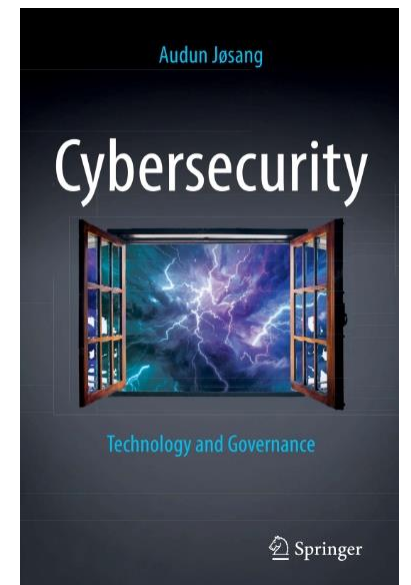


Chapter 11: Secure by design

Cybersecurity: Technology and Governance

Audun Jøsang,

1st Edition, Springer Nature, 2024.



Overview

- 7 steps of the Security Development Lifecycle (SDL)
- Threat Modeling and STRIDE
- Secure software development and DevOps
- Application security and OWASP Top 10
- Security in the cloud
- CISA Guide for Secure by Design

Secure by design and privacy by design

- Security and privacy by design means that information security and privacy are explicitly considered throughout the lifecycle of software and applications.
- An important goal is to find and reduce vulnerabilities early in the development process so that there are fewer incidents and vulnerabilities to deal with during operations.
- Microsoft saw this need early on and has been a leader here, defining the Security Development Lifecycle (SDL)
 - This is now an integral part of the software development process at Microsoft (and others)
- We can describe the life cycle as a process of 7 phases:



Phase 1: Training

Everyone who participates in the development and operation of digital services and applications must have basic knowledge and understanding of:

- Information security and privacy
- Risk assessments
- Threat modeling

This is really a matter of course. It would be irresponsible to educate building architects and engineers without giving them knowledge about fire safety, because architects and engineers would then build fire traps into our buildings. In the same way, it is irresponsible to educate computer scientists and computer engineers without mandatory courses on information security, because the graduates would then necessarily build a vulnerable ICT infrastructure.



Phase 2: Information security and privacy requirements

- Sources of information security and privacy requirements:
 1. Requirements resulting from good practices for adequate security in applications and business processes.
 2. Requirement to limit safety risk to an acceptable acceptable level.
 3. Legal, regulatory, and contractual requirements for information security and privacy.
- Security requirements must be constantly updated to reflect changes in required functionality, threat landscape, laws, regulations, regulations,...
- Ideally early in the development process (initial design and planning phase)
- OWASP Top 10/ASVS defines best practices for application security requirements



User Story and Use Case

User Story – Seen from the user's perspective:

As a [user], I want [feature] to reach [result].

Example: As a Flickr user, I want to be able to define access control, so I can choose who can see the different photos I've posted.



Use Case – Seen from the designer's perspective:

Description of a set of interactions between a system and one or more actors (where an "actor" can be a user or another system).



Attacker Story and "Misuse Case"

(Attacker's objective and threat scenario)

Attacker Story – Attacker's goal:

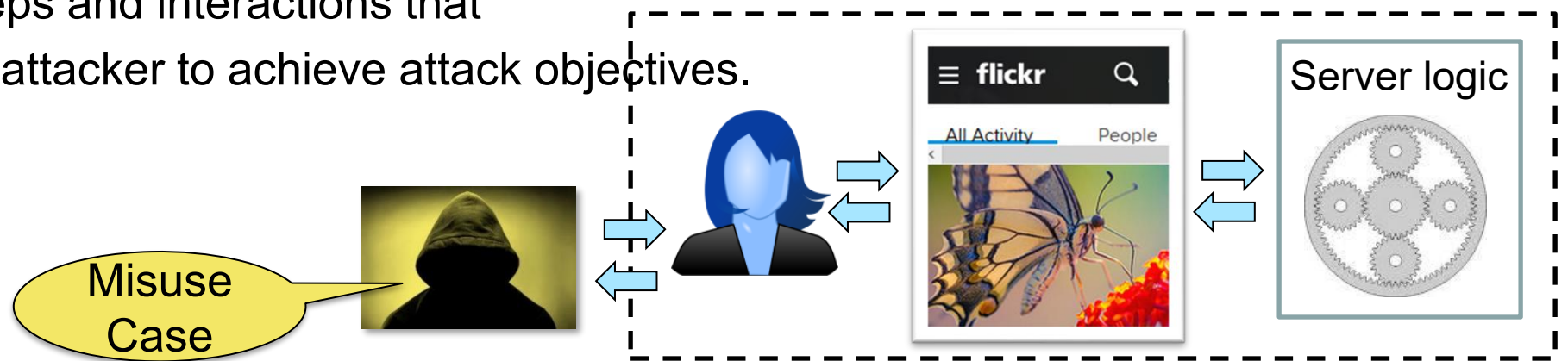
As [attacker], I want [function] to reach [result]. Example: As an attacker, I want to hack Flickr accounts to steal private photos and personal info.



Misuse Case (Threat scenario)

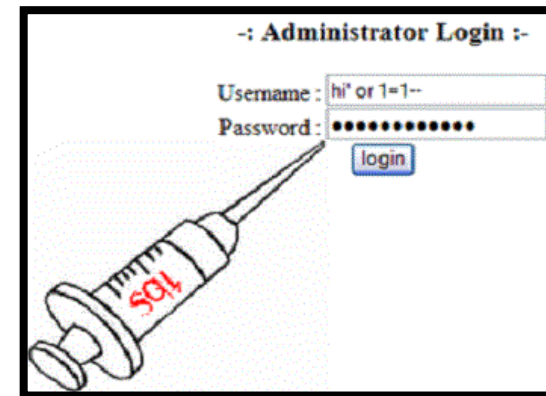
Seen from the threat actor's perspective:

Description of a set of steps and interactions that must be executed by the attacker to achieve attack objectives.



Functional and non-functional security

- Functional security:
 - Security requirements that make up a user story of their own
 - Security feature necessary for the user story
 - Security feature that provides a "visible" service to users
 - E.g.: User authentication
- Non-functional security:
 - Security requirements that are not a separate user story
 - Security feature that is unnecessary for the user story but necessary to prevent threats
 - Security feature that is "invisible" to the user
 - E.g.: Input validation against SQL injection and XSS



Phase 3: Secure Design, and Phase 4: Secure Coding

3. Secure design:

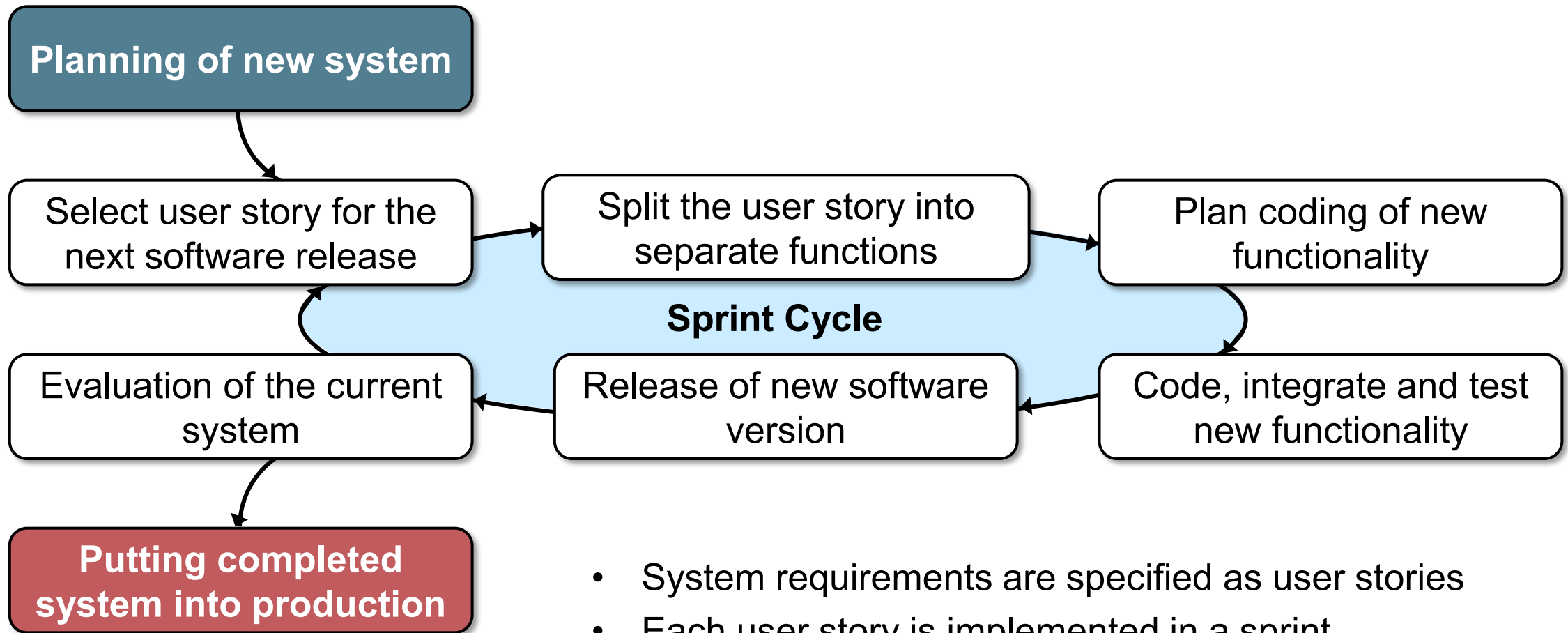
- Important part to specify "secure features", which are well designed with regard to security.
- The designer must have good expertise in controls such as crypto, authentication, logging etc.
- Threat modeling (more on this later) is an important part here to avoid vulnerabilities

4. Secure coding and implementation:

- The goal of secure coding is to prevent vulnerabilities from being built into the system during coding and to ensure that security features work according to requirements and design
- Modern developments rely heavily on third-party components, and it's important to understand the impact these can have on security. Unsafe components should not be used.
- Tools for code scanning and static analysis, reviewing code and using "secure" programming languages are important here. See guide:
- https://media.defense.gov/2025/Jun/23/2003742198/-1/-1/0/CSI_MEMORY_SAFE_LANGUAGES_REDUCING_VULNERABILITIES_IN_MODERN_SOFTWARE_DEVELOPMENT.PDF

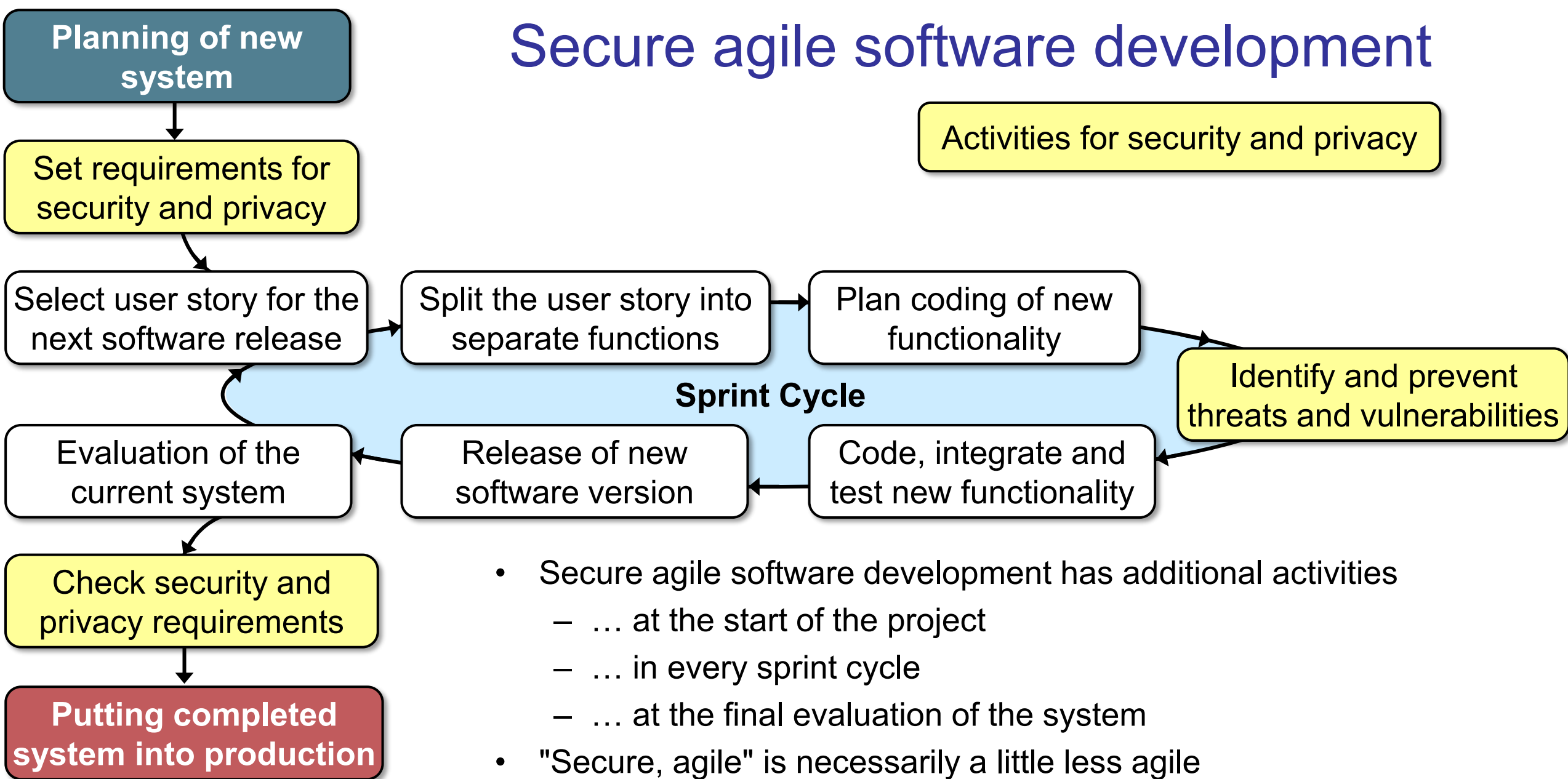


Agile software development



- System requirements are specified as user stories
- Each user story is implemented in a sprint
- Continues as long as there are user stories left
- The system is finished when all user stories have been created

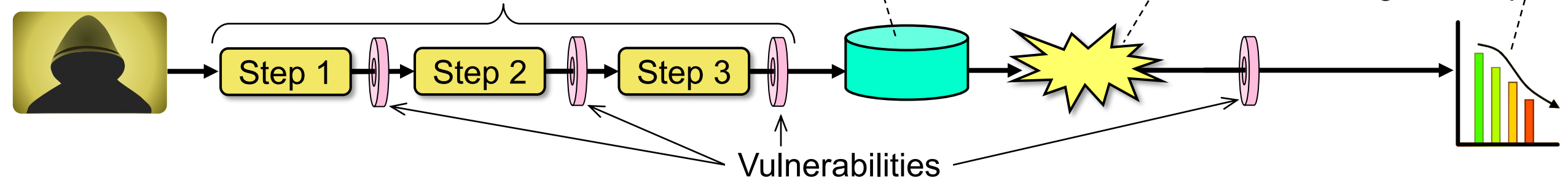
Secure agile software development



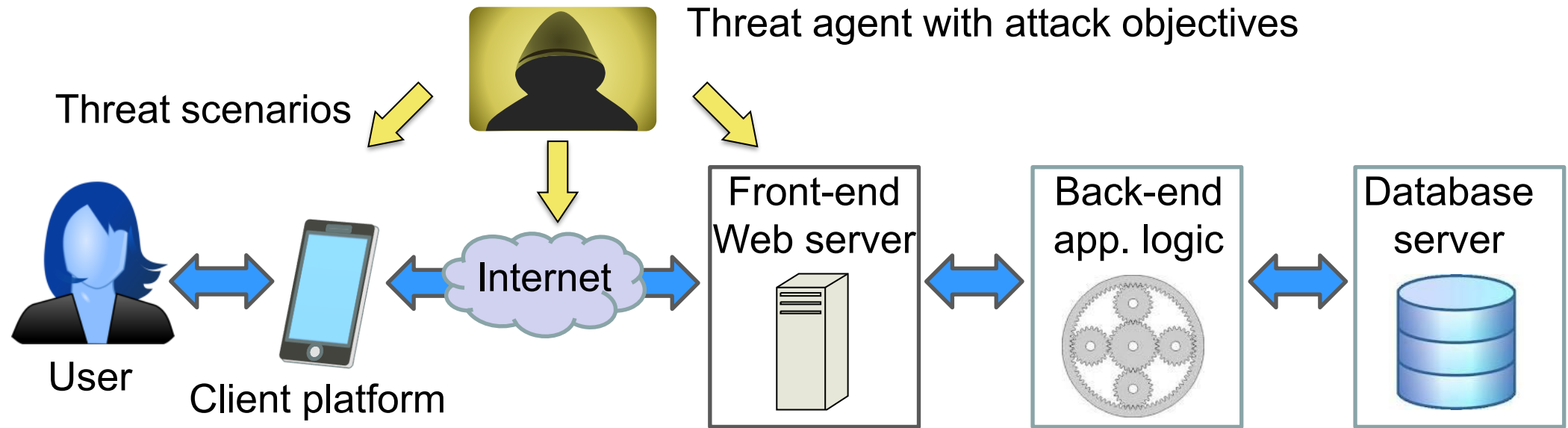
Assets, Threats, Vulnerabilities and Controls

- **Assets:** (Informational) resources that are of value to the organization.
 - Data, Systems, Applications, Networks, Devices, Services, People
 - The goal of information security is to protect the KIT of assets, depending on the need.
 - Personal Identifiable Information (PII)
- **Threat:** A potential attack scenario controlled by a threat actor that could harm your organization's assets
- **Vulnerability:** Lack of security controls against threats.
- **Security Control:** Method of preventing threats or mitigating consequences

Threat actor executes ... threat scenario ... which harms assets and causes incident ... with negative impact



Threat modeling



- Threat modeling is identifying, analyzing, and describing relevant attack scenarios.
- In secure, agile software development, threat modeling and a simple risk assessment should be performed as part of each sprint.
- Think: How can this new feature be misused or attacked? What assets can be damaged? What consequences could it have?
- Stop or mitigate the threat (remove vulnerabilities) during the sprint.

STRIDE Threat Modeling for Software Development

- There are several threat modeling frameworks.
- STRIDE is a popular framework developed by Microsoft, where the name comes from the initials of 6 categories
- The purpose is for these categories to help identify vulnerabilities

Letter	Threat category	Description	Security breach
S	<i>Spoofing</i> Identity theft	Can an attacker gain unauthorized access by stealing someone else's identity?	Breach of authenticity
T	<i>Tampering</i>	Can an attacker change the configuration or data processed by the system?	Breach of integrity
R	<i>Repudiation</i>	Can an attacker deny abuse because we lack traces and logs that point to the attacker?	Breach of availability
I	<i>Information disclosure</i>	Can an attacker gain access to confidential and personal data?	Breach of confidentiality
D	<i>Denial of service</i>	Can an attacker block or reduce the availability of the system?	Breach of availability
E	<i>Elevation of privilege</i>	Can an attacker gain elevated access and thus become a privileged user?	Breach of access control

- Open Web Application Security Project (OWASP)
 - Non-profit organization with the goal of improving the security of applications and services on the internet
 - Through advice, guidance and tools
 - Involving businesses, educational institutions, and individuals from all over the world
 - Several parallel projects
- OWASP ASVS (Application Security Verification Standard)
 - Standard with the goal of defining best practices for safe development and testing
- OWASP Top 10
 - Ranks the 10 most critical security risks for web applications
 - Provides advice on how to reduce vulnerability and risk
 - Updated every few years
 - The last edition came out in 2021, new edition is expected in 2025
 - Represents a very good source for getting started with threat modeling

1. Broken Access Control

- Attackers exploit flaws in how access control is enforced
- For example, reading or modifying other users' data

2. Cryptographic Failures

- Errors related to crypto that often result in sensitive data exposure or system compromise

3. Injection

- Manipulated input data is sent to an application as part of a request/command that tricks the application into performing accidental or unauthorized actions
- Multiple variants, including SQL injection
- Cross-site scripting (XSS) is (in the 2021 version) part of this category

4. Insecure Design

- Risks related to errors in the design

5. Security Misconfiguration

- Risk due to errors in configuration
- E.g. insecure default configuration, error messages that reveal sensitive information

6. Vulnerable and Outdated Components

- Exploiting vulnerable/outdated component
- E.g. through the use of an (external) library or software module that has more privileges as the application where they are used

7. Identification and Authentication Failures

- Checking the user's identity, authentication or managing the session is often implemented incorrectly
- E.g. bad and default passwords, password reset, session identifier in URL, ...

8. Software and Data Integrity Failures

- E.g. application uses module from unknown source or auto-update without good enough check of integrity

9. Security Logging and Monitoring Failures

- Means that it is not possible to detect and handle fractures

10. Server Side Request Forgery (SSRF)

- Attacker causes server application to make request to a domain specified by attacker
- For example, it can mean that the attacker can read/update internal resources.

Phase 5: Security Testing

- The goal is to uncover vulnerabilities that have not been discovered in the design or code phases.
- **Dynamic testing/vulnerability analysis** of the complete software checks functionality that becomes visible when all the components are integrated together. Checks, among other things, that the user gets access to information/functionality they should (and not information that the user should not have access to).
- **Penetration testing** goes a step further and is an (authorized) simulated attack to evaluate security ("ethical hacking" is often used for this). Distinguish between:
 - White box testing where the attacker has information about the system in advance
 - Black box testing where the attacker does not have such information.
- **Fuzztesting** attempts to provoke errors in the system by providing corrupt input values (random or malformed data).



Fase 6: Produksjonssetting og Fase 7: Forvaltning

6. Production setting

- **Plan for operation, maintenance and incident management** must define procedures for operation (incl. patching), deviation reporting and incident management.
- **Formal approval of production setting** will require verification and documentation that all security and privacy requirements have been met and identified vulnerabilities have been adequately removed. Formal responsibilities/mandates must be defined and relevant data and documentation archived.

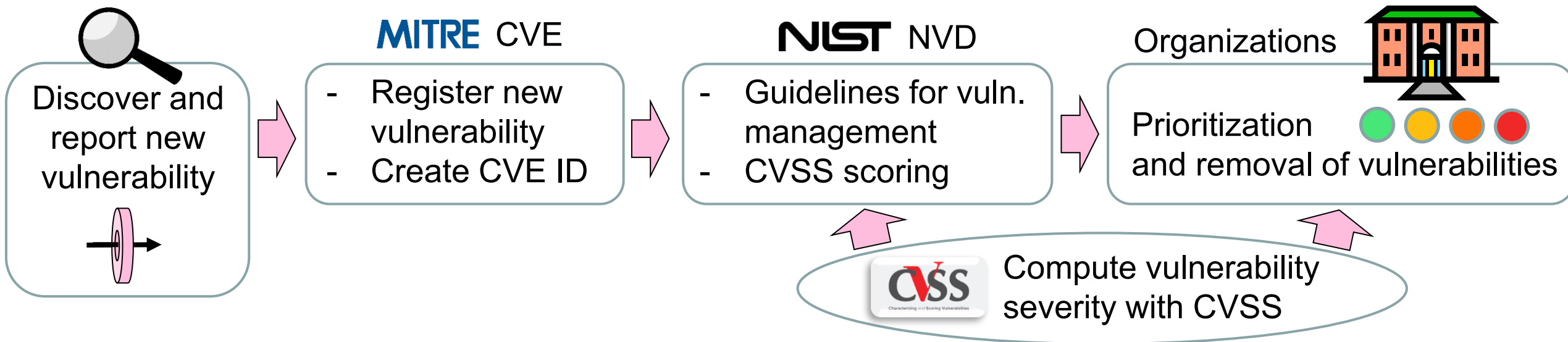
7. Operation

- **Operation and maintenance** means that procedures and routines for the operation and maintenance of software must be followed, also over time. Audits should be carried out regularly and an information security management system should be in place. You must clearly define what is to be logged and how the logs are handled.
- **Deviation and incident management-** Nonconformities and incidents must be reported as described in the plans.



Removal of vulnerabilities in production software

- The vulnerability management lifecycle involves several steps, from the time a vulnerability is discovered and reported to when it is removed in enterprise software instances.

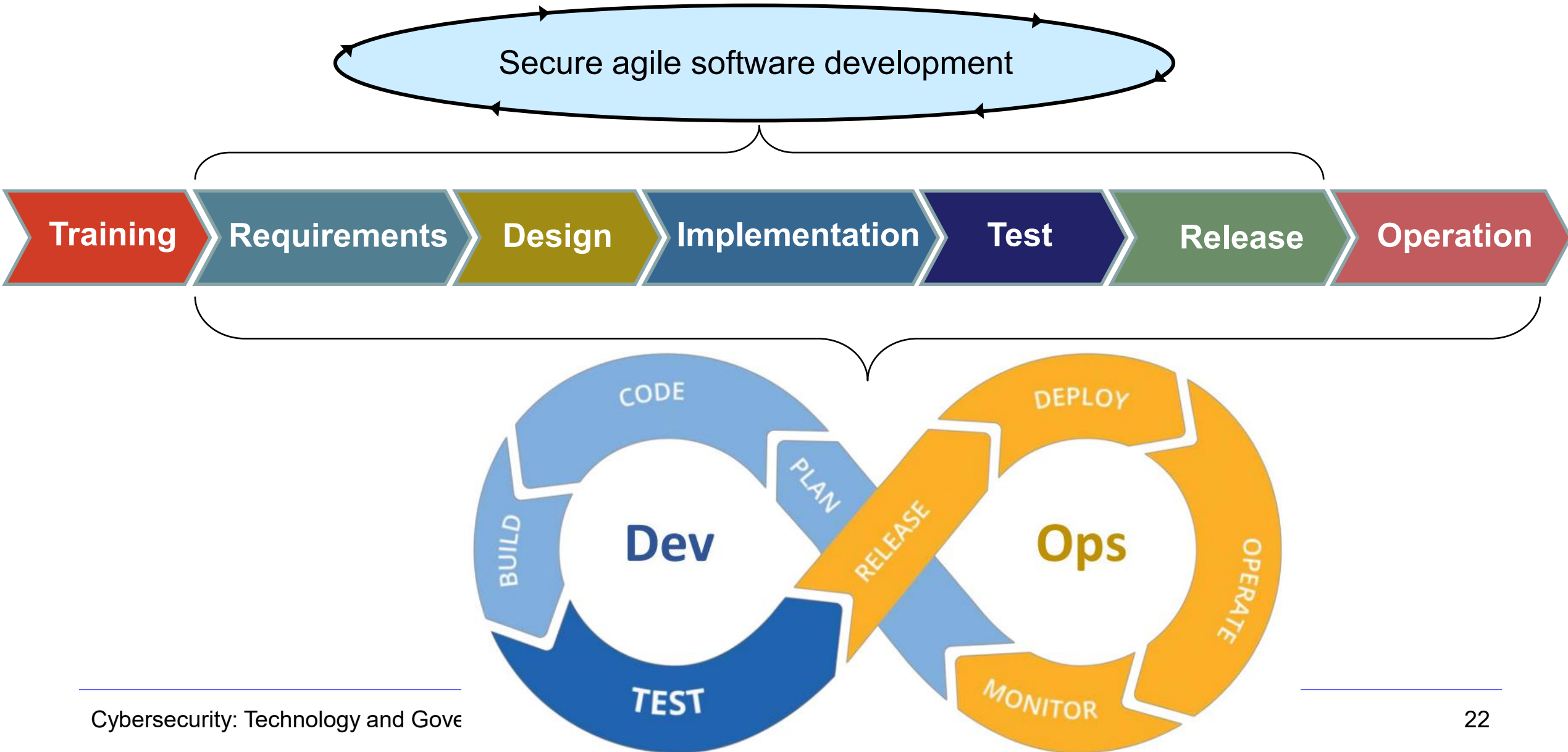


CVE and CVSS

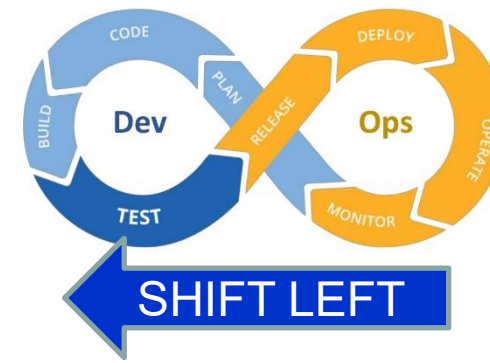
- CVE (Common Vulnerabilities and Exposures) is a database of vulnerabilities in all types of software.
- The NVD (National Vulnerability Database) imports data from CVE, adding guidelines for handling and removing the vulnerability in the software in question, typically with input from the software manufacturer.
- The severity is calculated using the Common Vulnerability Scoring System (CVSS), based on various factors, providing severity scoring on a scale of 0 to 10.
 - Critical: 9.0 – 10
 - High: 7.0 – 8.9
 - Medium: 4.0 – 6.9
 - Low: 0.1 – 3.9



DevOps: Secure agile software development in the cloud

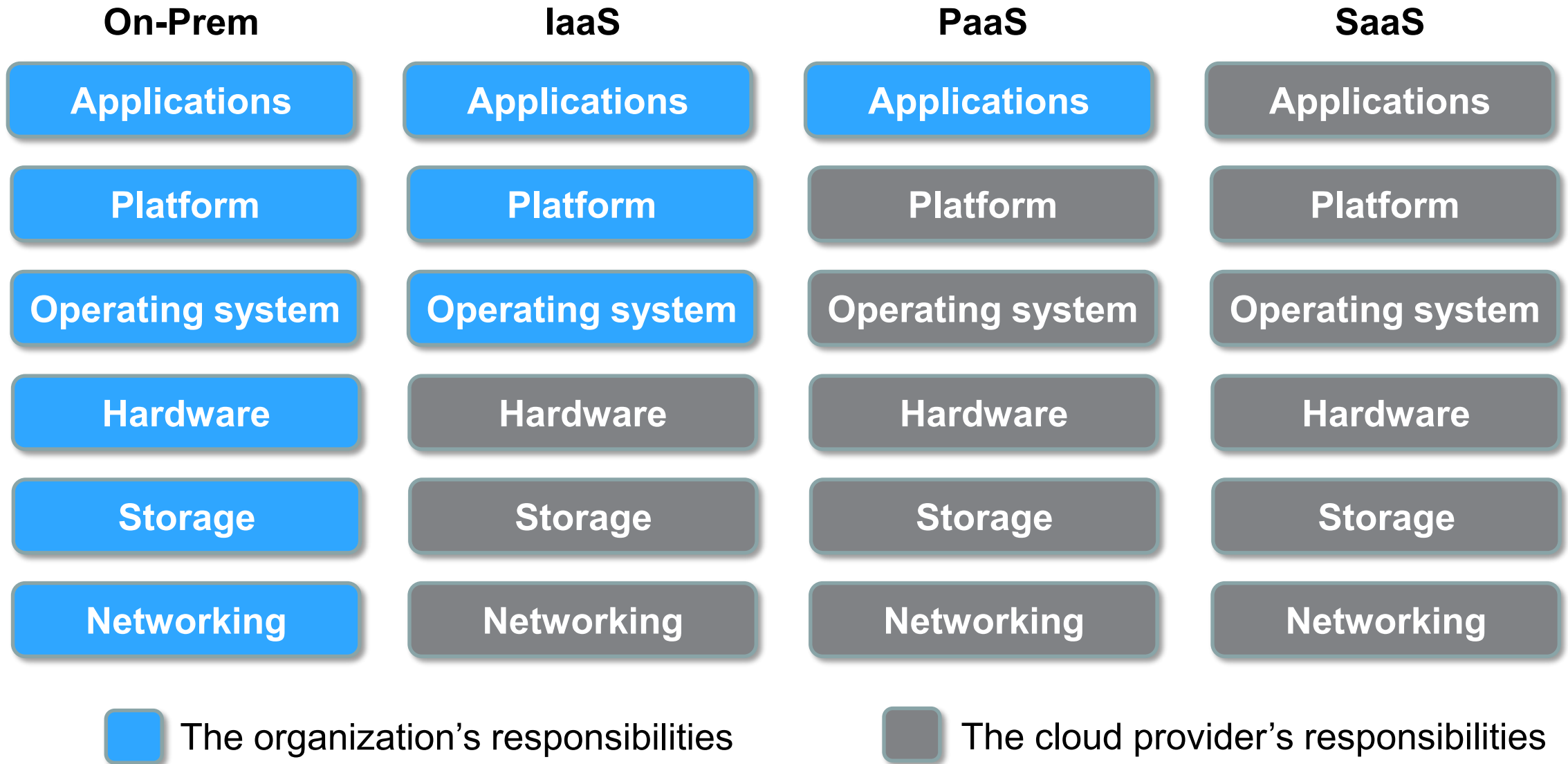


DevSecOps of SHIFT LEFT



- DevOps, also called DevSecOps, is, as the name indicates, combining software development (Development) and operations (Operations).
- DevOps aims to shorten the cycle of software development and operation and provide continuous delivery with high quality and security.
- DevOps evolved from secure agile, by including operations/management, while secure agile only goes through to production deployment.
- All tools needed throughout the DevOps cycle reside in the cloud, from development tools and test tools to tools for production, operations, and monitoring. Cloud models for DevOps must be either IaaS or PaaS.
- The term "SHIFT LEFT" means that there is more focus on security on the left side of the 8, so there are fewer security vulnerabilities and incidents during operation.

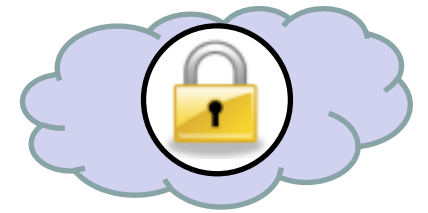
Shared responsibility for different cloud models



Shared responsibility for security in different cloud models

Responsibility	On-Prem	IaaS	PaaS	SaaS
Data classification and management				
Client and endpoint security				
Identity and access management				
Application security				
Network security				
System security				
Physical security				
Protection against corruption and insider threats				
Protection against (legal) access by foreign states				
 The organization's responsibility  The cloud provider's responsibility				

Security in the cloud



- Cloud providers often have considerable expertise and resources to maintain a high degree of security in their infrastructures, e.g. through rapid security updates, effective security monitoring and detection, and incident response.
- Aspects that can threaten security when using cloud solutions are, for example.:
 - Identity and access management using cloud services can pose a vulnerability if poorly implemented. Cloud services are technically accessible from across the internet, so robust access protection solutions are needed. «Identity is the new security perimeter»
 - Potential corruption or unfaithful servants at the cloud provider. The probability of this is usually very low, but may increase in countries where the level of corruption is relatively significant and the rule of law is relatively low.
 - Legal access to your company's data by governments in countries where the cloud provider is located. This can take place as part of intelligence or investigation of crime, without the company necessarily being informed.



SHIFTING THE BALANCE OF CYBERSECURITY RISK:

PRINCIPLES AND APPROACHES FOR
SECURE BY DESIGN SOFTWARE



CISA Guide for Secure by Design

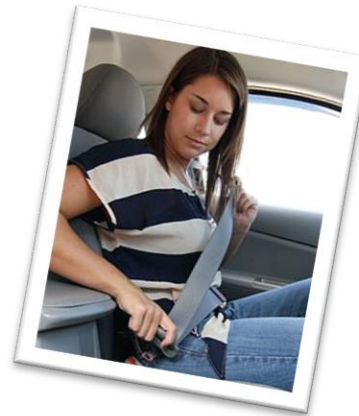


- CISA (Cybersecurity and Infrastructure Security Agency) is part of the United States Department of Homeland Security (DHS).
- CISA advises and coordinates cybersecurity for all levels of government in the US.
- The guide Secure by Design was published in 2023: <https://www.cisa.gov/securebydesign>
 - https://www.cisa.gov/sites/default/files/2023-10/SecureByDesign_1025_508c.pdf
- Concepts:
 - Secure by design (Built-in information security)
 - Secure by default
- Three basic principles:
 1. Developer must focus on software security during operations at customers
 2. Developer must show transparency and accountability around security in software development
 3. Management must set built-in information security as a strategic goal

Secure by Design and by Default

- **Secure by design** means that IT products are built in a way that reasonably prevents threat scenarios to devices, software, and IT systems. Software manufacturers should perform a risk assessment to identify and map relevant cyber threats to critical systems, and then include measures in the product design that provide proper protection against the identified threats.
- **Secure by default** means that products are configured to be resilient to widespread attack techniques "out of the box" at no additional cost to customers. These products protect against the most relevant threats and eliminate simple vulnerabilities without customers and end-users having to take additional steps to secure them. Products built with secure default settings are intended to make customers aware of possible deviations from secure settings, so that they are encouraged to tighten security settings. Secure default setting is a form of built-in information security. Secure default setting should be part of basic security in enterprises.
 - Analogy: Seat belts in cars are standard. Warning sound and lights when you forget them.

DESIGN



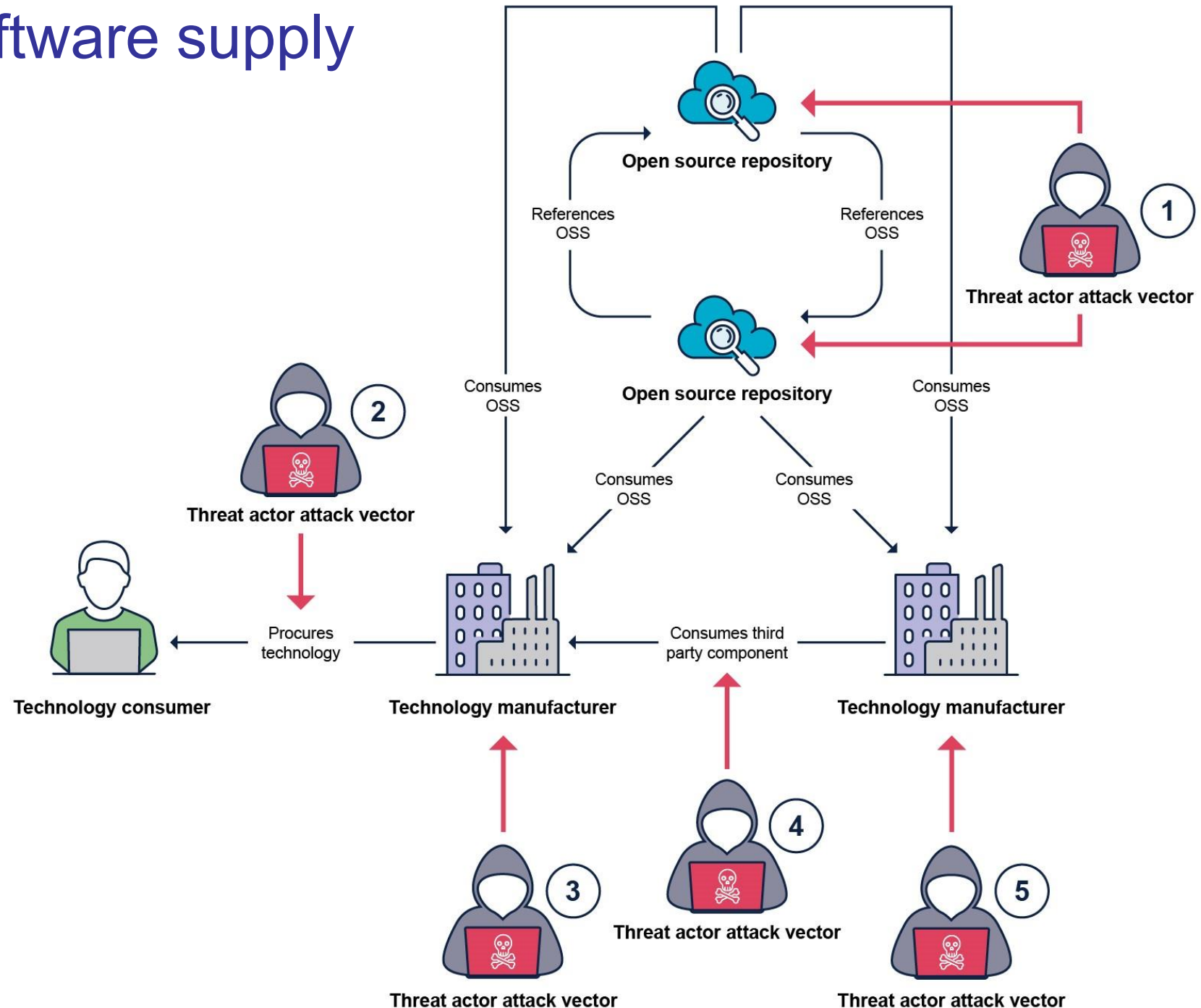
Choosing secure and verifiable technologies



- ACSC (Australian Cyber Security Centre) cooperates with CISA in the US
- The guide Choosing secure and verifiable technologies was published in 2024:
 - <https://www.cyber.gov.au/sites/default/files/2024-05/choosing-secure-and-verifiable-technologies.pdf>
 - <https://www.cyber.gov.au/resources-business-and-government/maintaining-devices-and-systems/outsourcing-and-procurement/cyber-supply-chains/choosing-secure-and-verifiable-technologies>
- Describes attack vectors against supply chains in software development
- Provides advice on security in the use of open source software FAQ: Open Source Software)

Attack vectors in software supply chains

- Open-source software (OSS) is particularly vulnerable to attack.
- There is basically little guarantee of quality and safety in the OSS.
- Challenging for developers to check security in the OS.
- Lack of functional quality is directly visible, while lack of safety is invisible.



SBOM: Software Bill of Materials

- SBOM (Software Bill of Materials) means that "digital bills of materials" are created for all components that are part of systems or software applications.
- Includes all software libraries and external components that are linked in.
- Makes it possible to map all known vulnerabilities in a system or application.
- The SBOM initiative started in 2018, and has gradually become an important part of software security and risk management for the supply chain.
- SBOM is a collaboration between various actors, governed by CISA (Cybersecurity and Infrastructure Security Agency) in the United States.



End of presentation

