

Green Pace

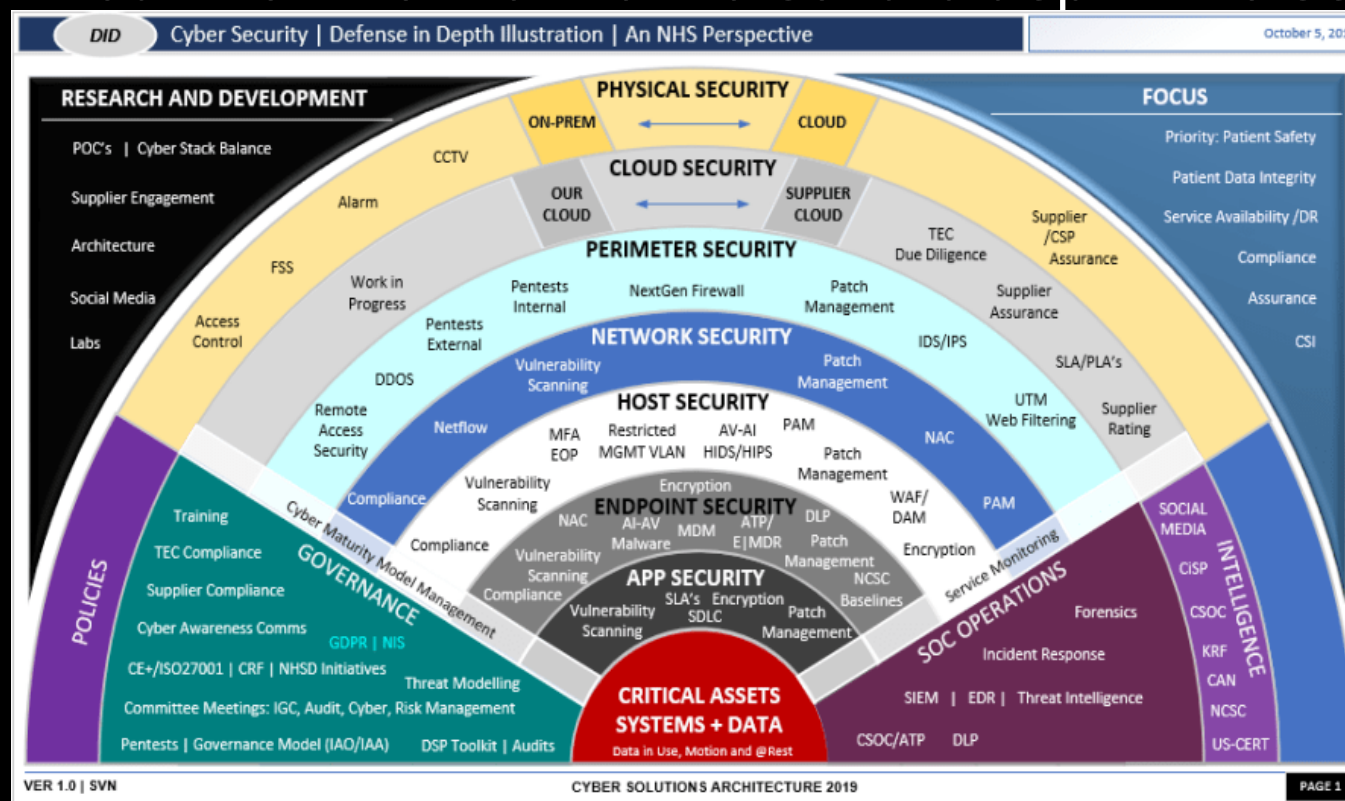
Security Policy Presentation
Developer: *Brandon Womack*



Green Pace

OVERVIEW: DEFENSE IN DEPTH

This model is to give an overview of the detailed methods of defense that we use in our work to maintain a solid blueprint to secure coding.



THREATS MATRIX

Secure Coding standards come with certain levels of vulnerability to measure the impact of that certain standard. Here is a chart to prioritize those levels.

<p>Likely Threats that are very likely to happen</p>	<p>Priority Standard with high relevancy</p>
<p>Low priority Standard with low relevancy.</p>	<p>Unlikely Threats are not as likely to happen.</p>

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 Defense 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. Use valid references, pointers, and iterators to reference elements of a container
- 3. Do not attempt to create a `std::string` from a null pointer
- 4. Do not store already-owned pointer value in an unrelated smart pointer
- 5. Properly deallocate dynamically allocated resources
- 6. Use a static assertion to test the value of a constant expression
- 7. Handle all exceptions thrown before `main()` begins executing
- 8. Do not alternately input and output from a file stream without an intervening positioning call
- 9. Do not invoke virtual functions from constructors or destructors
- 10. Value returning functions must return a value from all exit paths

ENCRYPTION POLICIES

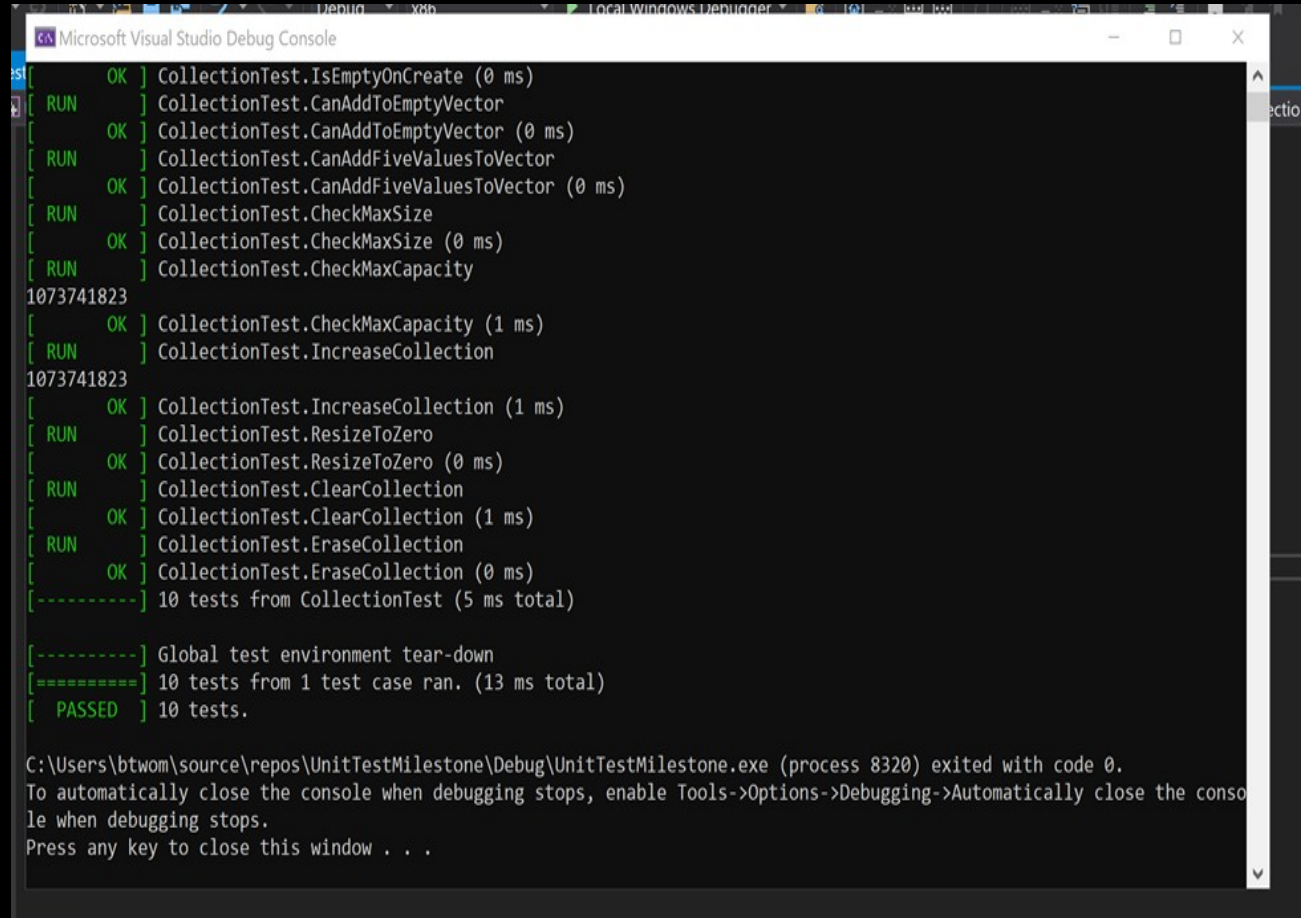
- **Encryption in rest** - Encryption at rest is designed to prevent the attacker from accessing the unencrypted data by ensuring the data is encrypted when on disk. If an attacker obtains a hard drive with encrypted data but not the encryption keys, the attacker must defeat the encryption to read the data.
- **Encryption at Flight** - The process of encrypting data while the data is being transmitted. In some applications, such as remote replication, data may be unencrypted while it is at rest on drive arrays, but encrypted while it is being transmitted to provide protection.
- **Encryption in Use** - Compromising data in use enables access to encrypted data at rest and data in motion. For example, someone with access to random access memory can parse that memory to locate the encryption key for data at rest. Once they have obtained that encryption key, they can decrypt encrypted data at rest.

TRIPLE-A POLICIES

- **Authentication** - Authentication is the process where the user is being confirmed as someone who has access to the system. This can include user login and password information for the user to be able to access parts of the system. Some newer methods use 2 step authentication or multi tier authentication.
- **Authorization** - Authorization is the level of access that a user has within the system. This can include if the user can read, create, delete, or modify files within the database. This can also lead to access to whether a user can add or delete files and users within the system.
- **Accounting** - Accounting is the process of monitoring what a user is doing with their level of access to the system. This will keep track of what databases are accessed, what was done when it was accessed, and what user accessed the system to begin with.

Unit Testing

Our unit test practices come early and often through out the development process to ensure that we have secure functioning code. Ex. Limiting the number of characters within a user input string to prevent buffer overflow.

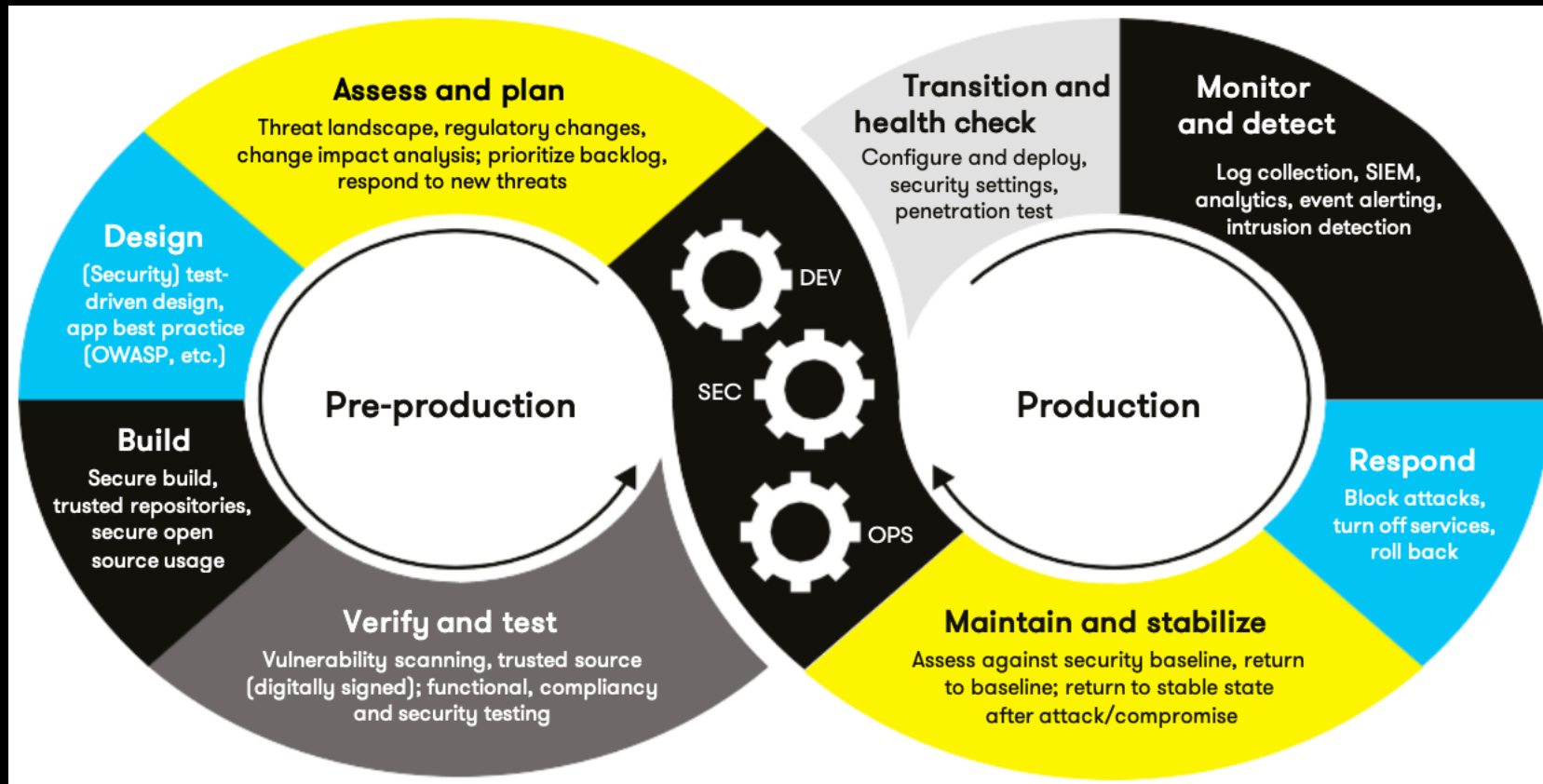


```
Microsoft Visual Studio Debug Console
[ OK ] CollectionTest.IsEmptyOnCreate (0 ms)
[ RUN ] CollectionTest.CanAddToEmptyVector
[ OK ] CollectionTest.CanAddToEmptyVector (0 ms)
[ RUN ] CollectionTest.CanAddFiveValuesToVector
[ OK ] CollectionTest.CanAddFiveValuesToVector (0 ms)
[ RUN ] CollectionTest.CheckMaxSize
[ OK ] CollectionTest.CheckMaxSize (0 ms)
[ RUN ] CollectionTest.CheckMaxCapacity
1073741823
[ OK ] CollectionTest.CheckMaxCapacity (1 ms)
[ RUN ] CollectionTest.IncreaseCollection
1073741823
[ OK ] CollectionTest.IncreaseCollection (1 ms)
[ RUN ] CollectionTest.ResizeToZero
[ OK ] CollectionTest.ResizeToZero (0 ms)
[ RUN ] CollectionTest.ClearCollection
[ OK ] CollectionTest.ClearCollection (1 ms)
[ RUN ] CollectionTest.EraseCollection
[ OK ] CollectionTest.EraseCollection (0 ms)
----- 10 tests from CollectionTest (5 ms total)

----- Global test environment tear-down
===== 10 tests from 1 test case ran. (13 ms total)
[ PASSED ] 10 tests.

C:\Users\btwom\source\repos\UnitTestMilestone\Debug\UnitTestMilestone.exe (process 8320) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```


AUTOMATION SUMMARY



TOOLS

- The DevSecOps pipeline is a secure coding method that has a full circle approach to enforcing a policy that has an infrastructure built on efficiently keeping code secure.
- This is a solid structure for the system, I would just always be mindful of defense in depth and making sure that you are testing early and often to detect any flaws or vulnerabilities so that we can be able to catch those bugs and errors early.

RISKS AND BENEFITS

- There will always be risk when coding because nothing can be 100% secure. Always assume that there are threats and flaws in the system and stay persistent with keeping up with all of today's common threats and prevention techniques, so continuing education is highly critical to the success of this policy.

RECOMMENDATIONS

- Keeping up with all the security threats and trends is a critical piece to maintaining the level of security we promise. We keep everything simple yet effective to get the job done.

CONCLUSIONS

- In conclusion, With the principles and standards mentioned within this presentation we can conclude that most of all the important topics were covered to display our plan to create and maintain a secure and proficient programing. We also have adopted a zero-trust policy when it comes to accessing things inside and outside of the company network to maintain security and privacy to keep all sensitive information safe and secure for all parties involved.



REFERENCES

- Seacord, R. (2020, November 18). Confluence. Retrieved from SEI External Wiki Home: <https://wiki.sei.cmu.edu/confluence/>

