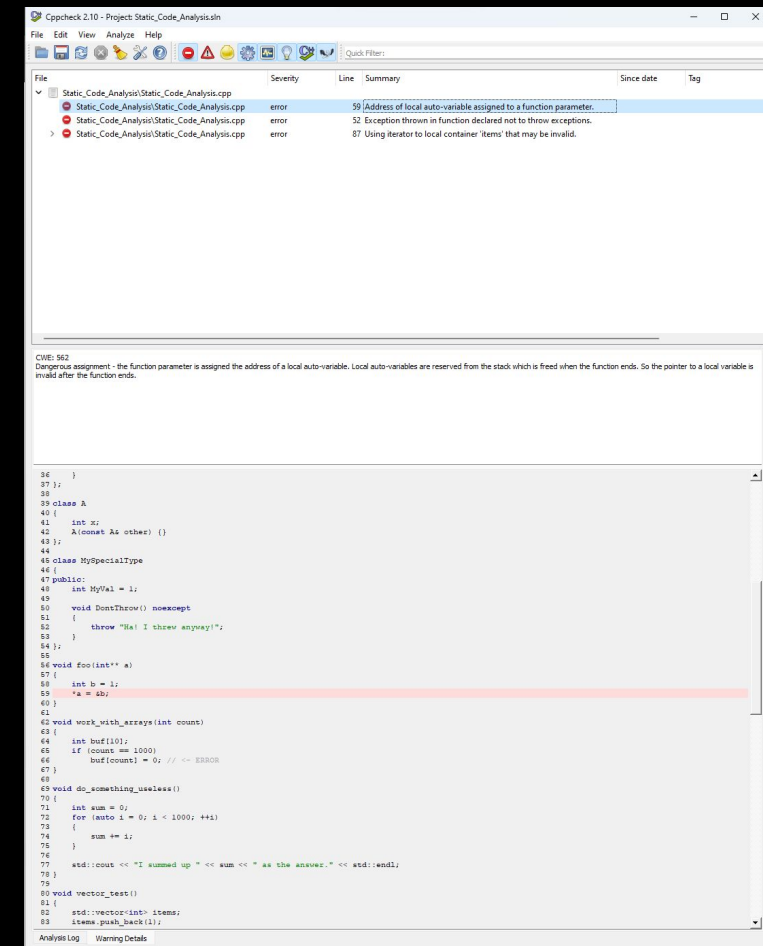
Authorization - The process of allowing or refusing access to specified resources or information depending on the user's identity, role, or other characteristics is known as authorization. Authorization ensures that users only have access to the resources or information they need to accomplish their job tasks. Access controls, such as role-based access controls (RBAC) or attribute-based access controls, are commonly used to do this. (ABAC). Any system or network that contains or processes sensitive information is subject to authorization policies. It is vital to ensure that users have just the information they need to fulfill their job tasks, since this can avoid data breaches and unauthorized access to sensitive information.

Accounting - Accounting is the process of tracking and recording all sensitive information-related actions, such as who accessed the information, what they did with it, and when. Accounting is used to ensure that all sensitive information-related actions are tracked and recorded, which can be done through audit logs or other tracking mechanisms. Accounting policies apply to any system or network that stores or processes sensitive information.



Unit Testing

In my testing for this assignment Cppcheck discovered 3 errors which included the following: Dangerous assignment (line 59), Exception thrown in function declared not to throw exceptions (line 52) and Using iterator to local container 'items' that may be invalid (line 87). Whereas, Visual Studio produced 0 errors and 5 warnings.



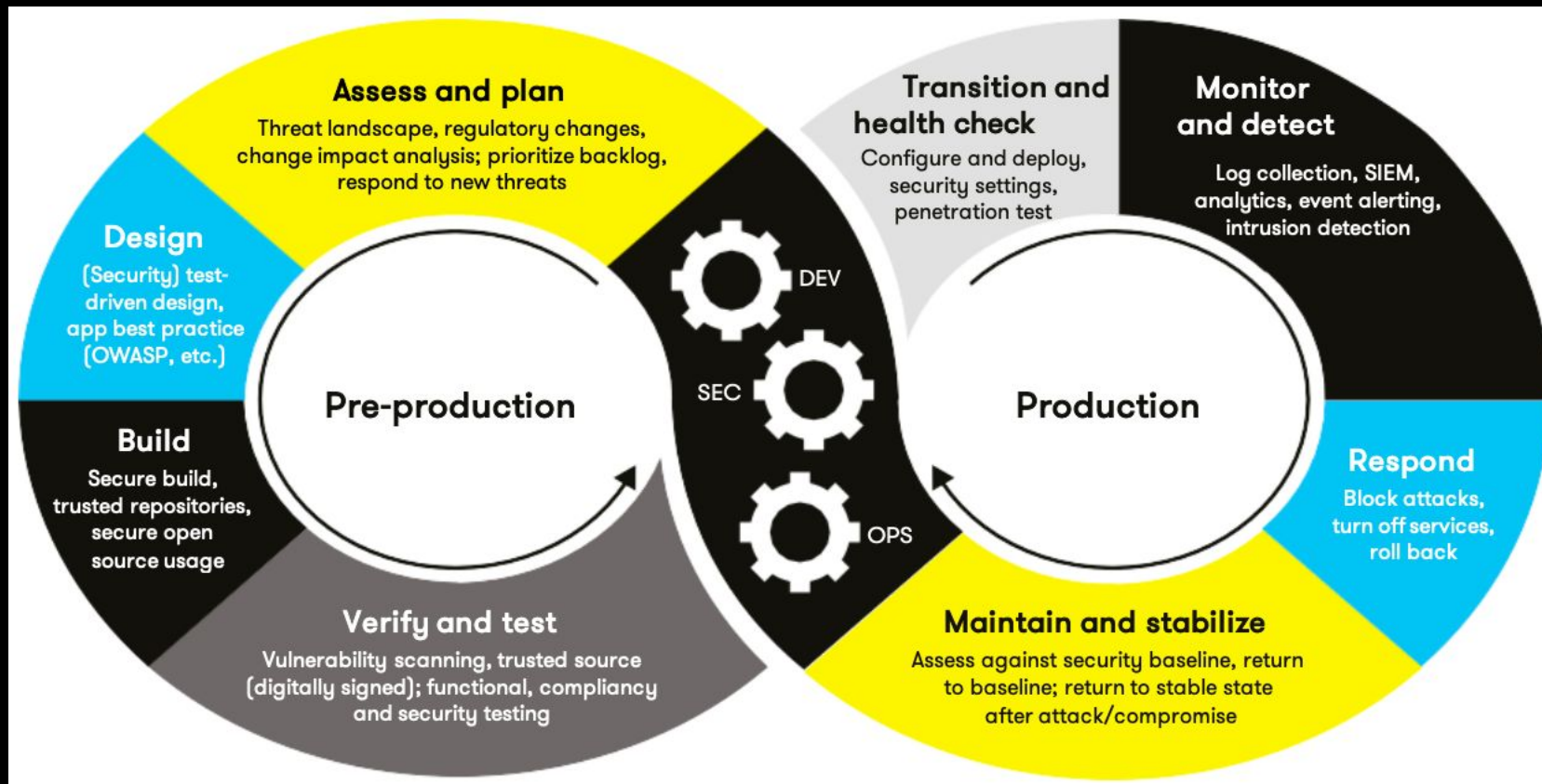
The screenshot shows the Cppcheck 2.10 interface. The top panel displays three error messages:

File	Severity	Line	Summary
Static_Code_Analysis/Static_Code_Analysis.cpp	error	59	Address of local auto-variable assigned to a function parameter.
Static_Code_Analysis/Static_Code_Analysis.cpp	error	52	Exception thrown in function declared not to throw exceptions.
Static_Code_Analysis/Static_Code_Analysis.cpp	error	87	Using iterator to local container 'items' that may be invalid.

Below the error messages, the source code is displayed. The code includes a class `A` with a method `MyVal` and a function `work_with_arrays`. The error messages correspond to the following lines in the code:

- Line 59: `int MyVal = 1;` (Assigned to a function parameter).
- Line 52: `throw "Ha! I threw anyway!";` (Exception thrown in function declared not to throw exceptions).
- Line 87: `items.push_back(i);` (Using iterator to local container 'items' that may be invalid).

AUTOMATION SUMMARY



TOOLS

- The DevSecOps pipeline is a software development approach that incorporates security measures throughout the development cycle. Continuous security testing, vulnerability assessments, and risk management are all part of ensuring that security is incorporated into the application from the start, rather than being an afterthought.
- The choice to automate a process should be based on a comprehensive examination of the task's complexity, process flow, implementation costs, and possible advantages. Automation should ideally be deployed at the start of the process to have the most impact. However, it is critical to properly test the automation to verify that it functions as intended.

RISKS AND BENEFITS

When coding, there will always be some danger because nothing is completely safe. Assume that there are dangers and faults in the system and remain committed to staying up to date on all of today's common threats and preventive measures, since continual education is crucial to the success of this strategy.

RECOMMENDATIONS

Identifying security policy gaps entails assessing existing policies and processes and identifying places where they do not sufficiently address security concerns. Some typical security policy gaps include a lack of clear password management rules, inability to address emerging risks, insufficient staff training and awareness programs, insufficient access controls, and poor incident response strategies. Failure to comply with industry standards or legal requirements, a lack of monitoring and accountability, and a failure to undertake regular security assessments or audits are all examples of gaps. It is critical to identify and close these gaps in order to guarantee that the security strategy is comprehensive and successful in safeguarding the organization's assets and data.

CONCLUSIONS

Organizations should develop standards that provide a complete framework for executing effective security measures to avoid future security issues. ISO/IEC 27001 and NIST SP 800-53 standards give instructions for building and maintaining an information security management system that comprises policies, procedures, and controls to safeguard sensitive data and resources. To prevent vulnerabilities in the software development process, businesses may consider implementing guidelines for safe coding techniques such as the OWASP Top Ten or the CERT safe Coding guidelines. Data encryption and protection standards such as AES or RSA, as well as compliance requirements for specific industries or laws such as HIPAA, PCI DSS, or GDPR, may also be used. Organizations may build a great reputation by adopting and adhering to these criteria.

REFERENCES

Top 10 Secure Coding Practices - CERT Secure Coding - Confluence. (2009, January 23). Top 10 Secure Coding Practices - CERT Secure Coding - Confluence.

<https://wiki.sei.cmu.edu/confluence/display/seccode/Top+10+Secure+Coding+Practices>

SEI CERT Coding Standards - CERT Secure Coding - Confluence. (2022, November 26). SEI CERT Coding Standards - CERT Secure Coding - Confluence.

<https://wiki.sei.cmu.edu/confluence/display/seccode/SEI+CERT+Coding+Standards>