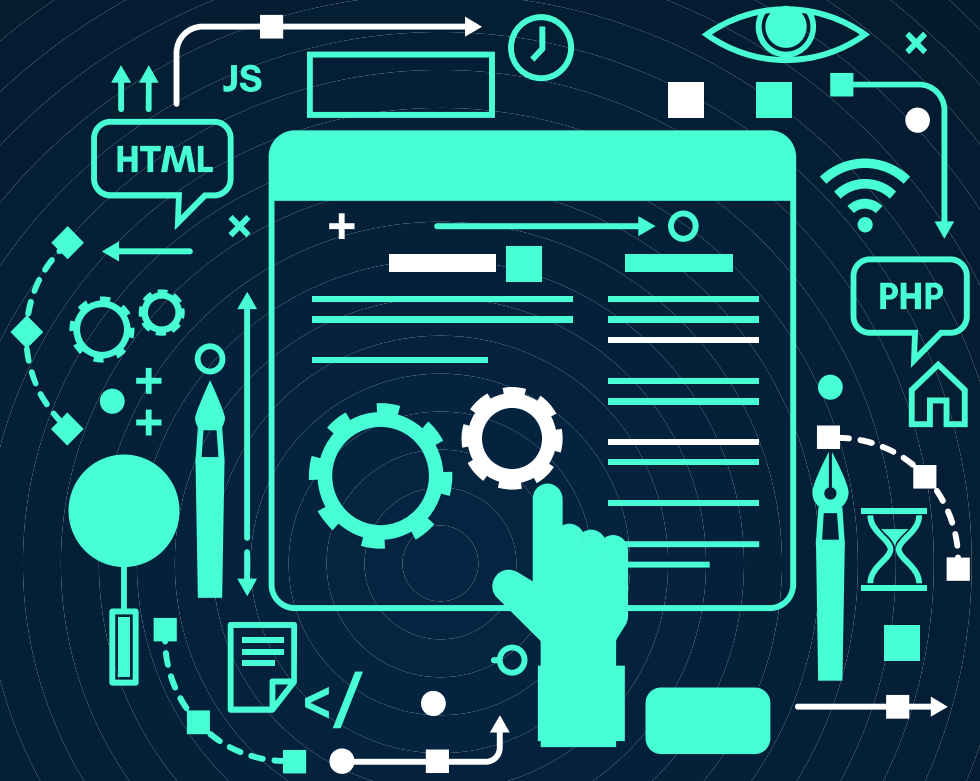


OWASP Top 10 - A04:2021 – Insecure Design

Presented By- Jalaj
null-meet Delhi @
eSec Forte® Technologies, Gurugram
17 December 2022

Jalaj@null:~$ whoami

- Cyber Security Geek
- CTF Player
- Open-Source Enthusiast
- Codes in C++, Python && Go
- CRAC CVE Research Group
- Ex-Suzuki & DCM Hyundai
- <3 Malware Analysis
- Crazy about Bikes and MMA

Structure of Presentation

- Introduction to OWASP Top 10
- A04: Insecure Design
- Demo: CWE-209,CWE-444
- Prevention
- QnA

Open Web Application Security Project® (OWASP)

OWASP TOP 10

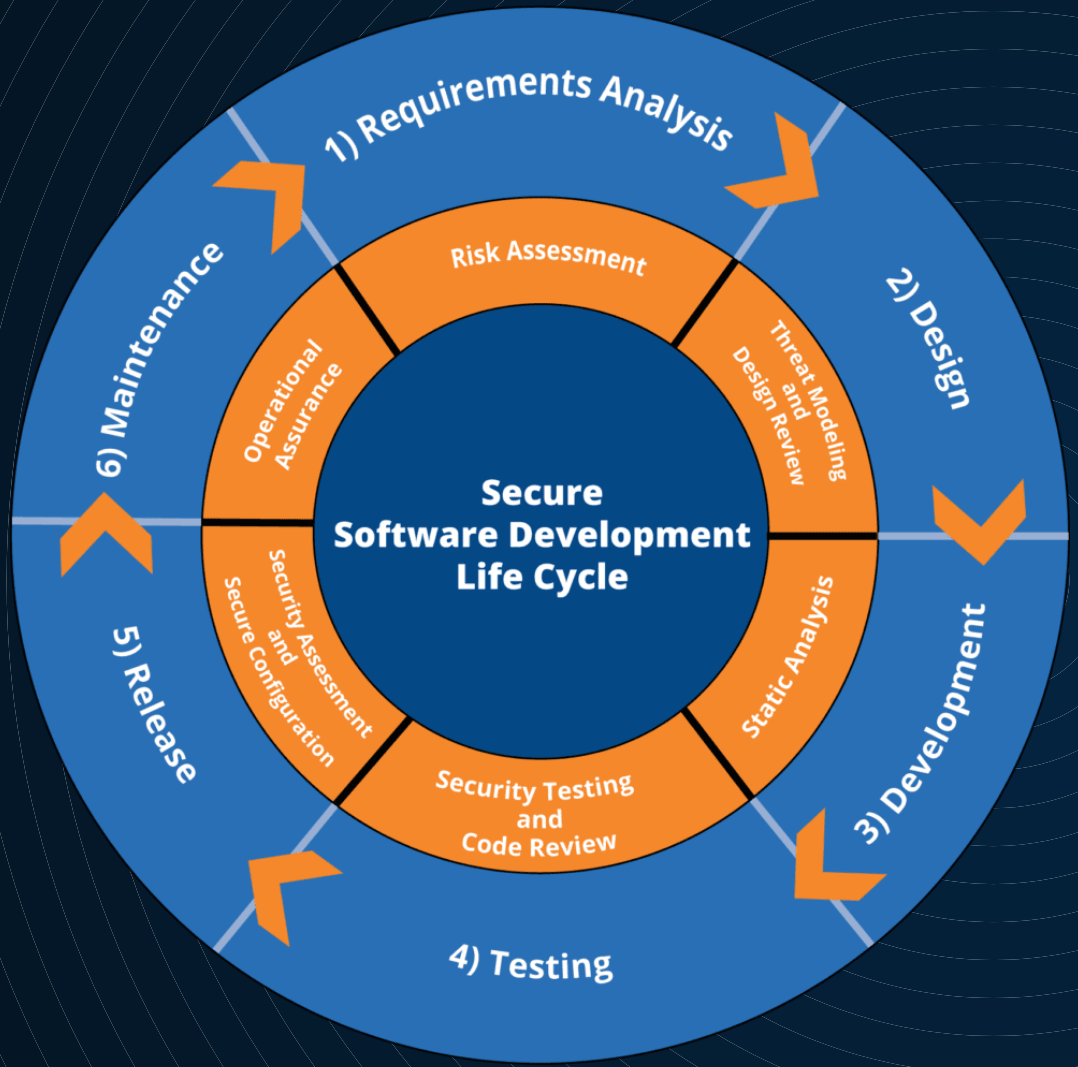
The OWASP Top 10 is a de facto industry standard that provides a list of the 10 Most Critical Web Application Security Risks

- A01 Broken Access Control
- A02 Cryptographic Failures
- A03 Injection
- A04 Insecure Design
- A05 Security Misconfiguration
- A06 Vulnerable and Outdated Components
- A07 Identification and Authentication Failures
- A08 Software and Data Integrity Failures
- A09 Security Logging and Monitoring Failures
- A10 Server Side Request Forgery (SSRF)

There is a difference between
insecure design and insecure implementation.

A04:2021 – Insecure Design

A new category for 2021 focuses on risks related to design and architectural flaws, with a call for more use of threat modelling, secure design patterns, and reference architectures. As a community we need to move beyond "shift-left" in the coding space to pre-code activities that are critical for the principles of Secure by Design.



“Shift-Left” ??

1) Requirements and Resource Management
2) Secure Design
3) Secure Development Lifecycle

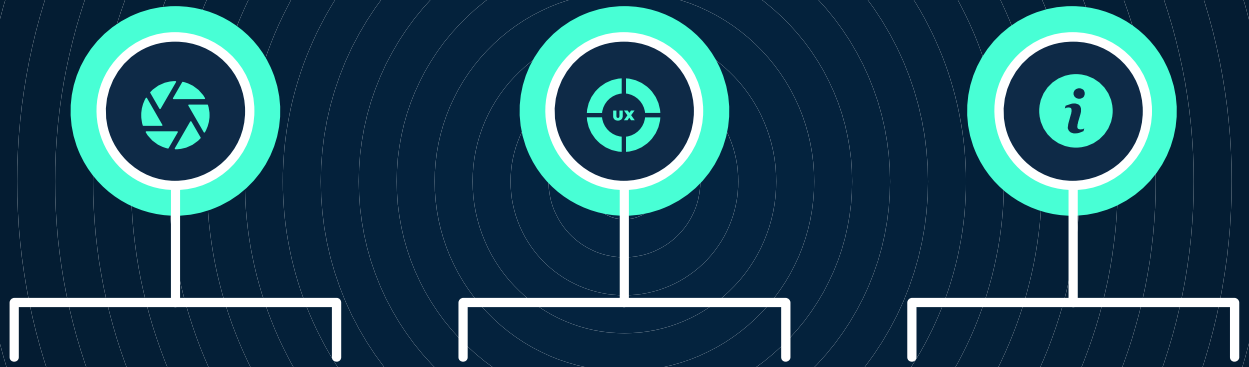
THE SECURE SOFTWARE DEVELOPMENT LIFECYCLE

Mapped Common Weekness Enumeration (CWE’s)

- CWE-73 External Control of File Name or Path
- CWE-183 Permissive List of Allowed Inputs
- CWE-209 Generation of Error Message Containing Sensitive Information
- CWE-213 Exposure of Sensitive Information Due to Incompatible Policies
- CWE-235 Improper Handling of Extra Parameters
- CWE-256 Unprotected Storage of Credentials
- CWE-257 Storing Passwords in a Recoverable Format
- CWE-266 Incorrect Privilege Assignment
- CWE-269 Improper Privilege Management
- CWE-280 Improper Handling of Insufficient Permissions or Privileges
- CWE-311 Missing Encryption of Sensitive Data
- CWE-312 Cleartext Storage of Sensitive Information
- CWE-313 Cleartext Storage in a File or on Disk
- CWE-316 Cleartext Storage of Sensitive Information in Memory
- CWE-419 Unprotected Primary Channel
- CWE-430 Deployment of Wrong Handler
- CWE-434 Unrestricted Upload of File with Dangerous Type
- CWE-444 Inconsistent Interpretation of HTTP Requests ('HTTP Request Smuggling')
- CWE-451 User Interface (UI) Misrepresentation of Critical Information
- CWE-472 External Control of Assumed-Immutable Web Parameter
- CWE-501 Trust Boundary Violation
- CWE-522 Insufficiently Protected Credentials
- CWE-525 Use of Web Browser Cache Containing Sensitive Information
- CWE-539 Use of Persistent Cookies Containing Sensitive Information
- CWE-579 J2EE Bad Practices: Non-serializable Object Stored in Session
- CWE-598 Use of GET Request Method With Sensitive Query Strings
- CWE-602 Client-Side Enforcement of Server-Side Security
- CWE-642 External Control of Critical State Data
- CWE-646 Reliance on File Name or Extension of Externally-Supplied File
- CWE-650 Trusting HTTP Permission Methods on the Server Side
- CWE-653 Insufficient Compartmentalization
- CWE-656 Reliance on Security Through Obscurity
- CWE-657 Violation of Secure Design Principles
- CWE-799 Improper Control of Interaction Frequency
- CWE-807 Reliance on Untrusted Inputs in a Security Decision
- CWE-840 Business Logic Errors
- CWE-841 Improper Enforcement of Behavioral Workflow
- CWE-927 Use of Implicit Intent for Sensitive Communication
- CWE-1021 Improper Restriction of Rendered UI Layers or Frames
- CWE-1173 Improper Use of Validation Framework

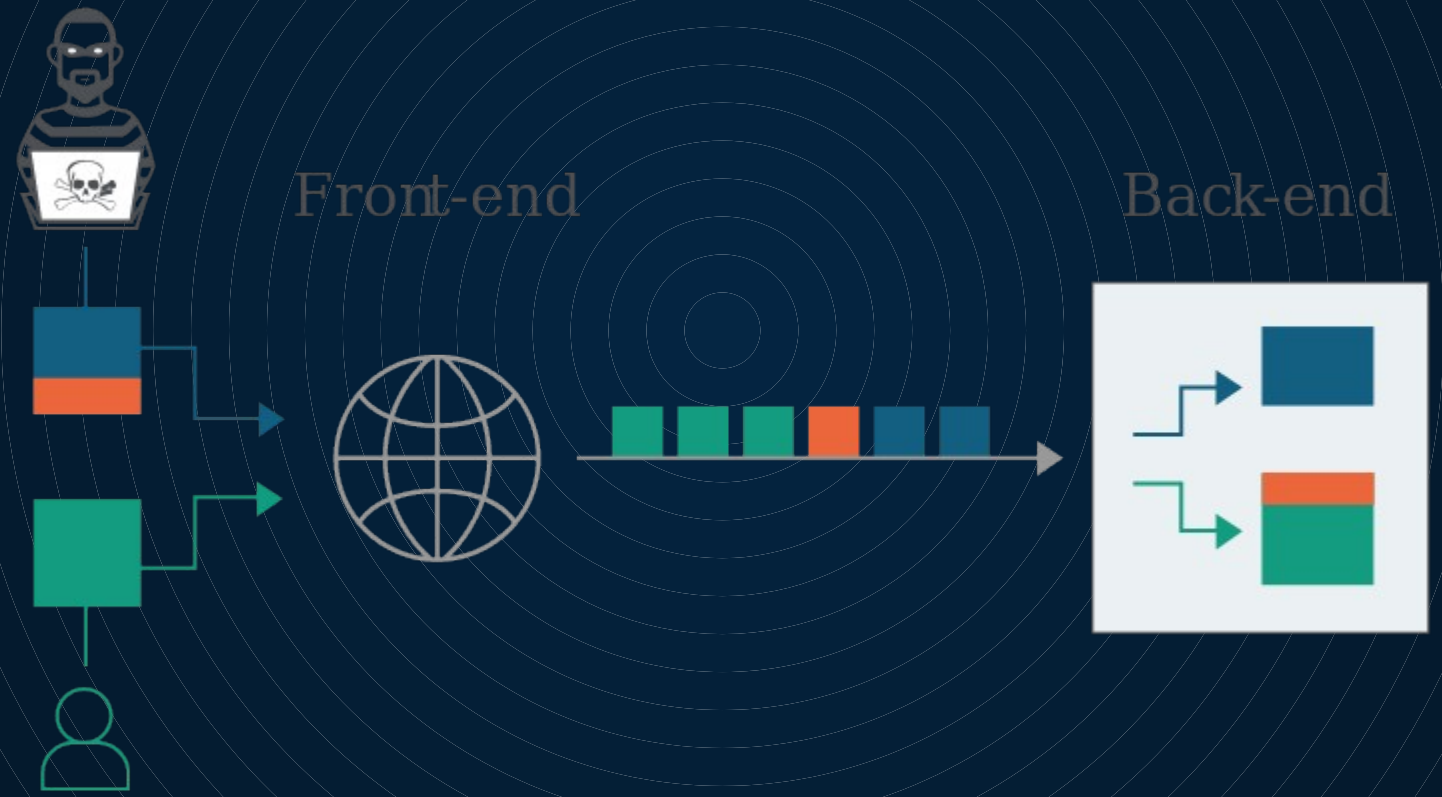
CWE-209: Generation of Error Message Containing Sensitive Information

How do information disclosure vulnerabilities arise?

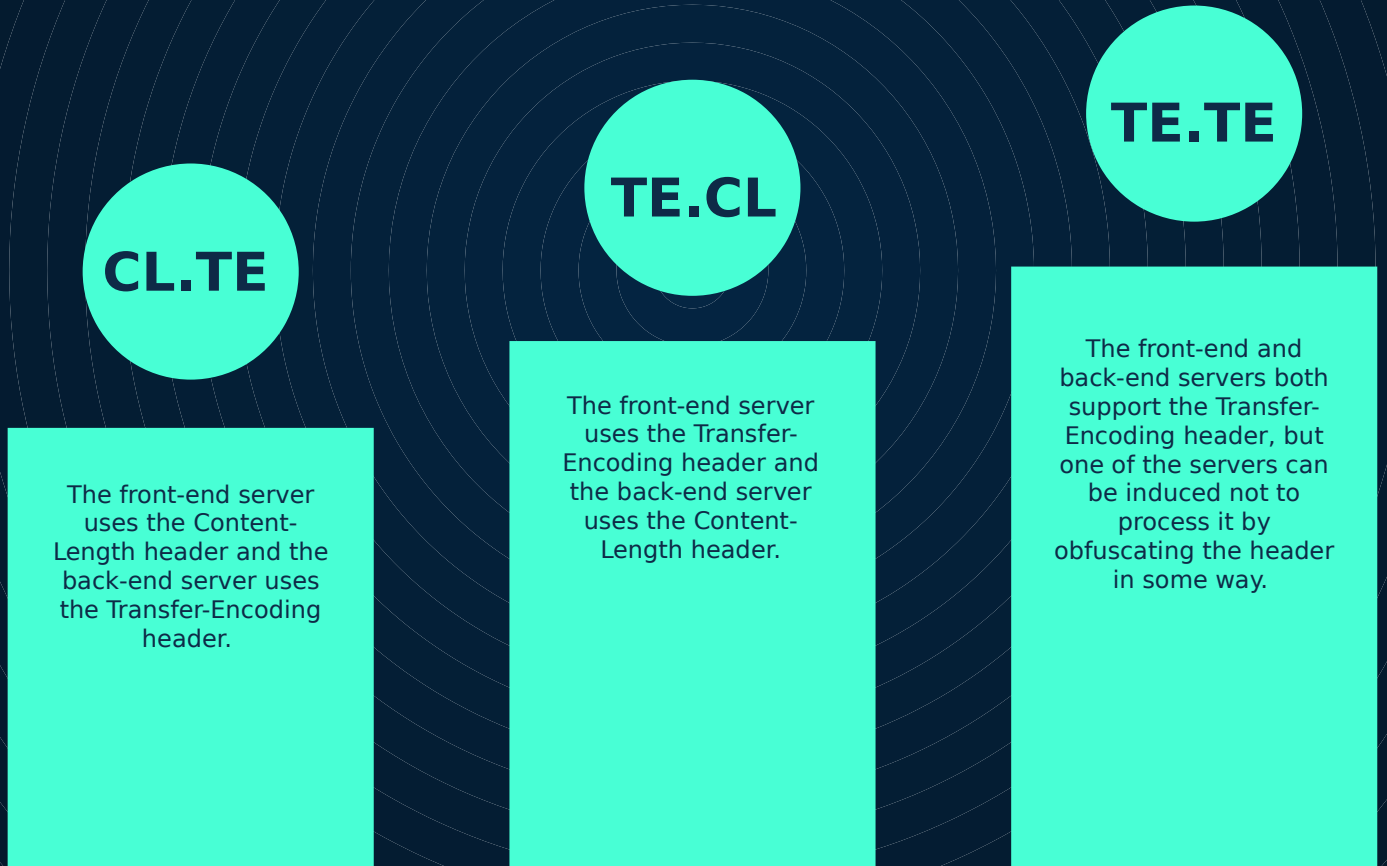


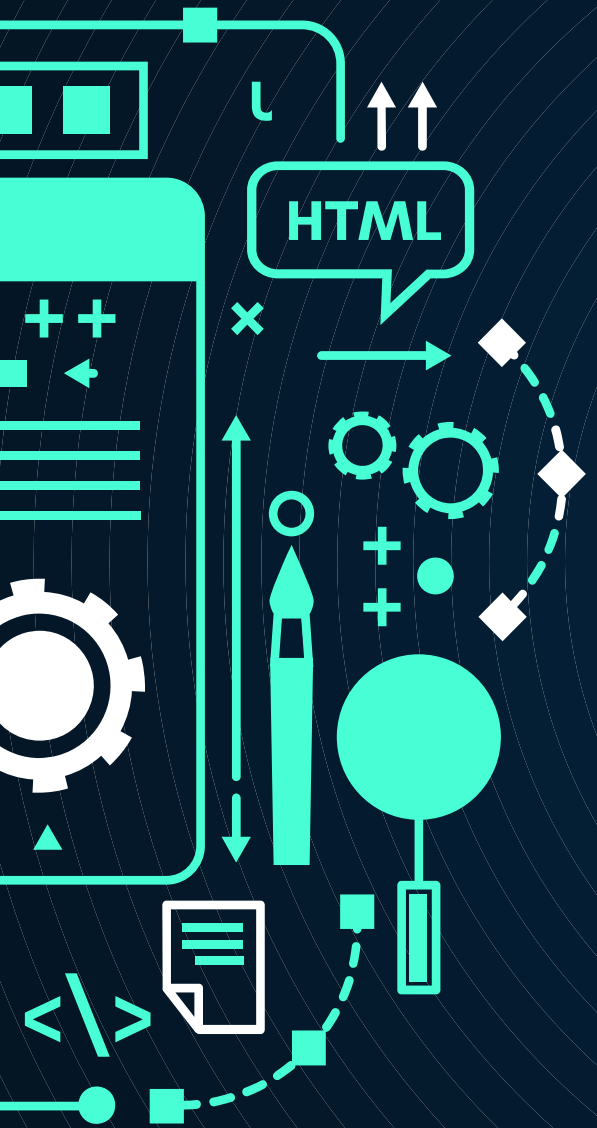
(DEMO)

CWE-444 Inconsistent Interpretation of HTTP Requests ('HTTP Request Smuggling')



CWE-444 Inconsistent Interpretation of HTTP Requests ('HTTP Request Smuggling')



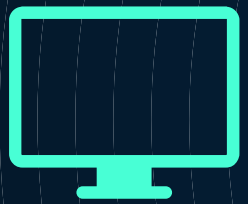


Prevention

- Establish and use a secure development lifecycle with AppSec professionals to help evaluate and design security and privacy-related controls
- Establish and use a library of secure design patterns or paved road ready to use components
- Use threat modeling for critical authentication, access control, business logic, and key flows
- Integrate plausibility checks at each tier of your application (from frontend to backend)
- Limit resource consumption by user or service

THREAT MODELING MANIFESTO

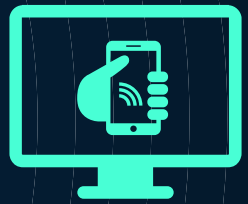
When you perform threat modeling, you begin to recognize what can go wrong in a system.



What are we working on?

What can go wrong?

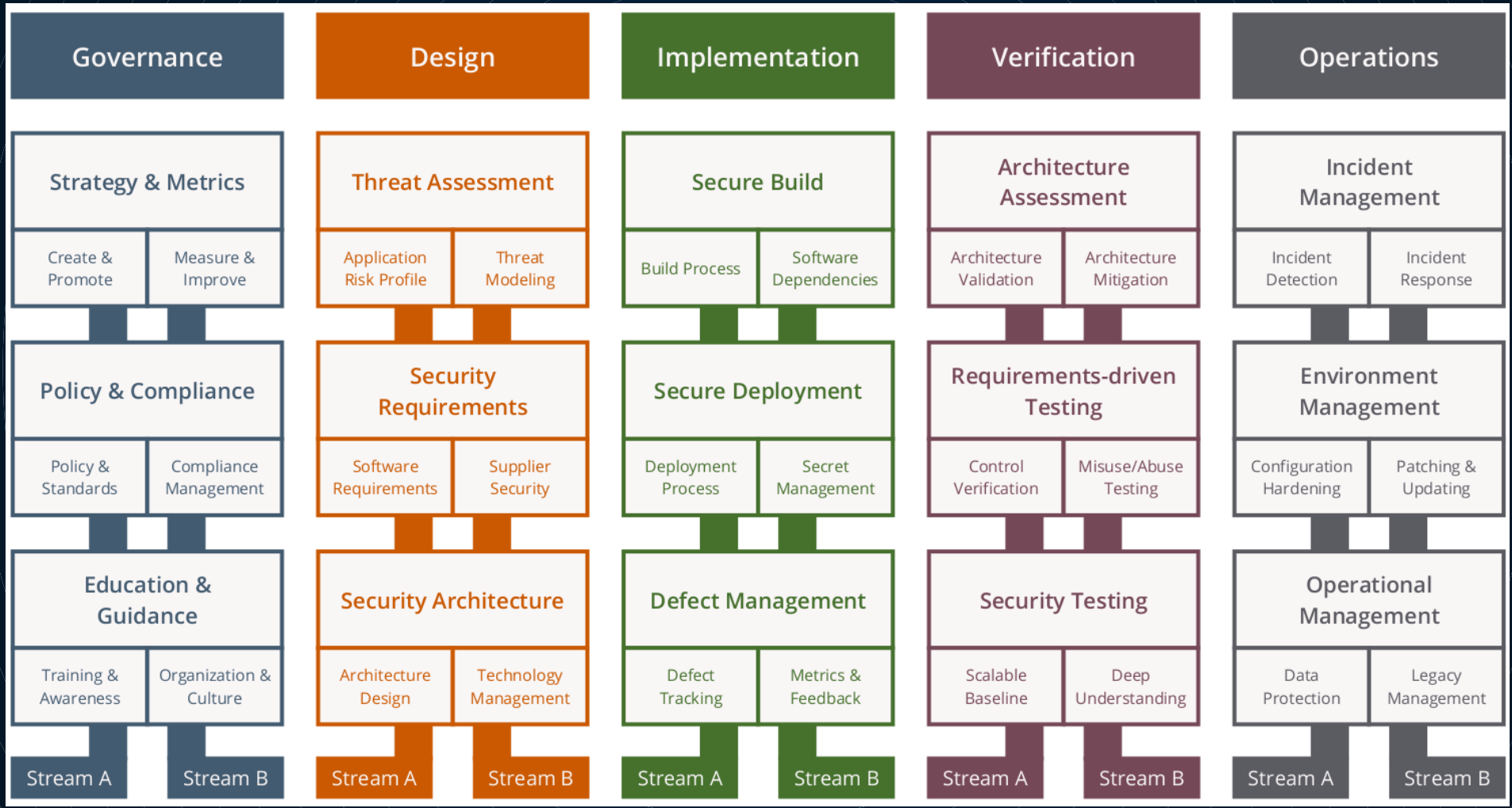
What are we going to do about it?



Did we do a good enough job?

Threat Modeling Tool

OWASP SAMM (Software Assurance Maturity Model)



Thank you,

Any questions ?