Threagile
Agile Threat Modeling

Threat Model Report
Sample Application

27 September 2023

Ciro Bologna

Table of Contents

Management Summary

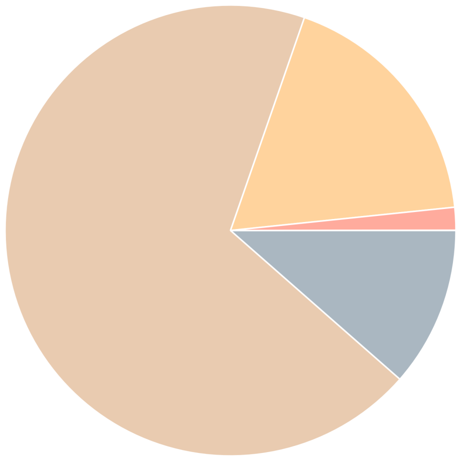
Threagile toolkit was used to model the architecture of "Sample Application" and derive risks by analyzing the components and data flows. The risks identified during this analysis are shown in the following chapters. Identified risks during threat modeling do not necessarily mean that the vulnerability associated with this risk actually exists: it is more to be seen as a list of potential risks and threats, which should be individually reviewed and reduced by removing false positives. For the remaining risks it should be checked in the design and implementation of "Sample Application" whether the mitigation advices have been applied or not.

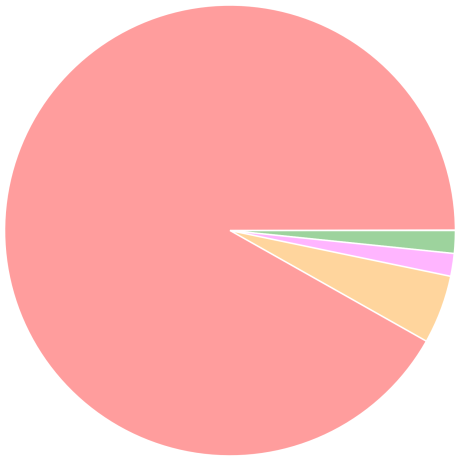
Each risk finding references a chapter of the OWASP ASVS (Application Security Verification Standard) audit checklist. The OWASP ASVS checklist should be considered as an inspiration by architects and developers to further harden the application in a Defense-in-Depth approach. Additionally, for each risk finding a link towards a matching OWASP Cheat Sheet or similar with technical details about how to implement a mitigation is given.

In total **61 initial risks** in **22 categories** have been identified during the threat modeling process:

1	**critical risk**
0	**high risk**
11	**elevated risk**
42	**medium risk**
7	**low risk**

56	**unchecked**
3	**in discussion**
1	**accepted**
0	**in progress**
1	***mitigated***
0	***false positive***





Threat modeling should be part of SDLC

Impact Analysis of 61 Initial Risks in 22 Categories

The most prevalent impacts of the **61 initial risks** (distributed over **22 risk categories**) are (taking the severity ratings into account and using the highest for each category):
Risk finding paragraphs are clickable and link to the corresponding chapter.

Critical: **Some Individual Risk Example**: 1 Initial Risk - Exploitation likelihood is *Likely* with *Medium* impact.
Some text describing the impact...

Elevated: **Missing Authentication**: 1 Initial Risk - Exploitation likelihood is *Likely* with *Medium* impact.
If this risk is unmitigated, attackers might be able to access or modify sensitive data in an unauthenticated way.

Elevated: **Missing Cloud Hardening**: 1 Initial Risk - Exploitation likelihood is *Unlikely* with *Very High* impact.
If this risk is unmitigated, attackers might access cloud components in an unintended way.

Elevated: **Missing Hardening**: 3 Initial Risks - Exploitation likelihood is *Likely* with *Medium* impact.
If this risk remains unmitigated, attackers might be able to easier attack high-value targets.

Elevated: **SQL/NoSQL-Injection**: 1 Initial Risk - Exploitation likelihood is *Very Likely* with *Medium* impact.
If this risk is unmitigated, attackers might be able to modify SQL/NoSQL queries to steal and modify data and eventually further escalate towards a deeper system penetration via code executions.

Elevated: **Unguarded Access From Internet**: 10 Initial Risks - Exploitation likelihood is *Very Likely* with *Medium* impact.
If this risk is unmitigated, attackers might be able to directly attack sensitive systems without any hardening components in-between due to them being directly exposed on the internet.

Elevated: **Untrusted Deserialization**: 1 Initial Risk - Exploitation likelihood is *Likely* with *High* impact.
If this risk is unmitigated, attackers might be able to execute code on target systems by exploiting untrusted deserialization endpoints.

Medium: **Accidental Secret Leak**: 3 Initial Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk is unmitigated, attackers which have access to affected sourcecode repositories or artifact registries might find secrets accidentally checked-in.

Medium: **Code Backdooring**: 4 Initial Risks - Exploitation likelihood is *Unlikely* with *High* impact.
If this risk remains unmitigated, attackers might be able to execute code on and completely takeover production environments.

Medium: **Container Base Image Backdooring**: 3 Initial Risks - Exploitation likelihood is *Unlikely* with *High* impact.
If this risk is unmitigated, attackers might be able to deeply persist in the target system by executing code in deployed containers.

Medium: **Container Platform Escape**: 1 Initial Risk - Exploitation likelihood is *Unlikely* with *High* impact.
If this risk is unmitigated, attackers which have successfully compromised a container (via other vulnerabilities) might be able to deeply persist in the target system by executing code in many deployed containers and the container platform itself.

Medium: **Missing Identity Store**: 1 Initial Risk - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk is unmitigated, attackers might be able to exploit risks unseen in this threat model in the identity provider/store that is currently missing in the model.

Medium: **Missing Network Segmentation**: 3 Initial Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk is unmitigated, attackers successfully attacking other components of the system might have an easy path towards more valuable targets, as they are not separated by network segmentation.

Medium: **Missing Two-Factor Authentication (2FA)**: 6 Initial Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk is unmitigated, attackers might be able to access or modify highly sensitive data without strong authentication.

Medium: **Missing Vault Isolation**: 1 Initial Risk - Exploitation likelihood is *Unlikely* with *High* impact.
If this risk is unmitigated, attackers successfully attacking other components of the system might have an easy path towards highly sensitive vault assets and their datastores, as they are not separated by network segmentation.

Medium: **Missing Web Application Firewall (WAF)**: 1 Initial Risk - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk is unmitigated, attackers might be able to apply standard attack pattern tests at great speed without any filtering.

Medium: **Server-Side Request Forgery (SSRF)**: 5 Initial Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk is unmitigated, attackers might be able to access sensitive services or files of network-reachable components by modifying outgoing calls of affected components.

Medium: **Unchecked Deployment**: 5 Initial Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk remains unmitigated, vulnerabilities in custom-developed software or their dependencies might not be identified during continuous deployment cycles.

Medium: **Unencrypted Communication**: 1 Initial Risk - Exploitation likelihood is *Unlikely* with *High* impact.
If this risk is unmitigated, network attackers might be able to to eavesdrop on unencrypted sensitive data sent between components.

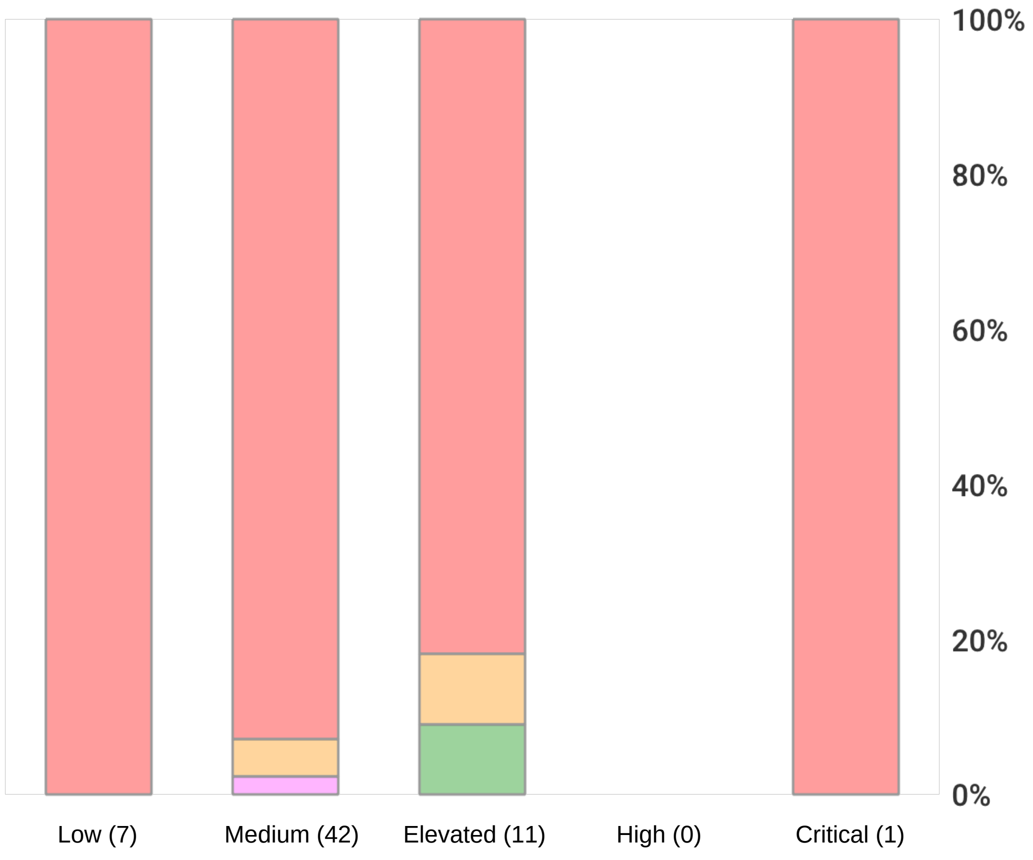
Medium: **Unencrypted Technical Assets**: 7 Initial Risks - Exploitation likelihood is *Unlikely* with *High* impact.
If this risk is unmitigated, attackers might be able to access unencrypted data when successfully compromising sensitive components.

Low: **DoS-risky Access Across Trust-Boundary**: 1 Initial Risk - Exploitation likelihood is *Unlikely* with *Low* impact.
If this risk remains unmitigated, attackers might be able to disturb the availability of important parts of the system.

Low: **Mixed Targets on Shared Runtime**: 1 Initial Risk - Exploitation likelihood is *Unlikely* with *Low* impact.
If this risk is unmitigated, attackers successfully attacking other components of the system might have an easy path towards more valuable targets, as they are running on the same shared runtime.

Risk Mitigation

The following chart gives a high-level overview of the risk tracking status (including mitigated risks):

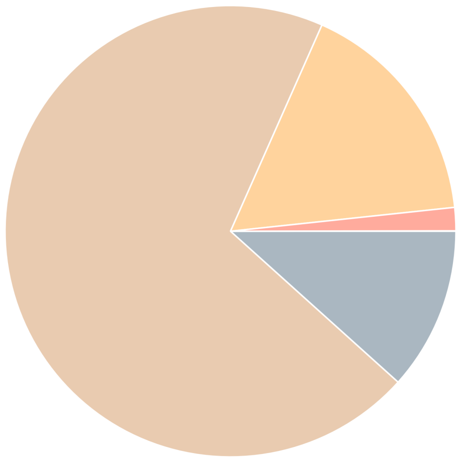


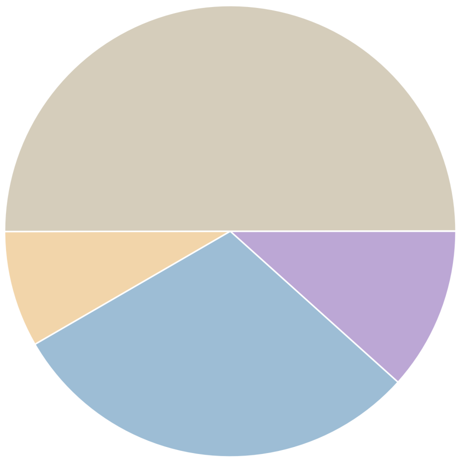
56 unchecked
3 in discussion
1 accepted
0 in progress
1 *mitigated*
0 *false positive*

After removal of risks with status *mitigated* and *false positive* the following **60 remain unmitigated**:

1 unmitigated critical risk
0 unmitigated high risk
10 unmitigated elevated risk
42 unmitigated medium risk
7 unmitigated low risk

7 business side related
18 architecture related
5 development related
30 operations related





Impact Analysis of 60 Remaining Risks in 21 Categories

The most prevalent impacts of the **60 remaining risks** (distributed over **21 risk categories**) are (taking the severity ratings into account and using the highest for each category):
Risk finding paragraphs are clickable and link to the corresponding chapter.

Critical: **Some Individual Risk Example**: 1 Remaining Risk - Exploitation likelihood is *Likely* with *Medium* impact.
Some text describing the impact...

Elevated: **Missing Authentication**: 1 Remaining Risk - Exploitation likelihood is *Likely* with *Medium* impact.
If this risk is unmitigated, attackers might be able to access or modify sensitive data in an unauthenticated way.

Elevated: **Missing Cloud Hardening**: 1 Remaining Risk - Exploitation likelihood is *Unlikely* with *Very High* impact.
If this risk is unmitigated, attackers might access cloud components in an unintended way.

Elevated: **Missing Hardening**: 3 Remaining Risks - Exploitation likelihood is *Likely* with *Medium* impact.
If this risk remains unmitigated, attackers might be able to easier attack high-value targets.

Elevated: **Unguarded Access From Internet**: 10 Remaining Risks - Exploitation likelihood is *Very Likely* with *Medium* impact.
If this risk is unmitigated, attackers might be able to directly attack sensitive systems without any hardening components in-between due to them being directly exposed on the internet.

Elevated: **Untrusted Deserialization**: 1 Remaining Risk - Exploitation likelihood is *Likely* with *High* impact.
If this risk is unmitigated, attackers might be able to execute code on target systems by exploiting untrusted deserialization endpoints.

Medium: **Accidental Secret Leak**: 3 Remaining Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk is unmitigated, attackers which have access to affected sourcecode repositories or artifact registries might find secrets accidentally checked-in.

Medium: **Code Backdooring**: 4 Remaining Risks - Exploitation likelihood is *Unlikely* with *High* impact.
If this risk remains unmitigated, attackers might be able to execute code on and completely takeover production environments.

Medium: **Container Base Image Backdooring**: 3 Remaining Risks - Exploitation likelihood is *Unlikely* with *High* impact.
If this risk is unmitigated, attackers might be able to deeply persist in the target system by executing code in deployed containers.

Medium: **Container Platform Escape**: 1 Remaining Risk - Exploitation likelihood is *Unlikely* with *High* impact.
If this risk is unmitigated, attackers which have successfully compromised a container (via other vulnerabilities) might be able to deeply persist in the target system by executing code in many deployed containers and the container platform itself.

Medium: **Missing Identity Store**: 1 Remaining Risk - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk is unmitigated, attackers might be able to exploit risks unseen in this threat model in the identity provider/store that is currently missing in the model.

Medium: **Missing Network Segmentation**: 3 Remaining Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk is unmitigated, attackers successfully attacking other components of the system might have an easy path towards more valuable targets, as they are not separated by network segmentation.

Medium: **Missing Two-Factor Authentication (2FA)**: 6 Remaining Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk is unmitigated, attackers might be able to access or modify highly sensitive data without strong authentication.

Medium: **Missing Vault Isolation**: 1 Remaining Risk - Exploitation likelihood is *Unlikely* with *High* impact.
If this risk is unmitigated, attackers successfully attacking other components of the system might have an easy path towards highly sensitive vault assets and their datastores, as they are not separated by network segmentation.

Medium: **Missing Web Application Firewall (WAF)**: 1 Remaining Risk - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk is unmitigated, attackers might be able to apply standard attack pattern tests at great speed without any filtering.

Medium: **Server-Side Request Forgery (SSRF)**: 5 Remaining Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk is unmitigated, attackers might be able to access sensitive services or files of network-reachable components by modifying outgoing calls of affected components.

Medium: **Unchecked Deployment**: 5 Remaining Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
If this risk remains unmitigated, vulnerabilities in custom-developed software or their dependencies might not be identified during continuous deployment cycles.

Medium: **Unencrypted Communication**: 1 Remaining Risk - Exploitation likelihood is *Unlikely* with *High* impact.
If this risk is unmitigated, network attackers might be able to to eavesdrop on unencrypted sensitive data sent between components.

Medium: **Unencrypted Technical Assets**: 7 Remaining Risks - Exploitation likelihood is *Unlikely* with *High* impact.
If this risk is unmitigated, attackers might be able to access unencrypted data when successfully compromising sensitive components.

Low: **DoS-risky Access Across Trust-Boundary**: 1 Remaining Risk - Exploitation likelihood is *Unlikely* with *Low* impact.
If this risk remains unmitigated, attackers might be able to disturb the availability of important parts of the system.

Low: **Mixed Targets on Shared Runtime**: 1 Remaining Risk - Exploitation likelihood is *Unlikely* with *Low* impact.
If this risk is unmitigated, attackers successfully attacking other components of the system might have an easy path towards more valuable targets, as they are running on the same shared runtime.

Application Overview

Business Criticality

The overall business criticality of "Sample Application" was rated as:

(archive | operational | **IMPORTANT** | critical | mission-critical)

Business Overview

This sample app can be used to send and store greetings to your friendly security architect

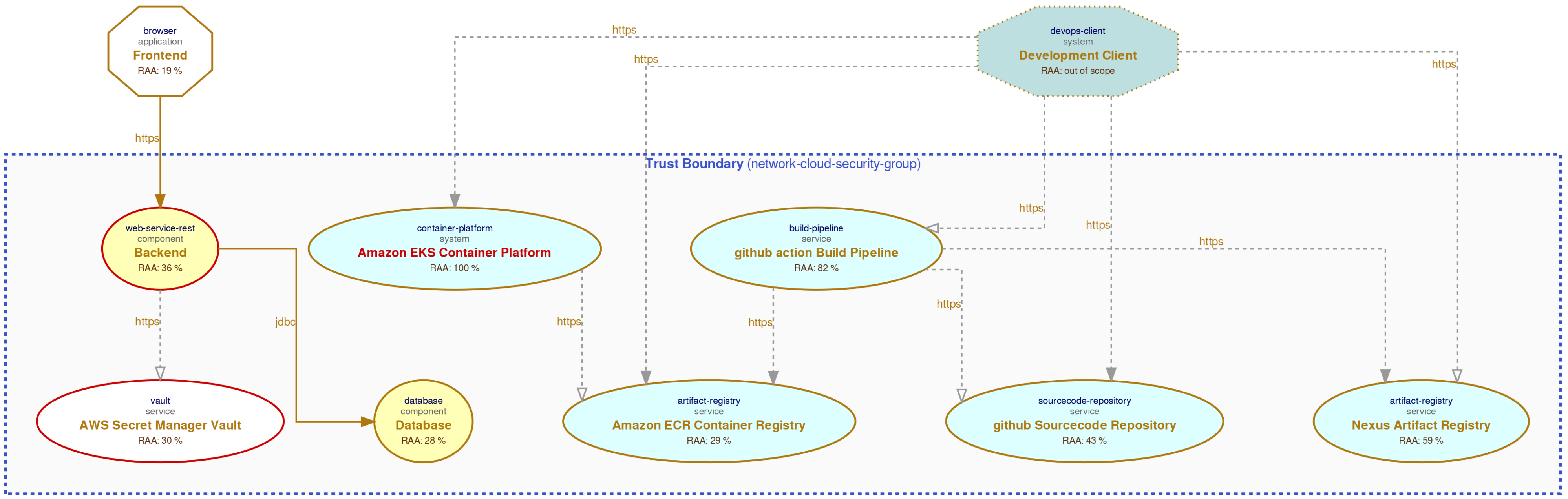
Technical Overview

frontend is in react (which I never used before), backend is in spring boot, database is mysql

Data-Flow Diagram

The following diagram was generated by Threagile based on the model input and gives a high-level overview of the data-flow between technical assets. The RAA value is the calculated *Relative Attacker Attractiveness* in percent. For a full high-resolution version of this diagram please refer to the PNG image file alongside this report.

Data-Flow Diagram - Sample Application



Security Requirements

This chapter lists the custom security requirements which have been defined for the modeled target.

Authentication
Authentication is required to avoid unpleasant greetings from trolls.

Input Validation
Strict input validation is required to reduce the overall attack surface.

This list is not complete and regulatory or law relevant security requirements have to be taken into account as well. Also custom individual security requirements might exist for the project.

Abuse Cases

This chapter lists the custom abuse cases which have been defined for the modeled target.

Abuse Case 1
A malicious user can impersonate somebody else and say bad words to the security architect

Abuse Case 2
A malicious external user can attempt exfiltrating greetings meant for the security architect eyes only

Abuse Case 3
A malicious admin with access to the database can attempt modifying the greeting value

This list is not complete and regulatory or law relevant abuse cases have to be taken into account as well. Also custom individual abuse cases might exist for the project.

Tag Listing

This chapter lists what tags are used by which elements.

amazon ecr
Amazon ECR Container Registry

amazon eks
Amazon EKS Container Platform, Amazon EKS Runtime

aws secret manager
AWS Secret Manager Vault

github
github Sourcecode Repository

github action
github action Build Pipeline

mysql
Database

nexus
Nexus Artifact Registry

nginx
Frontend

react
Frontend

spring
Backend

tomcat
Backend

STRIDE Classification of Identified Risks

This chapter clusters and classifies the risks by STRIDE categories: In total **61 potential risks** have been identified during the threat modeling process of which **1 in the Spoofing** category, **19 in the Tampering** category, **1 in the Repudiation** category, **16 in the Information Disclosure** category, **1 in the Denial of Service** category, and **23 in the Elevation of Privilege** category.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Spoofing

Medium: **Missing Identity Store**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *Medium* impact.
The modeled architecture does not contain an identity store, which might be the risk of a model missing critical assets (and thus not seeing their risks).

Tampering

Elevated: **Missing Cloud Hardening**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *Very High* impact.
Cloud components should be hardened according to the cloud vendor best practices. This affects their configuration, auditing, and further areas.

Elevated: **Missing Hardening**: 3 / 3 Risks - Exploitation likelihood is *Likely* with *Medium* impact.
Technical assets with a Relative Attacker Attractiveness (RAA) value of 55 % or higher should be explicitly hardened taking best practices and vendor hardening guides into account.

Elevated: **SQL/NoSQL-Injection**: 0 / 1 Risk - Exploitation likelihood is *Very Likely* with *Medium* impact.
When a database is accessed via database access protocols SQL/NoSQL-Injection risks might arise. The risk rating depends on the sensitivity technical asset itself and of the data assets processed or stored.

Elevated: **Untrusted Deserialization**: 1 / 1 Risk - Exploitation likelihood is *Likely* with *High* impact.
When a technical asset accepts data in a specific serialized form (like Java or .NET serialization), Untrusted Deserialization risks might arise.

Medium: **Code Backdooring**: 4 / 4 Risks - Exploitation likelihood is *Unlikely* with *High* impact.
For each build-pipeline component Code Backdooring risks might arise where attackers compromise the build-pipeline in order to let backdoored artifacts be shipped into production. Aside from direct code backdooring this includes backdooring of dependencies and even of more lower-level build infrastructure, like backdooring compilers (similar to what the XcodeGhost malware did) or dependencies.

Medium: **Container Base Image Backdooring**: 3 / 3 Risks - Exploitation likelihood is *Unlikely* with *High* impact.
When a technical asset is built using container technologies, Base Image Backdooring risks might arise where base images and other layers used contain vulnerable components or backdoors.

Medium: **Missing Web Application Firewall (WAF)**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *Medium* impact.
To have a first line of filtering defense, security architectures with web-services or web-applications should include a WAF in front of them. Even though a WAF is not a replacement for security (all components must be secure even without a WAF) it adds another layer of defense to the overall system by delaying some attacks and having easier attack alerting through it.

Medium: **Unchecked Deployment**: 5 / 5 Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
For each build-pipeline component Unchecked Deployment risks might arise when the build-pipeline does not include established DevSecOps best-practices. DevSecOps best-practices scan as part of CI/CD pipelines for vulnerabilities in source- or byte-code, dependencies, container layers, and dynamically against running test systems. There are several open-source and commercial tools existing in the categories DAST, SAST, and IAST.

Repudiation

Critical: **Some Individual Risk Example**: 1 / 1 Risk - Exploitation likelihood is *Likely* with *Medium* impact.
Some text describing the risk category...

Information Disclosure

Medium: **Accidental Secret Leak**: 3 / 3 Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
Sourcecode repositories (including their histories) as well as artifact registries can accidentally contain secrets like checked-in or packaged-in passwords, API tokens, certificates, crypto keys, etc.

Medium: **Server-Side Request Forgery (SSRF)**: 5 / 5 Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
When a server system (i.e. not a client) is accessing other server systems via typical web protocols Server-Side Request Forgery (SSRF) or Local-File-Inclusion (LFI) or Remote-File-Inclusion (RFI) risks might arise.

Medium: **Unencrypted Communication**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *High* impact.
Due to the confidentiality and/or integrity rating of the data assets transferred over the communication link this connection must be encrypted.

Medium: **Unencrypted Technical Assets**: 7 / 7 Risks - Exploitation likelihood is *Unlikely* with *High* impact.
Due to the confidentiality rating of the technical asset itself and/or the processed data assets this technical asset must be encrypted. The risk rating depends on the sensitivity technical asset itself and of the data assets stored.

Denial of Service

Low: **DoS-risky Access Across Trust-Boundary**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *Low* impact.
Assets accessed across trust boundaries with critical or mission-critical availability rating are more prone to Denial-of-Service (DoS) risks.

Elevation of Privilege

Elevated: **Missing Authentication**: 1 / 1 Risk - Exploitation likelihood is *Likely* with *Medium* impact.
Technical assets (especially multi-tenant systems) should authenticate incoming requests when the asset processes or stores sensitive data.

Elevated: **Unguarded Access From Internet**: 10 / 10 Risks - Exploitation likelihood is *Very Likely* with *Medium* impact.
Internet-exposed assets must be guarded by a protecting service, application, or reverse-proxy.

Medium: **Container Platform Escape**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *High* impact.
Container platforms are especially interesting targets for attackers as they host big parts of a containerized runtime infrastructure. When not configured and operated with security best practices in mind, attackers might exploit a vulnerability inside an container and escape towards the platform as highly privileged users. These scenarios might give attackers capabilities to attack every other container as owning the container platform (via container escape attacks) equals to owning every container.

Medium: **Missing Network Segmentation**: 3 / 3 Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
Highly sensitive assets and/or datastores residing in the same network segment than other lower sensitive assets (like webservers or content management systems etc.) should be better protected by a network segmentation trust-boundary.

Medium: **Missing Two-Factor Authentication (2FA)**: 6 / 6 Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
Technical assets (especially multi-tenant systems) should authenticate incoming requests with two-factor (2FA) authentication when the asset processes or stores highly sensitive data (in terms of confidentiality, integrity, and availability) and is accessed by humans.

Medium: **Missing Vault Isolation**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *High* impact.
Highly sensitive vault assets and their datastores should be isolated from other assets by their own network segmentation trust-boundary (execution-environment boundaries do not count as network isolation).

Low: **Mixed Targets on Shared Runtime**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *Low* impact.
Different attacker targets (like frontend and backend/datastore components) should not be running on the same shared (underlying) runtime.

Assignment by Function

This chapter clusters and assigns the risks by functions which are most likely able to check and mitigate them: In total **61 potential risks** have been identified during the threat modeling process of which **7 should be checked by Business Side**, **18 should be checked by Architecture**, **6 should be checked by Development**, and **30 should be checked by Operations**.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Business Side

Critical: **Some Individual Risk Example**: 1 / 1 Risk - Exploitation likelihood is *Likely* with *Medium* impact.
Some text describing the mitigation...

Medium: **Missing Two-Factor Authentication (2FA)**: 6 / 6 Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
Apply an authentication method to the technical asset protecting highly sensitive data via two-factor authentication for human users.

Architecture

Elevated: **Missing Authentication**: 1 / 1 Risk - Exploitation likelihood is *Likely* with *Medium* impact.
Apply an authentication method to the technical asset. To protect highly sensitive data consider the use of two-factor authentication for human users.

Elevated: **Unguarded Access From Internet**: 10 / 10 Risks - Exploitation likelihood is *Very Likely* with *Medium* impact.
Encapsulate the asset behind a guarding service, application, or reverse-proxy. For admin maintenance a bastion-host should be used as a jump-server. For file transfer a store-and-forward-host should be used as an indirect file exchange platform.

Elevated: **Untrusted Deserialization**: 1 / 1 Risk - Exploitation likelihood is *Likely* with *High* impact.
Try to avoid the deserialization of untrusted data (even of data within the same trust-boundary as long as it is sent across a remote connection) in order to stay safe from Untrusted Deserialization vulnerabilities. Alternatively a strict whitelisting approach of the classes/types/values to deserialize might help as well. When a third-party product is used instead of custom developed software, check if the product applies the proper mitigation and ensure a reasonable patch-level.

Medium: **Missing Identity Store**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *Medium* impact.
Include an identity store in the model if the application has a login.

Medium: **Unchecked Deployment**: 5 / 5 Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
Apply DevSecOps best-practices and use scanning tools to identify vulnerabilities in source- or byte-code,dependencies, container layers, and optionally also via dynamic scans against running test systems.

Development

Elevated: **SQL/NoSQL-Injection**: 0 / 1 Risk - Exploitation likelihood is *Very Likely* with *Medium* impact.
Try to use parameter binding to be safe from injection vulnerabilities. When a third-party product is used instead of custom developed software, check if the product applies the proper mitigation and ensure a reasonable patch-level.

Medium: **Server-Side Request Forgery (SSRF)**: 5 / 5 Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
Try to avoid constructing the outgoing target URL with caller controllable values. Alternatively use a mapping (whitelist) when accessing outgoing URLs instead of creating them including caller controllable values. When a third-party product is used instead of custom developed software, check if the product applies the proper mitigation and ensure a reasonable patch-level.

Operations

Elevated: **Missing Cloud Hardening**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *Very High* impact.
Apply hardening of all cloud components and services, taking special care to follow the individual risk descriptions (which depend on the cloud provider tags in the model).

Elevated: **Missing Hardening**: 3 / 3 Risks - Exploitation likelihood is *Likely* with *Medium* impact.
Try to apply all hardening best practices (like CIS benchmarks, OWASP recommendations, vendor recommendations, DevSec Hardening Framework, DBSAT for Oracle databases, and others).

Medium: **Accidental Secret Leak**: 3 / 3 Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
Establish measures preventing accidental check-in or package-in of secrets into sourcecode repositories and artifact registries. This starts by using good .gitignore and .dockerignore files, but does not stop there. See for example tools like *"git-secrets" or "Talisman"* to have check-in preventive measures for secrets. Consider also to regularly scan your repositories for secrets accidentally checked-in using scanning tools like *"gitleaks" or "gitrob"*.

Medium: **Code Backdooring**: 4 / 4 Risks - Exploitation likelihood is *Unlikely* with *High* impact.
Reduce the attack surface of backdooring the build pipeline by not directly exposing the build pipeline components on the public internet and also not exposing it in front of unmanaged (out-of-scope) developer clients.Also consider the use of code signing to prevent code modifications.

Medium: **Container Base Image Backdooring**: 3 / 3 Risks - Exploitation likelihood is *Unlikely* with *High* impact.
Apply hardening of all container infrastructures (see for example the *CIS-Benchmarks for Docker and Kubernetes* and the *Docker Bench for Security*). Use only trusted base images of the original vendors, verify digital signatures and apply image creation best practices. Also consider using Google's *Distroless* base images or otherwise very small base images. Regularly execute container image scans with tools checking the layers for vulnerable components.

Medium: **Container Platform Escape**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *High* impact.
Apply hardening of all container infrastructures.

Medium: **Missing Network Segmentation**: 3 / 3 Risks - Exploitation likelihood is *Unlikely* with *Medium* impact.
Apply a network segmentation trust-boundary around the highly sensitive assets and/or datastores.

Medium: **Missing Vault Isolation**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *High* impact.
Apply a network segmentation trust-boundary around the highly sensitive vault assets and their datastores.

Medium: **Missing Web Application Firewall (WAF)**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *Medium* impact.
Consider placing a Web Application Firewall (WAF) in front of the web-services and/or web-applications. For cloud environments many cloud providers offer pre-configured WAFs. Even reverse proxies can be enhances by a WAF component via ModSecurity plugins.

Medium: **Unencrypted Communication**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *High* impact.
Apply transport layer encryption to the communication link.

Medium: **Unencrypted Technical Assets**: 7 / 7 Risks - Exploitation likelihood is *Unlikely* with *High* impact.
Apply encryption to the technical asset.

Low: **DoS-risky Access Across Trust-Boundary**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *Low* impact.
Apply anti-DoS techniques like throttling and/or per-client load blocking with quotas. Also for maintenance access routes consider applying a VPN instead of public reachable interfaces. Generally applying redundancy on the targeted technical asset reduces the risk of DoS.

Low: **Mixed Targets on Shared Runtime**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *Low* impact.
Use separate runtime environments for running different target components or apply similar separation styles to prevent load- or breach-related problems originating from one more attacker-facing asset impacts also the other more critical rated backend/datastore assets.

RAA Analysis

For each technical asset the **"Relative Attacker Attractiveness"** (RAA) value was calculated in percent. The higher the RAA, the more interesting it is for an attacker to compromise the asset. The calculation algorithm takes the sensitivity ratings and quantities of stored and processed data into account as well as the communication links of the technical asset. Neighbouring assets to high-value RAA targets might receive an increase in their RAA value when they have a communication link towards that target ("Pivoting-Factor").

The following lists all technical assets sorted by their RAA value from highest (most attacker attractive) to lowest. This list can be used to prioritize on efforts relevant for the most attacker-attractive technical assets:
Technical asset paragraphs are clickable and link to the corresponding chapter.

Amazon EKS Container Platform: RAA 100%
Amazon EKS Container Platform

github action Build Pipeline: RAA 82%
github action Build Pipeline

Nexus Artifact Registry: RAA 59%
Nexus Artifact Registry

github Sourcecode Repository: RAA 43%
github Sourcecode Repository

Backend: RAA 36%
Some Description

AWS Secret Manager Vault: RAA 30%
AWS Secret Manager Vault

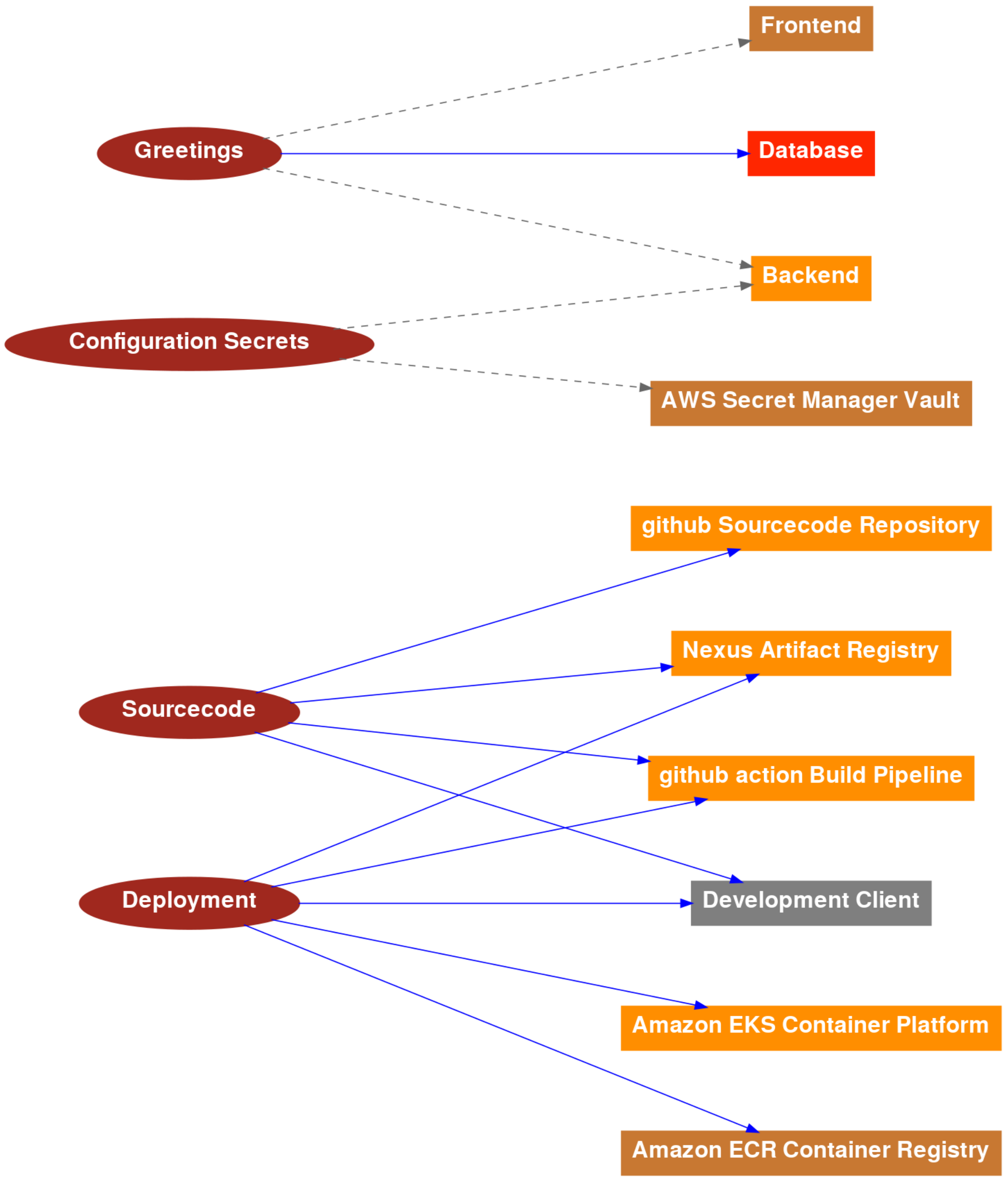
Amazon ECR Container Registry: RAA 29%
Amazon ECR Container Registry

Database: RAA 28%
Some Description

Frontend: RAA 19%
react frontend

Data Mapping

The following diagram was generated by Threagile based on the model input and gives a high-level distribution of data assets across technical assets. The color matches the identified data breach probability and risk level (see the "Data Breach Probabilities" chapter for more details). A solid line stands for *data is stored by the asset* and a dashed one means *data is processed by the asset*. For a full high-resolution version of this diagram please refer to the PNG image file alongside this report.



Out-of-Scope Assets: 1 Asset

This chapter lists all technical assets that have been defined as out-of-scope. Each one should be checked in the model whether it should better be included in the overall risk analysis:
Technical asset paragraphs are clickable and link to the corresponding chapter.

Development Client: out-of-scope
Development client is not directly in-scope of the application.

Potential Model Failures: 1 / 1 Risk

This chapter lists potential model failures where not all relevant assets have been modeled or the model might itself contain inconsistencies. Each potential model failure should be checked in the model against the architecture design:

Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium: **Missing Identity Store**: 1 / 1 Risk - Exploitation likelihood is *Unlikely* with *Medium* impact.
The modeled architecture does not contain an identity store, which might be the risk of a model missing critical assets (and thus not seeing their risks).

Questions: 1 / 2 Questions

This chapter lists custom questions that arose during the threat modeling process.

Some question with an answer?
Some answer

Some question without an answer?
- answer pending -

Identified Risks by Vulnerability Category

In total **61 potential risks** have been identified during the threat modeling process of which **1 are rated as critical**, **0 as high**, **11 as elevated**, **42 as medium**, and **7 as low**.

These risks are distributed across **22 vulnerability categories**. The following sub-chapters of this section describe each identified risk category.

Some Individual Risk Example: 1 / 1 Risk

Description (Repudiation): CWE 693

Some text describing the risk category...

Impact

Some text describing the impact...

Detection Logic

Some text describing the detection logic...

Risk Rating

Some text describing the risk assessment...

False Positives

Some text describing the most common types of false positives...

Mitigation (Business Side): Some text describing the action...

Some text describing the mitigation...

ASVS Chapter: V0 - Something Strange
Cheat Sheet: example.com

Check

Check if XYZ...

Risk Findings

The risk **Some Individual Risk Example** was found **1 time** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Critical Risk Severity

Example Individual Risk at **Some Technical Asset**: Exploitation likelihood is *Likely* with *Medium* impact.
something-strange@database
Unchecked

Missing Authentication: 1 / 1 Risk

Description (Elevation of Privilege): CWE 306

Technical assets (especially multi-tenant systems) should authenticate incoming requests when the asset processes or stores sensitive data.

Impact

If this risk is unmitigated, attackers might be able to access or modify sensitive data in an unauthenticated way.

Detection Logic

In-scope technical assets (except load-balancer, reverse-proxy, service-registry, waf, ids, and ips and in-process calls) should authenticate incoming requests when the asset processes or stores sensitive data. This is especially the case for all multi-tenant assets (there even non-sensitive ones).

Risk Rating

The risk rating (medium or high) depends on the sensitivity of the data sent across the communication link. Monitoring callers are exempted from this risk.

False Positives

Technical assets which do not process requests regarding functionality or data linked to end-users (customers) can be considered as false positives after individual review.

Mitigation (Architecture): Authentication of Incoming Requests

Apply an authentication method to the technical asset. To protect highly sensitive data consider the use of two-factor authentication for human users.

ASVS Chapter: V2 - Authentication Verification Requirements
Cheat Sheet: Authentication_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Missing Authentication** was found **1 time** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Elevated Risk Severity

Missing Authentication covering communication link **User Traffic** from **Frontend** to **Backend**: Exploitation likelihood is *Likely* with *Medium* impact.

missing-authentication@frontend>user-traffic@frontend@backend

Unchecked

Missing Cloud Hardening: 1 / 1 Risk

Description (Tampering): CWE 1008

Cloud components should be hardened according to the cloud vendor best practices. This affects their configuration, auditing, and further areas.

Impact

If this risk is unmitigated, attackers might access cloud components in an unintended way.

Detection Logic

In-scope cloud components (either residing in cloud trust boundaries or more specifically tagged with cloud provider types).

Risk Rating

The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed and stored.

False Positives

Cloud components not running parts of the target architecture can be considered as false positives after individual review.

Mitigation (Operations): Cloud Hardening

Apply hardening of all cloud components and services, taking special care to follow the individual risk descriptions (which depend on the cloud provider tags in the model).

For **Amazon Web Services (AWS)**: Follow the *CIS Benchmark for Amazon Web Services* (see also the automated checks of cloud audit tools like *"PacBot", "CloudSploit", "CloudMapper", "ScoutSuite", or "Prowler AWS CIS Benchmark Tool"*).
For EC2 and other servers running Amazon Linux, follow the *CIS Benchmark for Amazon Linux* and switch to IMDSv2.
For S3 buckets follow the *Security Best Practices for Amazon S3* at https://docs.aws.amazon.com/AmazonS3/latest/dev/security-best-practices.html to avoid accidental leakage.
Also take a look at some of these tools: https://github.com/toniblyx/my-arsenal-of-aws-security-tools

For **Microsoft Azure**: Follow the *CIS Benchmark for Microsoft Azure* (see also the automated checks of cloud audit tools like *"CloudSploit" or "ScoutSuite"*).

For **Google Cloud Platform**: Follow the *CIS Benchmark for Google Cloud Computing Platform* (see also the automated checks of cloud audit tools like *"CloudSploit" or "ScoutSuite"*).

For **Oracle Cloud Platform**: Follow the hardening best practices (see also the automated checks of cloud audit tools like *"CloudSploit"*).

ASVS Chapter: V1 - Architecture, Design and Threat Modeling Requirements
Cheat Sheet: Attack_Surface_Analysis_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Missing Cloud Hardening** was found **1 time** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.

Risk finding paragraphs are clickable and link to the corresponding chapter.

Elevated Risk Severity

Missing Cloud Hardening risk at **Trust Boundary**: Exploitation likelihood is *Unlikely* with *Very High* impact.

missing-cloud-hardening@trusted-boundary

Unchecked

Missing Hardening: 3 / 3 Risks

Description (Tampering): CWE 16

Technical assets with a Relative Attacker Attractiveness (RAA) value of 55 % or higher should be explicitly hardened taking best practices and vendor hardening guides into account.

Impact

If this risk remains unmitigated, attackers might be able to easier attack high-value targets.

Detection Logic

In-scope technical assets with RAA values of 55 % or higher. Generally for high-value targets like datastores, application servers, identity providers and ERP systems this limit is reduced to 40 %

Risk Rating

The risk rating depends on the sensitivity of the data processed or stored in the technical asset.

False Positives

Usually no false positives.

Mitigation (Operations): System Hardening

Try to apply all hardening best practices (like CIS benchmarks, OWASP recommendations, vendor recommendations, DevSec Hardening Framework, DBSAT for Oracle databases, and others).

ASVS Chapter: V14 - Configuration Verification Requirements
Cheat Sheet: Attack_Surface_Analysis_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Missing Hardening** was found **3 times** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Elevated Risk Severity

Missing Hardening risk at **Amazon EKS Container Platform**: Exploitation likelihood is *Likely* with *Medium* impact.

missing-hardening@amazon-eks-container-platform

in Discussion 2023-01-10 Ciro Bologna XYZ-1234
The hardening measures on the docker images are completed, the network ones in progress

Medium Risk Severity

Missing Hardening risk at **Nexus Artifact Registry**: Exploitation likelihood is *Likely* with *Low* impact.

missing-hardening@nexus-artifact-registry

in Discussion 2023-01-10 Ciro Bologna XYZ-1234
The hardening measures on the docker images are completed, the network ones in progress

Missing Hardening risk at **github action Build Pipeline**: Exploitation likelihood is *Likely* with *Low* impact.

missing-hardening@github-action-build-pipeline

in Discussion 2023-01-10 Ciro Bologna XYZ-1234
The hardening measures on the docker images are completed, the network ones in progress

Unguarded Access From Internet: 10 / 10 Risks

Description (Elevation of Privilege): CWE 501

Internet-exposed assets must be guarded by a protecting service, application, or reverse-proxy.

Impact

If this risk is unmitigated, attackers might be able to directly attack sensitive systems without any hardening components in-between due to them being directly exposed on the internet.

Detection Logic

In-scope technical assets (excluding load-balancer) with confidentiality rating of confidential (or higher) or with integrity rating of critical (or higher) when accessed directly from the internet. All web-server, web-application, reverse-proxy, waf, and gateway assets are exempted from this risk when they do not consist of custom developed code and the data-flow only consists of HTTP or FTP protocols. Access from monitoring systems as well as VPN-protected connections are exempted.

Risk Rating

The matching technical assets are at low risk. When either the confidentiality rating is strictly-confidential or the integrity rating is mission-critical, the risk-rating is considered medium. For assets with RAA values higher than 40 % the risk-rating increases.

False Positives

When other means of filtering client requests are applied equivalent of reverse-proxy, waf, or gateway components.

Mitigation (Architecture): Encapsulation of Technical Asset

Encapsulate the asset behind a guarding service, application, or reverse-proxy. For admin maintenance a bastion-host should be used as a jump-server. For file transfer a store-and-forward-host should be used as an indirect file exchange platform.

ASVS Chapter: V1 - Architecture, Design and Threat Modeling Requirements
Cheat Sheet: Attack_Surface_Analysis_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Unguarded Access From Internet** was found **10 times** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Elevated Risk Severity

Unguarded Access from Internet of **Amazon EKS Container Platform** by **Development Client** via **Container Platform Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.
unguarded-access-from-internet@amazon-eks-container-platform@development-client@development-client>container-platform-traffic
Unchecked

Unguarded Access from Internet of **Nexus Artifact Registry** by **Development Client** via **Artifact Registry Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.
unguarded-access-from-internet@nexus-artifact-registry@development-client@development-client>artifact-registry-traffic
Unchecked

Unguarded Access from Internet of **Nexus Artifact Registry** by **github action Build Pipeline** via **Artifact Registry Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.
unguarded-access-from-internet@nexus-artifact-registry@github-action-build-pipeline@github-action-build-pipeline>artifact-registry-traffic
Unchecked

Unguarded Access from Internet of **github Sourcecode Repository** by **Development Client** via **Sourcecode Repository Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.
unguarded-access-from-internet@github-sourcecode-repository@development-client@development-client>sourcecode-repository-traffic
Unchecked

Unguarded Access from Internet of **github Sourcecode Repository** by **github action Build Pipeline** via **Sourcecode Repository Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.
unguarded-access-from-internet@github-sourcecode-repository@github-action-build-pipeline@github-action-build-pipeline>sourcecode-repository-traffic
Unchecked

Unguarded Access from Internet of **github action Build Pipeline** by **Development Client** via **Build Pipeline Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.
unguarded-access-from-internet@github-action-build-pipeline@development-client@development-client>build-pipeline-traffic
Unchecked

Medium Risk Severity

Unguarded Access from Internet of **Amazon ECR Container Registry** by **Amazon EKS Container Platform** via **Container Platform Pull**: Exploitation likelihood is *Very Likely* with

Low impact.

unguarded-access-from-internet@amazon-ecr-container-registry@amazon-eks-container-platform@amazon-eks-container-platform>container-platform-pull

Unchecked

Unguarded Access from Internet of **Amazon ECR Container Registry** by **Development Client** via **Container Registry Traffic**: Exploitation likelihood is *Very Likely* with *Low* impact.

unguarded-access-from-internet@amazon-ecr-container-registry@development-client@development-client>container-registry-traffic

Unchecked

Unguarded Access from Internet of **Amazon ECR Container Registry** by **github action Build Pipeline** via **Container Registry Traffic**: Exploitation likelihood is *Very Likely* with *Low* impact.

unguarded-access-from-internet@amazon-ecr-container-registry@github-action-build-pipeline@github-action-build-pipeline>container-registry-traffic

Unchecked

Unguarded Access from Internet of **Backend** by **Frontend** via **User Traffic**: Exploitation likelihood is *Very Likely* with *Low* impact.

unguarded-access-from-internet@backend@frontend@frontend>user-traffic

Unchecked

Untrusted Deserialization: 1 / 1 Risk

Description (Tampering): CWE 502

When a technical asset accepts data in a specific serialized form (like Java or .NET serialization), Untrusted Deserialization risks might arise.

See https://christian-schneider.net/JavaDeserializationSecurityFAQ.html for more details.

Impact

If this risk is unmitigated, attackers might be able to execute code on target systems by exploiting untrusted deserialization endpoints.

Detection Logic

In-scope technical assets accepting serialization data formats (including EJB and RMI protocols).

Risk Rating

The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed and stored.

False Positives

Fully trusted (i.e. cryptographically signed or similar) data deserialized can be considered as false positives after individual review.

Mitigation (Architecture): Prevention of Deserialization of Untrusted Data

Try to avoid the deserialization of untrusted data (even of data within the same trust-boundary as long as it is sent across a remote connection) in order to stay safe from Untrusted Deserialization vulnerabilities. Alternatively a strict whitelisting approach of the classes/types/values to deserialize might help as well. When a third-party product is used instead of custom developed software, check if the product applies the proper mitigation and ensure a reasonable patch-level.

ASVS Chapter: V5 - Validation, Sanitization and Encoding Verification Requirements
Cheat Sheet: Deserialization_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Untrusted Deserialization** was found **1 time** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Elevated Risk Severity

Untrusted Deserialization risk at **Database**: Exploitation likelihood is *Likely* with *High* impact.

untrusted-deserialization@database

Unchecked

Accidental Secret Leak: 3 / 3 Risks

Description (Information Disclosure): CWE 200

Sourcecode repositories (including their histories) as well as artifact registries can accidentally contain secrets like checked-in or packaged-in passwords, API tokens, certificates, crypto keys, etc.

Impact

If this risk is unmitigated, attackers which have access to affected sourcecode repositories or artifact registries might find secrets accidentally checked-in.

Detection Logic

In-scope sourcecode repositories and artifact registries.

Risk Rating

The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed and stored.

False Positives

Usually no false positives.

Mitigation (Operations): Build Pipeline Hardening

Establish measures preventing accidental check-in or package-in of secrets into sourcecode repositories and artifact registries. This starts by using good .gitignore and .dockerignore files, but does not stop there. See for example tools like *"git-secrets" or "Talisman"* to have check-in preventive measures for secrets. Consider also to regularly scan your repositories for secrets accidentally checked-in using scanning tools like *"gitleaks" or "gitrob"*.

ASVS Chapter: V14 - Configuration Verification Requirements
Cheat Sheet: Attack_Surface_Analysis_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Accidental Secret Leak** was found **3 times** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Accidental Secret Leak risk at **Amazon ECR Container Registry**: Exploitation likelihood is *Unlikely* with *Medium* impact.

accidental-secret-leak@amazon-ecr-container-registry

Unchecked

Accidental Secret Leak risk at **Nexus Artifact Registry**: Exploitation likelihood is *Unlikely* with *Medium* impact.

accidental-secret-leak@nexus-artifact-registry

Unchecked

Accidental Secret Leak risk at **github Sourcecode Repository**: Exploitation likelihood is *Unlikely* with *Medium* impact.

accidental-secret-leak@github-sourcecode-repository

Unchecked

Code Backdooring: 4 / 4 Risks

Description (Tampering): CWE 912

For each build-pipeline component Code Backdooring risks might arise where attackers compromise the build-pipeline in order to let backdoored artifacts be shipped into production. Aside from direct code backdooring this includes backdooring of dependencies and even of more lower-level build infrastructure, like backdooring compilers (similar to what the XcodeGhost malware did) or dependencies.

Impact

If this risk remains unmitigated, attackers might be able to execute code on and completely takeover production environments.

Detection Logic

In-scope development relevant technical assets which are either accessed by out-of-scope unmanaged developer clients and/or are directly accessed by any kind of internet-located (non-VPN) component or are themselves directly located on the internet.

Risk Rating

The risk rating depends on the confidentiality and integrity rating of the code being handled and deployed as well as the placement/calling of this technical asset on/from the internet.

False Positives

When the build-pipeline and sourcecode-repo is not exposed to the internet and considered fully trusted (which implies that all accessing clients are also considered fully trusted in terms of their patch management and applied hardening, which must be equivalent to a managed developer client environment) this can be considered a false positive after individual review.

Mitigation (Operations): Build Pipeline Hardening

Reduce the attack surface of backdooring the build pipeline by not directly exposing the build pipeline components on the public internet and also not exposing it in front of unmanaged (out-of-scope) developer clients.Also consider the use of code signing to prevent code modifications.

ASVS Chapter: V10 - Malicious Code Verification Requirements
Cheat Sheet: Vulnerable_Dependency_Management_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Code Backdooring** was found **4 times** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Code Backdooring risk at **Amazon ECR Container Registry**: Exploitation likelihood is *Unlikely* with *High* impact.
code-backdooring@amazon-ecr-container-registry
Unchecked

Code Backdooring risk at **Nexus Artifact Registry**: Exploitation likelihood is *Unlikely* with *High* impact.
code-backdooring@nexus-artifact-registry
Unchecked

Code Backdooring risk at **github Sourcecode Repository**: Exploitation likelihood is *Unlikely* with *High* impact.
code-backdooring@github-sourcecode-repository
Unchecked

Code Backdooring risk at **github action Build Pipeline**: Exploitation likelihood is *Unlikely* with *High* impact.
code-backdooring@github-action-build-pipeline
Unchecked

Container Base Image Backdooring: 3 / 3 Risks

Description (Tampering): CWE 912

When a technical asset is built using container technologies, Base Image Backdooring risks might arise where base images and other layers used contain vulnerable components or backdoors.

See for example:
https://techcrunch.com/2018/06/15/tainted-crypto-mining-containers-pulled-from-docker-hub/

Impact

If this risk is unmitigated, attackers might be able to deeply persist in the target system by executing code in deployed containers.

Detection Logic

In-scope technical assets running as containers.

Risk Rating

The risk rating depends on the sensitivity of the technical asset itself and of the data assets.

False Positives

Fully trusted (i.e. reviewed and cryptographically signed or similar) base images of containers can be considered as false positives after individual review.

Mitigation (Operations): Container Infrastructure Hardening

Apply hardening of all container infrastructures (see for example the *CIS-Benchmarks for Docker and Kubernetes* and the *Docker Bench for Security*). Use only trusted base images of the original vendors, verify digital signatures and apply image creation best practices. Also consider using Google's *Distroless* base images or otherwise very small base images. Regularly execute container image scans with tools checking the layers for vulnerable components.

ASVS Chapter: V10 - Malicious Code Verification Requirements
Cheat Sheet: Docker_Security_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS/CSVS applied?

Risk Findings

The risk **Container Base Image Backdooring** was found **3 times** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Container Base Image Backdooring risk at **Backend**: Exploitation likelihood is *Unlikely* with *High* impact.

container-baseimage-backdooring@backend

Unchecked

Container Base Image Backdooring risk at **Database**: Exploitation likelihood is *Unlikely* with *Medium* impact.

container-baseimage-backdooring@database

Unchecked

Container Base Image Backdooring risk at **Frontend**: Exploitation likelihood is *Unlikely* with *Medium* impact.

container-baseimage-backdooring@frontend

Unchecked

Container Platform Escape: 1 / 1 Risk

Description (Elevation of Privilege): CWE 1008

Container platforms are especially interesting targets for attackers as they host big parts of a containerized runtime infrastructure. When not configured and operated with security best practices in mind, attackers might exploit a vulnerability inside an container and escape towards the platform as highly privileged users. These scenarios might give attackers capabilities to attack every other container as owning the container platform (via container escape attacks) equals to owning every container.

Impact

If this risk is unmitigated, attackers which have successfully compromised a container (via other vulnerabilities) might be able to deeply persist in the target system by executing code in many deployed containers and the container platform itself.

Detection Logic

In-scope container platforms.

Risk Rating

The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed and stored.

False Positives

Container platforms not running parts of the target architecture can be considered as false positives after individual review.

Mitigation (Operations): Container Infrastructure Hardening

Apply hardening of all container infrastructures. See for example the *CIS-Benchmarks for Docker and Kubernetes* as well as the *Docker Bench for Security* (https://github.com/docker/docker-bench-security) or *InSpec Checks for Docker and Kubernetes* (https://github.com/dev-sec/cis-docker-benchmark and https://github.com/dev-sec/cis-kubernetes-benchmark). Use only trusted base images, verify digital signatures and apply image creation best practices. Also consider using Google's **Distroless base images or otherwise very small base images. Apply namespace isolation and nod affinity to separate pods from each other in terms of access and nodes the same style as you separate data.**

ASVS Chapter: V14 - Configuration Verification Requirements
Cheat Sheet: Docker_Security_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS or CSVS chapter applied?

Risk Findings

The risk **Container Platform Escape** was found **1 time** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Container Platform Escape risk at **Amazon EKS Container Platform**: Exploitation likelihood is *Unlikely* with *High* impact.
container-platform-escape@amazon-eks-container-platform

Unchecked

Missing Identity Store: 1 / 1 Risk

Description (Spoofing): CWE 287

The modeled architecture does not contain an identity store, which might be the risk of a model missing critical assets (and thus not seeing their risks).

Impact

If this risk is unmitigated, attackers might be able to exploit risks unseen in this threat model in the identity provider/store that is currently missing in the model.

Detection Logic

Models with authenticated data-flows authorized via enduser-identity missing an in-scope identity store.

Risk Rating

The risk rating depends on the sensitivity of the enduser-identity authorized technical assets and their data assets processed and stored.

False Positives

Models only offering data/services without any real authentication need can be considered as false positives after individual review.

Mitigation (Architecture): Identity Store

Include an identity store in the model if the application has a login.

ASVS Chapter: V2 - Authentication Verification Requirements
Cheat Sheet: Authentication_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Missing Identity Store** was found **1 time** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Missing Identity Store in the threat model (referencing asset **github Sourcecode Repository** as an example): Exploitation likelihood is *Unlikely* with *Medium* impact.
missing-identity-store@github-sourcecode-repository

Unchecked

Missing Network Segmentation: 3 / 3 Risks

Description (Elevation of Privilege): CWE 1008

Highly sensitive assets and/or datastores residing in the same network segment than other lower sensitive assets (like webservers or content management systems etc.) should be better protected by a network segmentation trust-boundary.

Impact

If this risk is unmitigated, attackers successfully attacking other components of the system might have an easy path towards more valuable targets, as they are not separated by network segmentation.

Detection Logic

In-scope technical assets with high sensitivity and RAA values as well as datastores when surrounded by assets (without a network trust-boundary in-between) which are of type client-system, web-server, web-application, cms, web-service-rest, web-service-soap, build-pipeline, sourcecode-repository, monitoring, or similar and there is no direct connection between these (hence no requirement to be so close to each other).

Risk Rating

Default is low risk. The risk is increased to medium when the asset missing the trust-boundary protection is rated as strictly-confidential or mission-critical.

False Positives

When all assets within the network segmentation trust-boundary are hardened and protected to the same extend as if all were containing/processing highly sensitive data.

Mitigation (Operations): Network Segmentation

Apply a network segmentation trust-boundary around the highly sensitive assets and/or datastores.

ASVS Chapter: V1 - Architecture, Design and Threat Modeling Requirements
Cheat Sheet: Attack_Surface_Analysis_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Missing Network Segmentation** was found **3 times** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Missing Network Segmentation to further encapsulate and protect **Amazon EKS Container Platform** against unrelated lower protected assets in the same network segment, which might be easier to compromise by attackers: Exploitation likelihood is *Unlikely* with *Medium* impact.
missing-network-segmentation@amazon-eks-container-platform
Unchecked

Low Risk Severity

Missing Network Segmentation to further encapsulate and protect **Nexus Artifact Registry** against unrelated lower protected assets in the same network segment, which might be easier to compromise by attackers: Exploitation likelihood is *Unlikely* with *Low* impact.
missing-network-segmentation@nexus-artifact-registry
Unchecked

Missing Network Segmentation to further encapsulate and protect **github action Build Pipeline** against unrelated lower protected assets in the same network segment, which might be easier to compromise by attackers: Exploitation likelihood is *Unlikely* with *Low* impact.
missing-network-segmentation@github-action-build-pipeline
Unchecked

Missing Two-Factor Authentication (2FA): 6 / 6 Risks

Description (Elevation of Privilege): CWE 308

Technical assets (especially multi-tenant systems) should authenticate incoming requests with two-factor (2FA) authentication when the asset processes or stores highly sensitive data (in terms of confidentiality, integrity, and availability) and is accessed by humans.

Impact

If this risk is unmitigated, attackers might be able to access or modify highly sensitive data without strong authentication.

Detection Logic

In-scope technical assets (except load-balancer, reverse-proxy, waf, ids, and ips) should authenticate incoming requests via two-factor authentication (2FA) when the asset processes or stores highly sensitive data (in terms of confidentiality, integrity, and availability) and is accessed by a client used by a human user.

Risk Rating

medium

False Positives

Technical assets which do not process requests regarding functionality or data linked to end-users (customers) can be considered as false positives after individual review.

Mitigation (Business Side): Authentication with Second Factor (2FA)

Apply an authentication method to the technical asset protecting highly sensitive data via two-factor authentication for human users.

ASVS Chapter: V2 - Authentication Verification Requirements
Cheat Sheet: Multifactor_Authentication_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Missing Two-Factor Authentication (2FA)** was found **6 times** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Missing Two-Factor Authentication covering communication link **Artifact Registry Traffic** from **Development Client** to **Nexus Artifact Registry**: Exploitation likelihood is *Unlikely* with *Medium* impact.
missing-authentication-second-factor@development-client>artifact-registry-traffic@development-client@nexus-artifact-registry
Unchecked

Missing Two-Factor Authentication covering communication link **Build Pipeline Traffic** from **Development Client** to **github action Build Pipeline**: Exploitation likelihood is *Unlikely* with *Medium* impact.
missing-authentication-second-factor@development-client>build-pipeline-traffic@development-client@github-action-build-pipeline
Unchecked

Missing Two-Factor Authentication covering communication link **Container Platform Traffic** from **Development Client** to **Amazon EKS Container Platform**: Exploitation likelihood is *Unlikely* with *Medium* impact.
missing-authentication-second-factor@development-client>container-platform-traffic@development-client@amazon-eks-container-platform
Unchecked

Missing Two-Factor Authentication covering communication link **Container Registry Traffic** from **Development Client** to **Amazon ECR Container Registry**: Exploitation likelihood is *Unlikely* with *Medium* impact.
missing-authentication-second-factor@development-client>container-registry-traffic@development-client@amazon-ecr-container-registry
Unchecked

Missing Two-Factor Authentication covering communication link **Sourcecode Repository Traffic** from **Development Client** to **github Sourcecode Repository**: Exploitation likelihood is *Unlikely* with *Medium* impact.
missing-authentication-second-factor@development-client>sourcecode-repository-traffic@development-client@github-sourcecode-repository
Unchecked

Missing Two-Factor Authentication covering communication link **User Traffic** from **Frontend** to **Backend**: Exploitation likelihood is *Unlikely* with *Medium* impact.
missing-authentication-second-factor@frontend>user-traffic@frontend@backend
Unchecked

Missing Vault Isolation: 1 / 1 Risk

Description (Elevation of Privilege): CWE 1008

Highly sensitive vault assets and their datastores should be isolated from other assets by their own network segmentation trust-boundary (execution-environment boundaries do not count as network isolation).

Impact

If this risk is unmitigated, attackers successfully attacking other components of the system might have an easy path towards highly sensitive vault assets and their datastores, as they are not separated by network segmentation.

Detection Logic

In-scope vault assets when surrounded by other (not vault-related) assets (without a network trust-boundary in-between). This risk is especially prevalent when other non-vault related assets are within the same execution environment (i.e. same database or same application server).

Risk Rating

Default is medium impact. The impact is increased to high when the asset missing the trust-boundary protection is rated as strictly-confidential or mission-critical.

False Positives

When all assets within the network segmentation trust-boundary are hardened and protected to the same extend as if all were vaults with data of highest sensitivity.

Mitigation (Operations): Network Segmentation

Apply a network segmentation trust-boundary around the highly sensitive vault assets and their datastores.

ASVS Chapter: V1 - Architecture, Design and Threat Modeling Requirements
Cheat Sheet: Attack_Surface_Analysis_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Missing Vault Isolation** was found **1 time** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Missing Vault Isolation to further encapsulate and protect vault-related asset **AWS Secret Manager Vault** against unrelated lower protected assets **in the same network segment**, which might be easier to compromise by attackers: Exploitation likelihood is *Unlikely* with *High* impact.
missing-vault-isolation@aws-secret-manager-vault
Unchecked

Missing Web Application Firewall (WAF): 1 / 1 Risk

Description (Tampering): CWE 1008

To have a first line of filtering defense, security architectures with web-services or web-applications should include a WAF in front of them. Even though a WAF is not a replacement for security (all components must be secure even without a WAF) it adds another layer of defense to the overall system by delaying some attacks and having easier attack alerting through it.

Impact

If this risk is unmitigated, attackers might be able to apply standard attack pattern tests at great speed without any filtering.

Detection Logic

In-scope web-services and/or web-applications accessed across a network trust boundary not having a Web Application Firewall (WAF) in front of them.

Risk Rating

The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed and stored.

False Positives

Targets only accessible via WAFs or reverse proxies containing a WAF component (like ModSecurity) can be considered as false positives after individual review.

Mitigation (Operations): Web Application Firewall (WAF)

Consider placing a Web Application Firewall (WAF) in front of the web-services and/or web-applications. For cloud environments many cloud providers offer pre-configured WAFs. Even reverse proxies can be enhances by a WAF component via ModSecurity plugins.

ASVS Chapter: V1 - Architecture, Design and Threat Modeling Requirements
Cheat Sheet: Virtual_Patching_Cheat_Sheet

Check

Is a Web Application Firewall (WAF) in place?

Risk Findings

The risk **Missing Web Application Firewall (WAF)** was found **1 time** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Missing Web Application Firewall (WAF) risk at **Backend**: Exploitation likelihood is *Unlikely* with *Medium* impact.
missing-waf@backend
Unchecked

Server-Side Request Forgery (SSRF): 5 / 5 Risks

Description (Information Disclosure): CWE 918

When a server system (i.e. not a client) is accessing other server systems via typical web protocols Server-Side Request Forgery (SSRF) or Local-File-Inclusion (LFI) or Remote-File-Inclusion (RFI) risks might arise.

Impact

If this risk is unmitigated, attackers might be able to access sensitive services or files of network-reachable components by modifying outgoing calls of affected components.

Detection Logic

In-scope non-client systems accessing (using outgoing communication links) targets with either HTTP or HTTPS protocol.

Risk Rating

The risk rating (low or medium) depends on the sensitivity of the data assets receivable via web protocols from targets within the same network trust-boundary as well on the sensitivity of the data assets receivable via web protocols from the target asset itself. Also for cloud-based environments the exploitation impact is at least medium, as cloud backend services can be attacked via SSRF.

False Positives

Servers not sending outgoing web requests can be considered as false positives after review.

Mitigation (Development): SSRF Prevention

Try to avoid constructing the outgoing target URL with caller controllable values. Alternatively use a mapping (whitelist) when accessing outgoing URLs instead of creating them including caller controllable values. When a third-party product is used instead of custom developed software, check if the product applies the proper mitigation and ensure a reasonable patch-level.

ASVS Chapter: V12 - File and Resources Verification Requirements
Cheat Sheet: Server_Side_Request_Forgery_Prevention_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Server-Side Request Forgery (SSRF)** was found **5 times** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Server-Side Request Forgery (SSRF) risk at **Amazon EKS Container Platform** server-side web-requesting the target **Amazon ECR Container Registry** via **Container Platform Pull**: Exploitation likelihood is *Unlikely* with *Medium* impact.
server-side-request-forgery@amazon-eks-container-platform@amazon-ecr-container-registry@amazon-eks-container-platform>container-platform-pull

Unchecked

Server-Side Request Forgery (SSRF) risk at **Backend** server-side web-requesting the target **AWS Secret Manager Vault** via **Vault Access (backend)**: Exploitation likelihood is *Unlikely* with *Medium* impact.
server-side-request-forgery@backend@aws-secret-manager-vault@backend>vault-access-backend

Unchecked

Server-Side Request Forgery (SSRF) risk at **github action Build Pipeline** server-side web-requesting the target **Amazon ECR Container Registry** via **Container Registry Traffic**: Exploitation likelihood is *Unlikely* with *Medium* impact.
server-side-request-forgery@github-action-build-pipeline@amazon-ecr-container-registry@github-action-build-pipeline>container-registry-traffic

Unchecked

Server-Side Request Forgery (SSRF) risk at **github action Build Pipeline** server-side web-requesting the target **Nexus Artifact Registry** via **Artifact Registry Traffic**: Exploitation likelihood is *Unlikely* with *Medium* impact.
server-side-request-forgery@github-action-build-pipeline@nexus-artifact-registry@github-action-build-pipeline>artifact-registry-traffic

Unchecked

Server-Side Request Forgery (SSRF) risk at **github action Build Pipeline** server-side web-requesting the target **github Sourcecode Repository** via **Sourcecode Repository Traffic**: Exploitation likelihood is *Unlikely* with *Medium* impact.
server-side-request-forgery@github-action-build-pipeline@github-sourcecode-repository@github-action-build-pipeline>sourcecode-repository-traffic

Unchecked

Unchecked Deployment: 5 / 5 Risks

Description (Tampering): CWE 1127

For each build-pipeline component Unchecked Deployment risks might arise when the build-pipeline does not include established DevSecOps best-practices. DevSecOps best-practices scan as part of CI/CD pipelines for vulnerabilities in source- or byte-code, dependencies, container layers, and dynamically against running test systems. There are several open-source and commercial tools existing in the categories DAST, SAST, and IAST.

Impact

If this risk remains unmitigated, vulnerabilities in custom-developed software or their dependencies might not be identified during continuous deployment cycles.

Detection Logic

All development-relevant technical assets.

Risk Rating

The risk rating depends on the highest rating of the technical assets and data assets processed by deployment-receiving targets.

False Positives

When the build-pipeline does not build any software components it can be considered a false positive after individual review.

Mitigation (Architecture): Build Pipeline Hardening

Apply DevSecOps best-practices and use scanning tools to identify vulnerabilities in source- or byte-code,dependencies, container layers, and optionally also via dynamic scans against running test systems.

ASVS Chapter: V14 - Configuration Verification Requirements
Cheat Sheet: Vulnerable_Dependency_Management_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Unchecked Deployment** was found **5 times** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Unchecked Deployment risk at **Development Client**: Exploitation likelihood is *Unlikely* with *Medium* impact.
unchecked-deployment@development-client
Unchecked

Unchecked Deployment risk at **github action Build Pipeline**: Exploitation likelihood is *Unlikely* with *Medium* impact.
unchecked-deployment@github-action-build-pipeline
Unchecked

Low Risk Severity

Unchecked Deployment risk at **Amazon ECR Container Registry**: Exploitation likelihood is *Unlikely* with *Low* impact.
unchecked-deployment@amazon-ecr-container-registry
Unchecked

Unchecked Deployment risk at **Nexus Artifact Registry**: Exploitation likelihood is *Unlikely* with *Low* impact.
unchecked-deployment@nexus-artifact-registry
Unchecked

Unchecked Deployment risk at **github Sourcecode Repository**: Exploitation likelihood is *Unlikely* with *Low* impact.
unchecked-deployment@github-sourcecode-repository
Unchecked

Unencrypted Communication: 1 / 1 Risk

Description (Information Disclosure): CWE 319

Due to the confidentiality and/or integrity rating of the data assets transferred over the communication link this connection must be encrypted.

Impact

If this risk is unmitigated, network attackers might be able to to eavesdrop on unencrypted sensitive data sent between components.

Detection Logic

Unencrypted technical communication links of in-scope technical assets (excluding monitoring traffic as well as local-file-access and in-process-library-call) transferring sensitive data.

Risk Rating

Depending on the confidentiality rating of the transferred data-assets either medium or high risk.

False Positives

When all sensitive data sent over the communication link is already fully encrypted on document or data level. Also intra-container/pod communication can be considered false positive when container orchestration platform handles encryption.

Mitigation (Operations): Encryption of Communication Links

Apply transport layer encryption to the communication link.

ASVS Chapter: V9 - Communication Verification Requirements
Cheat Sheet: Transport_Layer_Protection_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Unencrypted Communication** was found **1 time** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.

Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Unencrypted Communication named **Server Traffic** between **Backend** and **Database** transferring authentication data (like credentials, token, session-id, etc.): Exploitation likelihood is *Unlikely* with *High* impact.

unencrypted-communication@backend>server-traffic@backend@database

Unchecked

Unencrypted Technical Assets: 7 / 7 Risks

Description (Information Disclosure): CWE 311

Due to the confidentiality rating of the technical asset itself and/or the processed data assets this technical asset must be encrypted. The risk rating depends on the sensitivity technical asset itself and of the data assets stored.

Impact

If this risk is unmitigated, attackers might be able to access unencrypted data when successfully compromising sensitive components.

Detection Logic

In-scope unencrypted technical assets (excluding reverse-proxy, load-balancer, waf, ids, ips and embedded components like library) storing data assets rated at least as confidential or critical. For technical assets storing data assets rated as strictly-confidential or mission-critical the encryption must be of type data-with-enduser-individual-key.

Risk Rating

Depending on the confidentiality rating of the stored data-assets either medium or high risk.

False Positives

When all sensitive data stored within the asset is already fully encrypted on document or data level.

Mitigation (Operations): Encryption of Technical Asset

Apply encryption to the technical asset.

ASVS Chapter: V6 - Stored Cryptography Verification Requirements
Cheat Sheet: Cryptographic_Storage_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Unencrypted Technical Assets** was found **7 times** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Unencrypted Technical Asset named **Amazon EKS Container Platform**: Exploitation likelihood is *Unlikely* with *High* impact.
unencrypted-asset@amazon-eks-container-platform

Unchecked

Unencrypted Technical Asset named **Backend**: Exploitation likelihood is *Unlikely* with *High* impact.
unencrypted-asset@backend

Unchecked

Unencrypted Technical Asset named **Amazon ECR Container Registry**: Exploitation likelihood is *Unlikely* with *Medium* impact.
unencrypted-asset@amazon-ecr-container-registry

Unchecked

Unencrypted Technical Asset named **Nexus Artifact Registry**: Exploitation likelihood is *Unlikely* with *Medium* impact.
unencrypted-asset@nexus-artifact-registry

Unchecked

Unencrypted Technical Asset named **github Sourcecode Repository**: Exploitation likelihood is *Unlikely* with *Medium* impact.
unencrypted-asset@github-sourcecode-repository

Unchecked

Unencrypted Technical Asset named **github action Build Pipeline**: Exploitation likelihood is *Unlikely* with *Medium* impact.
unencrypted-asset@github-action-build-pipeline

Unchecked

Unencrypted Technical Asset named **Database**: Exploitation likelihood is *Unlikely* with *Medium* impact.
unencrypted-asset@database

Accepted	2023-10-01	Ciro Bologna	XYZ-1234

Risk accepted as tolerable

DoS-risky Access Across Trust-Boundary: 1 / 1 Risk

Description (Denial of Service): CWE 400

Assets accessed across trust boundaries with critical or mission-critical availability rating are more prone to Denial-of-Service (DoS) risks.

Impact

If this risk remains unmitigated, attackers might be able to disturb the availability of important parts of the system.

Detection Logic

In-scope technical assets (excluding load-balancer) with availability rating of critical or higher which have incoming data-flows across a network trust-boundary (excluding devops usage).

Risk Rating

Matching technical assets with availability rating of critical or higher are at low risk. When the availability rating is mission-critical and neither a VPN nor IP filter for the incoming data-flow nor redundancy for the asset is applied, the risk-rating is considered medium.

False Positives

When the accessed target operations are not time- or resource-consuming.

Mitigation (Operations): Anti-DoS Measures

Apply anti-DoS techniques like throttling and/or per-client load blocking with quotas. Also for maintenance access routes consider applying a VPN instead of public reachable interfaces. Generally applying redundancy on the targeted technical asset reduces the risk of DoS.

ASVS Chapter: V1 - Architecture, Design and Threat Modeling Requirements
Cheat Sheet: Denial_of_Service_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **DoS-risky Access Across Trust-Boundary** was found **1 time** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Low Risk Severity

Denial-of-Service risky access of **Backend** by **Frontend** via **User Traffic**: Exploitation likelihood is *Unlikely* with *Low* impact.

dos-risky-access-across-trust-boundary@backend@frontend@frontend>user-traffic

Unchecked

Mixed Targets on Shared Runtime: 1 / 1 Risk

Description (Elevation of Privilege): CWE 1008

Different attacker targets (like frontend and backend/datastore components) should not be running on the same shared (underlying) runtime.

Impact

If this risk is unmitigated, attackers successfully attacking other components of the system might have an easy path towards more valuable targets, as they are running on the same shared runtime.

Detection Logic

Shared runtime running technical assets of different trust-boundaries is at risk. Also mixing backend/datastore with frontend components on the same shared runtime is considered a risk.

Risk Rating

The risk rating (low or medium) depends on the confidentiality, integrity, and availability rating of the technical asset running on the shared runtime.

False Positives

When all assets running on the shared runtime are hardened and protected to the same extend as if all were containing/processing highly sensitive data.

Mitigation (Operations): Runtime Separation

Use separate runtime environments for running different target components or apply similar separation styles to prevent load- or breach-related problems originating from one more attacker-facing asset impacts also the other more critical rated backend/datastore assets.

ASVS Chapter: V1 - Architecture, Design and Threat Modeling Requirements
Cheat Sheet: Attack_Surface_Analysis_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **Mixed Targets on Shared Runtime** was found **1 time** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Low Risk Severity

Mixed Targets on Shared Runtime named **EKS** might enable attackers moving from one less valuable target to a more valuable one: Exploitation likelihood is *Unlikely* with *Low* impact.
mixed-targets-on-shared-runtime@eks

Unchecked

SQL/NoSQL-Injection: 0 / 1 Risk

Description (Tampering): CWE 89

When a database is accessed via database access protocols SQL/NoSQL-Injection risks might arise. The risk rating depends on the sensitivity technical asset itself and of the data assets processed or stored.

Impact

If this risk is unmitigated, attackers might be able to modify SQL/NoSQL queries to steal and modify data and eventually further escalate towards a deeper system penetration via code executions.

Detection Logic

Database accessed via typical database access protocols by in-scope clients.

Risk Rating

The risk rating depends on the sensitivity of the data stored inside the database.

False Positives

Database accesses by queries not consisting of parts controllable by the caller can be considered as false positives after individual review.

Mitigation (Development): SQL/NoSQL-Injection Prevention

Try to use parameter binding to be safe from injection vulnerabilities. When a third-party product is used instead of custom developed software, check if the product applies the proper mitigation and ensure a reasonable patch-level.

ASVS Chapter: V5 - Validation, Sanitization and Encoding Verification Requirements
Cheat Sheet: SQL_Injection_Prevention_Cheat_Sheet

Check

Are recommendations from the linked cheat sheet and referenced ASVS chapter applied?

Risk Findings

The risk **SQL/NoSQL-Injection** was found **1 time** in the analyzed architecture to be potentially possible. Each spot should be checked individually by reviewing the implementation whether all controls have been applied properly in order to mitigate each risk.
Risk finding paragraphs are clickable and link to the corresponding chapter.

Elevated Risk Severity

SQL/NoSQL-Injection risk at **Backend** against database **Database** via **Server Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.

sql-nosql-injection@backend@database@backend>server-traffic

Mitigated	2023-01-10	Ciro Bologna	XYZ-1234

The backend is using hibernate and not manipulating SQL directly

Identified Risks by Technical Asset

In total **61 potential risks** have been identified during the threat modeling process of which **1 are rated as critical**, **0 as high**, **11 as elevated**, **42 as medium**, and **7 as low**.

These risks are distributed across **9 in-scope technical assets**. The following sub-chapters of this section describe each identified risk grouped by technical asset. The RAA value of a technical asset is the calculated "Relative Attacker Attractiveness" value in percent.

Database: 4 / 4 Risks

Description

Some Description

Identified Risks of Asset

Risk finding paragraphs are clickable and link to the corresponding chapter.

Critical Risk Severity

Example Individual Risk at **Some Technical Asset**: Exploitation likelihood is *Likely* with *Medium* impact.

something-strange@database

Unchecked

Elevated Risk Severity

Untrusted Deserialization risk at **Database**: Exploitation likelihood is *Likely* with *High* impact.

untrusted-deserialization@database

Unchecked

Medium Risk Severity

Container Base Image Backdooring risk at **Database**: Exploitation likelihood is *Unlikely* with *Medium* impact.

container-baseimage-backdooring@database

Unchecked

Unencrypted Technical Asset named **Database**: Exploitation likelihood is *Unlikely* with *Medium* impact.

unencrypted-asset@database

Accepted	2023-10-01	Ciro Bologna	XYZ-1234

Risk accepted as tolerable

Asset Information

ID:	database
Type:	process
Usage:	business
RAA:	28 %
Size:	component

Technology:	database
Tags:	mysql
Internet:	false
Machine:	container
Encryption:	none
Multi-Tenant:	false
Redundant:	false
Custom-Developed:	true
Client by Human:	false
Data Processed:	Greetings
Data Stored:	Greetings
Formats Accepted:	Serialization

Asset Rating

Owner:	Some Owner	
Confidentiality:	confidential	(rated 4 in scale of 5)
Integrity:	critical	(rated 4 in scale of 5)
Availability:	critical	(rated 4 in scale of 5)
CIA-Justification:	Some Justification	

Incoming Communication Links: 1

Source technical asset names are clickable and link to the corresponding chapter.

Server Traffic (incoming)

Some Description

Source:	Backend
Protocol:	jdbc
Encrypted:	false
Authentication:	credentials
Authorization:	none
Read-Only:	false
Usage:	business
Tags:	none
VPN:	false
IP-Filtered:	false
Data Received:	Greetings

Data Sent:	Greetings

Amazon EKS Container Platform: 7 / 7 Risks

Description

Amazon EKS Container Platform

Identified Risks of Asset

Risk finding paragraphs are clickable and link to the corresponding chapter.

Elevated Risk Severity

Unguarded Access from Internet of **Amazon EKS Container Platform** by **Development Client** via **Container Platform Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.

unguarded-access-from-internet@amazon-eks-container-platform@development-client@development-client>container-platform-traffic

Unchecked

Missing Hardening risk at **Amazon EKS Container Platform**: Exploitation likelihood is *Likely* with *Medium* impact.

missing-hardening@amazon-eks-container-platform

in Discussion 2023-01-10 Ciro Bologna XYZ-1234
The hardening measures on the docker images are completed, the network ones in progress

Medium Risk Severity

Container Platform Escape risk at **Amazon EKS Container Platform**: Exploitation likelihood is *Unlikely* with *High* impact.

container-platform-escape@amazon-eks-container-platform

Unchecked

Unencrypted Technical Asset named **Amazon EKS Container Platform**: Exploitation likelihood is *Unlikely* with *High* impact.

unencrypted-asset@amazon-eks-container-platform

Unchecked

Missing Network Segmentation to further encapsulate and protect **Amazon EKS Container Platform** against unrelated lower protected assets in the same network segment, which might be easier to compromise by attackers: Exploitation likelihood is *Unlikely* with *Medium* impact.

missing-network-segmentation@amazon-eks-container-platform

Unchecked

Missing Two-Factor Authentication covering communication link **Container Platform Traffic** from **Development Client** to **Amazon EKS Container Platform**: Exploitation likelihood is *Unlikely* with *Medium* impact.

missing-authentication-second-factor@development-client>container-platform-traffic@development-client@amazon-eks-container-platform

Unchecked

Server-Side Request Forgery (SSRF) risk at **Amazon EKS Container Platform** server-side web-requesting the target **Amazon ECR Container Registry** via **Container Platform Pull**: Exploitation likelihood is *Unlikely* with *Medium* impact.

server-side-request-forgery@amazon-eks-container-platform@amazon-ecr-container-registry@amazon-eks-container-platform>container-platform-pull

Unchecked

Asset Information

ID:	amazon-eks-container-platform
Type:	process
Usage:	devops
RAA:	100 %
Size:	system
Technology:	container-platform
Tags:	amazon eks
Internet:	true
Machine:	virtual
Encryption:	none
Multi-Tenant:	false
Redundant:	false
Custom-Developed:	false
Client by Human:	false
Data Processed:	Deployment
Data Stored:	Deployment
Formats Accepted:	File

Asset Rating

Owner:	devops team	
Confidentiality:	confidential	(rated 4 in scale of 5)
Integrity:	mission-critical	(rated 5 in scale of 5)
Availability:	mission-critical	(rated 5 in scale of 5)
CIA-Justification:	Container platform components are rated as 'mission-critical' in terms of integrity and availability, because any malicious modification of it might lead to a backdoored production system.	

Outgoing Communication Links: 1

Target technical asset names are clickable and link to the corresponding chapter.

Container Platform Pull (outgoing)

Container Platform Pull

Target:	Amazon ECR Container Registry
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	technical-user
Read-Only:	true
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Sent:	none
Data Received:	Deployment

Incoming Communication Links: 1

Source technical asset names are clickable and link to the corresponding chapter.

Container Platform Traffic (incoming)

Container Platform Traffic

Source:	Development Client
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	enduser-identity-propagation
Read-Only:	false
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Received:	Deployment
Data Sent:	Deployment

Backend: 9 / 10 Risks

Description

Some Description

Identified Risks of Asset

Risk finding paragraphs are clickable and link to the corresponding chapter.

Elevated Risk Severity

Missing Authentication covering communication link **User Traffic** from **Frontend** to **Backend**: Exploitation likelihood is *Likely* with *Medium* impact.

missing-authentication@frontend>user-traffic@frontend@backend

Unchecked

SQL/NoSQL-Injection risk at **Backend** against database **Database** via **Server Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.

sql-nosql-injection@backend@database@backend>server-traffic

Mitigated 2023-01-10 Ciro Bologna XYZ-1234
The backend is using hibernate and not manipulating SQL directly

Medium Risk Severity

Container Base Image Backdooring risk at **Backend**: Exploitation likelihood is *Unlikely* with *High* impact.

container-baseimage-backdooring@backend

Unchecked

Unencrypted Communication named **Server Traffic** between **Backend** and **Database** transferring authentication data (like credentials, token, session-id, etc.): Exploitation likelihood is *Unlikely* with *High* impact.

unencrypted-communication@backend>server-traffic@backend@database

Unchecked

Unencrypted Technical Asset named **Backend**: Exploitation likelihood is *Unlikely* with *High* impact.

unencrypted-asset@backend

Unchecked

Missing Two-Factor Authentication covering communication link **User Traffic** from **Frontend** to **Backend**: Exploitation likelihood is *Unlikely* with *Medium* impact.

missing-authentication-second-factor@frontend>user-traffic@frontend@backend

Unchecked

Missing Web Application Firewall (WAF) risk at **Backend**: Exploitation likelihood is *Unlikely* with *Medium* impact.

missing-waf@backend

Unchecked

Server-Side Request Forgery (SSRF) risk at **Backend** server-side web-requesting the target **AWS Secret Manager Vault** via **Vault Access (backend)**: Exploitation likelihood is *Unlikely* with *Medium* impact.

server-side-request-forgery@backend@aws-secret-manager-vault@backend>vault-access-backend

Unchecked

Unguarded Access from Internet of **Backend** by **Frontend** via **User Traffic**: Exploitation likelihood is *Very Likely* with *Low* impact.

unguarded-access-from-internet@backend@frontend@frontend>user-traffic

Unchecked

Low Risk Severity

Denial-of-Service risky access of **Backend** by **Frontend** via **User Traffic**: Exploitation likelihood is *Unlikely* with *Low* impact.

dos-risky-access-across-trust-boundary@backend@frontend@frontend>user-traffic

Unchecked

Asset Information

ID:	backend
Type:	process
Usage:	business
RAA:	36 %
Size:	component
Technology:	web-service-rest
Tags:	spring, tomcat
Internet:	false
Machine:	container
Encryption:	none
Multi-Tenant:	false
Redundant:	false
Custom-Developed:	true
Client by Human:	false
Data Processed:	Configuration Secrets, Greetings
Data Stored:	none
Formats Accepted:	JSON

Asset Rating

Owner:	Some Owner	
Confidentiality:	confidential	(rated 4 in scale of 5)
Integrity:	critical	(rated 4 in scale of 5)
Availability:	critical	(rated 4 in scale of 5)
CIA-Justification:	Some Justification	

Outgoing Communication Links: 2

Target technical asset names are clickable and link to the corresponding chapter.

Vault Access (backend) (outgoing)

Vault Access Traffic (by backend)

Target:	AWS Secret Manager Vault
Protocol:	https
Encrypted:	true
Authentication:	externalized
Authorization:	technical-user
Read-Only:	true
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Sent:	none
Data Received:	Configuration Secrets

Server Traffic (outgoing)

Some Description

Target:	Database
Protocol:	jdbc
Encrypted:	false
Authentication:	credentials
Authorization:	none
Read-Only:	false
Usage:	business
Tags:	none
VPN:	false

IP-Filtered:	false
Data Sent:	Greetings
Data Received:	Greetings

Incoming Communication Links: 1

Source technical asset names are clickable and link to the corresponding chapter.

User Traffic (incoming)

Some Description

Source:	Frontend
Protocol:	https
Encrypted:	true
Authentication:	none
Authorization:	none
Read-Only:	false
Usage:	business
Tags:	none
VPN:	false
IP-Filtered:	false
Data Received:	Greetings
Data Sent:	none

Nexus Artifact Registry: 9 / 9 Risks

Description

Nexus Artifact Registry

Identified Risks of Asset

Risk finding paragraphs are clickable and link to the corresponding chapter.

Elevated Risk Severity

Unguarded Access from Internet of **Nexus Artifact Registry** by **Development Client** via **Artifact Registry Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.

unguarded-access-from-internet@nexus-artifact-registry@development-client@development-client>artifact-registry-traffic

Unchecked

Unguarded Access from Internet of **Nexus Artifact Registry** by **github action Build Pipeline** via **Artifact Registry Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.

unguarded-access-from-internet@nexus-artifact-registry@github-action-build-pipeline@github-action-build-pipeline>artifact-registry-traffic

Unchecked

Medium Risk Severity

Code Backdooring risk at **Nexus Artifact Registry**: Exploitation likelihood is *Unlikely* with *High* impact.

code-backdooring@nexus-artifact-registry

Unchecked

Accidental Secret Leak risk at **Nexus Artifact Registry**: Exploitation likelihood is *Unlikely* with *Medium* impact.

accidental-secret-leak@nexus-artifact-registry

Unchecked

Missing Two-Factor Authentication covering communication link **Artifact Registry Traffic** from **Development Client** to **Nexus Artifact Registry**: Exploitation likelihood is *Unlikely* with *Medium* impact.

missing-authentication-second-factor@development-client>artifact-registry-traffic@development-client@nexus-artifact-registry

Unchecked

Unencrypted Technical Asset named **Nexus Artifact Registry**: Exploitation likelihood is *Unlikely* with *Medium* impact.

unencrypted-asset@nexus-artifact-registry

Unchecked

Missing Hardening risk at **Nexus Artifact Registry**: Exploitation likelihood is *Likely* with *Low* impact.

missing-hardening@nexus-artifact-registry

in Discussion 2023-01-10 Ciro Bologna XYZ-1234
The hardening measures on the docker images are completed, the network ones in progress

Low Risk Severity

Missing Network Segmentation to further encapsulate and protect **Nexus Artifact Registry** against unrelated lower protected assets in the same network segment, which might be easier to compromise by attackers: Exploitation likelihood is *Unlikely* with *Low* impact.

missing-network-segmentation@nexus-artifact-registry

Unchecked

Unchecked Deployment risk at **Nexus Artifact Registry**: Exploitation likelihood is *Unlikely* with *Low* impact.

unchecked-deployment@nexus-artifact-registry

Unchecked

Asset Information

ID:	nexus-artifact-registry
Type:	process
Usage:	devops
RAA:	59 %
Size:	service
Technology:	artifact-registry
Tags:	nexus
Internet:	true
Machine:	virtual
Encryption:	none
Multi-Tenant:	false
Redundant:	false
Custom-Developed:	false
Client by Human:	false
Data Processed:	Deployment, Sourcecode
Data Stored:	Deployment, Sourcecode
Formats Accepted:	File

Asset Rating

Owner:	devops team	
Confidentiality:	confidential	(rated 4 in scale of 5)
Integrity:	critical	(rated 4 in scale of 5)
Availability:	important	(rated 3 in scale of 5)
CIA-Justification:	Artifact registry components are at least rated as 'critical' in terms of integrity, because any malicious modification of it might lead to a backdoored production system.	

Incoming Communication Links: 2

Source technical asset names are clickable and link to the corresponding chapter.

Artifact Registry Traffic (incoming)

Artifact Registry Traffic

Source:	github action Build Pipeline
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	technical-user
Read-Only:	false
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Received:	Deployment
Data Sent:	Deployment

Artifact Registry Traffic (incoming)

Artifact Registry Traffic

Source:	Development Client
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	enduser-identity-propagation
Read-Only:	true
Usage:	devops
Tags:	none
VPN:	false

IP-Filtered:	false
Data Received:	none
Data Sent:	Deployment

github Sourcecode Repository: 8 / 8 Risks

Description

github Sourcecode Repository

Identified Risks of Asset

Risk finding paragraphs are clickable and link to the corresponding chapter.

Elevated Risk Severity

Unguarded Access from Internet of **github Sourcecode Repository** by **Development Client** via **Sourcecode Repository Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.

unguarded-access-from-internet@github-sourcecode-repository@development-client@development-client>sourcecode-repository-traffic

Unchecked

Unguarded Access from Internet of **github Sourcecode Repository** by **github action Build Pipeline** via **Sourcecode Repository Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.

unguarded-access-from-internet@github-sourcecode-repository@github-action-build-pipeline@github-action-build-pipeline>sourcecode-repository-traffic

Unchecked

Medium Risk Severity

Code Backdooring risk at **github Sourcecode Repository**: Exploitation likelihood is *Unlikely* with *High* impact.

code-backdooring@github-sourcecode-repository

Unchecked

Accidental Secret Leak risk at **github Sourcecode Repository**: Exploitation likelihood is *Unlikely* with *Medium* impact.

accidental-secret-leak@github-sourcecode-repository

Unchecked

Missing Identity Store in the threat model (referencing asset **github Sourcecode Repository** as an example): Exploitation likelihood is *Unlikely* with *Medium* impact.

missing-identity-store@github-sourcecode-repository

Unchecked

Missing Two-Factor Authentication covering communication link **Sourcecode Repository Traffic** from **Development Client** to **github Sourcecode Repository**: Exploitation likelihood is *Unlikely* with *Medium* impact.

missing-authentication-second-factor@development-client>sourcecode-repository-traffic@development-client@github-sourcecode-repository

Unchecked

Unencrypted Technical Asset named **github Sourcecode Repository**: Exploitation likelihood is *Unlikely* with *Medium* impact.

unencrypted-asset@github-sourcecode-repository

Unchecked

Low Risk Severity

Unchecked Deployment risk at **github Sourcecode Repository**: Exploitation likelihood is *Unlikely* with *Low* impact.

unchecked-deployment@github-sourcecode-repository

Unchecked

Asset Information

ID:	github-sourcecode-repository
Type:	process
Usage:	devops
RAA:	43 %
Size:	service
Technology:	sourcecode-repository
Tags:	github
Internet:	true
Machine:	virtual
Encryption:	none
Multi-Tenant:	false
Redundant:	false
Custom-Developed:	false
Client by Human:	false
Data Processed:	Sourcecode
Data Stored:	Sourcecode
Formats Accepted:	File

Asset Rating

Owner:	devops team	
Confidentiality:	confidential	(rated 4 in scale of 5)
Integrity:	critical	(rated 4 in scale of 5)
Availability:	important	(rated 3 in scale of 5)
CIA-Justification:	Sourcecode processing components are at least rated as 'critical' in terms of integrity, because any malicious modification of it might lead to a	

backdoored production system.

Incoming Communication Links: 2

Source technical asset names are clickable and link to the corresponding chapter.

Sourcecode Repository Traffic (incoming)

Sourcecode Repository Traffic

Source:	github action Build Pipeline
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	technical-user
Read-Only:	true
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Received:	none
Data Sent:	Sourcecode

Sourcecode Repository Traffic (incoming)

Sourcecode Repository Traffic

Source:	Development Client
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	enduser-identity-propagation
Read-Only:	false
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Received:	Sourcecode
Data Sent:	Sourcecode

github action Build Pipeline: 10 / 10 Risks

Description

github action Build Pipeline

Identified Risks of Asset

Risk finding paragraphs are clickable and link to the corresponding chapter.

Elevated Risk Severity

Unguarded Access from Internet of **github action Build Pipeline** by **Development Client** via **Build Pipeline Traffic**: Exploitation likelihood is *Very Likely* with *Medium* impact.

unguarded-access-from-internet@github-action-build-pipeline@development-client@development-client>build-pipeline-traffic

Unchecked

Medium Risk Severity

Code Backdooring risk at **github action Build Pipeline**: Exploitation likelihood is *Unlikely* with *High* impact.

code-backdooring@github-action-build-pipeline

Unchecked

Missing Two-Factor Authentication covering communication link **Build Pipeline Traffic** from **Development Client** to **github action Build Pipeline**: Exploitation likelihood is *Unlikely* with *Medium* impact.

missing-authentication-second-factor@development-client>build-pipeline-traffic@development-client@github-action-build-pipeline

Unchecked

Server-Side Request Forgery (SSRF) risk at **github action Build Pipeline** server-side web-requesting the target **Amazon ECR Container Registry** via **Container Registry Traffic**: Exploitation likelihood is *Unlikely* with *Medium* impact.

server-side-request-forgery@github-action-build-pipeline@amazon-ecr-container-registry@github-action-build-pipeline>container-registry-traffic

Unchecked

Server-Side Request Forgery (SSRF) risk at **github action Build Pipeline** server-side web-requesting the target **Nexus Artifact Registry** via **Artifact Registry Traffic**: Exploitation likelihood is *Unlikely* with *Medium* impact.

server-side-request-forgery@github-action-build-pipeline@nexus-artifact-registry@github-action-build-pipeline>artifact-registry-traffic

Unchecked

Server-Side Request Forgery (SSRF) risk at **github action Build Pipeline** server-side web-requesting the target **github Sourcecode Repository** via **Sourcecode Repository Traffic**: Exploitation likelihood is *Unlikely* with *Medium* impact.

server-side-request-forgery@github-action-build-pipeline@github-sourcecode-repository@github-action-build-pipeline>sourcecode-repository-traffic

Unchecked

Unchecked Deployment risk at **github action Build Pipeline**: Exploitation likelihood is *Unlikely* with *Medium* impact.

unchecked-deployment@github-action-build-pipeline

Unchecked

Unencrypted Technical Asset named **github action Build Pipeline**: Exploitation likelihood is *Unlikely* with *Medium* impact.

unencrypted-asset@github-action-build-pipeline

Unchecked

Missing Hardening risk at **github action Build Pipeline**: Exploitation likelihood is *Likely* with *Low* impact.

missing-hardening@github-action-build-pipeline

in Discussion 2023-01-10 Ciro Bologna XYZ-1234
The hardening measures on the docker images are completed, the network ones in progress

Low Risk Severity

Missing Network Segmentation to further encapsulate and protect **github action Build Pipeline** against unrelated lower protected assets in the same network segment, which might be easier to compromise by attackers: Exploitation likelihood is *Unlikely* with *Low* impact.

missing-network-segmentation@github-action-build-pipeline

Unchecked

Asset Information

ID:	github-action-build-pipeline
Type:	process
Usage:	devops
RAA:	82 %
Size:	service
Technology:	build-pipeline
Tags:	github action
Internet:	true
Machine:	virtual
Encryption:	none
Multi-Tenant:	false
Redundant:	false

Custom-Developed:	false
Client by Human:	false
Data Processed:	Deployment, Sourcecode
Data Stored:	Deployment, Sourcecode
Formats Accepted:	File

Asset Rating

Owner:	devops team	
Confidentiality:	confidential	(rated 4 in scale of 5)
Integrity:	critical	(rated 4 in scale of 5)
Availability:	important	(rated 3 in scale of 5)
CIA-Justification:	Build pipeline components are at least rated as 'critical' in terms of integrity, because any malicious modification of it might lead to a backdoored production system.	

Outgoing Communication Links: 3

Target technical asset names are clickable and link to the corresponding chapter.

Sourcecode Repository Traffic (outgoing)

Sourcecode Repository Traffic

Target:	github Sourcecode Repository
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	technical-user
Read-Only:	true
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Sent:	none
Data Received:	Sourcecode

Container Registry Traffic (outgoing)

Container Registry Traffic

Target:	Amazon ECR Container Registry
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	technical-user
Read-Only:	false
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Sent:	Deployment
Data Received:	Deployment

Artifact Registry Traffic (outgoing)

Artifact Registry Traffic

Target:	Nexus Artifact Registry
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	technical-user
Read-Only:	false
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Sent:	Deployment
Data Received:	Deployment

Incoming Communication Links: 1

Source technical asset names are clickable and link to the corresponding chapter.

Build Pipeline Traffic (incoming)

Build Pipeline Traffic

Source:	Development Client
Protocol:	https
Encrypted:	true
Authentication:	credentials

Authorization:	enduser-identity-propagation
Read-Only:	true
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Received:	none
Data Sent:	Deployment

AWS Secret Manager Vault: 1 / 1 Risk

Description

AWS Secret Manager Vault

Identified Risks of Asset

Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Missing Vault Isolation to further encapsulate and protect vault-related asset **AWS Secret Manager Vault** against unrelated lower protected assets **in the same network segment**, which might be easier to compromise by attackers: Exploitation likelihood is *Unlikely* with *High* impact.

missing-vault-isolation@aws-secret-manager-vault

Unchecked

Asset Information

ID:	aws-secret-manager-vault
Type:	process
Usage:	devops
RAA:	30 %
Size:	service
Technology:	vault
Tags:	aws secret manager
Internet:	false
Machine:	serverless
Encryption:	transparent
Multi-Tenant:	false
Redundant:	false
Custom-Developed:	false
Client by Human:	false
Data Processed:	Configuration Secrets
Data Stored:	none
Formats Accepted:	none of the special data formats accepted

Asset Rating

Owner:		
Confidentiality:	strictly-confidential	(rated 5 in scale of 5)
Integrity:	critical	(rated 4 in scale of 5)
Availability:	critical	(rated 4 in scale of 5)
CIA-Justification:	Vault components are rated as 'strictly-confidential'.	

Incoming Communication Links: 1

Source technical asset names are clickable and link to the corresponding chapter.

Vault Access (backend) (incoming)

Vault Access Traffic (by backend)

Source:	Backend
Protocol:	https
Encrypted:	true
Authentication:	externalized
Authorization:	technical-user
Read-Only:	true
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Received:	none
Data Sent:	Configuration Secrets

Amazon ECR Container Registry: 8 / 8 Risks

Description

Amazon ECR Container Registry

Identified Risks of Asset

Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Code Backdooring risk at **Amazon ECR Container Registry**: Exploitation likelihood is *Unlikely* with *High* impact.

code-backdooring@amazon-ecr-container-registry

Unchecked

Accidental Secret Leak risk at **Amazon ECR Container Registry**: Exploitation likelihood is *Unlikely* with *Medium* impact.

accidental-secret-leak@amazon-ecr-container-registry

Unchecked

Missing Two-Factor Authentication covering communication link **Container Registry Traffic** from **Development Client** to **Amazon ECR Container Registry**: Exploitation likelihood is *Unlikely* with *Medium* impact.

missing-authentication-second-factor@development-client>container-registry-traffic@development-client@amazon-ecr-container-registry

Unchecked

Unencrypted Technical Asset named **Amazon ECR Container Registry**: Exploitation likelihood is *Unlikely* with *Medium* impact.

unencrypted-asset@amazon-ecr-container-registry

Unchecked

Unguarded Access from Internet of **Amazon ECR Container Registry** by **Amazon EKS Container Platform** via **Container Platform Pull**: Exploitation likelihood is *Very Likely* with *Low* impact.

unguarded-access-from-internet@amazon-ecr-container-registry@amazon-eks-container-platform@amazon-eks-container-platform>container-platform-pull

Unchecked

Unguarded Access from Internet of **Amazon ECR Container Registry** by **Development Client** via **Container Registry Traffic**: Exploitation likelihood is *Very Likely* with *Low* impact.

unguarded-access-from-internet@amazon-ecr-container-registry@development-client@development-client>container-registry-traffic

Unchecked

Unguarded Access from Internet of **Amazon ECR Container Registry** by **github action Build Pipeline** via **Container Registry Traffic**: Exploitation likelihood is *Very Likely* with *Low* impact.

unguarded-access-from-internet@amazon-ecr-container-registry@github-action-build-pipeline@github-action-build-pipeline>container-registry-traffic

Unchecked

Low Risk Severity

Unchecked Deployment risk at **Amazon ECR Container Registry**: Exploitation likelihood is *Unlikely* with *Low* impact.

unchecked-deployment@amazon-ecr-container-registry

Unchecked

Asset Information

ID:	amazon-ecr-container-registry
Type:	process
Usage:	devops
RAA:	29 %
Size:	service
Technology:	artifact-registry
Tags:	amazon ecr
Internet:	true
Machine:	virtual
Encryption:	none
Multi-Tenant:	false
Redundant:	false
Custom-Developed:	false
Client by Human:	false
Data Processed:	Deployment
Data Stored:	Deployment
Formats Accepted:	File

Asset Rating

Owner:	devops team	
Confidentiality:	confidential	(rated 4 in scale of 5)
Integrity:	critical	(rated 4 in scale of 5)
Availability:	important	(rated 3 in scale of 5)
CIA-Justification:	Container registry components are at least rated as 'critical' in terms of	

integrity, because any malicious modification of it might lead to a backdoored production system.

Incoming Communication Links: 3

Source technical asset names are clickable and link to the corresponding chapter.

Container Registry Traffic (incoming)

Container Registry Traffic

Source:	github action Build Pipeline
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	technical-user
Read-Only:	false
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Received:	Deployment
Data Sent:	Deployment

Container Registry Traffic (incoming)

Container Registry Traffic

Source:	Development Client
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	enduser-identity-propagation
Read-Only:	false
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Received:	Deployment
Data Sent:	Deployment

Container Platform Pull (incoming)

Container Platform Pull

Source:	Amazon EKS Container Platform
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	technical-user
Read-Only:	true
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Received:	none
Data Sent:	Deployment

Frontend: 1 / 1 Risk

Description

react frontend

Identified Risks of Asset

Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Container Base Image Backdooring risk at **Frontend**: Exploitation likelihood is *Unlikely* with *Medium* impact.

container-baseimage-backdooring@frontend

Unchecked

Asset Information

ID:	frontend
Type:	process
Usage:	business
RAA:	19 %
Size:	application
Technology:	browser
Tags:	nginx, react
Internet:	true
Machine:	container
Encryption:	transparent
Multi-Tenant:	false
Redundant:	false
Custom-Developed:	true
Client by Human:	true
Data Processed:	Greetings
Data Stored:	none
Formats Accepted:	JSON

Asset Rating

Owner:	Some Owner	
Confidentiality:	public	(rated 1 in scale of 5)

Integrity:	critical	(rated 4 in scale of 5)
Availability:	operational	(rated 2 in scale of 5)
CIA-Justification:	Some Justification	

Outgoing Communication Links: 1

Target technical asset names are clickable and link to the corresponding chapter.

User Traffic (outgoing)

Some Description

Target:	Backend
Protocol:	https
Encrypted:	true
Authentication:	none
Authorization:	none
Read-Only:	false
Usage:	business
Tags:	none
VPN:	false
IP-Filtered:	false
Data Sent:	Greetings
Data Received:	none

Development Client: out-of-scope

Description

Development Client

Identified Risks of Asset

Risk finding paragraphs are clickable and link to the corresponding chapter.

Medium Risk Severity

Unchecked Deployment risk at **Development Client**: Exploitation likelihood is *Unlikely* with *Medium* impact.

unchecked-deployment@development-client

Unchecked

Asset Information

ID:	development-client
Type:	external-entity
Usage:	devops
RAA:	out-of-scope
Size:	system
Technology:	devops-client
Tags:	none
Internet:	true
Machine:	physical
Encryption:	none
Multi-Tenant:	false
Redundant:	false
Custom-Developed:	false
Client by Human:	true
Data Processed:	Deployment, Sourcecode
Data Stored:	Deployment, Sourcecode
Formats Accepted:	File

Asset Rating

Owner:	devops team	
Confidentiality:	confidential	(rated 4 in scale of 5)

Integrity:	critical	(rated 4 in scale of 5)
Availability:	important	(rated 3 in scale of 5)
CIA-Justification:	Sourcecode processing components are at least rated as 'critical' in terms of integrity, because any malicious modification of it might lead to a backdoored production system.	

Asset Out-of-Scope Justification

Development client is not directly in-scope of the application.

Outgoing Communication Links: 5

Target technical asset names are clickable and link to the corresponding chapter.

Sourcecode Repository Traffic (outgoing)

Sourcecode Repository Traffic

Target:	github Sourcecode Repository
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	enduser-identity-propagation
Read-Only:	false
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Sent:	Sourcecode
Data Received:	Sourcecode

Container Registry Traffic (outgoing)

Container Registry Traffic

Target:	Amazon ECR Container Registry
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	enduser-identity-propagation

Read-Only:	false
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Sent:	Deployment
Data Received:	Deployment

Container Platform Traffic (outgoing)
Container Platform Traffic

Target:	Amazon EKS Container Platform
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	enduser-identity-propagation
Read-Only:	false
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Sent:	Deployment
Data Received:	Deployment

Build Pipeline Traffic (outgoing)
Build Pipeline Traffic

Target:	github action Build Pipeline
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	enduser-identity-propagation
Read-Only:	true
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Sent:	none
Data Received:	Deployment

Artifact Registry Traffic (outgoing)

Artifact Registry Traffic

Target:	Nexus Artifact Registry
Protocol:	https
Encrypted:	true
Authentication:	credentials
Authorization:	enduser-identity-propagation
Read-Only:	true
Usage:	devops
Tags:	none
VPN:	false
IP-Filtered:	false
Data Sent:	none
Data Received:	Deployment

Identified Data Breach Probabilities by Data Asset

In total **61 potential risks** have been identified during the threat modeling process of which **1 are rated as critical**, **0 as high**, **11 as elevated**, **42 as medium**, and **7 as low**.

These risks are distributed across **4 data assets**. The following sub-chapters of this section describe the derived data breach probabilities grouped by data asset.
Technical asset names and risk IDs are clickable and link to the corresponding chapter.

Configuration Secrets: 15 / 15 Risks

Configuration secrets (like credentials, keys, certificates, etc.) secured and managed by a vault

ID:	configuration-secrets	
Usage:	devops	
Quantity:	very-few	
Tags:	none	
Origin:		
Owner:		
Confidentiality:	strictly-confidential	(rated 5 in scale of 5)
Integrity:	critical	(rated 4 in scale of 5)
Availability:	critical	(rated 4 in scale of 5)
CIA-Justification:	Configuration secrets are rated as being 'strictly-confidential'.	
Processed by:	AWS Secret Manager Vault, Backend	
Stored by:	none	
Sent via:	none	
Received via:	Vault Access (backend)	
Data Breach:	**probable**	
Data Breach Risks:	This data asset has data breach potential because of 15 remaining risks:	

Probable: container-baseimage-backdooring@backend

Probable: container-platform-escape@amazon-eks-container-platform

Probable: missing-cloud-hardening@trusted-boundary

Possible: missing-authentication@frontend>user-traffic@frontend@backend

Possible: missing-authentication-second-factor@frontend>user-traffic@frontend@backend

Possible: server-side-request-forgery@amazon-eks-container-platform@amazon-ecr-container-registry@amazon-eks-container-platform>container-platform-pull

Possible: server-side-request-forgery@backend@aws-secret-manager-vault@backend>vault-access-backend

Possible: server-side-request-forgery@github-action-build-pipeline@amazon-ecr-container-registry@github-action-build-pipeline>container-registry-traffic

Possible: server-side-request-forgery@github-action-build-pipeline@nexus-artifact-registry@github-action-build-pipeline>artifact-registry-traffic

Possible: server-side-request-forgery@github-action-build-pipeline@github-sourcecode-repository@github-action-build-pipeline>sourcecode-repository-traffic

Possible: unguarded-access-from-internet@backend@frontend@frontend>user-traffic

Improbable: missing-vault-isolation@aws-secret-manager-vault

Improbable: missing-waf@backend

Improbable: mixed-targets-on-shared-runtime@eks

Improbable: unencrypted-asset@backend

Deployment: 36 / 36 Risks

Deployment unit being installed/shipped

ID:	deployment
Usage:	devops
Quantity:	very-few
Tags:	none
Origin:	
Owner:	devops team
Confidentiality:	confidential (rated 4 in scale of 5)
Integrity:	critical (rated 4 in scale of 5)
Availability:	important (rated 3 in scale of 5)
CIA-Justification:	Deployment units are at least rated as 'critical' in terms of integrity, because any malicious modification of it might lead to a backdoored production system.
Processed by:	Amazon ECR Container Registry, Amazon EKS Container Platform, Development Client, Nexus Artifact Registry, github action Build Pipeline
Stored by:	Amazon ECR Container Registry, Amazon EKS Container Platform, Development Client, Nexus Artifact Registry, github action Build Pipeline
Sent via:	Container Registry Traffic, Container Registry Traffic, Container Platform Traffic, Artifact Registry Traffic
Received via:	Container Registry Traffic, Container Registry Traffic, Container Platform Traffic, Container Platform Pull, Build Pipeline Traffic, Artifact Registry Traffic, Artifact Registry Traffic
Data Breach:	**probable**
Data Breach Risks:	This data asset has data breach potential because of 36 remaining risks:

Probable: accidental-secret-leak@amazon-ecr-container-registry

Probable: accidental-secret-leak@nexus-artifact-registry

Probable: code-backdooring@amazon-ecr-container-registry

Probable: code-backdooring@nexus-artifact-registry

Probable: code-backdooring@github-action-build-pipeline

Probable: missing-cloud-hardening@trusted-boundary

Possible: missing-authentication-second-factor@development-client>artifact-registry-traffic@development-client@nexus-artifact-registry

Possible: missing-authentication-second-factor@development-client>build-pipeline-traffic@development-client@github-action-build-pipeline

Possible: missing-authentication-second-factor@development-client>container-platform-traffic@development-client@amazon-eks-container-platform

Possible: missing-authentication-second-factor@development-client>container-registry-traffic@development-client@amazon-ecr-container-registry

Possible: server-side-request-forgery@amazon-eks-container-platform@amazon-ecr-container-registry@amazon-eks-container-platform>container-platform-pull

Possible: server-side-request-forgery@backend@aws-secret-manager-vault@backend>vault-access-backend

Possible: server-side-request-forgery@github-action-build-pipeline@amazon-ecr-container-registry@github-action-build-pipeline>container-registry-traffic

Possible: server-side-request-forgery@github-action-build-pipeline@nexus-artifact-registry@github-action-build-pipeline>artifact-registry-traffic

Possible: server-side-request-forgery@github-action-build-pipeline@github-sourcecode-repository@github-action-build-pipeline>sourcecode-repository-traffic

Possible: unchecked-deployment@amazon-ecr-container-registry

Possible: unchecked-deployment@development-client

Possible: unchecked-deployment@nexus-artifact-registry

Possible: unchecked-deployment@github-action-build-pipeline

Possible: unguarded-access-from-internet@amazon-ecr-container-registry@amazon-eks-container-platform@amazon-eks-container-platform>container-platform-pull

Possible: unguarded-access-from-internet@amazon-ecr-container-registry@development-client@development-client>container-registry-traffic

Possible: unguarded-access-from-internet@amazon-ecr-container-registry@github-action-build-pipeline@github-action-build-pipeline>container-registry-traffic

Possible: unguarded-access-from-internet@amazon-eks-container-platform@development-client@development-client>container-platform-traffic

Possible: unguarded-access-from-internet@nexus-artifact-registry@development-client@development-client>artifact-registry-traffic

Possible: unguarded-access-from-internet@nexus-artifact-registry@github-action-build-pipeline@github-action-build-pipeline>artifact-registry-traffic

Possible: unguarded-access-from-internet@github-action-build-pipeline@development-client@development-client>build-pipeline-traffic

Improbable: missing-network-segmentation@amazon-eks-container-platform

Improbable: missing-network-segmentation@nexus-artifact-registry

Improbable: missing-network-segmentation@github-action-build-pipeline

Improbable: unencrypted-asset@amazon-ecr-container-registry

Improbable: unencrypted-asset@amazon-eks-container-platform

Improbable: unencrypted-asset@nexus-artifact-registry

Improbable: unencrypted-asset@github-action-build-pipeline

Improbable: missing-hardening@amazon-eks-container-platform

Improbable: missing-hardening@nexus-artifact-registry

Improbable: missing-hardening@github-action-build-pipeline

Greetings: 20 / 21 Risks

few words

ID:	greetings	
Usage:	business	
Quantity:	many	
Tags:	none	
Origin:	Users	
Owner:	Security Architect	
Confidentiality:	confidential	(rated 4 in scale of 5)
Integrity:	critical	(rated 4 in scale of 5)
Availability:	archive	(rated 1 in scale of 5)
CIA-Justification:	greetings should remain private and should not be disclosed to unauthorized users	
Processed by:	Backend, Database, Frontend	
Stored by:	Database	
Sent via:	User Traffic, Server Traffic	
Received via:	Server Traffic	
Data Breach:	**probable**	
Data Breach Risks:	This data asset has data breach potential because of 20 remaining risks:	

Probable: container-baseimage-backdooring@backend

Probable: container-baseimage-backdooring@database

Probable: container-baseimage-backdooring@frontend

Probable: container-platform-escape@amazon-eks-container-platform

Probable: something-strange@database

Probable: missing-cloud-hardening@trusted-boundary

Probable: untrusted-deserialization@database

Possible: missing-authentication@frontend>user-traffic@frontend@backend

Possible: missing-authentication-second-factor@frontend>user-traffic@frontend@backend

Possible: server-side-request-forgery@amazon-eks-container-platform@amazon-ecr-container-registry@amazon-eks-container-platform>container-platform-pull

Possible: server-side-request-forgery@backend@aws-secret-manager-vault@backend>vault-access-backend

Possible: server-side-request-forgery@github-action-build-pipeline@amazon-ecr-container-registry@github-action-build-pipeline>container-registry-traffic

Possible: server-side-request-forgery@github-action-build-pipeline@nexus-artifact-registry@github-action-build-pipeline>artifact-registry-traffic

Possible: server-side-request-forgery@github-action-build-pipeline@github-sourcecode-repository@github-action-build-pipeline>sourcecode-repository-traffic

Possible: unencrypted-communication@backend>server-traffic@backend@database

Possible: unguarded-access-from-internet@backend@frontend@frontend>user-traffic

Improbable: missing-waf@backend

Improbable: mixed-targets-on-shared-runtime@eks

Improbable: unencrypted-asset@backend

Improbable: unencrypted-asset@database

Sourcecode: 30 / 30 Risks

Sourcecode to build the application components from

ID:	sourcecode	
Usage:	devops	
Quantity:	few	
Tags:	none	
Origin:		
Owner:	devops team	
Confidentiality:	confidential	(rated 4 in scale of 5)
Integrity:	critical	(rated 4 in scale of 5)
Availability:	important	(rated 3 in scale of 5)
CIA-Justification:	Sourcecode is at least rated as 'critical' in terms of integrity, because any malicious modification of it might lead to a backdoored production system.	
Processed by:	Development Client, Nexus Artifact Registry, github Sourcecode Repository, github action Build Pipeline	
Stored by:	Development Client, Nexus Artifact Registry, github Sourcecode Repository, github action Build Pipeline	
Sent via:	Sourcecode Repository Traffic	
Received via:	Sourcecode Repository Traffic, Sourcecode Repository Traffic	
Data Breach:	**probable**	
Data Breach Risks:	This data asset has data breach potential because of 30 remaining risks:	

Probable: accidental-secret-leak@nexus-artifact-registry

Probable: accidental-secret-leak@github-sourcecode-repository

Probable: code-backdooring@nexus-artifact-registry

Probable: code-backdooring@github-sourcecode-repository

Probable: code-backdooring@github-action-build-pipeline

Probable: missing-cloud-hardening@trusted-boundary

Possible: missing-authentication-second-factor@development-client>artifact-registry-traffic@development-client@nexus-artifact-registry

Possible: missing-authentication-second-factor@development-client>build-pipeline-traffic@development-client@github-action-build-pipeline

Possible: missing-authentication-second-factor@development-client>sourcecode-repository-traffic@development-client@github-sourcecode-repository

Possible: server-side-request-forgery@amazon-eks-container-platform@amazon-ecr-container-registry@amazon-eks-container-platform>container-platform-pull

Possible: server-side-request-forgery@backend@aws-secret-manager-vault@backend>vault-access-backend

Possible: server-side-request-forgery@github-action-build-pipeline@amazon-ecr-container-registry@github-action-build-pipeline>container-registry-traffic

Possible: server-side-request-forgery@github-action-build-pipeline@nexus-artifact-registry@github-action-build-pipeline>artifact-registry-traffic

Possible: server-side-request-forgery@github-action-build-pipeline@github-sourcecode-repository@github-action-build-pipeline>sourcecode-repository-traffic

Possible: unchecked-deployment@development-client

Possible: unchecked-deployment@nexus-artifact-registry

Possible: unchecked-deployment@github-sourcecode-repository

Possible: unchecked-deployment@github-action-build-pipeline

Possible: unguarded-access-from-internet@nexus-artifact-registry@development-client@development-client>artifact-registry-traffic

Possible: unguarded-access-from-internet@nexus-artifact-registry@github-action-build-pipeline@github-action-build-pipeline>artifact-registry-traffic

Possible: unguarded-access-from-internet@github-sourcecode-repository@development-client@development-client>sourcecode-repository-traffic

Possible: unguarded-access-from-internet@github-sourcecode-repository@github-action-build-pipeline@github-action-build-pipeline>sourcecode-repository-traffic

Possible: unguarded-access-from-internet@github-action-build-pipeline@development-client@development-client>build-pipeline-traffic

Improbable: missing-network-segmentation@nexus-artifact-registry

Improbable: missing-network-segmentation@github-action-build-pipeline

Improbable: unencrypted-asset@nexus-artifact-registry

Improbable: unencrypted-asset@github-sourcecode-repository

Improbable: unencrypted-asset@github-action-build-pipeline

Improbable: missing-hardening@nexus-artifact-registry

Improbable: missing-hardening@github-action-build-pipeline

Trust Boundaries

In total **1 trust boundaries** has been modeled during the threat modeling process.

Trust Boundary
Some Description

ID:	trusted-boundary
Type:	network-cloud-security-group
Tags:	none
Assets inside:	Amazon ECR Container Registry, Amazon EKS Container Platform, AWS Secret Manager Vault, Backend, Database, github action Build Pipeline, github Sourcecode Repository, Nexus Artifact Registry
Boundaries nested:	none

Shared Runtimes

In total **2 shared runtimes** have been modeled during the threat modeling process.

Amazon EKS Runtime
Amazon EKS Runtime

ID:	amazon-eks-container-runtime
Tags:	amazon eks
Assets running:	none

EKS
AWS containerization

ID:	eks
Tags:	none
Assets running:	Frontend, Backend, Database

Risk Rules Checked by Threagile

Threagile Version: 1.0.0
Threagile Build Timestamp: 20211121124511
Threagile Execution Timestamp: 20231001225711
Model Filename: /app/work/threagile.yaml
Model Hash (SHA256): c5fdd215aaa7a56d14a2d8c352bd0d8a576fcabc976f67cabdae54ab7e424023

Threagile (see https://threagile.io for more details) is an open-source toolkit for agile threat modeling, created by Christian Schneider (https://christian-schneider.net): It allows to model an architecture with its assets in an agile fashion as a YAML file directly inside the IDE. Upon execution of the Threagile toolkit all standard risk rules (as well as individual custom rules if present) are checked against the architecture model. At the time the Threagile toolkit was executed on the model input file the following risk rules were checked:

Some Individual Risk Example
something-strange

Individual Risk Category

STRIDE:	Repudiation
Description:	Some text describing the risk category...
Detection:	Some text describing the detection logic...
Rating:	Some text describing the risk assessment...

Accidental Secret Leak
accidental-secret-leak

STRIDE:	Information Disclosure
Description:	Sourcecode repositories (including their histories) as well as artifact registries can accidentally contain secrets like checked-in or packaged-in passwords, API tokens, certificates, crypto keys, etc.
Detection:	In-scope sourcecode repositories and artifact registries.
Rating:	The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed and stored.

Code Backdooring
code-backdooring

STRIDE:	Tampering
Description:	For each build-pipeline component Code Backdooring risks might arise where attackers compromise the build-pipeline in order to let backdoored artifacts be shipped into production. Aside from direct code backdooring this includes backdooring of dependencies and even of more lower-level build infrastructure, like backdooring compilers (similar to what the XcodeGhost malware did) or dependencies.
Detection:	In-scope development relevant technical assets which are either accessed by out-of-scope unmanaged developer clients and/or are directly accessed by any kind

	of internet-located (non-VPN) component or are themselves directly located on the internet.
Rating:	The risk rating depends on the confidentiality and integrity rating of the code being handled and deployed as well as the placement/calling of this technical asset on/from the internet.

Container Base Image Backdooring

container-baseimage-backdooring

STRIDE:	Tampering
Description:	When a technical asset is built using container technologies, Base Image Backdooring risks might arise where base images and other layers used contain vulnerable components or backdoors.
Detection:	In-scope technical assets running as containers.
Rating:	The risk rating depends on the sensitivity of the technical asset itself and of the data assets.

Container Platform Escape

container-platform-escape

STRIDE:	Elevation of Privilege
Description:	Container platforms are especially interesting targets for attackers as they host big parts of a containerized runtime infrastructure. When not configured and operated with security best practices in mind, attackers might exploit a vulnerability inside an container and escape towards the platform as highly privileged users. These scenarios might give attackers capabilities to attack every other container as owning the container platform (via container escape attacks) equals to owning every container.
Detection:	In-scope container platforms.
Rating:	The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed and stored.

Cross-Site Request Forgery (CSRF)

cross-site-request-forgery

STRIDE:	Spoofing
Description:	When a web application is accessed via web protocols Cross-Site Request Forgery (CSRF) risks might arise.
Detection:	In-scope web applications accessed via typical web access protocols.
Rating:	The risk rating depends on the integrity rating of the data sent across the communication link.

Cross-Site Scripting (XSS)

cross-site-scripting

STRIDE:	Tampering

Description:	For each web application Cross-Site Scripting (XSS) risks might arise. In terms of the overall risk level take other applications running on the same domain into account as well.
Detection:	In-scope web applications.
Rating:	The risk rating depends on the sensitivity of the data processed or stored in the web application.

DoS-risky Access Across Trust-Boundary

dos-risky-access-across-trust-boundary

STRIDE:	Denial of Service
Description:	Assets accessed across trust boundaries with critical or mission-critical availability rating are more prone to Denial-of-Service (DoS) risks.
Detection:	In-scope technical assets (excluding load-balancer) with availability rating of critical or higher which have incoming data-flows across a network trust-boundary (excluding devops usage).
Rating:	Matching technical assets with availability rating of critical or higher are at low risk. When the availability rating is mission-critical and neither a VPN nor IP filter for the incoming data-flow nor redundancy for the asset is applied, the risk-rating is considered medium.

Incomplete Model

incomplete-model

STRIDE:	Information Disclosure
Description:	When the threat model contains unknown technologies or transfers data over unknown protocols, this is an indicator for an incomplete model.
Detection:	All technical assets and communication links with technology type or protocol type specified as unknown.
Rating:	low

LDAP-Injection

ldap-injection

STRIDE:	Tampering
Description:	When an LDAP server is accessed LDAP-Injection risks might arise. The risk rating depends on the sensitivity of the LDAP server itself and of the data assets processed or stored.
Detection:	In-scope clients accessing LDAP servers via typical LDAP access protocols.
Rating:	The risk rating depends on the sensitivity of the LDAP server itself and of the data assets processed or stored.

Missing Authentication

missing-authentication

STRIDE:	Elevation of Privilege

Description:	Technical assets (especially multi-tenant systems) should authenticate incoming requests when the asset processes or stores sensitive data.
Detection:	In-scope technical assets (except load-balancer, reverse-proxy, service-registry, waf, ids, and ips and in-process calls) should authenticate incoming requests when the asset processes or stores sensitive data. This is especially the case for all multi-tenant assets (there even non-sensitive ones).
Rating:	The risk rating (medium or high) depends on the sensitivity of the data sent across the communication link. Monitoring callers are exempted from this risk.

Missing Two-Factor Authentication (2FA)

missing-authentication-second-factor

STRIDE:	Elevation of Privilege
Description:	Technical assets (especially multi-tenant systems) should authenticate incoming requests with two-factor (2FA) authentication when the asset processes or stores highly sensitive data (in terms of confidentiality, integrity, and availability) and is accessed by humans.
Detection:	In-scope technical assets (except load-balancer, reverse-proxy, waf, ids, and ips) should authenticate incoming requests via two-factor authentication (2FA) when the asset processes or stores highly sensitive data (in terms of confidentiality, integrity, and availability) and is accessed by a client used by a human user.
Rating:	medium

Missing Build Infrastructure

missing-build-infrastructure

STRIDE:	Tampering
Description:	The modeled architecture does not contain a build infrastructure (devops-client, sourcecode-repo, build-pipeline, etc.), which might be the risk of a model missing critical assets (and thus not seeing their risks). If the architecture contains custom-developed parts, the pipeline where code gets developed and built needs to be part of the model.
Detection:	Models with in-scope custom-developed parts missing in-scope development (code creation) and build infrastructure components (devops-client, sourcecode-repo, build-pipeline, etc.).
Rating:	The risk rating depends on the highest sensitivity of the in-scope assets running custom-developed parts.

Missing Cloud Hardening

missing-cloud-hardening

STRIDE:	Tampering
Description:	Cloud components should be hardened according to the cloud vendor best practices. This affects their configuration, auditing, and further areas.

Detection:	In-scope cloud components (either residing in cloud trust boundaries or more specifically tagged with cloud provider types).
Rating:	The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed and stored.

Missing File Validation

missing-file-validation

STRIDE:	Spoofing
Description:	When a technical asset accepts files, these input files should be strictly validated about filename and type.
Detection:	In-scope technical assets with custom-developed code accepting file data formats.
Rating:	The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed and stored.

Missing Hardening

missing-hardening

STRIDE:	Tampering
Description:	Technical assets with a Relative Attacker Attractiveness (RAA) value of 55 % or higher should be explicitly hardened taking best practices and vendor hardening guides into account.
Detection:	In-scope technical assets with RAA values of 55 % or higher. Generally for high-value targets like datastores, application servers, identity providers and ERP systems this limit is reduced to 40 %
Rating:	The risk rating depends on the sensitivity of the data processed or stored in the technical asset.

Missing Identity Propagation

missing-identity-propagation

STRIDE:	Elevation of Privilege
Description:	Technical assets (especially multi-tenant systems), which usually process data for endusers should authorize every request based on the identity of the enduser when the data flow is authenticated (i.e. non-public). For DevOps usages at least a technical-user authorization is required.
Detection:	In-scope service-like technical assets which usually process data based on enduser requests, if authenticated (i.e. non-public), should authorize incoming requests based on the propagated enduser identity when their rating is sensitive. This is especially the case for all multi-tenant assets (there even less-sensitive rated ones). DevOps usages are exempted from this risk.
Rating:	The risk rating (medium or high) depends on the confidentiality, integrity, and availability rating of the technical asset.

Missing Identity Provider Isolation

missing-identity-provider-isolation

STRIDE:	Elevation of Privilege
Description:	Highly sensitive identity provider assets and their identity datastores should be isolated from other assets by their own network segmentation trust-boundary (execution-environment boundaries do not count as network isolation).
Detection:	In-scope identity provider assets and their identity datastores when surrounded by other (not identity-related) assets (without a network trust-boundary in-between). This risk is especially prevalent when other non-identity related assets are within the same execution environment (i.e. same database or same application server).
Rating:	Default is high impact. The impact is increased to very-high when the asset missing the trust-boundary protection is rated as strictly-confidential or mission-critical.

Missing Identity Store

missing-identity-store

STRIDE:	Spoofing
Description:	The modeled architecture does not contain an identity store, which might be the risk of a model missing critical assets (and thus not seeing their risks).
Detection:	Models with authenticated data-flows authorized via enduser-identity missing an in-scope identity store.
Rating:	The risk rating depends on the sensitivity of the enduser-identity authorized technical assets and their data assets processed and stored.

Missing Network Segmentation

missing-network-segmentation

STRIDE:	Elevation of Privilege
Description:	Highly sensitive assets and/or datastores residing in the same network segment than other lower sensitive assets (like webservers or content management systems etc.) should be better protected by a network segmentation trust-boundary.
Detection:	In-scope technical assets with high sensitivity and RAA values as well as datastores when surrounded by assets (without a network trust-boundary in-between) which are of type client-system, web-server, web-application, cms, web-service-rest, web-service-soap, build-pipeline, sourcecode-repository, monitoring, or similar and there is no direct connection between these (hence no requirement to be so close to each other).
Rating:	Default is low risk. The risk is increased to medium when the asset missing the trust-boundary protection is rated as strictly-confidential or mission-critical.

Missing Vault (Secret Storage)

missing-vault

STRIDE:	Information Disclosure
Description:	In order to avoid the risk of secret leakage via config files (when attacked through

	vulnerabilities being able to read files like Path-Traversal and others), it is best practice to use a separate hardened process with proper authentication, authorization, and audit logging to access config secrets (like credentials, private keys, client certificates, etc.). This component is usually some kind of Vault.
Detection:	Models without a Vault (Secret Storage).
Rating:	The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed and stored.

Missing Vault Isolation

missing-vault-isolation

STRIDE:	Elevation of Privilege
Description:	Highly sensitive vault assets and their datastores should be isolated from other assets by their own network segmentation trust-boundary (execution-environment boundaries do not count as network isolation).
Detection:	In-scope vault assets when surrounded by other (not vault-related) assets (without a network trust-boundary in-between). This risk is especially prevalent when other non-vault related assets are within the same execution environment (i.e. same database or same application server).
Rating:	Default is medium impact. The impact is increased to high when the asset missing the trust-boundary protection is rated as strictly-confidential or mission-critical.

Missing Web Application Firewall (WAF)

missing-waf

STRIDE:	Tampering
Description:	To have a first line of filtering defense, security architectures with web-services or web-applications should include a WAF in front of them. Even though a WAF is not a replacement for security (all components must be secure even without a WAF) it adds another layer of defense to the overall system by delaying some attacks and having easier attack alerting through it.
Detection:	In-scope web-services and/or web-applications accessed across a network trust boundary not having a Web Application Firewall (WAF) in front of them.
Rating:	The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed and stored.

Mixed Targets on Shared Runtime

mixed-targets-on-shared-runtime

STRIDE:	Elevation of Privilege
Description:	Different attacker targets (like frontend and backend/datastore components) should not be running on the same shared (underlying) runtime.
Detection:	Shared runtime running technical assets of different trust-boundaries is at risk. Also mixing backend/datastore with frontend components on the same shared runtime is

	considered a risk.
Rating:	The risk rating (low or medium) depends on the confidentiality, integrity, and availability rating of the technical asset running on the shared runtime.

Path-Traversal
path-traversal

STRIDE:	Information Disclosure
Description:	When a filesystem is accessed Path-Traversal or Local-File-Inclusion (LFI) risks might arise. The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed or stored.
Detection:	Filesystems accessed by in-scope callers.
Rating:	The risk rating depends on the sensitivity of the data stored inside the technical asset.

Push instead of Pull Deployment
push-instead-of-pull-deployment

STRIDE:	Tampering
Description:	When comparing push-based vs. pull-based deployments from a security perspective, pull-based deployments improve the overall security of the deployment targets. Every exposed interface of a production system to accept a deployment increases the attack surface of the production system, thus a pull-based approach exposes less attack surface relevant interfaces.
Detection:	Models with build pipeline components accessing in-scope targets of deployment (in a non-readonly way) which are not build-related components themselves.
Rating:	The risk rating depends on the highest sensitivity of the deployment targets running custom-developed parts.

Search-Query Injection
search-query-injection

STRIDE:	Tampering
Description:	When a search engine server is accessed Search-Query Injection risks might arise.
Detection:	In-scope clients accessing search engine servers via typical search access protocols.
Rating:	The risk rating depends on the sensitivity of the search engine server itself and of the data assets processed or stored.

Server-Side Request Forgery (SSRF)
server-side-request-forgery

STRIDE:	Information Disclosure
Description:	When a server system (i.e. not a client) is accessing other server systems via typical web protocols Server-Side Request Forgery (SSRF) or Local-File-Inclusion (LFI) or Remote-File-Inclusion (RFI) risks might arise.

Detection:	In-scope non-client systems accessing (using outgoing communication links) targets with either HTTP or HTTPS protocol.
Rating:	The risk rating (low or medium) depends on the sensitivity of the data assets receivable via web protocols from targets within the same network trust-boundary as well on the sensitivity of the data assets receivable via web protocols from the target asset itself. Also for cloud-based environments the exploitation impact is at least medium, as cloud backend services can be attacked via SSRF.

Service Registry Poisoning
service-registry-poisoning

STRIDE:	Spoofing
Description:	When a service registry used for discovery of trusted service endpoints Service Registry Poisoning risks might arise.
Detection:	In-scope service registries.
Rating:	The risk rating depends on the sensitivity of the technical assets accessing the service registry as well as the data assets processed or stored.

SQL/NoSQL-Injection
sql-nosql-injection

STRIDE:	Tampering
Description:	When a database is accessed via database access protocols SQL/NoSQL-Injection risks might arise. The risk rating depends on the sensitivity technical asset itself and of the data assets processed or stored.
Detection:	Database accessed via typical database access protocols by in-scope clients.
Rating:	The risk rating depends on the sensitivity of the data stored inside the database.

Unchecked Deployment
unchecked-deployment

STRIDE:	Tampering
Description:	For each build-pipeline component Unchecked Deployment risks might arise when the build-pipeline does not include established DevSecOps best-practices. DevSecOps best-practices scan as part of CI/CD pipelines for vulnerabilities in source- or byte-code, dependencies, container layers, and dynamically against running test systems. There are several open-source and commercial tools existing in the categories DAST, SAST, and IAST.
Detection:	All development-relevant technical assets.
Rating:	The risk rating depends on the highest rating of the technical assets and data assets processed by deployment-receiving targets.

Unencrypted Technical Assets
unencrypted-asset

STRIDE:	Information Disclosure

Description:	Due to the confidentiality rating of the technical asset itself and/or the processed data assets this technical asset must be encrypted. The risk rating depends on the sensitivity technical asset itself and of the data assets stored.
Detection:	In-scope unencrypted technical assets (excluding reverse-proxy, load-balancer, waf, ids, ips and embedded components like library) storing data assets rated at least as confidential or critical. For technical assets storing data assets rated as strictly-confidential or mission-critical the encryption must be of type data-with-enduser-individual-key.
Rating:	Depending on the confidentiality rating of the stored data-assets either medium or high risk.

Unencrypted Communication
unencrypted-communication

STRIDE:	Information Disclosure
Description:	Due to the confidentiality and/or integrity rating of the data assets transferred over the communication link this connection must be encrypted.
Detection:	Unencrypted technical communication links of in-scope technical assets (excluding monitoring traffic as well as local-file-access and in-process-library-call) transferring sensitive data.
Rating:	Depending on the confidentiality rating of the transferred data-assets either medium or high risk.

Unguarded Access From Internet
unguarded-access-from-internet

STRIDE:	Elevation of Privilege
Description:	Internet-exposed assets must be guarded by a protecting service, application, or reverse-proxy.
Detection:	In-scope technical assets (excluding load-balancer) with confidentiality rating of confidential (or higher) or with integrity rating of critical (or higher) when accessed directly from the internet. All web-server, web-application, reverse-proxy, waf, and gateway assets are exempted from this risk when they do not consist of custom developed code and the data-flow only consists of HTTP or FTP protocols. Access from monitoring systems as well as VPN-protected connections are exempted.
Rating:	The matching technical assets are at low risk. When either the confidentiality rating is strictly-confidential or the integrity rating is mission-critical, the risk-rating is considered medium. For assets with RAA values higher than 40 % the risk-rating increases.

Unguarded Direct Datastore Access
unguarded-direct-datastore-access

STRIDE:	Elevation of Privilege

Description:	Datastores accessed across trust boundaries must be guarded by some protecting service or application.
Detection:	In-scope technical assets of type datastore (except identity-store-ldap when accessed from identity-provider and file-server when accessed via file transfer protocols) with confidentiality rating of confidential (or higher) or with integrity rating of critical (or higher) which have incoming data-flows from assets outside across a network trust-boundary. DevOps config and deployment access is excluded from this risk.
Rating:	The matching technical assets are at low risk. When either the confidentiality rating is strictly-confidential or the integrity rating is mission-critical, the risk-rating is considered medium. For assets with RAA values higher than 40 % the risk-rating increases.

Unnecessary Communication Link

unnecessary-communication-link

STRIDE:	Elevation of Privilege
Description:	When a technical communication link does not send or receive any data assets, this is an indicator for an unnecessary communication link (or for an incomplete model).
Detection:	In-scope technical assets' technical communication links not sending or receiving any data assets.
Rating:	low

Unnecessary Data Asset

unnecessary-data-asset

STRIDE:	Elevation of Privilege
Description:	When a data asset is not processed or stored by any data assets and also not transferred by any communication links, this is an indicator for an unnecessary data asset (or for an incomplete model).
Detection:	Modelled data assets not processed or stored by any data assets and also not transferred by any communication links.
Rating:	low

Unnecessary Data Transfer

unnecessary-data-transfer

STRIDE:	Elevation of Privilege
Description:	When a technical asset sends or receives data assets, which it neither processes or stores this is an indicator for unnecessarily transferred data (or for an incomplete model). When the unnecessarily transferred data assets are sensitive, this poses an unnecessary risk of an increased attack surface.
Detection:	In-scope technical assets sending or receiving sensitive data assets which are neither processed nor stored by the technical asset are flagged with this risk. The

	risk rating (low or medium) depends on the confidentiality, integrity, and availability rating of the technical asset. Monitoring data is exempted from this risk.
Rating:	The risk assessment is depending on the confidentiality and integrity rating of the transferred data asset either low or medium.

Unnecessary Technical Asset
unnecessary-technical-asset

STRIDE:	Elevation of Privilege
Description:	When a technical asset does not process or store any data assets, this is an indicator for an unnecessary technical asset (or for an incomplete model). This is also the case if the asset has no communication links (either outgoing or incoming).
Detection:	Technical assets not processing or storing any data assets.
Rating:	low

Untrusted Deserialization
untrusted-deserialization

STRIDE:	Tampering
Description:	When a technical asset accepts data in a specific serialized form (like Java or .NET serialization), Untrusted Deserialization risks might arise.
Detection:	In-scope technical assets accepting serialization data formats (including EJB and RMI protocols).
Rating:	The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed and stored.

Wrong Communication Link Content
wrong-communication-link-content

STRIDE:	Information Disclosure
Description:	When a communication link is defined as readonly, but does not receive any data asset, or when it is defined as not readonly, but does not send any data asset, it is likely to be a model failure.
Detection:	Communication links with inconsistent data assets being sent/received not matching their readonly flag or otherwise inconsistent protocols not matching the target technology type.
Rating:	low

Wrong Trust Boundary Content
wrong-trust-boundary-content

STRIDE:	Elevation of Privilege
Description:	When a trust boundary of type network-policy-namespace-isolation contains non-container assets it is likely to be a model failure.
Detection:	Trust boundaries which should only contain containers, but have different assets inside.

Rating:	low

XML External Entity (XXE)
xml-external-entity

STRIDE:	Information Disclosure
Description:	When a technical asset accepts data in XML format, XML External Entity (XXE) risks might arise.
Detection:	In-scope technical assets accepting XML data formats.
Rating:	The risk rating depends on the sensitivity of the technical asset itself and of the data assets processed and stored. Also for cloud-based environments the exploitation impact is at least medium, as cloud backend services can be attacked via SSRF (and XXE vulnerabilities are often also SSRF vulnerabilities).

Disclaimer

Ciro Bologna conducted this threat analysis using the open-source Threagile toolkit on the applications and systems that were modeled as of this report's date. Information security threats are continually changing, with new vulnerabilities discovered on a daily basis, and no application can ever be 100% secure no matter how much threat modeling is conducted. It is recommended to execute threat modeling and also penetration testing on a regular basis (for example yearly) to ensure a high ongoing level of security and constantly check for new attack vectors.

This report cannot and does not protect against personal or business loss as the result of use of the applications or systems described. Ciro Bologna and the Threagile toolkit offers no warranties, representations or legal certifications concerning the applications or systems it tests. All software includes defects: nothing in this document is intended to represent or warrant that threat modeling was complete and without error, nor does this document represent or warrant that the architecture analyzed is suitable to task, free of other defects than reported, fully compliant with any industry standards, or fully compatible with any operating system, hardware, or other application. Threat modeling tries to analyze the modeled architecture without having access to a real working system and thus cannot and does not test the implementation for defects and vulnerabilities. These kinds of checks would only be possible with a separate code review and penetration test against a working system and not via a threat model.

By using the resulting information you agree that Ciro Bologna and the Threagile toolkit shall be held harmless in any event.

This report is confidential and intended for internal, confidential use by the client. The recipient is obligated to ensure the highly confidential contents are kept secret. The recipient assumes responsibility for further distribution of this document.

In this particular project, a timebox approach was used to define the analysis effort. This means that the author allotted a prearranged amount of time to identify and document threats. Because of this, there is no guarantee that all possible threats and risks are discovered. Furthermore, the analysis applies to a snapshot of the current state of the modeled architecture (based on the architecture information provided by the customer) at the examination time.

Report Distribution

Distribution of this report (in full or in part like diagrams or risk findings) requires that this disclaimer as well as the chapter about the Threagile toolkit and method used is kept intact as part of the distributed report or referenced from the distributed parts.