

André Moen, Arvid Moemeni, Farhad Mangal and
Patrik Andre Olaussen

Securing API Authentication and Authorisation with Integration of Digital Identities

Bachelor's thesis in DIGSEC
Supervisor: Erjon Zoto and Guoqiang Li
May 2024

André Moen, Arvid Moemeni, Farhad Mangal and
Patrik Andre Olaussen

Securing API Authentication and Authorisation with Integration of Digital Identities

Bachelor's thesis in DIGSEC
Supervisor: Erjon Zoto and Guoqiang Li
May 2024

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology



Securing API Authentication and Authorisation with Integration of Digital Identities

André Moen Arvid Moemeni Farhad Mangal
Patrik Andre Olausen

May 20, 2024

Preface

We extend our gratitude to our supervisors, Erjon Zoto and Guoqiang Li, for their invaluable guidance and unwavering support throughout this project. Their insights and expertise have been fundamental in shaping both the direction and execution of the report.

Special thanks are also due to Rune Moen, who invested his time to review our report and provided excellent feedback that greatly enhanced the quality of our work.

We are equally grateful to our contact persons at NBIM, Stian Hagbø Olsen and Celina Heimdal Brynildsen. Their regular feedback on technical aspects and the structure of our report has been immensely helpful.

Abstract

Title:	Securing API Authentication and Authorisation with Integration of Digital Identities
Date:	May 21, 2024
Participants:	André Moen Arvid Moemeni Farhad Mangal Patrik Andre Olaussen
Supervisors:	Erjon Zoto Guoqiang Li
Employers:	Celina Heimdal Brynildsen and Stian Hagbø Olsen, Norges Bank Investment Management (NBIM)
Keywords:	API Security, Authentication, Authorisation, Digital Identities
Pages:	94 without appendix 187 with appendix
Appendices:	6
Availability:	Open

APIs are essential for modern software applications but are vulnerable to cyber-attacks due to their exposure of application logic and data. This report focuses on enhancing API security by outlining best practices for API authentication and authorisation and integrating digital identities for robust authentication and authorisation.

The report identifies key threats based on a threat model, such as spoofing, tampering, repudiation, information disclosure, denial of service, and elevation of privilege. Mitigations based on industry best practices are outlined to help mitigate the threats in the threat model.

A proof of concept was made to demonstrate how to incorporate digital identities into the authentication and authorisation process. The proof of concept aims to strengthen security by linking API requests to users and applying fine-grained access control based on their digital identities.

By implementing these recommendations, organisations can significantly enhance their API authentication and authorisation processes, ensuring better protection of digital assets and secure API interactions.

Sammendrag

Tittel:	Sikring av API Autentisering og Autorisasjon ved Integrering av Digitale Identiteter.
Dato:	21. mai 2024
Deltakere:	André Moen Arvid Moemeni Farhad Mangal Patrik Andre Olaussen
Veiledere:	Erjon Zoto Guoqiang Li
Arbeidsgivere:	Celina Heimdal Brynildsen og Stian Hagbø Olsen, Norges Bank Investment Management (NBIM)
Nøkkelord:	API Sikkerhet, Autentisering, Autorisasjon, Digitale Identiteter
Sider:	94 uten vedlegg 187 med vedlegg
Vedlegg:	6
Tilgjengelighet:	Åpen

APIer er essensielle for moderne programvareapplikasjoner, men er sårbare for cyberangrep på grunn av deres eksponering av applikasjonslogikk og data. Denne rapporten fokuserer på å forbedre API-sikkerheten ved å samle beste praksis for API-autentisering og autorisasjon, samt integrere digitale identiteter for robust autentisering og autorisasjon.

Rapporten identifiserer nøkkeltrusler basert på en trusselmodell, slik som spoofing, manipulering, fornektelse, informasjonsavsløring, tjenestenekt og privilegiumsøkning. Tiltak basert på bransjens beste praksis er presentert for å bidra til å redusere truslene fra trusselmodellen.

Det ble laget et proof of concept for å demonstrere hvordan man kan integrere digitale identiteter i autentiserings- og autorisasjonsprosessen. Proof of concept har som mål å styrke sikkerheten ved å knytte API-forespørsler til brukere og anvende finkornet tilgangskontroll basert på deres digitale identiteter.

Ved å implementere disse anbefalingene kan organisasjoner vesentlig forbedre sine API-autentiserings- og autorisasjonsprosesser, og sikre bedre beskyttelse av digitale eiendeler og sikre API-interaksjoner.

Contents

Contents	iv
Figures	viii
Tables	x
Code Listings	xi
Acronyms	xii
Glossary	xiv
1 Introduction	1
1.1 Background	1
1.1.1 Problem Area	1
1.1.2 Limitations	2
1.1.3 Task Description	2
1.2 Target Audience	2
1.3 Report Objectives	3
1.4 Group Background	3
1.4.1 What needs to be learned	4
1.5 Framework	4
1.5.1 Project Organisation	4
1.5.2 Timeframes	4
1.5.3 Partners	5
1.6 Methodology	5
1.6.1 Software	6
1.7 Report Layout	6
2 Theory	8
2.1 Introduction	8
2.2 Authentication	8
2.3 Concepts and Systems Related to API Authentication and Author- isation	8
2.3.1 Application Programming Interface	8
2.3.2 Digital Identities	9
2.3.3 Federation	9
2.3.4 Identity and Access Management System	10
2.4 Tokens	10
2.4.1 Access Tokens	11
2.4.2 ID Tokens	11

2.4.3	Bearer Tokens	11
2.4.4	Refresh Tokens	11
2.4.5	JSON Web Token	12
2.5	Authorisation	14
2.5.1	Role-Based Access Control	14
2.5.2	Attribute-Based Access Control	15
2.5.3	Just-in-Time	16
2.6	Authentication and Authorisation Protocols	16
2.6.1	OAuth 2.0	16
2.6.2	OpenID Connect	19
2.6.3	Federation and Single Sign-On	20
2.6.4	Security Assertion Markup Language	20
3	Threat Model	22
3.1	Introduction	22
3.1.1	Threat Model Scenario	23
3.2	Assess Criticality	24
3.2.1	CIA Security goals	24
3.2.2	Risk Appetite	25
3.3	Identify Assets	26
3.3.1	Main Logical Components	26
3.4	Threat Identification	28
3.4.1	STRIDE	28
3.5	Risk Assessment	33
3.5.1	Risk Matrix	33
3.5.2	DREAD	35
3.6	Discussion	42
4	API Security	44
4.1	Introduction	44
4.2	Digital Identities	44
4.3	Securing the API	45
4.3.1	Identity and Access Management System	46
4.3.2	API Gateway	47
4.3.3	Authentication	47
4.3.4	Authorisation	50
4.3.5	Zero Trust	56
4.3.6	Other Considerations	57
4.4	Securing the Digital Identity	58
4.4.1	Multi Factor Authentication	59
4.4.2	Proof of Key Code Exchange	59
4.4.3	Session Management	60
4.5	Compatibility Considerations	60
4.6	Risk Assessment	61
4.6.1	Risk Matrix After Mitigations	61
4.6.2	DREAD After Mitigations	62

4.6.3	Bowtie Modelling	63
4.6.4	Discussion	72
5	Proof of Concept	74
5.1	Introduction	74
5.2	Design Choices	75
5.2.1	Choice of Cloud Platforms	76
5.2.2	API Infrastructure	76
5.2.3	Postman	77
5.3	Implementation	77
5.3.1	Microsoft Entra ID Setup	77
5.3.2	Amazon Web Services Setup	78
5.3.3	Requesting Access Tokens	80
5.3.4	Data Flow	82
6	Discussion	85
6.1	Introduction	85
6.2	Results	85
6.2.1	Learning Goals	85
6.2.2	Effect Goals	86
6.2.3	Result Goals	86
6.3	Alternatives	87
6.4	Sustainability	87
6.5	Use of Artificial Intelligence	88
6.6	Criticism of the Thesis	88
6.7	Evaluation of Group Work	89
7	Conclusion	90
7.1	Introduction	90
7.2	Research Questions	90
7.3	Further Work	92
7.4	Final Thoughts	94
	Bibliography	95
A	Project Plan	103
B	Standard Agreement	120
C	Thesis Descripton	134
D	Timetables	138
D.1	Summary Table	138
D.2	Timetable - André	140
D.3	Timetable - Arvid	142
D.4	Timetable - Farhad	145
D.5	Timetable - Patrik	148
E	Meeting Minutes From Meetings With Stakeholders	151
E.1	17.01.2024	151
E.2	24.01.2024	152
E.3	07.02.2024	153
E.4	28.02.2024	153

E.5	06.03.2024	154
E.6	20.03.2024	154
E.7	03.04.2024	155
E.8	24.04.2024	155
E.9	30.04.2024	156
E.10	08.05.2024	157
E.11	16.05.2024	158
F	Meeting Minutes From Meetings With Supervisors	159
F1	11.01.2024	159
F2	19.01.2024	159
F3	26.01.2024	160
F4	02.02.2024	161
F5	16.02.2024	161
F6	23.02.2024	162
F7	08.03.2024	163
F8	15.03.2024	163
F9	22.03.2024	164
F10	05.04.2024	165
F11	19.04.2024	165
F12	19.04.2024	167
F13	03.05.2024	167
F14	10.05.2024	168

Figures

1.1	Gantt diagram displaying timeframe for the bachelor thesis	5
2.1	Illustration of how an IDP in a federated domain can share user identity information with a trusted relying party [10, section 5] . .	10
2.2	OAuth 2.0 Authorisation Code Grant with PKCE [24, p. 73]	18
2.3	OAuth 2.0 Client Credentials Grant [24, p. 80]	19
3.1	Figure illustrating the used threat modelling method	22
3.2	Illustration of the organisation’s data flow that the threat model is based on	27
4.1	Bowtie diagram showing preventions and recoveries related to spoofing from the STRIDE analysis	64
4.2	Bowtie diagram showing preventions and recoveries related to tampering from the STRIDE analysis	65
4.3	Bowtie diagram showing preventions and recoveries related to reputation from the STRIDE analysis	67
4.4	Bowtie diagram showing preventions and recoveries related to Information disclosure from the STRIDE analysis	68
4.5	Bowtie diagram showing preventions and recoveries related to DoS from the STRIDE analysis	70
4.6	Bowtie diagram showing preventions and recoveries related to elevation of privilege from the STRIDE analysis	71
5.1	Illustration displaying which entities can reach which endpoints in the AWS infrastructure	75
5.2	JWT authoriser for the developer group in AWS	79
5.3	Illustration displaying how to gain access tokens for the developer group in the PoC	81
5.4	Illustration displaying how to gain access tokens for the machine-machine communication in the PoC	82
5.5	Data flow diagram displaying how users from the developer group would access the AWS infrastructure and get access to resources stored in the organisation’s database	82

5.6 Mock data received from the /devprojects endpoint in the PoC . . . 84

Tables

2.1	Registered claims available for JWTs [16]	13
2.2	Overview of attribute types used in ABAC [20]	15
2.3	Parties involved in an OAuth 2.0 flow	17
3.1	Overview of the three CIA triad elements	24
3.2	NTNU Levels of Data Security Requirements [29]	24
3.3	Criteria for assigning probability values in risk matrix [39]	33
3.4	Criteria for assigning consequence values in risk matrix [39]	34
3.5	Description of importance of colours in risk matrix	34
3.6	Risk matrix illustrating consequences versus probability for each scenario before mitigations are implemented	35
3.7	DREAD values for ranking damage potential [40, p. 214]	36
3.8	DREAD values for ranking reproducibility [40, p. 215]	36
3.9	DREAD values for ranking exploitability [40, p. 215]	36
3.10	DREAD values for ranking affected users [40, p. 215]	37
3.11	DREAD values for ranking discoverability [40, p. 215]	37
3.12	Criteria of risk levels assigned to DREAD values [40, p. 216]	37
3.13	DREAD risk assessment of the organisation's API before mitigations are implemented	38
4.1	Digital identities involved in the authentication and authorisation process for an API	45
4.2	Overview of mitigations linked to each relevant scenario from the STRIDE analysis for securing the API	46
4.3	Overview of mitigations linked to each relevant scenario for securing the digital identity	59
4.4	Risk matrix illustrating consequences versus probability for each scenario after mitigations are implemented	61
4.5	DREAD risk assessment of the organisation's API after mitigations are implemented	63
5.1	Overview of which entity in the Microsoft Entra ID Tennat can access which endpoints in the AWS infrastructure	83

Code Listings

2.1	Example of a JWT header	12
2.2	Example of JWT claims	13
2.3	Example of JWT signature with secret as its secret	14
2.4	Example of a complete signed JWT token	14
2.5	Example of how resources can be restricted to Oslo-based users in the Security Group with either Analyst or Administrator role using ABAC	15

Acronyms

- ABAC** Attribute-Based Access Control. 15, 50, 52, 53, 61, 65, 69, 72, 91, 92
- API** Application programming interface. 1–11, 15, 17, 19, 22–32, 34, 39–42, 44, 45, 47, 48, 50, 52–58, 60–62, 64–67, 69, 71–74, 76, 77, 82, 84–88, 90–94
- AWS** Amazon Web Services. 2–4, 6, 26–28, 47, 51, 57, 58, 75–78, 80, 82–84, 86–88, 90–93
- CA** Conditional Access. 56, 57, 59, 61, 62, 65, 69, 70, 72, 75, 78, 83, 91, 92
- DDoS** Distributed denial-of-service. 58
- DoS** Denial-of-service. 31, 32, 41, 42, 58, 70, 71, 91
- ECDSA** Elliptic Curve Digital Signature Algorithm. 12, 49
- GUI** Graphical User Interface. 4
- HMAC** Hash-based Message Authentication Code. 12, 13, 49, 55
- IaC** Infrastructure as code. 76, 93
- IoT** Internet of Things. 2, 17
- JIT** Just-in-Time. 16, 50, 53, 69, 72, 91, 92
- JSON** JavaScript Object Notation. 12
- JWT** JSON Web Token. 5, 11–14, 19, 27, 28, 42, 49, 50, 55, 62, 64, 76, 77, 79, 83, 87, 91–93
- MFA** Multi-Factor Authentication. 8, 44, 48, 56, 58–61, 91
- MITM** Man in the Middle. 31, 41, 58, 60, 69, 91
- NBIM** Norges Bank Investment Management. 1–5, 76

- NIST** National Institute of Standards and Technology. 5, 6, 47, 52, 53, 55, 59
- NSA** National Security Agency. 5, 53, 56, 58, 59
- NTNU** Norwegian University of Science and Technology. 3, 5, 24, 78
- OAuth 2.0** Open Authorisation 2.0. 5, 16–20, 27, 48, 50, 54, 59, 65, 66, 72, 77, 81, 86, 87, 90–92
- OIDC** Open ID Connect. 5, 9, 11, 16, 19, 20, 27, 47, 48, 64, 66, 67, 86, 87, 91, 92
- OWASP** Open Web Application Security Project. 44, 50, 78
- PKCE** Proof Key for Code Exchange. 18, 48, 54, 58–60, 62, 66, 67, 81, 91–93
- PoC** Proof of Concept. 1–4, 6, 7, 74–78, 84, 86–89, 92–94
- PoLP** Principle of Least Privilege. 47, 50–52, 62, 65, 66, 68, 69, 72, 90–92
- RBAC** Role-Based Access Control. 14, 15, 50–52, 61, 65, 69, 72, 75, 91, 92
- SAML** Security Assertion Markup Language. 9, 16, 20, 21, 47, 50, 55, 61, 65–67, 69, 72, 86, 87, 90–92
- SDG** UN Sustainable Development Goals. 87
- SSO** Single Sign On. 10, 20, 46, 47, 64, 92
- WAF** Web Application Firewall. 26, 28, 30, 32, 39, 40, 42, 50, 51, 58, 61, 62, 66, 69–72, 76–78, 83, 87, 90, 91

Glossary

Amazon Cognito Amazon Cognito is a fully managed identity and user management service provided by Amazon Web Services. Its primary purpose is to help developers add authentication, authorisation, and user management to their applications quickly and securely. 2, 87, 92

API Gateway An API Gateway is a server that acts as an intermediary between clients and backend services. It handles all the tasks involved in accepting and processing concurrent API calls, including traffic management, authorisation, access control, monitoring, and API version management.. 27–32, 39, 40, 42, 47, 48, 62, 64, 68, 70–72, 76, 77, 79, 83, 87, 90, 91

API Keys An API key is a unique identifier used to authenticate and authorise access to an API. They help track and control how the API is being used, typically by associating the key with a set of access rights and limits. 45, 53, 54, 58, 65, 67, 90

Back-channel Back-channel communication involves the indirect, often informal exchange of information during a communication process, without using redirects through an intermediary such as a browser text. 9, 17, 20, 48, 54, 57

Base64 Base64 encoding is a method used to encode binary data into ASCII string formats, making the data easier to transport. This is, however, not an encryption method but rather a way to ensure compatibility across different media that might not support binary data. 10, 13, 14

Content Delivery Network A Content Delivery Network (CDN) is a distributed server system that delivers web content to users based on their geographic location, aiming to increase the speed and efficiency of web page loading. 26, 32, 70

Cryptographic Cryptographic refers to anything related to cryptography, which is the science and practice of secure communication techniques. Cryptography involves using mathematical algorithms and techniques to encrypt information, making it unreadable to unauthorised parties. 11, 18, 49

DREAD DREAD is a model to prioritise threats. DREAD stands for Damage, Reproducibility, Exploitability, Affected users, and Discoverability. 23, 25, 26, 35, 37, 38, 40, 42, 43, 62, 72, 73, 88, 89

Front-channel Front-channel communication refers to the direct and explicit exchange of information or messages in a communication process, typically accomplished by appending HTTP query parameters to URLs. 17

GitHub GitHub is a web-based platform and service that provides a central place for software developers to collaborate on, manage, and version control their code repositories. 3, 75, 79, 84, 86

Identity and Access Management System An Identity and Access Management (IAM) system is a framework of policies and technologies that ensures the appropriate individuals in an organisation can access the resources they need to perform their duties. IAM systems help organisations manage user identities, authenticate users, and authorise users to access specific resources or data. 10, 16, 20, 26, 46, 47, 51, 62, 64, 66, 69, 72, 76, 86

Identity Provider An Identity Provider (IDP) is a service that manages and authenticates user identities within a system or application. Its primary function is to verify the identity of users trying to access a particular resource or service. 2, 9, 20, 21, 26, 28, 47, 48, 50, 55, 69, 76, 87, 92, 93

Microsoft Entra ID Microsoft Entra ID is a family of identity and access products from Microsoft designed to secure access for every user, application, and device. It is a reimagining of Microsoft's Active Directory. 2, 4, 6, 26–30, 39, 42, 47, 75–77, 79, 80, 82, 83, 86–88, 91–93

Risk-Based Authentication Risk-based authentication (RBA) is a security measure that evaluates the risk level of user access requests based on factors such as user behaviour, device location, and access time. This method dynamically adjusts authentication requirements; if the risk is higher, it may prompt additional verification. 47, 48, 61, 64, 90, 92

Scrumban Scrumban is a hybrid approach that combines elements of both Scrum and Kanban methodologies. It's primarily used in software development and project management to improve workflow and team productivity. 3, 85, 89

Service Provider A Service Provider (SP) is an entity that provides a service, resource, or application to users. 20, 21, 50, 55, 69

Single Page Application A Single Page Application (SPA) is a website without a back end where the logic resides in the browser. They rely on JavaScript to

communicate with APIs and, therefore, cannot hold a secret hidden from users. 48

STRIDE STRIDE is one of the most recognised frameworks to identify and categorise potential threats to a system. STRIDE categorises threats into six primary types. S - Spoofing, T - Tampering, R - Repudiation, I - Information Disclosure, D - Denial of Service and E - Elevation of Privileges. 23, 28, 29, 33–35, 63

Transport Layer Security Transport layer security (TLS) is a cryptographic protocol to secure communication over a computer network. 49, 58

Zero Trust Is a cybersecurity strategy that requires continuously verifying the legitimacy of users, devices, and network requests, allowing access only after ensuring they meet the organisation's security criteria. 47, 56, 62, 65, 66, 69, 72, 91, 92

1 Introduction

1.1 Background

The group's assignment has been given by Norges Bank Investment Management (NBIM). NBIM is responsible for assuring long-term administration of the profits from Norway's oil and gas resources. The official name of the fund is "Statens pensjonsfond utland". The fund has become one of the world's largest, on average, it has ownership in 1.5 per cent of all listed companies globally [1].

In the digital age, Application programming interface (API)s have emerged as the backbone of internet connectivity and communication. Enabling seamless interactions between different software applications, APIs are integral to the operation of web services, cloud technologies, and mobile applications. They constitute 83 per cent [2] of internet traffic, highlighting their critical role in the digital ecosystem.

However, this substantial volume of API traffic also presents significant security challenges. APIs exposes application logic and sensitive data, making them attractive targets for cyberattacks [3]. As the conduits through which different software services communicate, APIs, if left unprotected, can become the weakest link in an organisation's cybersecurity armour.

As the custodian of a significant portion of Norway's wealth, NBIM must maintain impeccable cybersecurity practices, a mandate that includes rigorous API security. In line with this imperative, the assignment involves developing a comprehensive report that outlines best practices for securing authentication and authorisation of APIs, coupled with a Proof of Concept (PoC) for integrating digital identities into these APIs. The absence of robust identity verification can lead to breaches and unauthorised access to sensitive data and services. By incorporating digital identities, organisations can establish a strong link between API requests and legitimate users, apply fine-grained access control, and prevent fraudulent and malicious access attempts [4].

1.1.1 Problem Area

Digital security practices can be a modern business's greatest defence or biggest weakness. 73 per cent of all internet traffic is made from malicious sources and

bots [5], and they are all searching for that one mistake in an organisation's security configuration. A foundational element of innovation in today's app-driven world is APIs. From banks, retail and transportation to Internet of Things (IoT), autonomous vehicles and smart cities, APIs are a critical part of modern mobile, software as a service (SaaS), and web applications can be found in customer-facing, partner-facing and internal applications [3]. This report will explain how to properly authenticate and authorise individuals accessing APIs.

1.1.2 Limitations

The report's main section is constrained by the scenarios outlined in the threat model. The mitigations outlined in the report have been written with a system based on centralised identity in mind. The group is not going to deliver an API that is ready to be deployed, only a working PoC for incorporating digital identities into the API. The group have, therefore, not been testing the API with already established infrastructure. The testing is done after the group decides on the specifications. The PoC is based on modules found in Amazon Web Services (AWS), as the workload of exploring other cloud service provider options would be too great for the scope of the task. The only exception to this is the use of an Identity Provider (IDP), and the group used Microsoft's IDP solution, Microsoft Entra ID, instead of AWS's IDP solution Amazon Cognito.

1.1.3 Task Description

The purpose of the report is to outline best practices for API authentication and authorisation, as well as to provide a technical PoC demonstrating how digital identities could be integrated into APIs. The group's focus was on validating user and system identities and implementing precise authorisation controls. The report took into account the sensitivity of the data that the API would provide access to. It will recommend appropriate security controls for authentication and authorisation based on a threat model for the API, using best practices.

1.2 Target Audience

The report is designed with a broad perspective, aiming to address the needs and concerns of various organisations interested in API security, including but not limited to NBIM, with a focus on API authentication and authorisation best practices. While initially guided by NBIM's requirements, the findings and the PoC provided are broadly applicable, enhancing the security and efficiency of API systems for a diverse audience.

1.3 Report Objectives

Research Questions

There are two research questions that this thesis will attempt to answer:

- What are the main threats to APIs and digital identities during authentication and authorisation, and how can they be mitigated?
- What are the current industry standard protocols and technologies regarding API authentication and authorisation?

The group set several goals they will attempt to achieve during the project. These are divided into three categories: effect, result and learning goals.

Effect Goals

- Receive better knowledge about best practices for authentication and authorisation of APIs using digital identities.
- Improve security measures and practices for API usage at NBIM and other organisations that want to secure their APIs.
- Give a better overview of how to secure APIs and how digital identities can be implemented into an API.

Result Goals

- Deliver a report that can be used to improve API security.
- Deliver a PoC of the group's findings, which will be an API showcasing how to incorporate digital identities into the authentication and authorisation process.

Learning Goals

- Get familiar with scrumban.
- Gain better knowledge of industry practices for authentication and authorisation to APIs using digital identities.
- Learn how to use cloud computing tools.
- Learn to use GitHub to host source code and AWS as a deployment environment.
- Receive a good grade for the bachelor thesis.

1.4 Group Background

The group's members have completed nearly three years of education in Digital Infrastructure and Cyber Security at the Norwegian University of Science and Technology (NTNU) Gjøvik. Their studies have equipped them with knowledge in

several areas, including programming, risk management, cyber security, networking, and teamwork. This skill set ensures the group is well-prepared to tackle the challenges presented by NBIM, leveraging their understanding of IT principles.

1.4.1 What needs to be learned

The digital infrastructure and cybersecurity curriculum offered limited exposure to APIs, necessitating a focused effort to learn both the fundamentals and more advanced aspects of API usage, specifically in the context of authentication and authorisation standards, methods, and protocols. The project required developing a PoC and implementing best practices for authenticating and authorising users to an API.

Moreover, the PoC was designed using cloud platforms such as AWS and Microsoft Entra ID, tools with which the group previously had no experience. This aspect of the project prompted the group to delve into the workings of cloud services, learning the general principles and the specific functionalities of AWS and Microsoft Entra ID. The group adopted an infrastructure-as-code approach to streamline the development process and minimise potential configuration errors, utilising CloudFormation templates. This methodology saved time by eliminating the need for manual configurations through a Graphical User Interface (GUI) and enhanced the reliability of testing procedures by minimising human error.

1.5 Framework

1.5.1 Project Organisation

The Scrum framework was adopted as the primary project organisation to facilitate effective self-organisation and streamline project delivery. The agile approach involves daily standups, where group members coordinate ongoing tasks and set short-term goals. In practice, this meant that each Monday, the group agreed on what tasks should be finished during the week and who should do which task, as well as daily standup meetings to monitor progress. If a task proved too large, work would continue into the next week, or other group members would be reassigned to help. Additionally, the group had in-depth discussions of the work before drafts were submitted to supervisors and stakeholders. Transparency and engagement with both stakeholders and supervisors were ensured through weekly meetings.

1.5.2 Timeframes

- The group will work on the bachelor project from 04/01/24 to 21/05/24 and deliver the report on 21/05/24.
- The group plans to deliver a first draft on 30/03/24.

- A second draft is planned to be delivered on 03/05/24 to ensure the supervisors have sufficient time to give feedback and the group has time to correct weak points in the report before the final deadline.
- The group will present their findings on the 5th or 6th of June.

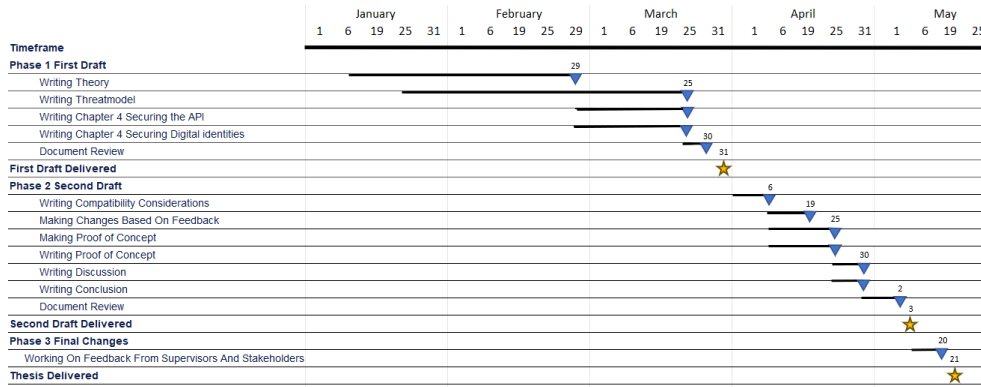


Figure 1.1: Gantt diagram displaying timeframe for the bachelor thesis

1.5.3 Partners

The group’s partners for this project are supervisors from NTNU and stakeholders from NBIM. There will be weekly meetings with both parties unless it is deemed unnecessary. The supervisors will provide feedback on the report’s layout and structure. The stakeholders will provide feedback on the content, ensuring it stays within their vision of what the report should encompass.

1.6 Methodology

The group’s sources stem from different professional institutions and organisations. For papers regarding the subject of API authentication and authorisation, Google Scholar was utilised. The group used keywords such as "API", "Digital Identities", "Authentication", "Authorisation", and others which could lead to answers to their research questions. Papers that were released after 2019 were the goal to ensure the information is up to date. If a match were found, a group member would read the abstract and check the chapters to assess if it was a usable source.

When questions arose about how different protocols and technologies worked and how they were best utilised, RFCs from the Internet Engineering Task Force (IETF) and documents from National Institute of Standards and Technology (NIST) and the National Security Agency (NSA) were used. These were chosen because of their high trust and reliability in cybersecurity. The home websites for specific protocols, such as Open Authorisation 2.0 (OAuth 2.0), Open ID Connect (OIDC), and JSON Web Tokens (JWTs), were also used for gathering information. These

were chosen because the group believed there aren't any better sources on how something works than the people who created it.

The same is true about AWS and Microsoft Entra ID. The group's questions regarding these technologies when creating the PoC were answered by reading documentation and articles written by Amazon and Microsoft.

Other sources are also cited. These are mainly blog posts and articles written by cybersecurity professionals regarding smaller questions that arose during research. The subjects these sources are answering were often too small or too niche to have a dedicated publication by the IETF or NIST, so the group resorted to using these alternative sources.

1.6.1 Software

Throughout the project, the group utilised several software products to improve the quality of the thesis work. Here is a list of the software products used:

- ChatGPT: Primarily used to rephrase and find flaws in the text.
- Grammarly: Used for correcting grammar and sentence structure.
- Overleaf: LaTeX editor used for writing the thesis.
- Excel Timesheet: Used to track hourly work and achieve transparency on time used on tasks.
- GitHub: Containing a repository with all code utilised in the PoC, and the kanban board.

1.7 Report Layout

Throughout the report, there are hyperlinks to different figures, tables, chapters, abbreviations, acronyms, sources and sections in chapters; these can be clicked and redirects to the relevant place. Every time a scenario from the threat model is mentioned, it will have a hyperlink redirecting to the relevant scenario, e.g. S1.

Chapter 1 - Introduction Introduction to the thesis.

Chapter 2 - Theory Relevant theory to understand the content presented in the main part of the report.

Chapter 3 - Threat Model A threat model describing a fictional organisation's API and relevant threats to them. The rest of the report will use the threat model as the scope for what will be covered.

Chapter 4 - API Security An overview of various technologies and tools will be presented to assist in reducing the risks discussed in the threat model chapter. In addition, a risk matrix will be presented, along with bowtie models that illustrate the threats to the API after the mitigations discussed in this chapter have been implemented.

Chapter 5 - Proof of Concept The PoC goes over the group's practical implementation of incorporating digital identities into the API discussed in the threat model.

Chapter 6 - Discussion Discusses choices made throughout the project, alternatives, weak points in the thesis, use of AI and the thesis contribution to sustainability.

Chapter 7 - Conclusion Gives a conclusion to the thesis findings and further work that can be done.

2 Theory

2.1 Introduction

This chapter provides an overview of key concepts and technologies for understanding the report's later sections. Specifically, this chapter will cover various components, protocols, methods, and tools for implementing API security, focusing on authentication and authorisation.

2.2 Authentication

"Authentication is the process of identifying a user, a system, or a thing in a unique manner to prove that it is the one who it claims to be." [6, p. 59]. In authentication, there are three factors: something the user knows (e.g. a password), something the user has (e.g. an authenticator app on a phone) and something the user is (e.g. biometrics like a fingerprint). Only using one of these is called single-factor authentication, and using more is referred to as Multi-Factor Authentication (MFA) [7]. Using multiple of the same factors does not make it MFA; it needs to be at least two different factors to be counted as MFA.

2.3 Concepts and Systems Related to API Authentication and Authorisation

Concepts and systems related to API authentication and authorisation must be established before the actual authentication and authorisation process can begin. These components encompass the API itself, the individuals making the requests, and the systems that store information about access permissions and give access.

2.3.1 Application Programming Interface

APIs are a set of rules, protocols, and tools that allow different software applications to communicate and interact with each other. Essentially, they define how different components of software systems can interact and exchange data.

APIs are commonly used in software development for various purposes, such as enabling communication between different application modules, integrating third-party services into an application, or allowing other applications to share data and functionality.

In web development, APIs are often used to enable communication between a client-side application and a server-side application or service. This allows developers to access specific functionality or data the server-side application provides in a structured and standardised way.

APIs can be public, meaning they are openly accessible to users, or private, where access is restricted to authorised users or applications. They can also require authentication and authorisation mechanisms to control access to sensitive data or functionality [8].

2.3.2 Digital Identities

Digital identities are fundamental to digital interactions, representing the unique expression of entities such as individuals, organisations, or devices within a specified namespace. These identities are crafted from attributes and data ranging from personal information such as names and email addresses to dynamic aspects consisting of digital behavioural patterns, device usage, IP addresses and location [9, p. 131]. Digital identities are crucial for identifying and distinguishing users in a digital ecosystem and serve as a critical component in authentication and authorisation.

2.3.3 Federation

Federation involves the sharing of information between different trust domains. This information could be about users, authenticators, identity assertions and authorisation decisions [7]. Some protocols used for assertions with federation are Security Assertion Markup Language (SAML) and OIDC. Figure 2.1 shows an example of how an IDP in a federated domain can share user identity information with a trusted relying party (RP) to allow a client access to resources held by the RP. In this way, the user only needs to authenticate themselves once with the IDP and then get access to multiple resources being held in the trusted domain. The communication between the IDP and the RP is often done through a secured Back-channel.

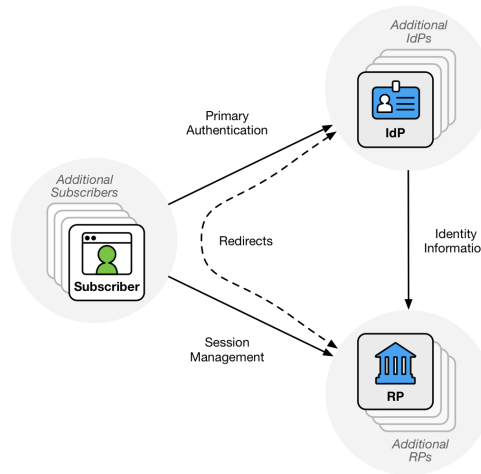


Figure 2.1: Illustration of how an IDP in a federated domain can share user identity information with a trusted relying party [10, section 5]

2.3.4 Identity and Access Management System

An Identity and Access Management (IAM) system is a framework of policies and technologies that ensures the appropriate individuals in an organisation can access the resources needed to perform their duties. IAM systems help organisations manage user identities and authenticate and authorise users to access specific resources or data [11]. Through digital identities, IAM systems can authenticate users against the unique attributes and data that constitute a digital identity. Following authentication, the IAM framework proceeds to authorise, determining access and permissions based on predefined policies and roles.

2.4 Tokens

In API security, tokens are essential for secure interactions between software systems and users. These tokens, varying in form from base64 encoded strings to random strings, serve as temporary access credentials. Unlike traditional authentication methods requiring username-password combinations, tokens enhance security by minimising direct credential exposure and reducing verification overhead [12, p. 102]. There exist several different types of tokens, each serving its purpose. Some are access tokens, bearer tokens, ID tokens, and refresh tokens.

Token-based authentication simplifies the process: Users obtain a token through an authorisation endpoint, which they use for subsequent resource access requests. This approach eliminates the need for repeated credential transmission, conserving resources and bolstering security [12, p. 109]. This method supports various authentication mechanisms, including those without traditional password sign-ins, like Single Sign On (SSO), highlighting the versatility and efficiency of token-based authentication in modern API interactions [12, p. 109].

2.4.1 Access Tokens

Access tokens serve as digital credentials, granting temporary authorisation for accessing specific resources or performing certain actions within an API. Access tokens are also used in token-based authentication schemes to securely validate user identity and authorise secure interactions with protected resources. These tokens are often short-lived and can be configured with specific permissions, expiration times, and scopes to control access levels. To obtain an access token, users authenticate to the service's authorisation server and receive an access token with permissions based on what the user should have access to [12, p. 219].

2.4.2 ID Tokens

ID tokens are primarily used to identify the authenticated user or application. ID tokens can contain claims about the authenticated entity, such as their unique identifier, display name, email address, and other relevant information. ID tokens are crucial in providing context about the authenticated user or application to the relying party, enabling personalised experiences and tailored access control. ID tokens are generated by the OIDC protocol and must be in a JWT format [13].

2.4.3 Bearer Tokens

A bearer token is an access token commonly used in authentication mechanisms within API ecosystems. Unlike other types of tokens, such as ID tokens, bearer tokens do not inherently contain information about the authenticated user or application. Instead, these tokens serve as proof of authentication, granting access to protected resources based solely on the token's possession [12, p. 160].

One of the key characteristics of bearer tokens is their simplicity and flexibility. Complex cryptographic operations are not required for validation, as the token itself serves as the sole credential for authentication. However, this simplicity also means that bearer tokens must be handled securely to prevent unauthorised access. If a bearer token is intercepted or stolen, it can be used by an attacker to gain unrestricted access to the associated resources. To mitigate the risk of misuse, bearer tokens are often short-lived and may be revoked or invalidated after a certain period or when specific conditions are met.

2.4.4 Refresh Tokens

Tokens usually have a short lifetime to prevent being hijacked. When a user wants a new token, credentials have to be re-authenticated. Short-lived tokens that require the user to frequently authenticate create a nuisance for the user. To avoid this problem, refresh tokens are sent with the token obtained from the authorisation server. With refresh tokens, the user can present it to the authorisation server to receive a new token, such as a new access token. Refresh tokens usually have a longer lifetime than other tokens [14].

2.4.5 JSON Web Token

JWT is a standardised format (specified in RFC 7519¹) for transmitting information securely between parties as a JavaScript Object Notation (JSON) object. This format is compact and self-contained, making it easily transferable and verifiable. The information in a JWT can be trusted as it is digitally signed. JWTs can be signed using either a secret key, using the Hash-based Message Authentication Code (HMAC) algorithm, or using a public/private key pair, using RSA or Elliptic Curve Digital Signature Algorithm (ECDSA) [15]. One of the key advantages of JWTs is that they are self-contained, meaning all the information needed for validation is contained within the token itself. This reduces the need for server-side storage and database lookups, making JWTs efficient for distributed systems and stateless authentication mechanisms [16].

JWTs are structured into three parts separated by periods: the header, the payload, and the signature. The header typically contains metadata about the token, such as the type of token and the hashing algorithm used to generate the signature [16]. Code listing 2.1 shows an example of a JWT header.

Code listing 2.1: Example of a JWT header

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

The payload contains registered and custom claims, statements about an entity, and additional data that the authorisation server might need to perform authorisation. These claims can include information such as the user's ID, role, or any other relevant data. Registered claims are predefined fields with a specific meaning and are standardised across implementations. The registered claims are shown in Table 2.1. It is recommended that the registered claims be used so the token makes sense [16].

¹<https://datatracker.ietf.org/doc/html/rfc7519>

Claim	Description
iss	Identifies which entity has issued the JWT. This could be an authentication server or a service issuing the token.
sub	Identifies the entity for which the JWT is issued. This could be the identity of a user, a device, or another entity the JWT represents.
aud	Specifies the recipient or audience for which the JWT is intended. This can be a single recipient or a list of recipients.
exp	Specifies when the JWT expires and should no longer be considered valid. After this time, the JWT should not be accepted by recipients.
nbf	Specifies when the JWT becomes valid and can start being used. Before this time, the JWT should not be accepted by recipients.
iat	Specifies the time when the JWT was issued. This can be useful for checking the token's age and implementing token replay prevention.
jti	Provides a unique identifier for the JWT. This can be useful for identifying and tracking the token, especially in cases where there's a need to revoke or handle the tokens individually.

Table 2.1: Registered claims available for JWTs [16]

Custom claims consist of public and private claims that are defined at will. Public claims often contain generic information such as name or email and must be registered or use collision-resistant names. Private claims are made specifically for the application's usage of the JWT and are only valid inside your implementation of JWT [16]. Code listing 2.2 shows an example of the claims part of a JWT token.

Code listing 2.2: Example of JWT claims

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "iat": 1516239022,
  "exp": 1516239022
}
```

Finally, the signature of a JWT ensures the integrity and authenticity of the token. It provides a means for verifying that the token has not been tampered with and was issued by a trusted entity. The header and payload are both base64 encoded to create the signature. Then, the cryptography algorithm specified in the header is used along with the encoded header, payload, and a secret key to generate the signature [16]. If HMAC SHA256 is the cryptography algorithm used, the signature part will look like it does in Code listing 2.3 bellow.

be time-consuming, as the roles need to be well thought out and vetted [18].

2.5.2 Attribute-Based Access Control

Unlike RBACs role-based policies, Attribute-Based Access Control (ABAC) uses device and user attributes. One of these attributes could be a role, but they can also be much more. ABAC is an evolved and more complex version of RBAC [19]. When a user tries to access a resource, the decision to let the request go through or not is based on the following factors;

Attributes	Description
Subject	Represents the request's initiator, in this case, the user. Subject attributes include ID, roles, groups, security clearance, and other identifying criteria.
Resource	Represents the resource or asset that the subject tries to access. Let's assume it's an API. The resource attributes are the APIs identifying characteristics such as location, name or ID, API type, group, etc.
Action	What the user is trying to do with the resource. The most common are GET, POST, PUT, DELETE, or PATCH. The action attributes vary based on the resource and how a user can interact with it.
Environment	More about the access request in general. Attributes could be user device, request time, location, and more.

Table 2.2: Overview of attribute types used in ABAC [20]

Code listing 2.5 illustrates an example. Someone is requesting access to an API. This API is reachable only to a select number of users, as it handles sensitive information. For access to be granted, both the attributes of the user making the request and the request itself have to fulfil all of the following attribute requirements:

Code listing 2.5: Example of how resources can be restricted to Oslo-based users in the Security Group with either Analyst or Administrator role using ABAC

```
Subject "job.role" = "Analyst" || "Administrator"
Subject "group" = "Security"
Resource "id" = "sec_api_1"
Resource "type" = "API"
Action "GET"
Environment "user.department" = "Oslo"
```

ABAC can provide fine-grained access control while being highly flexible in policy-making. An administrator can make as many or as few attributes as needed for their organisation, as there is no set-in-stone attribute set one needs to use. However, this strength could also be a weakness. If ABAC is implemented for a larger

organisation, the range of attributes that may be required could make the implementation quite complex [20].

2.5.3 Just-in-Time

Just-in-Time (JIT) grants users or services temporary permission on an as-needed basis rather than relying on static predefined permissions. It is a form for identity and access management often used to address scenarios where a user may not regularly require access to a specific application or system. Still, they need temporary entry during certain situations or tasks [21].

In the JIT access model, delegating temporary permissions is a process that leverages dynamic access control mechanisms. This delegation often employs a request approval workflow, where a user submits a request for access to a particular resource for a specific duration and purpose. The request is then evaluated based on predefined security policies. Approval workflows can be automated based on role-based policies or escalated to human approval for sensitive access requests. This process is facilitated by the IAM system, which grants temporary permissions.

2.6 Authentication and Authorisation Protocols

Authentication and authorisation protocols define the mechanisms and standards through which identities are verified, and access rights are granted. These protocols are fundamental to security frameworks, ensuring only authenticated and authorised entities can access resources. Popular protocols include OAuth 2.0 for delegating access and OIDC for identity services. Additionally, SAML is widely used in enterprise environments to enable secure, cross-domain authentication and authorisation. Each protocol serves distinct roles, some focusing exclusively on authentication or authorisation, while others provide comprehensive solutions encompassing both aspects.

2.6.1 OAuth 2.0

OAuth 2.0 is an authorisation standard that allows third-party applications to attain a predetermined level of access to a service on behalf of the owner of the service in question [14]. The predetermined access is called a "scope". OAuth 2.0 achieves this without giving the resource owners credentials to the third party.

Four main actors are involved in a usual OAuth 2.0 dataflow as shown in Table 2.3.

Attributes	Description
Resource Owner	The owner of the resource in question. If a third-party website wants to access a high school diploma for a job application, the applicant is the resource owner.
Resource Server	Where the protected resource is stored. Following the previous example, if the resource is a high school diploma, the resource server for all Norwegian students is the national diploma database.
Client	The application that wants access to the resource in question. The client would be the third-party job application website in the ongoing example.
Authorisation Server	The entity which issues OAuth 2.0 access tokens. In the example used, the authorisation server could be either "Feide" or "ID-porten".

Table 2.3: Parties involved in an OAuth 2.0 flow

Grant Types

In OAuth 2.0, the term “grant type” refers to how an application gets an access token. OAuth 2.0 defines several grant types, and OAuth 2.0 extensions can also define additional grant types. Each grant type is optimised for a particular use case, whether a web app, a native app, a device without the ability to launch a web browser or server-to-server applications [22].

The main grants used are the authorisation code grant, client credentials grant and device authorisation grant. The device authorisation grant is used for IoT devices like smart TVs and won't be used within this project's scope. Two other grant types also exist natively in OAuth 2.0, implicit grant and resource owner password credentials grant, but Okta strongly recommends against using these, as they are inherently insecure [23].

Authorisation Code Grant

The most common grant type is the "Authorisation Code Grant". The authorisation code grant uses two requests from the application to the authorisation server to obtain an access token. Firstly, the user browser is redirected to the authorisation server through the front-channel to authorise an API call for the user. The authorisation server then interacts with the user to obtain consent for the authorisation request. After getting consent, the authorisation server redirects the user back to the application with an authorisation code. The application then uses the authorisation code to send a second, Back-channel request to the authorisation server to get an access token. The application now finally receives an access token issued, which it can use to call the API in question [24, p. 71].

Proof Key for Code Exchange

Proof Key for Code Exchange (PKCE) is a mechanism that can be used with OAuth 2.0 requests to prevent a malicious process from intercepting an authorisation code and using it to get an access token, especially on mobile or public devices. PKCE does not authenticate clients, but it ensures that the application that requested an authorisation code is the same application that uses it to get an access token.

When using PKCE, the application creates a cryptographically random string called a code verifier [25]. A code verifier should have enough entropy that it would take an attacker longer to guess the value than for the OAuth 2.0 token exchange to complete. The application computes a derived value called a code challenge from the code verifier. This derived value is typically a hash of the code verifier.

Figure 2.2 shows the flow of the Authorisation Code Grant, with the elements of PKCE represented by colours. The code challenge and the derivation method are sent from the client application to the authorisation server when sending the initial access request. The code challenge is represented in blue, and the derivation method is represented in red. When the client sends their authorisation code to the authorisation server, the client includes the code verifier, represented by green. The authorisation server checks that the code segments received from both messages are equal using the derivation method previously received. This allows an authorisation server to detect if someone is trying to use a stolen authorisation code [24, p. 73].

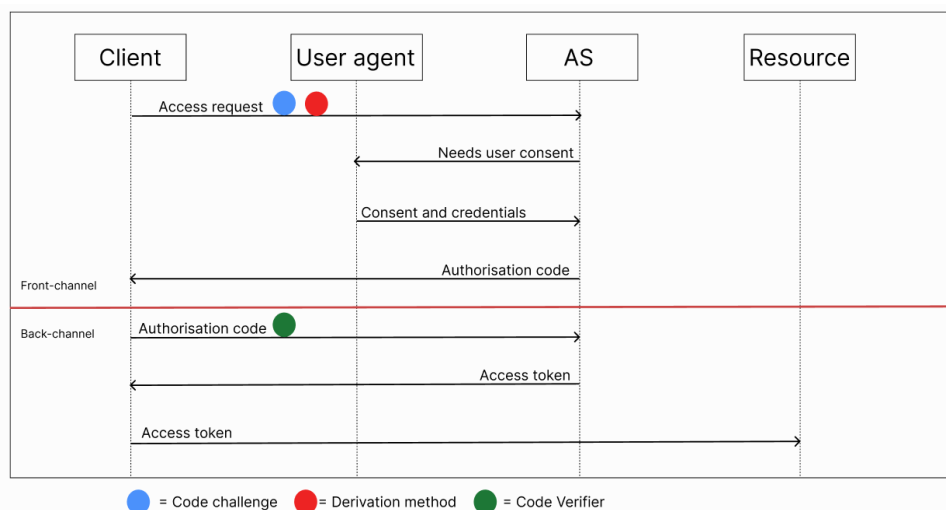


Figure 2.2: OAuth 2.0 Authorisation Code Grant with PKCE [24, p. 73]

Client Credentials Grant

The client credentials grant is used when an application calls an API to access resources the application already owns, not on behalf of a user [26]. When an application uses the client credentials grant type, it authenticates to the authorisation server with its credentials to obtain an access token. The client credentials grant is a more streamlined version of the authorisation code grant, as no end-user interaction is needed. As shown in Figure 2.3, consent and credentials are sent with the initial request, making the token exchange very fast. The use of this grant type requires that the application can maintain confidential secrets to authenticate itself [24, p. 80].

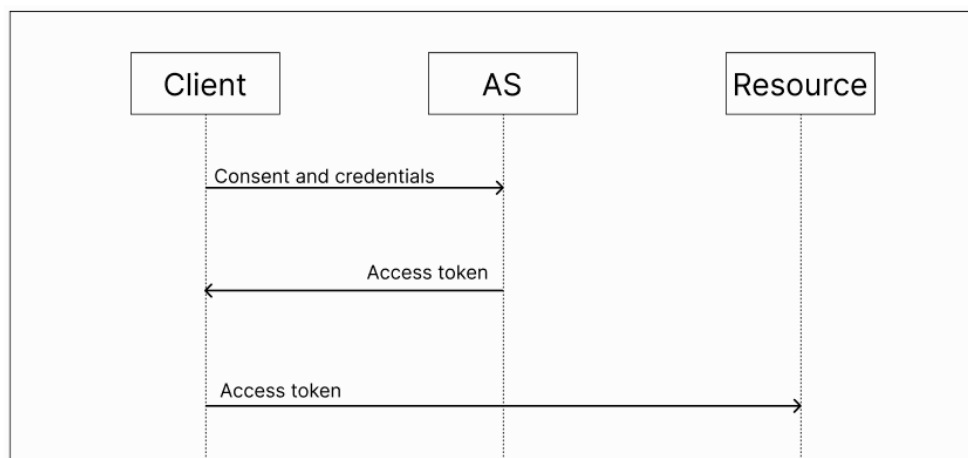


Figure 2.3: OAuth 2.0 Client Credentials Grant [24, p. 80]

2.6.2 OpenID Connect

While OAuth 2.0 mainly focuses on granting authorisation, OIDC ensures the authenticity of the user's identity. By establishing a standard method of verifying a user's credentials, OIDC adds a layer of identity to the OAuth 2.0 framework [24, p. 103]. This is done by introducing ID tokens, which are JWTs containing essential user information. These tokens empower client applications to verify the user's authenticity in a standardised and secure manner without directly handling sensitive user credentials. OIDC uses four grant types: the authorisation code grant, client credentials grant, implicit grant and resource owner password grant. For the same reasons as in OAuth 2.0 the implicit grant and resource owner password grant is not recommended.

Authorisation Code Grant

OIDC grants cater to different application needs [24, p. 109]. The most common one is OIDC authorisation code grant, which follows the same steps as in OAuth

2.0, except that the application gets an ID token after sending the authorisation code to the authorisation server using a Back-channel. This method requires an end user to explicitly grant permission to a relying party so they can request ID tokens and access tokens from an authorisation server, often called an OpenID provider for OIDC.

Client Credentials Grant

Next is the Client credentials grant, designed for client applications without an end user. This facilitates machine-to-machine interactions as in with OAuth 2.0. This approach requires the application to operate on the server side due to the need to securely manage the client's secrets. Since the credentials are embedded directly within the application, it's unsuitable for direct end-user involvement [27].

2.6.3 Federation and Single Sign-On

OIDC also aims at enhancing the user experience by facilitating SSO and cross-domain identity federation [6, p. 129], enabling users to log in with the same identity across multiple services. This allows individuals to utilise a singular identity for authentication across many services. Organisations can integrate the IAM system with OIDC to authenticate users and communicate ID tokens to third-party applications. This process ensures that users are required to authenticate only once, leveraging the ID tokens issued by the OpenID provider for subsequent access without repeated logins. An illustrative example of this use case is when a user intends to register a new account at company A and opts to use an existing Google account for authentication. By selecting this option, the user explicitly authorises company A to request and receive user information from Google's OpenID provider. Company A can rely on the ID token information to establish a new user account, thereby obviating the need for the user to undergo the traditional registration process involving the creation of new credentials. The streamlined SSO capability significantly simplifies access across services and negates the exposure to potential security risks associated with credential management.

2.6.4 Security Assertion Markup Language

SAML is an open standard protocol with an XML-based framework to facilitate the secure exchange of authentication and authorisation information between entities. By enabling the exchange of information between an IDP and Service Provider (SP), SAML offers SSO and identity federation across domains. At its essence, SAML simplifies the user experience by allowing access to multiple applications and services with a single set of credentials, addressing the challenges of password management and fatigue [24, p.127-128].

The architecture of SAML is built upon three critical entities. The user agent, which in most situations is the user's web browser. The SP, which holds the resources the

user is trying to access. Lastly, the IDP, the entity responsible for user authentication. For the SP to verify and grant access to the user, trust must be established with the IDP. This is achieved through SAML metadata, an XML document containing information necessary for the SP to communicate with the IDP [24, p.129].

Once the trust relationship is established through exchanging and validating SAML metadata, the IDP is then in a position to issue SAML assertions, an XML document containing statements about the user. These statements encompass the user's authentication status and attributes and serve as evidence of the user's identity and access rights. Firstly, the authentication statements within the assertion provide details about how and when the user was authenticated by the IDP. The statements includes information about the authentication method and a timestamp marking the authentication event. Secondly, the attribute statements list specifies user attributes that the IDP shares with the SP. These could range from user identifiers, such as email addresses, employee numbers, groups, or other information the SP requires for authorisation decisions. Upon receiving a valid SAML assertion, the SP grants the user access to its resources. This mechanism streamlines access management and enhances security by abstracting the user authentication information from the SP.

SAML 2.0 Flows

In SAML 2.0, there are two primary communication flows that can be initiated by the user agent: the SP-initiated flow and the IDP-initiated flow. In the SP flow, the process begins when a user attempts to access resources held by an SP without an existing session. The SP redirects the user agent to a trusted IDP with an authentication request. Upon authentication, the IDP sends back a SAML assertion to the SP for access to the resources [24, p.130-131]. In the IDP flow, the user agent first logs into the IDP, providing a list of available services or applications. When a user selects a service to access, the IDP sends a SAML assertion to the SP without waiting on additional requests, efficiently logging the user into the service. This flow is helpful in scenarios where a central portal is used to access multiple services [24, p.131-132].

3 Threat Model

3.1 Introduction

This chapter aims to perform a threat modelling of an API environment. In API security, it is important to understand the specific environment an API operates within, along with the various potential threats that emerge. A threat can be defined as an event or set of circumstances that defeats the security goals of an API. For example, an attacker stealing names and address details from a customer database threatens confidentiality [12, p. 16].

Threat modelling is a proactive security exercise to examine and identify potential threats, vulnerabilities, or attacks that could be leveraged against API components and data flows [12]. There are several ways to conduct threat modelling. This threat model follows the process described in ISO27005:2022, focusing on identifying, assessing and handling threats [28].

As illustrated in Figure 3.1, the group's threat modelling methodology begins with assessing criticality, which involves determining the importance of assets using CIA and defining a risk appetite for the organisation based on the criteria set in the CIA analysis.



Figure 3.1: Figure illustrating the used threat modelling method

The second step involves identifying assets within the API environment. This step includes identifying the main logical components. Identifying assets within a threat model is important for determining what needs protection and for implementing security measures tailored to the specific needs of each asset.

Next is the threat identification using the STRIDE methodology. This approach allows potential threats to the environment to be systematically uncovered and provides a robust framework for further threat analysis.

After identifying threats, the model moves on to the risk assessment phase. This involves evaluating the impact and the likelihood of each threat from the STRIDE analysis using a risk matrix. The threat model uses the DREAD model to evaluate the impact of each threat in more detail. This helps to quantify each threat's risk, providing a detailed understanding of potential security vulnerabilities. Using a risk matrix followed by the DREAD model is beneficial because it first provides a visualisation of risk levels and then offers a detailed assessment of each threat's potential impact.

Finally, the model integrates the findings into a bowtie model, visually representing threats and the correlating mitigation strategies. This model clarifies the relationships between threats and their impacts and highlights effective intervention strategies, enhancing the overall security architecture of the environment. This is showcased later in the report.

3.1.1 Threat Model Scenario

To conduct a threat assessment, one must identify the main logical components in an API environment. However, before diving into this explanation, a scenario will be introduced that showcases the challenges the threat model will address.

Imagine a financial data service provider providing secure access to sensitive financial information for internal users subscribing to the API leveraging a centralised authority. This means all authentication happens within the organisation's system, and users must be part of the organisation's user directory to be authenticated. The system must be secure, preventing unauthorised modifications or access to data. The API should be responsive, with high uptime for users and be accessible globally. The API should support multiple identity types, each with different levels of access privileges as outlined in the CCSK guide [9].

To handle this complexity, imagine different users, such as a developer, financial staff, and an admin. Each of these needs different access levels. For example, the developer might require access to the full range of API functions for application development, the financial staff may need broader access to financial data for processing transactions, and the admin should have comprehensive access to all system data, including transactional records and user management capabilities for compliance and oversight.

The API's goal is to ensure that these diverse actors can interact with the API securely and efficiently, with the assurance that financial data is protected from unauthorised access and manipulation. An API system with robust authentication, detailed authorisation controls, and reliable data integrity checks is required to achieve this.

3.2 Assess Criticality

3.2.1 CIA Security goals

When designing the threat model for this API environment, the security goals are established to safeguard the system and its users. The security goals are centred around the core principles of the CIA Triad: Confidentiality, Integrity, and Availability [12, p. 14].

Principle	Description
Confidentiality	The objective is to ensure that sensitive information is accessed only by authorised users.
Integrity	The objective is to maintain the accuracy and completeness of data. This ensures that information remains unaltered and trustworthy from its source to its destination.
Availability	The objective is to guarantee that data and resources are accessible to authorised users whenever needed. This means ensuring systems are running and information can be accessed without delay.

Table 3.1: Overview of the three CIA triad elements

The CIA rank assesses the entire API system's confidentiality, integrity, and availability. These ranks indicate the system's overall security risk level, ranging from 1 to 4, where 1 signifies a low risk, and 4 denotes a critical risk. To set the values for the CIA elements NTNU's criteria for assigning the different values has been used¹ Table 3.2 gives a broad overview of the requirements.

Rank	Confidentiality	Integrity	Availability
1	Public	No requirement	No requirement
2	Internal	Expected	2 days
3	Confidential	Required	4 hours
4	Strictly confidential	Critical	Immediately

Table 3.2: NTNU Levels of Data Security Requirements [29]

The API system has been assessed and given the following rank:

Confidentiality Rated 2 (Internal): The confidentiality rating of 2 indicates a moderate level of sensitivity associated with the data handled by the API system. While the system does not contain highly sensitive information, it does manage

¹Detailed guidelines for setting the values can be found here: <https://i.ntnu.no/wiki/-/wiki/English/Policy+for+Classification+of+Information+Assets>

personal data for its customers and employees, along with financial data that is subject to confidentiality and shouldn't be shared outside the organisation.

Integrity Rated 3 (Required): The integrity rating of 3 assigned to the API reflects the high level of authenticity and accuracy required for the financial data it processes. This rating underscores the reliance of the API on delivering precise and untampered information for financial transactions and investments. A compromise in data integrity could lead to significant economic losses, bad decision-making, damage to the organisation's reputation, and legal repercussions, given the strict regulatory standards governing financial data.

Availability Rated 2 (2 days): The availability rating of 2 for the API indicates that it primarily affects isolated systems rather than being critical to the core business operations or entire departments. It is mainly used by employees to access customer and financial data. While essential for these specific tasks, the API does not impact multiple systems, so a temporary disruption would not critically affect overall operations.

3.2.2 Risk Appetite

Risk appetite refers to the level of risk that an organisation is willing to accept while pursuing its objectives. It acts as a guideline for making risk decisions, helping to ensure that the risks taken align with the organisation's strategic goals and capacity to handle those risks [28].

It was determined that the risk appetite could be moderate, given the assessed CIA values. The system's confidentiality, integrity, and availability ratings indicate that while some controls are necessary, the overall sensitivity and criticality of the data and operations are not extremely high. The appetite can not be higher due to a potential security breach could lead to substantial financial or reputational damage. Therefore, it is crucial to maintain strict controls over the acceptable risks. As such, to consider the system secure and the operations viable, the following criteria have been established based on the DREAD modelling and risk matrix assessments:

- No threat should be evaluated by the DREAD model, as seen in Table 3.13, to a ranking higher than low.
- No threat should be categorised in the risk matrix at a level above important, placed in the yellow section in Table 3.6.

The reason the risk appetite accepts a higher rating in the risk matrix compared to the DREAD analysis is that the risk matrix uses more rating levels, with four levels instead of DREAD's three. Additionally, in the risk matrix, the green and yellow fields do not account for half of the boxes despite representing half of the ranking categories. This difference in structure and categorisation results in the risk matrix allowing a broader range of acceptable risks. Allowing medium risks

from the DREAD model would represent a significant step up from low, making it unacceptable.

3.3 Identify Assets

3.3.1 Main Logical Components

The API environment consists of several components, each essential for keeping the system running safely. These components are the following.

Actor

The actor refers to any end-user or system interacting with the API. In the API environment, actors initiate the flow of operations by contacting the central user directory where the authentication process starts, making the actors the starting point for all transactions within the system. The actor can have several different digital identities, such as an administrator of the organisation, a user accessing the API or a server set up to retrieve information from the API.

Identity Provider

The API utilises Microsoft Entra ID as its IAM system, providing a centralised user directory for all users associated with the API. Additionally, Microsoft Entra ID serves as the IDP for authentication and authorisation, ensuring that requests have valid access permissions and match a digital identity in the IAM system before issuing any access tokens. They ensure that requests have access to the API endpoint they are requesting access to.

Content Delivery Network

CloudFront acts as the Content Delivery Network (CDN). It is an AWS CDN that speeds up the delivery of web content and APIs by caching content in global edge locations. In an API environment, it reduces latency and load on the server, making the API faster and more reliable for users worldwide while efficiently managing high traffic volumes. Considering the aim to provide service on a global scale, CloudFront's global network is crucial. It ensures all users have quick and reliable access to services regardless of location [30].

Firewall

AWS Web Application Firewall (WAF) is a firewall that secures APIs against common web threats and attacks by filtering incoming API requests based on pre-defined security rules. Its role in the environment safeguards the APIs from malicious traffic, known threat actors, and exploits, such as SQL injection and cross-site scripting, ensuring that only legitimate requests reach the backend services [31].

API Gateway

AWS’s API Gateway acts as the front door for the APIs resources, allowing its users to create, publish, and manage secure APIs at scale. It forwards requests to the right services in the infrastructure, including traffic management, authorisation, and access control, making deploying and maintaining APIs as part of an environment easier [32].

Token Authorisers

The API uses a JWT authoriser, which is an AWS module for validating JWT tokens from either the OAuth 2.0 or the OIDC framework. The JWT authoriser checks whether the audience, issuer, and scope are valid based on its configuration [33]. In Figure 3.2, the JWT authoriser handles the authorisation to the endpoint based on the scope in the access tokens received from Microsoft Entra ID.

Business Logic

Business logic happens in HTTP integrations, which are designed for API endpoints that redirect the received HTTP request to a URI where the request is performed[34]. In this case, a GET request is forwarded to resources in an external site, where data for the endpoints are stored.

Data Flow

A data flow has been made based on the fictional organisation presented at the start of the threat modelling chapter. Figure 3.2 illustrates the data flow for the organisation using one API endpoint as an example. The AWS environment is represented by a grey background.

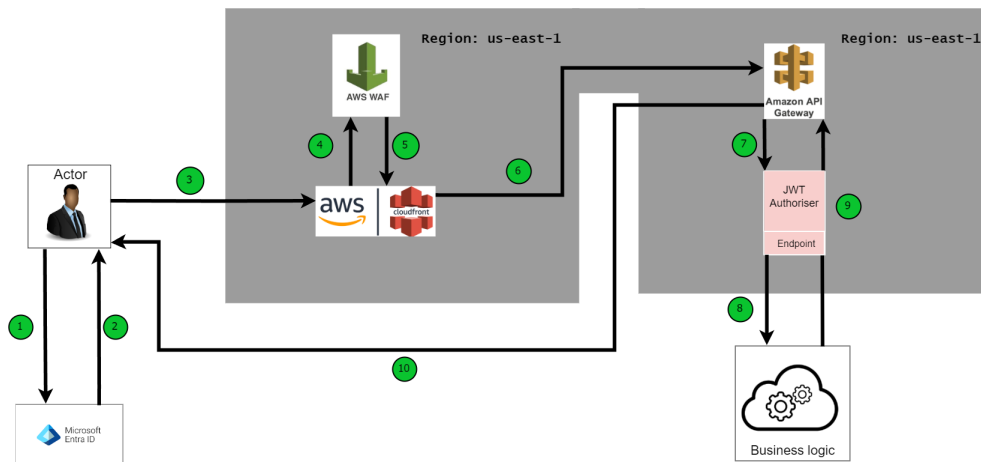


Figure 3.2: Illustration of the organisation’s data flow that the threat model is based on

Step 1 Authentication request: The actor authenticates themselves with the IDP Microsoft Entra ID to obtain an access token. Step 1 involves initiating the request, sending credentials and completing other authentication methods required.

Step 2 Authentication request response: Microsoft Entra ID authenticates the actor, and if their credentials are accepted, an access token is returned.

Step 3 Actor to CloudFront: Holding the access token, the actor makes a request to CloudFront, targeting a specific endpoint with the access token included in the request header. This marks the entry of the request into the AWS infrastructure.

Step 4 CloudFront to AWS WAF: CloudFront then sends the request to WAF to inspect the request for potential security threats.

Step 5 WAF to CloudFront: If the request passes the configured rulesets in the WAF it gets forwarded back to CloudFront.

Step 6 CloudFront to API Gateway: CloudFront forwards the approved request to the API Gateway, which serves as an entry point to validate the authorisation and returns the data requested later in the flow.

Step 7 API-Gateway to JWT Authoriser: The API Gateway invokes the specified endpoint's JWT authoriser responsible for authorising the request. The JWT authoriser validates the data in the access token against its own configuration.

Step 8 Endpoint to resource The configured HTTP endpoint within AWS forwards the request to an external site, exiting the AWS infrastructure, to fetch the required data.

Step 9 resource to API Gateway: The collected data from the external site gets sent back to the API Gateway, again entering the AWS infrastructure.

Step 10 API Gateway to the actor: Finally, the API Gateway sends the data retrieved back to the actor. This is the final step, where the data exits the AWS infrastructure and is transmitted over the internet back to the user's client.

This data flow sequence creates a secure route from the external actor through different AWS services available through the API, ensuring that security is maintained and data is appropriately managed at every stage before the client is granted access.

3.4 Threat Identification

3.4.1 STRIDE

In the approach to securing the API environment, the STRIDE methodology has been used, complemented by the foundational principles of the CIA Triad, to identify and categorise potential security threats. The STRIDE methodology is a model for identifying security threats and categorising them into six categories.

It serves as a framework for considering possible threats to a system and helps in planning appropriate security measures and countermeasures. Each category under STRIDE aligns with the core objectives of information security outlined by the CIA Triad [12, p. 18]. These are the six STRIDE categories, which will be explained in further detail below.

- Spoofing Identity
- Tampering
- Repudiation
- Information Disclosure
- Denial of Service (DoS)
- Elevation of Privilege

Spoofing

Spoofing is pretending to be somebody else to gain privileges that the user usually would not have. Spoofing attacks directly threaten the confidentiality, availability, and integrity of the system by attempting to access sensitive information under a false identity [12, p. 18].

- **S1 Unauthorised Access via Stolen Credentials**
The attacker uses stolen credentials or exploits weak authentication systems to masquerade as a legitimate user. This can potentially lead to gaining unauthorised access to the API.
- **S2 Authentication Bypass in API Gateway**
When attackers exploit weaknesses in the API Gateway's authentication mechanisms to gain unauthorised access to the system, effectively bypassing the normal authentication procedures. This could involve various techniques, such as exploiting misconfigurations, to deceive the system into granting access without proper authentication.
- **S3 Identity Spoofing in Microsoft Entra ID**
Attackers attempt to spoof legitimate identities to gain access to the system. This can be done by manipulating authentication tokens or mimicking IP addresses to appear as authorised users.
- **S4 Session Hijacking**
An attacker hijacks a user's session by capturing or predicting their session token, allowing unauthorised actions within the API [35].

Tampering

Tampering is the action of performing unauthorised changes that affect the Integrity and Availability and can compromise the system's trustworthiness [12, p. 18].

- **T1 CloudFront Cache Manipulation**
An attacker manipulates the cache by deceiving CloudFront into storing and distributing altered content, impacting all users accessing that specific resource later.
- **T2 WAF Rule Manipulation**
Malicious actors could attempt to modify the WAF rules to allow harmful requests through or to block legitimate traffic.
- **T3 Unauthorised Modification**
Attackers exploit a vulnerability in the API, which allows them to update the email address associated with a user's account without requiring the user's current password for verification. By obtaining the victim's ID token, attackers can manipulate the email address and potentially initiate a password reset, leading to account takeover [4].

Repudiation

Repudiation refers to a user denying to have done something that they did, such as making a transaction or sending a message, without a way for the other party to prove otherwise [12, p. 18]. Addressing repudiation reinforces the integrity aspect of the CIA Triad by ensuring that actions cannot be falsely denied, thus preserving a reliable audit trail. Additionally, it upholds non-repudiation, ensuring that every transaction or communication within the system is attributable and verifiable, which is crucial for maintaining trust and accountability in digital interactions.

- **R1 Lack of Audit Trail in API Gateway**
An authenticated user performs an action via the API, such as modifying data, but later denies making this change because there is no audit trail.
- **R2 Microsoft Entra ID Account Compromise Denial**
A user accesses sensitive information, claims their account was compromised, and denies accessing the data.

Information Disclosure

Information disclosure violates Confidentiality by exposing sensitive data to unauthorised parties [12, p. 18].

- **I1 API Gateway Misconfiguration**
If an API endpoint is not configured correctly to restrict access to sensitive data, it could result in unauthorised disclosure of personal user information when a query parameter is manipulated.
- **I2 Intercepted Data Transit**
Data in transit between components (e.g., from API Gateway to HTTP integrations) is intercepted by a Man in the Middle (MITM) attack, revealing sensitive information.
- **I3 Data Endpoint Manipulation**
The API provides an online platform for its clients to access and manage their accounts, including viewing financial statements and transaction histories. To view these accounts, an API endpoint can be called in this way: `/accounts/accountName/transactions.json`. An attacker, by inspecting the browser requests on their own account page, discovers the API endpoint used to fetch transaction data and identifies the pattern used in the URL. Additionally, the attacker finds another API endpoint that lists all the client account names hosted by the financial institution. With this information, the attacker crafts a script that cycles through the list of account names, substituting `accountName` in the URL to fetch transaction data for each account [36].
- **I4 Unauthorised User Data Access**
The API has a function to report issues. However, the API endpoint for reporting issues is vulnerable, granting authenticated users access to sensitive information about the API, such as information about other reported issues containing user or business data. [37].
- **I5 Unauthorised Access to User Data via Admin Endpoint**
The API exposes an endpoint, `GET /api/admin/v1/users/all`, intended exclusively for administrators. This endpoint retrieves details of all users without implementing function-level authorisation checks. Exploiting knowledge of the API structure, an attacker guesses and accesses this endpoint, exposing sensitive user information. [38].

Denial of Service

Denial-of-service (DoS) attacks is a cybersecurity threat where attackers flood a system, server, or network with overwhelming traffic to render it inaccessible to intended users. DoS attacks threaten the Availability of the API [12, p. 18].

- **D1 CloudFront DoS Attack**

While CDN is designed to absorb large amounts of traffic, an attacker could still target the system with a large-scale DoS attack with a massive amount of traffic using a botnet to overwhelm the CDN's ability to respond to legitimate requests.

- **D2 WAF Scripted Request Flood**

An attacker deploys a script designed to swiftly create HTTP requests filled with malicious payloads. The goal is to activate the WAF rule evaluation process, thereby consuming computational resources.

- **D3 API Gateway Application-layer DoS Attack**

The API Gateway is targeted with an application-layer DoS attack. The attacker floods the system with a high volume of complex requests, which forces the backend to execute slow, resource-demanding processes. This drains the available system resources, compromising the performance and availability of the service.

Elevation of Privilege

Elevation of privilege is a security vulnerability that allows an attacker to gain higher access levels than originally authorised, which enables the execution of commands, access to confidential data, and the ability to perform unauthorised actions within a system. It undermines the system's integrity, availability and confidentiality [12, p. 18].

- **E1 HTTP Integration Attack**

An attacker might find a vulnerability within an HTTP integration (e.g., due to outdated libraries or insecure code) and exploit it to execute commands or access sensitive data beyond what the function is normally allowed to do.

- **E2 API Gateway Misconfiguration Exploit**

A misconfiguration in the API Gateway allows an attacker to inject malicious code through an API request, which is then executed by the backend service, potentially leading to the elevation of privileges.

- **E3 Misconfigured Identity Policy**

An attacker gains initial access to the system with limited privileges but discovers a misconfigured identity policy that allows them to escalate privileges for their account.

3.5 Risk Assessment

3.5.1 Risk Matrix

Following the STRIDE analysis, the group transitioned to the risk matrix to further evaluate and prioritise the identified threats discovered in subsection 3.4.1, on how likely they are to happen and the impact they could have. The risk matrix is a clear guide that helps determine which issues should be tackled first, ensuring the most critical issues are addressed. This risk matrix is aligned with the IEC 62443-3-2:2020 standard, which ensures a consistent and globally recognised approach to risk assessment [39].

To calculate the values used in the risk matrix, probability and consequence level must first be assigned to each scenario. Table 3.3 outlines the criteria for assigning the probability level and Table 3.4 outlines the criteria for the consequence level. The criteria for consequence and probability are based on guidelines from IEC 62443-3-2:2020 [39]. The combination of consequence and probability gives the risk level for the threat. Columns in the risk matrix represent the probability values, and the rows represent the consequence values for the threat. Based on the consequence and probability values, the threat lands in the 5x5 grid in the risk matrix, and the box the threat lands in represents the risk level. Table 3.5 outlines the severity of the different risk levels.

Probability Scale	Guideword	Description
1	Rare	Highly unlikely to occur.
2	Unlikely	Possible to occur.
3	Moderate	Likely to occur.
4	Likely	Almost certain to occur.
5	Certain	Sure to occur.

Table 3.3: Criteria for assigning probability values in risk matrix [39]

Consequence Scale	Guideword	Description
1	Insignificant	Will not compromise API security.
2	Minor	Can compromise API security to a limited extent.
3	Significant	Can compromise API security and may require remediation.
4	Major	Compromises API security significantly and requires immediate action.
5	Severe	Leads to complete compromise of API security, posing a critical threat.

Table 3.4: Criteria for assigning consequence values in risk matrix [39]

In the risk matrix, consequence and probability are not weighted equally, as demonstrated by the asymmetrical design of the matrix. This design decision is aligned with the guidelines provided in IEC 62443-3-2:2020 [39]. The rationale for this approach is that risks characterised by frequent occurrences yet minor consequences are generally less critical than those that occur less frequently, but result in significant consequences. Therefore, the matrix emphasises the severity of potential consequences rather than the frequency of occurrence. This prioritisation facilitates a more effective allocation of resources towards mitigating risks that, despite their infrequency, could cause substantial harm or disruption, thereby enhancing strategic risk management.

Risk Level	Description
Insignificant	No need for risk mitigation measures unless cost-effective.
Important	Evaluate risk mitigation measures. Risks must be monitored at a minimum.
Substantial	Implementation of risk mitigation measures is required.
Critical	Unacceptable risk level. Risk mitigation measures and immediate action are required.

Table 3.5: Description of importance of colours in risk matrix

Table 3.6 presents a risk matrix based on identified threat scenarios for the API system, outlined in subsection 3.4.1. Each threat from the STRIDE analysis has had their consequence and probability evaluated and placed in the risk matrix. As seen in the risk matrix, 15 threats are ranked as critical. Identifying these high-risk scenarios highlights significant vulnerabilities that require urgent mitigation to enhance the API system's security. Implementing mitigations and preventive controls will help lower these risks from high to manageable levels, ensuring system integrity and aligning with best security practices.

Probability	Consequence				
	Insignificant 1	Minor 2	Significant 3	Major 4	Severe 5
Certain 5					
Likely 4			I3		
Moderate 3			T3 R1 R2 I2 E2	S1 S2 S3 S4 T1 T2 I1 I4 I5 D1 D2 D3 E1 E3	
Unlikely 2					
Rare 1					

Table 3.6: Risk matrix illustrating consequences versus probability for each scenario before mitigations are implemented

The organisation’s risk appetite mandates that all scenarios within the risk matrix remain within the yellow zones, indicating acceptable risk. Presently, only one scenario is classified within these zones, the majority are situated in the red and orange zones, indicating high risk. This reveals that the system currently is outside the acceptable risk thresholds. Therefore, the system should not be utilised until further security measures are implemented to align the operational risk levels with the organisation’s defined risk appetite.

3.5.2 DREAD

After employing the risk matrix, the DREAD model has been utilised to prioritise threats by assigning each of its five risk categories, Damage, Reproducibility, Exploitability, Affected Users, and Discoverability, a numerical range from 1 to 3. The criteria for assigning the different rankings are outlined below. This has been done for all the scenarios identified in the STRIDE analysis. The Official Guide to the CSSLP CBK [40] states that using a smaller range, such as 0 to 3, is preferred instead of the standard 1 to 10. This makes the ranking more defined, the vulnerabilities less ambiguous, and the categories more meaningful. The severity rating for a particular threat is computed by averaging the scores across these 5 DREAD categories. This averaged score provides a quantifiable measure of threat severity, guiding stakeholders in identifying which vulnerabilities require immediate attention and resource allocation. By systematically evaluating each threat, organisations can effectively prioritise mitigation efforts based on these severity ratings, ensuring a targeted and efficient response to the most critical vulnerabilities[40].

Damage Potential

"Ranks the damage that will be caused when a threat is materialised, or vulnerability exploited" [40, p. 214].

Score	Category rank
0	Nothing.
1	Individual user data is compromised or affected.
2	Affects a substantial portion of the system.
3	Complete system or data destruction.

Table 3.7: DREAD values for ranking damage potential [40, p. 214]

Reproducibility

"Ranks the ease of being able to recreate the threat and the frequency of the threat, exploiting the underlying vulnerability successfully" [40, p. 215].

Score	Category rank
0	Highly unlikely to reproduce the threat.
1	Very hard to reproduce the threat.
2	One or two steps required to reproduce the threat.
3	Very easy. Just the address bar in a web browser is sufficient without authentication.

Table 3.8: DREAD values for ranking reproducibility [40, p. 215]

Exploitability

"Ranks the effort that is necessary for the threat to be manifested and the preconditions, if any, that are needed to materialise the threat" [40, p. 215].

Score	Category rank
0	Highly unlikely to exploit the threat.
1	Requires Specialised tools and knowledge.
2	Malware exists on the Internet, or an exploit is easily performed using available attack tools.
3	Just a web browser.

Table 3.9: DREAD values for ranking exploitability [40, p. 215]

Affected Users

"Ranks the number of users or installed instances of the software that will be impacted if the threat materialises" [40, p. 215].

Score	Category rank
0	None.
1	Few users.
2	Several users but not all.
3	All users.

Table 3.10: DREAD values for ranking affected users [40, p. 215]

Discoverability

"Ranks how easy it is for external researchers and attackers to discover the threat if left unaddressed" [40, p. 215].

Score	Category rank
0	Highly unlikely to discover.
1	Difficult: Inside knowledge or source code access is necessary.
2	Moderate: Can figure it out by guessing or by monitoring network traces.
3	Easy: Information is visible in the web browser address bar or in a form.

Table 3.11: DREAD values for ranking discoverability [40, p. 215]

After assigning values to each category, the average is calculated to determine a numerical risk ranking. Based on the scenario's value, it is placed into low, medium or high categories. Table 3.12 shows the colour assigned to the average score, the colour is used in Table 3.13 to visualise which DREAD rating the scenario has been given. The results from Table 3.13 can be used to prioritise mitigation efforts [40, p. 216].

Risk Level	Average Score
Low	0.0 - 1.0
Medium	1.1 - 2.0
High	2.1 - 3.0

Table 3.12: Criteria of risk levels assigned to DREAD values [40, p. 216]

Threat	D	R	E	A	D	Avg.	Rank
S1	2	2	2	2	2	2.0	Medium
S2	2	2	3	3	2	2.4	High
S3	2	2	3	2	2	2.2	High
S4	2	2	2	3	3	2.4	High
T1	3	2	2	3	2	2.4	High
T2	3	2	2	3	2	2.4	High
T3	1	2	2	1	2	1.6	Medium
R1	1	3	3	1	3	2.2	High
R2	1	3	3	1	3	2.2	High
I1	2	2	2	3	2	2.2	High
I2	2	2	2	1	1	1.6	Medium
I3	3	3	2	2	2	2.6	High
I4	3	3	2	2	2	2.6	High
I5	3	2	2	3	2	2.4	High
D1	2	2	2	3	3	2.4	High
D2	2	2	2	3	3	2.4	High
D3	2	2	2	3	2	2.2	High
E1	3	2	2	2	2	2.2	High
E2	3	2	2	1	1	1.8	Medium
E3	3	2	2	2	2	2.2	High

Table 3.13: DREAD risk assessment of the organisation's API before mitigations are implemented

The organisation's risk appetite states that all threat ratings determined by the DREAD analysis must not be rated higher than low. None of the evaluated threats are ranked as low, with the majority ranked as high. This indicates that the system does not conform to the predefined risk thresholds. Therefore, implementing mitigations is required to ensure the system's risk profile aligns with the acceptable risk appetite before it can be deemed safe.

S1 Actor- Unauthorised Access via Stolen Credentials

This spoofing threat is rated as medium due to its moderate damage potential and the broad impact on users. Although it's not the easiest threat to execute, given its medium reproducibility, it still poses a significant security risk due to the effects of a successful attack. The likelihood of this attack increases in the present digital landscape, where personal details can often be discovered or deduced through methods like social engineering or data leaks from other sources.

S2 Authentication Bypass in API Gateway

This threat scenario is ranked as high due to the critical role of API Gateways in managing access to APIs and the potential for widespread disruption if attackers successfully disguise themselves as legitimate users. The combination of high exploitability and high affected users, along with the relative ease of reproducing the attack under certain conditions, indicates a significant risk.

S3 Identity Spoofing in Microsoft Entra ID

The attempt to spoof legitimate identities to gain access to the system via Microsoft Entra ID is deemed high risk because of the extensive potential damage and the critical role of identity management. The scores for high exploitability and affected users reflect the challenges in executing such an attack, necessitating advanced knowledge or access. However, the impact on all users and the essential nature of identity services elevate the overall threat level. Protecting against such scenarios requires advanced authentication mechanisms and vigilant monitoring of access attempts.

S4 Session Hijacking

Session hijacking represents a high threat level, with the potential for complete account takeovers posing a direct threat to data integrity and confidentiality. While the scenario requires specific conditions for success, making it moderately difficult to reproduce, the ease of exploiting unsecured or predictable session tokens increases its feasibility. Affected users might vary, but the potential for widespread impact, particularly if high-value sessions are targeted, underscores the serious nature of this threat. Ensuring secure session management and detecting unusual session activities are critical to defending against session hijacking.

T1 CloudFront Cache Manipulation

The potential for damage in this scenario is assessed as high, given that altering cached content could mislead users, distribute malware, or tarnish the organisation's reputation. The manipulation of cached content might require specialised knowledge or tools, placing reproducibility and exploitability at a moderate level. However, given CloudFront's extensive use, any tampered content could affect a wide audience, making the number of affected users extensive. Discoverability is moderate as altered content may not be immediately apparent but could be detected through inconsistencies in content delivery or active security monitoring.

T2 WAF Rule Manipulation

Tampering with WAF rules to enable malicious requests or block legitimate traffic poses a significant threat, directly compromising the security of web applications. Specific tools or detailed knowledge of the WAF's configuration are required to

exploit this vulnerability. However, it would have significant consequences for administrators or a wide range of users if exploited. Although exploiting the vulnerability may be challenging, the outcome would be severe and could affect many individuals. Such unauthorised changes to WAF rules may take time to be evident but can be detected over time through unusual traffic patterns or during security reviews.

T3 Unauthorised Modification

The potential damage is moderate, as unauthorised alteration of user email addresses can lead to account takeover and misuse of personal information. This scenario's reproducibility and exploitability are also moderate, requiring some technical knowledge and access to victims' ID tokens. The affected users could be extensive, as attackers may manipulate multiple user accounts. Discoverability is moderate, as unauthorised changes may not be immediately apparent but could trigger suspicion through inconsistencies in user account data or reports of unauthorised access attempts. Therefore, overall, the DREAD score suggests a moderate level of risk.

R1 Lack of Audit Trail in API Gateway

The absence of an audit trail in the API Gateway presents a high risk, primarily due to the potential for unauthorised data modifications to occur undetected. The high reproducibility and exploitability reflect the access requirements and technical knowledge needed to exploit this vulnerability.

R2 Microsoft Entra ID Account Compromise Denial

This repudiation threat is deemed high risk, primarily because attackers can easily exploit common vulnerabilities or employ social engineering tactics. The significant impact on administrative users and the potential for widespread data compromise underscore the critical importance of robust authentication measures and account activity monitoring.

I1 API Gateway - Incorrect Configuration

An incorrectly configured API endpoint poses a high risk of unauthorised data disclosure, potentially affecting a broad user base. The scenario's damage potential is critically high due to the sensitive nature of the data involved. While exploiting this misconfiguration requires specific knowledge, the moderate levels of reproducibility, exploitability, and discoverability indicate the need for rigorous endpoint security configurations and regular audits to prevent such vulnerabilities.

I2 Data Transferring - Man-in-the-Middle Attack

The threat of an MITM attack during data transfer is moderately concerning due to its potential to expose sensitive information. Although reproducing and exploiting such an attack requires specific conditions and technical skills, its impact on affected users is broad and damaging.

I3 Data Endpoint Manipulation

The potential damage is rated as high, as unauthorised access to sales data from multiple online stores could lead to significant financial losses and reputational damage. Reproducibility and exploitability are also high, as attackers can easily manipulate API requests to access different store data and compile revenue information. The affected users could be extensive, as the data breach may impact numerous online retailers and their customers. Discoverability is moderate, as unauthorised data access may not be immediately apparent but could be detected through irregularities in revenue reports or data analytics.

I4 Unauthorised User Data Access

The potential damage is rated as high, as attackers exploiting the vulnerable reporting endpoint could gain access to sensitive information about reported issues, compromising user and business data integrity. Reproducibility is high, as attackers with authenticated access can easily exploit the vulnerability to access sensitive data. The exploitability is moderate, requiring some technical knowledge and authenticated access. The affected users are limited to those whose data is exposed through the compromised API endpoint. Discoverability is moderate, as unauthorised data access may not be immediately apparent.

I5 Unauthorised Access to User Data via Admin Endpoint

The potential damage is also rated as high, as attackers exploit the exposed endpoint intended exclusively for administrators to gain unauthorised access to sensitive user information. Reproducibility and exploitability are moderate, as attackers can exploit the lack of function-level authorisation checks to access user data. The affected users are limited to those whose information is exposed through the compromised API endpoint. Discoverability is moderate, as unauthorised data access may not be immediately apparent.

D1 CloudFront DoS Attack

The threat of a DoS attack on CloudFront is high due to the potential for widespread service disruption. While executing such an attack requires a botnet and technical knowledge, its impact justifies strong preventive measures like traffic monitoring and response strategies to ensure resilience against large-scale traffic floods.

D2 WAF Scripted Request Flood

Targeting WAF with scripted request floods poses a high threat by potentially degrading service performance. While such an attack is reproducible and exploitable with available tools, its overall impact might be mitigated by the WAF's defensive capabilities.

D3 API Gateway Application-layer DoS Attack

An application-layer DoS attack on API Gateway represents a high threat due to its potential to disrupt service for a broad user base. Crafting the attack requires specific knowledge but remains a feasible exploit.

E1 HTTP Integration Attack

An attack on HTTP integrations or JWT authorisers exploiting vulnerabilities poses a significant threat due to the potential for system compromise. While such vulnerabilities require technical knowledge to exploit, they are not beyond the reach of a determined attacker.

E2 API Gateway Misconfiguration Exploit

Exploiting a misconfiguration in the API Gateway is a moderate security concern due to the ease of exploitation and potential damage. Such a vulnerability could allow attackers to significantly impact the service and its users. While the damage potential and ease of exploitability make it a serious threat, the lower probability of occurrence and specific conditions required for exploitation contribute to its overall medium risk rating.

E3 Misconfigured Identity Policy

Discovering and exploiting a misconfigured identity policy within Microsoft Entra ID poses a severe risk, potentially granting attackers undue control over the system. Although the potential impact is significant, the complexity and specific conditions required to exploit such a misconfiguration contribute to its overall medium risk rating.

3.6 Discussion

The findings of the risk assessment and threat modelling for the API environment highlight significant areas of concern that need to be addressed to align with the established risk appetite and CIA triad principles. Based on the risk appetite and CIA triad assessment, it was determined that the risk levels should not exceed green in the DREAD model and yellow in the risk matrix. This would ensure

that the risks remain within acceptable limits, given the moderate sensitivity and criticality of the system's confidentiality, integrity, and availability.

Out of the 20 identified threats, 13 are categorised as critical and placed in the red section of the risk matrix, while the remaining threats are mostly ranked as substantial and placed in the orange section. This indicates that there are many high-level risks that are unacceptable under the current risk appetite. Furthermore, the threats are categorised as either medium or higher in the DREAD model, further emphasising the elevated risk levels.

These findings clearly show that the current risk levels exceed the acceptable thresholds established by the risk appetite and CIA assessments. The presence of numerous critical and substantial threats indicates significant vulnerabilities that could potentially lead to severe financial, reputational, and operational impacts if not mitigated. Given the elevated risk levels, it is important to implement mitigation strategies to reduce the threats to acceptable levels. These mitigations will be discussed in detail in the subsequent sections of the report.

4 API Security

4.1 Introduction

Authentication and authorisation are critical components of API security, serving as the gatekeepers that control access to data and functionality. Their importance cannot be overstated, as they help ensure that only legitimate users can access sensitive information and perform actions within a system. Despite their critical importance, authentication and authorisation remain the primary challenges in API security. This is underscored by the Open Web Application Security Project (OWASP) Top 10 API in which the top three security concerns are caused by authentication and authorisation misconfigurations [3]. This highlights the critical nature of implementing good API authentication and authorisation security. The following part of the report will review suitable security measures concerning authorisation and authentication for APIs using digital identities.

4.2 Digital Identities

Given the diverse nature of digital identities, applying a one-size-fits-all approach to their management is impractical, particularly regarding authentication and authorisation. For instance, a digital identity representing a human user differs significantly from one representing a server. While the former is backed by a person capable of directly engaging with MFA challenges, the latter lacks this interactive capacity. This distinction necessitates categorising digital identities into subgroups tailored to their specific interaction capabilities within the authentication and authorisation frameworks, Table 4.1 below summarises these digital identities.

Role	Description	Authentication Methods	Authorisation Techniques
API Providers	Servers or services that expose the API. They have to ensure the digital identities they're communicating with are who they claim to be to ensure they don't share data with unwanted clients.	Verification of digital identity to confirm communications are with the intended clients.	Implementation of policies to ensure data is only shared as permitted.
End Users	Representing the actual human users behind the API calls, other APIs or applications might act on behalf of the end users.	Delegated authorisation, user information verification, interactive and user friendly methods.	Granular access control based on user consent and the scope of the request.
Devices	IoT scenarios or when APIs are accessed by specific hardware.	Authentication using certificates or pre-shared keys.	Access control policies tailored to device capabilities and roles.
Bots and Automated Agents	Automated systems like crawlers or scripts that access APIs.	Rate limiting, use of API key or tokens for identification.	Access control ensures permission for tasks and prevents abuse.

Table 4.1: Digital identities involved in the authentication and authorisation process for an API

By acknowledging and addressing each digital identity type's unique characteristics and interaction capabilities, more effective and secure authentication and authorisation strategies can be devised.

4.3 Securing the API

Securing the API involves strengthening the interface to block unauthorised access and manage user permissions effectively, thereby preventing improper manipulation or access to content. It includes deploying strong authorisation and authentication frameworks to ensure access is granted only to approved entities. Further,

it involves setting up stringent access controls to restrict excessive usage and promptly implementing measures to detect and counteract malicious entities and their attack methods, minimising the risk of harm. Table 4.2 below links different mitigations that will be presented to different scenarios they help mitigate from the threat model.

Risk mitigations	Scenario
Identity and Access Management System	S1 S2 S3 T3 I5
API Gateway	S2 I1 E2
Risk-Based Authentication	S1 S3
OpenID Connect	S2 T1 T2 T3 R1 R2
JSON Web Token	S3 R1 R2
Security Assertion Markup Language (Authentication)	S1 S2 S4 T1 T2 T3 R1 R2 I2
Principle of Least Privilege	S1 S3 S4 T1 T2 T3 I1 I3 I4 I5 E1 E2 E3
Web Application Firewall	S1 S2 T1 I3 D1 D2 D3 E2
Role-Based Access Control	S2 I1 I3 I4 I5 E3
Attribute-Based Access Control	S2 I1 I3 I4 I5 E3
Just-in-time	I1 I4 I5 E3
API Keys	ALL S R1
OAuth 2.0	S1 S2 T1 T2T3 E3
Access Tokens	S1 S2
Security Assertion Markup Language (Authorisation)	I1 T1 T2 T3E3
Zero Trust	All S T3 I1 I3 I4 I5
Conditional Access	ALL S I1 I3 I4 I5 D1 D2 D3
Logging	T1 T2 T3 R1 R2
Input Validation	E1 E2 I1
Rate Limiting	I3 D1 D2 D3
Encryption	I2

Table 4.2: Overview of mitigations linked to each relevant scenario from the STRIDE analysis for securing the API

4.3.1 Identity and Access Management System

IAM helps address S1, S2, S3 and I5 by providing a solution for managing digital identities by assigning every entity with a digital identity. This involves centralising identity management in a unified location, such as a company's user directory. This allows managers to control which categories of entities have access to which resources or data [41]. IAM can also assist in implementing a SSO solution, enabling organisations to simplify login processes and avoid using multiple credentials for

individual systems and applications.

Centralising identity management also helps identify and restrict breached accounts, improving security and increasing operational efficiency through SSO services. Since one or a few servers are responsible for centralised identity management, it can be thoroughly secured and easily monitored. This makes it more difficult for unauthorised individuals to modify its content to access other accounts for their own purposes, thereby helping to reduce the risk of scenario T3.

SSO simplifies the login process by allowing users to access multiple applications or systems with a single set of credentials. This reduces the need for numerous passwords, mitigates the risk of "password fatigue," [42] and promotes the adoption of stronger, unique passwords, helping prevent attackers brute force or guess credentials which helps mitigate S1. By implementing an SSO solution within the IAM framework, organisations gain enhanced security and improved user experience. SSO consolidates multiple credentials into a single, more robust set, reducing complexity and potential security vulnerabilities. It also enables centralised control over access rights, minimising the risks of unauthorised access and aligning with the zero trust model for a secure digital environment.

A solution to IAM is Microsoft Entra ID, which is an active directory and IAM system solution with multiple tools and features to help improve security and usability [43]. One of the tools delivered by Microsoft Entra ID is an IDP solution.

4.3.2 API Gateway

When the API Gateway registers a request, it invokes microservices such as IAM to authenticate the request against internal directories. Since the API Gateway becomes the entry point for internal microservices, NIST recommends that the architecture should implement authentication, access control, load balancing, and caching as microservices that get invoked by the API Gateway. In addition, AWS recommends implementing Principle of Least Privilege (PoLP) for privileges attached to accessing, creating, reading or updating the API Gateway [44]. To reduce the risk of unauthorised access to API endpoints, it's important to properly configure the API Gateway and ensure that authentication mechanisms are working as expected. This involves avoiding configuration errors that could allow access to API endpoints that should be restricted. By doing so, scenario S2, I1 and E2 are mitigated.

4.3.3 Authentication

Ensuring secure authentication involves implementing strong measures that cannot be bypassed or tricked to grant access to unauthenticated users. This includes using methods such as Risk-Based Authentication (RBA), as well as robust authentication protocols like OIDC or SAML.

Risk-Based Authentication

RBA effectively mitigates scenarios S1 and S3 in the threat model. For scenario S1, where attackers might utilise stolen credentials or exploit weak authentication systems, RBA plays a crucial role by configuring the system to assign a higher risk score to login attempts that exhibit unusual patterns. These patterns could include access from a device or IP address that differs from the user's historical norm or login attempts at odd hours inconsistent with the user's typical behaviour. When such anomalies are detected, RBA can request additional authentication factors such as MFA codes, answers to security questions, or biometrics [45]. In this way, RBA can establish a robust barrier against unauthorised access, even in cases where the attacker possesses the user's primary credentials.

RBA's flexibility and adaptability become an asset for scenario S3, which involves attempts to spoof legitimate identities. It evaluates information such as the user's location, device attributes, and which network the request originated from. When the access attempt originates from a new device or a location that has never been associated with the user, RBA can automatically adjust its requirements, asking for further verification before granting access. Thereby significantly reducing the likelihood of successful identity spoofing.

This approach ensures that the security measures tighten only when necessary, maintaining a strict defence where the risk is most significant. Simultaneously, it streamlines the process for users when risks are low, improving usability by minimising unnecessary hurdles.

OpenID Connect

Central to leveraging the full potential of OIDC is understanding and selecting the appropriate grant of authentication based on the type of user interaction. For web applications with a server-side backend, industry standards, such as those recommended by Okta, advocate using the OAuth 2.0 authorisation code flow with OIDC [46]. This approach ensures a secure communication process by leveraging Back-channel communications to handle sensitive information and user consent.

To further secure the authorisation code flow, Okta recommends implementing PKCE for all public clients, native applications and Single page application's [47] as it ensures tokens aren't hijacked in transit. Using PKCE helps ensure that tokens aren't hijacked. For server-to-server communication, Okta recommends using the client credential grant [48].

For APIs protected by services like API Gateway, OIDC will enforce secure authentication by requiring clients to present valid ID tokens obtained after successful authentication. This ensures that only requests from authenticated users with tokens issued by a trusted IDP are accepted and mitigates S2, T1, T2 and T3. By clearly linking requests to ID tokens, it ensures each message can reliably be traced back to a user, helping mitigate R1 and R2.

JSON Web Token

Key among recommendations from RFC8725 is the necessity of algorithm verification to ascertain the secure and proper use of algorithms for encryption and signing of JWTs [49]. This process ensures that only algorithms meeting the strongest security criteria are valid. A specific recommendation is to avoid using RSA-PKCS1 v1.5 encryption algorithms due to their vulnerabilities and advise a preference for algorithms like RSA-OAEP and ECDSA, which are considered more secure. Algorithms that do not meet the established security standards, even if technically valid, should be deemed invalid [49, p. 7]. However, in scenarios where JWTs are protected by Transport Layer Security (TLS), the recommendation states that applying an additional layer of protection, such as encrypting the JWT might not be necessary [49, p. 7].

Another key practice is validating all cryptographic operations within the JWT. If any cryptographic operation fails validation, the entire token should be rejected to prevent security breaches. This includes ensuring that the entropy and randomness of keys are sufficient, particularly emphasising that human-memorable passwords should not be used for keys in HMAC algorithms. Passwords, if used, should be for key encryption rather than directly encrypting content.[49, p. 8]. This ensures that JWT uses a secure algorithm and that the algorithm specified in the JWT matches the one used for cryptographic operations, preventing attackers from impersonating another entity as detailed in S3.

Another key practice is validating all cryptographic operations within the JWT. If any cryptographic operation fails validation, the entire token should be rejected to prevent security breaches. This includes ensuring that the entropy and randomness of keys are sufficient, particularly emphasising that human-memorable passwords should not be used for keys in HMAC algorithms. Passwords, if used, should be for key encryption rather than directly encrypting content.[49, p. 8]. This ensures that JWT uses a secure algorithm and that the algorithm specified in the JWT matches the one used for cryptographic operations, preventing attackers from impersonating another entity as detailed in S3.

The issuer and subject fields within a JWT should be validated. If any fields contain unexpected values or the signature is invalid, it should be discarded, and all requests accompanied with it should be denied [49, p. 8-9]. This validation is crucial for establishing trust in the token's authenticity and ensuring it has not been tampered with. With proper validation, including verifying that the cryptographic keys used belong to the issuer and all fields are correct, it helps mitigate scenario R1.

Similarly, validating the audience field of the JWT is essential to ensure that the token is being used in its intended context and by the intended audience. This step prevents tokens from being repurposed for unauthorised access or services, reinforcing the system's security. Proper validation of these fields ensures that

JWTs fulfil their role in securing communications by preventing unauthorised use and ensuring that tokens are only accepted in their correct and intended contexts [49, p. 8]. This mitigates scenario T3 and R2 by validating these fields, as there is a clear trail of which issuer generated the token and for whom.

Security Assertion Markup Language (Authentication)

Validation of SAML assertions is critical to maintaining the integrity and authenticity of the authentication data as it travels across networks. The most common method for ensuring this integrity and authenticity is through digital signatures with certified keys. As noted by OWASP, "A digitally signed message with a certified key is the most common solution to guarantee message integrity and authentication." [50]. This process confirms that the assertions have not been altered during transit as in scenario T3.

If an organisation uses SAML, the number of times a user has to input credentials is severely limited. This reduces the chance of credentials being stolen since they are only sent by the user to the IDP a few times, depending on the lifetime of the SAML assertion, effectively mitigating S1. All further authentication is done without user interaction between the IDP and SP. These SPs need to be linked with the central IDP for SAML to work, leading to only sharing authentication info with trusted actors. These aspects of SAML make scenario I2 an unlikely event.

Encrypted SAML assertions sent by the IDP are only decryptable by a connected SP and vice versa. This makes scenario S4 a difficult task, as well as further protecting against I2. In addition, such a robust authentication measure will also mitigate S2, T1, T2 and T3. By securely validating digital signatures on SAML assertions, it's possible to ensure non-repudiation, as each message can be reliably traced back to its sender. This prevents attackers from denying their actions, addressing the threat of repudiation such as scenario R1 and R2.

4.3.4 Authorisation

To ensure secure authorisation, it is important to implement measures that prevent unauthorised access to the API's resources. This includes adopting PoLP, using WAF and JIT access, and implementing robust access control mechanisms such as RBAC and ABAC. Additionally, employing a strong authorisation protocol like OAuth 2.0 or SAML ensures the authorisation process is secure and cannot be exploited. These strategies ensure that only authorised users can access sensitive resources, safeguarding the vital parts of the API.

Principle of Least Privilege

PoLP emphasises limiting access rights for users, programs, and systems to only those resources necessary to perform their function. "Improperly constrained user and application accesses can lead to excessive disclosure of sensitive data and

promote malicious movement through the cloud." [51, p. 8]. Microsoft recommends preventing overprivileged applications by revoking unused permissions. They also suggest keeping PoLP in mind during all stages of development and reviewing permissions regularly [52].

By applying PoLP, an organisation can limit the damage if a user's account is compromised by severely restricting the attacker's ability to access sensitive information and make unauthorised changes, thereby mitigating information disclosure and the use of unauthorised resources. Even if an attacker gains access, adherence to Microsoft's recommendations for PoLP, combined with regular auditing of permissions, can effectively mitigate the issues identified in scenarios S1, S3, S4, T2, I1, I3, I4, I5, E1, E2 and E3. Microsoft advises organisations to "Audit the deployed applications periodically to identify those overprivileged." [53] as a proactive measure to reduce the attack surface and potential impact of accidents or security breaches.

Web Application Firewall

According to AWS, implementing best practices for WAF can significantly enhance the security posture of web applications hosted on their platform. This mitigates a wide range of threats and vulnerabilities outlined in the threat model [31].

WAF allows organisations to create rules to filter web traffic according to criteria such as IP addresses, HTTP headers, and body content. This functionality provides an additional layer of protection against web attacks that attempt to exploit vulnerabilities in custom or third-party web applications. By blocking common web exploits, organisations can mitigate scenarios such as T1, D1, D2 and D3.

AWS WAF also provides real-time metrics and captures raw requests, offering valuable intelligence for swiftly identifying and addressing security threats. Through real-time monitoring of web traffic patterns and metrics, organisations can detect and mitigate scenarios such as I3, T1 and E2 before they escalate or completely stop them if the WAF rules are configured properly.

Lastly, all web requests are logged by WAF, providing organisations with valuable forensic data for investigating security incidents and identifying attack patterns. This can help mitigate scenarios involving information disclosure, such as S1 and S2, by enabling organisations to identify and respond to unauthorised access attempts in real-time.

Role-Based Access Control

According to Microsoft, when assigning privileged roles in a RBAC system, one should assign roles to premade groups, not individual users [54]. IAM systems allow an organisation to create groups from which an administrator can easily add or remove users. An IAM system is not required to use RBAC but is recommended because it makes implementing RBAC much easier. Assigning roles to groups

instead of individuals helps reduce the overall number of roles in the system, leading to less administrative work and making the onboarding and offboarding processes easier [54]. After role configuration is done, thorough testing is important. Administrators need to make sure users who have been assigned roles can do precisely what has been planned for, nothing more and nothing less. Keep in mind, it is easy to create the RBAC roles and then never think about them again. Rather, create brand new roles when the need arises instead of re-iterating upon the pre-existing ones, leading to what is called "role explosion" [55]. An organisation should have regular role reviews to ensure that roles are still relevant and that users are assigned appropriate roles. This will reduce the likelihood of scenario S2 and E3 in which misconfiguration in the identity policy can be used to elevate privileges to an account.

RBAC is an effective mechanism for mitigating I1, I3, I4 and I5. By adhering to best practices and enforcing PoLP, RBAC ensures that attackers attempting to exploit vulnerabilities to gain elevated access or access to resources they shouldn't are restricted to the permissions of their assigned role. This significantly limits the potential damage and data exposure.

Attribute-Based Access Control

ABAC can be used when an organisation needs an access control method to make decisions without previous knowledge of the user or the resource in an access request. This means that a user from Organisation A could attempt to access a resource from Organisation B, and the access control system in place at Organisation B would grant or deny access based on the user's attributes correctly. This relies on attributes being consistently defined between organisations, as recommended by NIST 800-162 [56, p. 29]. One way to do this is to adopt a pre-existing attribute set, like Export Compliance-US¹. This allows interoperability between infrastructures while maintaining the same level of access control as an enclosed infrastructure. ABAC also allows fine-grained access control on the individual level, as opposed to RBAC, which mainly provides it on a per-group basis. Per-user is possible on RBAC, but not recommended [54].

Even though many aspects of ABAC promote interoperability, there should be a limit to how wide one should stretch a network. To allow an organisation to access ABAC secured endpoints hosted by another organisation can lead to complex interactions. What may have once been a simple internal API request might now require multiple attribute validations made by logically and physically dispersed entities. These interactions have a performance cost that should be taken into consideration before ABAC collaboration is initiated [56, p. 19].

By utilising ABACs potential for strict and personal access control, correctly applied to both users and API endpoints, threats such as those described in I1, I4 and

¹<https://docs.oasis-open.org/xacml/3.0/xacml-3.0-ec-us-v1-spec-en.html>

I5 will take a lot of work to achieve. An attacker would first need to figure out what attributes are required to access the target endpoint and its specific functions, then find or create a suitable user to make a successful access request. Creating a user would mean getting administrator privileges, and a new user account would be easy to spot in the logs.

Well-implemented ABAC policies would prevent the scenario described in I3 and E3. This is because even if an attacker is able to craft a script that would normally get them access to an API or initial access to the system, they would not have the required attributes for the request to be accepted or perform any actions within the system.

Just-in-time

Sensitive operations, such as changing the configuration of a production environment or accessing a highly sensitive resource, require special privileges due to their scope and potential impact [21]. Instead of implementing these as static privileges, it is recommended by the NSA to use JIT for limiting privileged access and improving tracking of privileged actions [51, p. 8-9]. This will reduce the risk associated with scenario I1, I4, I5 and E3 as an attacker would have to request additional privileges, which would be denied.

API Keys

API keys should be subject to restrictions that limit their usage to specific users and contexts, as recommended in NIST Special Publication 800-204 [57]. Such restrictions can be implemented based on the IP address or the application making the request. Moreover, the scope of API keys should be confined strictly to the functionalities and data the recipient of the key is authorised to access. Additionally, the extent of access permitted by an API key should be calibrated according to the trustworthiness of the identity verification process involved [57]. Effective governance is crucial in avoiding the pitfalls of shared API key usage, which complicates logging and necessitates frequent rotations, disrupting all users. Instituting a robust API key management system ensures that keys can be individually tracked and managed, significantly reducing these risks[58].

By allocating API keys on an individual level, the risk for repudiation attacks are significantly lowered. Attack strategies such as R1 in which a user tries to deny having performed actions to the API would be easily tracked back to the individual using the API key.

API keys are an effective method to mitigate S1, S2, S3 and S4 in the threat model, where the risk involves attackers disguising themselves as legitimate users by bypassing authentication measures. API keys authenticate and attribute each request to a known digital identity, dramatically reducing the opportunity for attackers to exploit the system anonymously and need to get their hands on more

information to break into the system as an access token or credentials won't be sufficient to gain access without the API key. Securing the API key is important, as its compromise not only fails to mitigate the threat posed by S2 but also exposes the system to the vulnerabilities outlined in S1, offering attackers an avenue to exploit both scenarios.

OAuth 2.0

Using OAuth 2.0 removes the need to share credentials with third-party applications. By only sharing credentials with the authorisation server in the API environment, several threats regarding credentials are removed, such as an attacker stealing an actor's credentials either when in transit or from insecure storage on a third-party database. A critical security aspect of token-based authorisation is the time-to-live for these tokens.

With a combination of well-managed tokens, pre-determined scopes and PKCE to prevent token hijacking, OAuth 2.0 provides a secure and structured method for users and services alike to attain authorised access to APIs.

When using OAuth 2.0 as the authorisation method, one should consider the grant type to use in every scenario. However, as attack methods and technology have advanced, it is recommended that organisations should only use two grant types [59]. The authorisation code grant type when human interaction is involved, or the client credentials grant type for machine-to-machine authorisation. This is because other grant types have been deemed too insecure and have been deprecated [60]. The authorisation code grant is more secure because of its use of PKCE and how it transfers the access token. It is transferred between the authorisation server and third parties using a secure Back-channel. All communication sent during a client credential access grant between the authorisation server and a client is done on the Back-channel and, for the same reason, is considered more secure.

When creating an application that will be accessed by several devices, OAuth 2.0 is what one would want to authorise requests. OAuth 2.0 has already established best practices for mobile [61], browser-based applications [62] and no-browser/limited input devices [63]. This makes OAuth 2.0 a desirable choice for modern, multi-device applications.

Using OAuth 2.0 for authorisation is an effective method of mitigating attacks such as those described in S1, S2, T1, T2 and T3. Using a token-based authorisation method, a third party can be granted a scope to a service on behalf of the resource owner. This is done without ever exchanging password information with a third party. This greatly reduces the opportunity for an attacker to get an actor's credentials either when in transit or stored insecurely on a third-party database. OAuth 2.0 also plays a crucial role in preventing unauthorised access and operations, directly addressing the risk of elevation of privileges, particularly in scenario E3.

Depending on token time-to-live, the threat presented by T3 could be avoided. If

tokens are configured to have short lifespans, unauthorised account modification will prove difficult, as an attacker would not be able to use an expired token. This means the user account must be re-authorised to perform any changes.

Access Tokens

According to NIST Special Publication 800-204 [57], APIs that interact with sensitive data should exclusively employ tokens signed or verified by an authoritative server for authentication purposes. Tokens must be cryptographically secured, utilising mechanisms such as HMAC schemes or being handle-based for added protection. In instances where stateless authentication tokens, like JWTs, are utilised, the lifespan of these should be minimised to mitigate the risk of misuse in the event of a token compromise. Furthermore, the secret key employed for signing the token must be securely managed; it should not be embedded within library code but instead stored as a dynamic variable, accessible through environment variables or specified in an environment data file, to enhance security [57].

Some recommendations for access tokens are first to have short-lived access tokens and long-lived refresh tokens. Okta recommends this combination, considered the best balance between security, flexibility, and usability. The second approach is short-lived access tokens and no refresh tokens. This method is the least user-friendly, as users need to continually re-authorise applications. However, there are some security advantages. The damage potential of a leaked access token is significantly lower, leading to high-risk services adopting this method [64].

Token lifespans should be as short as possible, as this would help mitigate the threats presented by S1 and S2. Due to token lifespan, an attacker with a stolen token would likely not have time to use it. If the stolen token is a JWT, the signature would be used to verify that the sender of the JWT is who they say they are and to ensure that the message wasn't changed along the way [15].

Security Assertion Markup Language (Authorisation)

In SAML, authorisation is efficiently handled through attributes in SAML assertions, these are statements from the IDP regarding a user. These attributes detail the user's roles, permissions, and entitlements, enabling the SP to make informed access control decisions and mitigate scenarios like I1. By specifying what resources a user is allowed to access and the operations they can perform, SAML plays a crucial role in preventing unauthorised access and operations, directly addressing the risk of elevation of privileges and tampering, particularly in scenarios T1, T2, T3 and E3.

Since the authorisation part of SAML travels in the same assertion as the authentication part of SAML assertions, the same recommendations for validation and encrypting apply, see subsection 4.3.3.

4.3.5 Zero Trust

Adopting a zero trust architecture, as the NSA advocates, is crucial for enhancing an organisation's security measures [65]. This approach requires rigorous verification of all users, devices, and data flows before granting access. Organisations should operate under the assumption of a breach, scrutinising all network interactions and continuously monitoring activities for suspicious behaviour [65].

Zero trust mandates a secure authentication and authorisation process for accessing resources. The architecture should prioritise protecting critical data, assets, applications, and services and enforcing access control policies across all platforms. Comprehensive visibility into network activities is vital for promptly detecting and responding to potential security breaches [65].

A cornerstone of zero trust involves establishing comprehensive visibility into all network activities through diligent inspection and logging of all traffic and resource accesses. This practice is vital for enabling analytics that can pinpoint and act upon suspicious activities, thereby enhancing the organisation's ability to detect and respond to anomalies and potential security breaches promptly [65].

Integrating zero trust principles into the organisation's API authentication and authorisation framework significantly reduces risks associated with unauthorised access and data breaches. Specifically, it helps mitigate all spoofing-related scenarios and T3 as it limits what the attacker can do if they manage to spoof a user or gain control over their account. Zero trust also helps mitigate I1, I3, I4 and I5 as it requires rigorous authentication and authorisation controls before accessing any resources, preventing any attacker from accessing information or functions they shouldn't have.

Conditional Access

According to the NSA, mitigations such as secure passwords, MFA, and login tokens are not enough to protect user accounts [51, p. 6]. Organisations should implement Conditional Access (CA) controls in their API ecosystem to enhance security practices. These controls enforce access restrictions based on contextual factors, complementing traditional authentication methods. The NSA recommends leveraging CA to limit privileged access and improve tracking of privileged actions [51, p. 6]. Additionally, by implementing CA controls, organisations can enforce access policies based on various factors, aligning with the principles of zero trust to verify every request and minimise the risk of unauthorised access.

Enforcing CA policies based on geographical location or IP address is another best practice. By restricting access to resources based on location, organisations can help mitigate scenarios such as all spoofing scenarios, T3, I1, I3, I4, I5 and E3. For example, if an organisation only operates in specific regions, it can create policies to block access attempts from locations outside those regions. This ensures that

only users connecting from authorised locations can access sensitive resources, reducing the risk of unauthorised access [66].

Performing device compliance checks as part of CA policies is essential for enhancing security. Only compliant devices with up-to-date security configurations should be able to access sensitive resources. Organisations can reduce the risk of data breaches and unauthorised access by enforcing these compliance checks regularly. This would prevent compromised or non-compliant devices from accessing sensitive resources [67].

Integrating CA with real-time risk detection capabilities is crucial for proactive threat management. By leveraging real-time risk detection, organisations can mitigate I1, D1 D2 and D3 scenarios by detecting and responding to suspicious behaviour as it occurs. Real-time risk detection allows organisations to identify and block access attempts from potentially compromised or malicious entities, minimising the impact of potential attacks and protecting sensitive resources [67].

4.3.6 Other Considerations

The following section discusses security measures addressed in the threat model that are separate from API authentication and authorisation. However, these measures are important for enhancing basic API security and strengthening authentication and authorisation techniques and are therefore worth mentioning.

Logging

AWS and Microsoft recommend collecting logs in a centralised logging system. The logs should include sign-in activity, audit logs, and risk events [68]. The organisation should decide how long and the type of logs they want to save based on their security requirements. Logs should have limited access; only people with a reason to view the logs should have access. Tools should be implemented to help automate logging, check log integrity, and review logs [69].

A logging solution, as mentioned above, would help mitigate R1 and R2 by providing logs that show the changes made by the user, including which resources were accessed and the user's location at the time of login. Logging would also aid in detecting breaches and identifying any unauthorised alterations, helping to address scenarios like T1, T2 and T3.

Input Validation

Input validation should be performed as early as possible to prevent malicious data from being processed by the API. It's crucial to handle all data received from users with caution, assuming it could be malicious. This principle also applies to data sent over Back-channels from suppliers, partners, vendors, or regulators, as these sources may have been compromised. While input validation is important

for preventing Cross-Site Scripting and SQL injection attacks, it should not be the main defence against these threats [70]. Proper implementation of input validation can help address specific vulnerabilities, especially scenarios E1, E2 and I1, by adding an extra layer of security to complement other defensive measures. AWS WAF is one tool that offers input validation, the managed rules `AWSManagedRulesSQLiRuleSet` [71] and `AWSManagedRulesKnownBadInputsRuleSet` [72] protect against known bad inputs and patterns associated with SQL exploitation.

Rate Limiting

AWS advises utilising rate limiting to safeguard against DoS and Distributed denial-of-service (DDoS) threats, as highlighted in scenarios D1, D2 and D3. Rate limiting, or throttling, restricts the number of requests a user can make within a certain timeframe, which helps to block potential attackers by ensuring they're blocked when sending too many requests. Rate limiting can throttle the traffic from individual requestors, preventing a single IP address or API consumer from consuming excessive resources and impacting service for others. By using rate limits, services can ensure that the API is available to legitimate users while reducing costs and managing the load of accepted requests, even during unexpected surges in traffic. Rate limiting also helps against scenarios such as I3, where a script is used to send many requests extracting data from different users. One tool that offers rate limiting is AWS WAF, which provides rule sets for establishing basic rate limits and rate limits based on API keys and IPs [73]. Another tool to help prevent DDoS attacks is AWS Shield².

Encryption

NSA recommends encrypting all data in transit, using secure protocols like TLS 1.2 or higher [51, p. 6]. Enforcing end-to-end encryption using TLS [51] and regularly updating and managing certificates can help prevent I2 and protect against MITM attacks. TLS certificates should not be stored in plain text, and a secrets manager should be used to manage certificates. User certificates used in public key infrastructure (PKI) must be handled carefully so as not to be obtained by unwanted actors and, therefore, compromised [51].

4.4 Securing the Digital Identity

Securing the digital identity includes verifying that users are who they claim to be and safeguarding their accounts or sessions against unauthorised access. Several measures are implemented to achieve this, such as MFA, PKCE, and session management. Table 4.3 links mitigations presented below to different scenarios from the threat model.

²<https://aws.amazon.com/shield/>

Mitigation	Scenario
Multi Factor Authentication	S1 S2 S3
Proof of Key Code Exchange	S2 S3 S4 T3 R1
Session Management	S2 S4 I2

Table 4.3: Overview of mitigations linked to each relevant scenario for securing the digital identity

4.4.1 Multi Factor Authentication

To address scenarios S1, S2, and S3, the implementation of MFA and robust password policies are essential. According to NIST's Special Publication 800-63-4 [7], effective MFA systems must utilise at least two authentication factors. Which can be integrated in one of the following ways:

1. The system can be designed to require the presentation of multiple factors to the verifier.
2. Alternatively, certain factors may safeguard a secret, which is then presented to the verifier.

The list below exemplifies some of the combinations of authentication factors:

- **Something you know:** Password/Key Pair
- **Something you have:** Secure Device
- **Something you are:** Biometric Data

For example, a system might combine a memorised secret with a physically separate device to authenticate a user, or a hardware token secured via biometric verification might produce a cryptographic key for authentication. Knowledge-based authentication, requiring answers to personal questions, does not qualify as a secure factor under NIST guidelines. Similarly, biometrics alone are insufficient as they do not constitute secrets [7].

Agencies like the NSA and CISA recommend phishing-resistant MFA methods such as PK-based FIDO/WebAuthn Authentication or PKI-based MFA (e.g., CAC/PIV cards) to enhance security [51]. Implementing stringent MFA and password policies greatly enhances security, making it challenging for attackers to brute force or guess passwords. Should credentials become compromised, MFA provides an additional security layer, preventing unauthorised access. Furthermore, these security requirements can be centralised and enforced through CA, ensuring consistent application across all system access points [67].

4.4.2 Proof of Key Code Exchange

For better security when using the authorisation code grant, it is recommended by the creators of OAuth 2.0 to use the PKCE extension [14]. By utilising PKCE,

an organisation can mitigate threats presented by S2, S3 S4 and T3 in the threat model. If PKCE is implemented, even a successful theft of an authorisation code will result in an unsuccessful attack. The time an attacker requires to recreate the code challenge is longer than it takes to use the authorisation code it is connected to. It does not matter if the attacker later guesses the code challenge since an authorisation server should never accept an already used authorisation code. Using PKCE, only the original requester will be granted access using any access token. Suppose the authorisation server tracks when and where requests originate. This will mitigate the threat presented by scenario R1, where an authenticated user performs an action but later denies making it. An administrator would know who performed the malicious act by linking the access token to a specific device and time.

4.4.3 Session Management

To prevent sessions from being hijacked as described in scenario S4 and MITM attacks as described in scenario I2, it's essential to implement HTTPS and use secure cookie attributes such as 'HttpOnly' and 'Secure' to protect session tokens from being intercepted during transit and by client-side scripts [74]. Another method to prevent hijacking is using random, unpredictable tokens that expire after a short duration. Monitoring for unusual session patterns could also help detect hijacking attempts, strengthening the system.

4.5 Compatibility Considerations

In the discussion of compatibility issues between the different digital identities and API security measures, it's important to understand the unique characteristics and capabilities of each of the digital identities presented at the start of chapter 4; API providers, end users, devices, and bots. These identities often interact with security measures in complex ways, which may enhance or hinder system security.

These digital identities can broadly be classified into human users and machines initiating requests. This classification necessitates distinct compatibility considerations. For instance, human users can interact more dynamically with authentication systems, such as responding to MFA challenges, which Microsoft reports can prevent up to 99.9 per cent of unauthorised account access attempts [75].

Conversely, machines are limited in their interaction with authentication processes. They cannot use the authorisation code grant, which is more secure because it requires credential-based user authentication. Instead, machines typically use the client credentials grant, which depends solely on possessing a client secret from the authorisation server. Implementing PKCE is problematic for machines, as they do not support interactive authentication flows necessary for the authorisation code flow, which is the only flow that supports PKCE.

A downside to SAML is that it requires a lot of configuration contributing to the setup of the authorisation and authentication process, and XML setup. Interoperability between domains also requires several administrative agreements [76, p. 11], and SAML may be incompatible with many mobile and desktop applications [77].

The implementation of some security measures also needs to be taken into account. One is the frequency of when MFA and re-authentications are used. Frequent use of these improves security but comes at the cost of user experience. So, finding the right balance between usability and security is important.

Security measures that require specialised configuration like, WAF, CA, RBAC, ABAC, rate limiting, RBA and logging are also to be considered. They must be configured accurately to prevent the obstruction of legitimate traffic, avoid overly restrictive access for legitimate users, and ensure precise logging of security events. This precision is vital to maintaining both functionality and protection in digital environments.

4.6 Risk Assessment

4.6.1 Risk Matrix After Mitigations

Several strategies to enhance API authentication and authorisation security have been presented so far. The subsequent section will integrate these strategies into the hypothetical organisation described in chapter 3. This aims to demonstrate the efficiency of these strategies in mitigating identified threats. To visualise this, an updated risk matrix, based on the API presented in the threat model with all the security measures implemented, will be employed to depict the changes in the overall threat level. The terminology used for risk levels, consequences and probabilities are taken from section 3.5.

Probability	Consequence				
	Insignificant 1	Minor 2	Significant 3	Major 4	Severe 5
Certain 5					
Likely 4					
Moderate 3		E1			
Unlikely 2	S2	S1 S4 T3 I2 I3 I4 E2 E3	T1 T2 I1 I5 D1 D2 D3		
Rare 1		S3 R1 R2			

Table 4.4: Risk matrix illustrating consequences versus probability for each scenario after mitigations are implemented

After implementing various risk mitigations, the risk levels across most scenarios

have decreased noticeably. Initially, many threats had high probabilities and consequences, reflecting the system's significant vulnerabilities. Initially, there were 14 risk scores which were categorised as critical.

The risk mitigations have significantly improved the security posture by reducing the probability and the consequence of these threats. For instance, the probability of unauthorised access via stolen credentials S1 dropped from a moderate probability to unlikely, reducing the overall risk score from a critical level to an important level. This is due to the effective implementation. In scenarios like session hijacking S4 and unauthorised modification T3, risk mitigations such as JWT, PKCE, and WAF have helped lower the risk by securing session management and access controls. As a result, the overall risk levels for these scenarios have decreased significantly.

Most risks still have a significant consequence level due to the severe consequences they can cause, even though their probabilities have been reduced. By implementing these mitigations, the risk matrix satisfied the organisation's risk appetite. However, to fully satisfy the organisation's risk appetite, the DREAD values must also not be above low level.

4.6.2 DREAD After Mitigations

Implementing various risk mitigations has significantly reduced the DREAD values across all identified threats in the API environment. Prior to implementing these mitigations, most threats were rated as medium to high risk, indicating substantial vulnerabilities that could lead to severe consequences such as unauthorised access, data breaches, and service disruptions.

For example, mitigations like the use of IAM, WAF, API Gateway, and CA controls have effectively lowered the reproducibility and exploitability scores by introducing stronger authentication and authorisation protocols, input validation, and rate limiting. These measures make it more difficult for attackers to reproduce and exploit vulnerabilities within the system.

Additionally, adopting principles like PoLP, zero trust architecture, and comprehensive logging has reduced the damage potential and affected users scores. By limiting access rights to the minimum necessary and continuously verifying all access requests, these strategies minimise the impact of any successful attacks, thereby protecting sensitive data and reducing the number of affected users.

Despite these improvements, it's important to note that while the overall risk levels have been lowered to medium, the organisation's risk appetite dictates that no threat should be evaluated by the DREAD model at a ranking higher than low. This means that further risk mitigation efforts are necessary to fully align with the organisation's risk appetite.

Threat	D	R	E	A	D	Avg.	Rank
S1	0	1	1	1	2	1.0	Low
S2	1	1	0	1	0	0.6	Low
S3	1	0	0	1	0	0.4	Low
S4	1	1	1	1	1	1.0	Low
T1	3	0	0	3	0	1.2	Medium
T2	3	0	0	3	0	1.2	Medium
T3	1	0	0	1	1	0.6	Low
R1	1	0	0	1	0	0.4	Low
R2	1	0	0	1	0	0.4	Low
I1	1	2	1	3	1	1.6	Medium
I2	1	0	0	1	1	0.6	Low
I3	2	0	0	2	0	0.8	Low
I4	2	0	0	2	0	0.8	Low
I5	3	0	0	3	0	1.2	Medium
D1	2	1	2	1	3	1.8	Medium
D2	2	1	2	1	3	1.8	Medium
D3	2	1	1	1	1	1.2	Medium
E1	1	1	1	2	1	1.2	Medium
E2	2	0	0	1	1	0.8	Low
E3	2	0	0	1	1	0.8	Low

Table 4.5: DREAD risk assessment of the organisation's API after mitigations are implemented

4.6.3 Bowtie Modelling

The Bowtie model is a visual tool used to analyse and manage risks. At the centre of the model is the event, which represents a potential incident or occurrence, in this report the events are the six categories from the STRIDE analysis. On the left side of the bowtie are various threats that could lead to the event that is identified. Adjacent to these threats are probability-reducing barriers, measures implemented to decrease the likelihood of the event occurring. The potential consequences of the event are detailed on the right side of the model. Next to each consequence are consequence-reducing barriers. These barriers are designed to mitigate the event's impacts if they occur. This structured approach helps organisations visualise and manage the risks associated with their operations. The risk levels described in Table 3.5 and used in Table 4.4 and Table 3.6 have been used to apply mitigations and are referred to in the discussion of each bowtie model.

Spoofting

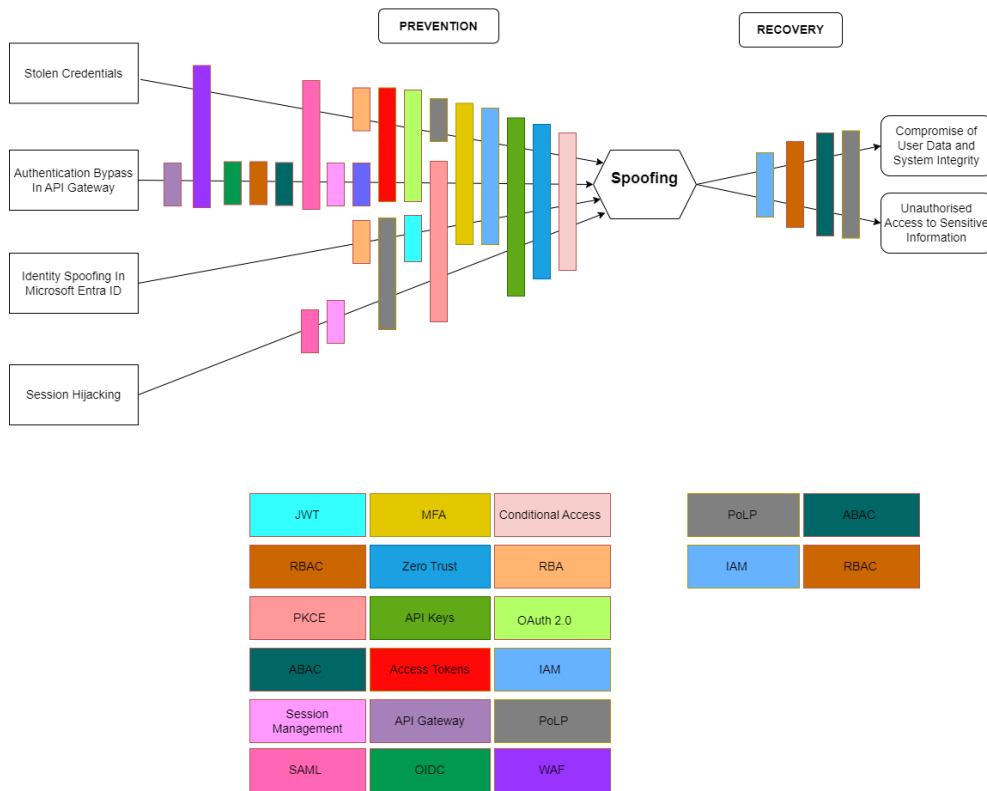


Figure 4.1: Bowtie diagram showing preventions and recoveries related to spoofing from the STRIDE analysis

To thwart spoofing attempts, preventive measures are crucial in safeguarding against unauthorised access to sensitive information and compromising user data and system integrity. Before implementing risk mitigations, the scenarios S1, S2, S3, and S4 had moderate probabilities and major consequences, leading to critical risk levels. These were at unacceptable risk levels where risk mitigations were required. The introduction of various risk mitigation measures has notably improved the security landscape. Systems like IAM have centralised identity management, assigning digital identities and implementing SSO, which reduces the probability of stolen credentials and enhancing security. Configured to authenticate requests against internal directories, an API Gateway ensures that only authorised requests are processed, reducing the probability of scenarios such as S2. Furthermore, RBA assigns higher risk scores to suspicious login attempts, lowering the probability of stolen credentials and identity spoofing. JWTs help mitigating S3 by using signature validation. Additionally, OIDC requires valid ID tokens for API access, ensuring authenticated requests and reducing identity spoofing, mitigating scenario S2.

Further strengthening security, SAML provide strong cryptographic protection, reducing the probability of scenario S4. PoLP limits access rights, minimising the damage of compromised accounts, while RBAC and ABAC restrict user access to only necessary resources, reducing the risk of unauthorised access attempts. OAuth 2.0 and API key secure API access, decreasing the probability of successful spoofing attempts. A zero trust architecture rigorously verifies all access requests, significantly lowering the risk of spoofing-related scenarios. Additionally, centralised logging aids in tracking and analysing user activities, mitigating the impact of unauthorised access attempts, and CA policies enforce contextual restrictions, further enhancing security. These risk mitigations have effectively reduced the probability and consequence of spoofing scenarios, making the system more secure and resilient against potential attacks. This has allowed all the spoofing scenarios to reach an insignificant risk level in the risk matrix, which means that there are currently no need for further mitigations measures.

Tampering

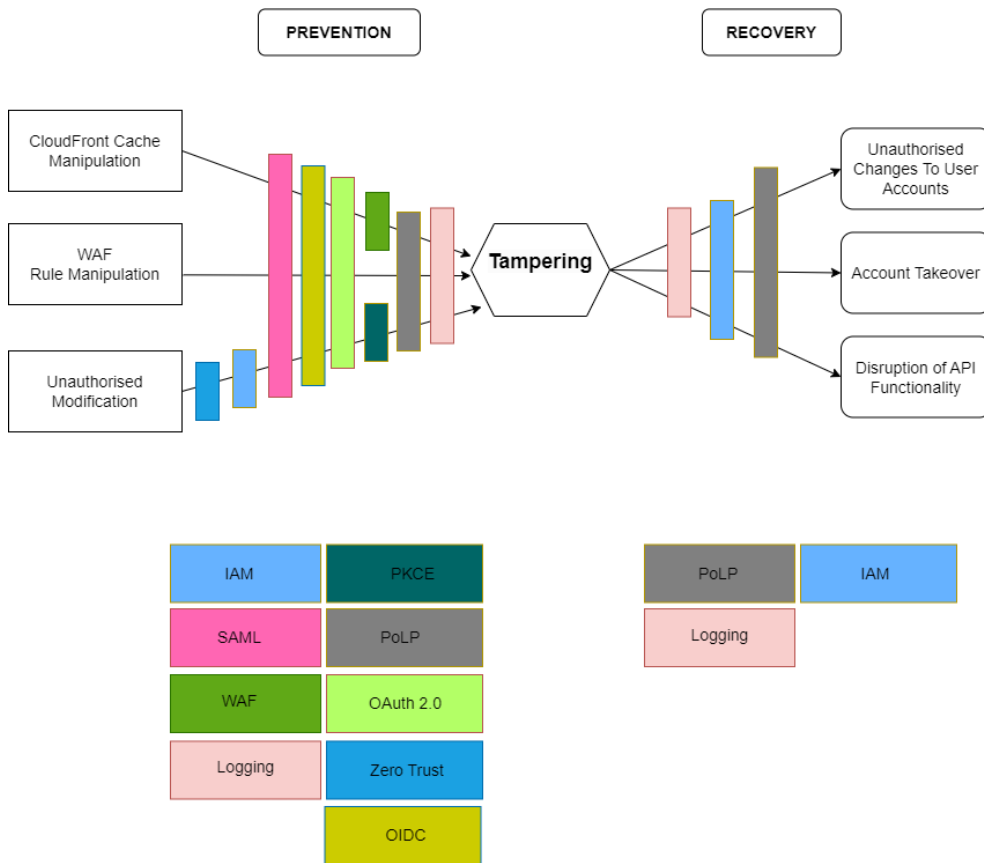


Figure 4.2: Bowtie diagram showing preventions and recoveries related to tampering from the STRIDE analysis

Before implementing risk mitigations, the scenarios for tampering T1 and T2 posed a critical threat and T3 posed a substantial threat. These high-risk levels underscored the urgency of deploying effective risk mitigation strategies to safeguard against unauthorised modifications, WAF rule manipulations, and CloudFront cache manipulation. These could lead to unauthorised changes to user accounts, account takeovers, and disruption of API functionality.

For T1, the implementation of WAF and logging helps to prevent attackers from manipulating the CloudFront cache by filtering out malicious requests and providing a record of all access attempts and changes, which enhances the ability to detect and respond to suspicious activities promptly. This reduces the likelihood of unauthorised content being stored and distributed, mitigating the risk of user account changes and API disruptions. For T1, T2 and T3, applying PoLP ensures that only authorised users have access to modify WAF rules, thereby reducing the chance of malicious actors altering these rules. Logging further aids in monitoring and auditing changes to WAF configurations, allowing for swift detection and correction of any unauthorised modifications. This combination lowers the probability of harmful requests passing through the WAF and blocking legitimate traffic, mitigating the risk of unauthorised account changes and API functionality disruptions. The deployment of a comprehensive IAM system, alongside authentication protocols like SAML or OIDC, strengthens the security of the authentication process, reducing the likelihood of unauthorised modifications. Implementing PKCE with OAuth 2.0 enhances token security, preventing token hijacking and unauthorised account modifications. Additionally, zero trust principles enforce strict verification for all access requests, and logging provides an audit trail for all actions, facilitating rapid detection and response to tampering attempts. Post-attack mitigations such as PoLP and IAM help contain the impact by restricting access and quickly disabling compromised accounts.

These mitigations have effectively reduced the risk levels of T1 and T2 to an important risk level and T3 to an insignificant risk level, thus enhancing overall system resilience against tampering threats.

Reputation

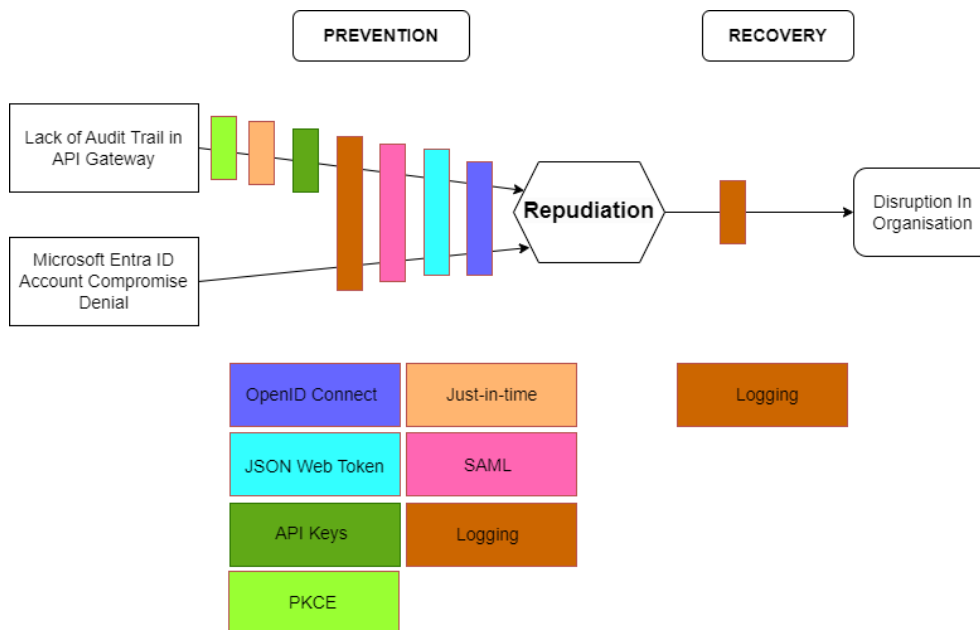


Figure 4.3: Bowtie diagram showing preventions and recoveries related to repudiation from the STRIDE analysis

As seen in Figure 4.3, there are two threats with one main potential consequence. Before applying risk mitigations, the scenarios R1 and R2 for repudiation posed substantial threats to the system.

The introduction of various risk mitigation measures has significantly enhanced the security posture against repudiation threats. For R1, implementing PKCE, API keys, SAML, OIDC, and comprehensive logging has greatly reduced the risk of an authenticated user denying actions performed via the API. PKCE ensures that only the person requesting can gain a token, which implies the one using the token is the one who requested the token. API keys authenticate and track API requests to specific users, ensuring the person making the request is who they are. Logging creates an audit trail for all actions, ensuring accountability and enabling swift detection of unauthorised modifications. Both SAML and OIDC authenticates the user in the system, and the changes they made can be directly linked to their logged in session.

Overall, these risk mitigations have effectively reduced the risk levels, with R1 and R2 now having an insignificant risk level, making the system more secure and resilient against repudiation threats.

Information Disclosure

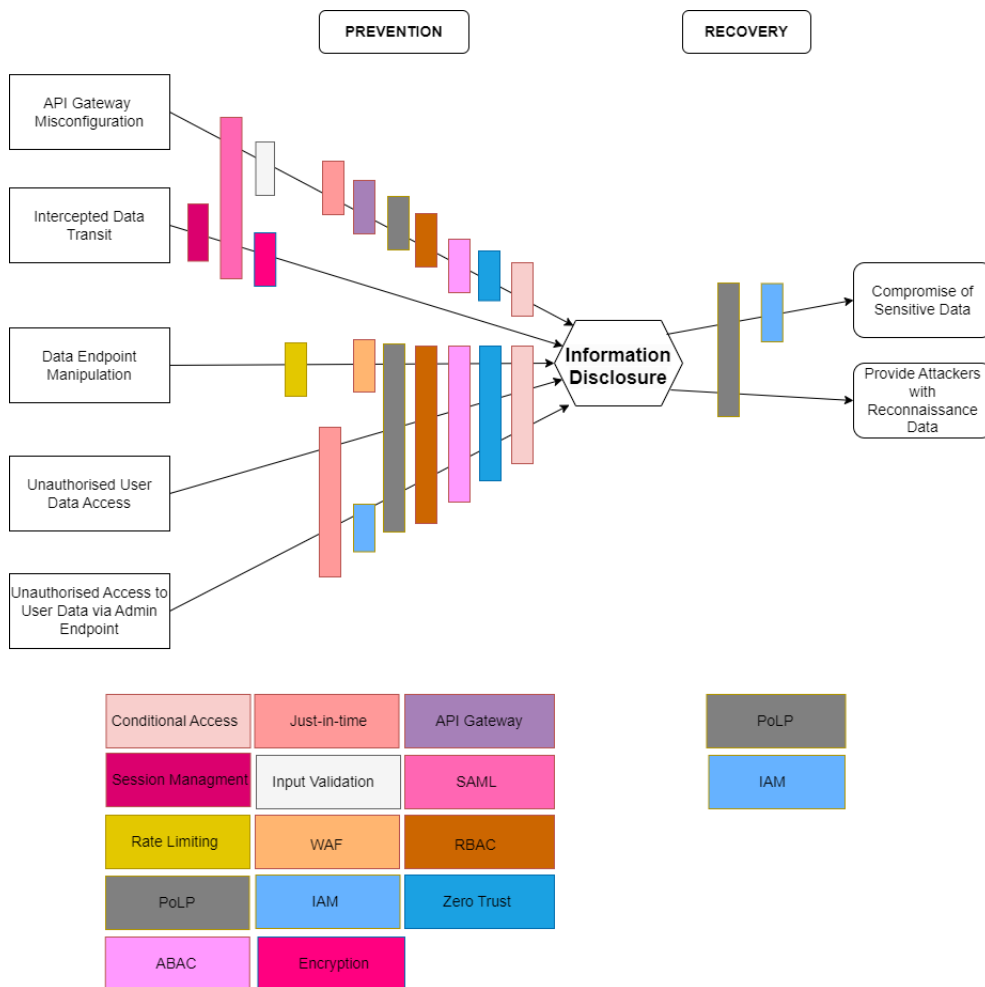


Figure 4.4: Bowtie diagram showing preventions and recoveries related to Information disclosure from the STRIDE analysis

As shown in Figure 4.4, there are five possible threats to achieving information disclosure. I2 and I3 posed a substantial threat while I1, I4 and I5 pose a critical threat. This underlines the strong need for risk mitigations to reduce the chances of compromise of sensitive data and to provide attackers with reconnaissance data. Multiple mitigations have been suggested for each risk to reduce their risk level significantly.

For I1, implementing an API Gateway ensures that only authorised requests are processed, filtering out unauthorised access attempts and mitigating the risk of incorrect configurations leading to data disclosure. In I3, PoLP ensures that only authorised users can access data endpoints, preventing data manipulation. For I1,

I4 and I5, PoLP limits user access to sensitive data, reducing the risk of unauthorised access to user and admin endpoints.

RBAC is applied to multiple scenarios. In I1, it enforces access control policies based on user roles, preventing unauthorised users from accessing sensitive data. For I3, RBAC ensures that only users with appropriate roles can manipulate data endpoints. In I4 and I5, RBAC restricts access to sensitive data based on user roles, further securing the API.

ABAC is another important mitigation. For I1, ABAC uses user attributes to make dynamic access decisions, preventing unauthorised access. In I3, ABAC ensures that only users with the appropriate attributes can access sensitive data. Similarly, in I4 and I5, ABAC applies fine-grained access control, reducing the risk of unauthorised access to data endpoints.

JIT access is implemented for I1 to provide temporary, time-bound access, reducing the window for unauthorised activities. It also applies to I4 and I5, minimising the risk of long-term unauthorised access by granting privileges only when necessary. SAML authorisation and authentication is used in I1 to ensure that only authenticated users can access the API. In I2, SAML authentication provides strong security mechanisms by encrypting data sent between the IDP and the SP and vice versa, protecting against MITM attacks.

A zero trust architecture is applied to I1 by requiring continuous verification of all access requests, significantly reducing the risk of unauthorised access. It is also used in I3, I4, and I5 to maintain strict access control and ensure security. CA is enforced in I1 to impose additional restrictions based on contextual factors, further enhancing security. It is also used in I3, I4, and I5 to limit access based on conditions such as location, device compliance, and user roles. By capturing raw requests, WAF can detect and filter out malicious traffic, preventing manipulation of endpoints, mitigating scenario I3.

Rate limiting is applied in I3 to control the number of requests a user can make, preventing abuse and reducing the risk of data extraction through automated scripts.

After the attack, IAM systems can quickly disable compromised accounts and reassign access permissions, limiting further damage. Utilising PoLP ensures that if an attacker gains access to the system, they have limited privileges to perform actions. PoLP in recovery also ensures that restored accounts or services are only given minimal necessary access.

By introducing these mitigations, the risk levels have significantly been reduced to important for I1 and I5. I2, I3 and I4 has been reduced to a risk level of insignificant.

Denial of Service

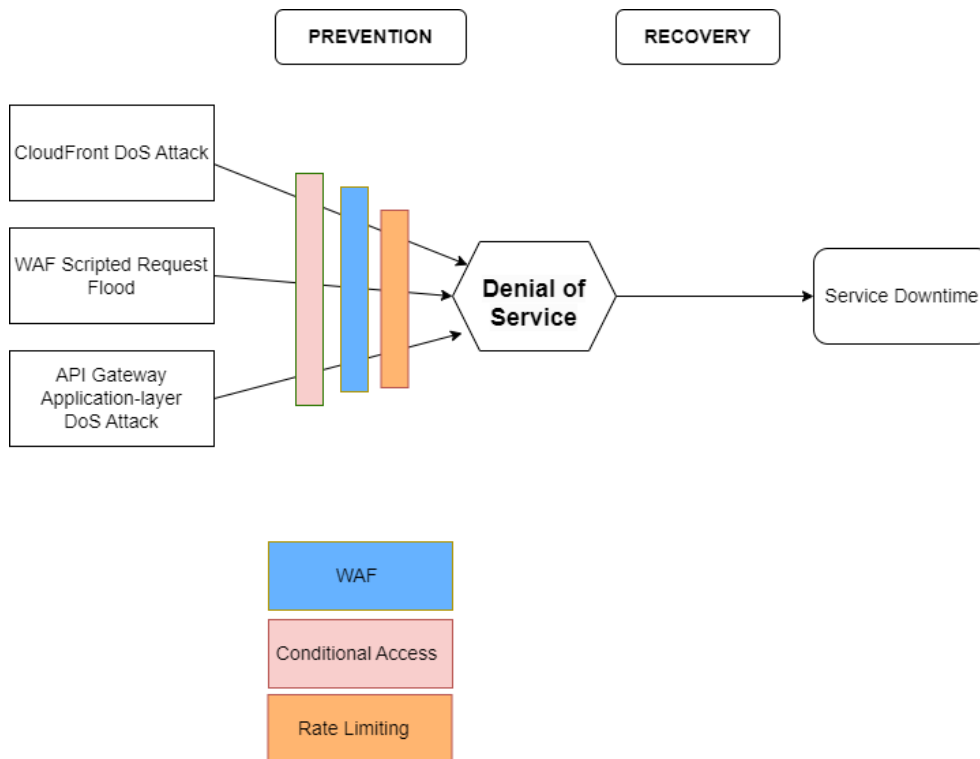


Figure 4.5: Bowtie diagram showing preventions and recoveries related to DoS from the STRIDE analysis

Before the implementation of risk mitigations, scenarios D1, D2, and D3 had a critical risk level. This critical risk level highlights the urgent need for effective mitigation measures to address these DoS scenarios.

For D1, WAF prevents the CDN from being overwhelmed by a large-scale DoS attack by blocking excessive and harmful traffic. In D2, WAF rules identify and block scripts generating malicious HTTP requests, conserving computational resources. For D3, WAF filters out complex, resource-intensive requests that could slow down the backend, maintaining the system's performance and availability. In D1, CA ensures that only legitimate traffic reaches the CDN, preventing unauthorised access and reducing the risk of overwhelming the system. For D2, CA limits the execution of WAF rule evaluation processes to legitimate users, conserving computational resources. In D3, it prevents unauthorised users from executing complex, resource-demanding processes, protecting the API Gateway from application-layer DoS attacks. Rate limiting helps in all the DoS scenarios as it limits the number of requests that can be sent in a certain timeframe, and since it blocks the requests, it helps reduce the consequences of a DoS attack. Instead of bringing down the system, it could instead result in some users not being able to access the system

due to a large amount of the requests being blocked by rate limiting. No recovery methods have been identified to reduce the consequences after a DoS attack with the mentioned mitigations.

After applying mitigations, all DoS scenarios had their risk levels reduced to important.

Elevation of Privilege

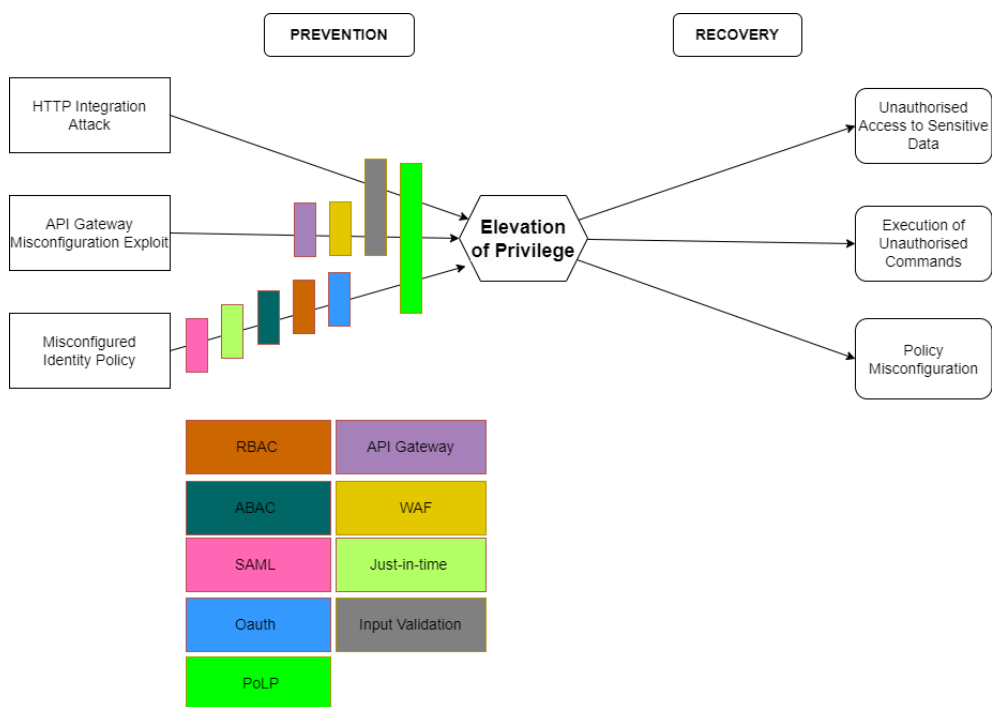


Figure 4.6: Bowtie diagram showing preventions and recoveries related to elevation of privilege from the STRIDE analysis

As seen in Figure 4.6, there are three possible threats to achieving elevation of privilege. For E1 and E3 the risk posed a critical level. E2 had a substantial risk level.

Before such an attack occurs, several preventive controls can be implemented. For E1, implementing input validation ensures that only properly formatted data is processed, preventing attackers from exploiting vulnerabilities within HTTP integration. By blocking malformed or malicious input, input validation reduces the risk of executing unauthorised commands or accessing sensitive data beyond the intended scope, lowering the likelihood of unauthorised access to sensitive data. This mitigation can also be used to mitigate E2 in the same way. The API Gateway ensures proper configuration and secure handling of API requests, preventing unauthorised access and command execution. WAF filters out malicious

requests before they reach the backend, blocking potential code injections. Input validation further strengthens these defences by ensuring only valid data passes through the system. Together, these measures significantly reduce the risk of an attacker exploiting misconfigurations to elevate their privileges. For E3, RBAC, ABAC, JIT access, and SAML authorisation have been applied. RBAC restricts access based on user roles, ensuring users can only access resources necessary for their job functions. ABAC provides fine-grained access control based on user attributes, actions, and environmental conditions, further limiting unauthorised access. JIT access grants temporary, time-bound privileges, reducing the window for potential misuse of elevated privileges. SAML authorisation or OAuth 2.0 ensures that only authorised users can access sensitive resources. These combined measures effectively reduce the risk of privilege escalation due to misconfigured identity policies, preventing unauthorised access to sensitive data and executing unauthorised commands.

After implementing these mitigations, E1 reached an important threat level, while E2 and E3 have been reduced to an insignificant risk level, which is acceptable according to the risk appetite.

4.6.4 Discussion

The risk assessment and threat modelling exercise for the API environment has demonstrated significant improvements in the security posture after implementing various mitigations. The risk levels for many threats have been substantially reduced, aligning them more closely with the organisation's risk appetite.

How the Risk Stands

Prior to implementing mitigations, the risk assessment identified numerous threats as critical or substantial, posing unacceptable risks to the API system. The risks have been significantly mitigated with robust security measures such as IAM, WAF, API Gateway, CA, PoLP, zero trust architecture, and comprehensive logging. As a result, most threats have been downgraded to medium or low-risk levels, as reflected in the updated DREAD model and risk matrix.

Despite the successful mitigation of many threats, some residual risks remain. Residual risk refers to the level of risk that persists even after all mitigation measures have been applied [40, p. 22]. The updated DREAD model shows that most threats have been reduced to low, with a few only being reduced to medium. However, no threats have been eliminated, underscoring the inherent nature of security risks in complex systems.

Viability

The concept of residual risk is critical in evaluating the system's viability. The current residual risks are mostly within the organisation's risk appetite, with a

few being barely over. It is, therefore, important to keep implementing mitigation strategies and monitor the risks continuously. The risk appetite was moderate, balancing the need for security with operational functionality. This moderate risk appetite reflects the confidentiality (internal), integrity (required), and availability (2 days) ratings of the API system, indicating that while some controls are necessary, the overall sensitivity and criticality of the data and operations are not extremely high.

The risk matrix categorises all threats as important or lower, indicating they are within the acceptable range for the defined risk appetite. However, the DREAD model, which uses a stricter three-level ranking system compared to the risk matrix's four levels, aims for all threats to be ranked as low but does not achieve this. The DREAD model is more detailed than the risk matrix and may contribute to its higher risk ratings compared to the risk matrix. Additionally, the differing risk levels between the two models make direct comparison challenging, resulting in the DREAD model not achieving its defined risk appetite while the risk matrix does. It may be beneficial to rework the rankings of the DREAD model and the risk matrix to better align their risk levels and ensure consistent results.

Given the moderate CIA values assigned to the API system, the stringent risk appetite is somewhat justified to ensure robust protection. However, there is an argument for a more balanced approach that considers operational efficiency and practical risk management.

5 Proof of Concept

5.1 Introduction

The main focus of the PoC is to demonstrate fine-grained access control based on the digital identity of the actor making the request, which will either be an end user or a machine. To do this, there will be two different implementations showcasing how the API provider can authenticate and authorise digital identities for the different end users and machines.

As illustrated in Figure 5.1, the implementation will have three entities representing end users. These three are developer, admin and financial staff, which are groups within the organisation. One last entity will be made, representing machine-machine communication. The machine entity is a server within the organisation. These four entities will have a corresponding API endpoint, with access restrictions tailored to their digital identity. Notably, the admin group will have access to all the endpoints for the end user groups, demonstrating how access can be granted based on the entity's digital identity.

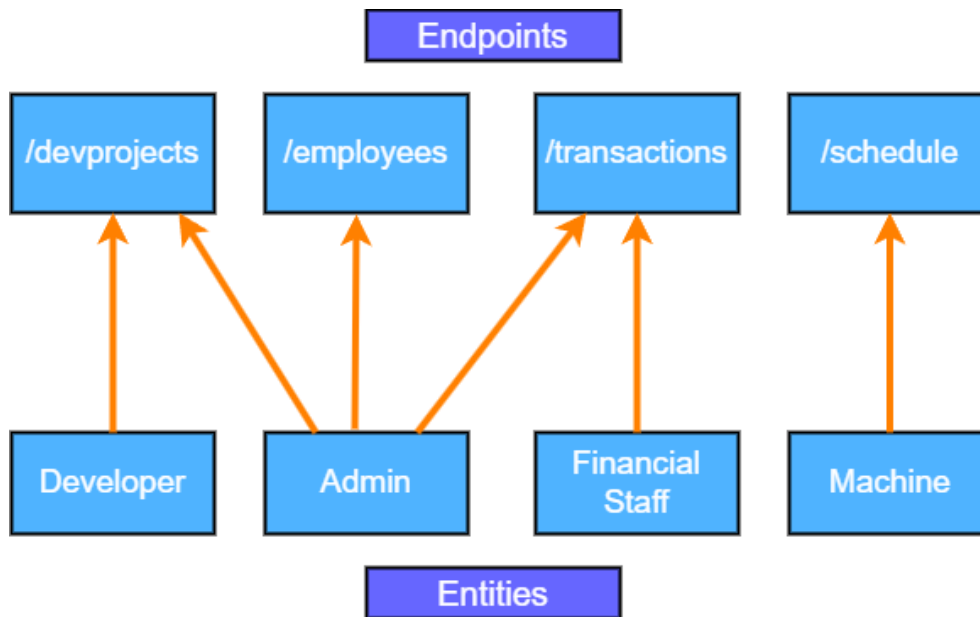


Figure 5.1: Illustration displaying which entities can reach which endpoints in the AWS infrastructure

It is important to note that the PoC alone cannot be considered fully secure until additional security measures are implemented, such as those covered in chapter 4. The purpose of the PoC is solely to showcase the access control capabilities based on these digital identities.

In addition to this chapter, which describes the design choices and implementation, the group has included all relevant files, such as templates, mock data and a step-by-step guide on the implementation in the group's GitHub repository¹. This guide goes through all the necessary steps to set up Microsoft Entra ID and AWS, making duplication easier for those who want to set up the PoC. It shows how to register and expose applications in Microsoft Entra ID. Linking RBAC groups to applications and returning the right information in their access tokens. It also shows how to set up the machine-to-machine application endpoint and its CA rules. On the AWS side, the guide shows how to use CloudFormation to launch the necessary infrastructure to handle the requests coming from Microsoft Entra ID.

5.2 Design Choices

This section will describe choices made by the group on which platforms and client applications are utilised to implement and showcase the PoC. First, the group

¹GitHub repository with a guide on implementation, mock data and templates for setting up the PoC <https://github.com/NBIMBachelor/Incorporating-Digital-Identities-Into-APIs>

chose two cloud platforms, one to host the API and the other to be used as an IDP. Lastly, a client program was chosen to showcase how an entity could contact both platforms.

5.2.1 Choice of Cloud Platforms

The decision to use Microsoft Entra ID as the IDP was primarily driven by stakeholder preferences. Given the group's lack of experience with IAM systems, there were no objections, as any other system would have presented similar challenges. The choice was deemed suitable for the project's needs.

The group's collaboration with NBIM strongly influenced the choice of using AWS to host the API infrastructure for the PoC. NBIM not only allocated a budget specifically for AWS services but also preferred AWS due to their existing use and familiarity with its environment. Additionally, choosing AWS enabled the group to benefit from direct support in troubleshooting by the stakeholders at NBIM, simplifying the resolution of any technical challenges encountered during the project.

5.2.2 API Infrastructure

The group implemented CloudFormation, a module in AWS that allows a template file to automatically deploy and manage resources across the AWS infrastructure. By writing a template file as Infrastructure as code (IaC), the group could create all the necessary resources and launch them simultaneously. Instead of using the UI, writing IaC is one of the core guidelines in the AWS Well-Architected Framework design principles [78]. This is because it enables more automation and easier duplication, reduces configuration errors that are prone in manual configurations and enables consistency[79].

The group selected various modules to launch from CloudFormation, including JWT authorisers to manage authorisation on the endpoints. There was the option to use lambda functions, which allows for the implementation of authorisation as code in Node.js or Python. However, the JWT authoriser had all the necessary functionalities to demonstrate the PoC. Therefore, the option for the group to familiarise itself with lambda authorisers was dismissed. JWT authorisers offer a straightforward method to manage access control based on predefined scopes, audiences and issuers, simplifying the implementation.

This decision required the adoption of API Gateway v2, as it is the version that supports JWT authorisers. However, this version of the API Gateway does not support AWS WAF, which the group wanted to utilise as a first layer of defence against incoming traffic to the infrastructure. The group wanted to implement WAF since it allows for rulesets that grant or deny access to the API Gateway based on predefined rules. These rules help limit bot requests and protect against common vulnerabilities, saving the group from unnecessary costs and better protecting

the AWS infrastructure. Because of this, AWS CloudFront was placed in front of the API Gateway, which forwards requests to WAF for approval before forwarding them to the API Gateway. In this way, firewall measures can be implemented despite the limitations of the initial version.

5.2.3 Postman

In the PoC, Postman illustrates how an entity could manually request an access token. The user could request tokens with other applications similar to Postman or command line interfaces. Applications that require the use of the API can also automate the token request process in code. This eliminates the need to input parameters for each request manually. Still, this PoC aims to showcase what information is needed to authenticate the user and how they can access endpoints in AWS. For this, Postman is chosen because it visually shows all the necessary information required to gain the access token.

5.3 Implementation

To authenticate the digital identity, Microsoft Entra ID will authenticate users and generate an access token following the OAuth 2.0 framework. Both end users and devices must authenticate at endpoints exposed by Microsoft Entra ID. Once a user or machine receives the token, it is included in the header of subsequent requests sent via Postman to an endpoint managed within AWS. The token will be in the JWT format and contain, among other things, scope, issuer and audience information. AWS, as the API provider, will use these properties to determine if the actor can access the requested resources.

5.3.1 Microsoft Entra ID Setup

The initial phase of the implementation involves establishing a tenant in Microsoft Entra ID to represent an organisation. This setup includes creating three distinct role-based groups: Admin, Developers, and Financial Staff. Each group is designed to simulate various end-user roles within the organisation.

In the subsequent phase, three applications are registered within the tenant, each explicitly linked to one of the aforementioned groups. These applications have an exposed endpoint the users of their group can contact to receive access tokens. It is configured to ensure that only designated group members, such as developers, can authenticate themselves against the Microsoft Entra ID developer application endpoint, thus giving each group the privilege to authenticate themselves. Importantly, these endpoints each return a different token that includes scopes specific to their group, such as a "developer.scope". This scope, along with other factors, is used downstream to authorise access to particular endpoints in AWS. If a user who is not a member of the correct group attempts to authenticate against

that group's application, they will receive an error message, ensuring that access privileges align with group membership.

For the machine-to-machine implementation, an additional application is created that filters authentication based on IP addresses and pre-shared secret keys. Unlike the applications made for end users, this application does not receive a scope in the token and is not linked to a specified group. The IP ranges are defined in its CA rules, which only allow IP addresses from NTNU's IP range, but optimally, it would contain the specific IP of the device that should be allowed to request access tokens.

5.3.2 Amazon Web Services Setup

The CloudFormation template file defines all modules used within AWS. The first is CloudFront, which is configured to forward all incoming traffic to AWS WAF. The group has implemented five general rule sets in WAF. Though customised rule sets are also possible to configure, this PoC only uses predefined rule sets from AWS. The first three rule sets are taken from the AWS WAF baseline rule groups as they are intended to be generally applicable to most web applications. The second group is taken from the IP reputation rule group, which uses AWS's threat intelligence to block known bad IP sources. These rule sets are:

AWSManagedRulesCommonRuleSet

- Provides protection against exploitation of a wide range of vulnerabilities, including those described in OWASP [72].

AWSManagedRulesAdminProtectionRuleSet

- Contains rules that allow the application to block external access to exposed admin pages [72].

AWSManagedRulesKnownBadInputsRuleSet

- Contains rules that allow the application to block request patterns known to be invalid and associated with exploitation or discovery of vulnerabilities [72].

AWSManagedRulesAmazonIpReputationList

- Rules that are based on Amazon threat intelligence. Blocks sources associated with bots or other threats known to AWS [80].

AWSManagedRulesAnonymousIpList

- Rules that block requests from services that allow obfuscation of viewer

identity. This can include requests originating from VPNs, proxies, Tor nodes, and hosting providers [80].

The next module defined in CloudFormation is the API Gateway, which is set up with four HTTP integrations and two JWT authorisers. One authoriser is for end users, and one is for machine authorisation. Both JWT authorisers have the same issuer field, which includes Microsoft Entra ID's tenant ID.

Next for end users, the JWT authoriser checks for the audience field with the expected value of one of three applications Client ID registered in Microsoft Entra ID. If these parameters are valid, the JWT authoriser grants access based on the scope included in the JWT token. In figure Figure 5.2 the JWT authoriser for developer shows that only scopes acquired from the admin or developer endpoints in Microsoft Entra ID are accepted. For the machine authoriser the difference is the audience field which is specified to be the machine application's client ID in Microsoft Entra ID. The reason the machine endpoint has its own authoriser is that it can't have a scope and, therefore, needs to be separated from the end user authorisers.

The screenshot displays the configuration for a JWT authoriser for the route GET /devprojects. The interface includes the following sections:

- Authorizer for route GET /devprojects**: Includes buttons for "Detach authorizer" and "Edit authorizer".
- Authorizer details**:
 - Authorizer name: JWTAuth
 - Authorizer type: JWT
 - Authorizer ID: o3 [redacted]
- Identity source**:
 - When this authorizer is invoked, API Gateway will use this selection expression to determine the source of the token: `$request.header.Authorization`
- Issuer**:
 - The issuer URI of the Identity Provider: `https://login.microsoftonline.com/[redacted]/2.0`
- Audience**:
 - The audience associated with this authorizer:
 - e03 [redacted]
 - 079 [redacted]
 - 744 [redacted]
- Authorization scopes**:
 - Specify one or more scopes that are required to access this API route:
 - scope.admin [Remove]
 - scope.dev [Remove]
 - [Add scope]

Figure 5.2: JWT authoriser for the developer group in AWS

The last part of the infrastructure is the API Gateway's HTTP integration. Its only job is to fetch mock data from the group's GitHub repository with the use of a

GET request. This repository is outside the AWS infrastructure and is included to showcase how different end-users or machines could request data based on their access privileges.

5.3.3 Requesting Access Tokens

Figure 5.3 displays a screenshot of Postman, configured to demonstrate how a developer registered in the Microsoft Entra ID tenant can request an access token. The authorisation tab is filled in with the necessary information: the authorisation endpoint URL, token endpoint URL, client ID, client secret, and the specific scope associated with developers. This is all generated to the unique tenant and application made in Microsoft Entra ID.

Configure New Token

Token Name: Proof of Concept - Developer

Grant type: Authorization Code (With PKCE)

Callback URL: https://oauth.pstmn.io/v1/callback

Authorize using browser

Auth URL: {{Authentication URL}}

Access Token URL: {{Access Token URL}}

Client ID: {{Developer ID}}

Client Secret: {{Developer Secret}}

Code Challenge Method: SHA-256

Code Verifier: Automatically generated if left blank

Scope: {{Developer Scope}}

State: State

Client Authentication: Send as Basic Auth header

> Advanced

Clear cookies

Get New Access Token

Figure 5.3: Illustration displaying how to gain access tokens for the developer group in the PoC

Admin, developers and financial staff, who can interact with their clients, use authorisation code grant with PKCE from the OAuth 2.0 framework as outlined as best practice in subsection 4.3.4. For the machine-to-machine implementation, client credential grant is used, as it follows best practice as outlined in subsection 4.3.4 for non-interactive OAuth 2.0 grant types.

Figure 5.4 shows how a machine could request an access token using the client credential grant. For this implementation to work, the request must include which

grant type is used, which client ID the machine requests a token from, and its preshared key. The scope is a value with the property "/.default" appended to the client ID. ".default signals that consent should be prompted for all required permissions listed in the application registration." [81].

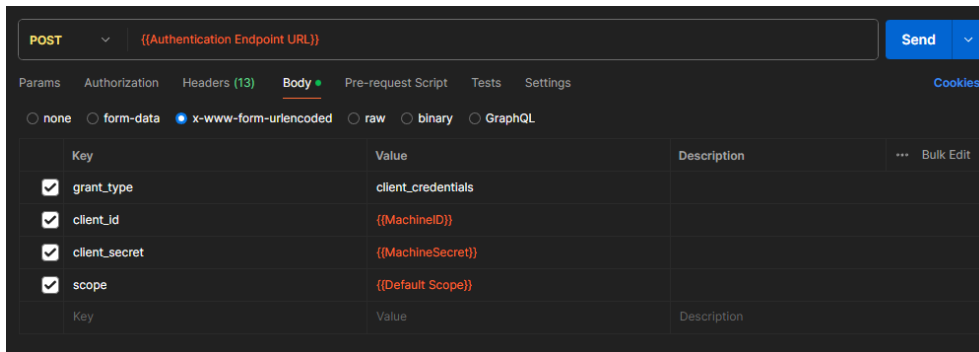


Figure 5.4: Illustration displaying how to gain access tokens for the machine-machine communication in the PoC

5.3.4 Data Flow

This section will show how a developer can authenticate themselves at Microsoft Entra ID and send requests to the API hosted in AWS. It will also describe the differences between end-user authentication and authentication for machines.

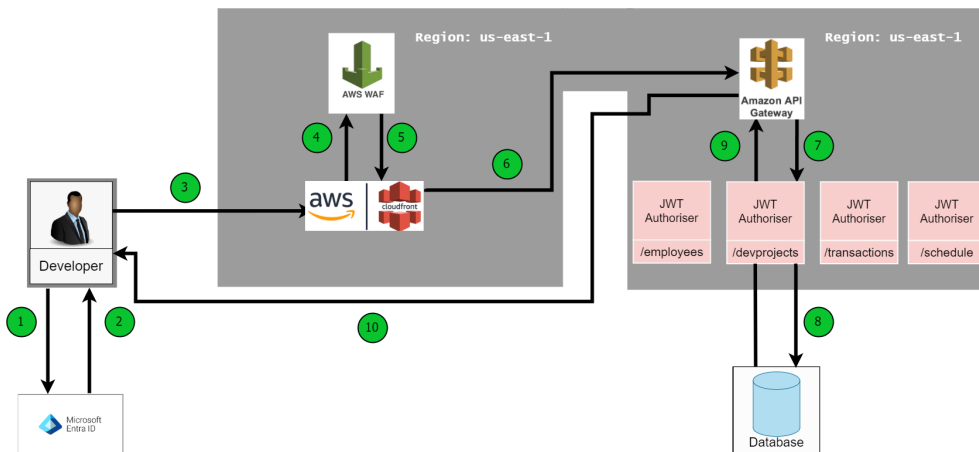


Figure 5.5: Data flow diagram displaying how users from the developer group would access the AWS infrastructure and get access to resources stored in the organisation's database

Request Initialisation

Figure 5.5 shows the flow for a developer trying to access the /devprojects endpoint in AWS. After requesting an access token, which is (step 1), the user will be redirected to Microsoft Entra ID, where they have to input their credentials to authenticate themselves. This authentication is based on the user's group role and checks if they are registered in the current application from which they request an access token. If they are authenticated, an access token will be sent back in (step 2). In Figure 5.3, the callback URL is configured to direct to Postman, enabling visual confirmation of the token.

For the machine-machine communication (step 1) and (step 2) entails sending a POST request to its application endpoint in Microsoft Entra ID as shown in Figure 5.4 to receive an access token. Microsoft Entra ID checks whether the request contains the client secret, and CA checks if the IP address is within the accepted range. The rest of the flow is similar to Figure 5.5 except the endpoint will be /schedule as shown in Table 5.1.

↓ Entity/Endpoint →	/devprojects	/employees	/transactions	/schedule
Admin	✓	✓	✓	✗
Developer	✓	✗	✗	✗
Financial Staff	✗	✗	✓	✗
Machine	✗	✗	✗	✓

Table 5.1: Overview of which entity in the Microsoft Entra ID Tennat can access which endpoints in the AWS infrastructure

First Point of Security and Access Control

As (step 3) shows in Figure 5.5 the developer sends a request to CloudFront. Before it gets sent to the API Gateway the request first gets sent to the WAF (step 4) to check if the request satisfies its rule sets. If the request is approved by WAF, it is then sent from CloudFront (step 5) to one of the endpoints handled by the API Gateway (step 6). Before the API Gateway allows the request to get resources from the endpoint, it will use its corresponding JWT authoriser to validate the JWT token.

For the /devprojects endpoint (step 7) the JWT authoriser validates the issuer, audience and scope in the JWT token. When the issuer and audience fields are verified, the authoriser checks, if the scope included, belongs to one of the accepted groups from Microsoft Entra ID.

Accessing the Resources

Once the authoriser approves the request, the API Gateway will use its HTTP integration, configured to fetch data from a database which, in this implementation,

is the GitHub repository in (step 8). For instance, in this scenario, the developer has requested access to the /devprojects endpoint. After the request to GitHub has returned (step 9), the content will subsequently be relayed back to Postman (step 10). In this implementation, developers and administrators can access the data associated with the /devprojects endpoint. The same would apply to scenarios in which an entity other than a developer attempts to access resources from their respective endpoints. (step 7-9) would have to change to correspond to the correct endpoint the entity is making a request to access. Figure 5.6 shows the mock data for /devprojects being returned after a successful request to AWS.

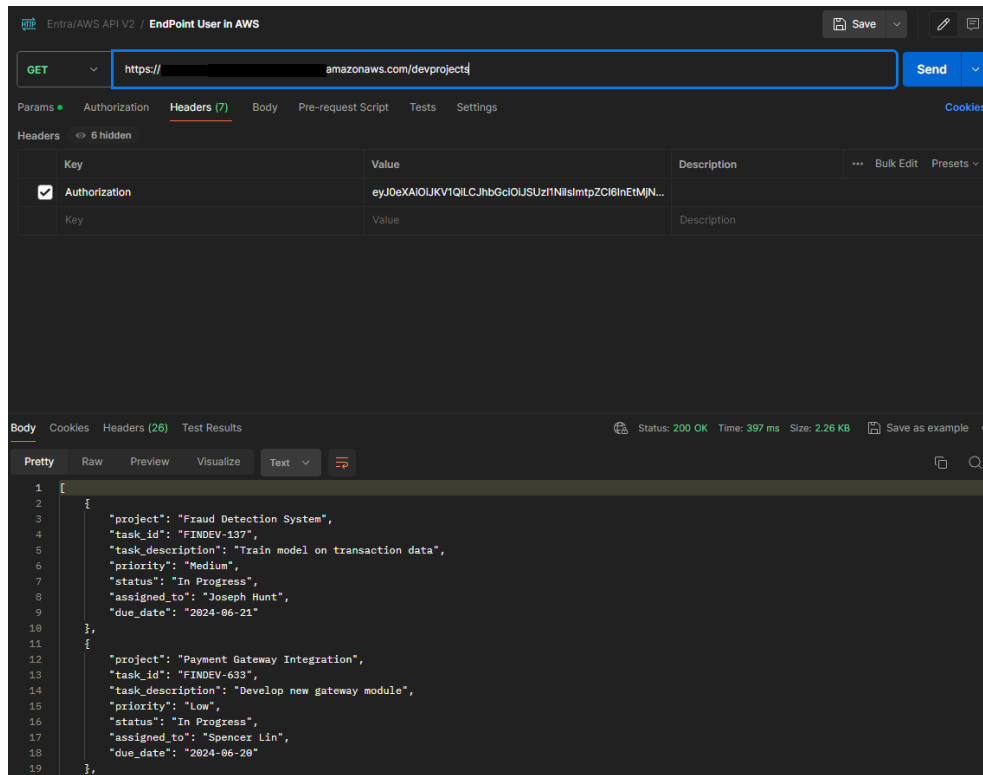


Figure 5.6: Mock data received from the /devprojects endpoint in the PoC

This marks the last step in the data flow for the PoC, where the API has done its job for the developer making the request, and the developer received the data they requested. The PoC has shown how to successfully implement different types of digital identities into the API's authentication and authorisation process while achieving fine-grained access control based on the actor's digital identity.

6 Discussion

6.1 Introduction

This chapter will discuss several aspects of the report. First, it will examine whether the project's goals have been reached. Then, it will explain why the group chose different technologies and protocols and suggest alternatives that could have been used. Afterwards, it will explore how the thesis touches upon different aspects of sustainability and explain how the group has utilised artificial intelligence, finishing with discussions regarding the thesis and group work.

6.2 Results

The following section will discuss whether the group reached the learning, effect, and result goals set at the project's start.

6.2.1 Learning Goals

After understanding the task, the group decided on the learning goals. Most, but not all, of these goals were reached during the group's work.

L1: *Get familiar with scrumban.* At the start of the project, the Kanban board and daily meetings were frequently used. This method was partly abandoned after the first month of work. The group stopped using the Kanban board, as the tasks were quite broad and didn't require keeping detailed track of, and the standup meetings covered what had to be kept track of. The group also decided against having a scrum master. The role felt like it added more work than value to the group's work. In the later parts of the project, the Kanban board was brought back as many small tasks started to pile up, and they needed to be kept track of. Based on this, the goal of getting familiar with the scrumban method has been reached.

L2: *Gain better knowledge of industry practices for authentication and authorisation to APIs using digital identities.* The goal of gaining a greater understanding of the industry practices regarding authentication and authorisation of APIs has been reached. The group has gone from barely knowing the names of the

protocols for API authentication and authorisation to having extensive knowledge of several of the industry standards such as OAuth 2.0, OIDC and SAML and their respective best practices.

L3: Learn how to use cloud computing tools. The group gained an understanding of how to utilise the cloud computing tools provided by AWS and Microsoft Entra ID. The group had some experience with cloud computing tools from other school projects but not with handling authorisation and authentication, and none using AWS and Microsoft Entra ID. Creating the PoC using these tools shows a higher level of understanding than before the project.

L4: Learn how to use GitHub to host source code. The goal of using GitHub to host source code was accomplished. The group created a working CloudFormation template to set up the AWS infrastructure. The group used GitHub to host their CloudFormation template, leading to a quick and efficient way of recreating the AWS environment.

6.2.2 Effect Goals

E1: Receive better knowledge about best practices for authentication and authorisation of APIs using digital identities. After finalising the PoC, the group has gotten a firm understanding of how digital identities can be used. Through the IAM service Microsoft Entra ID, the group uses the attributes of a user to grant an access token and then authorise access to endpoints hosted in AWS.

E2: Give a better overview of how to secure APIs and how digital identities can be implemented into an API. The group considers this goal as achieved. How to secure an API is discussed thoroughly in chapter 4 through different risk mitigation methods, and how to implement digital identities is explained in chapter 5.

6.2.3 Result Goals

R1: Deliver a report that can be used to improve API security. The group believes this goal is reached with the inclusion of the threat model, different methods to mitigate these threats, and the explanation of how to set up a small environment that utilises digital identities.

R2: Deliver a PoC of the group's findings, which will be an API showcasing how to incorporate digital identities into the authentication and authorisation process. This goal was reached. The PoC showcases the authentication of actors based not only on their credentials but also on attributes tied to their identity in the environment. In the case of the PoC, this attribute for end users was group membership; for machines, it was the location.

6.3 Alternatives

In the group's solution, token validation happens at the API Gateway, in the form of AWS's JWT authoriser. This means that every request that passes the WAF will be processed by CloudFront before it reaches the authoriser and is denied, which could lead to unnecessary processing of requests. Another solution is to authorise requests at CloudFront. The group is not validating tokens at this earlier stage because of their use of JWT authorisers. The choice to use JWT authorisers was made because they proved sufficient for the task and were more straightforward to implement than the more advanced lambda authorisers. Cloudfront does not support JWT authorisers [33]. Therefore, another solution is to use lambda authorisers instead of JWT authorisers. These are supported by CloudFront and would enable authorisation as the request first enters the AWS environment.

In the PoC, the group decided to use Microsoft Entra ID as the IDP. The choice was made because of stakeholder preference. If not for this preference, the choice would likely have been Amazon Cognito. This is because the first part of the PoC made was the endpoints, which were in AWS. Therefore, the most convenient solution would have been to stay in the AWS environment for the IDP as well.

The group chose OAuth 2.0 and OIDC as their protocols for authorisation and authentication. Another option would have been to use SAML. The reason for using OAuth 2.0 and OIDC was because it seemed much more manageable to implement. In the end, using SAML for the PoC would reach a similar result, although user experience might change.

6.4 Sustainability

The UN Sustainable Development Goals (SDG) provide a comprehensive framework for addressing pressing global challenges and promoting sustainable development worldwide [82].

The work done in the report aligns with several SDGs, notably Goal 9: Industry, Innovation, and Infrastructure. By focusing on API authorisation and authentication, this thesis contributes to enhancing digital infrastructure. Effective authentication and authorisation mechanisms are crucial for secure and reliable data exchange and sustainable digital infrastructure development. Moreover, by exploring diverse practices and implementing a technical PoC illustrating the utilisation of digital identities in APIs, the thesis fosters innovation in information technology, in line with the sustainability goal of advancing innovation through novel technological solutions. Lastly, by considering the data sensitivity and recommending appropriate security controls based on API threat modelling, the task ensures inclusive and sustainable industrialisation, fulfilling Goal 9 [83]. This ensures APIs adhere to stringent security standards for building a dependable digital infrastructure.

Additionally, by ensuring robust authentication and authorisation mechanisms in APIs, the task contributes to fostering strong institutions as outlined in Goal 16 [84]. One of the policies in goal 16 is the right to privacy. Ensuring that malicious actors cannot access a person's private information would assist in reaching this goal. This initiative builds trust and security in digital societies, potentially reducing cybercrime and fostering a safer online environment for individuals and organisations.

Lastly, by sharing insights and recommendations on security controls and best practices, the task can enhance collaboration among various organisations, companies, and governments to improve the data security and reliability of digital systems on a global scale. This supports Goal 17 of strengthening the implementation of sustainable development goals by promoting collaboration and partnerships across sectors and borders [85].

6.5 Use of Artificial Intelligence

During the bachelor thesis, the group utilised two prominent AI tools to improve the report and PoC. Firstly, ChatGPT 4.0 was used to troubleshoot different parts of the PoC. With limited prior experience in AWS, Microsoft Entra ID, and code formats like YAML, JSON, and XML, the team sought guidance from ChatGPT. It helped the group not only to identify code errors but also to navigate through AWS and Microsoft Entra ID. Additionally, ChatGPT helped improve the writing of some paragraphs and generated the mock data for the API endpoints in AWS for the PoC.

Secondly, the group used Grammarly to refine the sentence structure and grammar in the report. It was useful for correcting spelling errors and offering feedback on sentence structure and text flow. This ensured a high readability standard and professional presentation in the final submission.

6.6 Criticism of the Thesis

The group did not set a definitive scope for the thesis early enough in the work process. This did not become a problem until the later half of the thesis, when limiting what to research became troublesome. As the group learned more about the subject matter, it became apparent that securing API authentication and authorisation had a larger field of research than first believed. As the Theory chapter grew too long, the group realised that cuts had to be made, and a clear scope had to be set. If this had been done earlier, research and writing on subjects that would ultimately be deleted could have been avoided.

The criteria for the DREAD model and the risk matrix did not quite align with each other. The DREAD model was more stringent, as highlighted by the DREAD analysis not achieving the values set in the risk appetite, while the risk matrix did.

This discrepancy was mainly due to the different number of risk levels in the two models, the DREAD model had three levels, while the risk matrix had four levels, making direct comparison challenging. To improve the analysis for the DREAD model, a revised scoring system should be used that values the scores for each category equally.

6.7 Evaluation of Group Work

The group started off using the scrumban method. This includes a Kanban board for organising and delegating tasks, assigning a scrum master, and having four weekly stand-up meetings to discuss these tasks. However, after the first month of work, the group completely abandoned the Kanban board and chose not to have a scrum master, as this role felt more disruptive than advantageous to the group's workflow. However, they opted instead to have stand-up meetings every day, Monday through Friday. This change was partly due to the negligence of the Kanban board and partly because group members had little to no oversight of what the others were working on in between meetings. This led to daily meetings where the group could discuss what they had done and planned to do for the rest of the day.

The threat model became a much larger part of the thesis than originally planned. Although the entire group contributed by offering feedback on the threat model and guiding its development, the actual production tasks remained the responsibility of one individual.

The group did not have a predetermined method for approaching their sources¹. The method that evolved during the research was to find a relevant source for the task at hand and take notes while reading. Looking back, the group should have discussed this before the research phase began and agreed on a method of how to best analyse and organise the gathered information.

The group work has been strenuous at times, and most of the work has been solitary. The group had regular in-person meetings on Fridays, but these meetings were meant for discussing what to present during meetings with supervisors. Of course, it wasn't a rule to always work alone. It was always possible to ask questions when struggling with a task or having a chat over Discord. When the group was setting up the PoC, most of the group was working and talking together over several days.

Some group members felt there was a lack of feedback on their work, wanting the group to follow up on each other's work. However, other members felt that their feedback sometimes led to bad reactions. The product of this was that less and less feedback was given during the semester.

¹A method such as this, presented by Deakin University, could have been used. https://www.deakin.edu.au/__data/assets/pdf_file/0006/2527845/Science-Reading-and-Analysing-Scientific-Research.pdf

7 Conclusion

7.1 Introduction

This chapter presents the group's findings regarding the research questions. It also offers suggestions for possible future research areas that could build on this report's findings. Lastly, the chapter provides a summary highlighting the main contributions and importance of the research.

7.2 Research Questions

R1: What are the main threats to APIs and digital identities during authentication and authorisation, and how can they be mitigated?

This research question has been divided into two sub-questions, discussing the threats and mitigation methods for APIs and digital identities separately. This was done as the threats and mitigations for APIs and digital identities differ. To separate these differences, the question has been divided into two parts.

R1.1: What are the main threats to APIs during authentication and authorisation, and how can they be mitigated?

The main spoofing threat was unauthorised access via stolen credentials, which involves a threat actor successfully spoofing a legitimate user by using their credentials. The mitigation methods are to use well-monitored centralised identity management, RBA to detect unusual user activity, using a well implemented SAML or OAuth 2.0 system, utilising WAFs ability to identify unauthorised access attempts in real-time, incorporating the PoLP in the organisation and by utilising API keys.

The main tampering threats were CloudFront cache manipulation and AWS WAF rule manipulation. Manipulating the CloudFront cache could mislead users, distribute malware, or tarnish the organisation's reputation. By manipulating WAF rules, a threat actor could enable malicious requests or block legitimate traffic. The mitigation method for CloudFront cache manipulation is to block common web exploits using WAF, and the mitigation method for WAF is to incorporate the PoLP in the organisation.

The main information disclosure threats were incorrect configuration of the API

Gateway and unauthorised access to user data via an admin endpoint. The mitigation methods for these include implementing either SAML or OAuth 2.0 with OIDC, which utilises JWT tokens for authentication and authorisation. Furthermore, one should configure the API Gateway properly, incorporate PoLP in your organisation and implement RBAC or ABAC as your organisation's access control method. Lastly, implement JIT for privileged actions, utilise AWS WAF's ability to identify and stop threats in real-time, utilise CA policies and integrate zero trust principles into your organisation.

The main denial of service threats were CloudFront DoS attacks, WAF scripted request flood and API Gateway application-layer DoS attacks. DoS attacks on CloudFront or the API Gateway could lead to widespread service disruption. The mitigation methods for these are blocking common web exploits using WAF and integrating real-time risk detection using CA.

The main elevation of privilege threats was HTTP integration attacks. HTTP integration attacks could lead to a system compromise. The mitigation method for this scenario is to configure the API Gateway properly, utilising AWS WAF's ability to identify threats in real-time, implementing RBAC or ABAC as the organisation's access control method, implementing JIT for privileged actions, using a well implemented SAML system for authorisation and utilising CA policies.

R1.2: What are the main threats digital identities face during API authentication and authorisation, and how can they be mitigated?

The main spoofing threats to digital identities were unauthorised access via stolen credentials, identity spoofing in Microsoft Entra ID and session hijacking. These scenarios were chosen as they could all lead to an identity hijacking. By utilising MFA, a threat actor would need more than stolen credentials to hijack a digital identity. By utilising PKCE, a stolen access token won't be enough to pose as that user. By using session management, threat actors would be unable to intercept data in transit.

The main tampering threat to digital identities was unauthorised modification, which could lead to the altering of user credentials. This would mean that only the threat actor would have access to the identity. To mitigate this scenario, one should prevent unauthorised access from occurring. Both MFA and PKCE can help to achieve this.

The main information disclosure threat to digital identities was MITM attacks. A threat actor intercepting data during transfer could potentially leak sensitive information, such as credentials. Strong session management policies would prevent this data from being intercepted. MFA would ensure that even if credentials were intercepted, a successful login would require another authentication method.

However, securing the API would also ensure the information it handles. Therefore, the mitigations recommended for securing the API regarding spoofing, tampering and information disclosure are recommended for securing digital identities.

ies.

R2: What are the current industry standard protocols and technologies regarding API authentication and authorisation?

The current industry standard protocols for API authentication and authorisation are SAML and OAuth 2.0. SAML can be used for both authentication and authorisation. In contrast, OAuth 2.0 can only be used for authorisation and has to be used in conjunction with OIDC to offer authentication capabilities. SAML is mainly used by larger organisations to enable enterprise SSO capabilities. OAuth 2.0 and OIDC are often used by organisations of all sizes to authorise and authenticate users using an IDP they are familiar with, such as Facebook or Google. It is OAuth 2.0 best practice also to use the PKCE extension. While these protocols handle the actual authentication and authorisation of users, other protocols are involved in the process. JWTs are becoming the industry standard token format for authentication, owing to their self-contained, stateless design.

The industry standard methods used for access control are RBAC and ABAC. RBAC can govern access on a per-group level, whereas ABAC can do it per person by using attributes from both the user initiating a request and the API they want to access. Though RBAC is less fine-grained than ABAC, it is suitable for small to medium organisations, whereas ABAC is recommended for larger organisations.

The PoLP is used to limit what an organisation's users have access to. The idea is to grant the minimum level of privileges necessary to perform their required work and nothing more. Zero trust is the idea of never trusting, always verifying users inside or outside the environment, and always verifying their identity regardless. Combining these policies would reduce the damage of a potential breach.

7.3 Further Work

This section will introduce the group's ideas on how the thesis and PoC could be improved upon further. The reason for these improvements not being made in the first place could be a lack of priority for a certain task, that they fall outside of the group's scope or the size of a task.

Further work would include implementing more of the available security measures provided by Microsoft Entra ID and AWS in the PoC, as well as security measures presented in chapter 4. Due to the group's goal of implementing authorisation and authentication using digital identities, security aspects such as RBA, JIT and a more advanced CA setup were not prioritised.

Testing with a large number of requests would show how robust the PoC is and how cost-effective it is. Even if the PoC is secure and robust, if the cost per request is too high, it would not be a viable solution for most organisations. A comparison between the PoC in this report, which uses Microsoft Entra ID, and one using Amazon Cognito, and comparing the cost when handling a large number of re-

quests would tell which is cheapest. If definitive measures of security and cost could be done between the two, a report on this would be valuable to organisations.

The group did not manage to implement IaC for Microsoft Entra ID in the same way they did for AWS. If an IaC template for the Microsoft Entra ID system were in place, it would remove the time and effort needed to set up the Microsoft Entra ID environment required for the PoC, as well as remove the possibility of human error when doing it manually. The making of the Microsoft Entra ID IaC template should be done using a language such as Terraform.

The group wanted to add scopes for the machine-to-machine access tokens. Adding scopes would allow a JWT authoriser to allow or deny requests on the same level as human users, leading to better access control.

A more modern solution to managing digital identities is using decentralised identity. As it stands now, the PoC is using centralised identity management. This means any human user that wants to access the endpoints hosted in AWS has to have an identity in Microsoft Entra ID. This would mean full disclosure of their identity to the administrators of the Microsoft Entra ID environment. A logical next step in the PoC's development would be implementing decentralised identity management. Implementing decentralised identities means that user identities would be stored on user devices. The only information shared with Microsoft Entra ID would be selected by the owner of the information. This method lets individuals control their identities [86].

Another way to move away from centralised identity management could be implementing federated identity management. Through federated identity, remote users would not have to register their identity at the group's IDP. The group's IDP, Microsoft Entra ID, could enter a federation with other domains. Through federation, a user on the trusted domain can exchange access tokens from their IDP for access tokens from the Microsoft Entra ID IDP. Those tokens could then be used to access protected resources, which would be the endpoints hosted in AWS [87]. This would remove the threats involved in transmitting sensitive information across insecure channels. Users would authenticate in their own domain, and only encrypted and signed JWT tokens protected further with PKCE would be transmitted between domains. This also helps with the solution's usability, as users from other organisations don't have to be added to the organisations hosting the APIs infrastructure. Rather, they can employ federation and accept users from other organisations without adding them to their user directory.

The group acknowledges that there are threat mitigation methods for APIs not mentioned in the report. However, through the threat model presented in chapter 3 and the risk mitigations explained in chapter 4, the group has not managed to satisfy the organisation's risk appetite because the residual risk levels are too high. Therefore, further mitigation methods are required.

7.4 Final Thoughts

The group is happy to announce that they have completed their thesis project, adhering to the stakeholder's specifications and remaining within the project's defined scope. Stakeholder feedback was instrumental in shaping the group's goals and requirements, leading to a successful result that satisfied all parties involved.

The group has completed their task, providing a report outlining how to safeguard API authentication and authorisation with a focus on utilising digital identities and a PoC showcasing how digital identities can be used for access control. These have both been presented to and approved by the stakeholders, whom the group would like to thank for their continued support and feedback.

A secure API and safe digital identities mostly go unnoticed. It is not until an API goes down or identity is hijacked that organisations notice how much they rely on them. By following the recommendations of this thesis, the group hopes to help IT personnel keep the CEOs of the world blissfully unaware of the complex security implementations that keep the world turning.

Bibliography

- [1] NBIM. 'Dette er oljefondet.' (27th Feb. 2024), [Online]. Available: <https://www.nbim.no/no/oljefondet/slik-er-fondet-investert/aksjeforvaltningen/> (visited on 19/01/2024).
- [2] J. Snyder, R. Priddle and I. Foster, 'The state of apis and api security 2023,' Apr. 2023. [Online]. Available: <https://resources.firetail.io/hubfs/23167483/API%20Security%20Report%202023%20from%20FireTail.pdf>.
- [3] OWASP. 'Owasp api security project.' (n.d.), [Online]. Available: <https://owasp.org/www-project-api-security/> (visited on 18/01/2024).
- [4] O. A. S. Project. 'Api2:2023 broken authentication.' (2023), [Online]. Available: <https://owasp.org/API-Security/editions/2023/en/0xa2-broken-authentication/> (visited on 19/01/2024).
- [5] K. Townsend. 'Bad bots account for 73 percent of internet traffic: Analysis.' (16th Nov. 2023), [Online]. Available: <https://www.securityweek.com/bad-bots-account-for-73-of-internet-traffic-analysis/> (visited on 22/01/2024).
- [6] P. Siriwardena, *Advanced API security*. Apress, 2020.
- [7] D. Temoshok, D. Proud-Madruga, Y.-Y. Choong, R. Galluzzo, S. Gupta, C. LaSalle, N. Lefkovitz and A. Regenscheid, 'NIST special publication 800-63-4,' 8th Dec. 2023. [Online]. Available: <https://pages.nist.gov/800-63-4/sp800-63.html> (visited on 12/03/2024).
- [8] A. W. Services. 'What is an API? - application programming interface explained - AWS,' Amazon Web Services, Inc. (n.d.), [Online]. Available: <https://aws.amazon.com/what-is/api/> (visited on 01/03/2024).
- [9] R. Mogull, J. Arlen, F. Gilbert, A. Lane, D. Mortman, G. Peterson and M. Rothman, *Security guidance: For critical Areas of Focus In Cloud Computing v4.0*. Cloud Security Alliance, 2021.
- [10] D. Temoshok, D. Proud-Madruga, Y.-Y. Choong, R. Galluzzo, S. Gupta, C. LaSalle, N. Lefkovitz and A. Regenscheid, 'NIST special publication 800-63c,' 8th Dec. 2023. [Online]. Available: <https://pages.nist.gov/800-63-4/sp800-63c.html> (visited on 12/03/2024).

- [11] AWS. 'Identity and access management for AWS well-architected tool - AWS well-architected tool.' (n.d.), [Online]. Available: <https://docs.aws.amazon.com/wellarchitected/latest/userguide/security-iam.html> (visited on 18/03/2024).
- [12] N. Madden, *API Security in Action*. Manning Publications Co., 2020.
- [13] OKTA. 'What is openid connect (oidc)?' (n.d.), [Online]. Available: <https://auth0.com/intro-to-iam/what-is-openid-connect-oidc> (visited on 22/01/2024).
- [14] D. Hardt. 'The oauth 2.0 authorization framework.' (21st Jan. 2020), [Online]. Available: <https://datatracker.ietf.org/doc/rfc6749/> (visited on 23/01/2024).
- [15] auth0. 'Introduction to json web tokens.' (n.d.), [Online]. Available: <https://jwt.io/introduction> (visited on 23/01/2024).
- [16] M. B. Jones, J. Bradley and N. Sakimura, 'JSON web token (JWT),' Internet Engineering Task Force, Request for Comments RFC 7519, May 2015. [Online]. Available: <https://datatracker.ietf.org/doc/rfc7519> (visited on 06/02/2024).
- [17] Velotix. 'The 4 most common types of access control.' (11th Dec. 2023), [Online]. Available: <https://www.velotix.ai/resources/blog/the-4-most-common-types-of-access-control/> (visited on 06/03/2024).
- [18] OKTA. 'Role-based access control.' (n.d.), [Online]. Available: <https://auth0.com/docs/manage-users/access-control/rbac> (visited on 25/02/2024).
- [19] S. Das, S. Sural, B. Mitra, V. Atluri and J. Vadiya, *Policy Engineering in RBAC and ABAC*. Springer, 2018.
- [20] K. Casey. 'What is attribute-based access control (abac)?' (29th Sep. 2020), [Online]. Available: <https://www.okta.com/blog/2020/09/attribute-based-access-control-abac/> (visited on 25/02/2024).
- [21] A. W. Services. 'Temporary elevated access.' (23rd May 2023), [Online]. Available: <https://docs.aws.amazon.com/singlesignon/latest/userguide/temporary-elevated-access.html> (visited on 02/04/2024).
- [22] A. Parecki. 'What is the oauth 2.0 authorization code grant type?' (10th Apr. 2018), [Online]. Available: <https://developer.okta.com/blog/2018/04/10/oauth-authorization-code-grant-type> (visited on 22/02/2024).
- [23] O. team. 'Oauth 2.0 implicit grant.' (n.d.), [Online]. Available: <https://oauth.net/2/grant-types/implicit/> (visited on 22/02/2024).
- [24] Y. Wilson and A. Hingnikar, *Solving Identity Management in Modern Applications*. Springer Science + Business Media, 2019.
- [25] O. team. 'Authorization request.' (n.d.), [Online]. Available: <https://www.oauth.com/oauth2-servers/pkce/authorization-request/> (visited on 23/02/2024).

- [26] O. team. 'Client credentials.' (n.d.), [Online]. Available: <https://www.oauth.com/oauth2-servers/access-tokens/client-credentials/>.
- [27] Okta. 'Oauth 2.0 and openid connect overview.' (n.d.), [Online]. Available: <https://developer.okta.com/docs/concepts/oauth-openid/> (visited on 04/06/2024).
- [28] ISO, 'Iso/iec 27005:2022,' 2022. (visited on 30/04/2024).
- [29] NTNU. 'Klassifisering av informasjonsverdier - retningslinje.' (n.d.), [Online]. Available: <https://i.ntnu.no/wiki/-/wiki/English/Policy+for+Classification+of+Information+Assets> (visited on 30/04/2024).
- [30] AWS. 'What is amazon cloudfront?' (n.d.), [Online]. Available: <https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/Introduction.html> (visited on 04/05/2024).
- [31] AWS. 'Aws waf protect your web app.' (n.d.), [Online]. Available: <https://aws.amazon.com/waf/?ref=wellarchitected> (visited on 04/04/2024).
- [32] A. W. Services. 'Amazon api gateway,' Amazon Web Services, Inc. (n.d.), [Online]. Available: <https://aws.amazon.com/api-gateway/> (visited on 22/01/2024).
- [33] 'Controlling access to http apis with jwt authorizers - amazon api gateway.' (n.d.), [Online]. Available: <https://docs.aws.amazon.com/apigateway/latest/developerguide/http-api-jwt-authorizer.html> (visited on 07/05/2024).
- [34] A. W. Services. 'Working with aws service integrations for http apis.' (n.d.), [Online]. Available: <https://docs.aws.amazon.com/apigateway/latest/developerguide/http-api-develop-integrations-aws-services.html> (visited on 20/05/2024).
- [35] OWASP. 'Session hijacking attack.' (n.d.), [Online]. Available: https://owasp.org/www-community/attacks/Session_hijacking_attack (visited on 01/04/2024).
- [36] E. Yalon, I. Shkedy and P. Silva. 'Api1:2023 broken object level authorization.' (2023), [Online]. Available: <https://owasp.org/API-Security/editions/2023/en/0xa1-broken-object-level-authorization/> (visited on 18/03/2024).
- [37] E. Yalon, I. Shkedy and P. Silva. 'Api3:2023 broken object property level authorization.' (2023), [Online]. Available: <https://owasp.org/API-Security/editions/2023/en/0xa3-broken-object-property-level-authorization/> (visited on 24/03/2024).
- [38] E. Yalon, I. Shkedy and P. Silva. 'Api5:2023 broken function level authorization.' (2023), [Online]. Available: <https://owasp.org/API-Security/editions/2023/en/0xa5-broken-function-level-authorization/> (visited on 24/03/2024).

- [39] IEC, 'Iec 62443-3-2 security for industrial automation and control systems,' Jun. 2020.
- [40] P Mano, *Official Guide The CSSLP CBK*. Auerbach, 2013.
- [41] optimal. 'Why identity access management (IAM) is so important?' Optimal IdM. (7th Oct. 2017), [Online]. Available: <https://optimalidm.com/resources/blog/importance-of-iam/> (visited on 02/04/2024).
- [42] OWASP. 'Authentication - OWASP cheat sheet series.' (n.d.), [Online]. Available: https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html (visited on 18/03/2024).
- [43] barclayn, msmimart, alexbuckgit, shlipsey3, BryanLa, lorieide and Microsoft-GuyJFlo. 'What is microsoft entra ID? - microsoft entra.' (29th Mar. 2024), [Online]. Available: <https://learn.microsoft.com/en-us/entra/fundamentals/whatis> (visited on 02/04/2024).
- [44] A. W. Services. 'Security best practices in amazon api gateway.' (10th Oct. 2023), [Online]. Available: <https://docs.aws.amazon.com/apigateway/latest/developerguide/security-best-practices.html> (visited on 04/04/2024).
- [45] Okta. 'Risk-based authentication: What you need to consider.' (15th Sep. 2023), [Online]. Available: <https://www.okta.com/identity-101/risk-based-authentication/> (visited on 28/03/2024).
- [46] Okta. 'Authorization code flow with oidc.' (n.d.), [Online]. Available: <https://auth0.com/docs/authenticate/login/oidc-conformant-authentication/oidc-adoption-auth-code-flow> (visited on 25/03/2024).
- [47] Okta. 'Authorization code flow with proof key for code exchange (pkce).' (n.d.), [Online]. Available: <https://auth0.com/docs/get-started/authentication-and-authorization-flow/authorization-code-flow-with-pkce> (visited on 25/03/2024).
- [48] Okta. 'Client credentials flow with oidc.' (n.d.), [Online]. Available: <https://auth0.com/docs/authenticate/login/oidc-conformant-authentication/oidc-adoption-client-credentials-flow> (visited on 04/05/2024).
- [49] Y. Sheffer, D. Hardt and M. B. Jones. 'Json web token best current practices.' (Feb. 2020), [Online]. Available: <https://www.rfc-editor.org/rfc/rfc8725.pdf> (visited on 04/04/2024).
- [50] OWASP. 'Saml security cheat sheet.' (n.d.), [Online]. Available: https://cheatsheetseries.owasp.org/cheatsheets/SAML_Security_Cheat_Sheet.html (visited on 04/05/2024).
- [51] NSA and CISA. 'Use secure cloud identity and access management practices.' (Mar. 2024), [Online]. Available: <https://media.defense.gov/2024/Mar/07/2003407866/-1/-1/0/CSI-CloudTop10-Identity-Access-Management.PDF> (visited on 01/04/2024).

- [52] Microsoft. 'Enhance security with the principle of least privilege.' (10th Oct. 2023), [Online]. Available: <https://learn.microsoft.com/en-us/entra/identity-platform/secure-least-privileged-access> (visited on 02/04/2024).
- [53] Microsoft. 'Enhance security with the principle of least privilege.' (23rd Oct. 2023), [Online]. Available: <https://learn.microsoft.com/en-us/entra/identity-platform/secure-least-privileged-access> (visited on 07/05/2024).
- [54] R. Lyon, J. Barnett, J. Flores and S. Curzi. 'Best practices for azure rbac.' (30th Jan. 2024), [Online]. Available: <https://learn.microsoft.com/en-us/azure/role-based-access-control/best-practices> (visited on 13/03/2024).
- [55] Onelogin. 'Rbac vs abac: Make the right call.' (n.d.), [Online]. Available: <https://www.onelogin.com/learn/rbac-vs-abac> (visited on 07/05/2024).
- [56] V. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller and K. Scarfone, 'Guide to attribute based access control (ABAC) definition and considerations,' National Institute of Standards and Technology, NIST SP 800-162, Jan. 2014. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-162.pdf> (visited on 12/03/2024).
- [57] R. Chandramouli, 'Security strategies for microservices-based application systems,' National Institute of Standards and Technology, Gaithersburg, MD, Aug. 2019. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-204.pdf> (visited on 12/03/2024).
- [58] Google. 'Why and when to use api keys.' (26th Mar. 2024), [Online]. Available: <https://cloud.google.com/endpoints/docs/openapi/when-why-api-key> (visited on 27/03/2024).
- [59] A. Bilbie. 'Which oauth 2.0 grant should i implement?' (n.d.), [Online]. Available: <https://oauth2.thephpleague.com/authorization-server/which-grant/> (visited on 06/04/2024).
- [60] B. Pontarelli, A. Hashesh and D. Moore. 'Modern guide - what is oauth 2.0 and how does it work?' (n.d.), [Online]. Available: <https://fusionauth.io/articles/oauth/modern-guide-to-oauth#oauth-grants> (visited on 08/03/2024).
- [61] W. Denniss and J. Bradley. 'Oauth 2.0 for native apps.' (Oct. 2017), [Online]. Available: <https://datatracker.ietf.org/doc/html/rfc8252> (visited on 13/03/2024).
- [62] A. Parecki, D. Waite and P. D. Ryck. 'Oauth 2.0 for browser-based apps.' (28th Feb. 2024), [Online]. Available: <https://datatracker.ietf.org/doc/html/draft-ietf-oauth-browser-based-apps> (visited on 10/03/2024).

- [63] W. Denniss, J. Bradley, M. Jones and H. Tschofenig. 'Oauth 2.0 device authorization grant.' (Aug. 2019), [Online]. Available: <https://datatracker.ietf.org/doc/html/rfc8628> (visited on 10/03/2024).
- [64] Okta. 'Access token lifetime.' (16th Aug. 2016), [Online]. Available: <https://www.okta.com/oauth2-servers/access-tokens/access-token-lifetime/> (visited on 02/04/2024).
- [65] NSA, 'Embracing a zero trust security model,' Feb. 2021. [Online]. Available: https://media.defense.gov/2021/Feb/25/2002588479-1/-/CSI_EMBRACING_ZT_SECURITY_MODEL_U00115131-21.pdf (visited on 04/04/2024).
- [66] J. Flores. 'Conditional access: Network assignment.' (7th May 2024), [Online]. Available: <https://learn.microsoft.com/en-us/entra/identity/conditional-access/concept-assignment-network> (visited on 16/05/2024).
- [67] Microsoft. 'What is conditional access?' (19th Mar. 2024), [Online]. Available: <https://learn.microsoft.com/en-us/entra/identity/conditional-access/overview> (visited on 21/03/2024).
- [68] Gargi-Sinha, shlipse3, alexbuckgit, MicrosoftGuyJFlo, markwahl-msft and BryanLa. 'Best practices to secure with microsoft entra ID - microsoft entra.' (23rd Oct. 2023), [Online]. Available: <https://learn.microsoft.com/en-us/entra/architecture/secure-best-practices#logging-and-monitoring> (visited on 02/04/2024).
- [69] A. W. Services. 'SEC04-BP01 configure service and application logging - AWS well-architected framework (2023-04-10).' (10th Apr. 2023), [Online]. Available: https://docs.aws.amazon.com/wellarchitected/2023-04-10/framework/sec_detect_investigate_events_app_service_logging.html (visited on 02/04/2024).
- [70] OWASP. 'Input validation - OWASP cheat sheet series.' (n.d.), [Online]. Available: https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html (visited on 04/04/2024).
- [71] A. W. Services. 'Use-case specific rule groups - aws waf, aws firewall manager, and aws shield advanced.' (n.d.), [Online]. Available: <https://docs.aws.amazon.com/waf/latest/developerguide/aws-managed-rule-groups-use-case.html> (visited on 14/05/2024).
- [72] A. W. Services. 'Baseline rule groups - aws waf, aws firewall manager, and aws shield advanced.' (n.d.), [Online]. Available: <https://docs.aws.amazon.com/waf/latest/developerguide/aws-managed-rule-groups-baseline.html> (visited on 14/05/2024).
- [73] AWS. 'REL05-BP02 throttle requests - AWS well-architected framework.' (3rd Oct. 2023), [Online]. Available: https://docs.aws.amazon.com/wellarchitected/latest/framework/rel_mitigate_interaction_failure_throttle_requests.html (visited on 04/04/2024).

- [74] OWASP. 'Session management cheat sheet'. (n.d.), [Online]. Available: https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html (visited on 01/04/2024).
- [75] M. Maynes. 'One simple action you can take to prevent 99.9 percent of attacks on your accounts,' Microsoft Security Blog. (20th Aug. 2019), [Online]. Available: <https://www.microsoft.com/en-us/security/blog/2019/08/20/one-simple-action-you-can-take-to-prevent-99-9-percent-of-account-attacks/> (visited on 25/04/2024).
- [76] B. Campbell, C. Mortimore and M. B. Jones. 'Security assertion markup language (saml) 2.0 profile for oauth 2.0 client authentication and authorization grants.' (May 2015), [Online]. Available: <https://datatracker.ietf.org/doc/html/rfc7522> (visited on 17/03/2023).
- [77] S. Choudry. 'Why you wouldn't use saml in a spa and mobile app.' (1st Feb. 2021), [Online]. Available: <https://www.identityserver.com/articles/why-you-wouldn-t-use-saml-in-a-spa-and-mobile-app> (visited on 20/05/2024).
- [78] A. W. Services. 'Design principles.' (31st Mar. 2022), [Online]. Available: https://docs.aws.amazon.com/en_us/wellarchitected/2022-03-31/framework/oe-design-principles.html (visited on 07/05/2024).
- [79] A. W. Services. 'What is infrastructure as code?' (n.d.), [Online]. Available: <https://aws.amazon.com/what-is/iac/> (visited on 22/04/2024).
- [80] A. W. Services. 'Ip reputation rule groups.' (n.d.), [Online]. Available: <https://docs.aws.amazon.com/waf/latest/developerguide/aws-managed-rule-groups-ip-rep.html> (visited on 14/05/2024).
- [81] Microsoft. 'Scopes and permissions in the microsoft identity platform.' (17th Nov. 2023), [Online]. Available: <https://learn.microsoft.com/en-us/entra/identity-platform/scopes-oidc#the-default-scope> (visited on 07/05/2024).
- [82] UN. 'Sustainable development goals.' (n.d.), [Online]. Available: <https://www.un.org/sustainabledevelopment/sustainable-development-goals/> (visited on 04/11/2024).
- [83] UN. 'Sustainable development goals.' (n.d.), [Online]. Available: <https://www.un.org/sustainabledevelopment/infrastructure-industrialization/> (visited on 04/11/2024).
- [84] UN. 'Sustainable development goals.' (n.d.), [Online]. Available: <https://www.un.org/sustainabledevelopment/peace-justice/> (visited on 04/11/2024).
- [85] UN. 'Sustainable development goals.' (n.d.), [Online]. Available: <https://www.un.org/sustainabledevelopment/globalpartnerships/> (visited on 04/11/2024).

- [86] A. Johnson-Ubah. 'A beginners guide to decentralized identifiers (dids).' (27th Jul. 2022), [Online]. Available: <https://medium.com/veramo/a-beginners-guide-to-decentralized-identifiers-dids-5e842398e82c> (visited on 30/04/2024).
- [87] Microsoft. 'Overview of federated identity credentials in microsoft entra id.' (n.d.), [Online]. Available: <https://learn.microsoft.com/en-us/graph/api/resources/federatedidentitycredentials-overview?view=graph-rest-1.0> (visited on 26/04/2024).

A Project Plan

Table of Contents

Figures	ii
Tables	ii
List of Abbreviations and Acronyms	ii
Glossary	ii
Bibliography	iv
A Project Plan	v
A.1 Goals and Framework	v
A.1.1 Background	v
A.1.2 Project Goals	v
A.1.3 Framework	vi
A.2 Scope	vi
A.2.1 Problem areas	vi
A.2.2 Limitations	vi
A.2.3 Task Description	vi
A.3 Project Organization	vii
A.3.1 Responsibilities and Roles	vii
A.3.2 Procedures and Group Rules	vii
A.3.2.1 Routines	vii
A.3.2.2 Rules	viii
A.4 Planning, Monitoring, and Reporting	viii
A.4.1 Main Division of the Project	viii
A.4.1.1 Project Management Methodology	viii
A.4.1.2 Scrumban	ix
A.4.2 Plan for status meetings and decision points during the period	ix
A.5 Organization of Quality Assurance	ix
A.5.1 Documentation	ix
A.5.2 Standards	x
A.5.3 Tools	x
A.5.4 Plan for Inspections and Testing	xi
A.5.5 Risk Analysis at Project Level	xi
A.6 Implementation Plan - Gantt	xvi
B Thesis Descripton	xvii
C Standard agreement	xx

Figures

A.1 Gantt Chart	xvi
---------------------------	-----

Tables

A.1 Risk Matrix	xi
A.2 Risk matrix: Scenario 1	xii
A.3 Risk matrix: Scenario 2	xii
A.4 Risk matrix: Scenario 3	xii
A.5 Risk matrix: Scenario 4	xiii
A.6 Risk matrix: Scenario 5	xiii
A.7 Risk matrix: Scenario 6	xiii
A.8 Risk matrix: Scenario 7	xiv
A.9 Risk matrix: Scenario 8	xiv
A.10 Risk matrix: Scenario 9	xiv
A.11 Risk matrix: Scenario 10	xv
A.12 Risk matrix: Scenario 11	xv
A.13 Risk matrix: Scenario 12	xv

Code Listings

List of Abbreviations and Acronyms

API Application programming interface. v, vi, x, xi
AWS Amazon Web Services. v, vi, x, xi
ECS Elastic Container Service. xi
FIDO Fast Identity Online. x
IoT Internet of Things. vi
JSON JavaScript Object Notation. x
JWT JSON Web Token. x
NBIM Norges Bank Investment Management. v, vii, viii, xii–xv
NTNU Norwegian University of Science and Technology. vii
OAuth 2.0 Open Authorization 2.0. x, xi
OIDC Open ID Connect. x, xi
SaaS Software as a Service. vi
SAML Security Assertion Markup Language. x
SSO Single Sign On. xi

Glossary

- Cryptographic** Cryptographic refers to anything related to cryptography, which is the science and practice of secure communication techniques. Cryptography involves the use of mathematical algorithms and techniques to encrypt information, making it unreadable to unauthorized parties.. x
- Digital Identities** The unique online information that identifies individuals, organizations, or devices, including usernames, profiles, and credentials. Used for internet access, transactions, and security.. v, vi
- Fargate** Fargate is a technology offered by Amazon Web Services that simplifies the deployment of containerized applications. It allows developers to run containers without the need to manage the underlying infrastructure, such as servers or clusters.. xi
- GitHub** GitHub is a web-based platform and service that provides a central place for software developers to collaborate on, manage, and version control their code repositories.. v
- idp** An Identity Provider (IDP) is a service that manages and authenticates user identities within a system or application. Its primary function is to verify the identity of users who are trying to access a particular resource or service.. x, xi
- RESTful** RESTful refers to an architectural style for designing networked applications, especially web services. It stands for Representational State Transfer. RESTful design principles promote simplicity, scalability, and reliability in building web services and APIs.. xi
- Scrumban** Scrumban is a hybrid approach that combines elements of both Scrum and Kanban methodologies. It's primarily used in software development and project management to improve workflow and team productivity.. vi, viii, ix, xiii
- sp** A Service Provider (SP) is an entity that provides a service, resource, or application to users.. x
- WebSocket** WebSocket is a communication protocol that provides full-duplex, bidirectional communication. It allows for real-time, interactive communication between a client and a server.. xi

Bibliography

- [1] Norges Bank Investment Management. *Dette er Oljefondet*. URL: <https://www.nbim.no/no/oljefondet/om-oljefondet/> (visited on 19th Jan. 2024).
- [2] Jeremy Snyder, Riley Priddle and Ian Foster. *The State of APIs and API Security 2023*. 2023.
- [3] OWASP. *OWASP API Security Project*. URL: <https://owasp.org/www-project-api-security/> (visited on 18th Jan. 2024).
- [4] OWASP API Security Project. *API2:2023 Broken Authentication*. URL: <https://owasp.org/API-Security/editions/2023/en/0xa2-broken-authentication/> (visited on 19th Jan. 2024).
- [5] Kevin Townsend. *Bad Bots Account for 73 percent of Internet Traffic: Analysis*. URL: <https://www.securityweek.com/bad-bots-account-for-73-of-internet-traffic-analysis/> (visited on 22nd Jan. 2024).
- [6] Productplan. *Scrumban*. URL: <https://www.productplan.com/glossary/scrumban/> (visited on 11th Jan. 2024).
- [7] Amazon Web Services. *What is Scrum?* URL: <https://aws.amazon.com/what-is/scrum/> (visited on 11th Jan. 2024).
- [8] Dick Hardt. *The OAuth 2.0 Authorization Framework*. URL: <https://datatracker.ietf.org/doc/rfc6749/> (visited on 23rd Jan. 2024).
- [9] OKTA. *What is OpenID Connect (OIDC)?* URL: <https://auth0.com/intro-to-iam/what-is-openid-connect-oidc> (visited on 22nd Jan. 2024).
- [10] OpenID Foundation. *What is OpenID Connect*. URL: <https://openid.net/developers/how-connect-works/> (visited on 22nd Jan. 2024).
- [11] auth0. *Introduction to JSON Web Tokens*. URL: <https://jwt.io/introduction> (visited on 23rd Jan. 2024).
- [12] FIDO Alliance. *How FIDO Works*. URL: <https://fidoalliance.org/how-fido-works/> (visited on 12th Jan. 2024).
- [13] FIDO Alliance. *Passkeys*. URL: <https://fidoalliance.org/passkeys/> (visited on 12th Jan. 2024).
- [14] Joel Witts. *The Top 11 FIDO Authentication Solutions*. URL: <https://expertinsights.com/insights/top-11-fido-authentication-solutions/> (visited on 22nd Jan. 2024).
- [15] Amazon Web Services. *Amazon API Gateway*. Amazon Web Services, Inc. URL: <https://aws.amazon.com/api-gateway/> (visited on 22nd Jan. 2024).
- [16] Amazon Web Services. *What is AWS Lambda? - AWS Lambda*. URL: <https://docs.aws.amazon.com/lambda/latest/dg/welcome.html> (visited on 22nd Jan. 2024).
- [17] Amazon Web Services. *Amazon ECS on AWS Fargate*. URL: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/AWS_Fargate.html (visited on 22nd Jan. 2024).
- [18] Celina Heimdal Brynildsen et al. *Securing the Software Development Life Cycle*. 2022.

A Project Plan

A.1 Goals and Framework

A.1.1 Background

The group's assignment has been given by Norges Bank Investment Management (NBIM). NBIM is responsible for assuring long term administration of the profits from Norway's oil and gas resources. The official name of the fund is Statens pensjonsfond utland. The fund has become one of the world's largest, with partial ownership in almost 1.5 percent of all listed companies globally [1].

In the digital age, Application programming interface (API)s have emerged as the backbone of internet connectivity and communication. Enabling seamless interactions between different software applications, APIs are integral to the operation of web services, cloud technologies, and mobile applications. Astonishingly, they constitute 83% [2] of internet traffic, highlighting their critical role in the digital ecosystem. However, this substantial volume of API traffic also presents significant security challenges. APIs, by their very nature, expose application logic and sensitive data, making them attractive targets for cyberattacks [3]. As the conduits through which different software services communicate, APIs, if left unprotected, can become the weakest link in an organization's cybersecurity armor.

As the custodian of a significant portion of Norway's wealth, NBIM must maintain impeccable cybersecurity practices, a mandate that includes rigorous API security. In line with this imperative, the assignment involves developing a comprehensive report that outlines best practices for securing authentication and authorization of APIs, coupled with a proof of concept for integrating digital identities into these APIs. The absence of robust identity verification can lead to data breaches and unauthorized access to sensitive data and services. By incorporating digital identities, organizations can establish a strong link between API requests and legitimate users, apply fine-grained access control, and prevent fraudulent and malicious access attempts [4].

A.1.2 Project Goals

Effect goals

- Create a report outlining best practices for authentication and authorization of APIs.
- Improve security measures and practices for API usage at NBIM.
- Use GitHub to host source code and Amazon Web Services (AWS) as a deployment environment.
- Evaluate different practices for API security and create a proof of concept demonstrating the implementation of authentication and authorization with a focus on the principle of least privilege.

Result goals

- Create a report which can be used to better secure API security.
- Receive a satisfactory grade for the bachelor thesis.
- Give NBIM a better overview of how to secure APIs and how digital identities can be implemented into an API.
- Create a proof-of-concept of the groups findings, which will be an API with some of the best practice security measures built in.

Learning goals

- Get familiar with scrumban.
- Get a better understanding of APIs.
 - How they work in general.
 - How to implement an API and make it interact with other software.
 - How to secure them.
 - How to implement authorization and authentication in an API.
- Learn how to use cloud computing tools such as AWS.

A.1.3 Framework

- The group will be working on the bachelor project from 04/01/24 to 21/05/24 and deliver the report at 21/05/24.
- The group will present their findings 5th or 6th of June.

A.2 Scope

A.2.1 Problem areas

Digital security can be a modern business' greatest defense, or biggest weakness. 73% of all internet traffic is made from malicious sources and bots [5], all searching for that one mistake in your security configuration. A foundational element of innovation in today's app-driven world is APIs. From banks, retail and transportation to Internet of Things (IoT), autonomous vehicles and smart cities, APIs are a critical part of modern mobile, Software as a Service (SaaS) and web applications and can be found in customer-facing, partner-facing and internal applications [3]. This paper will explain how to properly authenticate and authorize individuals accessing you APIs, and what an API should and shouldn't have to mitigate vulnerabilities and security risks.

A.2.2 Limitations

The group will not deliver an API that is ready to be deployed, only a working proof-of-concept for incorporating digital identities into the API. The group will therefore not be testing the API with already established infrastructure. The testing will be done after specifications the group decides on. The proof-of-concept will be based on modules found in AWS, as the workload of exploring other cloud service providers options would be too great for the tasks scope. The only exception to this is the use of an IDP, the group may use Microsoft's IDP solution instead of AWS' IDP solution.

A.2.3 Task Description

Write a report outlining best practices for authentication and authorization of APIs. The group wants to focus on validating user and system identities and applying fine-grained authorization control. It will be based on reviewing different practices, as well as implementing a technical proof of concept demonstrating how digital identities can be incorporated into APIs. The report will take into consideration the sensitivity of the data the API will grant access to and use best practices to recommend appropriate security controls for authentication and authorization based on a threat model for the API. The group will also consider the developer and user experience and scalability to an enterprise environment.

A.3 Project Organization

A.3.1 Responsibilities and Roles

Group leader: Patrik Andre Olaussen

The group leader is responsible for coordinating and overseeing the general direction of the bachelor project. The role also involves taking important decisions. The group leader will also be responsible for leading the meetings with supervisors and stakeholders, going through the agenda and keeping the meeting on track. In case of conflicts within the group, the leader is responsible for facilitating a resolution.

Head of Communication: André Moen

The head of communication is responsible for facilitating communication between supervisors at Norwegian University of Science and Technology (NTNU) and the stakeholder, NBIM. This role involves being the main contact person between different groups involved.

Secretaries: Arvid Moemeni, André Moen, Patrik Andre Olaussen

The role of the secretary is divided among three individuals. This role involves planning and organizing meetings, booking meeting rooms, sending out invitations, recording meeting minutes, and assisting the group leader when necessary.

- André will be responsible for meetings with supervisors.
- Arvid will be responsible for meetings with NBIM.
- Patrik will be responsible for internal group meetings.

Quality Assurance: Farhad Mangal

QA will be responsible for checking the quality of the task at the end of each sprint log, which is every other week. The quality assurance role aims to enhance the overall quality of the project by implementing best practices and ensuring that the project meets specified criteria and expectations. This can be accomplished by having grammar and language checks, content accuracy and adherence to guidelines. The role will be covered by one person throughout the project but will be collectively shared among all group members during the two last sprint logs.

Source manager: Farhad Mangal

The source manager will have the responsibility to make sure that all the used sources are following the right structure and according to the correct source style.

Scrum master:

The role of scrum master will be rotated among each group member every two weeks. The Scrum Master is responsible for overseeing and managing tasks in the sprint log, leading internal group meetings for coordination, monitoring the delivery process to ensure efficiency and to make sure that the scrum method is properly followed.

A.3.2 Procedures and Group Rules

A.3.2.1 Routines

- Meetings with supervisors will be held at campus every Friday 11:00 – 11:45. If both the supervisors and the group deem a meeting unnecessary in a given week, it will be cancelled.
- Meetings with NBIM, the stakeholder, is set to take place every Wednesday 14:00 – 15:00 through Teams. If both the stakeholder and the group deem a meeting unnecessary in a given week, it will be cancelled.
- It is expected that the group will meet every

- Monday: 12:00 – 16:00 digitally.
 - Thursday: 10:00 – 16:00 at campus.
 - Friday: 10:00 – 11:00 & 11:45 – 15:00 at campus.
- Each group member will individually record their working hours on a shared timesheet. At the end of each week, each member should verify that all hours are logged, along with a description of the tasks completed. The timesheet will be presented during group meetings.
 - It is anticipated that each group member will dedicate a minimum of 30 hours per week to project-related work. Deviations from this expectation should be communicated with a valid reason.
 - All communication within the group should be on the groups discord channel.
 - Communication with supervisors will be through a Teams channel made by the group.
 - Communication with NBIM will be through mail through André.
 - All documents will be stored in SharePoint or Overleaf.
 - All code will be stored in GitHub.
 - The group will make weekly copies of documents and code to ensure that they have local backups, providing an extra layer of security for the work.
 - Tasks to be done will be created and assigned on the scrumban board in the GitHub repository.
 - All code that is committed to GitHub must be commented formally and understandable.

A.3.2.2 Rules

- If a conflict arises within the group, it should be handled internally. If the group is not able to solve the conflict on its own, a supervisor should be contacted.
- When a task has been given to a group member, they should complete it in the given time. If it's not possible, then the rest of the group should be notified so the workload can be redistributed.
- If a group member arrives late to a meeting, they must notify the rest of the group as soon as possible on the group's Discord channel. An arrival of 15 minutes or more after the agreed meeting time will result in a fine in the form of buying a cake for the rest of the group.
- If a group member cannot attend a meeting, they must notify the rest of the group at least 24 hours in advance. It is expected that the group member has a valid reason for not attending. Acute sickness can be notified the same day as a meeting.

A.4 Planning, Monitoring, and Reporting

A.4.1 Main Division of the Project

A.4.1.1 Project Management Methodology

In selecting the project management methodology, the group carefully considered various traditional and agile approaches. Given the uncertainties in the project development, the group recognized the need for an agile methodology that allows the group to adapt as the project unfolds. Scrum emerged as the choice due to its straightforward framework, which facilitates continuous improvements throughout the project. It encompasses daily stand-up meetings, sprint planning and retrospectives. The group values its simplicity, providing a structured approach to set deadlines and promoting coordination within the group. However, the group identified that a pure

Scrum approach might not entirely address the needs. Especially a way to store and track all given tasks and deadlines. This led the group to scrumban, a hybrid methodology taking the structured approach of Scrum and adding the visual component of Kanban's visual capabilities in the form of a scrumban board. [6]

A.4.1.2 Scrumban

"Scrum is a management framework that groups use to self-organize and work towards a common goal. It describes a set of meetings, tools, and roles for efficient project delivery." [7]. By embracing continuous improvement and adaptability, this methodology allows for agile adjustments throughout the project lifecycle. The work is organized into sprints, each lasting two weeks, with planning conducted every two weeks during group meetings held on Mondays. Tasks will be allocated evenly to the participants, ensuring equitable contributions.

To maintain transparency and keep both the project group and stakeholders well informed about the progress and obstacles in each sprint, regular meetings are conducted after every cycle. These sprint reviews will be held with the stakeholder and the work will be assessed according to the outlined project goals and progress rate. Subsequently, an internal meeting is held after each sprint to conduct a sprint retrospective. These sessions focus on evaluating the previous sprint and identifying potential ways of improving productivity for the upcoming sprint.

The group conducts daily 15-minute stand-up meetings, excluding Tuesdays. Should the group deem it necessary to hold a meeting on Tuesday, then they will arrange one. During these sessions, participants discuss completed tasks emphasizing on exchanging information and outlining their objectives until the next meeting.

By leveraging scrumban boards, the group can visualize the backlog of work tasks, allowing easy tracking of tasks in progress or completed tasks. Developing a scrumban board will also help with transparency within the group and help order the task priorities.

A.4.2 Plan for status meetings and decision points during the period

The group will conduct brief daily stand-up meetings four times a week, at the start of each work session. Each group member will share updates on their progress, tasks for the day, and any obstacles they are facing.

As an aspect of the scrumban methodology, the group will have a meeting every week with stakeholder to review the groups progress and receive further guidance on the project. After the meeting with the stakeholder, the group will have an internal meeting to thoroughly analyze and discuss the received feedback amongst themselves.

A.5 Organization of Quality Assurance

A.5.1 Documentation

In our project we prioritize a structured documentation to foster effective collaboration. All project related documents, including meeting minutes and timesheets are stored in our Microsoft Teams SharePoint channel that our group and supervisors have access to. This is done to promote accessibility and transparency to keep the relevant parties informed about the project progress. This centralized repository is used to ensure secure storage and collaborative access.

To manage tasks efficiently, we use our scrumban board to categorize tasks in the categories: Backlog, Ready to be assigned, in progress and review. The review stage involves a collective evaluation by the group, ensuring that completed tasks align with the project objectives, meet quality standards, and gets approved by all group members. These reviews are the first order of business in every internal group meeting. The scrumban board is stored in our GitHub repository together with any source code and code documentation produced by the group.

A.5.2 Standards

Open Authorization 2.0 (OAuth 2.0)

OAuth 2.0 is an authorization standard that allows third-party applications to attain predetermined access to a service on behalf of the end user.[8] This is done either by coordinating an authorization process between the service and the end user, or by allowing third-party applications to gain access on the end users behalf. After the authorization process is complete, the third-party application is sent an access token which they can use to authenticate themselves until the token duration runs out. These tokens are limited in scope and duration, thus minimizing risk of compromise and severity of attacks. By directing the end user to authorize themselves with the service that stores user resources, the application gets access to the data without needing to store users credentials.

Utilizing OAuth 2.0 in our project offers several key advantages. Firstly, OAuth 2.0 will serve as a secure protocol for authorization and enabling third-party applications to access our service on behalf of the end users. Additionally we will take advantage of the access tokens introduced by OAuth 2.0 to create a a more user-friendly and privacy conscious environment.

Open ID Connect (OIDC)

OIDC is an identity protocol that utilizes the authorization mechanisms of OAuth 2.0. It is used to verify the identity of a user to a client service [9]. OIDC also avoids sharing user credentials with services [10]. The group will employ the OIDC authentication protocol as it is easy, reliable, secure, and eliminates storing and managing users credentials.

Security Assertion Markup Language (SAML)

SAML 2.0 is a standard for exchanging authentication and authorization data between parties, in particular, between an Identity Provider (IDP) and a Service Provider (SP). The group will compare SAML 2.0 with other solutions. When a user attempts to access an API, SAML 2.0 enables the group to confirm the user's identity and authorization by validating the provided assertions, ensuring that only authenticated and authorized individuals can interact with the API.

JSON Web Token (JWT)

JWT is a secure method for transmitting data between two parties, ensuring the integrity of the data remains intact. Encoded in JavaScript Object Notation (JSON) format, the contents of a JWT are signed, meaning any alteration would render the signature invalid [11]. This feature is particularly valuable when it's crucial to guarantee that data exchanged, such as during authorization processes, has not been tampered with. The group will be utilizing JWTs in OAuth 2.0 and OIDC as the preferred format for access- and ID tokens to make sure they haven't been tampered with.

Fast Identity Online (FIDO)

FIDO is a password-less authentication method. It creates a unique cryptographic key pair for each new web service domain the user connects to. The user device retains the private key and registers the public key with the online service [12]. The group will adhere to this standard as it fulfills the stakeholders criteria of usability, security and scalability. The user experience will be familiar and consistent across many of the user's devices in the form of a simple verification of their fingerprints, face, or a device PIN when logging in [13]. FIDO authentication security is proven to be resistant to threats of phishing, credential stuffing and other remote attacks. Furthermore, service providers can offer passkeys without needing passwords as an alternative sign in or account recovery method. Lastly, with passkeys, users do not need to create a new FIDO credential on each service or each new device. The user's passkeys are available whenever they need them - even if they replace their device. This coupled with FIDO being supported by Chrome, Windows, FireFox, iOS, MacOS, and Android [14] makes this a highly scalable solution.

A.5.3 Tools

To develop and deploy the APIs the group is creating a proof-of-concept for, the group will utilize AWS. AWS, a comprehensive cloud computing platform by Amazon, offers an extensive array of functionalities. The groups focus will be on leveraging the API Gateway component of AWS. This

tool is adept at creating, publishing, maintaining, monitoring, and securing both WebSocket and RESTful APIs [15].

To power the APIs, the group will employ AWS Lambda and AWS Elastic Container Service (ECS) with Fargate. AWS Lambda represents a serverless computing service, enabling the execution of code in response to events, all while eliminating the need for server management [16]. The group might need to create containers for some of the tools to work and will be using ECS with Fargate. ECS simplifies container deployment, allowing the group to run containers without the hassle of provisioning, configuring, or scaling clusters of virtual machines [17]. This combination offers an efficient and streamlined approach to managing the API infrastructure.

The group plans to implement a Single Sign On (SSO) solution for authentication, necessitating the use of an IDP. The IDP’s role will be to verify credentials and confirm the identities of users attempting to log in. To fulfill this requirement, the group is considering two potential options: Microsoft Entra and AWS Cognito. Both these platforms will be thoroughly evaluated against each other to determine which best meets the specific needs for secure and efficient user authentication in the proof-of-concept.

A.5.4 Plan for Inspections and Testing

The group will be performing tests on the proof-of-concept API by using Postman to test the APIs and check whether the responses are correct and the right access levels have been achieved.

To confirm that a single endpoint returns the correct response to a given request, the group will be performing unit testing. Before doing these tests, the group must establish a single source of truth for what each request and response should look like.

Additionally the group will test implementation of OAuth 2.0 and OIDC by using free available debuggers, which are used to check that OAuth 2.0 and OIDC has been set up correctly, by changing the redirect URL and checking the authentication code received. The debuggers to be used are as follows:

- <https://oauthdebugger.com/>
- <https://oidcdebugger.com/>

A.5.5 Risk Analysis at Project Level

The following is a risk matrix that visually represents the likelihood and impact of potential risks. It uses colors to indicate whether a risk is acceptable (green), moderate (yellow), or unacceptable (red). [7] We used another bachelor project’s risk matrix as a template for creating ours [18].

Consequences	Severe	1				
	Major		3,7,9,12			
	Moderate		6,8,10	4,5,11		
	Minor				2	
	Insignificant					
	Rare	Unlikely	Possible	Likely	Certain	
	Probability					

Table A.1: Risk Matrix

Risk scenario 1

Risk scenario	Incomplete Project
Description	The project has not met the deadline due to various potential causes, including technical errors, data loss, miscalculation of required time, and other factors.
Probability	Rare
Consequence	Severe
Overall risk	Moderate

Table A.2: Risk matrix: Scenario 1

Measures:

To address the risk of a project not being completed on time, the group plans to establish two internal deadlines. The first deadline is aimed at delivering a preliminary draft to the supervisor for feedback. The second draft will be an improved version based on the feedback from the supervisor. This helps the group make necessary changes before the final deadline, increasing the likelihood of finishing the project on time. If the group realises that they still cannot complete all aspects of the task, this needs to be conveyed to NBIM as soon as possible.

Risk scenario 2

Risk scenario	A group member experiences a minor illness.
Description	A group member is temporarily absent due to a cold or a similar illness for less than a week.
Probability	Likely
Consequence	Minor
Overall risk	Moderate

Table A.3: Risk matrix: Scenario 2

Measures:

Due to the groups regular meetings and the rotation of the Scrum Master role, it is essential for each member to have a comprehensive understanding of the tasks performed by others. Good documentation and communication are important factors in this. In case of a short-term illness, other group members should be able to step in for the absent member.

Risk scenario 3

Risk scenario	A group member experiences a severe illness.
Description	A group member is ill and absent for more than a week.
Probability	Unlikely
Consequence	Major
Overall risk	Moderate

Table A.4: Risk matrix: Scenario 3

Measures:

The group will try to cover for the absent member and carry the extra workload.

Risk scenario 4

Risk scenario	Internal conflict
Description	Disagreements on how to accomplish a task.
Probability	Possible
Consequence	Moderate
Overall risk	Moderate

Table A.5: Risk matrix: Scenario 4

Measures:

The group will frequently meet in person, leading to increased cooperation and cohesion, and planning future work together. This will minimise opportunities for disagreements. However, if a standstill occurs, the current scrum master will have an extra vote to tip the scales. If the group is not able to solve an issue internally, the groups supervisors will be asked for their input.

Risk scenario 5

Risk scenario	Miscommunication internally
Description	Disagreements on how to accomplish a task.
Probability	Possible
Consequence	Moderate
Overall risk	Moderate

Table A.6: Risk matrix: Scenario 5

Measures

As stated earlier, due to the groups frequent meetings and rotating role as scrum master, every member should understand what the others are doing. Furthermore, the group will be using a scrumban board to monitor ongoing work and who does what. With these measures in place, internal miscommunication should be highly unlikely.

Risk scenario 6

Risk scenario	Lack of communication with stakeholder
Description	A stop in communication from NBIM may result in a misunderstanding regarding the assignment's end goals.
Probability	Unlikely
Consequence	Moderate
Overall risk	Moderate

Table A.7: Risk matrix: Scenario 6

Measures:

To keep NBIM interested in working with the group, the group will act professionally, keep deadlines and show up on time to meetings. If communication is lost however, the group would try to contact NBIM through previous channels. If the group still do not get a reply, the group will keep up work on the project and make their own interpretations as to what the result should be like.

Risk scenario 7

Risk scenario	Data loss
Description	Loss of report or source code.
Probability	Unlikely
Consequence	Major
Overall risk	Moderate

Table A.8: Risk matrix: Scenario 7

Measures:

The groups report data will be saved on Microsoft Teams SharePoint and as a project on Overleaf. The source code of the project will be saved to a GitHub repository. At the first meeting each week, all members will pull the repository and save it locally.

Risk scenario 8

Risk scenario	Scope creep
Description	Working outside set project scope. This can happen due to lack of technical understanding or giving to much time and attention to a single task.
Probability	Unlikely
Consequence	Moderate
Overall risk	Moderate

Table A.9: Risk matrix: Scenario 8

Measures:

Having a scrum master always checking in on what everyone is doing several times a week, and four weekly meetings where the group presents what they have worked on to the rest of the group, will minimize the risk of scope creep. If the group or NBIM want to expand the assignment, the group will ask supervisors and NBIM, along with an internal evaluation in the group if it is achievable.

Risk scenario 9

Risk scenario	Changes in stakeholder requirements
Description	NBIM changing the method of task completion, adjusting the objectives they want the group to attain, or what technologies they prefer the group to use.
Probability	Unlikely
Consequence	Major
Overall risk	Moderate

Table A.10: Risk matrix: Scenario 9

Measures:

If possible, the group will accommodate the changes. If it is not possible, the members of the group are the owners of the assignment, and the project work will continue with the original plan and goals in mind.

Risk scenario 10

Risk scenario	Going over budget
Description	Going over budget could cause the group to lose access to vital platforms and software. This would lead to delays and possibly render the group unable to complete or show the proof-of-concept
Probability	Unlikely
Consequence	Moderate
Overall risk	Moderate

Table A.11: Risk matrix: Scenario 10

Measures:

Find out the cost of every platform and service that will be used ahead of time and make a monthly budget. This internal budget will be kept well below the 8000kr maximum. In case of overspending, the group should conduct an quick internal review, make necessary adjustments, and communicate transparently with NBIM to ensure project progress remains on track.

Risk scenario 11

Risk scenario	Unable to complete proof-of-concept
Description	The proof-of-concept is not ready for the deadline due to fault of group members.
Probability	Possible
Consequence	Moderate
Overall risk	Moderate

Table A.12: Risk matrix: Scenario 11

Measures:

If the group is not able to complete the proof-of-concept in time for the deadline due to the groups own failings, they have to inform NBIM as soon as possible. If they provide a good explanation for what they wanted to do and why the group couldn't achieve it, the group have been told it is not a problem, as long as they have something to show. If the group does not have any aspect of the proof-of-concept ready, they will shift their focus to making the report better.

Risk scenario 12

Risk scenario	Technical issues
Description	The proof-of-concept is not ready due to technical difficulties outside of the groups control.
Probability	Unlikely
Consequence	Major
Overall risk	Moderate

Table A.13: Risk matrix: Scenario 12

Measures:

If the group is not able to present the proof-of-concept due to a technical error beyond their control, such as issues with Amazon Web Services or GitHub, the group can seek assistance from NBIM.

A.6 Implementation Plan - Gantt

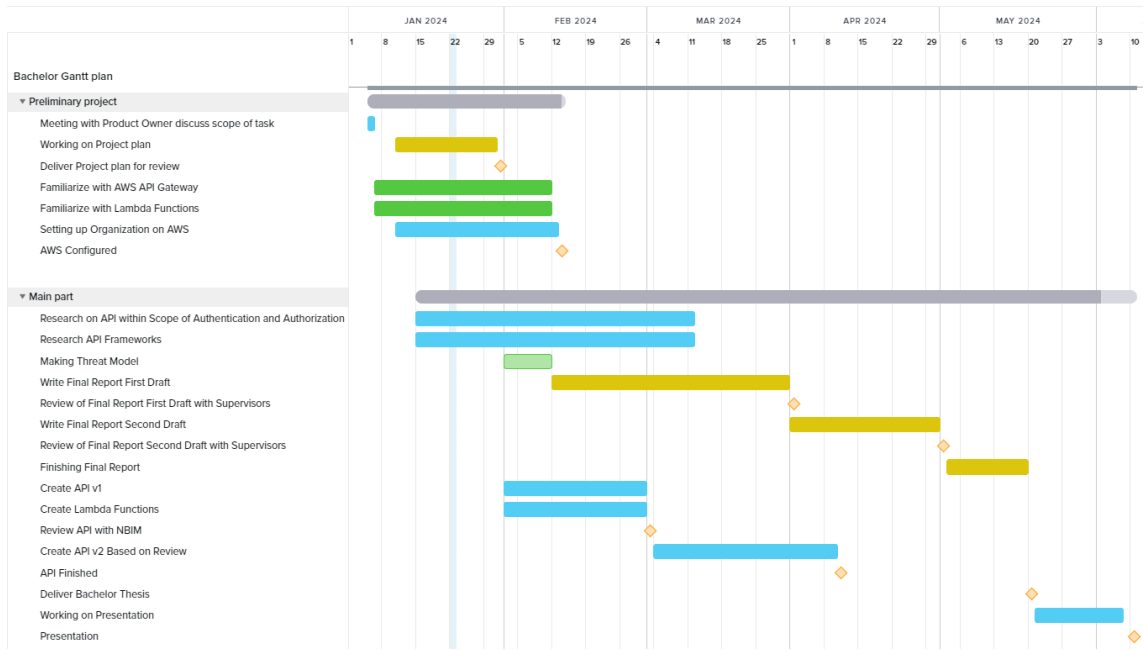


Figure A.1: Gantt Chart

B Standard Agreement

DocuSign Envelope ID: DDAA12F1-F91E-4752-ABCD-25A932996964

11. januar 2024



Fastsatt av prorektor for utdanning 10.12.2020

STANDARDAVTALE

om utføring av studentoppgave i samarbeid med ekstern virksomhet

Avtalen er ufravikelig for studentoppgaver (heretter oppgave) ved NTNU som utføres i samarbeid med ekstern virksomhet.

Forklaring av begrep

Opphavsrett

Er den rett som den som skaper et åndsverk har til å fremstille eksemplarer av åndsverket og gjøre det tilgjengelig for allmennheten. Et åndsverk kan være et litterært, vitenskapelig eller kunstnerisk verk. En studentoppgave vil være et åndsverk.

Eiendomsrett til resultater

Betyr at den som eier resultatene bestemmer over disse. Utgangspunktet er at studenten eier resultatene fra sitt studentarbeid. Studenten kan også overføre eiendomsretten til den eksterne virksomheten.

Bruksrett til resultater

Den som eier resultatene kan gi andre en rett til å bruke resultatene, f.eks. at studenten gir NTNU og den eksterne virksomheten rett til å bruke resultatene fra studentoppgaven i deres virksomhet.

Prosjektbakgrunn

Det partene i avtalen har med seg inn i prosjektet, dvs. som vedkommende eier eller har rettigheter til fra før og som brukes i det videre arbeidet med studentoppgaven. Dette kan også være materiale som tredjepersoner (som ikke er part i avtalen) har rettigheter til.

Utsatt offentliggjøring

Betyr at oppgaven ikke blir tilgjengelig for allmennheten før etter en viss tid, f.eks. før etter tre år. Da vil det kun være veileder ved NTNU, sensorene og den eksterne virksomheten som har tilgang til studentarbeidet de tre første årene etter at studentarbeidet er innlevert.

1. Avtaleparter

Norges teknisk-naturvitenskapelige universitet (NTNU) Institutt: Institutt for informasjonssikkerhet og kommunikasjonsteknologi IIK
Veileder ved NTNU: e-post og tlf. Guoqiang Li: guoqiang.li@ntnu.no +47 979 27 047 Erjon Zoto: erjon.zoto@ntnu.no +47 984 33 097
Ekstern virksomhet: Norges Bank, dens avdeling Norges Bank Investment Management Ekstern virksomhet sin kontaktperson, e-post og tlf.: Stian Hagbø Olsen, stian.hagbo.olsen@nbim.no , tlf. +47 92829840 Celina Heimdal Brynildsen, celina.heimdal.brynildsen@nbim.no , tlf. +47 9742 9588
Student: André Moen Fødselsdato: 14.07.02
Student: Arvid Moemeni Fødselsdato: 21.12.95
Student: Farhad Mangal Fødselsdato: 23.01.95
Student: Patrik Andre Olaussen Fødselsdato: 08.05.98

Partene har ansvar for å klarere eventuelle immaterielle rettigheter som studenten, NTNU, den eksterne eller tredjeperson (som ikke er part i avtalen) har til prosjektbakgrunn før bruk i forbindelse med utførelse av oppgaven. Eierskap til prosjektbakgrunn skal fremgå av eget vedlegg til avtalen der dette kan ha betydning for utførelse av oppgaven.

2. Utførelse av oppgave

Studenten skal utføre: (sett kryss)

Masteroppgave	
Bacheloroppgave	X
Prosjektoppgave	
Annen oppgave	

Startdato: 04.01.24
Sluttdato: 11.06.24

Oppgavens arbeidstittel er:
Bachelor Thesis: Incorporating Digital Identities into APIs

Ansvarlig veileder ved NTNU har det overordnede faglige ansvaret for utforming og godkjenning av prosjektbeskrivelse og studentens læring.

3. Ekstern virksomhet sine plikter

Ekstern virksomhet skal stille med en kontaktperson som har nødvendig faglig kompetanse til å gi studenten tilstrekkelig veiledning i samarbeid med veileder ved NTNU. Ekstern kontaktperson fremgår i punkt 1.

Formålet med oppgaven er studentarbeid. Oppgaven utføres som ledd i studiet. Studenten skal ikke motta lønn eller lignende godtgjørelse fra den eksterne for studentarbeidet. Utgifter knyttet til gjennomføring av oppgaven skal dekkes av den eksterne. Aktuelle utgifter kan for eksempel være reiser, materialer for bygging av prototyp, innkjøp av prøver, tester på lab, kjemikalier. Studenten skal klarere dekning av utgifter med ekstern virksomhet på forhånd.

Ekstern virksomhet skal dekke følgende utgifter til utførelse av oppgaven:
NBIM dekker kostnader til innkjøp av software lisenser og liknende, oppad begrenset til totalt kr. 8000 (ex. mva). Kostnadene refunderes mot fremleggelse av kvitteringer. Studentene må bli enige seg imellom om hvordan dette totalbeløpet skal benyttes.

For ordens skyld, NBIM dekker ikke kostnader til reiser.

Ekstern virksomhet stiller med to kontaktpersoner, som hver vil bidra med én time per uke for hele studentgruppen (ikke per student). Veiledningen vil skje via Teams.

Dekning av utgifter til annet enn det som er oppført her avgjøres av den eksterne underveis i arbeidet.

4. Studentens rettigheter

Studenten har opphavsrett til oppgaven¹. Alle resultater av oppgaven, skapt av studenten alene gjennom arbeidet med oppgaven, eies av studenten med de begrensninger som følger av punkt 5, 6 og 7 nedenfor. Eiendomsretten til resultatene overføres til ekstern virksomhet hvis punkt 5 b er avkrysset eller for tilfelle som i punkt 6 (overføring ved patenterbare oppfinnelser).

I henhold til lov om opphavsrett til åndsverk beholder alltid studenten de ideelle rettigheter til eget åndsverk, dvs. retten til navngivelse og vern mot krenkende bruk.

¹ Jf. Lov om opphavsrett til åndsverk mv. av 15.06.2018 § 1

Studenten har rett til å inngå egen avtale med NTNU om publisering av sin oppgave i NTNUs institusjonelle arkiv på Internett (NTNU Open). Studenten har også rett til å publisere oppgaven eller deler av den i andre sammenhenger dersom det ikke i denne avtalen er avtalt begrensninger i adgangen til å publisere, jf. punkt 8.

5. Den eksterne virksomheten sine rettigheter

Der oppgaven bygger på, eller videreutvikler materiale og/eller metoder (prosjektbakgrunn) som eies av den eksterne, eies prosjektbakgrunnen fortsatt av den eksterne. Hvis studenten skal utnytte resultater som inkluderer den eksterne sin prosjektbakgrunn, forutsetter dette at det er inngått egen avtale om dette mellom studenten og den eksterne virksomheten.

Alternativ a) (sett kryss) Hovedregel

N/A	Ekstern virksomhet skal ha bruksrett til resultatene av oppgaven
-----	--

Dette innebærer at ekstern virksomhet skal ha rett til å benytte resultatene av oppgaven i egen virksomhet. Retten er ikke-eksklusiv.

Alternativ b) (sett kryss) Unntak

N/A	Ekstern virksomhet skal ha eiendomsretten til resultatene av oppgaven og studentens bidrag i ekstern virksomhet sitt prosjekt
-----	---

Begrunnelse for at ekstern virksomhet har behov for å få overført eiendomsrett til resultatene: N/A

6. Godtgjøring ved patenterbare oppfinnelser

Dersom studenten i forbindelse med utførelsen av oppgaven har nådd frem til en patenterbar oppfinnelse, enten alene eller sammen med andre, kan den eksterne kreve retten til oppfinnelsen overført til seg. Dette forutsetter at utnyttelsen av oppfinnelsen faller inn under den eksterne sitt virksomhetsområde. I så fall har studenten krav på rimelig godtgjøring. Godtgjøringen skal fastsettes i samsvar med arbeidstakeroppfinnelsesloven § 7. Fristbestemmelsene i § 7 gis tilsvarende anvendelse.

7. NTNU sine rettigheter

De innleverte filer av oppgaven med vedlegg, som er nødvendig for sensur og arkivering ved NTNU, tilhører NTNU. NTNU får en vederlagsfri bruksrett til resultatene av oppgaven, inkludert vedlegg til denne, og kan benytte dette til undervisnings- og forskningsformål med de eventuelle begrensninger som fremgår i punkt 8.

8. Utsatt offentliggjøring

Hovedregelen er at studentoppgaver skal være offentlige.

Sett kryss

<input checked="" type="checkbox"/>	Oppgaven skal være offentlig
-------------------------------------	------------------------------

I særlige tilfeller kan partene bli enige om at hele eller deler av oppgaven skal være undergitt utsatt offentliggjøring i maksimalt tre år. Hvis oppgaven unntas fra offentliggjøring, vil den kun være tilgjengelig for student, ekstern virksomhet og veileder i denne perioden. Sensurkomiteen vil ha tilgang til oppgaven i forbindelse med sensur. Student, veileder og sensorer har taushetsplikt om innhold som er unntatt offentliggjøring.

Oppgaven skal være underlagt utsatt offentliggjøring i (sett kryss hvis dette er aktuelt):

Sett kryss

Sett dato

<input type="checkbox"/>	N/A	ett år	
<input type="checkbox"/>	N/A	to år	
<input type="checkbox"/>	N/A	tre år	

Behovet for utsatt offentliggjøring er begrunnet ut fra følgende:

N/A

Dersom partene, etter at oppgaven er ferdig, blir enig om at det ikke er behov for utsatt offentliggjøring, kan dette endres. I så fall skal dette avtales skriftlig.

Vedlegg til oppgaven kan unntas ut over tre år etter forespørsel fra ekstern virksomhet. NTNU (ved instituttet) og student skal godta dette hvis den eksterne har saklig grunn for å be om at et eller flere vedlegg unntas. Ekstern virksomhet må sende forespørsel før oppgaven leveres.

De delene av oppgaven som ikke er undergitt utsatt offentliggjøring, kan publiseres i NTNUs institusjonelle arkiv, jf. punkt 4, siste avsnitt. Selv om oppgaven er undergitt utsatt offentliggjøring, skal ekstern virksomhet legge til rette for at studenten kan benytte hele eller deler av oppgaven i forbindelse med jobbsøknader samt videreføring i et master- eller doktorgradsarbeid.

9. Generelt

Denne avtalen skal ha gyldighet foran andre avtaler som er eller blir opprettet mellom to av partene som er nevnt ovenfor. Dersom student og ekstern virksomhet skal inngå avtale om

konfidensialitet om det som studenten får kjennskap til i eller gjennom den eksterne virksomheten, kan NTNUs standardmal for konfidensialitetsavtale benyttes.

Den eksterne sin egen konfidensialitetsavtale, eventuell konfidensialitetsavtale den eksterne har inngått i samarbeidprosjekter, kan også brukes forutsatt at den ikke inneholder punkter i motstrid med denne avtalen (om rettigheter, offentliggjøring mm). Dersom det likevel viser seg at det er motstrid, skal NTNUs standardavtale om utføring av studentoppgave gå foran. Eventuell avtale om konfidensialitet skal vedlegges denne avtalen.

Eventuell uenighet som følge av denne avtalen skal søkes løst ved forhandlinger. Hvis dette ikke fører frem, er partene enige om at tvisten avgjøres ved voldgift i henhold til norsk lov. Tvisten avgjøres av sorenskriveren ved Sør-Trøndelag tingrett eller den han/hun oppnevner.

Denne avtale er signert i sju eksemplarer hvor partene skal ha hvert sitt eksemplar. Avtalen er gyldig når den er underskrevet av NTNU v/instituttleder.

Signaturer:

Norges teknisk-naturvitenskapelige universitet (NTNU)


Instituttleder: Basel Katt

Dato:

Veileder ved NTNU:

Guoqiang Li

Dato: 11 January 2024 | 3:27

DocuSigned by:
PM CET

C5FA5F3874E44B6...

Erjon Zoto

Dato: 12 January 2024 | 1:31

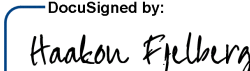
DocuSigned by:
PM CET

2F49B998424E481...

Ekstern virksomhet: Norges Bank, dens avdeling Norges Bank Investment Management

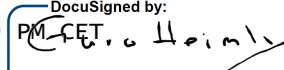
Hakon Fjelberg (Global Head of Technology)

Dato: 11 January 2024 | 3:25 PM CET

DocuSigned by:

91D4B52F1F12454...

Guro Heimly (Legal Counsel)

Dato: 11 January 2024 | 2:40

DocuSigned by:
PM CET

27634E21C8A545B...

Student: André Moen

Dato: 11 January 2024 | 4:26 PM CET

DocuSigned by:

André Moen

422ACCAE31ED430...

Student: Arvid Moemeni

Dato: 11 januar 2024 | 6:43 PM CET

DocuSigned by:

Arvid Moemeni

48E1726E67184DE...

Student: Farhad Mangal

Dato: 12 January 2024 | 10:35 AM CET

DocuSigned by:

Farhad Mangal

8CC3179E4CAD4E...

Student: Patrik Andre Olausen

Dato: 11 January 2024 | 6:56 PM CET

DocuSigned by:

Patrik Andre Olausen

43604A0B4C004EE...

Certificate Of Completion

Envelope Id: DDAA12F1F91E4752ABCD25A932996964	Status: Sent
Subject: Complete with DocuSign: 2024 Standard Agreement NTNU_final(1259.1).docx	
Source Envelope:	
Document Pages: 7	Signatures: 8
Certificate Pages: 6	Initials: 0
AutoNav: Enabled	Envelope Originator:
Envelopeld Stamping: Enabled	Carina Østli
Time Zone: (UTC+01:00) Brussels, Copenhagen, Madrid, Paris	Bankplassen 2
	0151, Oslo 0151
	carina.ostli@nbim.no
	IP Address: 134.238.48.150

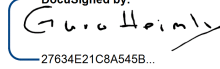
Record Tracking

Status: Original	Holder: Carina Østli	Location: DocuSign
1/11/2024 1:56:30 PM	carina.ostli@nbim.no	

Signer Events

Guro Heimly
guh@nbim.no
Legal Advisor
Security Level: Email, Account Authentication (None)

Signature

DocuSigned by:

27634E21C8A545B...
Signature Adoption: Drawn on Device
Using IP Address: 134.238.48.150

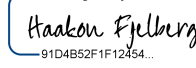
Timestamp

Sent: 1/11/2024 2:37:56 PM
Viewed: 1/11/2024 2:39:47 PM
Signed: 1/11/2024 2:40:49 PM

Electronic Record and Signature Disclosure:

Accepted: 1/11/2024 2:39:47 PM
ID: 9a83cd70-47fd-4f6e-b0a6-81639b5f5b0e

Haakon Fjelberg
hfj@nbim.no
Global Head of Technology
Security Level: Email, Account Authentication (None)

DocuSigned by:

91D4B52F1F12454...
Signature Adoption: Pre-selected Style
Using IP Address: 134.238.46.172

Sent: 1/11/2024 2:37:57 PM
Viewed: 1/11/2024 3:23:22 PM
Signed: 1/11/2024 3:25:07 PM

Electronic Record and Signature Disclosure:

Accepted: 1/11/2024 3:23:22 PM
ID: 87d6ef72-1f15-4c27-a64d-8e310f9b72f8

André Moen
amoe@stud.ntnu.no
Security Level: Email, Account Authentication (None)

DocuSigned by:

422ACCAE31ED430...
Signature Adoption: Pre-selected Style
Using IP Address: 158.248.1.131

Sent: 1/11/2024 3:25:09 PM
Viewed: 1/11/2024 4:18:06 PM
Signed: 1/11/2024 4:26:59 PM

Electronic Record and Signature Disclosure:

Accepted: 1/11/2024 4:18:06 PM
ID: e4696dae-54bf-4f13-97b1-8bb805c01af6

Arvid Moemeni
arvid.moemeni@ntnu.no
Security Level: Email, Account Authentication (None)

DocuSigned by:

48E1726E67184DE...
Signature Adoption: Pre-selected Style
Using IP Address: 84.210.146.106

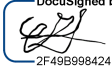
Sent: 1/11/2024 3:25:10 PM
Viewed: 1/11/2024 6:39:39 PM
Signed: 1/11/2024 6:43:06 PM


Electronic Record and Signature Disclosure:

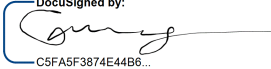
Accepted: 1/11/2024 6:39:39 PM
ID: e41b48fe-9560-41aa-ba6b-90043c7f6dbc

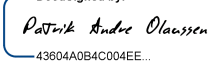
Signer Events	Signature	Timestamp
---------------	-----------	-----------

<p>Basel Katt basel.katt@ntnu.no Security Level: Email, Account Authentication (None) Electronic Record and Signature Disclosure: Not Offered via DocuSign</p>		<p>Sent: 1/11/2024 3:25:14 PM</p>
---	--	-----------------------------------

<p>Erjon Zoto erjon.zoto@ntnu.no Security Level: Email, Account Authentication (None) Electronic Record and Signature Disclosure: Accepted: 1/12/2024 1:27:27 PM ID: 486329af-e6b4-4735-a17c-172f0c204fe4</p>	<p>DocuSigned by:  2F49B998424E481...</p> <p>Signature Adoption: Drawn on Device Using IP Address: 85.164.77.22</p>	<p>Sent: 1/11/2024 3:25:11 PM Viewed: 1/12/2024 1:27:27 PM Signed: 1/12/2024 1:31:35 PM</p>
--	--	---

<p>Farhad Mangal farhad.mangal@ntnu.no Security Level: Email, Account Authentication (None) Electronic Record and Signature Disclosure: Accepted: 1/11/2024 3:30:12 PM ID: c819e44f-2fd2-4eb2-a45c-f771644d86b9</p>	<p>DocuSigned by:  8CC3179E4CAD44E...</p> <p>Signature Adoption: Pre-selected Style Using IP Address: 84.213.11.139</p>	<p>Sent: 1/11/2024 3:25:11 PM Viewed: 1/11/2024 3:30:12 PM Signed: 1/12/2024 10:35:06 AM</p>
--	--	--

<p>Guoqiang Li guoqiang.li@ntnu.no Security Level: Email, Account Authentication (None) Electronic Record and Signature Disclosure: Accepted: 1/11/2024 3:26:35 PM ID: ae32972a-b416-40f8-9bb4-cf5100f1d0e7</p>	<p>DocuSigned by:  C5FA5F3874E44B6...</p> <p>Signature Adoption: Drawn on Device Using IP Address: 129.241.236.117</p>	<p>Sent: 1/11/2024 3:25:12 PM Viewed: 1/11/2024 3:26:35 PM Signed: 1/11/2024 3:27:52 PM</p>
--	---	---

<p>Patrik Andre Olaussen patrikao@stud.ntnu.no Security Level: Email, Account Authentication (None) Electronic Record and Signature Disclosure: Accepted: 1/11/2024 6:53:48 PM ID: 33ba5c9c-1e32-4ad2-8973-5e12f73d3655</p>	<p>DocuSigned by:  43604A0B4C004EE...</p> <p>Signature Adoption: Pre-selected Style Using IP Address: 178.232.36.55</p>	<p>Sent: 1/11/2024 3:25:13 PM Viewed: 1/11/2024 6:53:48 PM Signed: 1/11/2024 6:56:38 PM</p>
--	--	---

In Person Signer Events	Signature	Timestamp
-------------------------	-----------	-----------

Editor Delivery Events	Status	Timestamp
------------------------	--------	-----------

Agent Delivery Events	Status	Timestamp
-----------------------	--------	-----------

Intermediary Delivery Events	Status	Timestamp
------------------------------	--------	-----------

Certified Delivery Events	Status	Timestamp
---------------------------	--------	-----------

Carbon Copy Events	Status	Timestamp
Witness Events		
	Signature	Timestamp
Notary Events		
	Signature	Timestamp
Envelope Summary Events		
	Status	Timestamps
Envelope Sent	Hashed/Encrypted	1/11/2024 2:37:57 PM
Certified Delivered	Security Checked	1/11/2024 6:53:48 PM
Signing Complete	Security Checked	1/11/2024 6:56:38 PM
Payment Events		
	Status	Timestamps
Electronic Record and Signature Disclosure		

ELECTRONIC RECORD AND SIGNATURE DISCLOSURE

From time to time, Norges Bank (we, us or Company) may be required by law to provide to you certain written notices or disclosures. Described below are the terms and conditions for providing to you such notices and disclosures electronically through the DocuSign system. Please read the information below carefully and thoroughly, and if you can access this information electronically to your satisfaction and agree to this Electronic Record and Signature Disclosure (ERSD), please confirm your agreement by selecting the check-box next to 'I agree to use electronic records and signatures' before clicking 'CONTINUE' within the DocuSign system.

Getting paper copies

At any time, you may request from us a paper copy of any record provided or made available electronically to you by us. You will have the ability to download and print documents we send to you through the DocuSign system during and immediately after the signing session and, if you elect to create a DocuSign account, you may access the documents for a limited period of time (usually 30 days) after such documents are first sent to you. After such time, if you wish for us to send you paper copies of any such documents from our office to you, you will be charged a \$0.00 per-page fee. You may request delivery of such paper copies from us by following the procedure described below.

Withdrawing your consent

If you decide to receive notices and disclosures from us electronically, you may at any time change your mind and tell us that thereafter you want to receive required notices and disclosures only in paper format. How you must inform us of your decision to receive future notices and disclosure in paper format and withdraw your consent to receive notices and disclosures electronically is described below.

Consequences of changing your mind

If you elect to receive required notices and disclosures only in paper format, it will slow the speed at which we can complete certain steps in transactions with you and delivering services to you because we will need first to send the required notices or disclosures to you in paper format, and then wait until we receive back from you your acknowledgment of your receipt of such paper notices or disclosures. Further, you will no longer be able to use the DocuSign system to receive required notices and consents electronically from us or to sign electronically documents from us.

All notices and disclosures will be sent to you electronically

Unless you tell us otherwise in accordance with the procedures described herein, we will provide electronically to you through the DocuSign system all required notices, disclosures, authorizations, acknowledgements, and other documents that are required to be provided or made available to you during the course of our relationship with you. To reduce the chance of you inadvertently not receiving any notice or disclosure, we prefer to provide all of the required notices and disclosures to you by the same method and to the same address that you have given us. Thus, you can receive all the disclosures and notices electronically or in paper format through the paper mail delivery system. If you do not agree with this process, please let us know as described below. Please also see the paragraph immediately above that describes the consequences of your electing not to receive delivery of the notices and disclosures electronically from us.

How to contact Norges Bank:

You may contact us to let us know of your changes as to how we may contact you electronically, to request paper copies of certain information from us, and to withdraw your prior consent to receive notices and disclosures electronically as follows:

To contact us by email send messages to: nde@nbim.no

To advise Norges Bank of your new email address

To let us know of a change in your email address where we should send notices and disclosures electronically to you, you must send an email message to us at nde@nbim.no and in the body of such request you must state: your previous email address, your new email address. We do not require any other information from you to change your email address.

If you created a DocuSign account, you may update it with your new email address through your account preferences.

To request paper copies from Norges Bank

To request delivery from us of paper copies of the notices and disclosures previously provided by us to you electronically, you must send us an email to nde@nbim.no and in the body of such request you must state your email address, full name, mailing address, and telephone number. We will bill you for any fees at that time, if any.

To withdraw your consent with Norges Bank

To inform us that you no longer wish to receive future notices and disclosures in electronic format you may:

- i. decline to sign a document from within your signing session, and on the subsequent page, select the check-box indicating you wish to withdraw your consent, or you may;
- ii. send us an email to nde@nbim.no and in the body of such request you must state your email, full name, mailing address, and telephone number. We do not need any other information from you to withdraw consent.. The consequences of your withdrawing consent for online documents will be that transactions may take a longer time to process..

Required hardware and software

The minimum system requirements for using the DocuSign system may change over time. The current system requirements are found here: <https://support.docusign.com/guides/signer-guide-signing-system-requirements>.

Acknowledging your access and consent to receive and sign documents electronically

To confirm to us that you can access this information electronically, which will be similar to other electronic notices and disclosures that we will provide to you, please confirm that you have read this ERSD, and (i) that you are able to print on paper or electronically save this ERSD for your future reference and access; or (ii) that you are able to email this ERSD to an email address where you will be able to print on paper or save it for your future reference and access. Further, if you consent to receiving notices and disclosures exclusively in electronic format as described herein, then select the check-box next to 'I agree to use electronic records and signatures' before clicking 'CONTINUE' within the DocuSign system.

By selecting the check-box next to 'I agree to use electronic records and signatures', you confirm that:

- You can access and read this Electronic Record and Signature Disclosure; and
- You can print on paper this Electronic Record and Signature Disclosure, or save or send this Electronic Record and Disclosure to a location where you can print it, for future reference and access; and
- Until or unless you notify Norges Bank as described above, you consent to receive exclusively through electronic means all notices, disclosures, authorizations, acknowledgements, and other documents that are required to be provided or made available to you by Norges Bank during the course of your relationship with Norges Bank.

C Thesis Descripton

Oppgave 22

DIGSEC

3 stk

Oppgavetittel: Incorporation Digital Identities into APIs

Bedrift: Norwegian Bank Investment Management (NBIM)

Kontaktperson: Celina Heimdal Brynhildsen

E-post: Celina.heimdal.brynhildsen@nbim.no

Telefon: 97429588

Lokasjon: Nittedal

Se vedlegg for oppgavebeskrivelse.

Bachelor Thesis: Incorporating Digital Identities into APIs

Company: Norges Bank Investment Management (NBIM)

Address: Bankplassen 2
P.O. Box 1179 Sentrum
NO-0107 Oslo, Norway

Contact persons:

- Astri Marie Ravnaas, +47 91 69 07 85, astri.marie.ravnaas@nbim.no
- Celina Heimdal Brynildsen, +47 97 42 95 88, celina.heimdal.brynildsen@nbim.no

Background

Incorporating Digital Identities into APIs (Application Programming Interface) is a fundamental step in securing the modern digital ecosystem. APIs often provide access to a wide range of functionalities and data, and not all users and services should have access to all the offerings behind the API to adhere to the principal of least privilege. Digital Identities are digital equivalents of real-world identification, where the absence of robust identity verification can lead to data breaches and unauthorized access to sensitive data and services. By leveraging Digital Identities, organizations can establish a strong link between API requests and legitimate users, apply fine-grained access control and prevent fraudulent and malicious access attempts. Incorporating Digital Identities into APIs ultimately builds trust in the digital landscape.

Goal

Create a report outlining best practices for authentication and authorization of APIs. We want to focus on the validating user and system identities and applying fine-grained authorization control. It should be based on reviewing different practices, as well as implementing a technical proof of concept demonstrating how Digital Identities can be incorporated into APIs. The report should take into consideration the sensitivity of the data the API will grant access to and use best practices to recommend appropriate security controls for authentication and authorization based on a threat model for the API. The developer and user experience and scalability to an enterprise environment should be taken into consideration.

Summary

- Create a report outlining best practices for authentication and authorization of APIs.
- Use GitHub to host source code and AWS as deployment environment.

- Evaluate different practices for API security and create a proof of concept demonstrating implementation of authentication and authorization with focus on the principal of least privilege.

D Timetables

D.1 Summary Table

Week no	André Moen	Arvid Moemeni	Farhad Mangal	Patrik Andre Olausen	Sum hours week
Week 1	30	28	30	27	115
Week 2	31	28	30	30	119
Week 3	31	30	30	28	119
Week 4	32	30	28	30	120
Week 5	30	30	24	22	106
Week 6	30	30	26	30	116
Week 7	30	31	29	30	120
Week 8	30	30	31	30	121
Week 9	30	30	31	30	121
Week 10	31	30	30	30	121
Week 11	30	30	28.5	29	117.5
Week 12	25	20	24	8	77
Week 13	46	41	37	31	155
Week 14	34	30	28	27	119
Week 15	30	29	28	25	112
Week 16	32	31	30	30	123
Week 17	31	31	30	26	118
Week 18	31	25	33	26	115
Week 19	42	36	48	38	164
Week 20	18	18	18	18	72
Week 21	0	0	0	0	0
Week 22	0	0	0	0	0
Total sum hours pr person	624	588	593.5	545	2350.5

D.2 Timetable - André

Timesheet	Category	Duration (hours)	Work done	Timesheet	Category	Duration (hours)	Work done
Information search	Administration	1	Scheduling, planning, preparing and sending summons for first meeting with	Project report	Quality assurance	11	Continue writing main part about authentication, using ISO, IAM, Forrester, Standup
Team meeting with stakeholder	Administration	1	Meeting with stakeholders.	Team meetings		1	Standup
Team meetings	Administration	1	Group discussion after meeting with stakeholders.	Project report	Quality assurance	2	Fixed some of the layout
Project plan	Administration	6	Discussed project plan and started writing it.	Self-education		8	Gartner, NIST, Federation, CSI cloud top 10, IAM
Self-education	Documentation	4	Reading previous bachelor thesis.	Team meetings		1	Standup
Team meetings with supervisor	Documentation	1	Meeting with supervisors.	Project report		2	Creating figures for best practice with Arvid
Project plan	Documentation	1	Individual work on the project plan.				
Self-education	Documentation	2	Lecture Lynkurs i prosjektskyring.	Team meeting with stakeholder		1	
Self-education	Documentation	9	Reading up on AWS API gateway.	Team meetings		3	Went over threat model and some comments from Celina
Self-education	Documentation	2	Reading up on OpenID, JWT and OAuth2.0	Team meetings with supervisor		1	
Week 1		30		Week 11		30	
Self-education	Category	Duration (hours)	Work done	Self-education	Category	Duration (hours)	Work done
Self-education		6	Read up on OIDC, OAuth 2.0, JWT, SAML and IAM.	Project report		8	Started writing about digital identities and thinking about which ones to use
Team meetings		1	Planned the week.	Team meetings		5	Restructure of main part of report
Self-education		6	API security, OIDC OAuth 2.0	Self-education		6	IAM, zero trust and digital identities
Project plan		1	Formatting report in latex	Team meetings		3	Standup
Self-education		6	Reading up on OWASP top 10	Project report	Quality assurance	3	Going through comments giving responses to them and fixing what i can on my
Team meeting with stakeholder		1	Second meeting with stakeholders				
Team meetings		1	Going over what has been done and making plans for finishing up the project				
Self-education		1	Reading up on cloud security alliance prep kit				
Team meetings		4	Team meeting with focus on project plan, Git kanban setup and gant.				
Project plan		2	Migrated word document to latex, wrote background and finished up gant				
Team meetings		1	Preparation for meeting with supervisors				
Team meetings with supervisor		1	Went over some questions regarding the project plan.				
Week 2		31		Week 12		25	
Self-education	Category	Duration (hours)	Work done	Self-education	Category	Duration (hours)	Work done
Self-education		13	SAML, OIDC, OAuth 2.0, threat modeling, XACML, AWS stuff, Azure Entra.	Project report	Quality assurance	4	Going over what has been done on the report since last time and fixing up text
Project plan	Quality assurance	2	Read through project plan, correcting errors, commenting on weak parts and	Project report		14	Writing securing the api, digital identity, Session management, MFA, IAM,
Team meetings		1	Read up on what has been done and making plans for finishing up the project	Self-education		6	CSA IAM, microsoft zero trust, entra, AWS WA logging, zero trust
Project plan		1	Wrote part about OWASP top 10 and JWT	Team meetings		2	Standup meeting
Team meeting with stakeholder		1	Demo of how the proof-of-concept should look.	Team meeting with stakeholder		1	NBIM meeting
Team meetings		1	Discussed meeting with stakeholders	Quality assurance		5	Reading through whole rapport, fixing grammar and commenting on weak
Project plan		2	Fixing order of sources, creating glossary and acronym list and fixing format	Team meetings		12	Reading through report and preparing first draft
Self-education		2	AWS CU	Team meetings with supervisor		1	
Team meetings		3	Read up on how to integrate git with AWS and WAF				
Team meetings with supervisor		1	Meeting with supervisors, went through some questions.				
Team meetings		3	Group meeting, preparation for meeting with supervisors and going through				
Week 3		31		Week 13		46	
Team meetings	Category	Duration (hours)	Work done	Self-education	Category	Duration (hours)	Work done
Self-education		1	Team meeting	Self-education		5	Reading about apiV2, cf, waf, apiV1 vs v2, manifest
Project plan		2	Looking for good papers about API security.	Team meetings		4	Standup meeting
Self-education		2	Fixing acronym and glossary lists.	AWS		3	Troubleshooting previous cloudformation template
Self-education		6	Reading API security in action and AWS CloudFormation	AWS		4	Troubleshooting AWS and Entra id with Arvid
Self-education		7	Reading API security in Action	AWS		2	Setting up APIv2 with working OAuth
Project plan	Quality assurance	1	Reading through plan and fixing issues	AWS		3	Setting up cloudfront for APIv2, new endpoints with authorization
Team meetings		3	Discussing feedback from supervisors on project plan.	AWS		3	Going through what has been done with arvid and patrik
Team meetings		6	Worked on API v1 with group	AWS		6	Making cloudformation template for APIv2
AWS		3	Trying to figure out cloudformation	Microsoft Entra		2	Fixing token to send v2 token, started creating custom policy
Team meetings		1	Preparation for meeting with supervisors.				
Team meeting with stakeholder		1					
Week 4		31		Week 14		36	
Self-education	Category	Duration (hours)	Work done	Self-education	Category	Duration (hours)	Work done
Self-education		13	API security in action, API keys, lambda authorizer with JWT	Self-education		3	Figuring out what has been done, reading about adding custom scopes based
Team meetings		1	Discuss what has been done and plan what has to be done	Microsoft Entra		6	Working on Entra with Andre and Patrik
AWS		5	Made cloudformation template with working endpoints and usage plan	Team meetings		4	Standup
AWS		4	Trouble shooting api and creating template with 3 endpoints and looking into	Microsoft Entra		2	Fixing entra to work with different groups access different apps
AWS		1	Creating possible WAF blocking IP sets	Microsoft Entra		1	Working with Arvid on removing unnecessary parts of entra setup
Team meeting with stakeholder		1	Went over cloudformation template and received feedback on it	Microsoft Entra		1	Cleaning up Entra and AWS
AWS		4	Implementing WAF and making sure it works	AWS		5	Updating cloudformation template to work with new entra id setup and writing
Team meetings		1	Patrik showed what has been done in entra	Presentation including preparation		2	Going through PoC guide Andre made, showing how to implement it, reading
				Team meeting with stakeholder		1	Presenting PoC
				AWS		1	Troubleshooting problem with cloudfront which blocks requests from postman
				Self-education		1	Reading proof of concept from earlier thesis
				Team meetings with supervisor		1	Supervisor meeting, getting feedback on thesis so far
				Project report		2	Started fixing comments from Erjon and Li
Week 5		30		Week 15		30	
Self-education	Category	Duration (hours)	Work done	Self-education	Category	Duration (hours)	Work done
Self-education		20	API security in action, ECC, DH, RSA, TLS, HMAC	Project report		3	Going through what has been done and commenting on it and discussing some
Team meetings		2	Went over what has been done and made plans for what to be done	Project report		7	Moving content from duplicate report to new report, creating table that links
Team meetings		1	Short summary of work that has been done and plans	Microsoft Entra		6	Trying to implement auth for machine digital identity as well and showing it to
Team meetings		1	Short summary of work that has been done and plans	Team meetings		3	Standup meeting
Project report		2	Customizing layout	Project report		2	Updating threat model
Team meetings		1		Microsoft Entra		1	Preparing CA to ask Stan about at next meeting
Team meetings with supervisor		1		AWS		3	Updating github readme to match new setup, updating cloudformation
Self-education		2	Read License Management in Closed Offline Networks Using Modern	Team meeting with stakeholder		1	Meeting with NBIM, got feedback on threat, PoC, effect goals etc.
				Team meetings with supervisor		1	Meeting with Li
				Team meetings		5	Went over report, evaluated new work, resolved comments, delegated work
Week 6		30		Week 16		32	
Project report	Category	Duration (hours)	Work done	Project report	Category	Duration (hours)	Work done
Self-education		4	Writing about tokens	Project report		2	Creating tables, working on tasks in kanban board
Information search		8	Looking for relevant ISO and NIST standards, Read License Management in	Team meetings		4	Standup meeting
Team meetings		1	Went over what has been done and made plans for what to be done	Project report		10	Going through scenarios and mitigations making sure they match
Project report		3	Fixing Latex layout and trying to find memory problem	Team meeting with stakeholder		1	Feedback on report
Team meeting with stakeholder		1	Got answers to some questions	Project report		2	Going through what has been done reviewing and commenting on it.
Self-education		3	Looking over AWS Well-Architected	Project report		1	Working on comments from NBIM
Project report	Quality assurance	2	Going over and commenting what has been done, also a bit of writing on	Team meetings		4	Doing the finishing touches in the report, reviewing tasks in kanban board
Team meetings		2	Went over report together	Team meetings with supervisor		1	
Team meetings with supervisor		1	Feedback on project structure	Project report		1	Presenting report and task to third party reviewer
Week 7		30		Week 17		31	
Project report	Category	Duration (hours)	Work done	Project report	Category	Duration (hours)	Work done
Team meetings	Quality assurance	3	Went over and commented on what has been done over the weekend and	Project report		3	Discussing comments with third party reviewer
Self-education		16	Looking for relevant work and reading about it	Project report		19	Working on feedback looking at what needs to be delivered
Project report		1	Writing API theory in API	Project report		1	Adding meeting minutes to appendices
Team meeting with stakeholder		1	Asked a few questions and discussed some group work with the group later	Team meetings		4	Standup meeting
Self-education		7	AWS Well-Architected	Team meeting with stakeholder		1	Stakeholders gives overall feedback on the report
Team meetings		1	Went over what has been done and made plans for next week	Microsoft Entra		1	Terraform
				Team meetings with supervisor		2	Receiving feedback on report
Week 8		30		Week 18		31	
Project report	Category	Duration (hours)	Work done	Self-education	Category	Duration (hours)	Work done
Self-education		4	Went over threat model, commented on it, fixed some small errors and	AWS		5	Checking everything works as intended and improving guide
Team meetings		2	AWS well-architected	Project report		3	Going over what has been done and giving feedback
Self-education		2	Standup meeting and went over threat model with group.	Project report		8	Working on feedback
Self-education		1	Article about passkeys from Stan	Team meetings		14	Going over the report
Self-education		1	CCSA prep kit and article about ABAC from Stan	Project report		3	Discussing threat model and conclusion with third party reviewer
Self-education		7	Searching for and reading best practices, reading vips link, reading main part	Team meetings		2	Standup meeting
Lectures		1	Course in report writing	Project report		5	Reading through whole report
Team meetings		2	Working on the layout for the project report	Team meetings with supervisor		1	Extra meeting with Erjon
Project report		1	Improving token part based on Stans feedback	Team meeting with stakeholder		1	Last meeting with stakeholders
Project report		3	Converting to new Latex file				
Team meeting with stakeholder		1	Went over some questions				
Team meetings		1	Cleaned up report				
Team meetings		3	Went over what has been done and plans for what to be done				
Team meetings with supervisor		1	Asked for guidance on layout for main part of report				
Week 9		30		Week 19		42	
Project report	Category	Duration (hours)	Work done	Self-education	Category	Duration (hours)	Work done
Self-education		13	Reading fusionauth link and searching and reading relevant work for best			18	last day with team at uni, going through everything and delivering
Team meetings		1	Standup meeting, went through what has been done and plans till the next				
Project report		9	Started writing main part and commenting on it				
Team meetings		1	Standup meeting, went through what has been done and plans till the next				
Team meetings		3	Standup meeting, went through what has been done and plans till the next				
Team meetings		3	Went over what everyone has done and comments from Stan				
Team meetings		1	Standup meeting, went through what has been done and plans till the next				
Team meetings with supervisor		1	Asked some questions about structure of report				
Week 10		31		Week 20		18	

D.3 Timetable - Arvid

Activity	Category	Duration (hours)	Work Done	Activity	Category	Duration (hours)	Work Done
Information search		3	AWS well-architected REST api Oauth & Oidc OWASP_WAF. master	Project report		2	Working on feedback from André fixed flow and postman picture
Information search		2	Oracle sin Oauth and OpenID Best Practices looked into	Team meetings		14	going over report
Team meetings		6	Daily Standup meetings	Project report		3	Fixed all pictures in PoC made new figure for introduction endpoints
Lectures		1	Lecture with frode about writing a solid raport	Project report		4	Write intro text of design choices, implementation, Designchoices,
Project report		3	Read through the links supplied by Stan, wrote IAM section	Project report		3	continue work on report
Project report		2	Making Image for basic auth Reading GDPR, datatilsynet	Team meetings		2	Standup
Project report		2	Fixed figure of basic auth, I wrote minutes and had meeting with NBIM			3	continue work on report
Information search		2	Vipps flows Azure b2c webhooks			5	Continue work on comments, PoC, and work with with Patrik
Information search		2	GA gjennom NBIM hoveddel og finn andre relevante best practice				
Team meetings with supervisor		3	Teammeeting into meeting with supervisors into internal meeting				
Self-education		1	Read: https://fusionauth.io/articles/oauth/modern-guide-to-				
Self-education		1	Read API keys for authentication Best practices for securing API keys				
Project report		1	Started writing API keys part for rapport				
Week 9		30		Week 19		36	
Activity	Category	Duration (hours)	Work Done	Activity	Category	Duration (hours)	Work Done
Project report		2	Finished writing key based authentication for theory read: what are			18	last day with team at uni. going though everythin and delivering
Information search		2	Reading best practices for API key usage, security concerns from Google				
Team meetings		7	Daily standups, Working together, going through the main and theat-				
Project report		2	NBIM 2023, trying to understand how the thesis is split Oidc notes for				
Team meetings		3	NIST 800-63C Digital Identity Guidelines Federation OpenID Connect				
Information search		2	Oidc native sso/SPA flows by Okta user interaction flows by Okta				
Information search		2	Oidc section in main				
Information search		2	Read up on best practice of API keys from NIST 800-204, Google Cloud,				
Project report		1	Wrote Api Keys with cites to best practices from Google/nist/aws				
Team meetings with supervisor		1	Feedback on thesis structure				
Illness		3	Illness @				
Project report		2	General figure best prac chap 3: auth / autntz				
Week 10		30		Week 20		18	

D.4 Timetable - Farhad

Timesheet				Farhad Mangal			
Activity	Category	Duration (hours)	Work Done	Activity	Category	Duration (hours)	Work Done
Team meetings		3	Discussed project plan and started writing it	Information search		4	Trying to find info for bow tie
Project plan		2	Continued working with the project plan. Filled out group rules and	Team meetings		1	Standup. Discussed weekend work and this week work
Team meeting with stakeholder		1	First meeting where we discussed on criterias and expectations	Team meetings		1	Standup
Team meetings with supervisor		1	Discussed our project plan and got feedback. We also asked question	Team meeting with stakeholder		1	
Self-education		2	Reading previous bachelor thesis'	Project report		3	Fixing comments on threat model and improving
Lectures		2	Lecture on tip to how to deliver a good bachelor project	Self-education		2	reading parts of project report that I had less knowledge on
Team meetings		2	We continued discussing the project plan with the feedback from	Self-education		3	reading on Federeation and doc on best practice (from NBIM)
Project plan		2	Wrote group roles	Self-education		5	read project report
Team meetings		2	Discussed what each member had done, discussed uncertainties.	Team meetings		3.5	
Project plan		2	Made some improvement to group roles and small fixes other places on	Team meetings with supervisor		1	
Self-education		3	Read about API getaway	Project plan		4	fixing feedback and bow tie
Self-education		2	Read about Fido Alliance framework				Look more into risk matrix and see if its good to include
Self-education		3	Reading on Oauth, Open ID, Scrumban				write in main part
Project plan		1	Wrote about status meetings				
Week 1		30		Week 11		28.5	
Activity	Category	Duration (hours)	Work Done	Activity	Category	Duration (hours)	Work Done
Team meetings		1	Standup meeting + planning weekly activities	Team meetings		2	standup meeting. discussed main report structure
Project plan		2	Worked with risk matrix in Lantex	Project report		3	write introduction for owasp
Team meetings		4		Project report		4	Worked with DREAD model
Self-education		4	Read about OAUTH 2.0, OIDC	Team meetings		1	Standup. Discussed DREAD model
Self-education		6	Read and watched lectures on API University	Information search		1	Read earlier reports + book on DREAD model
Project plan		1	Worked with fixing risk matrix syntax issues on latex	Project report		3	Dread model: Redid the score and write text again
Team meeting with stakeholder		2	2nd meeting with NBIM. Went through question with had.	Team meetings		1	standup meeting. Discussed progress and plans.
Team meetings		1	AWS setups	Project report		3	Finished Dread, started with risk matrix and also started writing intro
Project plan		1	Wrote project scope	Team meetings		1	standup meeting
Team meetings with supervisor		1	went over some questions in regards to project plan	Project report		5	Risk matrix
Team meetings		1	Planned before meeting with supervisors and fixed some of the issues				
Self-education		5	Finish 2nd part of API University				
Project plan		1	Fixing some issues on project plan				
Week 2		30		Week 12		24	
Activity	Category	Duration (hours)	Work Done	Activity	Category	Duration (hours)	Work Done
Self-education		4	Read about SAML, OIDC, Oauth 2.0	Team meetings		1	Standup. Read through document and fixing errors
Project plan		3	Wrote about SAML, fixed other part of text + quality reading project plan	Project report		5	Wrote new scenarios from OWASP.
Team meetings		1	standup meeting. Task given for new scrum period	Team meetings		1	
Self-education		4	read about Microsoft E	Project report		2	Working with DREAD
Team meeting with stakeholder		1	Demo of how the proof-of-concept should look.	Team meeting with stakeholder		1	
Team meetings		1	Review of project plan and kanban tasks	Team meetings		1	
Project plan		1		Project report		6	Finishing threat model, writing about Conditional Access and WAF
Self-education		5	watched videos on different standards, AWS WAF.	Team meetings		1	Standup meeting
Team meetings with supervisor		1	Discussed last part of project plan.	Team meetings		12	Finishing first draft of main report.
Team meetings		3	Worked with WAF and threat model. Fixed project plan feedback	Team meetings with supervisor		1	
Information search		4	Researching on threat planning and started writing notes to it				
Self-education		2	Read and saw videos about AWS Lambda and WAF				
Week 3		30		Week 13		31	
Activity	Category	Duration (hours)	Work Done	Activity	Category	Duration (hours)	Work Done
Self-education		2	Reading on configuration of threat modeling	Team meetings		4	Standup meeting
Team meetings		1	Reading and fixing minor issues on project plan	Project report		3	Risk Matrix, reading on PoC
Project plan		4	Reading book on threat model and creation of API	Self-education		3	Reading on sustainability and AI on google and previous bachelor
Self-education		1	standup meeting	Project report		3	Writing on sustainability
Team meetings		4	Working on threat model drawing of components, watching youtube on	Project report		3	Writing on bowtie text
Self-education		1	read through final version of final report + submitting it.	Information search		5	Reading on how to reduce consequence after attacks takes place
Project plan		2	Working with APIv2	Project report		5	Continuing to work with bowtie model
Team meetings		3	reading on API security and making notes for report				
Self-education		2	writing on threat model chapter				
Project report		6	threat model reading and writing				
Week 4		38		Week 14		36	
Activity	Category	Antall timer	Work Done	Activity	Category	Duration (hours)	Work Done
Team meetings		1	Progress meeting	Team meetings		4	standup meetings
Project report		6	Writing threat model analysis	Project report		3	reading through threat model for improvement
Self-education		7	Reading API Security	Team meeting with stakeholder		1	Feedback on first draft
Information search		7	Reading on mitigations for the threats in threat modelling	Information search		3	Read earlier reports on bowtie model
Project report		3	Making a more "advanced" /better threat model drawing	Team meetings with supervisor		1	Feedback on first draft
				Self-education		1	Was presented how PoC works
				Project report		12	Bow tie
Week 5		24		Week 15		25	
Activity	Category	Duration (hours)	Work Done	Activity	Category	Duration (hours)	Work Done
Information search		7	Read about mitigation on difference threads	Project report		3	Bowtie model text. Information disclosure and DoS.
Team meetings		2	Standup	Team meetings		4	Standup meeting
Team meetings		1	Group meeting.	Project report		6	Bowtie model: DoS and EoP
Team meetings		1	update meeting	Project report		3	Bowtie: EoP text
Project report		3	Wrote on top 10 owasp and continued with threat model	Project report		5	Risk matrix After mitigation
Self-education		4	Read API advance book	Project report		4	Discussion part for both Risk matrix
Team meetings with supervisor		1		Project report		3	Rewrite introduction to threat
Team meetings		1		Team meeting with stakeholder		1	Status meeting.
Project report		2		Team meetings with supervisor		1	Status meeting.
Information search		4					
Week 6		26		Week 16		30	
Activity	Category	Duration (hours)	Work Done	Activity	Category	Duration (hours)	Work Done
Self-education		5	Api security	Project report		14	Add more text to risk matrix, bowtie, risk appetite
Team meetings		1	Standup meeting	Team meetings		4	Standup meeting.
Information search		5	watched youtube videos, read different reports etc on api security	Team meeting with stakeholder		1	Got a lot of feedback on the report from the stakeholders
Self-education		4	Reading further on mitigation.	Team meetings with supervisor		1	
Team meeting with stakeholder		1	Showed threat model and asked question regarding best practice	Project report		10	Fixing small errors on threat model
Project report		2					
Self-education		3	Reading on Oauth 2.0				
Team meetings		1					
Team meeting with stakeholder		1	Asked question on				
Team meetings with supervisor		2	Went through eachothers text in the theory section and gave feedback.				
Team meetings		4	Reading more on API security				
Information search							
Week 7		29		Week 17		30	
Activity	Category	Duration (hours)	Work done	Activity	Category	Duration (hours)	Work Done
Team meetings		1	Standup meeting	Team meetings		4	Standup meeting
Information search		6	Reading on DREAD and CIA, and trying to figure out how to implement it	Team meetings with supervisor		2	Receiving feedback on report
Project report		1	Writing on overleaf	Team meeting with stakeholder		1	Status meeting.
Team meeting with stakeholder		1	Standup meeting with NBIM.	Project report		7	Going through rapport with team at school
Self-education		2	figuring out Figma and attempt to make a diagram	Project report		5	Working with new bowtie models
Project report		3	Writing on CIA triad and (almost?) completing mitigation section	Project report		7	Working with new risk matrix, fixing risk matrix text
Information search		4	Read about JWT, OAuth 2.0 and how it works in between Microsoft Entra	Project report		7	Reading through comments and making improvements in threat model
Team meetings		1	Progress report.				
Project report		5	Complete mitigation section				
Project report		1	Transfer everything to overleaf				
Project report		6	read over threat model. Trying to read further on how to implement				
Week 8		31		Week 18		33	

Activity	Category	Duration (hours)	Work done	Activity	Category	Duration (hours)	Work Done
Team meetings		3	Standup meetings x2	Team meetings		2	Standup meeting
Information search		2	Reading on CIA and trying to implement it on the rest of mitigations	Team meeting with stakeholder		1	Last meeting with stakeholders
Self-education		2	Reading about bow-tie and if I can implement it in threat model	Team meetings with supervisor		1	Extra meeting with Egon
Project plan		2	Fixing feedback from standup meeting in threat model	Project report		8	Rewriting new bowtie model text
Team meetings		2	Made table of contents for the project	Project report		10	Making introduction illustration, fixing feedbacks given by group and
				Project report		14	Going over report
Project plan		1	Make new diagrams according to discussed changes	Project plan		12	Fixing feedback on reports, bowtie, discussions, new dread model
Self-education		2	Reading on Digital Identities				
Team meeting with stakeholder		1	Questions about supervisor feedback				
Team meetings		1	Went through comments given by stakeholders				
Information search		4	reading further on cia triad and mitigation				
Team meetings with supervisor		3	Meeting with supervisor + Had internal meeting.				
Self-education		4	Vipps Flows, Digital identities, reading a few other bachelor thesis'				
Self-education		4	Read more on mitigation and CIA				
Week 9		31		Week 19		48	
Activity	Category	Duration (hours)	Work done	Activity	Category	Duration (hours)	Work Done
Self-education		2	Reading on Bow tie and risk matrix, and see if its possible to include			18	last day with team at uni. going though everythin and delivering
Team meetings		6	Meeting 1: Standup, Weekend report Meeting 2: Standup, Review of				
Project plan		5	Working with diagrams to Bow Tie model				
Information search		4	bowtie diagrams				
Information search		1	Read to former reports that contained bowtie model				
Project report		3	Fix given feedback from NBIM				
Project report		1	Write in main part: Idp				
Project report		6	Working with Bowtie...				
Team meetings		1	Prepare for meeting with supervisor				
Team meetings with supervisor		1	Showed overleaf report and got feedback				
Week 10		30		Week 20		18	

D.5 Timetable - Patrik

Activity	Category	Duration (hours)	Work Done	Activity	Category	Duration (hours)	Work Done
Team meetings		6	Team meetings	Team meetings		14	Going over report.
Self-education		1	Link fra Stian om passkeys.	Project report		7	Editing report based on comments.
Information search		1	Link fra Stian om ABAC + en ikke så bra lecture om API.	Team meetings		2	Standup.
Self-education		1	Securing OAuth and OpenID Connect Front-Channel.	Project report		6	Reading report.
Team meetings		2	Planned project structure	Team meeting with stakeholder		1	Final meeting with stakeholders.
Lectures		1	Lecture about writing a good report.	Project report		2	Editing report after team meeting.
Team meeting with stakeholder		1	Questions about supervisor feedback.	Project report		6	Reworking conclusion.
Information search		1	Vipps link from Stian + webhooks.				
Self-education		6	Digital Identities + NBIM 2023 bachelor +				
Information search		2	Postman and Okta articles about using digital identities.				
Team meetings with supervisor		1	Asked about report chapter layout.				
Self-education		4	Use cases for OAuth 2.0, OIDC, SAML 2 and FIDO fra fusionauth og				
Project report		3	Moved and edited parts from theory to main, as well as adding more.				
Week 9		30		Week 19		38	
Activity	Category	Duration (hours)	Work Done	Activity	Category	Duration (hours)	Work Done
Team meetings		4	Standup meetings.			15	Last day with team at uni, going through everything and delivering
Project report		2	Pros and cons for several technologies.				
Information search		1	Backing up claims made in pros and cons lists with secondary sources.				
Information search		1	Read up on access control best practices.				
Project report		2	RBAC and fixes to OAuth.				
Illness		3					
Information search		3	ABAC best practice.				
Team meetings		2	Going through the main part of the report and threat model together.				
Team meetings with supervisor		1	Presented report structure and got feedback				
Project report		1	Fixed commented on sections and sources.				
Project report		3	ABAC information search and writing.				
Information search		3	SAML research: https://datatracker.iETF.org/doc/html/rfc7522				
Project report		2	Writing SAML.				
Week 10		30		Week 20		18	

E Meeting Minutes From Meetings With Stakeholders

E.1 17.01.2024

Present: André Moen, Arvid Moemeni, Celina Heimdal, Farhad Mangal, Patrik Olausson and Stian Hagbø.

Absent: None

Do the product owners want access to our AWS organization/ OverLeaf/ GitHub Repo?

AWS: That's up to the group to decide, if we'd like to get some feedback we can send an invitation to Stian.

Overleaf: Yes, send us an invite to Overleaf so we can review while the team is working on the final rapport.

Git: Doesn't matter, if we want we can send them an invitation, but the repo has to be public anyways.

Feedback from Stian:

The group should make a TestDev account on AWS so the entire group can see the resources (API, lambda functions).

Theoretical part:

It's unlikely that the group will be able to make multiple best practice within the time limit, and also the cost will be significant. Its therefore better to focus on the general basis of securing API's and listing up the pros and cons and cost.

Read up on authentication flow in API standards. Make a plan for the rest of the semester.

How should the API(s) look when we're finished ?

You can make one single API with different paths, but take in consideration how this will affect scalability. Look into how "easy" it will be to make different paths for this single API. The important part for us in NBIM is authentication, authorization

and general security for the API. Also the different paths on the API should have different levels of access, to show us that you are able to make it differentiate user access.

E.2 24.01.2024

Present: André Moen, Arvid Moemeni, Celina Heimdal, Farhad Mangal, Patrik Olaussen and Stian Hagbø.

Absent: None

Started with some questions on how the project plan is coming along. Showed project plan to NBIM.

Stian then started the demonstration of their api workflow

Some notes that they mentioned we should take into consideration:

aws api gateway -> sending request to lambda.

Aws api gets checked by AWS WAF, which is a firewall that we can add rules to. Example shown by Stian was allowing only certain ip-address sources.

Use Postman for testing API

Use api key in V1 to authenticate, its long lived and not really secure.

Use these regions: Regional eu-west-1 (irland server)

Eu-north-1 can also work if we want (stockholm)

Stian recommends us to follow aws best practice and write everything in code.

Include a whitelist which allows certain ip addresses (source)

ACL is outside the firewall, and you can add rules like: IP reputation list, as in ips cant be coming from known (marked) ip-address ranges.

Read up on common rule set: search for aws baseline rule group.

Stian also mentioned:

V1 api is going to be a REST /http api

Authentication on V1 will be hard, unless the group wants to make one in javascript.

Lambda in v1 are only definitions on different authorization responses, eg. To be able to show fine-grained authorization.

V2: postman -> Cloudfront -> amazon api gateway

Cloudfront will be able to send the traffic to different regions base don request geolocation.

In v2 api gateway sends requests to Entra ID (IdP) for authorization access token. Since v2 doesn't allow AWS WAF they add this part has to be done on CloudFront. The rest of the demonstration shows how to use Entra ID and add users into groups, roles, mark them with roles.

E.3 07.02.2024

Present: André Moen, Arvid Moemeni, Celina Heimdal, Patrik Olaussen and Stian Hagbø.

Absent: Farhad Mangal

André showed the yaml file for API v1, which has different endpoints, lambda functions, API resources and WAF rule for accepted IP ranges for API connection.

Stian commented what the group should include in the WAF. Should include the 10 rules that AWS recommends on WAF.

André had already included IP range restrictions so only people could use the API from NTNU IP range.

Things to include: SQL injection rules, bot rules, blacklist of known bad IP's. And the 6 other best practice rules from AWS.

Comments about API v1 and V2

APIv1 has to use lambda functions for authentication, Stian suggested us to search for this code online and use this as a base.

APIv2 has this functionality built in.

E.4 28.02.2024

Present: André Moen, Arvid Moemeni, Celina Heimdal, Farhad Mangal, Patrik Olaussen and Stian Hagbø.

Absent: None

Stian asked if we have looked into OWASP threat modelling. - Farhad had included this in our threat.

Patrik asked, how in depth do we have to explain the different protocols that we are gonna use, like OAuth or SAML. Stian answered that we must explain them briefly, explain what it does and how it is used.

Then we just updated on what we have written this week, and Stian suggested that if we add too much now in the theory part then it's fine, we can cut it out later.

Then Farhad asked about the threat model, CIA and DREAD model. Stian answered that we can use both, we can also take the judgement ourselves.

E.5 06.03.2024

Present: André Moen, Arvid Moemeni, Celina Heimdal, Farhad Mangal, Patrik Olaussen and Stian Hagbø.

Absent: None

Starting the meeting with patrik asking how we are supposed to write the thesis without going in depth about the different technologies in api security.

Answer: they disagree on what the supervisors told us, they believe we should write all the technologies in the theory part, since the reader needs to understand this before we start telling our own opinions / start reflecting.

Stian and Celine suggested that we should start writing how different flows and technologies we would use in different situations based on identities. User, service user, server-to-server, admin, osv.

Stian will send us vippis recommended flows, the group should look into this

Stian and celine tells us that we need a part that goes into the best practices.

They also recommend us to write everything we feel fits in the thesis now, then cut down and remove unrelevant parts later.

The group should write the thesis in a general sense, not spesific to NBIM and their requirements for authentication.

E.6 20.03.2024

Present: André Moen, Arvid Moemeni, Farhad Mangal, Patrik Olaussen and Stian Hagbø.

Absent: Celina Heimdal

How deep should we go into each subject?

Chose some and prioritize them, base it on a scenario. Remember to explain why we didn't go into depth on other technologies. Chose the ones that are considered more secure, and only mention the ones that are further away from best practice. Mention that it exists and could be used, but don't go in depth about it.

Should we come up with something new ourselves?

No, follow standards that exist and explain what they do.

For the protocols and solutions, we talk about, we should not talk about the specifics of how they work, rather what problem they solve.

For the next time we should talk with supervisors and make it clearer what they expect from us, remember to send mail after meeting with them to stakeholders.

E.7 03.04.2024

Present: André Moen, Arvid Moemeni, Celina Heimdal, Farhad Mangal and Stian Hagbø.

Absent: Patrik Olaussen

Stian: Proof of concept is not that important to be complex. Try to make it simple.

We will send a mail to the stakeholders on friday when the first draft is done.

E.8 24.04.2024

Present: André Moen, Arvid Moemeni, Farhad Mangal, Patrik Olaussen and Stian Hagbø.

Absent: Celina Heimdal

Feedback from stian about chap 7 that Andre is writing about.

He is going to read more about it after the meeting

André: machine machine is now included in the PoC.

Stian suggested using lambda function for authorization in the Cloudfront so that the request gets checked as early as possible. We cant do this with JWT authorizers, but it is important that we describe this decision in the document.

Stian also wanted us to look into desentralised identifiers. We are basing our implementation on a single centralized provider (Entra). This should be included in the background in threat model.

This should say something about the users and stakeholders that are a part of the centralized authority.

Stian also wanted us to include system to system, user to system, system to machine identities. Since system to machine is not a part of our scope it should be discussed in the discussion section.

Stian also mentioned that our effect goal should include identity, he suggests having the first point replaced with identity.

Last comment: fix authentication in theory about API keys. He says its weird that its only mentioned API keys as a authentication method. True.

Lastly we showed the data flow which he thinks looks good. And we got feedback on threat model which he thinks is good.

E.9 30.04.2024

Present: André Moen, Arvid Moemeni, Celina Heimdal, Farhad Mangal, Patrik Olaussen and Stian Hagbø.

Absent: None

Stian and celina have given a lot of comments in our overleaf.

Celina: written a lot of comments in the documents and is reading through currently.

She wants to make clear that all comments are suggestions and that she is going to read through the document this week.

General comment by Celina: Overall very good, but some of the texts gets repeated in different sections.

Comments relating to adding figures in theory.

Stian: as last time the Api keys and RBA are alone in authentication. API keys should not be alone in the theory part. Feels off.

3.3.2 small issues with Entra ID ID being typed because glossary also includes this. Already fixed.

3.6.5 unclear how the scoring system works. Some parts 1 is the lowest, other parts, it's the highest.

3.5.7 unclear what is low and high score.

Table 3.11 risk level, where do these numbers come from?

Add the source, the levels are really high; if it's from somewhere, it's ok. But then it needs to be cited.

Figur 5.3 remove plaintext and either censor or use this guide: <https://learning.postman.com/docs/sending-requests/variables/variables/>

Table 5.1 needs explanation about machine to machine communication. Right now the stakeholders thought it also was a group. But its not. Its only an application.

In the implementation there should be more subsections. Right now there is just a block of text. Make it more visually pleasing by cleaning up and sectioning the different parts.

Discussion part: common thread instead of red thread.

6.6 needs cite's especially about decentralising identity management.

Further work needs to be worked a bit more on, focus on giving recommendations. We don't need to implement but it does show how we would continue working on the project.

Farhad asked if it's okay to implement different theories into the threat model. They both think that's fine.

Farhad asked about risk appetite. Is that something that they are interested in?

NBIM has a low risk appetite. Other organisations might have a bigger risk appetite.

CIA triad is included in the threat, but it's not really used throughout the model. We can either drop it or add the different components into the threat model.

If it doesn't fit then it doesn't fit. Api should follow the CIA triad. Should mentioned

Bowtie: grammar errors in privilege escalation

3.14 lack of numbers in table.

All in all the supervisors are happy with the report so far. A lot of small fixes needs to be implemented but overall its good.

E.10 08.05.2024

Present: André Moen, Arvid Moemeni, Patrik Olaussen and Stian Hagbø.

Absent: Farhad Mangal and Celina Heimdal

The meeting starts with the group thanking NBIM for the feedback.

Patrik summarises the last week's work.

Stian gives overall feedback: the work is well done, and he thinks that it's nice that we have a logical structure in the report. The only thing they want us to fix is the comments given in Overleaf.

Some small issues he wants us to look at:

Bowtie grammar issues. Figure 2.2 Authorisation written with Z.

Andre asked who is gonna be written as our taskgiver.

Answer: Stian and Celina

Last meeting is next week.

E.11 16.05.2024

Present: André Moen, Arvid Moemeni, Patrik Olaussen, Stian Hagbø, Farhad Mangal and Celina Heimdal

Absent: None

Patrik introduces the new title for the project thesis. - Stian suggests sending the title to supervisors to check what they think.

Patrik invites NBIM to join us at the presentation either online or in person. - The group will send them a link when this gets shared by NTNU

Farhad shows a figure summary showing threat model steps. Talked to the supervisors from NTNU about this, and they gave the feedback that we should make a copy from ISO. Farad's question is, what do you think about the model, is this something we should change?

- Celina answers: both work as long as we argument for why we have chosen the model and why it. - Should explain in text that we are inspired from ISO if we choose to use one from them.

- Stian: the model should probably show how the rest of the text is built up, so it shows the read thread throughout the rest of the threat model chapter.

Patrik asks about comments given by Celina about moving research questions to discussion instead of conclusion. She argues that research questions is something that the group should discuss and therefore it should be included in this chapter.

-Do as you want just a recommendation, and also if the supervisors at NTNU haven't given any bad feedback on it, we can keep it.

Patrik thanks NBIM supervisors for their help throughout the project.

Stian - God job throughtout the semester.

F Meeting Minutes From Meetings With Supervisors

F.1 11.01.2024

Present: André Moen, Arvid Moemeni, Erjon Zoto, Farhad Mangal, Guoqiang Li and Patrik Olaussen.

Absent: None

Received feedback on the work done on the project plan this far.

Presented the task to the supervisors and received some initial thoughts about the task.

Decided on weekly meetings with supervisors every Friday at 11:00 at Gjøvik campus.

Guoqiang Li suggested investigating scrumban as our work method throughout the project.

Fido was also suggested as a framework that could be used to validate our work.

F.2 19.01.2024

Present: André Moen, Arvid Moemeni, Erjon Zoto, Farhad Mangal, Guoqiang Li and Patrik Olaussen.

Absent: None

Should we merge problem area with problem description?

No. They should each be individual parts where problem area is a broader description of the problem. More in terms of preserving cybersecurity principles, why is lack of cybersecurity a problem. Should be deeper the further down.

Write about tools, technologies and standards in point 5 instead of Frameworks.

Project area: Cybersecurity in organizations, confidentiality, integrity and availability.

Limitations: API security.

Description: Task given by NBIM, more about API security.

In frameworks we need to talk about reasons for choosing a framework, advantages, features. If they have the same purpose, they should be compared up against each other. For example, explain API gateway.

FIDO, ISO, Oatuth, OWASP osv should be placed under standards in point 5.

Does any of the supervisors have knowledge of APIs, if not do they know anyone else who could have some more knowledge?

The supervisors do not have knowledge of APIs. We should hear with IDI.

F.3 26.01.2024

Present: André Moen, Arvid Moemeni, Erjon Zoto, Farhad Mangal, Guoqiang Li and Patrik Olausson.

Absent: None

Do we include SAML and FIDO?

Yes, we should include it as long as we're going to use it. Less work to do later.

Do we include OWASP top 10 and CCSK and where in the project plan should it be?

Put OWASP above provided learning material under documentation.

Documentation should be what tools we use for the report for documentation. How we document what we are working on. Part of quality assurance is to document what we're doing during the project period. Latex, Teams, Timesheet, meeting notes, how to organize project.

Mitigation for over budget?

Up to us to decide how to manage what happens if we go over budget. Mention under risk scenarios. Discuss with NBIM about what happens if we go over budget. Measure: Alert when we go over budget.

How should we conduct threat model?

DREAD, mention some now and then go through more the next weeks.

What do we want to write about the threat, what information do we need.

STRIDE, types of attack, may not reflect to APIs. Should understand the security issue before we design the API. Detail the security problem.

Threat model API gateway, API security. Combine search with threat model, API and Authentication and authorization.

Start the project by establishing what the threats are and find out what the requirements for authentication and authorization are.

Find out about what security measures that are in AWS.

F.4 02.02.2024

Present: André Moen, Arvid Moemeni, Erjon Zoto, Guoqiang Li and Patrik Olausen.

Absent: Farhad Mangal

How should we proceed after project plan?

Look at previous reports. Main activity should not be writing the report, but the preliminary work required to write the report. Follow the gantt plan, just work on what's on it.

Threat model important to outline the needs for the APIs security. Understand all the security issues required for the API.

Talk about the sources we're using to find information, like the book API security in action.

F.5 16.02.2024

Present: André Moen, Arvid Moemeni, Erjon Zoto, Farhad Mangal, Guoqiang Li and Patrik Olausen.

Absent: None

Should the threats be specific for our case or more general?

Too general wouldn't reflect our case, could be part of the background, theory, general view.

Should put things that are specific to our work. Related work should be put in the theory part.

Small section where we mention not as relevant threats. Summarized section about other threats and elements that aren't highly prioritized or relevant.

Have to split it up between theory and different chapters. Threat modelling should be in the beginning.

How to structure the main report?

The people reading it might not know about the content, should therefore start with introducing the task description. Something about related work, what has been done before, our work should be more about the local environment of NBIM and new technologies that haven't been discussed to much yet. Then theory about

relevant technologies. Start with what is written in the project plan and expand upon it.

Write what we want and when the template comes, we should place the different parts as we want. As long as we have sufficient content it can be restructured at any time.

How to define best practice?

Look at the standards, which recommends something and then later look at what the big users use.

Look at NIST and ISO standards and compare to what they say about API security. Look for European level standards. Might be some Norwegian standards.

Look at what is used in the industry, GARTNER, shows who is the top players in different fields.

E.6 23.02.2024

Present: André Moen, Arvid Moemeni, Erjon Zoto, Farhad Mangal, Guoqiang Li and Patrik Olaussen.

Absent: None

Old sources

Its fine to cite old sources. Should make sure it's the latest development even if its old.

What background should we expect the reader to have?

Should explain some of the main parts, not everyone in the computer science domain knows everything about our topic. Should have description about authorization server and such things, should be quite basic level. If we come with a term that the normal computer science student wouldn't know we should describe it.

Start with a general introduction, where we explain all the important terms that are used.

Good if we explain it like everything is read for the first time, shouldn't be only for the student but for anyone, it's a public report.

Give a short description of what the different attacks are in the threat model.

Where to place theory

Should have introduction before theory, shouldn't be a dedicated chapter for theory. Should be described and introduced when we use them in the report. Shouldn't give that much information about each term, only give a short description. Split

it and mention it when we come to a specific term. Should give a short paragraph and if the reader wants to understand more, they should find it themselves.

F.7 08.03.2024

Present: André Moen, Arvid Moemeni, Erjon Zoto, Farhad Mangal, Guoqiang Li and Patrik Olaussen.

Absent: None

How do we decide level of threat on the threat model, do we make it ourselves or base it on something?

We should try ourselves, and let Erjon take a look at it. Use our own experience and reasoning.

How should we structure best practice chapter?

Take recommendations from standards, write about them, talk about pros and cons from the different approaches. Best practice would be a little bit more practical. The structure that NBIM had last year is a good structure.

Should structure the report, especially the main part for the next time to get feedback on it.

F.8 15.03.2024

Present: André Moen, Arvid Moemeni, Erjon Zoto, Farhad Mangal, Guoqiang Li and Patrik Olaussen.

Absent: None

Feedback on project structure

More fair distribution of our chapters. The object with each chapter, the topic that is being discussed. Should look at IMRAD. Should have 6 chapters, if we have a lot of results we should add some more chapters for them. Section under theory about threat model theory. Should refer to cites in our figures as well, to not make it look like our own design. Threat model should be moved above implementation of best practice. We should make our own solution, not just repeat what has been done before.

Structure should be in a more high level approach, chapters shouldn't be as specific.

F.9 22.03.2024

Present: André Moen, Arvid Moemeni, Erjon Zoto, Farhad Mangal, Guoqiang Li and Patrik Olaussen.

Absent: None

Should we create something new or just recite best practice?

Should deliver what the task giver asks, to receive a good or average grade. As much as we're able to add up on the task given by the task giver. If we can and will do something further, like providing our own suggestion to the problem, it would be even better.

We should have existing work and related work. It is important for us to have our own idea/solution, existing work might have drawbacks and limitations that only work for certain problems. There isn't a perfect solution, combine several approaches to get our own. They need to do a lot of risk assessment to accept our own solution.

Combine different best practices to solve the issue. We design a better solution by combining multiple best practices.

What Erjon has been giving feedback on was mostly on our project plan, not the main part. Part of the work is to review what's already out there, then we have to provide suggestions for what best practices are. The third part is what we can add to the task. We don't need a big improvement a small step is fine, if we want a better-than-average grade, we should put some effort into coming up with our own takes on best practice. Talk about how some best practices drawbacks can be fixed/ strengthened by other best practices.

The customer might be happy with our task, but the final grading will be done by someone else, that might not be happy about our work. Could be bad writing, or bad formulation of parts of the thesis. The more we go beyond what we're asked, the better it is.

One of the goals of the thesis is to learn how to solve a problem, we always want to improve existing solutions and need to show some improvements, they might not use it, but might be valuable for some other readers.

Supervisors want an improved report they can look at for the next meeting.

We want to move threat model to the start

Supervisors agree on it, would make it easier to read the report and understand the problem.

Should we even have mitigations in the first part of the threat model?

Should separate it and have mitigations in the later parts.

F.10 05.04.2024

Present: André Moen, Arvid Moemeni, Erjon Zoto, Farhad Mangal, Guoqiang Li and Patrik Olaussen.

Absent: None

Feedback on report

Don't let readers confuse what exists and what our findings are, need to outline our findings and mitigations. Readers should be able to see what we have thought of ourselves. Find out the pros and cons of each, and combine them, and the combination is our solution as to how we do it. This is the methodology. Should look at several different best practices and combine them, that's our solution.

Talk about our contributions and future work. Readers look mostly at this part and want to see our reflections after we have done our work. Summarize our whole project. What are our contributions and what value did we add to it. The discussion and conclusion parts are where the value of our project lies.

F.11 19.04.2024

Present: André Moen, Arvid Moemeni, Erjon Zoto, Farhad Mangal, Guoqiang Li and Patrik Olaussen.

Absent: None

Feedback on first draft:

Li:

Section 1.1 Background should be changed to task description. It talks more about the task description.

Use the full names in the titles, not the abbreviations.

The ordering of subchapters in the proof of concept should be reorganized. Discussion – what we have done compared to what we could've done.

Target audience should highlight who the target audience is. Reorganize to add some research questions. Should have some research questions at the beginning of the report. Add a new subchapter 1.3.1 research questions.

The thesis doesn't have a method chapter, could help to add a method chapter, to perhaps deal with the structure a bit better. If it is challenging to add a method chapter it could be dropped, but it should help, a follow-up for the research questions.

Proof of concept should normally contain two parts, design (where we highlight why we chose the design we did) and then implementation.

Before we discuss the different types of tokens, we should give a short overview of the different tokens we'll be discussing.

We should describe authorization and authentication differences a bit earlier. Add a part about the differences a bit earlier.

Figures: Add citations to where the figures came from in the caption.

Chapter 3.7 DREAD, we mentioned DREAD several times before we explained what it was.

We should implement a risk matrix after the proof of concept that shows how the mitigations we've implemented have secured the API.

Chapter 4 Security mitigations: Should add a high-level table at the start that links all the mitigations to the different threats from the threat model.

Erjon:

Method chapter, not saying we should have it, but it can be helpful to have. It's a bit difficult to distinguish between what's our part and what's taken elsewhere. The method would help distinguish what has been read earlier and how we work with it. Make sure to distinguish the review between existing practices and our work. Some parts of chapters 3 and 4 are a bit theoretical.

Should talk a bit about the risk matrix, not just have it there.

Have a simple figure in main logical components that help visualise the components. If it's too overlapping with data flow the same picture can be used.

3.2 main logical components (Why is it located here) Should have an explanation of the components in the theory chapter. Have a threat model on those components.

We should try to link the different sections better together.

In chapter 2 it's a bit too many items under theory, should have higher level sections. Should put more of the items in lower sections. Sections including one paragraph should not be used.

There is nothing about what is out there, should include a part in theory about tools, solutions and papers that have worked on the subject earlier, but don't fit what the task giver asked.

What are we putting on our PoC: It would be nice to have some figures that show the design of the PoC. Show some use case diagrams. Argue about benefits.

Discussion about PoC should perhaps be in its own chapter.

Should take a look at previous reports that also have built something and take inspiration from them.

How does zero trust affect trust boundaries?

It's fine as it is, as zero-trust is implemented later. The trust boundaries are before the mitigations are implemented.

Is the structure of the threat model good?

The CIA triad explanation should go a further bit up.

F.12 19.04.2024

Present: André Moen, Arvid Moemeni, Farhad Mangal, Guoqiang Li and Patrik Olausson.

Absent: Erjon Zoto

Can we use both ISO and the book for the threat model?

Yes, its fine, important to highlight ISO standard as they're reputable.

Trust boundaries not included in the ISO pipeline for threat modelling, is it fine to add something that isn't in it

Yes, should mention that we combined them and why its better.

Is the data flow figure good, is it okay to have it specialized?

Yes, specialized is good. The figure is our own, which is an important contribution to the report.

Can we mention some of the mitigations early on in the threat model before the main part?

Yes, that just fine.

F.13 03.05.2024

Present: André Moen, Arvid Moemeni, Erjon Zoto, Farhad Mangal, Guoqiang Li and Patrik Olausson.

Absent: None

Literature study, is it good now?

It's good now as a part of the methodology. Everything we've done from the moment we started until the end.

Move educational courses, it does not belong in the methodology chapter, it should be put under the discussion chapter instead.

Should include the difference between the difference between the two the first time we mention them. Perhaps put it inside the authentication section. Erjon: might not be relevant to include the part with the difference between the two. Li:

would be nice to have it so it is understandable. Just include a small sentence of the start, to explain to readers the difference between the two.

F.14 10.05.2024

Present: André Moen, Erjon Zoto, Farhad Mangal, Guoqiang Li and Patrik Olaussen.

Absent: Arvid Moemeni

Feedback from Li:

Some changes can be made to make the bowtie models clearer. Some threats of the bowtie models are a bit unclear, they should be specified in the bowtie model.

In the figure the threats are showed, but they aren't explicitly mentioned in the text. The text should be the one that is more complete. The text doesn't need the figure to be understood.

Would be nice to have the guide on how to set up the PoC in the text instead of just referring to the github repo. Should have a summary version of it, dont want to go some where else to find the relevant info.

Should clear up that postman is just one tool that we're using in the PoC, can use many more ways to send the request.

Should explain what cloudformation is, as it has not been explained in the text.

In the PoC should describe what the different roles can do and why they're there, why are the developers there for example, shouldn't they be gone when the API is up and running?

In figure 5.4 change actor to developer. The github figure might be changed to something more general, like a database or datasource. Should also change Postman in the figure to something more general, dont have to use postman to send the request.

The machine in the table with entity and endpoints comes in a bit unclear. We're discussing we have three roles, and then suddenly a machine comes in.

Formating for the AWS WAF rules, no spacing between the words in titles.

Learning goals, need to mention why we abandoned scrumban.

Should try to write in a more positive way, need to write what we learned not what we didn't learn. Shows that we have something we didn't do.

Last goal on 6.1 should be rewritten. Don't give the readers a negative impression of the report.

6 – Alternatives, JWT auth needs to be specified to match AWS.

7.3 Should be rewritten.

The volume testing part is a bit hard to understand. Should use something other than volume testing as its related to something else.

7 – Conclusion. Should talk about how the main part of the report can also be used for de-centralized identity. Right now, it sounds like it only works for centralized identity and nothing else.

Feedback from Erjon:

Too many hyperlinks made it harder to comment, which could be problematic for the sensor.

Tend to put tables and images before text, should rather start with text.

Feel like there are a lot of tables, try to merge some of them or use another method to show the content. Not the biggest issue, if we have time, try to change them. A lot of tables only have two rows or columns which we should try to merge.

In the bowtie models, we need to make sure everything is readable. The reader shouldn't need to zoom in on the image to see it. Increase the font in images. In general, for the whole report.

Etag headers isn't described earlier, need to explain it, just describe it in one sentence.

4.7.2 need to rewrite paragraph, it seems like action points, not sentences.

Sometimes we have very short sentences, might want to extend them a bit or merge sentences together.

Add a summary section to the bowtie/risk matrix section. Feels like we're doing a big jump between chapters, seems to be something missing.

Might need summary for chapter 3 as well, should look at all the chapters and make sure there is an ending of it.

In the PoC specify what different roles have access to a bit earlier. Should add a few sentences about what we've thought about it, the description for it comes too late. The table which shows the endpoints might not be clear enough for that. Should add an overview of what the different roles should have access to do and what they could do.

Figure 5.1 not necessary.

Should make it clear postman is only one tool to get a token, can use many other applications or write it into code. Postman is just for illustrating how to gain access token. With a client application the end user doesn't have to do it manually.

The rules from AWS should specify that they're premade and we've chosen to use some of them. Should argue for why we choose the ones we did.

Some of the result goals looks more like effect goals. Take a look at it. Results are report, PoC, github.

In the use of AI should include some examples of how've used it, relate to different sections where it was used.

Dont need to focus that much on criticism of the thesis. Not relevant to include misunderstanding during meetings, data being stored on different platforms. Most of it is not relevant and should be summarized. Spent a lot of space on these issues. It's a whole page of text, but only 2-3 events. It makes it feel more severe than it really is.

In conclusion it's okay to use bulletpoints when there are many options, but when there are only a few it shouldnt be used but need to keep it consistent.

In the research question didn't mention best practice at all, but the taks description asked for best practice. We should add them to fix this.

The research questions are very good, but very connected to each other. Feels more like a main question and a sub question rather than question 1 and 2.

Title has nothing to do about security. Might need to change the title to make it include something about security.

Second question: Main treaths towards APIs and digital identities and split the answer into two.

First question: What is the current state of the art technologies tools when it comes to APIs and digital identities. Then later argue that we're focusing on this and that.

Dont need results to answer research questions, theory is enough to answer them.

The chapters structure is a bit different from the typical report. Summary section at the end of chapters could help fix this.



 **NTNU**

Norwegian University of
Science and Technology