BGSW_SJCE

TRAIN THE TRAINERS TRAINING

SECURITY TESTING

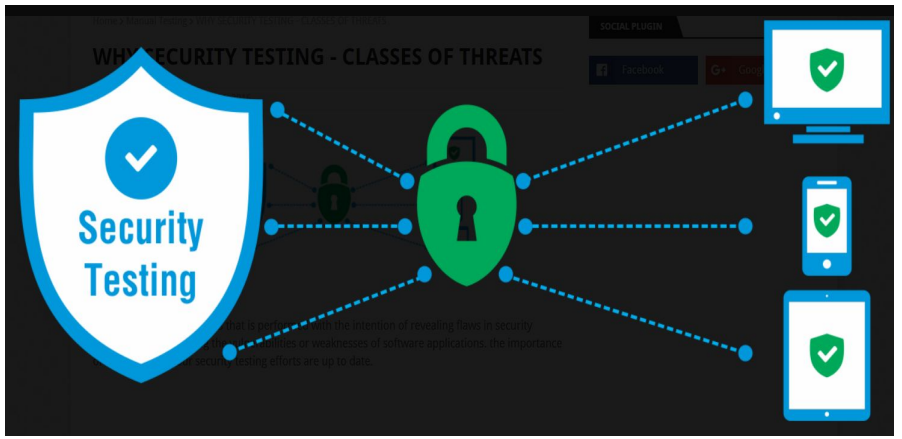
BOSCH

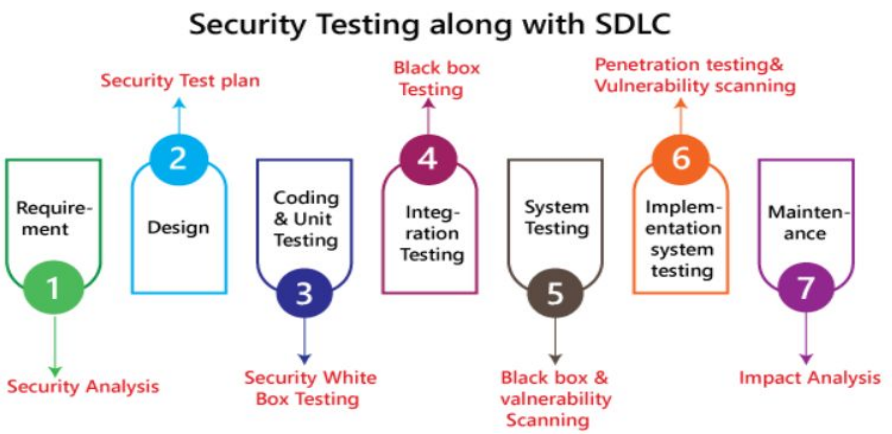
BGSW Curriculum
Security Testing

Security testing is a process that is performed with the intention of revealing flaws in security mechanisms and finding the vulnerabilities or weaknesses of software applications.

Developing and maintaining secure and robust products is an important pillar for product development organizations. During the whole product's lifecycle, security testing is used to ensure the security properties of the product are met.

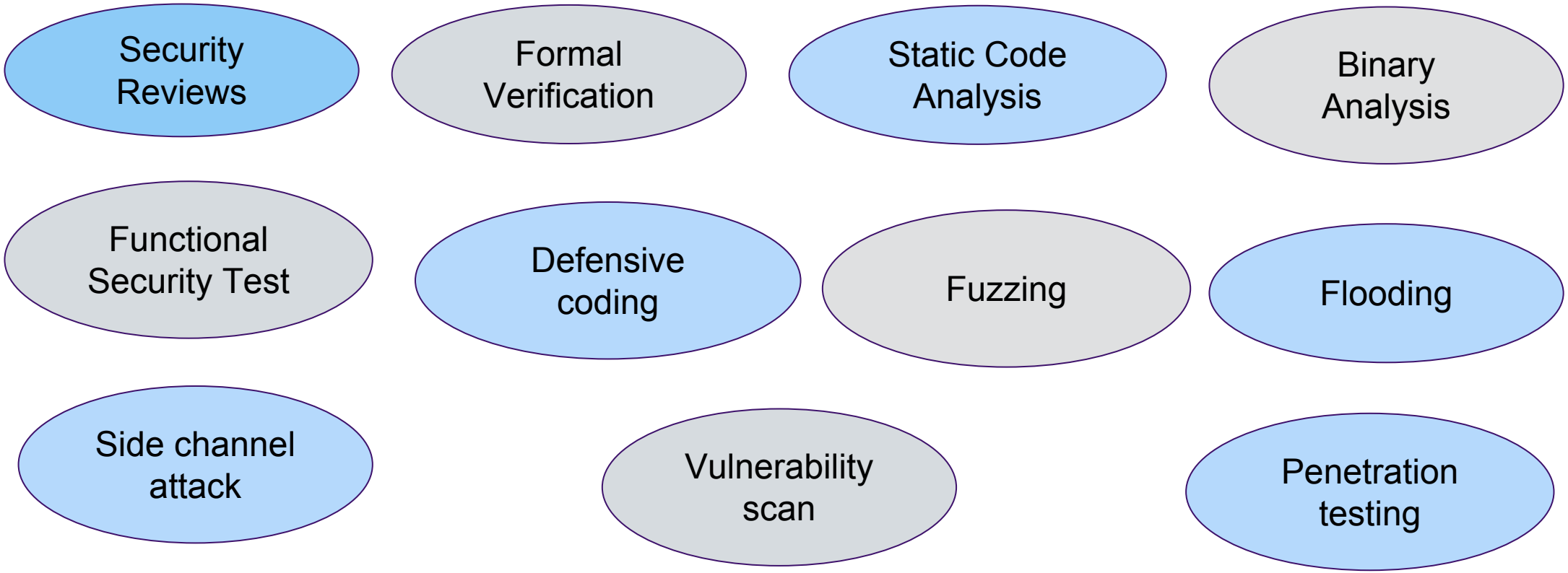
When a product is developed according to the Security Engineering Process **[SEP]**, the security concept contains the security requirements. Based on the security requirements a security testing concept shall be developed.





BGSW Curriculum

Few Security Testing Methodologies

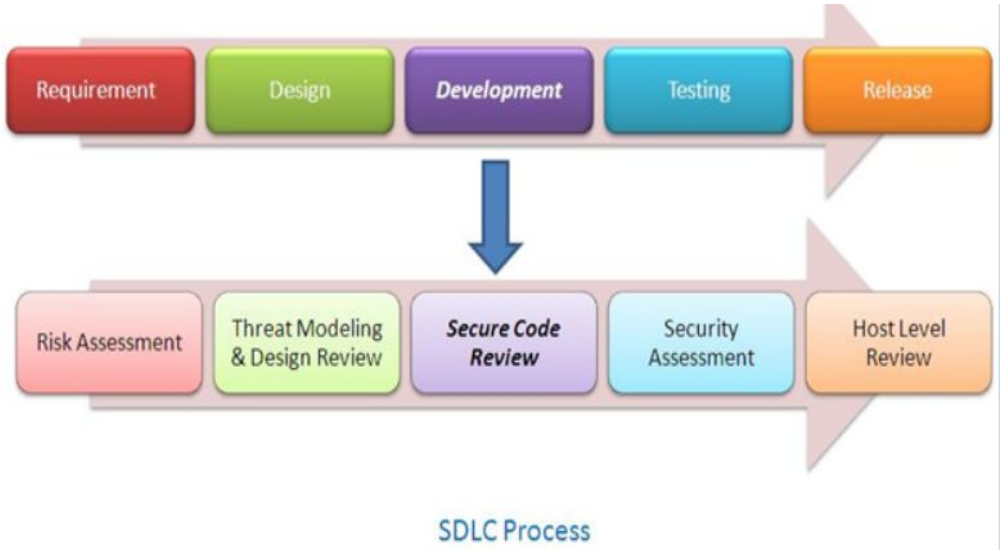


BOSCH

BGSW Curriculum

Security Review

A security review is a process during which security-relevant artefacts are examined by project personnel, managers, users, customers, user representatives, or other interested parties for comment or approval.



SDLC Process

Typical Use Cases:

- Design phase : Helps to identify the high-risk areas, flaws in the security mechanism and offers the concise security recommendation. Eg: Encryption of the data stored in the external flash…
- Code Review: Helps to identify critical security defects in the implementation and prevents the problem from propagating.eg: Buffer overflow, boundary condition, coding guidelines followed…

Test Goals:

Security reviews can be used in every phase of the product development. E.g., Requirement review, design review, code review, test strategy review…

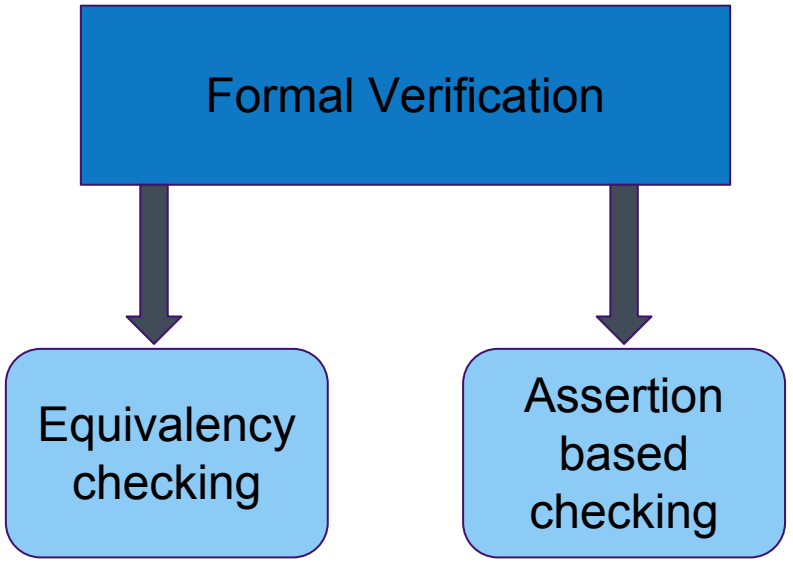
BOSCH

BGSW Curriculum
Formal Verification

Formal verification is an approach to prove or disprove the correctness of an algorithm with respect to certain formal specification.

Typical Use Cases:
- Verify the correctness of the cryptographic Algos. Eg: SHA-256, AES-CTR -128
- Assertion checks for the length of static data

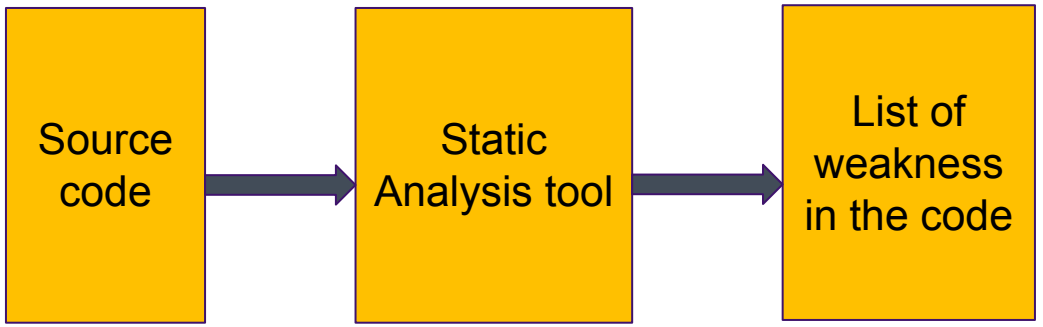
Test Goal:
The goal of this test method is to prove or to disprove the correctness of an algorithm.



BGSW Curriculum

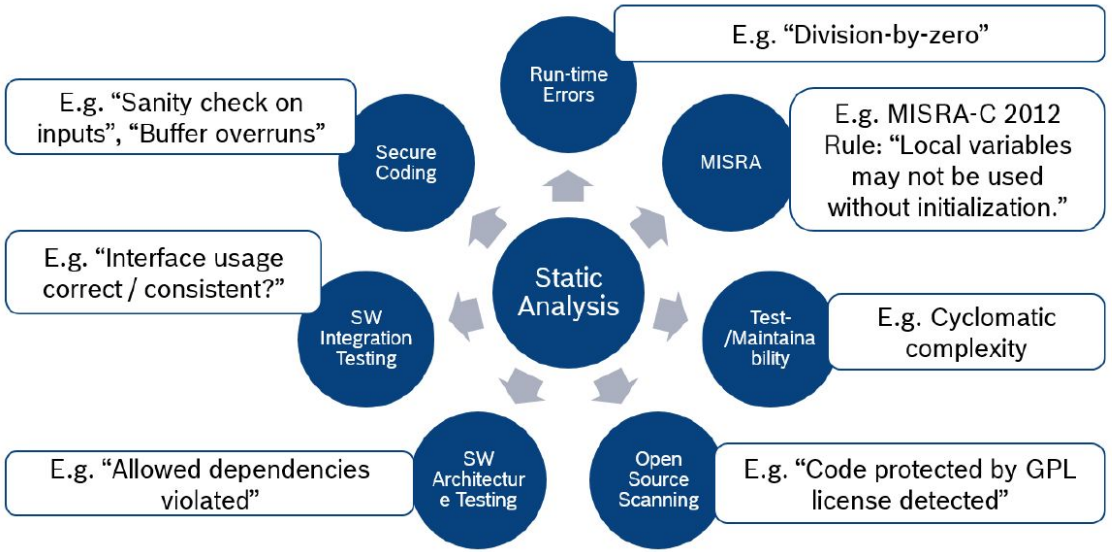
Static Code Analysis

Static code analysis refers to the analysis of the source code against a set of coding rules without the execution of the software.
This operation is performed by a static analysis tool(Helix C/C++, Coverity, Check Marx) .



Typical use cases: Secure coding, Run time errors, MISRA warning, Maintainability, open source scanning…

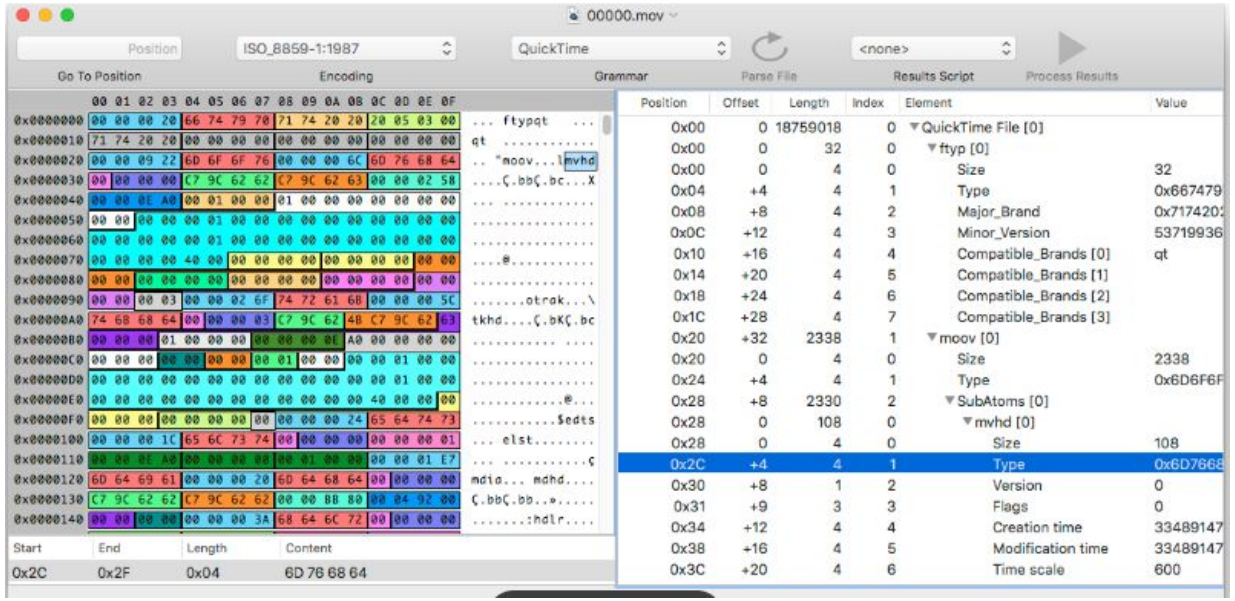
Test Goal:
The goal of this test method is to identify the weakness in the source , that might lead to a Vulnerability. Example Vulnerabilities discovered by static code analysis:
Buffer overflow, unchecked usage of inputs, Dangling pointers, usage of unsafe interface.



BGSW Curriculum

Binary Analysis

Binary analysis is the analysis of the executable binary file instead of analysis of the source code.



Typical use cases:

- Analysis of the 3rd party software that is not available as a source code.
- Vulnerability identification that are introduced during compilation and post implementation.

Test Goal:

The goal of this test method is identifying non-functional issues, Eg: memory problems and other generic vulnerabilities on binary code. Identification of 3rd party IP is also possible.

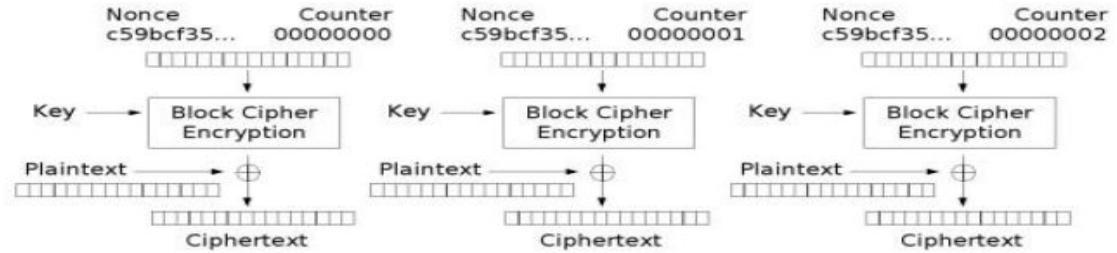
BOSCH

BGSW Curriculum
Functional security test

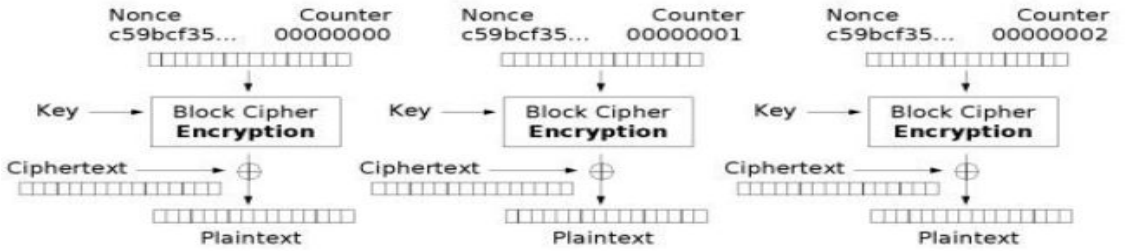
Functional security tests are conformance tests that check whether the actual implementation is in accordance with the functional security requirements, which are written and described in a system requirement specification.

Typical use cases: Encryption, Message Authentication, Secure access , Secure storage , Key Management, Verify user…

Test Goal:
The goal of this test method is to verify that functional security requirements have been implemented as expected.



Counter (CTR) mode encryption



Counter (CTR) mode decryption

BGSW Curriculum
Defensive coding

- **Defensive coding** is a form of defensive design intended to develop programs that can detect potential security abnormalities and make predetermined responses.
- Test cases are often designed, implemented and executed by developers as unit test i.e., dedicated test code is developed.

```
void report_overflow(void);

unsigned
add_unsigned(unsigned a, unsigned b)
{
  unsigned sum = a + b;
  if (sum < a) { // or sum < b
    report_overflow();
  }
  return sum;
}
```

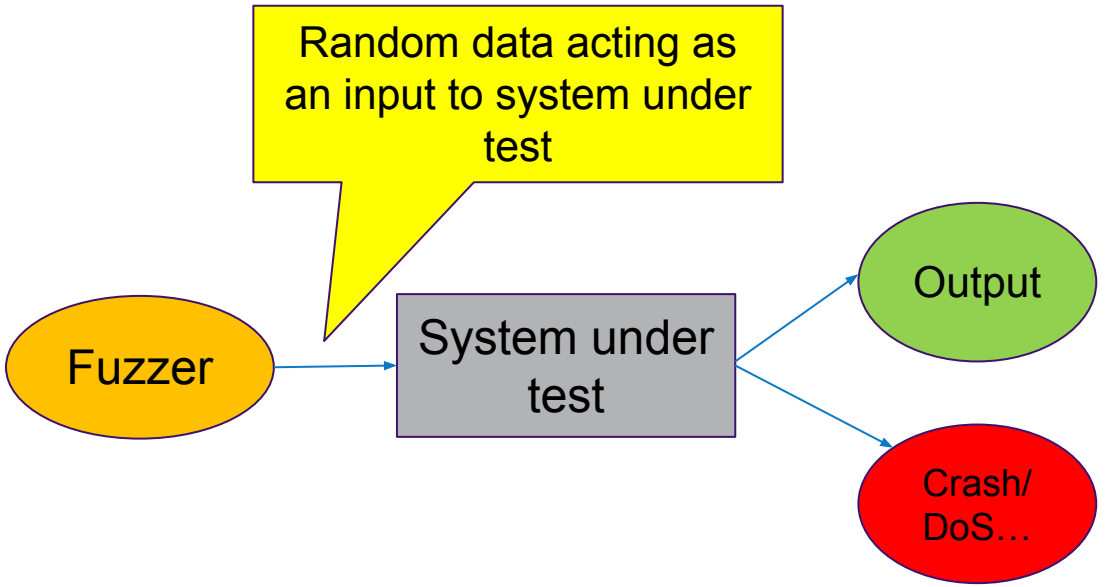
Typical use cases: exceptional conditions, boundary value checking and proper error handling behavior…

Test Goal:
Writing this type of tests helps improving the system's robustness.

BGSW Curriculum

Fuzzing

- Fuzzing is a dynamic software testing technique. Fuzzing automatically generates unexpected, malformed or random data and provides this data as input to a software under test. During execution, the software under test is monitored, e.g., for crashes or hangs.
- There are two types of Fuzzing
  Source code fuzzing: White box test.
  Protocol fuzzing: Black box test



Use cases: Functions, Network protocol, Parser of any kind, Formatted outputs …Fuzzing can detect bugs such as “Divide by zero, buffer overflow, infinite loops…”

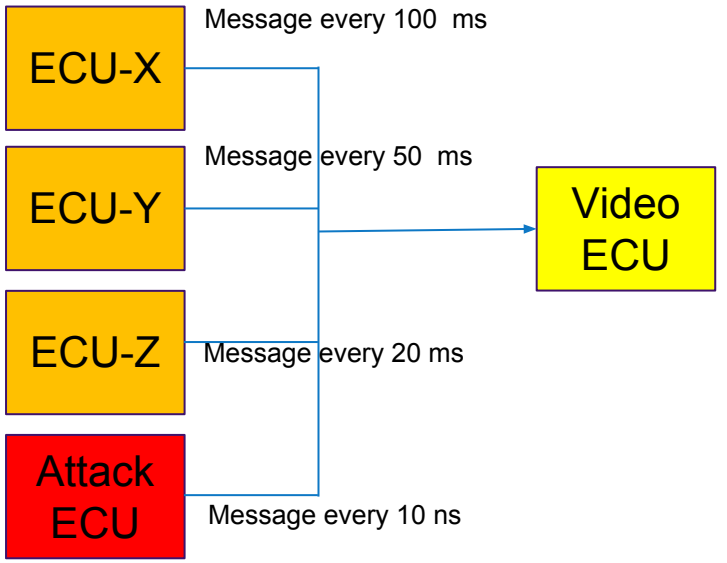
Test Goal:
The overall goal is to validate a robust program behavior. If a program accepts data from an untrusted input, the program should not display unwanted, observable behavior.

BGSW Curriculum
Flooding

- Flooding tests are a methodology where a mass of requests is used to exhaust system resources.
- These system resources can be Network bandwidth, CPU utilization, Memory utilization …

Use cases: Denial of service on communication bus, DoS due to increased processing time (More data items in NVM queue)…

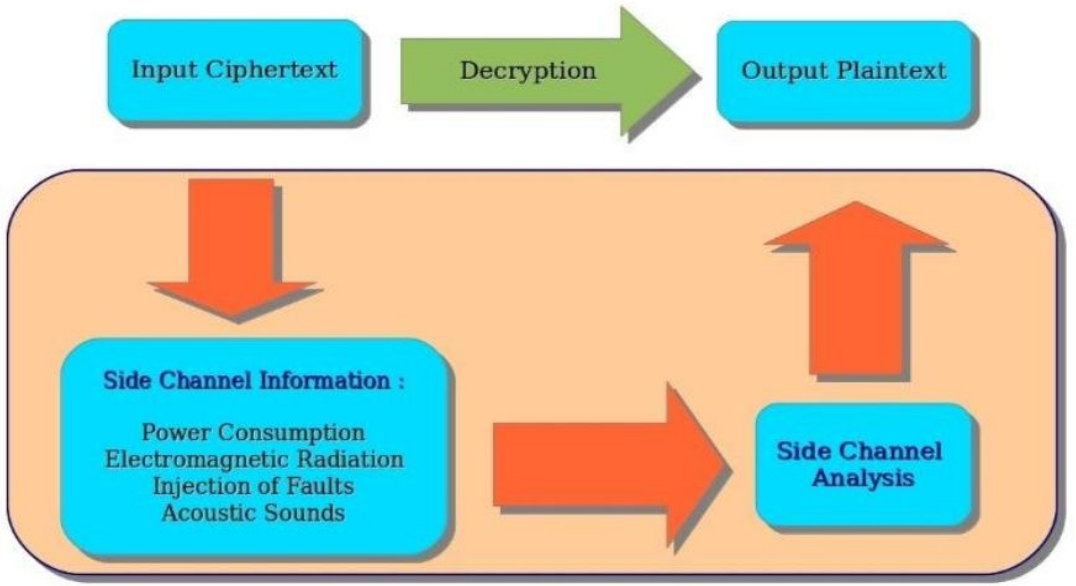
Test Goal:
The goal is to validate the robustness of the system, especially if the system can detect a flooding condition and to react accordingly, e.g., by rejecting new requests and therefore ensuring existing requests are still able to be served properly.



BGSW Curriculum

Side channel attack

- In security, a side-channel attack is **any attack based on information gained from the implementation of a system**, rather than weaknesses in the implemented algorithm itself. Eg: Power consumption during operations, timing monitoring …
- Some classes of side channel attacks are timing attacks, Power monitoring attacks, Cache attacks, Electromagnetic attacks…



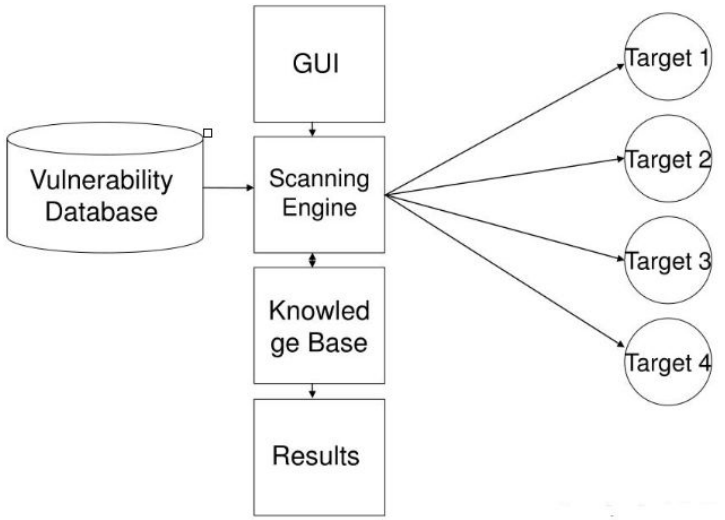
Typical use cases:
Simple power analysis, timing analysis, differential power analysis, simple fault analysis…

Test Goal:
Side-channel crypto-analysis typically has the aim to recover cryptographic keys (Private or symmetric keys) or a plaintext.

BGSW Curriculum

Vulnerability Scanning

- A vulnerability scanner is an automated vulnerability testing tool that monitors for misconfigurations or coding flaws that pose cybersecurity threats.
- Vulnerability scanners either rely on a database of known vulnerabilities or probe for common flaw types to discover unknown vulnerabilities.



Typical use cases:

- Vulnerability scanning can be integrated into the development process to catch vulnerabilities early.
- It is recommended that they run on production systems as well on a regular basis.

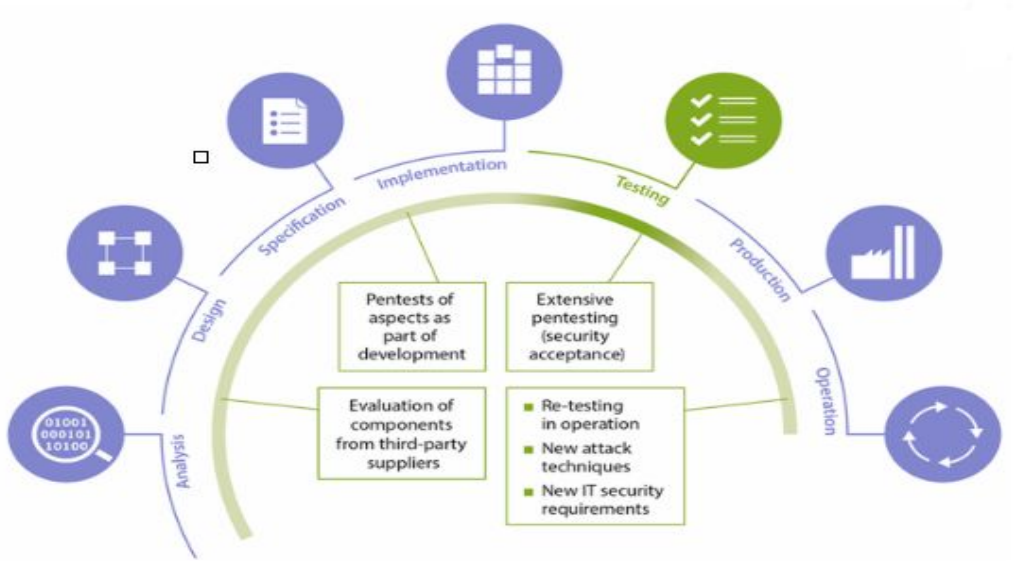
Test Goal:
The goal of a vulnerability scan is to quickly and automatically ensure the absence of well-known vulnerabilities in the software. Additionally, system configurations are scanned to ensure that the system is running with a secure configuration.

BGSW Curriculum
Penetration Test

- A penetration test (or pen test) is a practical security evaluation from an attacker's point of view. In other words, it can be called "authorized hacking".
- The test is performed to identify weaknesses (also referred to as vulnerabilities), including the potential for unauthorized parties to gain access to the system's features and data

Typical use cases:

- During the design phase, pen test of potential 3rd party components can be conducted to filter out insecure parts.
- During the prototype stage: i.e., once a feature complete prototype exists, a pen test can assess the security of the overall system.
- After the product is released.



Test Goal:
The goal of a pen test is to identify practically exploitable vulnerabilities in a target device. The result (deliverable) is a pen test report that describes the target device, the testing approach as well as identified findings and countermeasures.

BGSW Curriculum

Security Testing Methods Summary

Security Testing Methodology	Fault tolerance	Availability	Functional correctness	Confidentiality, Integrity, Non-repudiation, Accountability, Authenticity
Reviews			✔	
Formal Verification			✔	✔
Static Code Analysis	✔			
Binary Analysis	✔			
Functional Security Tests			✔	✔
Defensive Coding Tests	✔		✔	
Fuzzing Tests	✔			
Flooding	✔	✔		
Side Channel Attacks				✔
Vulnerability Scan			✔	✔
Penetration Test	✔	✔	✔	✔

QUESTIONS?