7-2 Project Two Script

Paloma Rodriguez

CS-405: Secure Coding

December 2nd, 2023

Link: https://youtu.be/aNIHIQE8H6Y

Transcript:

Slide 1

Good evening, everyone. Today, we're going to delve into our newly crafted security policy, designed to fortify our defense mechanisms against evolving threats. This presentation will outline why this policy is crucial and how it aligns with the defense-in-depth best practice. Let's begin.

Slide 2

Let's talk about a fundamental principle in our security strategy - defense in depth. Our digital home network is like a castle. Now, if we were to protect this castle, we wouldn't rely solely on a single line of defense, right? We'd have multiple layers of security, each complementing the other.

Defense in depth means having a series of security measures at various layers, creating a robust and layered defense strategy. It's like having multiple gates, guards, and security checks, ensuring that even if one layer is breached, there are others in place to thwart potential threats.

We integrate security at every level - from the code we write to the infrastructure supporting our applications. It's about being proactive, anticipating different attack, and creating a strong environment that can withstand various threats.

Slide 3

Let's discuss our Threat Matrix - a tool that helps us assess and prioritize potential risks.

Firstly, we have threats categorized as 'likely.' These are the ones that have a higher probability of occurring. It's essential to be vigilant against these and implement measures to counteract them.

Then, there are threats labeled 'priority.' These are the ones that, even if less likely, could have a significant impact on our systems or data. We give them special attention because of their potential severity.

Moving on, we have threats marked as 'low priority.' These are scenarios that are less likely to happen, and even if they do, they might not cause extensive damage. We keep an eye on them but may allocate fewer resources to address them.

Then are threats labeled as 'unlikely.' These are the least probable scenarios, and while we acknowledge them, they might not be immediate concerns. We monitor them but don't prioritize them as heavily in our security efforts.

Slide 4

Here, we list the 10 principles guiding our coding standards.

Let’s dive into the core principles that shape our approach to security. These guiding principles not only serve as the foundation of our secure coding practices but also form a robust framework for our entire development process.

1. Validate Input Data

First, we validate input data. By ensuring that the data our system receives is accurate and within expected parameters, we prevent potential injection attacks and enhance the integrity of our applications.

2. Heed Compiler Warnings

Compiler warnings act as our early warning system. We pay close attention to these messages, addressing them promptly. It's a proactive measure to catch potential vulnerabilities at the earliest stage of development.

3. Architect and Design for Security Policies

Security is not an afterthought; it's woven into our architecture and design. By aligning our systems with security policies from the outset, we create a solid foundation for a secure and resilient application.

4. Keep It Simple

Simplicity is a virtue in security. By keeping our code and designs straightforward, we reduce the potential attack surface and make it easier to identify and address security vulnerabilities.

5. Default Deny

We follow the 'Default Deny' principle, allowing only what is explicitly permitted. This minimizes the risk of unauthorized access and strengthens our overall security posture.

6. Adhere to the Principle of Least Privilege

Every user and system component has precisely the access they need – nothing more, nothing less. This principle of least privilege ensures that potential damage is limited even in the event of a compromise.

7. Sanitize Data Sent to Other Systems

When interacting with other systems, we ensure the data we send is thoroughly sanitized. This minimizes the risk of unintentionally exposing sensitive information or introducing vulnerabilities into external systems.

8. Practice Defense in Depth

Our security strategy is layered. We practice defense in depth, incorporating multiple security measures at various levels. It's an approach that ensures our systems remain resilient even if one layer is breached.

9. Use Effective Quality Assurance Techniques

Quality assurance is inseparable from security. We employ effective QA techniques to rigorously test our code, identifying and addressing potential vulnerabilities before they reach production.

10. Adopt a Secure Coding Standard

Our coding standard is not just a set of rules; it's a commitment to security. By adopting a secure coding standard, we ensure consistency in our codebase, making it more resistant to common vulnerabilities.

These principles collectively form the bedrock of our security mindset, guiding us towards creating and maintaining secure systems for Green Pace.

Slide 5

Prioritizing our coding standards is crucial. On this slide, you'll find our 10 coding standards listed in priority order. This system ensures that we address the most critical aspects of security first, enhancing the effectiveness of our development process.

1.Obey the One-Definition Rule:

First on our list is adhering to the One-Definition Rule. This principle ensures that each entity in our code is defined in a single place, preventing any unclarity and maintaining code clarity.

2.Do not write syntactically ambiguous declarations:

Following closely, we emphasize clarity in our code by avoiding syntactically ambiguous declarations. This enhances readability and reduces the risk of unintended behavior.

3.Guarantee that storage for strings has sufficient space for character data and the null terminator:

Next, we focus on string handling. Guaranteeing sufficient storage for strings, including the null terminator, is crucial to prevent buffer overflows and enhance the security of our applications.

4.Prevent SQL injection:

Security is at the forefront of our concerns. Preventing SQL injection is a key standard. By validating and sanitizing inputs, we fortify our applications against this attack vector.

5.Close files when they are no longer needed:

Resource management is a priority. We ensure that files are closed promptly when they are no longer needed, preventing potential issues related to resource leaks.

6.Do not access an object outside of its lifetime:

Maintaining code safety, we strictly adhere to not accessing an object outside of its lifetime. This prevents undefined behavior and contributes to the overall stability of our applications.

7.Do not access freed memory:

Memory management is a critical aspect of our coding standards. We strictly enforce not accessing freed memory to mitigate risks associated with dangling pointers and memory corruption.

8.Handle all exceptions:

Exception handling is integral to our code reliability. We ensure that all exceptions are handled appropriately, preventing unexpected program termination and providing a graceful response to unforeseen circumstances.

9.Gracefully handle self-copy assignment:

Maintaining code robustness, we place emphasis on gracefully handling self-copy assignment. This prevents unintended consequences and contributes to the predictability of our codebase.

10.Avoid deadlock by locking in a predefined order:

Concurrency is managed diligently. We avoid deadlock situations by adhering to a predefined order when locking resources, ensuring smooth execution even in multithreaded environments.

These coding standards collectively embody our commitment to producing high-quality, secure software here at Green Pace. By adhering to these principles, we ensure that our codebase remains resilient and trustworthy.

Slide 6

Our encryption strategy encompasses three main areas: in-flight, at-rest, and in-use. This slide summarizes each, ensuring comprehensive protection of sensitive data.

1. Encryption at Rest:

Imagine our data taking a nap in a safe. We use encryption to lock it up. Even if someone finds the safe, they can't understand what's inside without the special key – it's our way of keeping data safe when it's not doing anything

2.Encryption in Flight:

Now, when our data is on the move, like when we send it between computers or over the internet, we put it in a secret envelope. If someone tries to peek inside, all they see is gibberish. The real meaning is only known to us and the intended recipient.

3.Encryption in Use:

Lastly, even when our data is busy working in our programs, we keep it under wraps. It's like putting on an invisibility cloak – others might see it's there, but they can't understand it. We use encryption to make sure our data stays safe even when it's actively in use

By integrating encryption at rest, in flight, and in use, we establish a comprehensive security framework that safeguards our data throughout its lifecycle. It's a crucial part of our commitment to maintaining the confidentiality and integrity of the information we handle here at Green Pace.

Slide 7

Authentication, Authorization, and Accounting are the pillars of our Triple-A Framework. This slide provides a concise summary of the policies supporting each aspect, ensuring a robust security infrastructure.

Authentication is like checking your ID at the door. It's about verifying who you are. We use things like usernames and passwords to make sure you are who you say you are.

Authorization is like deciding what rooms you can enter once you're inside. Even if we know who you are, we need to determine what you're allowed to access. This is where we set up rules like 'Who can edit this document?' or 'Who can access this part of the system?' - it's about giving the right permissions to the right people.

Accounting is like having a watchful eye over everything. We keep a record of who did what and when. If something unexpected happens, we can look at these records to see what went wrong or who might need extra training. It's our way of keeping everyone accountable and ensuring that our systems are used the way they're supposed to be.

Triple-A is the key to keeping our digital doors secure – making sure the right people get in, they only access what they're allowed to, and we keep a watchful eye on everything that happens. It's the backbone of our security strategy here at Green Pace.

Slide 8

Let’s dive into the world of unit testing, a crucial aspect of our development process. Unit testing might sound more technical, but at its core, it's our way of ensuring the reliability and security of our software.

So, why do we do it?

Well, imagine building a complex structure – you wouldn't wait until the whole thing is complete to check if it stands strong, right? Unit testing is like checking the individual components as we build. It's our way of catching potential issues early in the game.

By breaking our code into smaller units and testing them individually, we ensure that each piece works as intended. If there's a weak link, we catch it right away, preventing it from becoming a major problem later. It's our safety net, allowing us to deliver not just functional but robust and secure software.

So, as we dive into the unit testing realm, remember, it's not just about finding bugs; it's about building a solid foundation for the reliability and security of our software at every step of the way.

Slide 9

Our DevSecOps diagram illustrates where security tools reside in our automation flow. It highlights the stages where security automation is integrated, such as during the compilation process. Automation is like having a helping hand that never gets tired. It's the art of getting our computers to do repetitive tasks, and it's not just a time-saver; it's a game-changer for efficiency, consistency, and, most importantly, security.

Why do we need automation?

Imagine if every time we had to check our code for security flaws or test new features, we had to do it manually. It would be time-consuming, and humans, being humans, might overlook things. That's where automation steps in.

Automation allows us to set up processes that can run automatically. Need to check for coding vulnerabilities? Automation can do it faster and more consistently than any human. Want to ensure our software meets certain quality standards every time? Automation has our back.

So, in essence, automation is not just about saving time; it's about boosting our efficiency and reinforcing the security of our development process. It's a true ally in our quest for excellence here at Green Pace.

Slide 10

Imagine our pipeline symphony where each instrument plays a crucial role. In the world of software development, our pipeline is a series of tools that seamlessly blend development, security, and operations.

Now, let's look at the example of one of our external tools that play a key role in enhancing the security aspects of our pipeline.

Git is a version control system. It helps us track changes in our code over time, creating a historical record of our project. Git is the backbone of collaboration. Multiple developers can work on the same project simultaneously without stepping on each other's toes. It's like a shared canvas where everyone contributes their piece, and Git ensures a seamless blending of those contributions.

Slide 11

As we consider implementing our security enhancements, it's crucial to assess the risks and benefits. The current state of our strategy, examining the problems we face, proposed solutions, and the associated risks and benefits. This is vital for steering our ship in the right direction.

In the journey of fortifying our security strategy, a critical self-assessment is key. By acknowledging and addressing current problems, implementing timely solutions, and mitigating risks, we pave the way for a more resilient and proactive security posture. Let's embark on this journey with a clear understanding of where we stand and where we aim to be.

Slide 12

As we develop software, we must be vigilant about its security. Regular testing at different stages allows us to uncover potential weaknesses that could expose us to cyber threats.

By adopting these recommendations, we not only identify and fill potential security gaps but also build a strong and resilient security posture throughout our software's journey. It's about staying one step ahead of evolving threats and ensuring the reliability of our software in an ever-changing digital landscape.

Slide 13

Wrapping up, at Green Pace, our dedication to ensuring security throughout our software development life cycle is rooted in our core values. We do this because we believe in:

Building Trust: Security is the foundation of trust, and by addressing potential gaps, we secure the confidence our users place in us.

Prioritizing Customers: We uphold a customer-centric approach, ensuring that the software we deliver is not just functional but reliable and secure.

Setting Industry Standards: As industry leaders, we aim not only to innovate but also to set the standard for secure software development practices.

In essence, our commitment to security is not just a process; it's a reflection of the values that define Green Pace. Thank you for joining us on this journey towards a secure and innovative future. Together, we shape a digital landscape where reliability, integrity, and innovation.

Slide 14

\*Outro Music\*