TU WIEN Informatics

Untersuchung der Sicherheit von Cloud-Workloads

Entwicklung und Evaluierung von Mikroangriffen zur Bewertung der Bedrohungserkennung von Microsoft Azure

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering & Internet Computing

eingereicht von

Manuel Kuß, BSc
Matrikelnummer 11722667

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.-Prof. Dipl.-Ing. Mag. Dr.techn. Edgar Weippl

Wien, 8. August 2025

Manuel Kuß	Edgar Weippl

TU WIEN Informatics

Investigation of Cloud Workload Security

Development and Evaluation of Micro-Attacks to Assess the Threat Detection Capabilities of Microsoft Azure

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering & Internet Computing

by

Manuel Kuß, BSc
Registration Number 11722667

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.-Prof. Dipl.-Ing. Mag. Dr.techn. Edgar Weippl

Vienna, August 8, 2025

Manuel Kuß	Edgar Weippl

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Manuel Kuß, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.
Ich erkläre weiters, dass ich mich generativer KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Im Anhang „Übersicht verwendeter Hilfsmittel“ habe ich alle generativen KI-Tools gelistet, die verwendet wurden, und angegeben, wo und wie sie verwendet wurden. Für Textpassagen, die ohne substantielle Änderungen übernommen wurden, habe ich jeweils die von mir formulierten Eingaben (Prompts) und die verwendete IT-Anwendung mit ihrem Produktnamen und Versionsnummer/Datum angegeben.

Wien, 8. August 2025

Manuel Kuß

Acknowledgements

This thesis was conducted in collaboration with Austrian Power Grid AG (APG), the operator of Austria's transmission grid. I am very grateful for the opportunity to conduct this research project at APG. Especially, the funding of the cloud infrastructure for the security testing experiments was of fundamental importance and made this project possible. Insights from a company that is in the process of outsourcing workloads to the cloud provided valuable information for the setup of a cloud environment that follows best practices. The insights and recommendations from my colleagues at APG, especially in the Security Operations Center team and the Cloud team, were valuable for the project's success. In particular, I would like to thank Michael Egger, who assisted me with the project and had the role of a mentor. Additionally, I would like to thank Ivo Friedberg for his valuable feedback and recommendations during the research process, and Gottfried Bauer for his assistance in setting up the Azure DevOps pipeline and Terraform. Furthermore, I would like to thank my supervisor, Edgar Weippl, for the discussions, advice and comments on the project.

Kurzfassung

Cloud Computing ist eine vielversprechende Technologie, die zahlreiche Vorteile bietet, aber auch die Risikolandschaft in Bezug auf Sicherheit und Datenschutz verändert. In dieser Arbeit wird die Sicherheit von Cloud-Workloads mithilfe des Angriffs-Emulations-Tools Atomic Red Team (ART) untersucht. Konkret wird die Bedrohungserkennungsfähigkeit von Microsoft Azure mithilfe eines risikobasierten Sicherheitstestansatzes bewertet. Die Durchführung dieser Sicherheitstests auf einer Cloud-Infrastruktur, die den Best Practices entspricht, liefert Erkenntnisse bezüglich der Bedrohungserkennungssysteme von Azure und deren Optimierungsmöglichkeiten. Im Rahmen dieser Arbeit wird die Fähigkeit von ART bewertet, die drei relevantesten Bedrohungen abzudecken; ferner wird untersucht, wie neue Mikroangriffe entwickelt werden können, um die Fähigkeiten von ART in diesem Bereich zu verbessern.

Diese Arbeit trägt dazu bei, die Lücke in der Literatur zu Cloud-Sicherheitstests und zur Bewertung von Cloud-Intrusion Detection Systems (IDSs) zu schließen. Die Ergebnisse dieser Arbeit sind zum einen die entwickelten Methodiken: eine risikobasierte Methodik für Sicherheitstests und eine weitere zur Identifikation von Angriffsszenarien für die Umsetzung neuer Mikroangriffe. Zum anderen werden konkrete Erkenntnisse zur Sicherheit von Cloud-Workloads präsentiert. Dazu gehören die Bedrohungslandschaft, eine Rangliste der relevantesten Bedrohungen, eine Evaluierung von ART, die Erkennungsrate eines Best-Practices-Angriffserkennungssystems in Azure, sowie der Effekt, den das Einfügen neuer Mikroangriffe auf die Abdeckungsmetrik hat. Die Arbeit kommt zu dem Schluss, dass ART noch nicht in der Lage ist, genügend Angriffsszenarien für umfassende Sicherheitstests bereitzustellen. Durch vier neue Mikroangriffe wurde die Sammlung der Angriffe in ART erweitert, wodurch sich die Testabdeckung verbesserte.

Abstract

Cloud computing is a promising technology that offers numerous benefits, but it also alters the risk landscape in terms of security and privacy. This study investigates the security of cloud workloads using the adversary emulation tool Atomic Red Team (ART). Specifically, the threat detection capabilities of Microsoft Azure are evaluated using a risk-based security testing approach. By applying this methodology to a cloud infrastructure that follows best practices, the research provides insights into Azure's threat detection environment and identifies optimization possibilities. The thesis assesses the suitability of ART for addressing the top three threats and explores how new micro-attacks can be developed to enhance ART's capabilities in this regard.

The thesis contributes to filling a gap in the literature on cloud security testing and the evaluation of cloud Intrusion Detection Systems (IDSs). Its contribution lies in both the development of methodologies and its technical findings. One methodology serves for performing risk-based security testing, another for identifying suitable attack scenarios for the realization of new micro-attacks. The technical results regarding the security of cloud workloads include: the threat landscape, a threat ranking, an evaluation of ART, the threat detection rate in a best-practices threat detection setup in Azure, and the effect of adding new micro-attacks on the coverage metrics. The thesis concludes that ART is not yet capable of providing a sufficient number of attack scenarios for comprehensive security testing. To address this limitation, four new micro-attacks have been developed to extend the set of attacks within ART and to enhance the test coverage rate.

Contents

CHAPTER 1

Introduction

The cloud computing paradigm has evolved over the past few decades into an omnipresent technology sector that requires many organizations to consider using cloud services. Whether it is a management service, online meeting service, AI service, or the utilization of virtual infrastructure for computational tasks, cloud services offer a significant opportunity for easily deployable, scalable, and highly available solutions. A primary concern that comes to mind when considering the usage of cloud services is security and privacy, since data is outsourced to another physical location under the control of another company, probably in another country with different jurisdictions. Furthermore, as presented by Jansen and Grance [1], the system complexity, shared multi-tenant environment, internet-facing services, and loss of control are some fundamental concerns that need to be addressed in security assessments.

The key security principle that applies to cloud computing is the shared responsibility model, which defines the delineation of responsibilities between the Cloud Service Provider (CSP) and the Cloud Service Customer (CSC). Consequently, security is a joint effort between those two entities. Over the past few decades, CSPs have evolved and become increasingly stable and professional in addressing security issues. The shift in top threat categories in recent threat reports from "vintage" cloud security issues, which lie under the responsibility of the CSP, such as technological or architectural vulnerabilities, to issues that fall within the responsibility of the CSC, underlines this statement [2]. In 2022, Crowdstrike's [3] observations revealed a 95 % increase in the number of cloud environment intrusions compared to 2021. In 2023, a 75 % [4] increase was observed compared to 2022. These statistics, in combination with the decline in security issues that fall within the responsibility of the CSPs, raise concerns about the security measures on behalf of the CSCs.

Gartner [5] predicts that 90 % of organizations will utilize cloud resources by 2027. This highlights the significant number of organizations that must address security and privacy issues associated with cloud services. The question that arises in this regard is how

secure the usage of cloud services is and how a CSC can assess the security of workloads outsourced to the cloud.

Aside from proactive security measures and incident response procedures, intrusion detection is an essential security component of an enterprise system, whose objective is to identify signs of incidents through monitoring. The effectiveness of an IDS in detecting attacks is reflected in the attack coverage property, which can be calculated through security testing using purely malicious workloads and the attack detection accuracy metric [6]. Malicious workloads can be simulated using exploit databases, one of which is the ART framework[1] by Red Canary. When compared to the most popular attack framework, Metasploit, ART leads in the number of attacks targeting cloud services and therefore appears to be an appropriate choice for security testing. ART enables one form of penetration testing that emulates steps within a potentially larger attack chain in the form of micro-attacks (also referred to as atomics). Micro-attacks, therefore, cover single attack steps, similar to the concept of unit tests used for testing positive software requirements.

1.1 Problem Statement and Research Questions

This thesis delves into the security assessment of cloud workloads with a focus on a key security aspect: intrusion detection capability. Evaluating IDS in cloud computing has been declared as an open issue by Liu et al. [7]. Furthermore, the authors suggest that the various IDS techniques listed in their review should be examined in actual situations.

The task of evaluating an IDS is closely tied to the question of how a CSC can perform comprehensive security tests, as the conduct of security testing is the means to evaluate the effectiveness of the IDS. The field of cloud testing is the subject of only a few publications [8] [9] [10]. No related work could be identified that presents a detailed methodology for conducting security testing, and no results of cloud security testing are available to assess the security of cloud workloads comprehensively. It is stated that the negative requirements in security testing can be addressed with risk-based testing, and the mentioned testing method is penetration testing [9] [11]. A risk assessment is considered necessary for strategizing and managing testing [11] [12]. Considering the top threat categories within the threat landscape is assumed to represent a fundamental risk assessment and is required within a sensible penetration testing methodology.

Compared to the number of micro-attacks targeting Windows (1175) or Linux (370), only a fraction of micro-attacks within ART[2] are dedicated to cloud infrastructure. The cloud services of AWS, Azure, and Google Cloud are the target of only 60 tests in total.

Two research gaps were identified in this thesis. Firstly, there is an absence of literature concerning the evaluation of cloud IDS and cloud security testing as a means of assessing

[1] https://www.atomicredteam.io/. Accessed: Jan. 7, 2025.

[2] Atomic Red Team version reference: https://github.com/redcanaryco/atomic-red-team/tree/29b3bc58315cfc764da11739e35f2a37e3c0b39e. Accessed: May 19, 2025.

the security of cloud workloads. Secondly, there is an insufficient number of micro-attacks within the ART framework, which makes it impossible to conduct comprehensive defence system testing.

The objective of this thesis is to assess the threat detection capabilities of Microsoft Azure by elaborating the threat landscape of cloud workloads to define relevant threats for risk-based security testing. Due to the small number of micro-attacks within ART (second research gap), the second objective was to develop new micro-attacks. Finally, an investigation of the intrusion detection capabilities resulting from the security testing discusses the state of cloud workload security and outlines optimization possibilities for intrusion detection. Note that a secondary objective of this thesis is to evaluate the suitability of using ART as an attack database to achieve the main objectives. The guiding research questions are outlined below. Sub-research questions present sub-goals that are required to answer the general RQ.

- General RQ: How can the development and evaluation of micro-attacks enhance threat detection capabilities within Microsoft Azure's cloud workload security framework?
- RQ1: What threats and concerns need to be considered when outsourcing workloads to the cloud?
- RQ2: How well does the test library represent the threat landscape and what micro-attacks can be developed to improve the test coverage?
- RQ3: How can threat detection be optimized to respond to the simulated attacks?

The research questions represent the steps for assessing the security of a system, as presented by Annunziata et al. [9]. RQ1 covers a risk assessment, followed by a security testing phase in RQ2 and RQ3. RQ3 evaluates and improves the threat detection system based on the findings from RQ2.

1.2 Structure of the Work

The thesis is structured as follows. Chapter 2 presents the necessary background information and related work on cloud security testing and evaluating threat detection systems in the cloud. In Chapter 3 the research questions are explained, and the instantiation of the research method design science is presented. Chapter 4 outlines the procedure for elaborating the threat landscape and presents insights into the resulting threat landscape. Building on this, Chapter 5 presents an investigation of ART to cover the threat landscape along with details on the existing atomics. Chapter 6 outlines the process to develop new atomics. Chapter 7 explains the execution of the security testing experiments on the Cloud under Test (CUT) infrastructure. Chapter 8 discusses the threat detection capabilities of Azure and presents threat detection rules for the new atomics. In Chapter

9, answers to the research questions are given and insights of the security assessment of cloud workloads in Azure are presented. Furthermore, ART is assessed regarding its suitability for performing security tests, the contribution aspects of the thesis are summarized, and the future work is outlined. Chapter 10 concludes the thesis.

CHAPTER 2

Background & Related Work

This section presents the context and outlines related research. First, the relevant terminology and concepts are defined. Second, the architecture of Microsoft Azure is introduced. Third, related work on evaluating threat detection systems in the cloud is presented, followed by fourth, the related work on cloud security testing. Fifth, the MITRE ATT&CK® framework is introduced. Sixth, the ART framework is introduced and compared with Metasploit.

2.1 Clarification of Terminology

This section defines the terms used within this thesis. Cloud-related terms and security-related terms are presented in the following sections.

2.1.1 Cloud-related Terms

Cloud Computing: Cloud computing is defined in the National Institute of Standards and Technology (NIST) definition of cloud computing [13] through five essential characteristics, the concept of cloud service models, and the concept of cloud deployment models [14]. Essential characteristics are *on-demand self-service*, *broad network access*, *resource pooling*, *rapid elasticity*, and *measured service*. In short, customers can provision computing resources via standard network access on behalf of a CSP, using a multi-tenant and highly scalable infrastructure.

Cloud Service Models: According to the NIST [13] there are three service models that compose the cloud model: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). SaaS is a service type that is available as a ready-to-use application. PaaS offers the deployment of customer-created applications on a runtime environment provided by the CSP. IaaS is the lowest-level service in the architecture stack and allows the customer to provision processing resources, such as

storage and network. These three models provide a rough indication of the delineation of responsibilities, as outlined below.

Shared Responsibility: Due to the nature of cloud computing, security is a common task between the CSC and CSP. The term shared responsibility is described as follows:

> "The term 'shared responsibility' is widely used by multiple CSPs and refers to the delineation where the CSP takes responsibility for security operations and controls. Below the line is the CSP's responsibility; anything a CSC builds above the line is the CSC's responsibility" [14].

Cloud Security Alliance (CSA) [14] presents an overview of this delineation across the architectural layers in relation to the three cloud service models.

Cloud Deployment Models: Cloud computing appears in four deployment models: private cloud, public cloud, community cloud, and hybrid cloud [13]. The most popular model is the public cloud deployment model [14], which is defined as follows:

> "The cloud infrastructure is provisioned for open use by the general public. It may be owned, managed, and operated by a business, academic, or government organization, or some combination of them. It exists on the premises of the cloud provider" [13].

The cloud deployment model of interest in this thesis is the public cloud Microsoft Azure, which is a public cloud. Note that the multi-tenancy nature of public clouds introduces several governance challenges [14].

Cloud workload: One key term used in this thesis is cloud workload, for which several definitions with varying precision exist. According to NIST a cloud workload is:

> "A logical bundle of software and data that is present in, and processed by, a cloud computing technology" [15].

CSA defines it as follows:

> "Cloud workload refers to the various tasks, applications, services, and processes run in cloud computing environments. Cloud workloads allow for scalability, flexibility, and efficiency, enabling businesses and individuals to access and run applications or data processing tasks without investing heavily in physical hardware. Cloud workloads encompass a range of resources, including virtual machines (VMs), containers, serverless functions (also referred to as function as a service (FaaS)), AI, and PaaS." [14].

The aim of the paper by Mulia et al. [16] was to establish a comprehensive characterization of cloud workloads. The proposed categories are as follows:

> "Big streaming data, big database creation and calculation, big database search and access, big data storage, in-memory database, many tiny tasks (ants), HPC, highly interactive single person, highly interactive multi-person jobs, single computer intensive jobs, private local tasks, slow communication, real-time local tasks, location aware computing, real-time geographically dispersed, access control, and voice or video over IP" [16].

Deducing from these definitions, the term cloud workload is a rather universal term for various usage scenarios in the cloud environment and is described rather by categories from practice than by a clear definition. Cloud workload is used in this thesis as a term to refer to any resource outsourced to the cloud. These resources may be, for example, Virtual Machines (VMs), serverless functions, or storage units. Note that this thesis investigates the cloud environment in which workloads are realized. The object of investigation is therefore the environment and the conditions of the CSP Azure, rather then specific workloads. Hence, a more detailed definition of the term cloud workload is not necessary at this point. As we shall see later, the main workload for developing new micro-attacks in this thesis are Function Apps, which is the offered serverless solution within Azure. Also note that the aspects of the cloud environment for workloads that form the investigation scope for this thesis are the top three threats to cloud computing, which are elaborated in Section 4.3.

2.1.2 Security-related Terms

This section defines several security-related terms and outlines relations among them.

Threat: A threat is a potential adverse event. It is defined as follows:

> "Any circumstance or event with the potential to adversely impact organizational operations (including mission, functions, image, or reputation), organizational assets, individuals, other organizations, or the Nation through an information system via unauthorized access, destruction, disclosure, modification of information, and/or denial of service" [17].

There are several further definitions. This definition was used in the special publication 800-144 from Jansen and Grance [1] on security and privacy in public cloud computing, and is therefore considered the most fitting for this thesis.

Attack: The term attack is defined as follows:

> "An attack, via cyberspace, targeting an enterprise's use of cyberspace for the purpose of disrupting, disabling, destroying, or maliciously controlling

> a computing environment/infrastructure; or destroying the integrity of the data or stealing controlled information" [18].

Vulnerability: The term vulnerability is defined as follows:

> "Weakness in an information system, system security procedures, internal controls, or implementation that could be exploited or triggered by a threat source" [19].

The terms threat, attack, and vulnerability are closely related. A threat is a potential adverse event, which, in the more precise realization (as a tactic or technique), is called an attack. Every attack is a threat, but not every threat is an attack. The relationship of attack and vulnerability is as follows. An attack can contain the exploitation of vulnerabilities as a means for its realization as an attack. A threat may also be the potential to exploit a vulnerability in the form of an attack. In this thesis, both attacks and vulnerabilities are understood as sub-categories of threats.

Intrusion: An intrusion is defined as follows.

> "A security event, or a combination of multiple security events, that constitutes a security incident in which an intruder gains, or attempts to gain, access to a system or system resource without having authorization to do so" [20].

A more intuitive definition is given by Liu et al. [7].

> "An intrusion is a kind of attack on information assets in which the instigator attempts to gain entry into a system or disturb its usual operations" [7].

Intrusions are the objects of investigation of IDSs, which are designed to detect malicious activities and attempted attacks. Intrusions may result, on success, in an incident.

(Cybersecurity) Incident: There do exist numerous definitions of the term *cybersecurity incidents* [21], which embrace the smaller category of *cybersecurity, cloud incidents*. Similar to the work of Carriegos et al. [21], this thesis also deals with threats that are identified and listed by external sources. Therefore, a concrete definition does not make sense. The broad term *cybersecurity, cloud incident* is a cybersecurity incident that concerns cloud infrastructure. Regardless of the understanding of the referenced sources of the term *cybersecurity incident*, it is sufficiently precise and in this thesis designated as *incident*. In comparison to threats and attacks, incidents are actual events that have occurred. They are materialized attack events. As previously defined, threats are potential events, attacks are more concrete scenarios, and incidents are actual

occurrences. A characteristic of an incident is that it can occur intentionally (through malicious activity) or unintentionally, such as an accidental data leak.

Concern, Risk, and Challenge: The terms concern, risk, and challenge are closely related. For simplicity, the term concern is used throughout this thesis to represent all three terms. A concern is defined as a "Matter of interest or importance to a stakeholder" [22], security risk is defined as "The effect of uncertainty on objectives pertaining to asset loss and the associated consequences" [23], and challenge is defined as "[...] a currently difficult or impossible task that is either unique to cloud computing or exacerbated by it" [24]. Inferring from these definitions, a concern is understood for this thesis as a more vague inception and wider-scoped area of threats. Note that they are often used very imprecisely in papers.

Threat Landscape: Kaspersky defines the threat landscape as the "entirety of potential and identified cyberthreats affecting a particular sector, group of users, time period, and so forth" [25].

2.2 Microsoft Azure

This section introduces the cloud computing platform Microsoft Azure, which is one of the big three CSPs[3] and which is the target environment of this thesis. The following sections provide a brief explanation of the infrastructure and outline the rules for performing penetration tests against Azure.

2.2.1 Azure Infrastructure

In short, Azure provides cloud services that a CSC can use (in general) on a pay-as-you-go basis. This section provides an overview of the foundational architectural components of Azure. The identity of a CSC in Azure is referred to as an account. As presented on the Microsoft Learn platform[4], the management infrastructure in Azure comprises resources, resource groups, subscriptions, and accounts. Cloud services, such as VMs, databases, or storage, are called resources in Azure. Resources can be grouped into resource groups, and further, resource groups lie within a subscription, which serves as the management and billing entity within Azure. Accounts can have multiple subscriptions, which are in a hierarchical structure of possibly nested management groups. Management groups are used for the management of policies and access control.

Azure provides various products[5]. For example, AI and machine learning services, computing services, containers, storage, DevOps, IoT, networking, virtual desktop infrastructure, and many more. More detailed explanations of resources that are relevant for this thesis are introduced in the according sections in which they are relevant. Access to Azure's

[3] https://www.statista.com/chart/18819/worldwide-market-share-of-leading-cloud-infrastructure-service-providers/. Accessed: Jun. 2, 2025.

[4] https://learn.microsoft.com/en-us/training/modules/describe-core-architectural-components-of-azure/6-describe-azure-management-infrastructure. Accessed: Jun. 2, 2025.

[5] https://azure.microsoft.com/en-us/products/. Accessed: Jun. 3, 2025.

services and the management of cloud resources is provided through the Azure Portal[6], which is a web-based, Graphical User Interface (GUI). Another access possibility is the Azure CLI, a cross-platform command-line tool to manage resources[7]. Key services within Azure are Entra ID and the security services, which are explained in the following sections.

Entra ID

Entra ID[8] is Azure's Identity and Access Management (IAM) service. It is stated to be part of the PaaS platform offering[9]. However, it is a unique Azure service that differs from ordinary cloud resources. Entra ID is a Microsoft-managed directory service, which, in distinction to a normal Active Directory (AD) service, is a multi-tenant directory service. Tenant is the term for the identity that is signed up for a subscription and forms the individual Entra instance. Multiple tenants can be created for a subscription, and a tenant may be used across multiple subscriptions. A tenant comprises access-related objects such as users, groups, and applications.

One fundamental building block for IAM in Azure is Role-Based Access Control (RBAC)[10], which is a mechanism to manage access of Entra identities to cloud resources. The key concept in RBAC is that of a role, which is a list of actions that can be performed and therefore defines a set of permissions. Azure includes many built-in roles that define commonly used groups of authorized actions. Further, custom roles can also be designed by defining a set of actions. Roles provide access to Azure resources and can be assigned to users, groups, service principals, and managed identities[11]. A group is a set of one or multiple roles that are activated together when the group is activated.

One key service within Entra ID is the Entra Privileged Identity Management (PIM) [12]. It is a service that requires a license and is used to manage, control, and monitor access to resources. Some of its main features encompass just-in-time privileged access, time-bound access, and Multi-Factor Authentication (MFA). In PIM, administrators manage the role and group assignments of users. Roles can be configured as active or eligible. Eligible means that users must activate their assigned eligible roles in the PIM to get access to cloud resources. Active does not require such an activation. Furthermore, role assignments may be permanent or time-bound.

[6] https://learn.microsoft.com/en-us/azure/azure-portal/azure-portal-overview. Accessed: Jun. 2, 2025.
[7] https://learn.microsoft.com/en-us/cli/azure/get-started-with-azure-cli?view=azure-cli-latest. Accessed: Jun. 6, 2025.
[8] https://learn.microsoft.com/en-us/entra/fundamentals/. Accessed: Jun. 2, 2025.
[9] https://learn.microsoft.com/en-us/training/modules/understand-azure-active-directory. Accessed: Jun. 2, 2025.
[10] https://learn.microsoft.com/en-us/azure/role-based-access-control/overview. Accessed: May 21, 2025.
[11] https://learn.microsoft.com/en-us/azure/role-based-access-control/built-in-roles. Accessed: Jul. 2, 2025.
[12] https://learn.microsoft.com/en-us/entra/id-governance/privileged-identity-management/pim-configure. Accessed: Jun. 2, 2025.

Azure Security

This section presents an overview of Azure's security products. At first, it is important to note that, like other components, the security services are also rapidly changing. The recent evolution goes towards the Microsoft Defender portal[13], which serves as a central place for security operations. The integration of other Azure security components into the Defender portal is an ongoing process. For example, in April and May 2025, the Defender portal has been extended to integrate all Microsoft Sentinel use cases[14]. Microsoft Defender Extended Detection and Response (XDR) is also part of this continuous integration project[15]. XDR itself is a relatively new and centralized security product, which is the receiver and correlator of all other Defender products. Prior to these evolution events, Microsoft Sentinel was intended to be the central and unified security component.

The Microsoft security services map[16] presents an overview of existing security services and serves as a collection of security recommendations. The three areas of this cloud security benchmark are *Secure and protect*, *Detect threats*, and *Investigate and respond*. The former two are relevant for this thesis, and the products contained therein will be presented in the following. The concrete infrastructure setup for the security testing phase within this thesis is outlined in section7.2.4.

Threat Detection:

For threat detection, two main services are presented in the security services map[16]: "Microsoft Defender for Cloud" (referred to in this thesis as DfC) and "Microsoft Sentinel" (referred to in this thesis as Sentinel). Additionally, "Microsoft Defender XDR" (referred to in this thesis as XDR) is listed as a native defense suite for the coordination of detection, prevention, investigation, and response across various resources. It comprises multiple different Defender products for the various resources. Further assisting services for detection include Entra ID Protection, Azure Firewall, Azure Network Watcher, and Azure Policy. Entra ID Protection automates the identity-based threat detection and remediation. Azure Firewall Premium comprises a signature-based Intrusion Detection and Prevention System (IDPS) for network traffic. The Azure Network Watcher is a diagnostic tool for Azure virtual networks, and Azure Policy is a component that helps to assess compliance with organizational standards.

Microsoft Defender for Cloud (DfC) is a Cloud-Native Application Protection Platform (CNAPP)[17] and protects workloads and applications via security measures and practices. Sentinel is an Security Information and Event Management (SIEM) and Security Orchestration Automated Response (SOAR) solution that includes security analytics and threat intelligence. It provides a "single solution for alert detection, threat visibility,

[13] https://learn.microsoft.com/en-us/unified-secops-platform/overview-defender-portal. Accessed: Jun. 3, 2025.

[14] https://learn.microsoft.com/en-us/azure/sentinel/whats-new. Accessed: Jun. 3, 2025.

[15] https://learn.microsoft.com/en-us/defender-xdr/whats-new. Accessed: Jun. 3, 2025.

[16] https://learn.microsoft.com/en-us/azure/security/fundamentals/end-to-end#microsoft-security-services-map, Jun. 3, 2025

[17] https://learn.microsoft.com/en-us/azure/defender-for-cloud/defender-for-cloud-introduction. Accessed: Jun. 3, 2025.

proactive hunting, and threat response"[18]. Besides Microsoft's threat intelligence and AI detection, individual threat intelligence solutions can also be realized. More details on DfC and Sentinel are laid out below. Figure 2.1 presents an overview of various Microsoft Defender services and their integration with XDR and Sentinel. The entire organization's signals are forwarded to XDR and DfC. Through a data connector, the SIEM relevant logs can be forwarded to Sentinel, which therefore functions as the central component for alerts in the Azure environment.

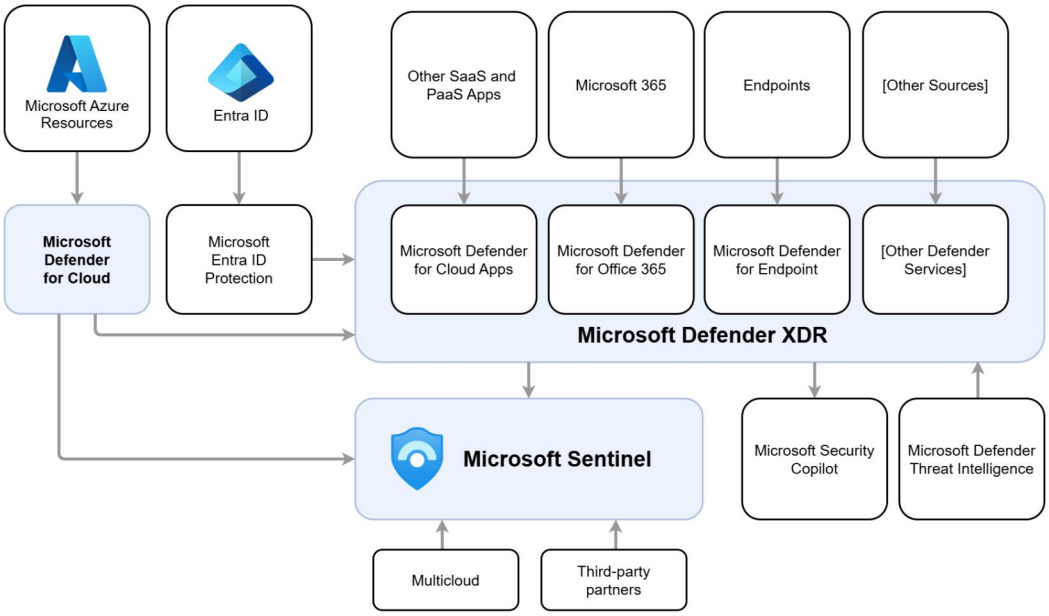

Figure 2.1: Adapted diagram of Microsoft Defender XDR integration with Microsoft Sentinel. The main threat detection services are colored blue. The central component is Microsoft Sentinel, which receives signals from DfC and XDR. Adapted from [26].

Sentinel[18]

Sentinel is the SIEM product in Azure, and it is the main component of interest for this thesis. It incorporates other Azure services, such as Log Analytics, Logic Apps, and Azure Monitor, which are the resources used to collect logs, automate workflows, and monitor log data. The ingestion of data from various sources in real time to Sentinel can be configured via data connectors[19]. Microsoft services can be integrated with out-of-the-box data connectors. Data from other services has to be integrated via custom connectors.

The threat detection capabilities of Sentinel include Analytics, ATT&CK coverage, Threat Intelligence, Watchlists, and Workbooks[18], out of which Analytics is the most important component for this thesis, since its functionality comprises the realization of Analytics

[18] https://learn.microsoft.com/en-us/azure/sentinel/overview?tabs=azure-portal. Accessed: Jun. 3, 2025.
[19] https://learn.microsoft.com/en-us/azure/sentinel/connect-data-sources?tabs=azure-portal. Accessed: Jun. 3, 2025.

rules (threat detection rules) and the generation of alerts. Aggregated and correlated alerts generate incidents. There do exist Analytics rules templates, which are pre-built prototype rules to cover known threats by their attack vectors[20]. Sentinel offers content types, such as Analytics rules, as out-of-the-box content accessible through the Content Hub[21]. More detailed information on the setup of Analytics rules is presented in Section 8.2.2.

Microsoft Defender for Cloud:

The second key threat detection service is the DfC, which, compared to Sentinel, performs rather static analyses and security scans of cloud services. However, active monitoring is also included in some Defender services, e.g., the monitoring of access attempts. The DfC encompasses Development Security Operations (DevSecOps), Cloud Security Posture Management (CSPM), and Cloud Workload Protection Platform (CWPP)[17]. DevSecOps ensures that security practices are incorporated in the development process. CSPM is about the proper configuration and deployment of cloud resources. It provides centralized policy management, generates a secure score, and presents a dashboard for weaknesses in the security posture. When misconfigurations are detected through policies, they turn into recommendations for changes. Cloud Workload Protection (CWP) (or CWPP)[17] capabilities include workload-specific security recommendations. The operations of the various Defender services, such as Defender for Servers and Defender for Storage, include, among others, monitoring access attempts, file integrity monitoring, and vulnerability assessments.

Secure and Protect:

Aside from threat detection services, *Secure and protect* within the Microsoft security services map[16] is another important category to ensure security within Azure. It is a category that comprises several proactive security services. In addition to the DfC, which is mentioned in this category as well, services such as Azure Policy, API Management, Azure Key Vault, Entra ID Protection, PIM within Entra ID, and MFA within Entra ID, among many others, are listed. The relevant services for this thesis, which haven't been explained already, will be briefly mentioned in the following. Azure Policy evaluates resources against a set of configured business rules (policy definitions), which can be assigned to all levels of groups of resources, from management groups to single resources. Thus, organizational standards can be enforced, and compliance checks can be conducted. One example could be to restrict resources to be deployed only in one defined Azure region, such as "West Europe". Conditional Access within Entra ID is another policy review service, used to block or grant access and enforce other security mechanisms, such as the enforcement of password changes and MFA for logins.

[20] https://learn.microsoft.com/en-us/azure/sentinel/threat-detection. Accessed: Jun. 3, 2025.
[21] https://learn.microsoft.com/en-us/azure/sentinel/sentinel-solutions. Accessed: Jun. 3, 2025.

2.2.2 Microsoft Penetration Testing Rules of Engagement

When conducting security testing against an Azure cloud infrastructure, it is important to comply with the "Microsoft Security Testing Rules of Engagement", as pointed out by Crawley [10]. These rules of Engagement[22] define a framework for security tests, such as penetration tests, and mainly comprise a list of prohibited activities and encouraged activities. Prohibited activities are, for example, performing Denial of Service (DoS) testing or interacting with storage accounts that are not part of one's subscription. Encouraged activities include, for example, testing the security monitoring and threat detection systems or testing the AI model's robustness[22]. The actions performed in the security testing experiments performed in this thesis comply with the rules of Engagement.

2.3 Intrusion Detection in the Cloud

Since this thesis focused on the threat detection capabilities in Microsoft Azure, this section presents related work on intrusion detection. Intrusion detection is part of threat detection, and NIST defines it as "The process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible incidents"[23]. In the literature, the terms IDS, Intrusion Prevention System (IPS), and the combination IDPS are often mentioned. IDS is about monitoring and recognizing active attacks, and IPS entails proactive and reactive prevention measures on a system [27]. IDPS combines both aspects and enriches IDS, which may detect a threat, with the ability to take responsive actions to prevent the intrusion attempt from succeeding [28].

Prabu et al. [27] provided an analytical assessment of IDS techniques for the cloud domain. An overview of the techniques, their attack detection capabilities, and the datasets used for evaluation is presented. According to Prabu et al. [27], IDS techniques can be categorized based on the source of information into the following types: Network-based IDSs and Host-Based IDSs. The former uses network traffic as a source of information, while the latter utilizes logs from individual host systems to perform monitoring and analytical actions. IDS techniques can also be categorized into the following types: Signature-based IDSs, Anomaly-based IDSs, and Hybrid techniques.

Another recent literature review on IDSs in cloud computing is presented by Liu et al. [7]. They systematically examined IDS mechanisms in cloud computing and presented another categorization of the different types of IDSs, i.e., Hypervisor-based IDS, Network-based IDSs, Machine Learning (ML)-based IDSs and Hybrid IDSs. There do exist further classifications according to various classification properties, such as those described by Kumar et al. [29] and Milenkoski et al. [6].

The investigation from Liu et al. [7] presents different techniques for implementing IDSs and points out that future work could examine these techniques in actual situations. Therefore, the investigation of IDSs in real-world attack simulations as part of a security

[22] https://www.microsoft.com/en-us/msrc/pentest-rules-of-engagement. Accessed: Jun. 6, 2025.
[23] https://csrc.nist.gov/glossary/term/intrusion_detection. Accessed: Jun. 14, 2025.

assessment phase formulates a research gap. This thesis does not evaluate a single IDS technique, as the publications in the reviews of Prabu et al. [27] and Liu et al. [7]. Instead, the threat detection capabilities within the Azure cloud in general are evaluated in a real-world attack simulation as part of a security assessment, and therefore contributes to the mentioned future work by Liu et al. [7]. Related work on the evaluation of IDS is presented in Section 2.3.1. This thesis focuses on the threat detection capabilities within Azure, in particular its Host-based IDS capabilities. Details on the threat detection capabilities of Azure are presented in Section 2.2.1.

2.3.1 Evaluating IDS

Liu et al. [7] declare that the evaluation of IDSs in cloud computing is an open issue and an active research area. The only paper that is cited in their work in this regard is from Milenkoski et al. [6]. Milenkoski et al. [6] present a systematization of standard practices for evaluating IDSs. They do not consider cloud-related IDSs in particular. However, the methodology is general and therefore suitable for the cloud domain as well. The standard components in an evaluation are the measurement methodology, workloads, and metrics. Workloads may be pure benign, pure malicious, or mixed, depending on the goal of the evaluation study. Metrics may be performance- or security-related. The latter one is relevant for this thesis and consists of basic metrics and composite metrics. The measurement methodology is defined as "the specification of the IDS properties of interest [...] and of the employed workloads and metrics" [6]. Properties of interest include the attack detection accuracy, attack coverage, attack detection- and reporting speed, and others.

The goal of measuring how effectively an IDS detects various attacks is reflected in the attack coverage property, which is defined as "The attack detection accuracy of an IDS in the presence of attacks without any background benign activity" [6]. This implies using purely malicious workloads for evaluations. Milenkoski et al. [6] state that the most promising method for the generation of pure malicious workloads is the usage of exploit databases. This thesis investigates the use of ART as an exploit database for the evaluation of Azure's threat detection capabilities. Refer to Section 2.6 for details on ART. A suitable metric for the attack coverage is the True Positive Rate (TPR), which is a basic metric that does not contain false alert measures. Background benign activity would generate false alerts. However, for the attack coverage, this workload type is not required. This thesis uses the TPR for the evaluation. Note that in this thesis, the term *detection rate* is used to refer to *attack coverage*, as it is more intuitive and easier to understand. Furthermore, *detection rate* is used as a synonym for TPR, which highlights the metric-centred attack coverage.

2.4 Testing Cloud Security

This section outlines related work on testing cloud security. In relation to evaluating an IDS, security testing has a broader scope, with the objective of assessing various security aspects and the overall system's security. The security testing method employed in this thesis is penetration testing, and the evaluation of the IDS attack coverage is the security aspect that will be investigated.

Bertolino et al. [8] determined in their survey that the term cloud testing is referenced in publications for the categories Testing of the Cloud (ToC), Testing in the Cloud (TiC), or Testing of the Cloud in the Cloud (ToiC). ToC refers to security tests that target cloud infrastructure. TiC uses cloud resources to conduct security tests, but the target is not in the cloud. ToiC combines both, i.e., the security tests are conducted within a cloud environment, and the target also lies in the cloud. Bertolino et al. [8] produced a comprehensive overview of publications that deal with cloud testing and concluded that for ToC, which is the field of interest for this thesis, the main objectives for cloud testing are performance tests, followed by functionality tests. Five publications are mentioned for conducting security testing on cloud infrastructure, one of which is the paper by Zech et al. [11], who propose a model-based solution for security testing, in which possible intrusion points are identified and malicious user techniques are applied to the system. They combine penetration tests and fuzzy testing, which are common security testing methods. However, their prototype wasn't finalized, and the evaluation hasn't been finished. The paper by Lee et al. [30] is mentioned as an example of ToiC with a security objective. Their study presents a cloud system that can efficiently simulate malicious code spread attacks. For publications categorized in TiC, security is mentioned as a test objective in five publications, in which various security concerns of non-cloud systems are tested. The publication of Bertolino et al. [8], therefore, shows that only a few publications exist that deal with security-related ToC.

Another recent study on testing cloud security is the paper from Annunziata et al. [9], who provide an overview of cloud security risk assessment studies. They mention that two procedures contribute to the security assessment on different levels. Integrating *risk assessment*, which enables proactive actions and the formulation of defensive strategies, into the *security testing* phase is stated to be critical for strengthening the overall system security. This risk-based testing approach is also mentioned to be highly relevant by Zech et al. [11] and stated as a best-practice approach for strategizing and managing testing in the ISO/IEC/IEEE 29119 [12]. Annunziata et al. [9] further mentions that risk assessments should adhere to standards, such as ISO/IEC 27017 [31], which addresses risk assessment for cloud security concerns. A risk assessment is stated to be crucial for selecting security testing methods, which comprise penetration testing, vulnerability scanning, code reviews, and security architecture reviews. The results from the literature review by Annunziata et al. [9] reveal that PaaS as a service delivery model is not addressed by any publication on its own. Instead, in some publications it is part of a risk assessment that considers all three service models jointly. On the contrary, SaaS-only and IaaS-only assessments do exist, and they dominate the cloud risk assessment domain.

Annunziata et al. [9] point out that this absence of PaaS-only publications suggests a potential gap in the risk assessment application purposes.

Other cloud testing literature exists that supplements this state of the literature with best practices. The CSA guideline [14] mentions the need for various testing executions according to the domain of interest. However, it provides only general information on testing and no detailed suggestions on methods and frameworks. Crawley [10] covers major CSPs and presents detailed information on the infrastructure, security tools, and pentesting possibilities for the top three CSPs. Therefore, it does contain information on the most common security testing method, i.e., penetration tests. However, their paper is intended to give assisting information for conducting penetration tests and does not present a comprehensive methodology to conduct a risk-based security assessment.

In conclusion, the field of cloud security testing has been the subject of several publications. However, no related work could be identified that explains a detailed methodology or presents results on cloud security testing. The security testing approach performed in this thesis can be categorized as security-related ToC. The scarcity of literature underscores the need for research in this field. The thesis goes beyond the state-of-the-art by performing a risk-based security testing on Azure using the relatively new open-source exploit database ART. ART realizes one form of penetration testing that splits steps within the kill chain of an Advanced Persistent Threat (APT) into micro-attacks, covering single attack steps, similar to the idea of unit tests for testing positive requirements of software. One paper that conducted research in a similar field is that of Oesch et al. [32]. They use the ART framework to build a training environment for autonomous cyber defense agents, called *Cyberwheel*. In distinction to Oesch et al. [32], this thesis investigates the potential to extend the ART library and evaluates the threat detection capability in the cloud by executing the existing atomic tests.

2.5 MITRE ATT&CK Framework

MITRE ATT&CK® is a knowledge base of adversary Tactics, Techniques, and Proceduress (TTPs), which is created from insights of real-world observations [33]. MITRE ATT&CK is an acronym that stands for “MITRE Adversarial Tactics, Techniques, and Common Knowledge” [34]. Strom et al. [33] introduce the terminology of the ATT&CK framework. The collection of TTPs is presented in the form of matrices. Columns represent tactics (the adversary’s tactical objective), and each tactic consists of a list of techniques that can be used to reach the objective. Techniques are broken down further into sub-techniques. There do exist three different matrices for different domains, i.e., Enterprise, Mobile, and Industrial Control Systems (ICS). Sub-matrices exist for the domain matrices, such as matrices for different operating systems. For this thesis, the cloud matrix is relevant, which is a sub-matrix representation of the Enterprise matrix. The cloud matrix consists of four further sub-matrices, i.e., Office Suite, Identity Provider, SaaS, and IaaS. Note

that the matrix composition changes regularly. This thesis uses ATT&CK version 16.1[24].

Strom et al. [33] clarify that ATT&CK is not a checklist of things to be addressed. Therefore, not all techniques can or should be a basis for analysis. Prioritizing and focusing on the relevant adversarial behaviors of the area of interest is necessary. According to Al-Sada et al. [34], ATT&CK is used as a foundation for threat modeling across various sectors, including academia and industry. It is also used for behavioral analytics, CTI enrichment, Adversary Emulation/Red Teaming, SOC Maturity Assessment, and Cyber Threat Intelligence Enrichment [34] [33].

This thesis uses ATT&CK as an intermediate representation of attacks for a defense gap assessment. As presented by Al-Sada [34] there exist some other papers that also use ATT&CK for a similar use case, however, none of the investigated papers deal with an application scenario in the cloud domain. The primary reason for using the ATT&CK framework for defense gap analysis is its popularity in threat modeling literature. This is also evident in the case of the ART framework, which maps micro-attacks onto techniques in the ATT&CK framework. ART is part of the investigation within this thesis and, therefore, presents the second reason for its suitability. ATT&CK is used as an intermediate format to connect existing micro-attacks to attacks in the threat landscape of cloud workloads.

2.6 Atomic Red Team

It was mentioned in Section 2.3.1 that exploit databases are used to evaluate IDS by simulating malicious workloads. This section introduces the ART framework, which will act for this thesis similarly to an exploit database. Note that the term exploit database may not be entirely accurate, as it does not contain ready-to-use exploits, but instead is used for simulating attack techniques in testing environments, for example, to validate the functionality of detection systems. Exploit databases such as Exploit DB[25] focus on exploiting reported vulnerabilities in software. Therefore, exploit databases are often used for vulnerability testing and may not be used exactly the same as ART. Section 2.6.1 introduces ART. Section 2.6.2 compares ART with Metasploit and motivates the usage of ART.

2.6.1 Introduction to Atomic Red Team

Atomic Red Team (ART)[26] is an open-source project by *Red Canary*. It is a collection of atomics, which are used for simulating adversarial activity. Atomics are generic attack scripts that can be executed with minimal effort by providing configurations through input parameters. The term micro-attack is related to the term atomic and is used in this thesis very similarly. The term micro-attack is used when setting the focus on the

[24] v16.1 corresponds to the time span October 31, 2024 to April 21, 2025.https://attack.mitre.org/resources/versions/. Accessed: May 17, 2025.

[25] https://www.exploit-db.com/. Accessed: Jun. 16, 2025.

[26] https://www.atomicredteam.io/atomic-red-team. Accessed: Mar. 1, 2025.

generic concept of the realization of attack steps. The term atomic (or atomic test) is equivalent to micro-attack, but it is used specifically in the context of ART. Atomics are mapped to the ATT&CK matrix, which enables a standardized representation of the executed attacks within the threat landscape. Atomics can be executed using the *Invoke-AtomicRedTeam*[27] tool, a PowerShell module provided by *Red Canary* that runs on Windows, Linux, and MacOS. Atomics do exist for the platforms *windows*, *linux*, *macos*, *containers*, *google-workspace*, *iaas:gcp*, *iaas:azure*, *azure-ad*, *iaas:aws*, and *office-365*. In Section 5.2 an evaluation of the existing Azure-related atomics is presented.

In the structural review of adversary emulation tools presented by Landauer et al. [35], ART is ranked third in their elaborated unified ranking of adversary emulation tools. It should be noted that their comparison does not include the cloud domain and, therefore, may not represent the situation regarding the scope of this thesis. This thesis evaluates the suitability of ART for conducting security tests in the Azure cloud and, therefore, contributes to the evaluation of ART as a tool for adversary emulation in a real-world scenario.

2.6.2 Comparison with Metasploit

This section discusses ART in relation to Metasploit[28]. Metasploit is the most famous penetration testing tool [6] [36]. It is an open-source framework maintained by *Rapid7*[29] and consists of various modules and payloads that can be used to perform attacks by exploiting vulnerabilities on a target infrastructure. At the time of writing, there existed 5,913 Metasploit modules[30]. There are 24 distinct modules that are dedicated to cloud infrastructure, which have been retrieved by searching for modules that contain the search term “cloud” (16 results), “azure” (2), or “aws” (10). The ART database[26] contains 1,722 atomics (without the added atomics of this thesis). There are 57 atomics that are dedicated to cloud infrastructure (*azure-ad*, *iaas:azure*, *iaas:aws*, *office-365*, *iaas:gcp*, or *google-workspace*). Therefore, compared to Metasploit, ART contains more cloud-related tests. When we compare the tests dedicated to the Azure cloud, we have two modules in Metasploit, versus 29 tests in ART. Inferring from this comparison, ART can be declared as a leading adversary emulation tool for cloud infrastructure.

There do exist further adversary emulation tools and exploit databases. However, a concrete comparison of ART to further frameworks lies outside the scope of this thesis. Sivamanikanta et al. [36] investigated the capabilities of Metasploit and provided a comparison with other penetration testing tools. Milenkoski et al. [6] also state similar exploit databases such as Nikto, w3af, and Nessus. This thesis focuses on ART. Evaluations of further tools are considered as future work.

[27] https://github.com/redcanaryco/invoke-atomicredteam. Accessed: Mar. 1, 2025.
[28] https://www.metasploit.com/. Accessed: Apr. 27, 2025.
[29] https://www.rapid7.com/. Accessed: Jul. 4, 2025.
[30] https://docs.metasploit.com/docs/modules.html#metasploit-modules. Accessed: Apr. 27, 2025.

CHAPTER 3

Research Method

This section presents the research method of this thesis. Firstly, the research questions and their structure are reasoned. Secondly, the instantiation of the research method design science by Wieringa et al. [37] is presented.

3.1 Research Questions

The goal of this thesis is to investigate and improve cloud workload security in Microsoft Azure by the development and evaluation of micro-attacks as a means to perform risk-based security testing. This superposed research goal is reflected in the general research question. In order to answer the general research question, three sub-research questions were defined. They follow the fundamental steps to correctly assess the security of a system, as presented by Annunziata et al. [9]. Hence, they are considered relevant and complete. RQ1 represents the *risk assessment*, while RQ2 and RQ3 correspond to the *security testing* phase. RQ2 evaluates and develops the attacks required for performing penetration testing, and RQ3 investigates the improvement of the threat detection system through the security testing. The mapping of research questions onto steps in the security assessment procedure is presented in Figure 3.1. The overall procedure is also known as risk-based testing, which is a method for testing negative security requirements and is stated to be highly relevant for security testing [38] [9]. Furthermore, the use of penetration testing as a concrete testing approach is the commonly used method for performing security tests [11].

General RQ: How can the development and evaluation of micro-attacks enhance threat detection capabilities within Microsoft Azure's cloud workload security framework?

The first part of the question assumes the ART framework to be not sophisticated enough to be used in security testing. To what degree the development of micro-attacks

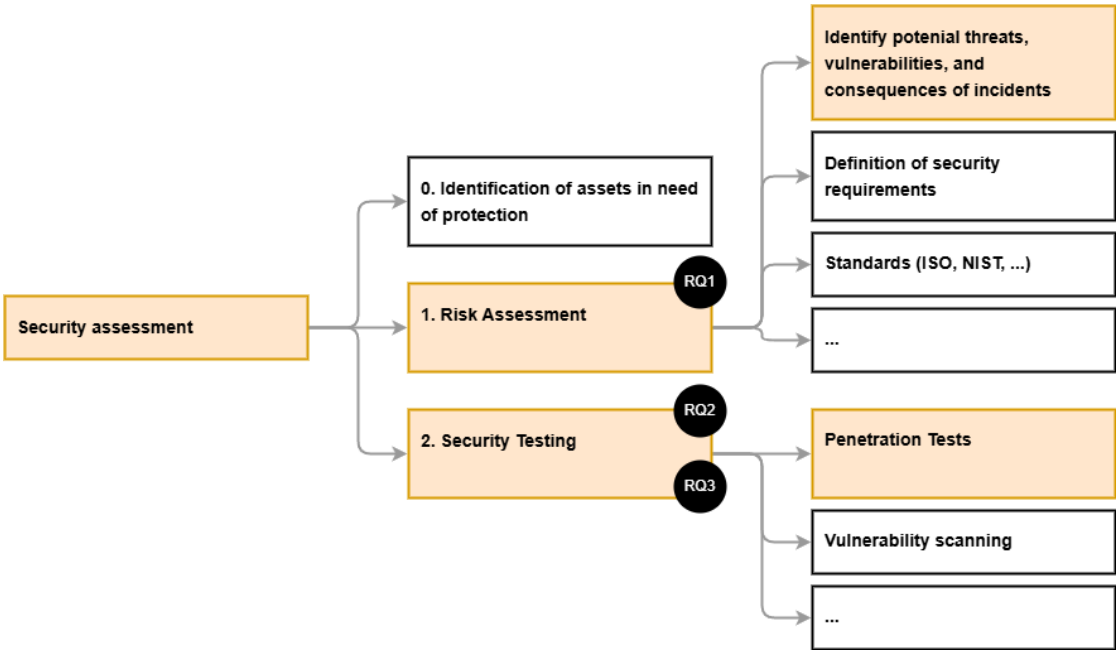

Figure 3.1: Research questions mapped onto the security assessment structure, as described by Annunziata et al. [9].

contributes to a reinforced tool for security testing is part of the research goal within this thesis. The goal of the second part of the question is to provide insights into the threat detection capabilities of Azure in the presence of a set of micro-attacks.

RQ1: What threats and concerns need to be considered when outsourcing workloads to the cloud?

RQ1 aims at presenting the threat landscape of cloud workloads. Annunziata et al. [9] mentions the identification of threats, vulnerabilities, and consequences of incidents as a crucial prerequisite for selecting a security testing method and proactively addressing security threats. It is required in order to understand which threat actors exist and which techniques they use to attack cloud workloads. The answer to this question provides the foundational information and insights into attack scenarios that need to be considered in security tests in order to assess threat detection systems. The top threats identified in RQ1 present the scope of investigation for the following thesis.

RQ2: How well does the test library represent the threat landscape and what micro-attacks can be developed to improve the test coverage?

RQ2 aims to develop new micro-attacks. First, an evaluation of the ART exploit database determined the coverage and state of existing micro-attacks aimed at addressing the most significant threats. These evaluation provides insights that guide the research for developing new micro-attacks that were not covered before, thereby increasing the overall test coverage rate. The outcomes of RQ2 lay the foundation for evaluating the threat

Object	Definition
Improvement problem	Securing cloud infrastructure to the current threat landscape: How can the development and evaluation of micro-attacks enhance threat detection capabilities within Microsoft Azure's cloud workload security framework?
Problem context	Azure cloud workload infrastructure
Artifacts 1	Micro attacks
Artifacts 2	IDS configuration, including detection rules

Table 3.1: Definitions of terms for the design science procedure in this thesis.

detection capabilities of Azure in RQ3. This evaluation takes into account the current threat landscape and how well the ART test library is able to represent it.

RQ3: How can threat detection be optimized to respond to the simulated attacks?

The investigation object of RQ3 is the threat detection system of Azure. The investigation involves the investigation and enhancement of the threat detection system in the context of attacks within the ART database. In order to answer RQ3, an evaluation of the IDS in Azure was conducted, and the optimization possibilities were discussed.

3.2 Design Science Methodology

This section outlines the research methodology employed to answer the research questions. It follows the design science methodology presented by Wieringa et al. [37] and was adapted to meet the research goals. Design science is used for improvement problems, which are solved by the interaction of artifacts in a problem context. The design science terms for this thesis are presented in Table 3.1. Besides the overall improvement problem, i.e., "securing cloud infrastructure to the current threat landscape", there are two engineering goals for this thesis. Firstly, the development of new micro-attacks for the ART framework, and secondly, configuring an intrusion detection infrastructure and designing detection rules for the newly developed attacks. These two engineering goals are reflected in the two types of artifacts, which correspond to a design cycle, respectively.

As defined by Wieringa et al. [37], design science research is a process that iterates over two activities, i.e., designing artifacts and empirically investigating those artifacts in a context. These two activities correspond to a design cycle, which is determined by a design problem, and an empirical cycle, which is defined by a knowledge question. An empirical cycle "is a rational way to answer scientific knowledge questions" [37]. When the empirical cycle is part of a higher-level engineering cycle, the knowledge question asks for the evaluation of a design implementation within a specific context. When the empirical cycle is not part of an engineering cycle, the application scenario for the results must be credible. In this thesis, each empirical cycle is either necessary to present the knowledge context as a basis for the design science process, or it is part of an engineering

cycle with a preceding design cycle. The cycles do not prescribe processes, but instead are used to clarify the methodology and ensure that when designing a treatment, the problem is defined properly and it is validated afterwards.

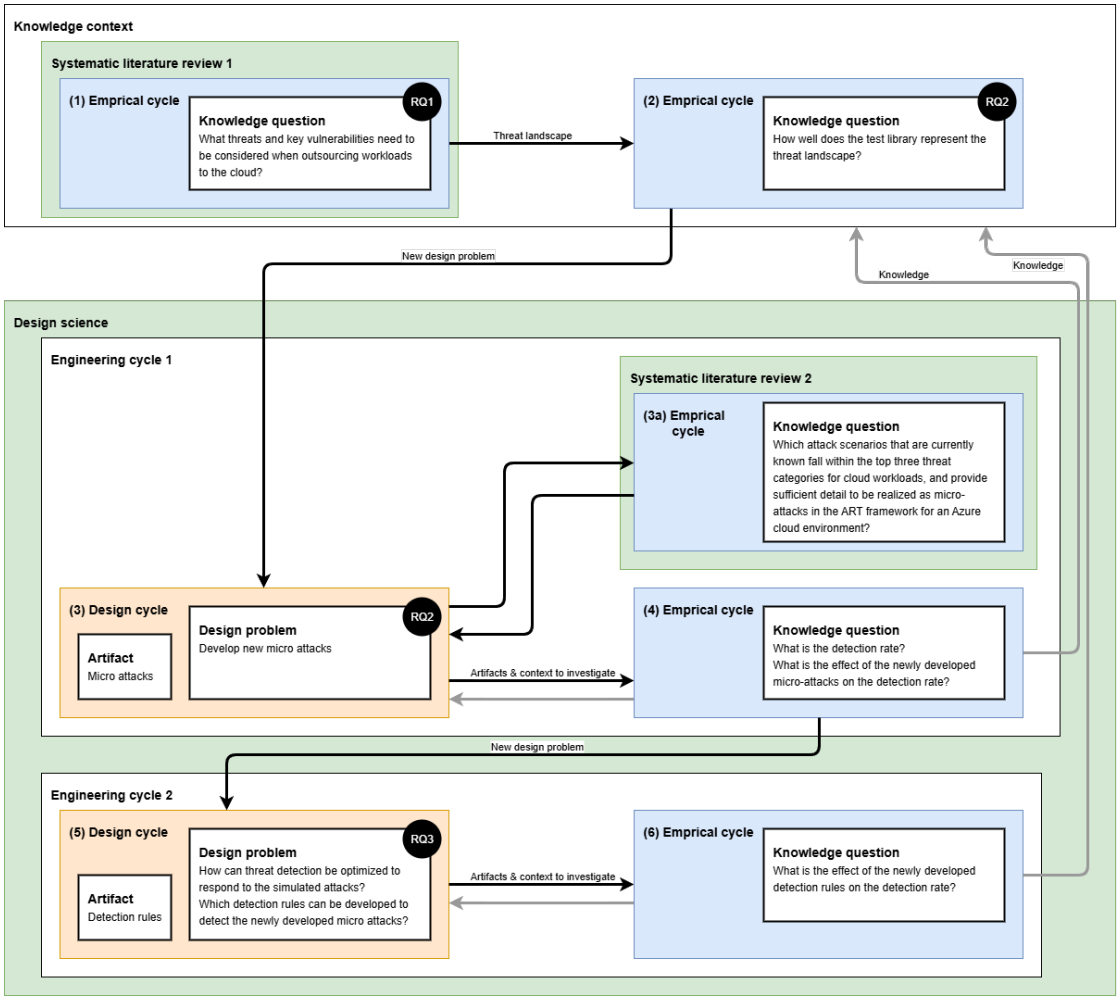

Figure 3.2: Methodological structure of this thesis.

The methodological structure of this thesis is presented in Figure 3.2. The upper box represents the knowledge context, which encompasses the state-of-the-art research knowledge, which was extended by an Systematic Literature Review (SLR) to investigate the threat landscape. The top left green box visualizes this SLR. It consists of one empirical cycle with the knowledge question RQ1. The output of this step leads to cycle two, i.e., the first part of RQ2, which elaborates the test coverage of existing micro-attacks in relation to the top threats in the threat landscape. The bottom, green box represents the design science process, which encompasses two engineering cycles, each comprising one design cycle, and a follow-up empirical cycle. Engineering cycle one contains one additional empirical cycle, which is necessary to research attack scenarios that may be

realized as micro-attacks in cycle three. Engineering cycle two is the subsequent cycle, in which the objective is to research and improve the threat detection setup.

In the following, each cycle of the research methodology is presented in more detail. The checklists for design and empirical cycles in Appendix A and B from Wieringa et al. [37] were used to ensure consistency with a correct design science methodology.

(1) Empirical Cycle:

The first cycle could be omitted as a cycle, since its result presents part of the knowledge context of the overall design science process. However, it was defined as the first empirical cycle, as the process needs to be carried out using a scientific methodology. The objective is to present the threat landscape, which will be achieved through an SLR, which is a research type that is not covered by the design science process presented by Wieringa et al. [37]. Instead, the SLRs follow the methodology presented by Kitchenham [39]. The answer to the knowledge question of this empirical cycle provides the necessary context information of threats to cloud workloads, as well as a ranking of threat classes. The result of cycle one will therefore contain a categorization of threats and concerns in the domain of cloud workloads. To narrow the research scope for this thesis, the top three threats identified in this cycle present the scope for the following two research questions in this thesis. Cycle one is covered in Chapter 4, and the design science terms are defined in Table 3.2.

Object	Definition
Knowledge goal	Problem investigation: Threat landscape
Improvement goal	The application scenario for the project result is RQ2.
Knowledge question	What threats and concerns need to be considered when outsourcing workloads to the cloud?
Research method	SLR by Kitchenham [39]

Table 3.2: Definitions of terms for cycle 1.

(2) Empirical Cycle:

In the second cycle, the test coverage of existing attacks within the ART library on the top three threat categories will be evaluated. All micro-attacks within the ART framework are assigned to a technique identifier of MITRE's ATT&CK framework. Therefore, the current set of existing attacks can be represented by the ATT&CK matrix. This representation, in combination with the representation of the top three threat categories, enables the calculation of the test coverage, which answers the first half of RQ2, i.e., how well does the test library represent the threat landscape? The test coverage is presented in Section 5.1. Answers to the second half of RQ2, i.e., what micro-attacks can be developed to improve the test coverage, are achieved within engineering cycle one. The design science terms for cycle two are given in Table 3.3.

Object	Definition
Knowledge goal	Problem investigation: Determine test coverage
Improvement goal	The application scenario for the project result is RQ2.
Knowledge question	How well does the test library represent the threat landscape?

Table 3.3: Definitions of terms for cycle 2.

(3) Design Cycle:

This design cycle aims to design micro-attacks. The design science terms are defined in Table 3.4, and the cycle is covered in Chapter 6. The design process consists of the following steps:

1. Perform SLR to find potentially relevant attack scenario(s).
2. Identify relevant attack scenario(s).
3. Define CUT.
4. Decompose relevant attack scenario(s) into micro-attacks.

Object	Definition
Artifact	Micro-attacks
Design problem	Develop new micro-attacks

Table 3.4: Definitions of terms for cycle 3.

Step one is outsourced to empirical cycle (3a), which is described below. Step two identifies a suitable attack scenario from the set of potentially relevant attack scenarios. Step three comprises the definition of a CUT, which will be used to develop micro-attack scenarios and evaluate the new treatments in the subsequent empirical cycle (cycle four).Therefore, it presents the concrete cloud architecture within the broader term "cloud workloads", which will be investigated in more detail in the remaining methodology of the thesis. In step four, attack scenarios and steps within these scenarios, as observed in intrusion reports and that are not yet represented as atomic tests in the ART test library, will be developed and added to the existing test set. Note that cycles three and four were not executed in succession, but rather simultaneously.

(3a) Empirical Cycle:

This empirical cycle aims to find potentially relevant attack scenarios. Similar to cycle one, this step was performed using the SLR methodology by Kitchenham [39] and is presented in Section 6.1. The design science terms are given in Table 3.5.

In Figure 3.2, this cycle is visualized in the top right box within engineering cycle one. The knowledge goal of this SLR is the identification of potentially relevant attack

Object	Definition
Knowledge goal	Find threat intelligence reports with information on the research question (knowledge question).
Improvement goal	The application scenario for the project results is design cycle three.
Knowledge question	Which attack scenarios that are currently known fall within the top three threat categories for cloud workloads, and provide sufficient detail to be realized as micro-attacks in the ART framework for an Azure cloud environment?
Research method	SLR by Kitchenham [39]

Table 3.5: Definitions of terms for cycle 3a.

scenarios. In the defined timespan of 2022 to 2024 (the last three years), cloud incidents from case reports were collected. In order to limit the search range for the scope of this thesis, the set of sources for gathering the data is defined to include the incident reports of three threat intelligence organizations, i.e., CSA, CrowdStrike, and Orca Security. The result of this part contains qualitative and quantitative results. Qualitative in terms of concrete information about incident events and quantitative in terms of counting the incidents according to the category they belong to, or the TTPs they encompass. This SLR provides the necessary insights into incidents that are essential for developing new attack scenarios.

(4) Empirical Cycle:

Cycles four and six represent the security testing phase in the overall security assessment goal of the thesis, as presented in the Security Testing Chapter 7. They are empirical cycles that evaluate the state of security of the CUT in the presence of a defined set of ART attacks by the attack coverage metric. The design science terms of cycle four are specified in Table 3.6. The second knowledge question asks for the effect of the newly developed micro-attacks on the detection rate. It represents the validation step of design cycle three. Empirical cycle four consists of the following steps:

1. Investigate and evaluate existing micro-attacks (Section 5.2).
2. Define a set of atomics that is suitable for security testing (Section 7.1).
3. Execute security testing experiment (Section 7.3).
4. Evaluate detection results (Section 7.4).

As the first step in this security testing phase, the existing micro-attacks need to be investigated, followed by the selection of a set of suitable atomics for the test setup. This set of existing atomics, including the newly developed atomics, is executed in the experimentation phase in step three. An evaluation of the metrics to answer the knowledge questions ends this cycle.

Object	Definition
Knowledge goal	Technology validation
Improvement goal	Part of the higher-level design/engineering cycle. See the goal of cycle three.
Knowledge questions	What is the detection rate achieved by the threat detection setup? What is the effect of the newly developed micro-attacks on the detection rate?

Table 3.6: Definitions of terms for cycle 4.

(5) Design Cycle:

This design cycle aims to improve the threat detection rate. The insights from empirical cycle four provide the necessary information to configure a threat detection system that detects the newly developed atomics. The artifacts of this second design cycle consist of detection configurations and detection rules. Cycle five is presented in Section 8.2 and the corresponding design science terms are defined in Table 3.7. The configuration of a detection configuration includes the following steps:

1. Configuration of the threat detection tools in Azure, including the realization of a log infrastructure to collect all required log sources.
2. Identification of potentially relevant log sources.
3. Development of detection rules for the newly developed micro-attacks.

Object	Definition
Artifact	IDS configuration, including detection rules
Design problem	How can threat detection be optimized to respond to the simulated attacks? Which detection rules can be developed to detect the newly developed micro-attacks?

Table 3.7: Definitions of terms for cycle 5.

Step one configures the threat detection setup of the CUT. It was not executed strictly after cycles three and four, but was rather conducted in an iterative process. The final execution of the experiments includes this configuration of the threat detection system in the CUT already in engineering cycle one. Note that it is necessary that this configuration is the same to have comparable detection metric results. Similarly to step one, the log infrastructure from step two is part of the CUT. Steps two and three comprise the actual design process, which consists of an investigation of relevant log sources and the development of detection rules to detect the newly developed micro-attacks.

Object	Definition
Knowledge goal	Technology validation
Improvement goal	Part of the higher-level design/engineering cycle. See the goal of cycle five.
Knowledge questions	What is the detection rate achieved by the threat detection setup? What is the effect of the newly developed detection rules on the detection rate?

Table 3.8: Definitions of terms for cycle 6.

(6) Empirical Cycle:

Similar to cycle four, this cycle evaluates the designed artifacts from the preceding design cycle. The empirical cycle consists of the following steps:

1. Execute security testing experiment.
2. Evaluate detection results.

Cycle six is the final cycle and should provide the required numbers for answering the general research question. The design science terms of the cycle are given in Table 3.8. Similar to cycle four, the knowledge question is defined to ask for the detection coverage. The changed context in this iteration is the presence of the detection solutions presented through cycle five.

CHAPTER 4

Threat Landscape

This section presents the procedure for elaborating the threat landscape of cloud workloads, along with insights into the resulting threat landscape. Furthermore, a ranking of the threats, the subsequent definition of a scope for this thesis, and a representation of that scope in the ATT&CK matrix format are presented.

4.1 Literature Review Process

This section presents the SLR to provide a comprehensive view of the threat landscape of cloud workloads. The theoretical framework of this SLR follows the guidelines given by Kitchenham [39]. It could also be defined as a systematic mapping study, as also mentioned by Kitchenham [39]. Systematic mapping studies are used to present an overview of a particular topic. They differ from SLRs in a more detailed data extraction phase, which is more of a categorization of data. Note that the data synthesis phase of this thesis is rather a summary instead of an in-depth meta-analysis.

The SLR review encompasses three phases: planning the review, conducting the review, and reporting the review. The phases were not conducted in a strict waterfall-like process, but they represent an iterative process. This section presents phase one in Section 4.1.1 (planning the review) and the steps within phase two (conducting the review) in Section 4.1.2 to 4.1.5. Phase three (reporting the review) is presented in Section 4.2, which presents the results.

4.1.1 Planning the Review

The planning phase of the SLR consists of (1) identification of the need for a review, (2) specifying the research question(s), and (3) developing a review protocol.

(1) Identification of the Need for a Review:

As presented in the Research Method in Chapter 3, this SLR represents empirical cycle one and aims to present the knowledge context. To the understanding of the author, there does not exist a comprehensive survey that addresses this knowledge question. Furthermore, the rather unspecific domain of cloud workloads offers a broad scope for interpretation and should therefore be systematically investigated, and the range of answers synthesized.

The ATT&CK matrix[31] provides a common overview of the attack landscape, presenting an in-depth analysis of attacks within the cloud domain. The matrix contains attack techniques that are categorized into tactics. Compared to the ATT&CK matrix, the threat landscape resulting from the SLR will, firstly, contain threats at a wider scope on cloud security. Recall that attacks are concrete manifestations of threats. The broader term threat does not need to be an attack, but can be, for example, a vulnerability. Therefore, threats, along with concerns as an even broader threat term, are missing in the ATT&CK matrix. They are required to conduct risk-based security testing. Secondly, the ATT&CK matrix does not contain a unified ranking of attacks, nor does it include classifications, such as the layer in the shared responsibility model to which an attack belongs. This information is required for this thesis to filter to a specific scope and prioritize certain threats over others. Although ATT&CK is not used as a representation of the threat landscape, the ATT&CK framework is still used as a standardized representation format for potential attacks. The corresponding MITRE techniques in which a threat falls is one classification category in the threat landscape. Details on the mapping process of threat landscape objects to the ATT&CK techniques are presented in Section 4.4.

(2) Specifying the Research Question:

The research question of the SLR is presented by the knowledge question of the empirical cycle, which is given in RQ1: "What threats and concerns need to be considered when outsourcing workloads to the cloud?". The result can be named *threat landscape*, which is a common term in security-related publications that designates the entirety of threats affecting a specific sector [25]. A collection of threats and a ranking of the top threats comprise the required data for this thesis.

(3) Developing a Review Protocol:

The review protocol defines the steps for conducting a specific SLR. As laid out by Kitchenham [39], the review protocol is a critical element and must be defined before conducting the review to minimize the possibility of bias. The review protocol[32] was accepted by the supervisor of this thesis. The structure of the defined review protocol and the SLR, which is presented schematically in Figure 4.1, is split into two parts according to the two types of literature sources. The reason for these discriminatory

[31] https://attack.mitre.org/matrices/enterprise/cloud/. Accessed: Jan. 29, 2025.

[32] https://github.com/manuelkuss/Investigation-of-Cloud-Workload-Security/blob/ab226cdb054e3e3629aafde1cbf8afab20d54236/Threat-Landscape/Systematic-Literature-Review/SLR_threat_landscape_review_protocol.md

search strategies is due to the nature of the sources. In part (1.1), scientific publications can be found in digital libraries and scientific search engines via Boolean expressions. The content of publications is indexed with keywords, the abstract, and further fields. Therefore, relevant publications can be found through search strings. In part (1.2), threat intelligence reports from reputable organizations may not be found in the same way. Reports are often not indexed, and other less-reputable organizations may be ranked higher on search engines, even though they may not contain the required information. Hence, the selection of literature was conducted in two distinct processes, which are further explained in Section 4.1.2 and 4.1.3. The processes of the subsequent data extraction phase (2) and data synthesis phase (3) were conducted according to the same strategy for both literature types.

(1)	Study Selection: Split into two parts according to the literature type				
	(1.1)	Selection strategy for scientific publications		(1.2)	Selection strategy for organizational reports
		(1.1.1) Find search query			(1.2.1) Potentially relevant organizations
		(1.1.2) Potential relevant primary literature			(1.2.2) Relevant organizations
		(1.1.3) Primary literature			(1.2.3) Potentially relevant reports
					(1.2.4) Relevant reports
(2)	Data Extraction				
(3)	Data Synthesis				

Figure 4.1: Schematic representation of the SLR process.

4.1.2 SLR (1.1): Study Selection Process of Scientific Publications

This section provides a brief description of the SLR process. More detailed information is given in the review protocol[32].

Identification of Research: Develop Search Query:

The search query was developed in step (1.1.1) by conducting a preliminary search, followed by two phases of trial searches, and a final modification phase.

The purpose of the preliminary search was to find initial papers that contain the required information in order to construct useful search terms. The preliminary search was conducted through manual searches in five relevant search engines, i.e., *Springer Nature Link*, *IEEE Xplore*, *ACM Digital Library*, *Google Scholar*, and the TU search engine *TU CatalogPlus*. The top three relevant publications were selected by manual assessment from the preliminary search. They are determined to contain relevant information to the research question and are therefore used in the subsequent trial searches to find relevant search terms. In the following, these three publications are designated as guidance papers.

Trial searches have the objective of constructing a search query that leads to the guidance papers. It is aimed to develop search queries that produce search results in which guidance papers are ranked among the top results. In the first trial search, the three guidance papers are considered separately. For each paper, a query was constructed by using

terms from the sections within each paper that contain information providing answers to the research question. A query was constructed of those terms that led to high ranks (rank 1 (*Springer Nature Link*), rank 1 (*IEEE Xplore*), and rank 2 (*TU CatalogPlus*)) in the search engine of the corresponding journal in which the paper was published, or the search engine TU CatalogPlus, in the case that it was not published in a journal that is associated with one of the relevant search engines defined before. In the second trial search, the three search queries from the first trial search are merged into a single query. This generalization intends to find a universal search query that produces good search results within all search engines. The first step of the generalization process is to break down the search terms of the three search queries into individual facets (categories). Afterwards, another execution of trial searches optimizes combinations of the search terms. The query that produced the best result for all three libraries was chosen as a generalization result. Ranks 1 (*Springer Nature Link*), 26 (*IEEE Xplore*), and 3 (*TU CatalogPlus*) are the results of this query.

In the final modification phase, the generalized search query is extended with synonyms and alternative expressions. The defined facets, which are *keywords on the cloud context*, *attack types* and *current trends*, remain the same in this step. The final query for this SLR is defined in Listing 4.1. The first group encompasses terms that appear in the context of cloud workloads. Recall that the term cloud workload is loosely defined, as explained in Section 2.1. The context for cloud workloads considered for this thesis will instead be derived from this SLR. The second group contains terms that represent the concept "attack types", and the third group contains terms that indicate current trends of threats and concerns.

```
("cloud computing" OR "cloud services" OR "SaaS" OR "Software as a Service" OR "IaaS"
    OR "Infrastructure as a Service" OR "PaaS" OR "Platform as a Service" OR "cloud
    security" OR "cloud workloads" OR "outsource to cloud" OR "migrate to cloud" OR "
    cloud outsourcing")
AND
("attack types" OR "attack techniques" OR "attack methods" OR "attack strategies" OR "
    security vulnerabilities" OR "security threats" OR "identified threats" OR "
    identified vulnerabilities")
AND
("current trends" OR "recent trends" OR "ongoing trends" OR "present trends" OR "latest
     trends" OR "threat landscape" OR "threat overview" OR "vulnerability landscape" OR
     "vulnerability overview" OR "existing threats" OR "existing vulnerabilities" OR "
    known threats" OR "known vulnerabilities" OR " " OR "potential threats" OR "
    security-related concerns" OR "security-related issues" OR "security-related risks"
     OR "security-related challenges" OR "security concerns" OR "security issues" OR "
    security risks" OR "security challenges")
```

Listing 4.1: Search query for SLR (1.1).

Having developed a search query that leads to relevant publications regarding the research question, the search engines (resources) to be searched must be defined. Three main digital libraries in computer science were chosen, i.e., *IEEE Xplore*, *ACM Ditial Library*, and *Springer Nature Link*. Furthermore two general search engines were selected to supplement the list of publications from the journals with further resources. The additional search engines are *Google Scholar* and the TU intern search engine, *TU CatalogPlus*.

Study Selection:

Having a search query and resources to be searched, the study selection criteria need to be defined. Two criteria further narrow down the search results, i.e., (1) the publication year must lie between 2022 and 2024, and (2) the publication type must be one of the following: Journal (research article), Magazine, Book, or Book Chapter. Conference proceedings and other types of content were excluded. This ensures that the results contain current and comprehensive (reviewed) information regarding the research question. The retrieval of *potentially relevant primary literature* in step (1.1.2) is achieved by conducting the search query on the five mentioned resources and the presented filter criteria. From each resource, the top 10 results were taken as a set of potentially relevant primary literature. The three papers found by manual research were also added to this set. This leads to a list of 48 distinct publications. In the subsequent step (1.1.3), the set of *relevant primary literature* was selected from the set of potentially relevant primary literature. The selection procedure was as follows. All publications were classified into categories such as *publication year*, *quality score*, *usability score*, *amount of relevant information*, and others. This classification enabled the selection among the following criteria. The numbers represent the number of publications that pass the filter criterion.

1. Potentially relevant literature: 48
2. Recheck publication type: 47
3. Contain relevant information for RQ: 30
4. Date greater than or equal to 2022: 22
5. Amount of relevant information is at least one page: 14
6. Usability score is at least 4: 6

This ensures that the resulting publications do contain relevant information for the research question, the data and publication type are relevant, and only publications are considered that provide a certain amount of relevant information. More details on the concrete selection procedure as well as the definition of the usability score are defined in the review protocol[32]. The resulting set of *relevant primary literature* consists of six publications, i.e., [40], [41], [42], [43], [44], [45].

4.1.3 SLR (1.2): Study Selection Process of Organizational Reports

Figure 4.1 shows the report selection process (1.2) on the right column, which consists of identifying (1.2.1) potentially relevant organizations, filtering out (1.2.2) relevant organizations, (1.2.3) collecting potentially relevant reports within those relevant organizations, and filtering out a set of (1.2.4) relevant reports.

The list of potentially relevant organizations (1.2.1) was defined as the six Leaders in Gartner's Magic Quadrant of Endpoint Protection Platforms (EPPs)[33], and was supplemented by CSA as a known relevant organization, and Red Canary, as the owner of the exploit database of this thesis, i.e., ART. The dominance of CSA in the cloud security field, as underlined by various scientific publications that reference their reports and guidelines, justifies its inclusion in the set.

Four out of those eight organizations were chosen as a set of (1.2.2) relevant organizations. The selection was made by the criteria that the organization publishes reports on cloud threats and therefore potentially contains information relevant to the research question, and that the reports must have been published within the time range of 2022 to 2024. The resulting set of organizations is defined as potentially relevant reports (1.2.3). The selection criterion to filter out relevant reports (1.2.4) was defined as follows: reports must contain an overview of threats to cloud workloads. In the case of multiple reports (e.g., a yearly report on cloud threats), only the newest report is chosen. The resulting set comprises four reports, i.e., [2], [46], [47], [48]. Details on SLR (1.2) are presented in the review protocol.

4.1.4 Data Extraction

Data extraction is the procedure to fetch relevant data from the set of relevant primary literature (six scientific publications and four organizational reports). The information categories of interest are defined by the data extraction form, which was developed according to a taxonomy of security-related terms and is part of the review protocol[32]. The definitions of the relevant terms, as well as the taxonomy, is provided in Section 2.1.2.

The procedure for data extraction is as follows: First, while reading a publication, the objects of interest are identified. *Objects of interest* can be either concrete security-related objects, such as threats, attacks, vulnerabilities, or they can be more general security-related objects, such as concern, risk, challenge, or issue. Second, the object of interest was inserted into a table to bring the extracted data into a uniform structure. The columns of this row were populated according to the information presented in the publication. Columns are the defined information categories in the data extraction form. They lie within four groups, i.e., security-related categories, cloud-environment-related categories, ranking information, and other auxiliary fields. Security-related categories define the type of the object of interest. Cloud-environment-related categories define the domain in which the object of interest can be located, such as the cloud service model or the specific cloud component. There is one field for information on the ranking of the object. Auxiliary fields are, for example, the reference to the page where the object is mentioned, or a comment with additional information.

[33] https://www.gartner.com/doc/reprints?id=1-2IV5W7LE&ct=240920&st=sb. Accessed: Jun. 19, 2025.

4.1.5 Data Synthesis

The data synthesis step uses the collected data objects from the data extraction and synthesizes them. The synthesis methodology is a descriptive synthesis that maps the extracted data objects onto generalized data objects. The concrete process is described in detail in the review protocol[32]. It follows the suggestions on structuring the table format by Kitchenham [39] and was conducted in multiple iterative phases. In each iteration, a copy of the extracted data object was created, and it was subsequently cleaned and revised. Objects may have been merged with another existing object into one unified object that fits both, or may have been added as a new object. The classification categories from the raw extracted data objects were revised, and the objects were finally classified as threat, concern, or marked as irrelevant. This final division into a threat landscape table and a concern landscape table is the result of the SLR.

4.2 Literature Review Results

This section presents the resulting threat landscape data of the SLR. An overview of the data and some relevant statistics are presented in sections 4.2.2 and 4.2.3. The subsequent sections (4.3, and 4.4) further investigate aspects of the threat landscape data that are necessary for answering RQ1.

4.2.1 Pre-processing

This section describes the pre-processing steps that were conducted on the synthesized threat landscape data.

Filter threats: A filter step removes the following threats: Firstly, threats that represent too general threat categories, such as “privilege escalation”, are removed. Secondly, threats, which lie fully within the responsibility of the CSP, are removed since this thesis deals with threats that must be addressed on behalf of the CSC. Third, threats that specify types of malware, such as ransomware and spyware, are removed. There is no gained knowledge from variations of malware for this thesis; however, it is rather relevant to discuss, for example, the impact of incidents, which may be relevant for other investigations.

Mapping: Two manual mapping steps were carried out. Firstly, the shared responsibility layer that corresponds to the identified threats was filled out. Two out of the three threat categories associated with the scope of this thesis align with a specific layer in the shared responsibility model. Additionally, we included a third category, *Is API-related*, in this mapping process, as it is the third threat category within the scope of this thesis. Recall that the scope is set to the top three threats and is presented in Section 4.4. Secondly, a mapping of each attack to the technique object of MITRE’s ATT&CK cloud enterprise matrix[31] was conducted. This mapping is required for the ATT&CK representation in Section 4.4. The reason why only attacks are mapped is that ATT&CK techniques represent attacks. Vulnerabilities are not included, so they cannot be mapped to that

matrix. The mapping process was conducted as follows. First, an initial manual mapping iteration was performed by the author of this thesis. Second, all mapping-relevant objects were cross-checked and finalized in collaboration with a cybersecurity expert from the collaborating company of this thesis.

4.2.2 Concerns in the Cloud

As defined in Section 2.1.2, concerns encompass a broader spectrum of threats and therefore provide an overview of the areas that companies planning to utilize cloud computing services must address. As presented in Figure F.1, cloud security concerns can be grouped into the following sectors: *Privacy & Compliance*, *Operational & Process risks*, *Cloud technology concerns*, *Data security, -management and -integrity*, *Infrastructure and network security*, and *Software/Application security*. All concern objects that result from the synthesis in the SLR are listed in the right column. They are grouped into categories and subcategories, presenting a hierarchy of all cloud security-related concerns for outsourcing workloads to the cloud. This hierarchy, as presented in Figure F.1, is a remnant of this thesis - a result from the threat landscape representation. It represents the concern landscape and consists of wider security concerns, which are no concrete attacks and vulnerabilities, and are therefore outsourced to this second table. Concerns should be used for a comprehensive security assessment and may be used in future works.

4.2.3 Threats in the Cloud

This section presents statistics on the data of the elaborated threat landscape. Recall that a threat is either an attack or a vulnerability. Other, more general threats are classified as concerns and are therefore present in the concern landscape table. In the raw threat landscape table, there are 87 attacks and 40 vulnerabilities. The pre-processing, as explained in Section 4.2.1, results in a filtered table, consisting of 85 threats, i.e., 48 attacks and 34 vulnerabilities. The numbers are presented in Table 4.1. Part of the data extraction categories are the cloud service models that are explicitly mentioned for some threats. We observe that there are 11 threats mentioned explicitly for the IaaS service model, 10 for PaaS, and 17 for SaaS. Note that these are the *explicitly* mentioned service models in the literature. No revision step has been performed to check and complete the according service models to all threats. Instead, all threats were categorized according to the shared responsibility layer, which was considered more useful for this thesis. It provides a more specific view and can be considered with respect to the shared responsibility layers between the CSC and CSP. Section 4.4 outlines the scope of interest for this thesis.

Statistics on the Shared Responsibility Layer Categorization:

Figure 4.2 shows the number of threats categorized in each shared responsibility layer. Note that the lowest four layers (“Virtualization”, “Servers”, “Storage”, and “Physical Security”) have no assigned threats, as they were removed in the pre-processing step. This thesis focuses only on threats for which the CSC is at least partly responsible. The

Category	Count (before)	Count (after)
Threat	139 (100 %)	85 (100 %)
Attack	87	48
Vulnerability	40	34
Is threat category	18	-
CSP is fully responsible	25	-
IaaS	11	11
PaaS	10	10
SaaS	17	17

Table 4.1: Count of categories in threat table before and after the pre-processing.

number of threats in the "Configuration" layer dominates the threat landscape, followed by threats in "IAM" and threats assigned to the "Application" layer. Note that a single threat may be mapped to multiple layers and therefore may be present in multiple bars in the plot.

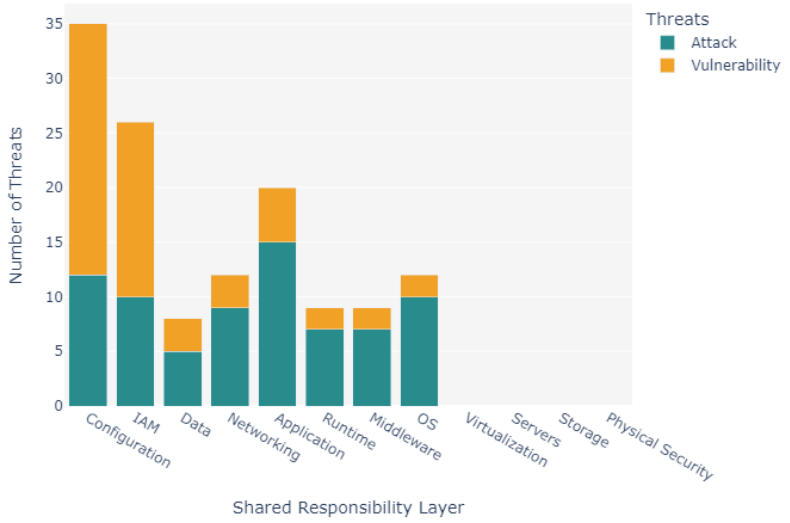

Figure 4.2: Number of threats categorized in each shared responsibility layer.

Statistics on the Top Referenced Threats:

Table 4.2 presents the top ten attacks according to a distinct count of the references in which they are mentioned. The top-referenced attacks are "DoS/DDoS attack" and "malware injection attack". They are mentioned in seven references. "Access/Expose sensitive data" is ranked third. Table 4.3 presents the top ten mentioned vulnerabilities. The top three vulnerabilities are "Insufficient/missing access control", "Insecure interfaces and APIs", and "Misconfigured services (storage/computing instances/...) and security-

Rank	Count	Attack
1	7	DoS/DDoS attack
1	7	(Cloud-related) malware injection attack
2	6	Access/Expose sensitive data
3	5	Phishing
4	4	Disable(d)/misconfigure(d) monitoring and logging (on a heterogeneous workload system)
6	3	Account hijacking / compromised accounts
6	3	SQL injection
6	3	Traffic eavesdropping
6	3	Sniffing
10	2	Password reset attack

Table 4.2: Top 10 most referenced attacks (distinct reference count).

Rank	Count	Vulnerability
1	7	Insufficient/missing access control
2	6	Insecure interfaces and APIs
3	5	Misconfigured services (storage/computing instances/...) and security-related settings
4	4	Disable(d)/misconfigure(d) monitoring and logging (on a heterogeneous workload system)
4	4	Software/application flaws (e.g., unpatched software)
4	4	Excessive permissions/privileges
4	4	Ineffective/insecure recovery/backup techniques (lack of a good disaster recovery plan)
8	3	Insufficient/missing authentication
9	2	Poor secret management and practices
9	2	Insufficient/missing authorization

Table 4.3: Top 10 most referenced vulnerabilities (distinct reference count).

related settings". Note that vulnerabilities in the elaborated threat landscape may not be very specific, but rather define threat categories, rather than concrete vulnerabilities that would have, for example, a Common Vulnerabilities and Exposures (CVE) number.

4.3 Threat Ranking

This section presents a threat ranking that results from insights into the raw extracted data of the previous SLR. The procedure to retrieve the ranking, the resulting ranking list, and the definition of the scope of the thesis to the top three threats is outlined in the following.

The methodology for retrieving a ranking of the top threats to cloud workloads involved identifying ranking information in the extracted data and unifying multiple lists into a synthesized ranking list. The data extraction form of the SLR includes the attribute "rank", which represents information on the severity of the threat. Investigating the raw extracted data (preceding data synthesis) from scientific publications, we observe that this attribute is empty for the great majority of all objects. For threats, where this field is not empty, in most cases, only a vague description of severity is present. Examples are "Major concern", "One of the major security issues", and "Second most serious cybersecurity concern". The threat information is singular in nature and not included in a ranked list of threats. One exception is the paper from Marinescu et al. [40], which presents a ranked list of threats, which was included from an organizational report from CSA on the top cloud security threats in 2016 [49]. This leads to another observation from the extracted data, i.e., data from organizational reports, as opposed to data from scientific literature. Three out of four organizational threat reports contain a clear ranking of threats, all of which were published in 2024 and are therefore considered up-to-date. The ranking lists are presented in Table 4.4. They constitute the source data for retrieving a united ranking.

The methodology for producing a unified ranking consists of several steps: First, threat names are combined into a common name. For example, "Inadequate access management" [47] and "Identity and Access Mgmt (IAM)" [2] are combined as "Mistakes in Identity and Access Management". Second, a rank was used to replace individual scoring systems. In Table 4.4, this general rank is already present. Several methodologies have been considered for the normalization step of the different ranking score systems, such as z-score and min-max scaling. However, for the goal of this step, i.e., identifying the top three threats, such statistical normalization would induce further problems for interpretation and is considered not required for the use case of this thesis. Third, a combined ranking was instead determined by the Borda Count[34], a simple voting system, to obtain a rough ordering of threats. The inverted rank is used as a score metric for this step. The final rank is computed by summing the scores of equivalent threat objects. Table 4.5 presents the ranking that results from the Bora Count method. In the example of "Insecure interfaces and APIs", rank 2 with score 9 (from CSA), rank 2 with score 10 (from PaloAlto Networks), and rank 8 with score 4 (from SentinelOne) are summed up. The resulting score is 23, which yields rank 1 in the final ranking. The final step in the procedure is a manual correction of the third-ranked threat. The threats ranked first and second clearly result from the Bora Count "sum of scores". The subsequent four threats have a

[34] https://towardsdatascience.com/classical-classifier-combination-techniques-voting-approaches-borda-counts-and-5b047faaffbc/. Accessed: Feb. 13, 2025.

Rank	CSA - Top Threats to Cloud Computing	PaloAlto - State of Cloud-Native Security Report	SentinelOne - Cloud Security Report
1	Misconfiguration & Inadequate Change Control	AI-Generated Code	Data breaches and theft of intelectual property
2	Identity & Access Mgmt (IAM)	API Risks	Malware and fileless attacks
3	Insecure Interfaces and APIs	AI-Powered Attacks	Unauthorized access
4	Inadequate Selection/ Implementation of Cloud Security Strategy	Inadequate Access Management	Violations of data privacy/confidentiality regulations
5	Insecure Third-Pary Resources	CI/CD's Impact on the Attack Surface	Account hijacking
6	Insecure Software Development	Insider Threats	Denial of services (DoS) attacks
7	Accidental Cloud Disclosure	Unknown, Unmanaged Assets	Accidental exposure of credentials
8	System Vulnerabilities		Insecure APIs
9	Limited Cloud Visibility/Observability		Data leakage by insiders
10	Unauthenticated Resource Sharing		Malicious insiders
11	Advanced Persistent Threats		Cryptomining and other cooption of cloud resources

Table 4.4: Threat rankings from the reports Cloud Security Alliance [2], PaloAlto Networks [47], and SentinelOne [48].

very close score (11 to 13). The threat "Misconfiguration & Inadequate Change Control" was chosen as the third-ranked threat due to several reasons: Firstly, the second-ranked threat "Mistakes in IAM" is often also categorized as a misconfiguration threat, which would advocate misconfigurations as a subcategory. Secondly, misconfigurations are more present in the threat landscape. As presented in the statistics on the shared layer categorization, as shown in Figure 4.2, the "Configuration" group comprises the largest group of threats, indicating the urgency of this threat type. Thirdly, when considering the statistics on the top vulnerabilities, as presented in Table 4.3, we observe that this threat ranks equal to the third-ranked vulnerability. It should be noted that the top three vulnerabilities are identical to the top three threats, as retrieved in this procedure. Furthermore, they are identical to the top three threats from the CSA ranking. This fact matches with the observation that CSA is referenced by many publications as the leading authority in the cloud security sector.

Definition of the Scope of the Thesis:

The top three threat categories that need to be considered when outsourcing workloads to the cloud are listed below. They define the investigation scope for the thesis. Each

Threat	CSA score	PaloAlto score	SentinelOne score	Sum of scores	Rank
Insecure Interfaces & APIs	9	10	4	23	1
Mistakes in Identity and Access Management	10	8		18	2
Misconfiguration & Inadequate Change Control	11			11	3
Insecure Software Development	6	7		13	4
AI-Generated Code		11		11	5
Data breaches and theft of intelectual property			11	11	6
Malware and fileless attacks			10	10	7
AI-Powered Attacks		9		9	8
Unauthorized access			9	9	9
Insider Threats		6	2	8	10

Table 4.5: Top ten results in the combined threat ranking.

threat category is described in more detail in the threat report of CSA [2].

1. Insecure Interfaces and APIs: Application Programming Interfaces (APIs) and User Interfaces (UIs) serve as entry points and are often integral components of developed software. Therefore, strong security practices are necessary. CSA [2] lists various reasons why these interfaces may become vulnerable. Examples are "inadequate authentication mechanisms", "lack of encryption", or "improper session management".

2. Mistakes in Identity and Access Management: This thesis will demonstrate through a privilege escalation scenario resulting from an incoherent and complex privileges structure, that identity management may be risky and complex. Differences among the various CSPs may lead to misunderstandings of concepts and, therefore, mistakes in the IAM realization. The rapidly changing nature of cloud architectures adds to the relevance of this threat [2].

3. Misconfiguration & Inadequate Change Control: This threat category encompasses the incorrect setup of cloud resources, which leads to vulnerabilities that attackers may exploit. Common misconfigurations, as presented by CSA [2], are, for example, "secrets management", "storage access", or "overly permissive access to virtual machines". Inadequate Change Control refers to the inadequate realization of change control practices for cloud infrastructure. The technological differences between traditional change control processes and change control processes in relation to the cloud may result in misconfigurations [2]. The highest-ranked threat represents a subcategory of this threat category. Mistakes in the IAM are also misconfigurations. However, this thesis defines this third category to represent all "Misconfiguration & Inadequate Change Control" threats that are not related to IAM.

4.4 Top Three Threat Categories Mapped to the MITRE ATT&CK Matrix

This section presents the mapping of attacks in the top three threat categories onto the ATT&CK matrix. The ATT&CK framework was introduced, and its usage has been justified in Section 2.5. The representation through ATT&CK enables the connection between the micro-attacks in ART and the scope of this thesis, i.e., the top three threats in the threat landscape. The matrix representation is therefore used in Section 5.1 as a standardized naming convention for attacks to answer the first part of RQ2, i.e., "How well does the test library represent the threat landscape?". The mapping procedure of attacks in the threat landscape was presented in Section 4.2.1. Attacks within the threat landscape have been enriched in a mapping step to the corresponding techniques in the ATT&CK knowledge base in column "ATT&CK number" and "ATT&CK name" in the threat landscape table[89].

Figure 4.3 visualizes the representation of attacks that lie in the top three threat categories. They are colored blue. The underlying matrix (white and blue cells) represents the cloud enterprise matrix, as specified by MITRE. The visualization process was performed using the "ATT&CK Navigator"[35] tool. Details on the generation of the visualizations are presented in Appendix D. Attack techniques in the top three threats cover 28.67 % of all cloud-related enterprise techniques.

[35] https://github.com/mitre-attack/attack-navigator/. Accessed: May 16, 2025.

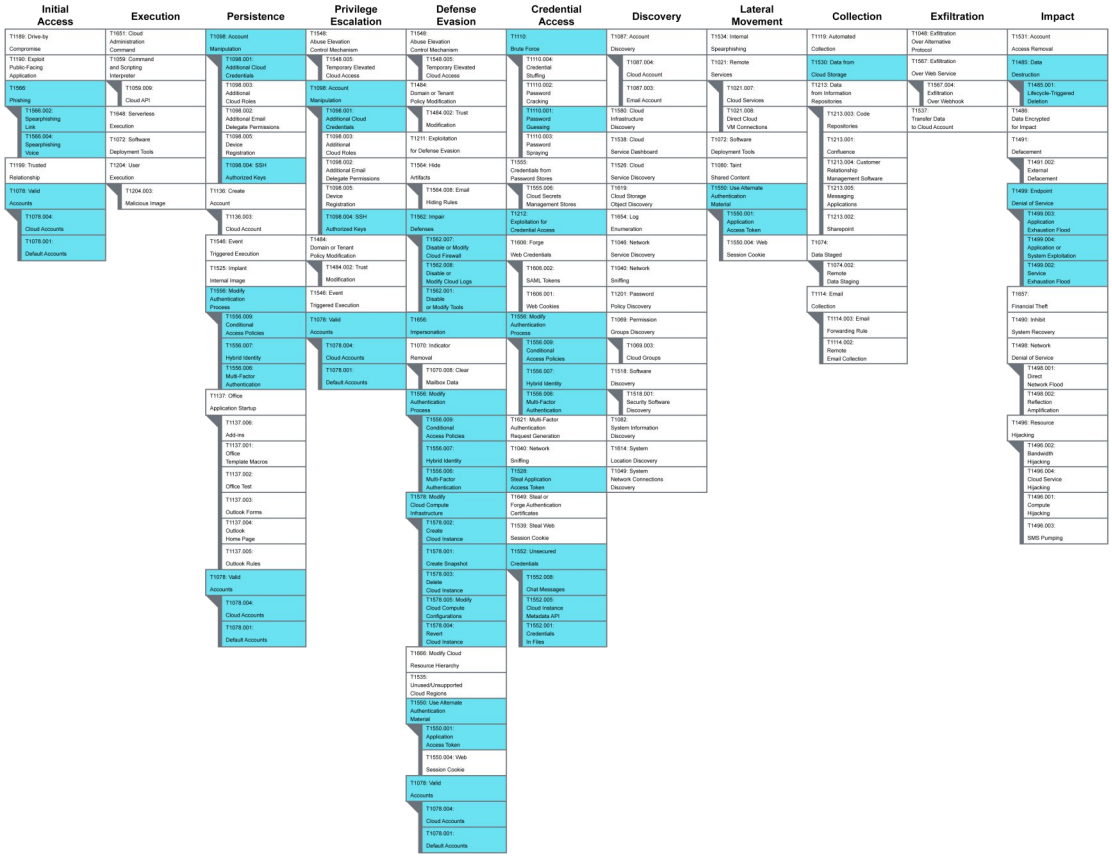

Figure 4.3: Visualization of attacks within the top threat categories (blue) in a filtered set of all cloud-relevant attack techniques.

CHAPTER 5

Investigation of Atomic Red Team

This section presents insights into the attack repository of ART. Its capability to cover the scope of the top three threats is presented in Section 5.1. The subsequent Section 5.2 presents detailed insights into the existing Azure-related atomics and thereby completes the discussion of the test coverage in 5.1 for assessing its suitability for comprehensive security testing. The ART repository was investigated on the 13th of March 2025, using the version[2] at that time. Section 5.2 deals with the state of the atomics at this exact state. However, due to the need to reclassify atomic tests in terms of the corresponding ATT&CK techniques and the supported domains, some changes were made prior to the test coverage calculation in Section 5.1. Except for Section 5.2, all sections refer to this modified state of the existing atomics. The changes are documented in Appendix C.

5.1 Test Coverage

Similar to the mapping of attacks within the top three threats onto the ATT&CK matrix in Figure 4.3, the existing atomics can be mapped onto the matrix. Figure 5.1 illustrates all MITRE techniques that are linked in Azure-related atomic tests within the ART database. It represents the test coverage of all enterprise cloud attack techniques that are related to the Azure cloud. Details on how the set of Azure-related atomics was retrieved are presented in Section 5.2. The enterprise cloud matrix is 11.19 % covered by atomics. Note that in Figure 5.1 there is no information on the number of existing atomics per technique. Also, note that a red-colored technique does not mean that the technique is 100 % covered by atomic tests. The visualization gives only a rough picture of cloud-related atomics in the ART database. In Section 5.2 a gradient representation that visualizes information on the number of atomics per technique is given.

The test coverage of the existing atomics on the scope of this thesis can be illustrated by combining both Figures (4.3 and 5.1). This combined visualization is presented in Figure 5.2. It represents the intersection of techniques from the top three threats

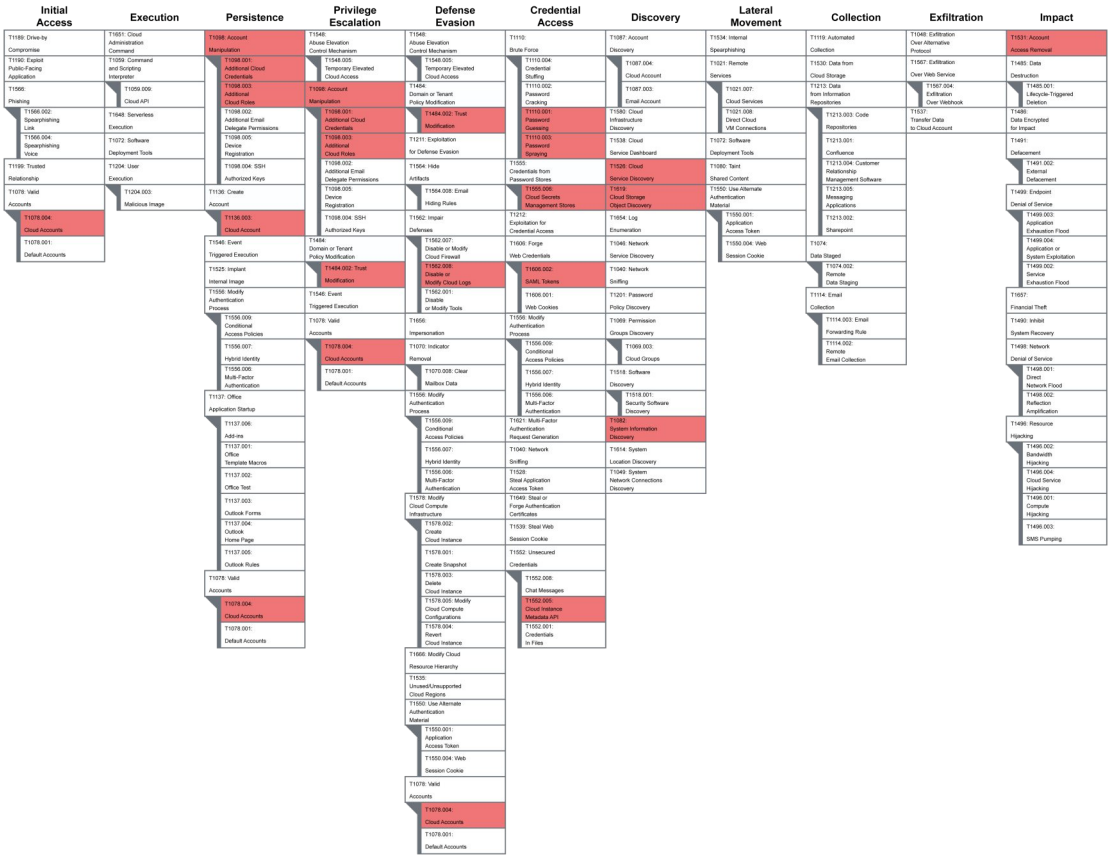

Figure 5.1: Visualization of ATT&CK techniques that are mentioned in atomic tests upon the cloud enterprise matrix.

(blue) and techniques mentioned in atomic tests (red). The intersected techniques are colored purple. The test coverage metric was calculated with the formula depicted in equation 5.1. Red-colored techniques are not in the defined scope and will therefore not be taken into account. Note that techniques in the ATT&CK matrix schema may be categorized at either the parent or child level (i.e., techniques and sub-techniques). Sub-techniques are more specific descriptions of techniques - they present a lower-level description. Despite the sub-technique being included in a technique, the metric considers all objects (techniques and sub-techniques) similar. The reason is that atomics can exist on both levels. Therefore, all objects are considered as if they are on the same level. Note that there is no need to take the number of atomics into consideration, since we don't get more information for the coverage perspective. For example, a technique with one atomic and a technique with two atomics both cannot answer the question of how well the technique itself is covered. For the objective of calculating a foundational test coverage, which will be discussed with more detailed insights into the atomics in Section 5.2, the magnitude is not included in this metric.

$$\text{test\_coverage} = \frac{\#\text{purple\_techniques}}{\#\text{purple\_techniques} + \#\text{blue\_techniques}} \tag{5.1}$$

Equation 5.1: Formula to calculate the test coverage. The variables are defined as follows: #purple_techniques := the number of techniques that are in the top three threats and have at least one associated atomic; #blue_techniques := the number of techniques that are in the top three threats and are not associated with any atomic.

The resulting test coverage of atomics within the top three threats is 10.26 %. This is a rough estimate that does not provide definitive information on how well ART represents the threat landscape. However, we can use it as an orientation point to interpret the suitability of the existing atomics. A deeper investigation of the existing atomics enables an interpretation of the test coverage number to assess the suitability of ART for security testing. One insight that directly follows from the test coverage metric is the inverse test coverage, which is 89.74 %. This number represents the number of techniques in the top three threat categories that are not covered by any atomic test.

5.2 Investigation of Existing Atomics

This section presents insights into the existing atomics within the ART repository. The investigation extends the assessment of the suitability of ART to cover the threat landscape, and it lays the foundation for defining an attack set for security testing in Section 7.1.

First, the existing atomics can be analyzed regarding their target platform. The number of atomics per platform is presented in Table 5.1. Note that atomics may contain multiple supported platforms. The number of atomics is therefore not a unique representation of the number of atomics across platform categories. We observe that most atomics are dedicated to *windows* (1,175 atomics), followed by *linux* (370) and *macos* (236). Cloud infrastructure (*iaas_aws*, *azure-ad*, *iaas_azure*, *office-365*, *iaas:gcp*, and *google-workspace*) is the target of 60 tests in total. The first finding, therefore, reveals that cloud-related atomics seem to be underrepresented, compared to atomics that target operating systems. This thesis is about cloud workload security within the Azure cloud. Therefore, the investigation of atomics within this thesis deals only with the "supported_platforms" *azure-ad* and *iaas:azure*, which together comprise 29 atomics. A categorization of these 29 atomics is presented in a CSV file on GitHub[36]. In this list, the following observations are made:

1. There are two atomics, each of which exists in two implementation versions (4 atomics in total). For example, the atomic test "Azure AD - Create a new user via Azure CLI" replaces "Azure AD - Create a new user". It uses the PowerShell

[36] https://github.com/manuelkuss/Investigation-of-Cloud-Workload-Security/blob/ab226cdb054e3e3629aafde1cbf8afab20d54236/Investigation-of-existing-Atomics/existingAtomics_investigation.csv

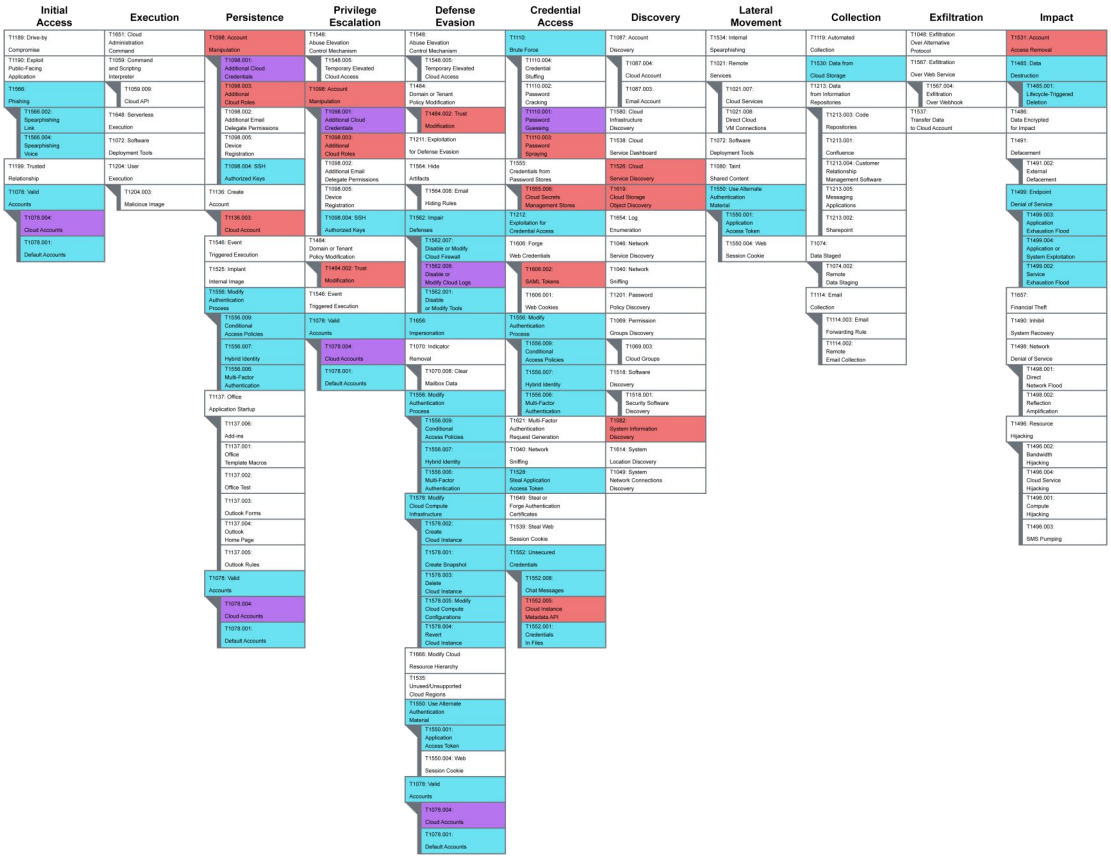

Figure 5.2: Combined visualization of threats within the top three threat categories (blue), and threats that are referenced by atomics in the ART database (red). Intersected techniques are colored purple.

module "Azure CLI" instead of the deprecated module "AzureAD", which is used by the older corresponding atomic. Note that some duplicate atomics do replace the stated objective of the atomic test, but the methodology may be different. Details may differ for execution.

2. One atomic is incorrectly classified with regard to the platform: the atomic "New-Inbox Rule to Hide E-mail in M365" should be categorized instead of the *azure-ad* platform to the *office-365* platform.

3. 16 atomics rely on deprecated PowerShell modules.

4. One atomic, i.e., "Azure - Dump Azure Instance Metadata from Virtual Machines", does not make sense in the existing version. The person who executes the attack must perform the action from within the VM, since the web request to the API is only possible within the VM. Hence, the scenario in the atomic test is not realistic,

Platform	Number of atomics
windows	1175
linux	370
macos	236
iaas:aws	20
azure-ad	19
containers	15
iaas:azure	10
office-365	6
iaas:gcp	5
google-workspace	1

Table 5.1: Number of atomics per platform.

Categories	Number of atomics
Basic user management activities	4
Basic role management activities	5
Basic resource operations	2
Login attempts	3
Storage access attempts	2
Reconnaissance operations to dump information	6
Other	7

Table 5.2: Categorization of atomics according to the attack objective.

since the whole ART repository and its execution tool would need to be installed on the VM before executing the atomic test.

5. 19 atomics assume an authentication method with basic credentials, i.e., username and password login mechanism. These 19 atomics fall into the MFA issue categories one and two, as defined in Table A.2. The other 10 atomics either use an interactive (MFA-compatible) login method, or no authentication is required in the attack scenario at all, for example, “Azure - Enumerate Azure Blobs with MicroBurst”.

The focus of this list was on external observations according to certain categories of interest. By examining the content and objectives of the atomics, the following categories were identified. Table 5.2 presents each category, along with the number of atomics associated with the category. Note that all atomics are in exactly one category. This observation is the foundation for the selection process in Section 7.1 and anticipates that the cloud-related atomic tests are only moderately well-suited for performing comprehensive security tests, as the attack operations primarily involve basic user and resource management operations, login attempts, and reconnaissance operations.

In Figure 5.1 there was no granular information on the number of existing atomics per technique. Figure 5.3 shows an additional illustration of the atomics with information on the number of tests represented by the intensity of the red color. We observe that most techniques that are within the test coverage do contain either one or two atomics. One

technique has five corresponding atomics. Concluding, when we take into consideration the low number (29) of Azure-related atomics, the moderate quality of atomics, having two duplicate atomics, 16 atomics that use deprecated PowerShell modules, 19 atomics work with an outdated authentication method and the fact that many atomics only simulate basic procedures like adding or deleting a user, the final assessment on the quality of existing atomics is considered at most moderate.

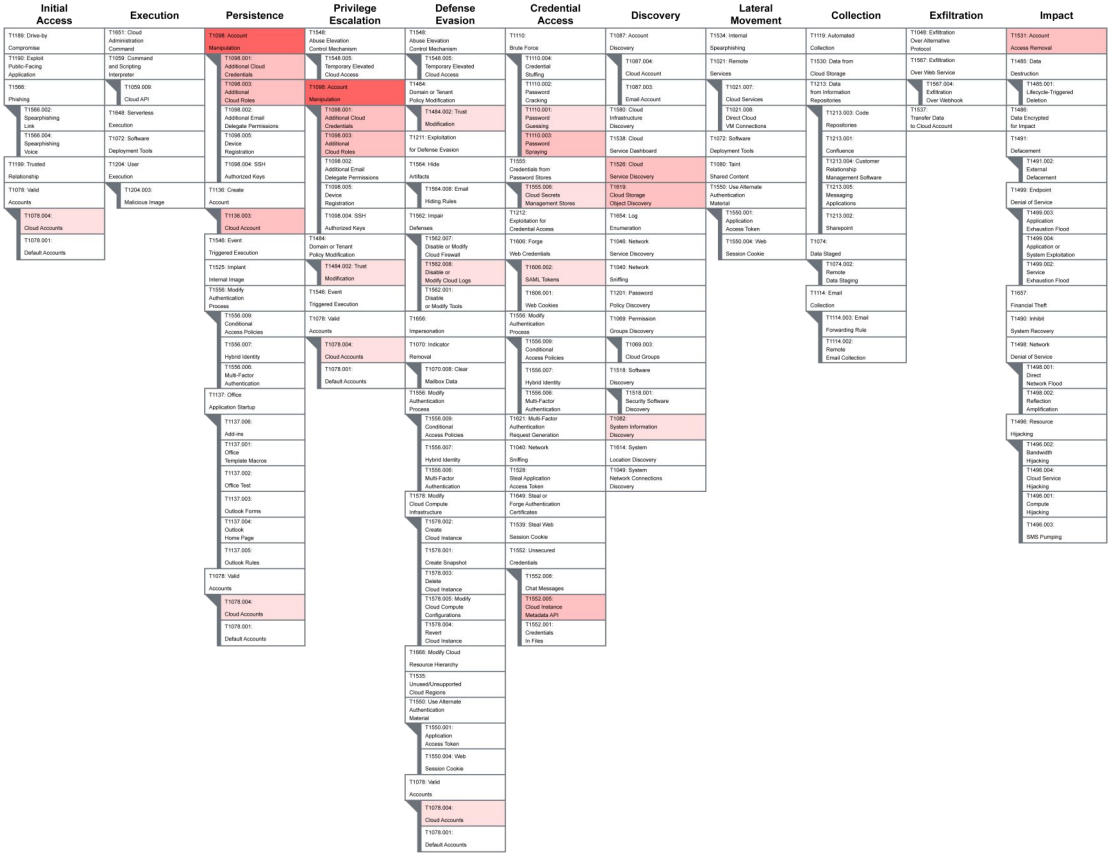

Figure 5.3: Visualization of techniques that are mentioned in atomic tests. Darker red correlates to a higher number of atomics per technique. The lightest red corresponds to one existing atomic test. The darkest red corresponds to five atomics.

As presented in Section 5.1, the test coverage reveals that most of the attack techniques in the top three threat categories are not covered by any atomic test. When we now remove the deprecated and duplicate atomics from the representation, the percentage of techniques that aren't covered at all (inverse test coverage metric) increases from 89.74 % to 97.44 %. This additional representation is visualized in Figure E.1 of Appendix E.

CHAPTER 6

Development of Micro-Attacks

Due to the investigation of ART and the conclusion that there is still much to be done within ART to increase the test coverage in relation to the top three cloud threat categories, the objective of this thesis was to develop new micro-attacks in this regard. Malicious behavior and attacks that appear in the wild are designated as incidents. Therefore, documentations of incidents were considered as a suitable source for developing micro-attacks for this thesis. Imitating the steps involved in incidents is precisely what is intended by micro-attacks, so that security teams can consider them for red team actions. The collection and analysis of incidents is, therefore, the prerequisite task for subsequent development.

This section refers to design cycle three, which defines the design problem to the development of new micro-attacks to increase the test coverage of ART in the context of the top three threat categories. The design process, as introduced in Section 3.2, consists of four steps. Section 6.1 refers to step one, which searches for suitable attack scenarios to be realized. Section 6.1.4 corresponds to step two and identifies one attack scenario from the incident collection. The chosen scenario is explained in Section 6.2. Step three of the design process is presented in the security testing chapter and is presented in Section 7.2. The decomposition of the attack scenario into micro-attacks is expounded in Section 6.3.

6.1 Incidents Collection Process

The collection of incidents was performed using an SLR, employing a simplified Kitchenham [39] methodology. It represents the empirical cycle (3a) in the design science methodology and was used to research suitable attack scenarios that can be realized as micro-attacks. The review planning phase is presented in Section 6.1.1. The report selection process is explained in Section 6.1.2, followed by the data extraction phase in Section 6.1.3. Finally, in Section 6.1.4, the collected data is analyzed and a suitable attack scenario is identified.

6.1.1 Planning the Review

(1) Identification of the Need for a Review:

The collection of incidents is defined as the first step in the design process of cycle three. The SLR is required to find suitable attack scenarios to improve the test coverage of the ART repository. Although other procedures may also be suitable for realizing new micro-attacks, the systematic identification and collection of incidents from threat intelligence reports was considered the most promising. Other approaches are mentioned in the future work Section 9.5. Another research field that deals with the collection and analysis of incidents is incident prediction research, such as the publications by Sun et al. [50] and Carriegos et al. [21]. Sun et al. [50] identifies, besides various datasets, organizational reports as a source that can be used for incident prediction. Organizational reports are chosen as the source type for collecting incidents in this thesis. Threat intelligence reports are assumed to contain the most detailed information about attacks.

(2) Specifying the Research Question:

The research question of the SLR is presented by the knowledge question of this empirical cycle, i.e., "Which attack scenarios that are currently known fall within the top three threat categories for cloud workloads, and provide sufficient detail to be realized as micro-attacks in the ART framework for an Azure cloud environment?"

(3) Developing a Review Protocol:

The review protocol[37] defines the steps for conducting the SLR and was approved by the supervisor of this thesis. It is very similar to the process for retrieving reports in Section 4.1.3 within the SLR for the threat landscape.

6.1.2 Report Selection

The report selection process consists of searching potentially relevant organizations (1), filtering for relevant organizations (2), finding potentially relevant reports (3) within those relevant organizations, and filtering for relevant reports (4).

The list of potentially relevant organizations (1) is built from the set of organizations identified in the threat landscape SLR, and also results from a Google search, from which a selection was made in discussion with a security expert from the collaborating company of this thesis. The search query is defined in Listing 6.1.

[37] https://github.com/manuelkuss/Investigation-of-Cloud-Workload-Security/blob/ab226cdb054e3e3629aafde1cbf8afab20d54236/Collection-of-Incidents/Systematic-Literature-Review/SLR_collect_incidents_review_protocol.md

```
("cloud security report" OR "cloud security risks" OR "cloud security incidents" OR "
    threat intelligence report" OR "threat intelligence" OR "attack intelligence" OR "
    attack intelligence report" OR "misconfigurations intelligence" OR "
    misconfigurations report" OR "vulnerability intelligence" OR "vulnerability report
    ")
AND
("Azure")
AND
("2022" OR "2023" OR "2024")
```

Listing 6.1: Search query for SLR to find relevant organizational reports.

The criteria for relevant organizations (2) are, firstly, that the organizations must have published reports that deal with cloud security incidents. Typically, these reports are designated threat intelligence reports. Second, the publication date must lie between 2022 and 2024, which increases the probability of potentially relevant information. The set of potentially relevant reports (3) is determined to comprise all reports from the relevant organizations that comply with the previously presented criteria. Finally, relevant reports (4) must contain information on the research question. They were filtered out by the following criteria:

- The report contains incidents that consist of attack steps that fall within the top three threat categories, as defined in Section 4.3.
- The report contains incidents with detailed information into the involved steps.
- The report contains incidents that took place in or are related to the Azure cloud.

The resulting reports are listed below:

- Orca Security: State of Cloud Security Report: Uncovering what is lurking in the depths of cloud environments (2024) [51]
- Cloud Security Alliance: Top Threats to Cloud Computing 2024 (2024) [2]
- CrowdStrike: 2023 Cloud Risk Report: The Rise of the Cloud-Conscious Adversary (2023) [46]

6.1.3 Data Extraction

Data extraction is the procedure of collecting incidents and extracting their corresponding information from the three selected reports. The information categories of interest are defined by the data extraction form, which is defined in the review protocol[37]. The form defines fields in the following groups: Incident information, source information, MITRE techniques, optional vulnerability information, further details, and assisting fields for the selection process to find a suitable attack scenario. The fields were populated in the data extraction procedure until a field was identified as incompatible for a realization. An example is an incident that depends on a vulnerability, whose cause has been fixed in

a patch. In this case, the field *realization_estimation* is set to "none", and no further fields are populated. Recall that the objective of the SLR is to identify a suitable attack scenario and not to provide an overview of incidents. The resulting data contains 44 incidents, out of which 30 were mentioned in direct relation to Azure. 34 incidents originate from the Orca Security report [51], 16 from the CSA report [2], and 3 from the CrowdStrike report [46].

6.1.4 Identify a suitable Attack Scenario

The identification of a suitable attack scenario is done by filtering for specific categories. Considering that the CSP of interest for this thesis is Azure, only incidents that are mentioned in relation to Azure are potential candidates. Furthermore, the *realization_estimation* field further narrows down the number of candidates by considering only medium and high-estimated incidents. Two incidents result from this process, one of which has the value high for the *realization_estimation*. This incident is a case report from Orca Security [51], i.e., "Azure Storage Account Keys Exploitation". In a more detailed investigation and review of its feasibility, this attack scenario was chosen to develop new micro-attacks. The attack is explained in Section 6.2.

6.2 Privilege Escalation Scenario

This section presents the attack scenario, which was considered the most promising of all incidents collected in the preceding SLR regarding the realization of new micro-attacks. The scenario was documented in the 2024 cloud security report from Orca Security [51], and published by Nisimi [52] in April 2023. As presented by Nisimi[52], the circumstances that allow the attack to happen are not regarded by Microsoft as a vulnerability, but are rather declared as a by-design flaw. This means that implementing a solution is not considered possible or practical. Design decisions on behalf of a CSP may enable certain attack paths and, as a consequence, pose a security risk. Realizing such scenarios in the form of atomic tests, which test an infrastructure's security posture in the presence of possibly existing misconfigurations, contributes to securing infrastructure against the threat landscape, which is the objective of engineering cycle one.

The overall idea of the attack is that a user with Storage Account privileges manipulates the code of a Function App, and as a consequence, exfiltrates the access token of that Function App. The user has thereby escalated the privileges. Further attacks on other resources may follow as a result of the increased privileges. Before examining the attack steps in more detail, some necessary background information on key components within the Azure cloud needs to be considered:

- **Storage Accounts:** Storage Accounts are entities that contain data objects, such as Blobs and Files. They form the superordinate entity of storage objects and

provide a unique namespace for the data to be accessible as cloud resources[38].

- **Shared Key Authorization:** There are two authorization methods to access (non-anonymously accessible) storage, i.e., Entra ID authorization, and Shared Key Authorization. Shared Key Authorization offers a token-based authorization method[39], which is the mechanism behind Storage Account Access Keys.

- **Storage Account Access Keys:** These keys are used via Shared Key Authorization to authorize access requests to Storage Accounts. Microsoft elucidates that, in the process of creating a Storage Account, a Shared Account Access Key is created automatically. Microsoft recommends realizing access with least privilege and notes the possibility to disallow Shared Key Authorization for a Storage Account[40]. Microsoft recommends managing those keys in a Key Vault[41]. Although Shared Key Authorization is enabled by default for Storage Accounts, it is stated that optimal security can be reached by using the Entra ID option for authorization instead of shared keys. This is a perilous contradiction, as will become clear in this section.

- **Azure Functions:** Azure Functions is a PaaS provider to realize serverless, event-driven computing resources[42].

- **Function App:** The Function App is a cloud component in which Azure Functions are deployed[43].

- **Relationship between Function Apps and Storage Accounts:** When creating a Function App, a corresponding Storage Account is created automatically. The Functions' source code, as well as configuration files, are placed in this Storage Account.

- **Managed Identities:** Managed Identities are a concept to eliminate the management of secrets, certificates, and keys on behalf of developers. Managed Identities provide tokens to Azure applications, and thereby can be seen as workload identities, similar to identities of users, but for resources. Managed Identities provide access to other resources[44].

[38] https://learn.microsoft.com/en-us/azure/storage/common/storage-account-overview. Accessed: May 20, 2025.

[39] https://learn.microsoft.com/en-us/rest/api/storageservices/authorize-with-shared-key. Accessed: May 26, 2025.

[40] https://learn.microsoft.com/en-us/azure/storage/common/storage-account-keys-manage?tabs=azure-portal. Accessed: May 20, 2025.

[41] https://learn.microsoft.com/en-us/rest/api/storageservices/authorize-with-shared-key. Accessed: may 20, 2025.

[42] https://learn.microsoft.com/en-us/azure/azure-functions/. Accessed: May 20, 2025.

[43] https://learn.microsoft.com/en-us/azure/azure-functions/functions-deployment-technologies. Accessed: May 20, 2025.

[44] https://learn.microsoft.com/en-us/entra/identity/managed-identities-azure-resources/overview#what-are-managed-identities. Accessed: May 26, 2025.

A schematic view of the cloud infrastructure for the attack scenario is presented in Figure 6.1. The Entra ID roles of interest comprise two built-in roles: *Storage Account Contributor* is a role that provides access to Storage Accounts, and *Virtual Machine Contributor* provides access to Virtual Machines. A user is assigned to the eligible role *Storage Account Contributor*. The Function App's Managed Identity is configured with the permanently active role *Virtual Machine Contributor*. In the subscription, there are three main types of resources, i.e., Virtual Machines, a Function App, and Storage Accounts. The Function App contains a Python Function and is related to three resources, i.e., a Storage Account, an Application Insights instance, and an App Service Plan. All resources lie within one Resource Group. Furthermore, participating entities are the Azure API Management Service and a user who becomes the attacker in the scenario.

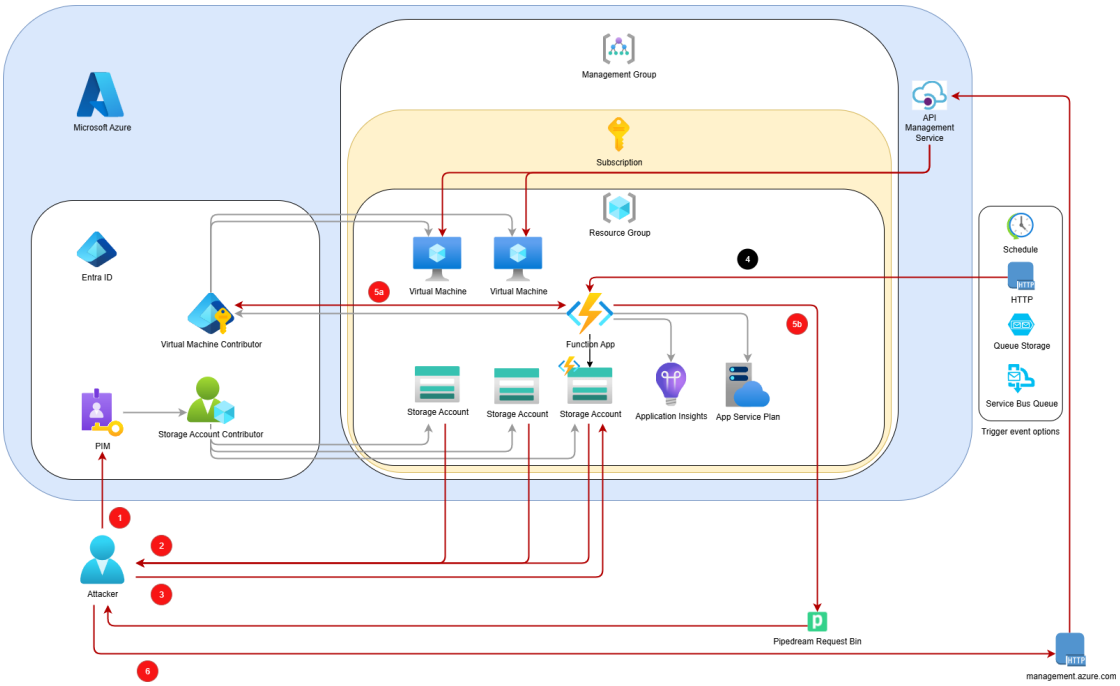

Figure 6.1: Schematic view of the cloud components related to the privilege escalation attack scenario, including the malicious insider user. Numbered circles represent the attack steps. Red-colored attack steps are triggered by the attacker. The black-colored attack step is triggered in the regular operation of the Function in its context.

The Function App, which is the key component in this cloud setup, has been configured with a system-assigned Managed Identity, which is given access to Virtual Machines via the role *Virtual Machine Contributor*. The reason for this permission is that the serverless Function within the Function App performs monitoring and management tasks for Virtual Machines. Several scenarios are imaginable for serverless programs that require permissions to other cloud resources. Another example would be automatic

operations on databases, e.g., cleanup, backup, or scaling operations, that run regularly[45].

Note that the user has access only to Storage Accounts. When the user tries to access other resources, such as the VM overview page in the Azure Portal, an empty page is presented. In the following, all steps within the attack scenario are explained. The steps are visualized in Figure 6.1 as numbered circles, and the corresponding chain of ATT&CK techniques is presented in Figure 6.2.

- **Step 1 (T1078.004):** First, the attacker (a malicious insider) activates the role *Storage Account Contributor* in the PIM service.
- **Step 2 (T1619):** In a reconnaissance step, the attacker searches through the Storage Accounts and the nested Storage Account objects. Their goal is to identify objects that are related to Functions for which there exist two indicators. First, a Storage Account may be named in a way that indicates the purpose of the associated Azure Function. Storage Accounts related to a Function are often named similarly. Second, the folder structure and existing files further indicate the relationship to Functions. One example is the configuration file 'host.json', which is located in the path 'site/wwwroot/host.json'. Therefore, all function-related Storage Accounts can be identified.
- **Step 2 (T1530):** Subsequently, the attacker downloads all function-related storage objects to get more insights into the Function they want to utilize. One file is the source code, which is of particular interest to the attacker.
- **Step 3 (T1528):** In step three, the attacker modifies the Function's source code and thereby integrates the functionality of step five. The original code should also be left in place, to ensure consistent execution of the Function to remain unnoticed. The injected code is presented in Listing 6.2.
- **Step 4:** Step four is the trigger event that invokes the Function. Functions can be invoked in a variety of ways, including time-scheduled triggers, HTTP triggers, Queue triggers, and those executed on Events via the Event Grid[46]. For the sake of demonstration purposes, the Function is configured with an HTTP trigger.
- **Step 5:** Step five represents the execution of the Function, and thereby the injected code, as presented in Listing 6.2. In (5a), the Function performs a request to retrieve its own Managed Identity access token. This is done using the two environment variables *IDENTITY_ENDPOINT* and *IDENTITY_HEADER*. The result is then sent in (5b) via a POST request to a specified endpoint, e.g., a hosted pipedream[47] instance, which the attacker monitors. The exemplary body of the received POST request is presented in Listing 6.3.

[45] https://learn.microsoft.com/en-us/azure/azure-functions/functions-scenarios?pivots=programming-language-csharp#run-scheduled-tasks. Accessed: May 26, 2025.

[46] https://learn.microsoft.com/en-us/azure/azure-functions/functions-triggers-bindings?tabs=isolated-process%2Cnode-v4%2Cpython-v2&pivots=programming-language-csharp. Accessed: May 20, 2025.

[47] https://pipedream.com/. Accessed: May 20, 2025.

- **Step 6 (T1550.001)** Finally, the token can be used to perform various actions on Virtual Machines. For demonstration purposes, in step six, the attacker lists all existing VMs via the API Management service and thereby performs a reconnaissance step using the escalated privileges. In the infrastructure depicted in Figure 6.1, the response would be a JSON object that contains two VM objects with fields such as name, id, location, others.

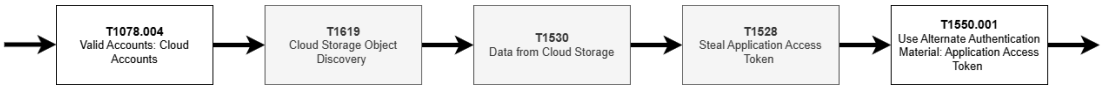

Figure 6.2: The chain of the attack techniques of the privilege escalation attack scenario. Grey-colored techniques represent micro-attacks that were developed as part of this thesis

With these six steps, the attacker managed to perform actions within the cloud infrastructure that would not be possible with their normal permissions. Nisimi [52] goes one step further in the attack scenario and uses the access token to move laterally by injecting a reverse shell on one of the listed VMs. This thesis doesn't cover such additional steps.

```
1 import os, subprocess
2 resp = subprocess.getoutput(f"curl -s \"{os.getenv('IDENTITY_ENDPOINT
    ')}/?resource=https://management.azure.com/&api-version
    =2019-08-01\" -H \"X-IDENTITY-HEADER: {os.getenv('IDENTITY_HEADER
    ')}\"")
3 subprocess.call(f"curl -s -X POST -d \"{resp}\" https://changeme.net
    ", shell=True)
```

Listing 6.2: Python code to retrieve the token of the Function App that executes the code, and subsequently sends a POST request with the token to a defined endpoint, which is represented by the string *https* : *//changeme.net*. Both curl requests are defined with $-s$ to operate in silent mode, which suppresses any progress information and error messages. The parameter $-X$ specifies the HTTP request method, $-H$ specifies extra headers, and $-d$ specifies the body data for the request.

```
1 {
2         access_token: eyJ0eXAiOiJKV1QiLCJ...,
3         resource: https://management.azure.com/,
4         token_type: Bearer,
5         client_id: ...
6 }
```

Listing 6.3: Response body of the POST request, as it would be observed on the attacker-controlled endpoint. It does contain the access_token.

6.2.1 Underlying Causes of the Attack

In the scenario, the attacker has access to Storage Accounts via the *Storage Account Contributor* role. This is bad enough, but escalating roles to privileges that grant access to additional resources greatly increases the possibilities of harm for an organization. This section discusses the reasons why the attack is possible.

There are three infrastructure components that enable the privilege escalation path. First, the concept of privileges to access Storage Accounts, second, the access option via shared access keys, and third, the increased privileges of a Function App to access other cloud resources. In the following, each cause is explained. Mitigation ideas are presented in the subsequent Section 6.2.2.

Privileges Structure in Azure:

Before discussing the circumstances of roles and privileges that enable the attack scenario, it is worth noting that the Azure cloud, like other cloud environments, is rapidly changing. To get an overview of all existing roles and their privileges is therefore not a one-time, final task. A periodic evaluation of the context of roles and privileges is required. This thesis reasons about the status of the Azure cloud between Jan. 2025 and May 2025..

An important concept for understanding operations within Azure is the division of control plane and data plane operations[48]. Control-plane operations are used to manage resources. Data-plane operations utilize the capabilities of the corresponding resource type. Considering Storage Accounts, data-plane operations are read and write access operations on data objects within a Storage Account, in contrast to control-plane operations, which are management operations, such as creating or deleting Storage Accounts, changing encryption settings, or configuring access keys. As Nisimi [52] pointed out, this leads to a misconception of how permissions are separated. The action "Microsoft.Storage/storageAccounts/listKeys/action" (referred to in this thesis as listkeys-action) is described as a sneaky anomaly of a control-plane operation, as it grants full access to data operations. This confusing fact may lead to misconfigurations, where, for example, the role *Storage Account Contributor* is expected to only have contributor permissions at a control plane level, but instead, through the listkeys-action, full access to all storage objects is granted when shared access keys are used. Note that Shared Key Authorization is enabled by default. More information on shared access keys follows later. The confusion continues when considering the categorization types for role definitions of control-plane and data-plane operations, i.e., the fields "Actions", "NotAction", "DataActions", "NotDataActions". Control plane actions are specified in "Actions" and "NotActions"[49]. For the role *Storage Account Contributor*, only control-plane actions in "Actions" are given[50]. The fields "NotActions", "DataActions", and "NotDataActions"

[48] https://learn.microsoft.com/en-us/azure/azure-resource-manager/management/control-plane-and-data-planeMicrosoft Documentation on control-plane-and-data-plane. Accessed: May 21, 2025.

[49] https://learn.microsoft.com/en-us/azure/role-based-access-control/role-definitions. Accessed: May 21, 2025.

[50] https://learn.microsoft.com/en-us/azure/role-based-access-control/built-in-roles/storage. Accessed: May 21, 2025.

are empty. Therefore, the role *Storage Account Contributor* may be considered on behalf of the CSC for control-plane access purposes, and at the same time, overlook that the listkeys-action provides full access to all Storage Accounts.

Considering the evolving environment of Azure, the previously presented confusing concepts of control-plane and data-plane actions make the design of secure IAM concepts complicated. Hence, the privilege structure in Azure is one aspect that enables such a misconfigured setup, which in turn enables the possibility of similar privilege escalation attacks. In the following, the privileges structure of Azure will be examined by looking for a solution for the case of read-only access to Storage Accounts. This thesis focuses on built-in roles, more precisely, built-in roles for storage access[50]. Table 6.1 presents all Storage-related roles, four of which contain the listkeys-action. As it is pointed out by Nisimi [52], the scenario of achieving read-only access to Storage Accounts is no straightforward task. First, there is no such role as *Storage Account Reader* in the list of built-in roles from Table 6.1. The *Reader*[51] role would be a suitable substitute, but then not only Storage Accounts can be viewed, but also other resources. The roles *Storage Blob Data Reader*, *Storage File Data Privileged Reader*, and *Storage File Data SMB Share Reader* (and their variations with Contributor instead of Reader) on their own are not sufficient to achieve read access to data. The reason is that the navigation to storage objects through the Azure Portal requires greater read permissions[52]. As a consequence, most CSCs would likely use the roles *Storage Account Contributor* or *Reader and Data Access* for the intended privileges. However, both have the listkeys-action, which provides full access to all Storage Accounts. This situation leads to a by-design flaw in which data access seems to be configurable only along with the listkeys-action. Although Microsoft stated that it would improve the Azure Portal access for *Storage Blob Data Reader* and *Storage File Data SMB Share Reader* [52], it wasn't possible at the time of writing this thesis. The risk of misconfigurations in this regard, therefore, persists.

Shared Key Authorization:

As mentioned before, Shared Key Authorization is an access option to Storage Account objects, and the listkeys-action grants access to these keys. All roles in Table 6.1 that have the listkeys-action are therefore granted access with full control over the Storage Account objects. From the preceding reasoning, it becomes clear that fine-granular read access to Storage Accounts is not easily possible without the listkeys-action.

Managed Identities for Function Apps:

The third reason for the privilege escalation path is the concept of Managed Identities for Function Apps. Due to the relationship between Function Apps and Storage Accounts, access to Storage Accounts along with a code injection scenario may result in a privilege escalation. Thereby, the Function App's Managed Identity can be exploited for operations with higher privileges.

[51] https://learn.microsoft.com/en-us/azure/role-based-access-control/built-in-roles/general#reader. Accessed. May 27, 2025

[52] https://learn.microsoft.com/en-us/azure/storage/blobs/assign-azure-role-data-access?tabs=portal#assign-an-azure-role. Accessed: May 27, 2025.

Role	listkeys-action
Classic Storage Account Contributor	✓
Classic Storage Account Key Operator Service Role	✓
Reader and Data Access	✓
Storage Account Backup Contributor	
Storage Account Backup Contributor	
Storage Account Contributor	✓
Storage Account Key Operator Service Role	✓
Storage Blob Data Contributor	
Storage Blob Data Owner	
Storage Blob Data Reader	
Storage Blob Delegator	
Storage File Data Privileged Contributor	
Storage File Data Privileged Reader	
Storage File Data SMB Share Contributor	
Storage File Data SMB Share Elevated Contributor	
Storage File Data SMB Share Reader	
Storage Queue Data Contributor	
Storage Queue Data Message Processor	
Storage Queue Data Message Sender	
Storage Queue Data Reader	
Storage Table Data Contributor	
Storage Table Data Reader	

Table 6.1: List of built-in roles for Storage access[50]. Roles related to Avere, Backup, Data Box, Data Lake, Defender for Storage Data Scanner, and Elastic SAN are not included. "listkeys-action" stands for either "Microsoft.ClassicStorage/storageAccounts/listkeys/action", or "Microsoft.Storage/storageAccounts/listkeys/action".

6.2.2 Mitigation Ideas

This section discusses three mitigation options for the presented privilege escalation scenario. In addition to the provided options, and in the case that all those suggestions are not applicable, a threat detection setup to detect suspicious behavior may assist in securing the system. A suitable threat detection setup is presented in Chapter 8.

Option 1: The best way to mitigate this risk is by following the least-privilege principle. This means for storage access, to assign only the least possible privileges for employees who need to access and operate in Storage Accounts. As outlined in the previous section, assigning suitable permissions that don't include the listkeys-action may resolve the mentioned security risk. Additionally, cloud resources may be organized in different subscriptions (or resource groups) to be able to assign *Reader*[51] privileges on a smaller scope. The restriction of granting permissions to cloud storage objects is also listed by MITRE (M1018) as a mitigation suggestion in relation to T1619[53].

[53] https://attack.mitre.org/techniques/T1619/. Accessed: May 23, 2025.

Option 2: As presented by Nisimi [52], disabling Shared Key Authorization may resolve the problem of too many privileges as well, since shared keys enable full Storage Account access. For example, examining the role *Storage Account Contributor*, it is limited to control-plane level operations if there is no shared key access. Modifications of data would then require additionally granted privileges. This suggestion is also included in the built-in Azure policy: "Storage accounts should prevent shared key access"[54]. When this policy is enabled, DfC will recommend to disable shared-key access. At this point, it should be noted that numerous recommendations originate from policies and often remain under the radar for organizations. One reason is that many policies are too general to consider for the specific infrastructure setup of an organization, and therefore don't make sense. This is particularly true if third-party products are used along with Microsoft-Azure's own products, e.g., using an external firewall vendor. Policies cannot take these different setups into account. Here, the rapidly evolving structure of Azure should be noted again. The set of policies changes often, which results in a high effort for organizations to keep up with the current status. Therefore, despite the fact that a policy recommends to disable Shared Key Authorization, the possibility of this misconfiguration is present. CSA [2] even mentions inconsistent policies as an example of IAM security issues that make security efforts more complicated, and therefore underlines the stated issue with policies.

Option 3: For the third mitigation option, a scenario is assumed in which shared key access is enabled along with active role assignments that include the listkeys-action permissions. In this case, all users that are assigned to this role should be clearly identified as if they have the same privileges as the Function App's Managed Identity, for Function Apps that have their corresponding Storage Account in the same resource scope. Considering that Shared Key Authorization can be configured on a Storage Account level, one further mitigation possibility would therefore be to split resources that should be in different authorization groups (e.g., storage resources and a Function App with access to further resources) into multiple different resource areas (e.g., subscriptions or resource groups). This makes it possible to define more granular levels of access to the various areas[55].

Other Mitigation Ideas from MITRE:

Further mitigation suggestions are listed by the techniques associated with the atomics from Section 6.3. For T1530[56] and T1528[57], the following mitigation ideas are considered relevant for the attack scenario. They provide general suggestions, which underline the reasoning presented before.

- (M1047) Audit: This idea motivates frequently performed checks of permission concepts.

[54] https://learn.microsoft.com/en-us/azure/storage/common/policy-reference. Accessed: May 21, 2025.
[55] https://learn.microsoft.com/en-us/azure/role-based-access-control/scope-overview. Accessed: May 26, 2025.
[56] https://attack.mitre.org/techniques/T1530/. Accessed: May 23, 2025.
[57] https://attack.mitre.org/techniques/T1528/. Accessed: May 28, 2025.

- (M1018) User Account Management: Suggests a proper configuration of IAM practices.

6.3 New Atomics

This section presents the micro-attacks developed to implement the privilege escalation scenario from the preceding section. Recalling the steps to retrieve the access token of a Function App, as visualized in Figure 6.1, the attack consists of (1) activating storage privileges, (2) reconnaissance and collection of Storage Account objects, and (3) code injection. Subsequent operations with the access token are part of a subsequent attack scenario and are therefore not included in the steps that realize the privilege escalation scenario.

Step (1) is taken as a prerequisite condition by assuming a malicious insider. Step (2) consists of two actions, which are realized in one atomic test each (ATOMIC_9, ATOMIC_10). Step (3) has two execution options, depending on the Function App's storage realization, and has been realized in two separate atomic tests (ATOMIC_11, ATOMIC_12). Therefore, three out of four atomics are necessary for a successful attack execution, i.e., ATOMIC_9, followed by ATOMIC_10, followed by either ATOMIC_11 or ATOMIC_12.

Before going into the individual tests, the commonalities should be considered. First, the atomics use the Azure CLI to interact with the Azure environment. Second, all tests are intended to run in interactive execution mode, which is an execution option that allows for providing input during the test execution. This is necessary for the login process, which leads to the next commonality, i.e., the usage of the interactive version of the Azure CLI login command, which invokes a pop-up window for the MFA procedure. The CUT, as presented in Section 7.2, considers MFA as best practice. Therefore, the attack steps are developed accordingly. Furthermore, all atomics contain cleanup commands that can be executed to reverse the changes afterwards. This is especially important for the atomics ATOMIC_11 and ATOMIC_12, since they go beyond read access by modifying a running Azure Function.

Figure E.2 in Appendix E illustrates the impact of these four new atomics on the test coverage. Two techniques (T1528 and T1530) were colored purple, i.e., through the addition of ATOMIC_10, ATOMIC_11, and ATOMIC_12. Technique T1619 remains colored red, since there already exist other atomics in this technique, and the technique is not part of the top three threat categories. The newly introduced atomics increased the test coverage by 5.13 %, i.e., from 10.26 % to 15.38 %.

ATOMIC_9 (T1619): Azure - Enumerate Storage Account Objects via Shared Key authorization using Azure CLI[58]:

Technique T1619[53] is defined as "Cloud Storage Object Discovery", in which adversaries enumerate cloud storage objects. The development of a new atomic for the enumeration of Storage Account objects via Shared Key Authorization was necessary, since no existing atomic sufficiently comprises this objective. One similar atomic exists, i.e., "Azure - Dump Subscription Data with MicroBurst", which realizes the cloud service discovery technique T1526[59]. This atomic dumps existing cloud resources into an output folder, which represents the structure of the existing cloud resources. Resource instances are listed in CSV files within the folder structure. The output also contains information about existing storage objects within Storage Accounts. Therefore, it implements a similar enumeration operation, and the test (T1526) can be seen as a superordinate technique compared to T1619. However, there are reasons to realize T1619 in addition to T1526. First, an attack step that enumerates many more objects than needed leaves more traces on the targeted organization. Attackers intend to operate stealthily, which motivates a more detailed breakdown into multiple atomics. Second, the access method via shared keys, which is one of the assumed infrastructure components that make the attack scenario possible, is distinct. Hence, this new atomic test has been developed. The new atomic fetches all existing Storage Accounts, and, if shared key access is enabled, enumerates all Storage Account objects, i.e., File Shares, Files in File Shares, Containers, Blobs within Containers, Tables, and Queues. The result is stored in a CSV file, which serves as input to the subsequent atomic, which is explained in the following.

ATOMIC_10 (T1530): Azure - Dump Azure Storage Account Objects via Azure CLI[60]:

Technique T1530 lies within the MITRE technique "Data from Cloud Storage"[56]. No other existing test realizes the requested objective of downloading storage objects. Hence, the new atomic[60] was developed. ATOMIC_10 takes a CSV file as an input parameter. This may be the output file of the discovered storage objects from ATOMIC_9. On execution, all storage objects defined in the file and accessible via shared key access will be downloaded to a specified output folder.

ATOMIC_11 (T1528): Azure - Functions code upload - Functions code injection via Blob upload (option1)[61]

T1528 refers to the objective "Steal Application Access Token"[57]. As mentioned in the explanation of Technique T1528, a short-lived token retrieved through the Instance Metadata Service (IMDS) can facilitate access to other resources. Depending on the

[58] https://www.atomicredteam.io/atomic-red-team/atomics/T1619#atomic-test-2---azure---enumerate-storage-account-objects-via-shared-key-authorization-using-azure-cli. Accessed: May 23, 2025.

[59] https://attack.mitre.org/techniques/T1526/. Accessed: May 29, 2025.

[60] https://www.atomicredteam.io/atomic-red-team/atomics/T1530#atomic-test-4---azure---dump-azure-storage-account-objects-via-azure-cli. Accessed: May 23, 2025.

[61] https://www.atomicredteam.io/atomic-red-team/atomics/T1528#atomic-test-1---azure---functions-code-upload---functions-code-injection-via-blob-upload. Accessed: Jul. 14, 2025.

storage type of the Function App[62], this technique is realized in two versions, i.e., option 1 (ATOIMC_11) and option 2 (ATOMIC_12). Option 1 deals with Functions that are deployed via Azure Blob storage. In this case, the source code is presented in a Blob object in the form of a ZIP file. The running Function instance gets read-only access to this file and, therefore, runs directly on the package. The code injection step, as presented in step three in Figure 6.1, replaces the whole package file, which causes the running function instance to immediately run the changed function code[63] when it is invoked.

ATOMIC_12 (T1528): Azure - Functions code upload - Functions code injection via File Share modification to retrieve the Functions identity access token (option2)[64]:

This atomic test implements option two from the function code upload scenario. It deals with a Function that is configured using Azure Files, which means that the source code is stored in a File within a File Share. The code upload operation, therefore, slightly differs from option one. The code to insert is specified in the input variable "code_to_insert_path". A generic version of the required code, as presented in Listing 6.2, has been added to the atomic test in the GitHub repository[65]. The atomic test works as follows. First, it is checked whether shared key access is allowed to the specified Storage Account that contains the File Share. If this is true, the original Function's source code file is downloaded as a backup for a later cleanup operation. Third, depending on the execution option, the code injection happens either by downloading the function code, inserting the code, and uploading the tampered file, or by uploading the file directly. In the latter option, the file defined in 'code_to_insert_path' should be prepared to contain all parts of the Function code, not just the code to exfiltrate the access token.

In distinction to option one, the predictability of the time until the code is actually used by the running function instance is worse than for package-deployed Functions[63]. It depends mainly on the hosting plan. For the experiments in this thesis, which used the hosting plan *Consumption*, the code change became active quickly (< 1 min). However, there is no documented guarantee for this rather immediate change. Azure does not officially support a feature of live production changes[66]. Therefore, on some occasions, the attacker may has to wait for a restart or (automatic) scaling event for the change to get active.

[62] https://learn.microsoft.com/en-us/azure/azure-functions/storage-considerations?tabs=azure-cli. Accessed: May 26, 2025.

[63] https://github.com/Azure/app-service-announcements/issues/84. Accessed: May 26, 2025.

[64] https://www.atomicredteam.io/atomic-red-team/atomics/T1528#atomic-test-2---azure---functions-code-upload---functions-code-injection-via-file-share-modification-to-retrieve-the-functions-identity-access-token. Accessed: Jul. 14, 2025.

[65] https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1528/src/code_to_insert.py. Accessed: Jul. 14, 2025.

[66] https://learn.microsoft.com/en-us/azure/azure-functions/functions-scale. Accessed: May 26, 2025.

CHAPTER 7

Security Testing

This section presents the experiments to test the security of Microsoft Azure, in particular the CUT. In concrete terms, the threat detection capabilities of Azure will be tested through conducting penetration tests (atomics) from the ART database. As defined in the design science cycles four and six in Section 3.2, the knowledge questions ask for the detection rate, i.e., the attack coverage of the threat detection system. The security testing serves not only as the elaboration of a security assessment, but also as validation of the artifacts developed in the preceding design cycles, i.e., cycles three and five.

For the security testing, the following steps needed to be done. First, in Section 7.1, a set of atomics needs to be selected from the set of existing atomics. This set represents the adversary actions performed in the penetration tests. Second, the CUT as the object under attack is defined in Section 7.2. Third, in Section 7.3, the experiment setup is defined. The setup contains the positioning of the attacker in relation to the CUT within an enterprise architecture, as well as the concrete experiment plan. Fourth, the experiment results are given in Section 7.4.

7.1 Set of Executed Atomics

From the existing atomics within ART, a selection of atomics needs to be made to define a suitable set of atomics for conducting the penetration tests. The set of potentially relevant atomics was obtained on March 13, 2025 from Atomic Red Team[26] using the search filter “supported_platforms:(iaas:azure or azure-ad)”. This results in a set of all atomics that are related to Azure. The selection of the relevant set of atomics considered various issues that were noticed in the investigation of existing atomics in Section 5. Details on the exclusion criteria that were applied to the list of potentially relevant atomics are presented in Appendix A.

The final set of atomics for the penetration tests is presented in Table 7.1, which comprises both, existing atomics and the four newly developed atomics from Section 6.3. ATOMIC_5 and ATOMIC_6 perform basic user management operations. ATOMIC_7 and ATOMIC_8 perform basic identity management operations within Entra ID. ATOMIC_2 creates an automation runbook and therefore performs a basic resource management operation. Hence, five atomics perform basic operations and are not considered as adversary behavior on their own. However, they are part of the attack set, since this thesis aims to test the suitability of using ART for security tests, and therefore, a minimal size of the attack set is required to demonstrate the risk-based testing approach within this thesis. The five basic atomics are not expected to be detected and, on the contrary, must not result in alerts. ATOMIC_1, ATOMIC_3, ATOMIC_4, and ATOMIC_9 are reconnaissance operations, and ATOMIC_9 to ATOMIC_12 represent the Function-related privilege escalation scenario in two versions.

Technique number	**Atomic name**	**New?**	**ATOMIC_ID**
T1136.003	Azure AD - Create a new user via Azure CLI		ATOMIC_5
T1531	Azure AD - Delete user via Azure CLI		ATOMIC_6
T1526	Azure - Dump Subscription Data with MicroBurst		ATOMIC_1
T1619	Azure - Enumerate Azure Blobs with MicroBurst		ATOMIC_3
T1098	Azure - adding user to Azure role in subscription		ATOMIC_7
T1098	Azure - adding service principal to Azure role in subscription		ATOMIC_8
T1078.004	Azure Persistence Automation Runbook Created or Modified		ATOMIC_2
T1619	Azure - Scan for Anonymous Access to Azure Storage (Powershell)		ATOMIC_4
T1619	Azure - Enumerate Storage Account Objects via Shared Key authorization using Azure CLI	YES	ATOMIC_9
T1530	Azure - Dump Azure Storage Account Objects via Azure CLI	YES	ATOMIC_10
T1528	Azure - Functions code upload - Functions code injection via Blob upload	YES	ATOMIC_11
T1528	Azure - Functions code upload - Functions code injection via File Share modification to retrieve the Functions identity access token	YES	ATOMIC_12

Table 7.1: The set of atomics used in the penetration tests. Each atomic test is labeled with an ATOMIC_ID as a reference within this thesis.

7.2 Cloud Under Test

This section presents the CUT setup that is attacked within the security testing of this thesis. To produce realistic results regarding the overall security of workloads outsourced to the cloud, the architecture of the CUT should follow best practices. There are numerous

security guidelines for securing various cloud components using state-of-the-art security approaches, such as [14]. However, a comprehensive review in this regard would exceed the scope of this thesis. The CUT that follows best practices was achieved in collaboration with a company in the energy sector, which has outsourced cloud workloads in the past few years and has implemented security best practices. By this collaboration, the goal of realizing a realistic and secure CUT for assessing the security of cloud workloads on behalf of the CSC is ensured. The security best practices are outlined along with details on the corresponding cloud components in this section. Note that the naming convention for cloud components that contain information about the collaboration company has been replaced with pseudonyms. The presentation of detailed information on the CUT and its deployment procedure ensures reproducibility.

The architecture of the CUT is schematically visualized in Figure 7.1. On the very top lies the Entra ID component. On the left side is the hierarchy of management groups, on which certain policies are activated. Two subscriptions are relevant for the thesis. *SUB-SOC_Playground2* is the subscription that holds all cloud resources that are targeted in the security testing. The subscription *management-dev-00* holds the Log Analytics workspace, which is the cloud component that receives logs from the diagnostic settings. Among others, diagnostic settings from Entra ID and from the *SUB-SOC_Playground2* subscription are configured to send logs to this workspace. More details on the log setup are explained in Section 7.2.4. At the very bottom are the security components of Azure. XDR holds multiple Defender instances, such as *Microsoft Defender for Identity*, which is the Defender of Entra ID. Each subscription has its own DfC entity, which is connected to *XDR*. *Sentinel* and *XDR* are connected, which allows Sentinel to handle all alerts within the CUT. *Sentinel* is the SIEM solution that receives logs from the Log Analytics Workspace and *XDR*.

The CUT comprises the whole cloud setup on behalf of the CSC, which mainly consists of (1) the Entra ID setup, (2) the setup of cloud resources, and (3) the security measures. In order to perform the security testing experiments, the necessary CUT components were derived from the final set of atomics for the experiment execution, as defined in Section 7.1. Table 7.2 lists the required privileges for the execution of each atomic test in the final set. Aside from this information, the required cloud component, as the target object of the respective atomic test, is also listed for the Entra ID setup.

The terminology and the architectural components within Azure are explained in Section 2.2. The Tenant and therefore also the Entra ID instance were provided by the collaboration company of this thesis. Note that the Tenant is used solely for testing purposes and is therefore separated from a production system. The subscription *SUB-SOC_Playground2* was created to realize the cloud resources of the CUT. It is depicted in Figure 7.1 on the right side. Recall that a subscription is a logical container that holds the cloud resources and represents a billing component. The subscription and the therein contained cloud workloads, which were realized in the CUT, are presented in Section 7.2.2, and the deployment method is explained in Section 7.2.3. The Entra ID setup is presented in Section 7.2.1, and the security setup is presented in Section 7.2.4. The log infrastructure

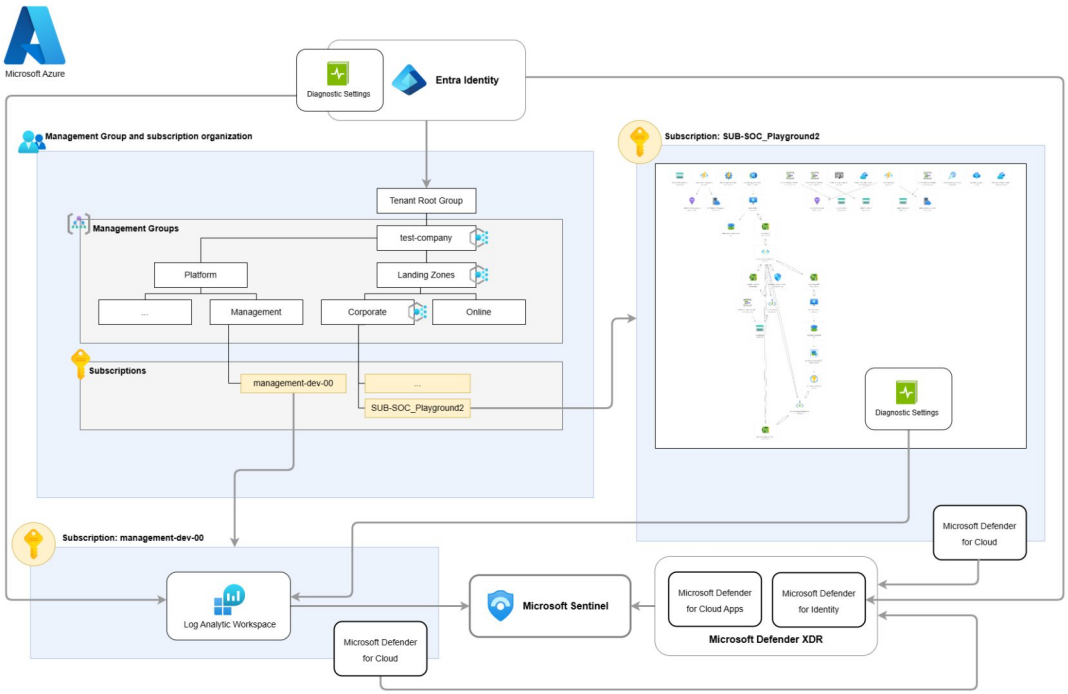

Figure 7.1: Schematic overview of the CUT setup. The connection between Entra ID, the hierarchy of management groups, and the existing subscriptions therein are depicted. Furthermore, the two relevant subscriptions for the thesis are schematically visualized, and the necessary resources for the log infrastructure are displayed. In the subscription *management-dev-00*, the relevant resource is a Log Analytics workspace, and in *SUB-SOC_Playground2*, the relevant resources are the diagnostic settings. The utilized security services and the connections to Sentinel are shown at the very bottom.

is another critical component of the CUT. The explanation of this architecture part is presented in Section 7.2.4.

7.2.1 Entra ID Setup

The Entra ID setup consists of users,conditional access rules, groups, and roles.

Users: Two users exist in the Entra ID setup, i.e., user A (userA@domain.at), and user B (userB@domain.at). User A is the malicious user who executes the atomic tests. User B is a normal user who is the target object of ATOMIC_7. The best practice to grant privileges to a user in Entra ID requires the usage of so-called eligible roles and eligible groups. Users can activate an eligible role (or group) for a limited time. Another best practice is the principle of least privilege. To meet both of these best practices, the setup was designed in a way to have the malicious user registered for several eligible roles (and groups) that grant only the least possible privileges required for the according

Atomic_ID	Technique number	Atomic Name	Privilege for execution	CUT component
ATOMIC_1	T1526	Azure - Dump Subscription Data with MicroBurst	del.az.ART-Tests-SOC-Playground2	At least some resources to dump information
ATOMIC_2	T1078.004	Azure Persistence Automation Runbook Created or Modified	del.az.ART-Tests-SOC-Playground2	Automation account
ATOMIC_3	T1619	Azure - Enumerate Azure Blobs with MicroBurst	None	Storage accounts with objects
ATOMIC_4	T1619	Azure - Scan for Anonymous Access to Azure Storage (Powershell)	None	At least one publicly accessible blob
ATOMIC_5	T1136.003	Azure AD - Create a new user via Azure CLI	del.az.ART-Tests-Entra	None
ATOMIC_6	T1531	Azure AD - Delete user via Azure CLI	del.az.ART-Tests-Entra	Existing user (e.g., created from execution of ATOMIC_5)
ATOMIC_7	T1098	Azure - adding user to Azure role in subscription	Owner	Existing user
ATOMIC_8	T1098	Azure - adding service principal to Azure role in subscription	Owner	Service principal (e.g., Existing Function App)
ATOMIC_9	T1619	Azure - Enumerate Storage Account Objects via Key-based authentication using Azure CLI	Storage Account Contributor	Storage accounts with objects
ATOMIC_10	T1530	Azure - Dump Azure Storage Account Objects via Azure CLI	Storage Account Contributor	Storage accounts with objects
ATOMIC_11	T1528	Azure - Functions code upload - Functions code injection via Blob upload	Storage Account Contributor	Function App (that is deployed via blob object)
ATOMIC_12	T1528	Azure - Functions code upload - Functions code injection via File Share modification to retrieve the Functions identity access token	Storage Account Contributor	Function App (that is deployed via Files in File Share)

Table 7.2: Overview of the required privileges and CUT components for the atomics within the final set.

attack scenario. The various eligible roles (and groups) could have been assigned to multiple different malicious users, but for the sake of experimentation, a single user setup is sufficient to perform the penetration tests.

Conditional Access rules: The set of conditional access policies comprises five rules:

- *Multifactor authentication and reauthentication for risky sign-ins*: This is a Microsoft-managed policy, which is enabled for all users. It includes sign-in risk detection in real-time and triggers in the case of a risky sign-in a MFA request, and a new sign-in procedure for each session.[67]
- *AllowedIPRanges* defines a whitelist of IP ranges for all users, resources, and networks.
- *MFA* defines that access is granted to all users only via MFA. Furthermore, a session is configured to last 10 hours. After that period, a reauthentication is necessary.

[67] https://learn.microsoft.com/en-us/entra/identity/conditional-access/managed-policies#multifactor-authentication-and-reauthentication-for-risky-sign-ins. Accessed: Jun. 26, 2025.

- *BlockLegacyAuthentication* blocks access from legacy authentication clients.
- *BlockUnsupportedOS* blocks access from unsupported OSes.

Eligible Roles and Groups: Recall that in the PIM service, role and group assignments can be configured with the assignment type *eligible*, which is the recommended option and requires the user to perform an action to activate the role(s) for a limited amount of time[68]. The setup of groups and roles is depicted in Table 7.3. There are two eligible roles and three eligible groups. All roles are either Azure resources built-in roles[69] or Entra ID built-in roles[70]. The group *del.az.ART-Tests-Entra* enables users to manage user and role assignments of users and applications. The group *del.az.ART-Tests-Soc-Playground2* assigns the Contributor role to the user, and *ART-Tests-Development* is a group used for development purposes. It was needed to design the log infrastructure for the Log Analytics workspace *test-la*, as well as access to Microsoft Sentinel to create Analytics rules and investigate incidents. All privileges that are required for the test executions are listed in the table.

7.2.2 Azure Resources Setup

The Azure resources within the CUT are presented in this section. They form the main target objects in the security testing and represent the workloads outsourced to the cloud. The set of resources was defined according to the "CUT component" column in Table 7.2, which lists the required CUT components for a sensible execution of the defined set of atomics. The resources that were created as attack targets lie in the resource groups *arttest7cafd9rg* and *arttest7cafd9-f-app-manual_group*. Table 7.4 lists all resources within these two resource groups. Note that the cloud infrastructure contains further resources in other resource groups. For example, the resource group *rg-agent* contains resources for the Azure DevOps agent VM, which is required for the deployment (see Section 7.2.3). Resources in *rg-tfstate* contain Terraform state files, as well as other resources not depicted in Table 7.4, including virtual networks, network security groups, and a network watcher. All resources mentioned previously are located within the subscription *SUB-SOC_Playground2*. Note that there are also resources outside this subscription. They are part of the security infrastructure, which is explained in Section 7.2.4. The main resources in Table 7.4 are associated with a resource affiliation and comprise the following workloads:

[68] https://learn.microsoft.com/en-us/azure/role-based-access-control/pim-integration. Accessed: Jun. 29, 2025.
[69] https://learn.microsoft.com/en-us/azure/role-based-access-control/built-in-roles. Accessed: Jun. 29, 2025.
[70] https://learn.microsoft.com/en-us/entra/identity/role-based-access-control/permissions-reference. Accessed: Jun. 29, 2025.

Group	Role	Scope
None	Owner	Subscription: SUB-SOC_Playground2
	Storage Account Contributor	Subscription: SUB-SOC_Playground2
del.az.ART-Tests-Entra	User Administrator	Directory
	Application Administrator	Directory
	Privileged Role Administrator	Directory
del.az.ART-Tests-SOC-Playground2	Contributor	Subscription: SUB-SOC_Playground2
del.az.ART-Tests-Development	User Administrator	Directory
	Application Administrator	Directory
	Privileged Role Administrator	Directory
	Owner	Subscription: SUB-SOC_Playground2
	Contributor	Subscription: SUB-SOC_Playground2
	Microsoft Sentinel Contributor	Subscription: management-dev-00
	Contributor	Workspace: Log Analytic workspace: test-la

Table 7.3: List of roles and groups in the Entra ID setup, along with the corresponding scope. The *Directory* scope is an Entra ID role that provides access to the full Tenant. The other scopes are Azure resource roles to either a subscription or a specific resource.

- S1 (Storage account 1) contains various Storage Account objects.
- F1 (Function app 1) is an HTTP trigger Function that is deployed via a Blob storage package.
- F2 (Function app 2) is an HTTP trigger Function that is deployed via a File in a File Share.
- A1 (Automation account 1)
- V1 (Virtual Machine 1)

7.2.3 Deployment

This section presents the deployment method for the resources within the CUT. Two software tools were used for the deployment. The DevOps solution *Azure DevOps*[71] was used to orchestrate the infrastructure-as-a-code software tool *Terraform*[72]. This

[71] https://azure.microsoft.com/en-us/products/devops/. Accessed: Jun. 29, 2025.
[72] https://developer.hashicorp.com/terraform. Accessed: Jun. 29, 2025.

Resource group	Type	Name	Resource affiliation
arttest7cafd9rg	Function App	arttest7cafd9-f-app	F1
arttest7cafd9rg	Application Insights	arttest7cafd9-f-insights	F1
arttest7cafd9rg	App Service plan	arttest7cafd9-f-service-plan	F1
arttest7cafd9rg	Storage account	arttest7cafd9stacc2	F1
arttest7cafd9-f-app-manual_group	Function App	arttest7cafd9-f-app-manual	F2
arttest7cafd9-f-app-manual_group	Application Insights	arttest7cafd9-f-app-manual	F2
arttest7cafd9-f-app-manual_group	Storage account	arttest7cafd9fappmaaee3	F2
arttest7cafd9-f-app-manual_group	App Service plan	ASP-arttest7cafd9fappmanualgroup-90a7	F2
arttest7cafd9rg	Action group	Application Insights Smart Detection	
arttest7cafd9rg	Automation Account	ART-AutomationAccountName-T1078-004	A1
arttest7cafd9rg	Virtual machine	arttest7cafd9-vm	V1
arttest7cafd9rg	Disk	arttest7cafd9-vm-test-os-disk	V1
arttest7cafd9rg	Network Interface	nic-vm-test-01	V1
arttest7cafd9rg	Storage account	arttest7cafd9diag	
arttest7cafd9rg	Event Grid System Topic	arttest7cafd9diag-d1236755-ad2d-445f-b8aa-4d6bb8c77650	
arttest7cafd9rg	Storage account	arttest7cafd9stacc1	S1
arttest7cafd9rg	Event Grid System Topic	arttest7cafd9stacc1-0b59802b-bb4f-4bbb-aed2-a04f2db04bab	
arttest7cafd9rg	Event Grid System Topic	arttest7cafd9stacc2-7c414829-11c4-4270-b8bd-c0d6fb30d9d5	
arttest7cafd9rg	Recovery Services vault	RSVault-westeurope-1dcf10f2-07be-5e2d-8d1f-5c1bf42	

Table 7.4: Resources in resource groups *arttest7cafd9rg* and *arttest7cafd9-f-app-manual_group* within subscription *SUB-SOC_Playground2*.

automatic deployment procedure ensures reproducibility of the results obtained in this thesis. The code is accessible on GitHub[73].

The resources within the resource group *arttest7cafd9rg* were deployed via Terraform.

[73] https://github.com/manuelkuss/Investigation-of-Cloud-Workload-Security/tree/ab226cdb054e3e3629aafde1cbf8afab20d54236/Cloud-Under-Test

In addition to this automatic deployment, some resources of the CUT were deployed manually: The second Function App (F2), which utilizes the storage account option of Files in File Shares, was deployed manually, since azurerm[74], the Terraform provider for Azure Resource Manager (version 4.11.0), does not provide that deployment option. The Function App and its corresponding resources lie within the resource group *arttest7cafd9-f-app-manual_group*. Also, some diagnostic settings had to be configured manually via the Azure Portal, as the functionality was not possible using Terraform. Section 7.2.4 explains the methodology for configuring the diagnostic settings.

A shortened version of the deployment pipeline is presented in the Listing 7.1. More precisely, the YAML stages definition is presented. It consists of the two stages *tfvalidate* and *tfdeploy*. The validation part consists of an installation, initialization, and validation task. The deployment process consists of the tasks: installation, initialization, plan, apply, and function code upload. The Terraform resource definitions (*.tf* files) are located in the folder *Terraform*. Listing 7.2 shows two exemplary resource definitions, i.e., a resource group and a storage account within that resource group.

```
stages:
  - stage: tfvalidate
    jobs:
      - job: validate
        steps:
          - task: ms-devlabs.custom-terraform-tasks.custom-terraform-
              installer-task.TerraformInstaller@1
            displayName: install
          - task: TerraformTaskV4@4
            displayName: init
            inputs:
              provider: 'azurerm'
              command: 'init'
          - task: TerraformTaskV4@4
            displayName: validate
            inputs:
              provider: 'azurerm'
              command: 'validate'
  - stage: tfdeploy
    condition: succeeded('tfvalidate')
    dependsOn: tfvalidate
    jobs:
      - job: apply
        steps:
          - task: ms-devlabs.custom-terraform-tasks.custom-terraform-
              installer-task.TerraformInstaller@1
            displayName: install
          - task: TerraformTaskV4@4
            displayName: init
            inputs:
```

[74] https://registry.terraform.io/providers/hashicorp/azurerm/4.11.0/docs. Accessed: Jun. 29, 2025.

```
29              provider: 'azurerm'
30              command: 'init'
31          - task: TerraformTaskV4@4
32            displayName: plan
33            inputs:
34              provider: 'azurerm'
35              command: 'plan'
36          - task: TerraformTaskV4@4
37            displayName: apply
38            inputs:
39              provider: 'azurerm'
40              command: 'apply'
41          - task: AzureFunctionApp@2
42            displayName: upload function code
43            inputs:
44              connectedServiceNameARM: 'TestTenant SoC Playground2'
45              appType: functionAppLinux # default is functionApp
46              appName: '$(function_app_name)'
47              package: $(System.DefaultWorkingDirectory)/Terraform/
                    files_for_pipeline/function_app_code.zip
48              deploymentMethod: zipDeploy
```

Listing 7.1: Shortened version of the Azure DevOps pipeline.

```
1  resource "azurerm_resource_group" "rg" {
2          name     = "${var.art_test_prefix}rg"
3          location = var.location
4  }
5
6  resource "azurerm_storage_account" "storage_account_test_resources" {
7          name                     = "${var.art_test_prefix}stacc1"
8          resource_group_name      = azurerm_resource_group.rg.name
9          location                 = azurerm_resource_group.rg.location
10         account_tier             = "Standard"
11         account_replication_type = "LRS"
12 }
```

Listing 7.2: Exemplary resource definitions in Terraform.

7.2.4 Azure Security Setup

The Azure security setup is the third infrastructure pillar in the CUT setup. General information on Azure's security services was presented in Section 2.2.1. The concrete security setup of the CUT is presented in this section.

Threat Detection

The threat detection setup in the CUT consists of the configurations of Sentinel, DfC, XDR, and the log infrastructure. The log infrastructure is presented in Section 7.2.4.

Sentinel is the central component in the threat detection setup - it is the central place where all generated events in the CUT infrastructure are collected and analyzed. A prerequisite for using Sentinel is an existing Log Analytics workspace. The Log Analytics workspace *test-la* with the pricing plan "Pay-as-you-go" was used for this purpose. In Sentinel, the Content hub is used to install out-of-the-box content solutions, such as Analytics rules, data connectors, and other Sentinel content types[75]. In the CUT, the content entries *Microsoft Defender for Cloud*, *Microsoft Defender XDR*, and *Microsoft Entra ID* were installed. The activated data connectors are presented in the following.

XDR is the superordinate Defender service that manages signals from DfC and multiple other Defender services. As visualized in Figure 2.1, which illustrates Sentinel in relation to other security components, XDR was connected to Sentinel via a data connector to stream all signals from XDR to Sentinel. Furthermore, data connectors have been enabled to direct all events in the cloud setup to Sentinel. The activated data connectors are: "Azure Activity", "Azure Key Vault", "Azure Storage Account", "Microsoft Defender for Cloud Apps", "Microsoft Defender for Endpoint", "Microsoft Defender for Identity", "Microsoft Defender for Office 365 (Preview)", "Microsoft Defender XDR", "Microsoft Entra ID", "Microsoft Defender ID Protection", "Network Security Groups", "Subscription-based Microsoft Defender for Cloud (Legacy)", and "Tenant-based Microsoft Defender for Cloud (Preview)". Figure 7.1 shows the log architecture of the CUT, in which the incoming arrows of Sentinel represent the most relevant data connectors.

Aside from the data connectors, Analytics rules are also installed through content objects. Analytic rules are the threat detection rules within Sentinel. They perform queries over the log data and generate alerts and incidents for matches. The three activated content objects comprise 104 Analytic rules.

Another security component that should be mentioned for the CUT threat detection setup is the Defender portal. Although Sentinel was integrated into the Defender portal, for this thesis, there was no benefit for the threat detection investigation. Note that the deployment process of the Sentinel integration in the Defender Portal was not finished on behalf of Microsoft at the time of conducting the experiments within this thesis[76]. In a future setup, this tool may provide additional features for threat detection.

Security and Protection

Aside from the threat detection setup, the CUT security setup consists of proactive measures. The conditional access rules are one part of this proactive setup. They were presented in Section 7.2.1. Another part is the configuration of the DfC, which comprises the activation of Defender plans and security policies. The activated Defender plans for subscription *SUB-SOC_Playground2* are as follows: In addition to the free "Foundational CSPM", the advanced "Defender CSPM" was enabled. Furthermore, all CWP plans have

[75] https://learn.microsoft.com/en-us/azure/sentinel/quickstart-onboard?tabs=defender-portal. Accessed: Jun. 3, 2025.

[76] https://learn.microsoft.com/en-us/unified-secops-platform/overview-defender-portal. Accessed: Jun. 3, 2025.

been activated, except “AI workloads” and “APIs”, as these services were not in use. The activated CWP plans include: “Servers”, “App Service”, “Databases”, “Storage”, “Containers”, “Key Vault”, and “Resource Manager”.

Policies define best practices of workload configurations and comprise one part of the CSPM capability of the DfC. The enabled security policies are: “Microsoft cloud security benchmark”[77] (per default enabled), “CIS Microsoft Azure Foundations Benchmark v2.0.0”[78] (per default enabled), and “Azure CSPM”[79]. “Azure CSPM” is a set of best-practice recommendations for workload security. Beyond these three policies, various other policies also exist, such as the “NIST SP 800-171 Rev.2” and “CIS Controls v8.1”. The investigation of such further sets of policies in security tests is out of scope for this thesis and may be undertaken in future work. The three activated policies are considered as best practice in a common infrastructure setup.

The DfC recommendations, which originate from both the CSPM and the CWP, were reviewed in collaboration with a security practitioner. Some were considered in the CUT configuration; however, most were considered not useful. The DfC total secure score was 67 % for the subscription *SUB-SOC_Playground2*, in which 28 out of 45 recommendations were categorized with low risk, 17 were medium, and 0 were assessed as high or critical. The DfC workload protection (CWP) covers 100 % of the resources within the subscription.

Log Infrastructure

The objective of the log infrastructure is to direct all possible log data from cloud resources to Sentinel, enabling Sentinel to utilize it for threat detection tasks. The Azure log infrastructure comprises the tools in the Azure Monitor service. Azure Monitor is a monitoring service for “collecting, analyzing, and responding to monitoring data from [...] cloud and on-premises environments”[80]. One of its tools is Log Analytics, which enables the analysis of log data via log queries[81]. The storage location of log data is a *Log Analytics workspace*. Figure 7.1 illustrates the log infrastructure schematically. Note that the Log Analytics workspace is in a different subscription from the Azure resources relevant to the atomics execution that lie within the *SUB-SOC_Playground2* subscription. Hence, this setup is representative of an infrastructure setup involving multiple subscriptions. Two primary sources of log data send logs to the Log Analytics workspace: the diagnostic settings defined in Entra ID and the diagnostic settings defined in the various resources within *SUB-SOC_Playground2*. Diagnostic settings for all resources in both resource groups, i.e., *arttest7cafd9rg* and *arttest7cafd9-f-app-manual_group*, were activated on all levels of detail. For Storage Accounts, diagnostic settings were configured on the Storage

[77] https://learn.microsoft.com/en-us/security/benchmark/azure/overview. Accessed: Jun. 29, 2025.

[78] https://learn.microsoft.com/en-us/azure/governance/policy/samples/cis-azure-2-0-0. Accessed: Jun. 29, 2025.

[79] https://learn.microsoft.com/en-us/azure/defender-for-cloud/concept-cloud-security-posture-management. Accessed: Jun. 29, 2025.

[80] https://learn.microsoft.com/en-us/azure/azure-monitor/fundamentals/overview. Accessed: Jun. 4, 2025.

[81] https://learn.microsoft.com/en-us/azure/azure-monitor/logs/log-analytics-overview?tabs=simple. Accessed: Jun. 4, 2025.

Account level and on the storage object level, which include “blob”, “queue”, “table”, and “file”. Each diagnostic setting was configured to include all possible log sources and metrics.

7.3 Experiment Setup

This section presents the experiment setup, which involves the positioning of the attacker within the enterprise infrastructure, and the experiment plan that defines the execution of the penetration tests.

The existing enterprise infrastructure for performing the security tests is presented in Figure 7.2. The architectural components are divided into two physical areas: an Azure cloud part and an on-premise Enterprise system part. The CUT, as presented in Section 7.2, is realized in a test Tenant, which is completely separated from the production Tenant. Therefore, the Entra ID configuration and the Azure resources are separated from the production setup. The experiments are performed on a client laptop, which has an Remote Desktop Protocol (RPD) connection to the *Attacker VM*. The attacker is assumed to be a malicious insider *UserA*, who has a client workstation, which is in the experiment setup the *Attacker VM* (colored orange). The *Attacker VM* can access the *Azure Test Tenant* (colored orange) via a firewall and proxy server located in the on-premises enterprise system. The setup simulates a scenario in which a malicious insider has legitimate access to an Azure Tenant.

Experiment Plan:

The experiment plan, which represents the security testing (penetration testing) methodology, consists of the definition of (1) an ordering of the executed atomics, (2) the phases of execution, and (3) an execution workflow. The execution order specifies the order of atomics, which are grouped based on the privileges required by the attacker. These groups form the execution steps of the security testing experiment. The order is defined as follows:

1. Step: Privileges: *del.az.ART-Tests-SOC-Playground2*

   (ATOMIC_1) T1526: Azure - Dump Subscription Data with MicroBurst

   (ATOMIC_2) T1078.004: Azure Persistence Automation Runbook Created or Modified

2. Step: No privileges required

   (ATOMIC_3) T1619: Azure - Enumerate Azure Blobs with MicroBurst

   (ATOMIC_4) T1619: Azure - Scan for Anonymous Access to Azure Storage (Powershell)

3. Step: Privileges: Group *del.az.ART-Tests-Entra*

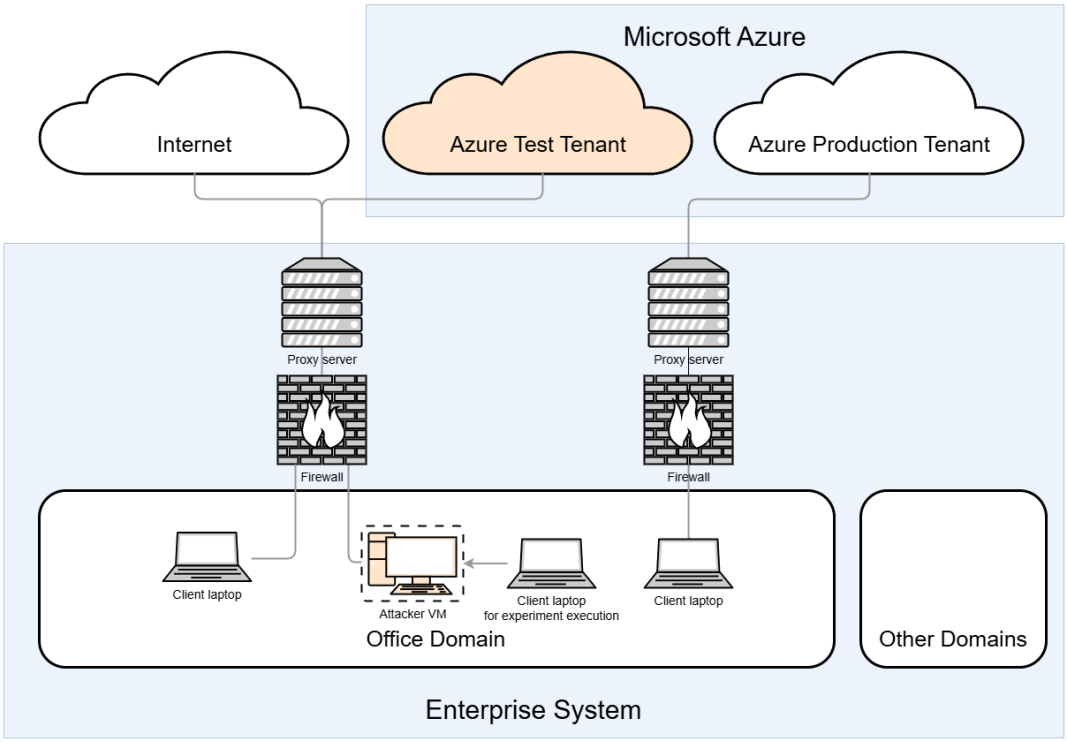

Figure 7.2: Schematic view of the enterprise infrastructure, in which the security testing experiment was conducted. The office domain and other domains are located within the on-premises area. Some client machines have access to the Azure cloud. Two tenants ensure that the security testing experiment is separated from resources running in the production tenant. The *Attacker VM* is granted access to the *Azure Test Tenant*.

(ATOMIC_5) T1136.003: Azure AD - Create a new user via Azure CLI

(ATOMIC_6) T1531: Azure AD - Delete user via Azure CLI

4. Step: Privileges: Role *Owner*

   (ATOMIC_7) T1098: Azure - adding user to Azure role in subscription

   (ATOMIC_8) T1098: Azure - adding service principal to Azure role in subscription

5. Step: Privileges: Role *Storage Account Contributor*

   (ATOMIC_9) T1619: Azure - Enumerate Storage Account Objects via Key-based authentication using Azure CLI

   (ATOMIC_10) T1530: Azure - Dump Azure Storage Account Objects via Azure CLI

(ATOMIC_11) T1528: Azure - Functions code upload - Functions code injection via Blob upload

(ATOMIC_12) T1528: Azure - Functions code upload - Functions code injection via File Share modification to retrieve the Functions identity access token

As presented in Section 7.2, the attacker is a malicious insider *UserA*. Different privileges are required for the attacker to perform the sequence of execution steps. This change of privileges is not a realistic attack scenario, but it is beneficial for the experiment execution. Switching privileges between the steps is therefore considered not part of the observation of the threat detection system. Possible alerts that may have been triggered in this time span would have been excluded from the detection statistics. On the other hand, the execution of the atomics themselves is a realistic simulation. The alerts triggered during these specific time spans comprise the data for calculating the detection metrics. Note that some atomics needed adaptations to work in the MFA setup of the cloud infrastructure. In Appendix B, the necessary adaptations are presented in more detail.

The experiments are conducted in two phases. In phase one, the unmodified detection rules of Azure's security tools are active. Details on the configuration of that threat detection setup were presented in Section 7.2.4. Phase two is executed with an optimized threat detection configuration that contains additional detection rules. The development of these new detection rules is presented in Section 8.2. In phase two, only the four atomics specified in step five were executed, as the development of detection rules resulted in detection rules for only ATOMIC_11 and ATOMIC_12, which both lie in step five. The other atomics are assumed to result in the same detection results.

The execution methodology is defined in pseudocode in Algorithm 7.1. The depicted algorithm provides a three-minute time gap between two subsequent atomics, which ensures a clear separation of the respective active atomic test. Hence, alerts that originate from log event correlations can be unambiguously assigned to the causing atomic test. The algorithm iterates over all atomics in the specified order, changes privileges if necessary, waits a certain amount of time, and executes the next atomic test. Towards the end, the detection results are collected, and the cleanup scripts of all atomics are executed to end in a clean state that corresponds to the state before the experiment execution.

7.4 Experiment Results

This section presents the results of the experiment phases. The IDS property of interest in this thesis is the attack coverage, as defined by Milenkoski et al. [6] as "The attack detection accuracy of an IDS in the presence of attacks without any background benign activity" [6]. A suitable metric for the attack coverage is the **TPR**, also known as sensitivity in statistics. The formula for calculating the TPR is given in Formula 7.1.

Algorithm 7.1: Pseudocode for the execution workflow of the security testing.

```
Set timer to 0;
Set the list of atomics to the required set of atomics for the corresponding phase;
For each atomic in the list of atomics:
    If atomic is the first in the group:
        Deactivate all not required roles;
        Activate all required roles;
    Wait for the timer to finish;
    Check if exactly the required roles for the atomic test are active;
    Execute atomic test;
    If atomic is not the last in the list of atomics:
        Set timer to 3 minutes;
Wait for any alerts to be triggered;
Collect detection results from Sentinel;
For each atomic in the list of atomics:
    Run cleanup script of atomic;
```

$$TruePositiveRate(TPR) = \frac{TruePositives(TP)}{TruePositives(TP) + FalseNegatives(FN)} \tag{7.1}$$

7.4.1 Phase 1

In phase one, all twelve atomics were executed. Only ATOMIC_1 was detected in Microsoft Sentinel, resulting in an attack coverage of **8.33 % (1/12)**. In the incidents dashboard in Sentinel, one incident with the name "MicroBurst exploitation toolkit used to enumerate resources in your subscriptions" is displayed. We observe that the security product that created the alert (refer to the Sentinel security alert schema reference[82]) is DfC. XDR is the alert provider. Hence, the alert originated in DfC, but was processed in XDR and sent further to Sentinel. We also observe that the concrete incident originates from the execution of ATOMIC_1, i.e., "Azure - Dump Subscription Data with MicroBurst". This was correlated with the time of the corresponding log data. The incident (with *Low* severity) determines the execution of an attack that used the MicroBurst exploitation toolkit. Therefore, Sentinel not only detected the malicious activity, but also precisely identified the executed action. The attacker's identity (PrincipalOid and Username), the corresponding IP address, and the executed PowerShell module are visible in the incident details. The exact detection logic is not accessible in Sentinel; however, in the alert description[83] it states that it was detected by analyzing Azure Resource Manager operations. Azure Resource Manager[84] is the

[82] https://learn.microsoft.com/en-us/azure/sentinel/security-alert-schema. Accessed: Jun. 29, 2025.

[83] https://learn.microsoft.com/en-us/azure/defender-for-cloud/alerts-resource-manager#microburst-exploitation-toolkit-used-to-enumerate-resources-in-your-subscriptions. Accessed: Jun. 3, 2025.

[84] https://learn.microsoft.com/en-us/azure/azure-resource-manager/management/overview. Accessed: Jun. 3, 2025.

component that receives and handles resource-related operations originating from an Software Development Kit (SDK) used by the Azure CLI.

7.4.2 Phase 2

In phase two, the developed detection rules both triggered an alert, which, when combined with the alert on ATOMIC_1, resulted in a detection coverage of **25 % (3/12)** for all twelve atomics. Compared to phase one, the atomics ATOMIC_11 and ATOMIC_12 were additionally detected. In the incidents dashboard in Sentinel two additional incidents are displayed that have both the name "Alert from Atomic Red Team execution". The "Alert product name" is Sentinel, and the "Incident provider name" is again XDR. Both incidents are categorized with *Informational* severity and have the name that was defined in the developed Analytics rules, which are presented in Section 8.2.

CHAPTER 8

Threat Detection

This section deals with engineering cycle two and has the objective of answering RQ3, i.e., "How can threat detection be optimized to respond to the simulated attacks?" The answer to RQ3 is required to answer the general research question by investigating the threat detection capabilities within Azure, particularly in the presence of attacks executed during the security testing phase in Chapter 7. First, the security capabilities within Azure are discussed in Section 8.1. Second, the design process of the threat detection setup, including the development of detection rules, will be outlined in Section 8.2. The detection rate is evaluated in Section 8.3.

8.1 Azure Security Capabilities

This section presents the security capabilities and possible tools for threat detection in Azure. The security products within Azure were already introduced in Section 2.2.1, and the concrete setup for the CUT was presented in Section 7.2.4. The Azure security capabilities comprise multiple cloud services that can be used solely or in combination with other services. However, the SIEM product Sentinel is the essential component for analyzing log data, generating alerts, and, consequently, incidents. DfC is the second main security product. It provides CSPM and CWP solutions. In the experimentation phase one in Section 7.4.1 we observed that only one out of twelve attacks was detected by the threat detection setup of the CUT. At first sight, this result seems to be quite bad. However, considering that many atomics simulate legit operations, it is no surprise that most of them are not detected. For example, when creating a user in ATOMIC_5 or a Persistence Automation Runbook in ATOMIC_2, there should even be no alert generated. Therefore, the set of atomics that realize legitimate operations, i.e., ATOMIC_2, ATOMIC_5, ATOMIC_6, ATOMIC_7, and ATOMIC_8, are rightly not detected. Also, ATOMIC_4 is a legitimate operation, as it attempts to access a

publicly accessible storage object. ATOMIC_3, on the other hand, uses a wordlist to probe for publicly accessible Blobs and is expected to leave enough traces to be detected.

In conclusion, for the existing atomics (ATOMIC_1 to ATOMIC_8), when considering only those that incorporate suspicious behavior that is expected to be detected, i.e., ATOMIC_1 and ATOMIC_3, one out of two atomics (50 %) was detected. When additionally considering the newly developed atomics in combination with these two atomics, one out of six atomics (16.67 %) was detected with the basic threat detection setup. The threat detection setup represents the basic capabilities of a straightforward utilization of security products in Azure, without more detailed detection logic being realized. In the following section, the threat detection setup will be improved by adding detection logic for the newly developed atomics to Sentinel.

8.2 Development of Detection Rules

This section presents two new detection rules and the corresponding Analytics rules in Sentinel. Note that the log setup, as presented in Section 7.2.4, is a prerequisite for detection and necessary for the development of detection queries. However, the same log setup already existed in the experimentation phase one to guarantee that the detection results of phase one are not worse than those of phase two due to a missing log infrastructure rather than incomplete detection logic.

8.2.1 KQL Queries

This section presents the development of Kusto Query Language (KQL) queries to detect the privilege escalation attack scenario described in Section 6.2. Recall that the essential operation in the attack is the modification of the source code of a Function, which is stored in a Storage Account. A user who should not be allowed to modify the Function can inject code into the Function and exfiltrate an access token. Therefore, we want to monitor the modification of a Function's storage object that originates from a user with insufficient privileges. The investigation of all available logs that are in relation to the attack scenario revealed two log sources indicative of the malicious behavior: the modification event of the storage object, and the preceding listkeys-action on the same storage resource, which can be used to trace the user who performed the action, including the assigned privileges of the related identity. AzureActivity[85] is a log source that originates from the Azure Monitor and provides subscription-level event logs. It includes the listkeys-action events that occur from shared key access. As mentioned by [52], the Azure Activity log does not record storage object events; however, storage object logs can be enabled in the storage account diagnostic settings, which was done in the log infrastructure setup. The second relevant log source is therefore StorageBlobLogs or StorageFileLogs, depending on the Function App's deployment type.

[85] https://learn.microsoft.com/en-us/azure/azure-monitor/platform/activity-log?tabs=powershell. Accessed: Jun. 4, 2025.

In Sentinel, detection rules are written in Azure Log Analytics, which is incorporated within Sentinel and uses the query language KQL[86]. KQL is a query language for structured and unstructured data, providing various data analytics functionalities in a hierarchical manner, similar to Structured Query Language (SQL).

The structure of the developed KQL queries for both options of the attack step, i.e., ATOMIC_11 and ATOMIC_12, is very similar. Listing 8.1 shows the query for ATOMIC_11, in which the storage object that is modified is a Blob package. The query joins the two relevant log events, i.e., StorageBlobLogs and AzureActivity, on the caller IP address and the respective related resource of the event. The latter is implemented using the substring operator "has" on the resource ID. The resource ID represents a path to the concrete resource name, i.e., it contains the subscription, resource group, provider, resource type, and resource name. Since the listkeys-action is performed on a Storage Account and the storage modification operation on a therein contained storage object, the substring check verifies whether the Storage Account in which the target object lies is the same as the one previously performed in a listkeys-action. Furthermore, constraints on the join operation include time checks and a check of the authorization role of the user who performed the listkeys-action. The first time constraint is that the listkeys-action event in AzureActivity must have occurred before the storage modification event, and the second constraint is that the storage modification event must have happened at most one minute earlier. This was considered suitable for the time relation between the two events. In the experiments, the StorageBlobLogs event follows the AzureActivity log event approximately five seconds later. In the joined log event, the authorization role is checked to be any role except roles that grant legitimate write access for Function Apps. As it was explained in Section 6.2.1, roles such as *Storage Account Contributor* may be the reason for the misconfiguration that enables the attack path.

One drawback of the KQL queries presented is that the Storage Accounts to be monitored for this type of behavior must be hard-coded in the *storageAccountsToWatch* variable in line one of Listing 8.1. Otherwise, legitimate storage modifications of all other Storage Accounts would also result in a code injection alert. There do exist log events that contain information on the correlation between a Function and its related Storage Account. However, these events don't occur at a time in relation to the malicious behavior, but instead, e.g., when the Function is started. Furthermore, there does not exist a functionality similar to environment variables for Analytic rules, which would enable the definition of this list outside the KQL query. The second detection rule developed detects ATOMIC_12 and has a similar structure. The difference is that instead of the *PutBlob* operation event in the *StorageBlobLogs*, the code injection event is *CreateFile* within the *StorageFileLogs* table.

```
1 let storageAccountsToWatch = dynamic(["arttest7cafd9stacc2", "
    arttest7cafd9fappmaaee3"]);
2 let storageBlobLogEvents = StorageBlobLogs
3     | where OperationName == "PutBlob"
```

[86] https://learn.microsoft.com/en-us/kusto/query/?view=microsoft-fabric. Accessed: Jun. 4, 2025.

```
4       | where _ResourceId has_any(storageAccountsToWatch)
5       | extend StorageBlobLogs_JoinAttrResourceId = _ResourceId
6       | extend StorageBlobLogs_CallerIpAddress = tostring(split(
            CallerIpAddress, ":", 0)[0])
7       | extend StorageBlobLogs_TimeGenerated = TimeGenerated
8       | project
9           StorageBlobLogs_TimeGenerated,
10          StorageBlobLogs_JoinAttrResourceId,
11          StorageBlobLogs_CallerIpAddress;
12  let azureActivityEvents = AzureActivity
13      | where OperationNameValue == "MICROSOFT.STORAGE/STORAGEACCOUNTS/
            LISTKEYS/ACTION"
14      | where ActivityStatusValue == "Success"
15      | extend Properties_json = parse_json(Properties)
16      | extend AzureActivity_JoinAttrResourceId = tostring(
            Properties_json.entity)
17      | where AzureActivity_JoinAttrResourceId has_any (
            storageAccountsToWatch)
18      | extend AzureActivity_CallerIpAddress = CallerIpAddress
19      | extend AzureActivity_TimeGenerated = TimeGenerated
20      | extend Authorization_json = parse_json(Authorization)
21      | extend Authorization_evidence_json = parse_json(
            Authorization_json.evidence)
22      | extend Authorization_role = Authorization_evidence_json.role
23      | project
24          AzureActivity_TimeGenerated,
25          AzureActivity_JoinAttrResourceId,
26          AzureActivity_CallerIpAddress,
27          Caller,
28          ActivitySubstatusValue,
29          Authorization_role;
30  storageBlobLogEvents
31  | join kind=inner (azureActivityEvents) on $left.
      StorageBlobLogs_CallerIpAddress == $right.
      AzureActivity_CallerIpAddress
32  | where StorageBlobLogs_JoinAttrResourceId has
      AzureActivity_JoinAttrResourceId
33  | where StorageBlobLogs_TimeGenerated > AzureActivity_TimeGenerated
34  | where StorageBlobLogs_TimeGenerated < datetime_add('minute', 1,
      AzureActivity_TimeGenerated)
35  | where Authorization_role !in~ ("Contributor", "Owner")
36  | summarize AzureActivity_TimeGenerated = arg_max(
      AzureActivity_TimeGenerated, *) by StorageBlobLogs_TimeGenerated
```

Listing 8.1: KQL query to detect ATOMIC_11, i.e., a non-permitted Function code injection event via a Blob upload action.

8.2.2 Analytics Rules

Threat detection rules are designated Analytics rules[87] in Sentinel. There exist four types of Analytics rules: scheduled rules, Near-real-time (NRT) rules, anomaly rules, and Microsoft security rules. The most common type is the scheduled rules, which run in regular intervals and examine the data from a defined period to the past. The NRT rules are a subset of the scheduled rules (they run every minute). Anomaly rules are ML rules to detect untypical behavior, and Microsoft security rules generate incidents from alerts that are ingested into Sentinel.

For the detection of the privilege escalation scenario, a scheduled rule was created for each KQL query, i.e., for each option of the function code injection action used to steal an access token. The rule frequency is set to 10 minutes, and the lookback period is set to 11 minutes. A short overlap was considered suitable to cover the exceptional case where the two log events indicating the illegal code injection fall coincidentally within two distinct periods. Therefore, the overlap ensures that the attack is also detected in this exceptional case. Further configurations are as follows: The rule threshold is set to "Trigger alert if query returns more than 0 results"; event grouping is disabled, meaning that one alert is triggered for each event; and incident creation from the rule is enabled, meaning that an incident is generated from this alert and listed as a consequence in the Sentinel dashboard.

8.3 Detection Rate

This section represents empirical cycle six, which is part of a higher-level engineering cycle. Cycle six has the objective of evaluating the design of a threat detection setup, including the developed Analytics rules, which were presented in the preceding sections. The knowledge question is defined as follows: "What is the effect of the newly developed detection rules on the detection rate?" As discussed in Section 2.3.1 and Section 7.4, the attack coverage, calculated using the TPR, is suitable for measuring how effectively an IDS detects attacks.

The execution of this second phase of the experiments was already presented in Section 7.4.2. The results show that both Analytics rules successfully detected the attack. The TPR was determined to be 25 % (3/12) on the full set of atomics of the security testing. Compared to phase one, which resulted in a TPR of 8.33 % (1/12), two more atomics were detected. When we consider only the six non-trivial atomics, as mentioned in Section 8.1, the attack coverage increases from 16.67 % (1/6) to 50 % (3/6). Since the set of atomics for the experiments is relatively small, the results are not significant for assessing the general threat detection capabilities of Azure. However, the developed threat detection setup, including the two Analytics rules, was capable of detecting the essential step in the privilege escalation scenario for both storage configurations of Functions, and thereby could improve the threat detection setup.

[87] https://learn.microsoft.com/en-us/azure/sentinel/threat-detection. Accessed: Jun. 4, 2025.

CHAPTER 9

Discussion

This chapter presents the findings on the security of cloud workloads. First, the research questions are answered. Second, the implications for a security assessment of Microsoft Azure (Section 9.2) and the suitability of the ART framework (Section 9.3) are discussed. Third, a summary of the thesis contribution is presented in Section 9.4, followed by a presentation of potential directions for future work in Section 9.5, which closes the chapter.

9.1 Answers to Research Questions

RQ1: What threats and concerns need to be considered when outsourcing workloads to the cloud?

The objective of this first research question was to elaborate the threat landscape of cloud workloads and to define the scope to the top three threat categories for the investigation within this thesis. The threat landscape has been elaborated through an SLR. As described in Section 2.1.1, the term cloud workload is rather universal, and for the sake of this thesis, a more detailed definition wasn't necessary. The reason is that the object of investigation is the cloud environment used for utilizing cloud resources, rather than a specific type of workload. This becomes clearer when examining the ranking of threats to cloud workloads, which contain, for example, mistakes in Identity and Access Management. Hence, the threat landscape for cloud workloads is the threat landscape to the cloud environment itself from the perspective of a CSC. The term threat landscape represents the entirety of threats to cloud workloads. This thesis presents a threat landscape, i.e., a collection of threats, which comprises the synthesized information from six scientific publications and four organizational reports. The data is available in a GitHub repository[88]. Concerns are part of the threat landscape. A brief overview of the

[88] https://github.com/manuelkuss/Investigation-of-Cloud-Workload-Security/tree/ab226cdb054e3e3629aafde1cbf8afab20d54236

two sets of data is given below:

- Threat landscape[89]: The collection of threats contains 139 threat objects. These threats were classified into attacks (87) and vulnerabilities (40). Additionally, there is a classification of the corresponding layers in the shared responsibility model of CSA [14], of whether it is a threat category, the CSP is fully responsible, and others. Furthermore, the attacks within the threat landscape scope of this thesis were mapped onto the ATT&CK matrix to have a standardized form for comparisons. This matrix representation is presented in Figure 5.2.
- Concern landscape[90]: The collection of concerns contains a classification into various categories such as "Regulatory & Legal Compliance" and "Accountability & Control". A hierarchical categorization tree, as presented in Figure F.1, presents an overview of all cloud security concern objects.

The second objective that is incorporated in the first research question is to identify which threats should be considered when using cloud resources. It was answered by elaborating a ranking of threats. Note that for this ranking, the extracted objects from both sets, i.e., concerns and threats, were considered (prior to the split into the two data sets). The order of threats in the ranking presents the prioritization that should be considered by organizations that think about outsourcing workloads to the cloud. The scope of this thesis has been set to the top three threats, which are (1) *Insecure Interfaces and APIs*, (2) *Mistakes in Identity and Access Management*, and (3) *Misconfiguration & Inadequate Change Control*. Note that the top threats lie under the responsibility of the CSC. As pointed out by CSA [2], over the past years, there was a shift from security issues that lie under the responsibility of the CSP, such as technological or architectural vulnerabilities, to issues that lie within the responsibility of the CSC. The top three threats must therefore be dealt with mainly on behalf of the CSC. Deducing from these findings, cloud environments are considered secure with regard to external attacks (not originating from within the CSP) and that fall within the responsibility of the CSP. However, as presented by the concern landscape[90], there do exist further issues, such as attacks that originate from within the CSP. Therefore, trust in the CSP is a key concern that must be addressed. Note that RQ1, together with the first part of RQ2, represents the risk assessment, which is after the identification of assets in need of protection, the first step within a security assessment [9].

Concluding, by answering RQ1, a threat landscape has been elaborated that goes beyond a mere overview of threat categories and areas of concern by providing a collection of threat and concern data and a unified ranking. The collected data was further enriched with a classification into several categories that can be used for further investigations.

[89] https://github.com/manuelkuss/Investigation-of-Cloud-Workload-Security/blob/ab226cdb054e3e3629aafde1cbf8afab20d54236/Threat-Landscape/data/threatLandscape.csv

[90] https://github.com/manuelkuss/Investigation-of-Cloud-Workload-Security/blob/ab226cdb054e3e3629aafde1cbf8afab20d54236/Threat-Landscape/data/concernLandscape.csv

Therefore, research question one presents the required knowledge context that is used in answering the other research questions.

RQ2: How well does the test library represent the threat landscape and what micro-attacks can be developed to improve the test coverage?

The objective of the second phase in this thesis was to investigate how well the ART database can represent the top three threat categories within the threat landscape, to subsequently identify areas that aren't yet properly represented. The development of new attack scenarios that lie within these areas represents the design objective of this thesis, which is represented in engineering cycle one.

The metric that represents the suitability of the ART database to cover the threat landscape is the test coverage, which has been determined to be 10.26 %. When interpreting this number, it is first important to note that it is a rough estimate, which does not provide clear information on how well ART represents the threat landscape. Note that in the calculation for the used metric, a technique that is represented by one atomic test is fully (100 %) covered, which is by no means true. Instead of direct reasoning from the test coverage number, the metric can be used in combination with a deeper investigation of the existing atomics to gain insights into the suitability of the ART repository and serves therefore as an aid for orientation regarding the test coverage. 10.26 % is the number of attack techniques in the top three threats, which are linked with at least one atomic test - it does not express an order of magnitude. One insight that can be derived from the metric is the inverse test coverage of 89.74 %, which shows the attack techniques that are not represented by any of the atomic tests. This already indicates that ART cannot be used within a comprehensive security testing that covers the top three threats. The situation gets even worse when taking the insights from the evaluation of the existing atomics into consideration. The suitability of the existing atomics is assessed to be at most moderate. From the 29 Azure-related atomics that have been evaluated in this thesis, the following insights have been achieved. There are two duplicate atomics, 16 atomics that use deprecated PowerShell modules, and 19 atomics that work with an outdated, credential-based authentication method, without MFA. Furthermore, it should be noted that many atomics only simulate basic procedures, such as adding or deleting a user, which are necessary for threat modeling, but they are not useful when considered by themselves.

Based on this investigation of test coverage, it is evident that most threats in the top three threat categories are not addressed by any atomic test. When removing deprecated and duplicate atomics, the percentage of techniques that are not covered at all increases from 89.74 % to 97.44 %. Taking into consideration the low quality of the atomics, the area in which new atomics could be developed to increase the test coverage, therefore, covers almost the entire threat landscape.

The second objective of RQ2, and simultaneously the engineering goal of the thesis, was the development of new atomics to increase the test coverage. One attack scenario has been identified in a second SLR, which represents cycle (3a) in the design science

methodology. The knowledge goal of this cycle was "Which attack scenarios that are currently known fall within the top three threat categories for cloud workloads, and provide sufficient detail to be realized as micro-attacks in the ART framework for an Azure cloud environment?" The answer to this question led to a privilege attack scenario that was used for the development of four atomics (see Section 6.3). The new atomics improved the test coverage by 5.13 %, i.e., from 10.26 % to 15.38 %.

The procedure presented in this thesis outlines a methodology for developing micro-attacks. The choice to identify micro-attacks that provide the most benefit in terms of increasing the test coverage may be made by considering Figure E.1. In this Figure, the blue techniques represent techniques for which no (not-deprecated) atomic test is associated yet. However, the set of existing Azure-related atomics within ART is so small that various atomics are expected to make a significant contribution to the increased capability of adversary simulation by the ART framework.

RQ3: How can threat detection be optimized to respond to the simulated attacks?

The third research question covers the investigation of the threat detection capabilities within Azure. Several methodological steps were necessary to address the question of improving the threat detection with respect to the micro-attacks within ART, including, and especially considering, the newly developed micro-attacks. As a first step towards an answer, a baseline detection rate had to be established on a cloud infrastructure that follows best practices. Afterwards, an improvement of the threat detection rate was achieved through the development of new detection rules. Finally, a security testing phase improved the detection rate, which is considered in answering RQ3 and, subsequently, the general research question. The baseline detection rate was established through a security testing phase that included defining a CUT infrastructure, along with its threat detection setup. This infrastructure is attacked in a security testing phase, in which the set of attacks from ART defines the adversarial actions to be tested. The systematic execution of these attacks produced the detection results. They represent the state of security prior to the improvement that followed afterwards.

The threat detection setup within the CUT primarily consisted of a log infrastructure and the configuration of the two main threat detection services within Azure, i.e., Sentinel and DfC. Sentinel is the SIEM and SOAR solution that was configured as the receiver of all log events and security alarms of the various Defender products. Furthermore, the installation of content through the Content hub and the activation of data connectors were part of the setup. Note that these two configuration options enable control over the sophistication of the detection ability in Sentinel. The detection ability primarily depends on the active threat detection rules (Analytics rules), assuming that the required data connections for log data are in place. The threat detection setup for this thesis did include the following content from the Content hub: *Microsoft Entra ID*, *Microsoft Defender XDR*, and *Microsoft Defender for Cloud*, which together comprise 104 Analytics rules. Note that there are various more content entities within the Content hub that can be used for threat detection. A comparison and assessment of these different contents is

considered out of scope for this thesis. The three chosen contents are considered the best practice setup that a CSC company likely uses. Note that this same threat detection setup was already in place in phase one of the experiments to establish a realistic baseline rate.

The baseline detection rate was determined in phase one of the security testing experiments, in which one out of twelve atomics (8.33 %) was successfully detected. Eleven atomics were not detected, and therefore, present cases that should be addressed by a blue team. The threat detection was concluded to be optimizable by the realization of detection rules that cover the remaining atomics.

In engineering cycle two, the objective was to develop new detection rules for detecting the privilege escalation scenario. The essential step for the attack is the execution of ATOMIC_11 or ATOMIC_12, depending on the setup of the Function App. Therefore, the focus of engineering cycle two was to detect these two atomics. Security testing phase two demonstrates the detection success of both developed Analytics rules, which generated incidents for both executed atomics. Hence, the detection rate increased from 8.33 % to 25 %. These percentage numbers cannot necessarily be taken by themselves and do not represent hard facts about a threat detection coverage, because the set of atomics within the test set is too small for statistical interpretation. However, even though these numbers are not expressive, the success of the threat detection setup in covering the investigated privilege escalation scenario could be proven.

Answering research question three, the threat detection can therefore be optimized by configuring an appropriate threat detection setup and developing Analytics rules to detect the simulated attacks. Activating or developing Analytics rules for yet undetected attacks leads to an increased detection rate.

General RQ: How can the development and evaluation of micro-attacks enhance threat detection capabilities within Microsoft Azure's cloud workload security framework?

The general research question reflects the main objective of this thesis, which is to elaborate a methodology to develop new micro-attacks and thereby improve the capability of securing cloud workloads in Azure by performing risk-based security testing using ART. The development and evaluation of micro-attacks contribute in the following ways to enhance threat detection and thereby the security of Azure:

- **Understand adversary tactics and techniques:** Developing attacks helps defenders recognize the tactics and techniques that real adversaries employ. Through the simulation of these behaviors, blue teams can anticipate and identify them with greater efficiency.
- **Improve Red Teaming by improving ART:** The first part of the question assumed the ART framework to be not sophisticated enough for comprehensive security testing. ART is a collection of micro-attacks that creates the opportunity

to perform proactive security tests for evaluating the threat detection capabilities of a cloud infrastructure. The answer to RQ2, however, produced a test coverage in combination with detailed insights into the quality of the existing atomics that concludes ART to be not suitable for comprehensive security testing. Developing new atomics significantly improves the ART framework by exposing the cloud infrastructure to new threats. Red Teaming benefits from new attack scenarios.

- **Evaluate and Improve Threat Detection:** The results of the conducted security testing phase identified atomics that had not been detected and thereby identified cases for which the threat detection system needed to be adjusted. The methodology of this thesis provides a strategy for optimizing threat detection. It discusses the threat detection possibilities in Azure, performs security tests, and develops detection rules for undetected attack scenarios. The result is an increased detection rate. Developing and activating Analytics rules from content in the Content hub in Sentinel is a key to enhancing the detection capabilities of attacks on cloud workloads in Azure.

- **Improve Incident Response:** Similar to the improvement of red teaming and threat detection, incident response routines also benefit from the development of attack scenarios and the execution of security tests.

9.2 Security Assessment of Microsoft Azure

As outlined in Section 3.1, the research questions represent a security assessment workflow that constitutes a risk-based security testing phase. The risk assessment from RQ1 sets the scope for the security testing phase to only consider highly relevant threats. RQ2 investigated how to improve the ART framework to provide a suitable set of test cases. A security testing phase then evaluates the security state of the specified CUT setup. This methodology of testing negative security requirements is stated to be highly relevant for security testing [38] [9], and using penetration testing as a concrete testing approach is a common method for conducting security tests [11].

Hence, an assessment of the security state for cloud workloads in Azure can be made. Note that the performed security testing experiments in this thesis only cover a small number of micro-attacks. The focus is on the investigated privilege escalation scenario, which itself covers only two attack paths. However, the investigation of the threat detection capabilities as part of the configuration of a CUT that follows best practices, in combination with the conducted security testing, allows some assessment points to be made:

(+) A time span of ten plus years provided in the Azure cloud a technology that can be stated as secure in terms of threats that lie within the responsibility of the CSP. Recall the shared responsibility model for cloud computing. As CSA [2] pointed out, the top security issues shifted in the past years to areas for which the CSC is responsible, which implies that CSPs succeeded in improving their security.

(+) The proactive security mechanisms in Azure provide a secure environment for deploying and managing workloads. Services such as PIM within Entra ID enable security best practices, including just-in-time access and time-bound access. The multiple Defender services and policies recommend configurations to strengthen the security posture.

(-) A drawback in Azure is the configuration complexity. Mistakes is IAM and misconfigurations are upon the top threat categories. This thesis investigated one example in this regard, which is the complexity of the privilege structure for read access to Storage Accounts, which may result in vulnerable configurations.

(-) Another negative point is the threat detection possibilities for CSC. The threat detection possibilities, which, besides the proper setup of the log infrastructure and configuration of the security products, mainly consist of the threat detection rules, can be determined by the number of active Analytics rules in the basic setup components to only provide the bottom edge of a secure detection setup. The purchase of additional content via the Content hub or the consideration of external threat detection products is recommended to strengthen the detection system. At this point, it is worth noting that the security products in Azure may incur various costs that should be considered when outsourcing workloads.

A complete security assessment goes beyond the scope of this thesis, and the reasoning within this section is intended to draw an inference based on the insights gained from this thesis. For a comprehensive assessment, the number of atomics in the security testing phase was too small and comprised only a small area within the threat landscape. The points regarding the security of Azure in this thesis are based on the premise that the CSP itself is a trusted entity. Such additional risks should be considered as part of a comprehensive security assessment conducted on behalf of the CSC. An overview of the existing concerns is presented in the concern landscape[90].

9.3 Evaluation of Atomic Red Team

Another level of investigation is the assessment of the ART framework with regard to conducting cloud security tests on Azure. The existing atomics had been evaluated in Section 5.2. It was necessary to provide insights for the definition of a set of atomics in Section 7.1, which was required to establish a baseline threat detection rate through security testing. In conclusion, the suitability of the Atomic Red Team framework is determined to be at most moderate for simulating adversarial behaviour for a security assessment of cloud infrastructure in Azure. It was observed that in the existing atomics, there exist two duplicate atomics, 16 atomics use deprecated PowerShell modules, 19 atomics do not work for an MFA login setup, one is misclassified, and one does not make sense in the existing version at all. Only eight atomics do not exhibit any of the mentioned negative aspects. The resulting small number of atomics is not capable of covering adversarial behavior and therefore does not come close to being suitable for conducting security tests. Recall that Windows-related and Linux-related tests form the great majority of atomics with over 1500 tests. The ART framework is therefore

considered to be in need of an extension and revision process in terms of cloud-related atomics. This extension was part of this thesis. The addition of more attack scenarios and updates to existing atomics contributed to the development of ART.

9.4 Contribution

This thesis investigated how the development of micro-attacks within the ART framework contributes to strengthening the state of cloud workload security. Furthermore, a risk-based testing procedure was conducted on an actual cloud infrastructure. The contribution of the thesis lies, therefore, in both its methodology and its technical results. The methodological aspect involves the development of micro-attacks and the risk-based testing methodology for the security assessment, as summarized in the following:

- **Methodology for developing micro-attacks:** To the author's knowledge, there is no paper on how to extend the coverage of adversary emulation tools. The elaborated methodology comprises the identification of the test coverage for a defined scope (the top three threat categories) using the ATT&CK framework as a standardized format to get an overview of the current status. Then, the systematic collection and categorization of incidents from threat intelligence reports enable the realization of attack steps in the form of micro-attacks, which contribute to improving the test coverage capability of the ART database.
- **Security assessment methodology:** In the related work section, it was determined that this thesis contributes to the research gap in the security testing literature and the evaluation of threat detection systems by conducting risk-based testing using the exploit database ART. The elaborated ranking was used to set the focus on the top threats and conduct security tests within this scope. Therefore, insights into the cloud workload security state can be obtained, providing valuable information for a security assessment. The sophistication of the set of attacks within the used exploit database in the scope determines the expressiveness of the results.

Technical contributions were made throughout this thesis, ranging from the presentation of a threat landscape to the results of the cloud security assessment in Azure. The main technical contribution aspects are listed in the following:

- **Presentation of Threat Landscape:** The threat landscape for cloud workloads was elaborated through an SLR. The collection and classification of detailed threat information objects within the threat landscape, as well as the established more general concern landscape, comprises this contribution aspect.
- **Threat Ranking:** A threat ranking has been elaborated, which was used to define the top three threat categories as the scope for the thesis.

- **Evaluation of the Atomic Red Team Framework:** The test coverage of the ART framework with respect to the defined scope was visualized on the ATT&CK matrix using the MITRE Navigator tool and was determined to be 10.26 %. A more detailed investigation of the existing atomics extended the information value of the bare percentage number and led to a rather bad rating. Note that only Azure-related atomics were assessed. Therefore, regarding the quality and suitability of the other atomics, especially Windows-and Linux-related atomics, which comprise the vast majority of atomics in the ART database, no statement is made.

- **Development of Atomics:** The uncovered areas in the test coverage representation were helpful in identifying an attack scenario that was used to develop new micro-attacks for the ART database. Four new atomics were developed and have been added to the ART GitHub[91] repository.

- **Improvement of Threat Detection Capabilities:** A CUT infrastructure that follows best practices and represents a state-of-the-art Azure environment has been established. In the security testing phase one, a baseline detection rate was established, and based on this detection rate, the threat detection capability of Azure was discussed. Two developed threat detection rules improved the detection coverage, which was determined in a second security testing phase to be 25 %. Therefore, the privilege escalation attack has been successfully detected.

- **Cloud Security Assessment:** The risk-based security testing procedure enabled the assessment of the security state of cloud workloads in the Azure environment. Due to the fact that the ART database was assessed as not being well-suited for comprehensive security testing and the limitation of the scope of this thesis, no general assessment conclusions were made. Instead, some positive and negative points regarding the cloud workload security in Azure are presented.

9.5 Future Work

The thesis has contributed to the limited literature on cloud security testing and the evaluation of cloud IDS. The focus of the thesis was to realize a working risk-based security testing approach and explore the impact of adding new micro-attacks, rather than presenting comprehensive results on the security state. In addition to the fact that there may be various other methods besides the risk-based security testing procedure elaborated in this thesis, the identified areas of future work mainly deal with investigating alternatives for the individual components of the testing approach and delving deeper into related fields of research. The following areas of future work result from this thesis.

Broaden the context: The thesis focused on the adversary emulation tool ART and the CSP Microsoft Azure. Although ART was considered the most suitable exploit database for emulating purely malicious workloads, no systematic investigation on other tools

[91] https://github.com/redcanaryco/atomic-red-team/pull/3093. Accessed: Jul. 4, 2025.

was conducted. Similar research to the work from Landauer et al. [35], who present a comparison of adversary emulation tools supporting Windows, Linux, MacOS, and Agents, could be conducted with the focus on the cloud domain. Another context that could be explored is that of the CSP. The thesis investigated the security of workloads within Azure. However, insights into the security of cloud workloads in general were limited because the focus was on elaborating on methodologies. Similar risk-based security testing with other adversary datasets and other CSPs may produce more comprehensive results, as was possible in this thesis.

Develop further atomics: As mentioned by Annunziata et al. [9], cloud workload security, especially with regard to PaaS, is handled relatively little in the literature. This thesis contributed to the security of PaaS by investigating an attack that utilizes a serverless function to perform a privilege escalation. However, the performed security testing phase only covered a limited number of attacks. Future work could extend the defined atomic test set using a similar methodology to increase the sophistication of ART, and consequently, provide a broader perspective on the state of cloud workload security. The elaborated method of using insights from threat intelligence reports is one example to identify attack scenarios that could be added to an attack database. Another interesting approach was presented by Chowdhury and Nguyen [53]. They developed a methodology to reveal new lateral movement paths in Azure using context-free grammar. This approach was not used in this thesis because the corresponding GitHub repository for their tool was unavailable at the time this thesis was written.

Explore other threat detection setups: In this thesis, we considered a basic, best-practice cloud setup. Future work could investigate the effect of different security policies in DfC on the proactive identification of security threats. Another area of the Azure security setup to be investigated is the threat detection configuration in Sentinel. Other installed content in the Content hub may be investigated and compared regarding the sophistication of their detection rules.

CHAPTER 10

Conclusion

Cloud computing is a significant technological development that influences virtually all companies. The associated alteration to the architecture of enterprise systems necessitates security and privacy considerations, as well as a security assessment on behalf of the CSC. This thesis investigated cloud workload security in Azure in the context of the attack simulation capabilities of the adversary emulation tool ART. The threat landscape of cloud workloads was elaborated in an SLR to provide a comprehensive view of existing threats and concerns. Furthermore, a threat ranking was presented that guides the process of securing cloud infrastructure and was used to define the top threat categories. The test (micro-attack) coverage of ART in relation to the top three threats was calculated to be 10.26 %. The evaluation of the existing atomics in combination within this metric determined that ART is not yet capable to provide attacks for performing comprehensive security testing. This procedure reflected a risk assessment methodology that presents relevant information regarding the sophistication of the subsequent security testing. The ART framework was extended with four new micro-attacks to improve the test coverage and, consequently, the possibilities to evaluate the defense system. The methodology used for this purpose, which included an SLR and the consideration of the test coverage numbers, proved to be helpful in identifying yet unrealized attack scenarios. Although ART was considered not suitable to perform a comprehensive security testing on Azure, it was used to demonstrate the risk-based security testing methodology that can be used to assess the security of cloud workloads. Furthermore, through conducting the security testing, the thesis presented a procedure to determine the effectiveness of the threat detection capability in a best-practice CUT in Azure. Due to the low quality of the micro-attacks within ART, no representative statistical numbers could be obtained. Details on the threat detection services, their configuration, and the two developed detection rules presented insights into the security of Azure and contributed to strengthening the cloud defense capabilities. Insights from this security assessment of cloud workloads in Azure were discussed, and ART was evaluated in terms of the existing Azure-related atomics.

APPENDIX A

Selection of Existing Atomics

This section presents details on the selection process of the existing atomics from the ART database. The filter criteria to remove atomics are presented in table A.1.

Exclusion Criterion ID	Exclusion Criterion Definition
C1	Atomics rely on deprecated Powershell modules
C2	Atomics have a newer atomic "replacing" them
C3	Atomics do not make sense in the existing version
C4	Atomics are not possible in the test infrastructure
C5	Atomics do not make sense in the CUT (e.g., not possible in an MFA setting), including MFA issue 1.

Table A.1: Definition of exclusion criteria.

Issues related to MFA are considered in two categories, which are presented in Table A.2. Issue one is included in C6 and results in the exclusion of the atomic test. Issue two results in an adaptation of the execution command for the respective atomic test to be usable for security testing.

MFA Issue ID	MFA Issue Definition
1	Atomics that rely on Powershell scripts/modules that connect to Azure (via AzureAD-, and/or Az modules) in a way that doesn't work with MFA enabled **are not executed**
2	Atomics that connect to Azure (via AzureAD-, and/or Az module) in the execution script (in the yaml) in a way that doesn't work with MFA enabled are **changed to a MFA-consistent way and are executed**

Table A.2: Definition of MFA issues.

The list of atomics that were excluded due to an exclusion criterion is presented in table A.3 along with the corresponding criteria. They are not part of the final atomic selection

for the security tests in this thesis. An extended table with notes on the exclusion criteria for each atomic test is presented on GitHub[36].

Atomic ID	Atomic name	Exclusion criteria
T1552.005	Azure - Search Azure AD User Attributes for Passwords	C1
T1082	Azure Security Scan with SkyArk	C1
T1528	Azure - Dump All Azure Key Vaults with Microburst	C1
T1098	Azure AD - adding permission to application	C1
T1098.001	Azure AD Application Hijacking - Service Principal	C1
T1098.001	Azure AD Application Hijacking - App Registration	C1
T1484.002	Add Federation to Azure AD	C1
T1098.003	Simulate - Post BEC persistence via user password reset followed by user added to company administrator role	C2
T1098	Azure AD - adding user to Azure AD role	C1, C2
T1098	Azure AD - adding service principal to Azure AD role	C1, C2
T1098.003	Azure AD - Add Company Administrator Role to a user	C1
T1110.001	Brute Force Credentials of single Azure AD user	C1, C5
T1110.003	Password spray all Azure AD users with a single password	C1, C5
T1136.003	Azure AD - Create a new user	C1, C2
T1531	Azure AD - Delete user via Azure AD PowerShell	C1, C2
T1552.005	Azure - Dump Azure Instance Metadata from Virtual Machines	C3
T1606.002	Golden SAML	C4
T1110.003	Password Spray Microsoft Online Accounts with MSOLSpray (Azure/O365)	C5
T1526	Azure - Enumerate common cloud services	C5
T1562.008	Azure - Eventhub Deletion	C1

Table A.3: List of the excluded atomics with the corresponding exclusion criteria.

APPENDIX B

Adaptations of Existing Atomics for Security Testing

Some atomics needed adaptations to work in the defined CUT setting. These adaptations are all related to MFA issue two, which is defined in Appendix A. Atomics are specified in YAML files, in which the section "executor" defines the execution command and the cleanup command. ATOMIC_1, ATOMIC_7, and ATOMIC_8 needed adaptations on the Azure CLI login command "Connect-AzAccount". Figure B.1 shows the changes to ATOMIC_7. The "Connect-AzAccount" command was changed to not include the "-Credential" parameter. This causes it to run in interactive mode, which allows the MFA request to be completed. These changes are necessary for any realistic execution scenario, since MFA is a best practice that should be enforced on every login mechanism.

Figure B.1: Changes to the login commands in ATOMIC_7.

APPENDIX C

Reclassifications of Existing Atomics

The ART repository was investigated in Section 5.2 on the version of ART on the 13th of March 2025[2]. The reclassifications in terms of the MITRE technique numbers are presented in Table C.1.

Atomic Name	**Technique ID**	**New Technique ID**	**Reference**
Azure - Scan for Anonymous Access to Azure Storage (Powershell)	1530	1619	Pull request 3121 [92]
Azure - Enumerate Azure Blobs with MicroBurst	1530	1619	Pull request 3120 [93]
Azure - Dump All Azure Key Vaults with Microburst	1528	1555.006	Pull request 3119 [94]

Table C.1: List of reclassified atomic tests in relation to the corresponding MITRE technique ID.

One atomic test is incorrectly classified with regard to the platform: the atomic test "New-Inbox Rule to Hide E-mail in M365" was migrated from the *azure-ad* platform to the *office-365* platform in pull request 3101[95].

[92] https://github.com/redcanaryco/atomic-red-team/pull/3121. Accessed: Jul. 15, 2025.
[93] https://github.com/redcanaryco/atomic-red-team/pull/3120. Accessed: Jul. 15, 2025.
[94] https://github.com/redcanaryco/atomic-red-team/pull/3119. Accessed: Jul. 15, 2025.
[95] https://github.com/redcanaryco/atomic-red-team/pull/3101. Accessed: May 19, 2025

APPENDIX D

MITRE Navigator Representation Process

This section describes the process of generating the ATT&CK Navigator[96] visualizations within this thesis. The ATT&CK navigator provides basic navigation and annotation features for the ATT&CK matrices, enabling the illustration of defensive coverage in the corresponding web application[97]. The process of visualizing attacks within the top three threat categories primarily involves generating a JSON file, which is the underlying data structure format that holds the information. Three types of representations were created for the thesis, i.e., (1) attack techniques that lie within the top three threat categories, (2) attack techniques that are referenced in micro-attacks, and (3) the combination of (1) and (2) to represent and visualize the test (micro-attack) coverage upon the top three threat categories. The techniques within the ATT&CK framework therefore serve as a standardized data format that enables a mapping between threat landscape objects and existing micro-attacks within ART. The process to generate the JSON files is as follows:

- The basic structure is given in a template file, which defines the platform filter (IaaS, SaaS, Office Suite, Identity Provider) and some other fields. It represents, therefore, the fundamental layer of the cloud enterprise matrix techniques.
- For each representation, the relevant techniques are enriched by metadata information that indicates the corresponding category (Micro-attacks, Top three threats) along with a count number that represents how many atomics or threats are represented by the technique.
- In a next step, further metadata is inserted for representations (1) and (3). The techniques in category (1) are enriched by parent techniques and sub-techniques.

[96] https://github.com/mitre-attack/attack-navigator. Accessed: Jul. 2, 2025.
[97] https://mitre-attack.github.io/attack-navigator/. Accessed: Jul. 2, 2025

For each technique in the representation, the following is checked. When the threat is a sub-technique, the corresponding parent technique is also added. Other sub-techniques of this parent technique are not colored if they are not part of the threat landscape themselves. When the technique is a parent technique, all the sub-techniques are added. This step is assumed to be the best way to represent the coverage of attack techniques in the threat landscape by utilizing the ATT&CK Navigator.

- Subsequently, a score is set for all techniques. Depending on the type of concrete representation, this score is calculated differently. Score 1 indicates the category of interest, score 0 represents the foundational layer upon which techniques with score 1 will be interpreted. The scoring representation is later used to calculate the coverage metrics. When calculating statistics for the representations, the percentage is derived with the formula "number of techniques with score 1 / (number of techniques with score 0 + number of techniques with score 1)". For example, the test coverage was calculated by assigning a score of 1 to all techniques that are present in the micro-attack representation and the top three threats representation. Score 0 is assigned to all other techniques in the top three threat categories. Consequently, it is possible to calculate the test coverage.

- The last step assigns a color to the techniques according to their categories. (1) Techniques within the top three threats are colored blue. (2) Techniques with an associated micro-attack are colored red. In the third representation (3), techniques that lie in (1) and (2) are colored purple and thereby represent the intersected techniques.

The navigator files, as well as the visualizations, are available in GitHub[98].

[98] https://github.com/manuelkuss/Investigation-of-Cloud-Workload-Security/tree/ab226cdb054e3e3629aafde1cbf8afab20d54236/Threat-Landscape/MITRE-Navigator-Representation

APPENDIX E

Additional Visualizations

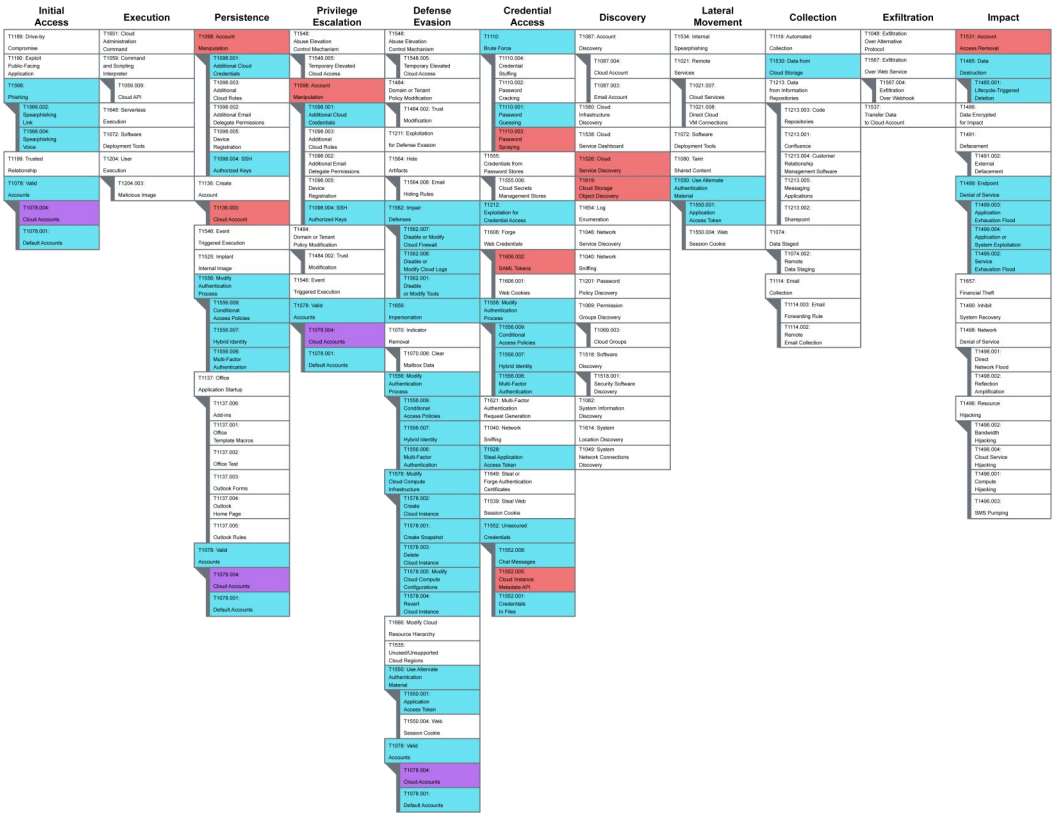

Figure E.1: Combined visualization of threats within the top three threat categories (blue), and threats that are referenced by atomics in the ART database (red). Intersected techniques are coloured purple. The set of atomics includes all atomics that do not rely on deprecated Powershell modules.

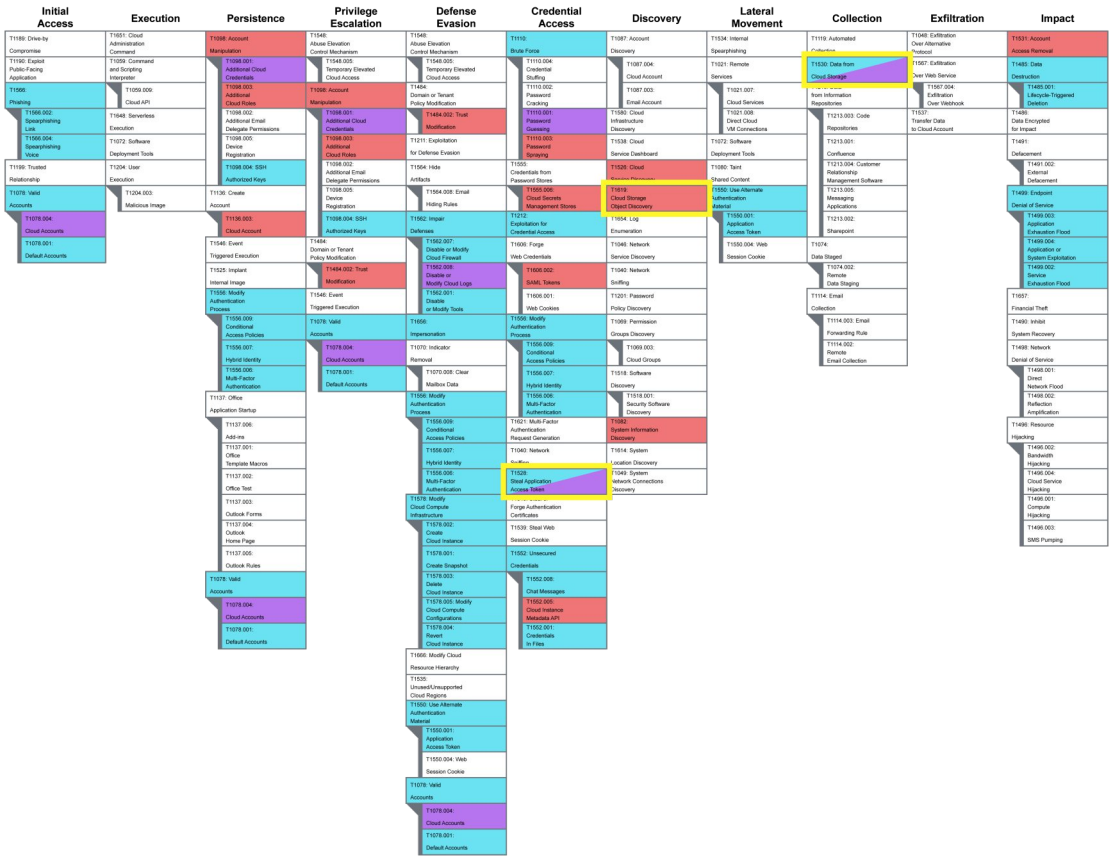

Figure E.2: Visualization of the effect of the developed four new atomics on the ATT&CK matrix. The techniques in which the new atomics lie are framed in yellow. A diagonally split technique visualizes the shift in the color scheme through the addition of the atomics. The left part in the split corresponds to the old color. The right part represents the updated color of the technique (as an effect of the new atomic tests). For technique T1619, the color stays the same. Threats within the top three threat categories are colored blue, and threats referenced by atomics in the ART database are colored red. Intersected techniques are colored purple.

APPENDIX F

Concern Landscape

Cloud security concerns	Privacy & Compliance	Regulatory & Legal Compliance	Auditing
			Non-compliance and fines
			Regulation differences because the service may be unable to assure compliance in all jurisdictions an organization operates in
			Verification of subscriber policies through provider controls
			Auditing of data-integrity protection
			Copyright
			Certification/accreditation requirements for a given cloud service
			Legal framework
			Liability
		Data Privacy & Ownership	(Violations of) data privacy / confidentiality regulations
			Potential unauthorized secondary use
			Data proliferation
			Data location: Consumers are unaware where the data is actually stored
			Privacy enforcement
			Threats related to third-party data control
			User behavior monitoring
			Data ownership
			Data secrecy
	Operational & Process risks	System & Operational Failures	Dynamic provisioning
			Impact of customer penetration testing on other users
			Impact of infrastructure modifications
			Operational failures
			System performance
			Availability and continuity risks
			Business disruption
			Feature delay
			Lost revenue
		Resource & Service Management	Managing costs
			SLA issues
			Cloud resource management
			Dependence on fast internet connections
			Cloud provider may subcontract some resources from a third party

Figure F.1: Categorization of cloud security concerns.

(cont.) Cloud security concerns	(cont.) Operational & Process risks	Accountability & Control	Lack/loss of control
			Inadequate diligence
			Limited cloud visibility/observability
			Insufficient/missing accounting
			Lack of trust
			Lack of visibility into CSPs operations (security, environment management)
	Cloud technology concern	Cloud as new technology, and complex nature of cloud	Lack of standardization
			Inadequate infrastructure design and planning
			Loss of devices that are connected to the cloud (e.g. mobile device)
			Phase transition phenomena and other effects specific to complex systems
			Incomplete data-deletion
			Commercial operating systems do not support multilayered security
			Lack of expertise
			The lack of skilled staff to manage the security of cloud applications used by an organization
			Company reputation
			Customer reputation
			Multi-tenant environment / data isolation
			Heterogeneous devices
			Configuration security
	Data security, -management and -integrity		(Permanent) Data loss
			Failures of the consistency management
			Recovery-related issues
			Data integrity
			Data integration
			Data encryption
			Data sharing
			Data recovery
			Lack of data management
			Ensuring consistency among the replicas
			Carry out timely backups
			Data remanence after de-provisioning
			Data distortion
			Data Breaches and Loss
			The inability to prevent the misuse of data or theft by insiders
			The inability to control and keep track of data in transit to and from cloud applications
	Infrastructure and network security		Side channeling
			Network breaches
			DNS service
			Loss of control in execution environment
			Protect the infrastructure used to connect to the cloud
			Heterogeneity in hardware and software cause flaws as the security setting for different resources would be different
			Insecure CSP default settings
			Network bandwidth
			(Data) Availability
	Software/Application security		Unforseen vulnerabilities and exploits introduced by AI-generated code
			Insecure software development
			Insecure Thrid-Party Resources
			Maintaining consistent security across multiple platforms or tools
			Forced version updates
			API-associated risks

Figure F.2: Categorization of cloud security concerns (cont.).

Overview of Generative AI Tools Used

The generative AI tools used for this thesis are listed in the following:

- Grammarly[99] and DeepL[100] were used as auxiliary tools for proofreading and editing to ensure correctness and clarity. Both were used intermittently between Apr. and Jul. 2025.
- ChatGPT[101] and Microsoft Copilot[102] were used intermittently between Mar. and Jun. 2025 for the following tasks:
  - Mapping attacks to the MITRE techniques: In the mapping task of Section 4.2.1, the corresponding MITRE techniques of attacks were identified using the auxiliary tools ChatGPT and Microsoft Copilot. Although they turned out to be useful for providing an overview of many MITRE techniques, no direct results were included without manual modifications. The prompts follow the structure presented below:

    > "Output for the following attacks appropriate MITRE techniques that cover them. Use MITRE version 16.1 and only consider MITRE techniques that are in the cloud matrix.
    > 1) (example) Disabled/misconfigured monitoring and logging (on a heterogeneous workload system)
    > 2) [...] "

  - Mapping incidents to the MITRE techniques: In the mapping task of the data extraction in Section 6.1.3, the corresponding MITRE techniques that are relevant for attack steps in the incident were populated with the auxiliary tool ChatGPT, preceding the final manual assessment. Similar to the task before, no direct results were included. Manual assessments of the incidents were necessary to identify all corresponding MITRE techniques that can be

[99] Grammarly Inc., "Grammarly", Available: https://www.grammarly.com/, Accessed: Jul. 31, 2025.
[100] DeepL SE, "DeepL", Available: https://www.deepl.com/en/whydeepl, Accessed: Jul. 31, 2025.
[101] OpenAI, "ChatGPT (GPT-4o model)", Available: https://chatgpt.com, Accessed: Jun. 30, 2025.
[102] Microsoft, "Microsoft Copilot", Available: https://copilot.microsoft.com/, Accessed: Apr. 15, 2025.

mapped to attack steps within the incidents. The prompts follow the structure presented below:

> "Output for the following incidents appropriate MITRE techniques that cover attack steps involved. Use MITRE version 16.1 and only consider MITRE techniques that are in the cloud matrix.
> 1) (example) DarkBeam leaks billions of email and password combinations. September 27, 2023.
> 2) [...] "

– Threat landscape categorization: The threat landscape categorization hierarchy, as presented in Section 4.2.2 and illustrated in Figures F.1 and F.2, is a remnant of the thesis - an additional output of the threat landscape representation. The categorization was created by utilizing ChatGPT. The list of concerns in the last column was provided as input to retrieve a categorization. A manual assessment and refinement process resulted in the final categorization. The prompts follow the structure presented below:

> "Categorize this list of concerns: [list of concerns]"

– The generative AI tools were used as auxiliary tools for researching on the definition of Terraform resource scripts, implementing data processing tasks, and other minor scripting tasks.

List of Figures

List of Tables

List of Listings

Acronyms

AD Active Directory. 10

API Application Programming Interface. 43, 50

APT Advanced Persistent Threat. 17

ART Atomic Red Team. ix, xi, 2–5, 15, 17–19, 21–23, 25–27, 36, 44, 47, 49–51, 53, 54, 69, 70, 93, 95–103, 105, 113, 114, 119, 120

CNAPP Cloud-Native Application Protection Platform. 11

CSA Cloud Security Alliance. 6, 17, 27, 36, 41–43, 56, 64, 94, 98

CSC Cloud Service Customer. 1, 2, 6, 9, 37, 38, 62, 71, 93, 94, 97–99, 103

CSP Cloud Service Provider. 1, 5–7, 9, 17, 37, 38, 43, 56, 94, 98, 99, 101, 102

CSPM Cloud Security Posture Management. 13, 79, 80, 87

CUT Cloud under Test. 3, 26–28, 65, 69–72, 74, 75, 77–81, 87, 96, 98, 101, 103, 107, 120

CVE Common Vulnerabilities and Exposures. 40

CWP Cloud Workload Protection. 13, 79, 80, 87

CWPP Cloud Workload Protection Platform. 13

DevSecOps Development Security Operations. 13

DfC Microsoft Defender for Cloud. 11–13, 64, 71, 78–80, 84, 87, 96, 102, 119

DoS Denial of Service. 14

EPP Endpoint Protection Platform. 36

GUI Graphical User Interface. 10

IaaS Infrastructure as a Service. 5, 16, 17, 38

IAM Identity and Access Management. 10, 41–43, 62, 64, 65, 99

ICS Industrial Control Systems. 17

IDPS Intrusion Detection and Prevention System. 11, 14

IDS Intrusion Detection System. ix, xi, 2, 8, 14–16, 18, 23, 28, 83, 91

IMDS Instance Metadata Service. 66

IPS Intrusion Prevention System. 14

KQL Kusto Query Language. 88, 89, 91

MFA Multi-Factor Authentication. 10, 13, 51, 65, 73, 83, 95, 99, 105, 107

ML Machine Learning. 14, 91

NIST National Institute of Standards and Technology. 5, 6, 14

PaaS Platform as a Service. 5, 10, 16, 17, 38, 57, 102

PIM Privileged Identity Management. 10, 13, 59, 74, 99

RBAC Role-Based Access Control. 10

RPD Remote Desktop Protocol. 81

SaaS Software as a Service. 5, 16, 17, 38

SDK Software Development Kit. 85

SIEM Security Information and Event Management. 11, 12, 71, 87, 96

SLR Systematic Literature Review. 24–27, 31–34, 36–38, 41, 53, 54, 56, 93, 95, 100, 103

SOAR Security Orchestration Automated Response. 11, 96

SQL Structured Query Language. 89

TiC Testing in the Cloud. 16

ToC Testing of the Cloud. 16, 17

ToiC Testing of the Cloud in the Cloud. 16

TPR True Positive Rate. 15, 83, 91

TTP Tactics, Techniques, and Procedures. 17, 27

UI User Interface. 43

VM Virtual Machine. 7, 50, 51, 60, 74

XDR Microsoft Defender Extended Detection and Response. 11, 12, 71, 78, 79, 84, 85, 119

Bibliography

[1] W. Jansen and T. Grance, *Guidelines on Security and Privacy in Public Cloud Computing*, National Institute of Standards and Technology, Special Publication 800-144, Dec. 2011. doi: 10.6028/nist.sp.800-144.

[2] Cloud Security Alliance, "Top threats to cloud computing 2024," Aug. 2024, Accessed: Oct. 30, 2024. [Online]. Available: https://cloudsecurityalliance.org/artifacts/top-threats-to-cloud-computing-2024

[3] Crowdstrike, "2023 global threat report," 2023, Accessed: Nov. 4, 2024. [Online]. Available: https://www.crowdstrike.com/resources/reports/crowdstrike-2023-global-threat-report

[4] ——, "2024 global threat report," 2024, Accessed: Nov. 4, 2024. [Online]. Available: https://www.crowdstrike.com/resources/reports/crowdstrike-2024-global-threat-report

[5] Gartner Inc., "Gartner forecasts worldwide public cloud end-user spending to total $723 billion in 2025," Nov. 2024, Accessed: Jul. 1, 2025. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2024-11-19-gartner-forecasts-worldwide-public-cloud-end-user-spending-to-total-723-billion-dollars-in-2025

[6] A. Milenkoski, M. Vieira, S. Kounev, A. Avritzer, and B. D. Payne, "Evaluating computer intrusion detection systems: A survey of common practices," *ACM Computing Surveys*, vol. 48, no. 1, pp. 1–41, Sep. 2015. doi: 10.1145/2808691.

[7] Z. Liu, B. Xu, B. Cheng, X. Hu, and M. Darbandi, "Intrusion detection systems in the cloud computing: A comprehensive and deep literature review," *Concurrency and Computation: Practice and Experience*, vol. 34, no. 4, Feb. 2022. doi: 10.1002/cpe.6646.

[8] A. Bertolino, G. D. Angelis, M. Gallego, B. García, F. Gortázar, F. Lonetti, and E. Marchetti, "A systematic review on cloud testing," *ACM Computing Surveys*, vol. 52, no. 5, pp. 1–42, Sep. 2019. doi: 10.1145/3331447.

[9] G. Annunziata, A. Sheykina, F. Palomba, A. De Lucia, G. Catolino, and F. Ferrucci, "Security risk assessment on cloud: A systematic mapping study," in *Proceedings of the*

28th International Conference on Evaluation and Assessment in Software Engineering, ser. EASE 2024, vol. 9. Salerno, Italy: Association for Computing Machinery, New York, NY, United States, Jun. 2024, pp. 604–613. doi: 10.1145/3661167.3661287 .

[10] K. Crawley, *Cloud Penetration Testing for Red Teamers: Learn How to Effectively Pentest AWS, Azure, and GCP Applications*, 1st ed. Birmingham, UK: Packt Publishing Ltd., 2023.

[11] P. Zech, M. Felderer, and R. Breu, "Towards a model based security testing approach of cloud computing environments," in *2012 IEEE Sixth International Conference on Software Security and Reliability Companion*, vol. 43. Gaithersburg, MD, United States: IEEE, Jun. 2012, pp. 47–56. doi: 10.1109/sere-c.2012.11 .

[12] *Software and systems engineering - Software testing - Part 2: Test processes*, ISO/IEC/IEEE Std. 29 119-2:2021, 2021. [Online]. Available: https://www.iso.org/obp/ui/#iso:std:iso-iec-ieee:29119:-2:ed-2:v1:en

[13] P. M. Mell and T. Grance, *The NIST definition of cloud computing*, National Institute of Standards and Technology, Special Publication 800-145, Sep. 2011. doi: 10.6028/nist.sp.800-145 .

[14] Cloud Security Alliance, "Security guidance: For critical areas of focus in cloudcomputing v5," Jul. 2024, Accessed: Oct. 30, 2024. [Online]. Available: https://cloudsecurityalliance.org/artifacts/security-guidance-v5

[15] National Institute of Standards and Technology, "cloud workload - Glossary | CSRC," Accessed: Jan. 24, 2025. [Online]. Available: https://csrc.nist.gov/glossary/term/cloud_workload

[16] W. D. Mulia, N. Sehgal, S. Sohoni, J. M. Acken, C. L. Stanberry, and D. J. Fritz, "Cloud workload characterization," *IETE Technical Review*, vol. 30, no. 5, pp. 382–397, 2013. doi: 10.4103/0256-4602.123121 .

[17] National Institute of Standards and Technology, "threat - Glossary | CSRC," Accessed: Jan. 1, 2025. [Online]. Available: https://csrc.nist.gov/glossary/term/threat

[18] ——, "attack - Glossary | CSRC," Accessed: Jan. 24, 2025. [Online]. Available: https://csrc.nist.gov/glossary/term/attack

[19] ——, "vulnerability - Glossary | CSRC," Accessed: May 25, 2025. [Online]. Available: https://csrc.nist.gov/glossary/term/vulnerability

[20] ——, "intrusion - Glossary | CSRC," Accessed: May 13, 2025. [Online]. Available: https://csrc.nist.gov/glossary/term/intrusion

[21] M. V. Carriegos, A. L. M. Castaneda, M. T. Trobajo, and D. A. De Zaballa, "On aggregation and prediction of cybersecurity incident reports," *IEEE Access*, vol. 9, pp. 102 636–102 648, 2021. doi: 10.1109/access.2021.3097834 .

[22] National Institute of Standards and Technology, "concern - Glossary | CSRC," Accessed: Jan. 24, 2025. [Online]. Available: https://csrc.nist.gov/glossary/term/concern

[23] ——, "security risk - Glossary | CSRC," Accessed: Jan. 24, 2025. [Online]. Available: https://csrc.nist.gov/glossary/term/security_risk

[24] ——, "challenge - Glossary | CSRC," Accessed: Jan. 24, 2025. [Online]. Available: https://csrc.nist.gov/glossary/term/challenge

[25] Kaspersky, "Threat landscape," Kaspersky IT Encyclopedia, Accessed: Jun. 8, 2025. [Online]. Available: https://csrc.nist.gov/glossary/term/challenge

[26] Microsoft, "Microsoft Defender XDR integration with Microsoft Sentinel | Microsoft Learn," Accessed: Jun. 29, 2025. [Online]. Available: https://learn.microsoft.com/en-us/azure/sentinel/microsoft-365-defender-sentinel-integration?tabs=azure-portal

[27] K. Prabu and P. Sudhakar, "A comprehensive survey: Exploring current trends and challenges in intrusion detection and prevention systems in the cloud computing paradigm," in *2024 2nd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)*. Bengaluru, India: IEEE, Jan. 2024, pp. 351–358. doi: 10.1109/idciot59759.2024.10467700 .

[28] A. Patel, M. Taghavi, K. Bakhtiyari, and J. Celestino Júnior, "An intrusion detection and prevention system in cloud computing: A systematic review," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 25–41, Jan. 2013. doi: 10.1016/j.jnca.2012.08.007 .

[29] R. Kumar and R. Goyal, "On cloud security requirements, threats, vulnerabilities and countermeasures: A survey," *Computer Science Review*, vol. 33, pp. 1–48, Aug. 2019. doi: 10.1016/j.cosrev.2019.05.002 .

[30] J. Lee, J. Cho, J. Seo, T. Shon, and D. Won, "A novel approach to analyzing for detecting malicious network activity using a cloud computing testbed," *Mobile Networks and Applications*, vol. 18, no. 1, pp. 122–128, May 2012. doi: 10.1007/s11036-012-0375-1 .

[31] *Information technology - Security techniques - Code of practice for information security controls based on ISO/IEC 27002 for cloud services*, ISO/IEC Std. 27 017:2015, 2015. [Online]. Available: https://www.iso.org/obp/ui/#iso:std:iso-iec:27017:ed-1:v1:en

[32] S. Oesch, A. Chaulagain, B. Weber, M. Dixson, A. Sadovnik, B. Roberson, C. Watson, and P. Austria, "Towards a high fidelity training environment for autonomous cyber defense agents," in *Proceedings of the 17th Cyber Security Experimentation and Test Workshop*, ser. CSET 2024, vol. 22. Philadelphia, PA,: Association for Computing Machinery, New York, NY, United States, Aug. 2024, pp. 91–99. doi: 10.1145/3675741.3675752 .

[33] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington, and C. B. Thomas, "Mitre att&ck: Design andphilosophy," *The MITRE Corporation*, Mar. 2020. [Online]. Available: https://attack.mitre.org/docs/ATTACK_Design_and_Philosophy_March_2020.pdf

[34] B. Al-Sada, A. Sadighian, and G. Oligeri, "Mitre att&ck: State of the art and way forward," *ACM Computing Surveys*, vol. 57, no. 1, pp. 1–37, Oct. 2024. doi: 10.1145/3687300 .

[35] M. Landauer, K. Mayer, F. Skopik, M. Wurzenberger, and M. Kern, "Red team redemption: A structured comparison of open-source tools for adversary emulation," in *2024 IEEE 23rd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, Dec. 2024, pp. 117–128. doi: 10.1109/trustcom63139.2024.00043 .

[36] M. Sivamanikanta, M. A. M. Abbas, and P. Das, "Exploring the capabilities of the metasploit framework for effective penetration testing," in *Data Science and Network Engineering. ICDSNE 2023. Lecture Notes in Networks and Systems*, S. Namasudra, M. C. Trivedi, R. G. Crespo, and P. Lorenz, Eds., vol. 791. Springer, Singapore, 2024, pp. 457–471. doi: 10.1007/978-981-99-6755-1_35 .

[37] R. J. Wieringa, *Design Science Methodology for Information Systems and Software Engineering*, 1st ed. Springer Berlin, Heidelberg, Nov. 2014. doi: 10.1007/978-3-662-43839-8 .

[38] P. Zech, "Risk-based security testing in cloud computing environments," in *2011 Fourth IEEE International Conference on Software Testing, Verification and Validation*. Berlin, Germany: IEEE, Mar. 2011, pp. 411–414. doi: 10.1109/icst.2011.23 .

[39] B. Kitchenham, "Guidelines for performing systematic literature reviews in software engineering," Software Engineering Group, School of Computer Science and Mathematics, Keele University, and Department of Computer Science, University of Durham, UK, EBSE Technical Report EBSE-2007-01, Jul. 9, 2007. [Online]. Available: https://www.researchgate.net/publication/302924724_Guidelines_for_performing_Systematic_Literature_Reviews_in_Software_Engineering

[40] D. C. Marinescu, *Cloud computing*, 3rd ed. Cambridge, MA, United States: Morgan Kaufmann, 2023. [Online]. Available: https://books.google.at/books?lr=&id=XOBWEAAAQBAJ&

[41] J. Bharti and S. Singh, "A review on security vulnerabilities in cloud computing," in *Data Engineering and Applications*, J. Agrawal, R. K. Shukla, S. Sharma, and C.-S. Shieh, Eds. Singapore: Springer Nature Singapore, 2024, pp. 229–247. doi: 10.1007/978-981-97-2451-2_16 .

[42] D. I. G. Amalarethinam and J. Vinnarasi, "A survey of challenges associated with cloud computing security," in *Applied Intelligence and Informatics*, M. Mahmud, H. Ben-Abdallah, M. S. Kaiser, M. R. Ahmed, and N. Zhong, Eds. Cham: Springer Nature Switzerland, 2024, pp. 470–483. doi: 10.1007/978-3-031-68639-9_31 .

[43] C. Surianarayanan and P. R. Chelliah, "Integration of the internet of things and cloud: Security challenges and solutions – a review," *International Journal of Cloud Applications and Computing*, vol. 13, no. 1, pp. 1–30, Jul. 2023. doi: 10.4018/ijcac.325624 .

[44] M. Humayun, M. Niazi, M. F. Almufareh, N. Z. Jhanjhi, S. Mahmood, and M. Alshayeb, "Software-as-a-service security challenges and best practices: A multivocal literature review," *Applied Sciences*, vol. 12, no. 8, p. 3953, Apr. 2022. doi: 10.3390/app12083953 .

[45] A. Wali and F. Alshehry, "A survey of security challenges in cloud-based scada systems," *Computers*, vol. 13, no. 4, p. 97, Apr. 2024. doi: 10.3390/computers13040097 .

[46] Crowdstrike, "2023 cloud risk report: The rise of the cloud-conscious adversary," 2023, Accessed: Oct. 30, 2024. [Online]. Available: https://www.crowdstrike.com/resources/reports/crowdstrike-2023-cloud-risk-report-executive-summary/

[47] Palo Alto Networks, "2024 state of cloud-native security report," Mar. 7, 2025, Accessed: Jun. 8, 2025. [Online]. Available: https://www.paloaltonetworks.com/resources/research/state-of-cloud-native-security-2024

[48] SentinelOne, "2024 cloud security report," Jun. 28, 2024, Accessed: Jun. 8, 2025. [Online]. Available: https://www.sentinelone.com/resources/2024-cloud-security-report-sentinelone/

[49] Cloud Security Alliance, "'The Treacherous Twelve' Cloud Computing Top Threats in 2016," Feb. 2016, Accessed: May 14, 2025. [Online]. Available: https://cloudsecurityalliance.org/artifacts/the-treacherous-twelve-cloud-computing-top-threats-in-2016/

[50] N. Sun, J. Zhang, P. Rimba, S. Gao, L. Y. Zhang, and Y. Xiang, "Data-driven cybersecurity incident prediction: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1744–1772, 2019. doi: 10.1109/comst.2018.2885561 .

[51] Orca Security, "State of cloud security report: Uncovering what is lurking in the depths of cloud environments," 2024, Accessed: Jun. 8, 2025. [Online]. Available: https://orca.security/wp-content/uploads/2024/02/2024-State-of-Cloud-Security-Report.pdf

[52] R. Nisimi, "From listkeys to glory: How we achieved a subscription privilege escalation and rce by abusing azure storage account keys," Orca Security, Apr. 2023, Accessed: May 30, 2025. [Online]. Available: https://orca.security/resources/blog/azure-shared-key-authorization-exploitation

[53] A. Chowdhury and H. Nguyen, "Cozure: Context free grammar co-pilot tool for finding new lateral movements in azure active directory," in *Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses*, ser. RAID 2023. Hong Kong, China: Association for Computing Machinery, New York, NY, United States, Oct. 2023, pp. 426–439. doi: 10.1145/3607199.3607228 .