TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Thomas Fankhauser 234170IVCM

Evaluating the OWASP IoT Security Testing Guide: A Case Study on Consumer IoT Penetration Testing

Master's thesis

Supervisor: Shaymaa Mamdouh Khalil

MSc

Tallinn 2026

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Thomas Fankhauser 234170IVCM

OWASP IoT-turvalisuse testimise juhendi hindamine: juhtumiuuring tarbijatele mõeldud IoT-seadmete läbitungimistestist

Magistritöö

Juhendaja: Shaymaa Mamdouh Khalil

MSc

Tallinn 2026

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Thomas Fankhauser

03.01.2026

Tallinn 2026

Abstract

Consumer Internet of Things (IoT) devices are becoming increasingly more complex, combining hardware, firmware, and wireless communication, connected to mobile applications or cloud services. While many areas of cybersecurity assessments have established methodology frameworks, systematic and reproducible penetration testing of such multi-layered ecosystems remains challenging. The Open Worldwide Application Security Project (OWASP) therefore published the IoT Security Testing Guide, which proposes a systematic methodology for IoT security assessments, although its effectiveness and applicability in real-world IoT environments have not yet been widely evaluated.

This thesis assesses the IoT Security Testing Guide as a methodological basis for testing the security posture of a consumer IoT device through an empirical case study. A comprehensive security evaluation of a commercially available consumer IoT product was performed. This assessment included all layers of the device's ecosystem, covering its hardware, firmware, wireless communication, mobile application, and cloud-based infrastructure. The testing procedure is based on categories and test cases provided by the IoT Security Testing Guide and expanded by aspects of other established methodologies where necessary. To comply with legal requirements, all conducted activities were within a formal agreement between the manufacturer and the author.

The results of this research demonstrate that the IoT Security Testing Guide offers a systematic and extensive structure, especially enabling comprehensive coverage of attack surfaces. Several security vulnerabilities were discovered during the empirical case study, including a critical flaw in credential handling. Additionally, practical limitations of the assessed framework are highlighted, including the need for individual test case interpretation, limited guidance for prioritization or workflows, and ambiguous test case scopes.

Overall, this thesis concludes that the IoT Security Testing Guide is a valuable framework suitable for consumer IoT security assessments, when combined with tester expertise and interpretation. The results contribute empirical understanding of the limitations and strengths of current methodologies of IoT security testing and offer guidance for their further development.

This thesis is written in English and is 85 pages long, including 7 chapters, 9 figures and 16 tables.

List of abbreviations and terms

API	Application Programming Interface
BHM	Baby Health Monitor
BLE	Bluetooth Low Energy
CIA	Confidentiality, Integrity, Availability
CVSS	Common Vulnerability Scoring System
CVD	Coordinated Vulnerability Disclosure
DDoS	Distributed Denial of Service
DNS	Domain Name System
GDPR	General Data Protection Regulation
IETF	Internet Engineering Task Force
IoT	Internet of Things
ISSAF	Information Systems Security Assessment Framework
ISTG	IoT Security Testing Guide
LoA	Letter of Authorization
LTE	Long-Term Evolution
MAC	Medium Access Control Address
MCU	Microcontroller Unit
MQTT	Message Queuing Telemetry Transport
NDA	Non-Disclosure Agreement
OSSTMM	Open Source Security Testing Methodology Manual
OTA	Over-the-Air
OWASP	Open Worldwide Application Security Project
PTES	Penetration Testing Execution Standard
SoC	System On A Chip
SWD	Serial Wire Debug
TLS	Transport Layer Security
UART	Universal Asynchronous Receiver / Transmitter

Table of contents

List of figures

List of tables

1 Introduction

With the rapid adoption of Internet of Things (IoT) devices in the consumer market, the presence of interconnected systems in households has expanded, leading not only to increased convenience but to new security considerations as well. These devices increasingly handle sensitive personal data and often interact with cloud services, thus ensuring their digital security became a significant requirement for customers, manufacturers, and cybersecurity researchers. Nevertheless, while other cybersecurity domains established methodological standards, security assessment practices for IoT systems remain fragmented. As a result, performing structured, comprehensive, and reproducible penetration tests becomes challenging.

This thesis aims to address these challenges by evaluating a recent IoT security testing framework through its application onto a real consumer IoT device in an empirical case study. The following sections describe the initial motivation for this thesis, outline the addressed problems, and define the objectives, the scope, and scientific contributions of this work.

1.1 Research Background and Motivation

As mentioned above, the security of consumer IoT devices becomes increasingly relevant. Therefore, understanding the wide technological landscape of IoT and the associated security considerations is fundamental for contextualizing this thesis' methodological focus. The subsequent sections outline significant developments in the domain of consumer IoT devices and review the background of why approaches for structured and reproducible security testing are needed.

1.1.1 The Rise of Consumer IoT Devices

A "smart" fridge, a "smart" door lock, a "smart" watch or a "smart" baby monitor – many everyday electronic devices became "smart" in the last decade. In most cases calling a product "smart" is a more consumer-friendly way of defining it as an Internet of Things

device. Eurostat, the official statistics organization of the European Union, defines the Internet of Things as a connection of devices equipped with sensors, processors, software, and potential other technologies enabling data exchange with other systems [1]. In the consumer domain, IoT includes devices such as different smart home products (cameras, door locks, temperature controls, lights, kitchen appliances) or wearables (watches, rings, glasses) [2]. A statistic published by Statista for a report on the global consumer IoT market [3] shows a 300% increase in the number of consumer IoT connections, growing from 3.14 billion connections in 2018 to 12.57 billion in 2025. The statistic also suggests a further forecasted rise to 20.3 billion connections in 2028. Through the increasing popularity of connected devices in everyday life, new digital interfaces are being introduced into personal spaces.

The widespread appeal might be strongly driven by the benefits in convenience, comfort and shrinking margins between smart and non-smart products. The average light bulb on Amazon Germany costs around 1.85€ per piece[1], while some Wi-Fi controlled smart bulbs can be found for around 3€ per bulb[2]. For many people a small price increase can be worth the added functionality, often simplifying tasks or making them more accessible. Often, new consumer IoT products can also be cheaper than traditional professionally installed solutions.

The increasing market share also brings consequences for the products themselves. In 2023, consumer IoT accounted for 22% of the 848 billion USD global IoT revenue [4]. Manufacturers are trying to keep up with every new IoT product, prioritizing quick development and cutting costs to keep the products affordable for consumers. This often leads to minimal security development, hardening or testing.

1.1.2 Importance of Security and Privacy in IoT Devices

IoT products often face unique difficulties, especially in the area of security, due to their complex mix of architectures. IoT ecosystems are characterised by a constantly increasing number of connected devices and sensors blending into everyday life, requiring little interaction from users [5]. Many IoT devices feature different sensors, processing units, actuators and different communication interfaces like Wi-Fi, Bluetooth or other wireless

[1] https://www.amazon.de/s?k=gl%C3%BChbirne
[2] https://amzn.eu/d/fMKNwd7

protocols. Moreover, IoT services are developing into increasingly dynamic, decentralized, and complex services, which results in a reduction of the robustness of traditional security boundaries. Similar to the Internet, it is expected that IoT architectures organically develop through independent contributions, rather than a coordinated standard. As a result, security remains a critical challenge in this evolution. [5]

Additionally, the general attack surface is also increased in comparison to non-IoT products. Especially consumer IoT devices provide comfortable control via smartphone apps and location-independent data management through cloud-based services [3]. A single product therefore provides attack surfaces on four different levels: hardware, software, cloud and mobile. Due to this large area of possible attacks, manufacturers face difficulties securing their product on all ends and often focus on the easiest (or cheapest) points to fix.

Concurrently the privacy aspect of consumer IoT products cannot be overlooked as well. Often found as smart-home devices, they are deeply connected into our everyday life, collecting personal data. This data can range from the timestamps when consumers leave their house to highly sensitive biometric, behavioural or health information. In the context of this thesis, a commercially available consumer IoT device, hereafter referred to as Baby Health Monitor (BHM), is assessed. The BHM consists of a base station and a sensor unit, which collect medical information of a minor and family habits, which according to the General Data Protection Regulation (GDPR) qualifies as sensitive information and requires strong protection [6].

1.2 Problem Statement

The average time-to-market for IoT products increased by 80% between 2020 and 2023, according to a report published by D. Paraskevopoulos. This increase is reasoned with the change of global regulations and product complexity [7]. Despite this increased development time, security testing seems to be often overlooked. In a statistic collected by Forrester and published in a Verizon report, IoT devices were shown to be the most common target of external cyber-attacks, with 33% of all recorded attacks of 2022 [8].

IoT baby monitors and other health-related smart devices often combine audio, video, sensors and other technologies. These complex systems can lead to security

vulnerabilities, as presented in research done by SEC Consult about a specific smart baby monitor. In their research they found multiple critical vulnerabilities, including video access to any baby monitor of that model [9]. Another security advisory, published by Bitdefender, shows a different brand of smart baby monitors having multiple security vulnerabilities as well, again including gaining access to the live feed of the camera [10]. These examples show, that IoT baby monitors might pose a significant risk for families and their sensitive information.

As the Internet-of-things is still a relatively new market, many processes for security testing have not been established yet. While other areas, such as web application security testing, have standardized workflows and testing strategies, in IoT penetration testing it is not yet common. Without proper generalized processes, security experts are forced to depend on their individual knowledge and preferences. This lack of standardization may reduce the reproducibility of tests, therefore hindering independent scientific evaluation or comparison of results.

The Open Worldwide Application Security Project (OWASP) has introduced the IoT Security Testing Guide (ISTG) in March 2024 to solve that problem and to provide a structured approach to IoT penetration testing [11]. While other OWASP guides or frameworks have become commonly used resources in the general cybersecurity field, the ISTG has yet to be tried and tested by professionals. At the time of writing, a Google Scholar search for the term "OWASP ISTG", returned just four results[3] , showing that the framework has yet not gained wide academic attention.

This gap is addressed by applying the ISTG framework in a black-box penetration test of a real consumer IoT health device for baby monitoring. Through this case study, the author evaluates the framework's usability, completeness and reproducibility. The penetration test is documented in detail to provide both, an academic contribution, and a practical reference for other security professionals utilizing the ISTG framework.

[3] Google Scholar search for "OWASP ISTG", conducted on 20 June 2025: https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=ISTG+owasp&btnG=

1.3 Research Objectives and Questions

The main goal of this thesis is to evaluate the OWASP ISTG as a methodology for penetration testing of consumer IoT devices. This is achieved by applying the framework to a real-world case study. A complete black-box penetration test of a smart baby monitor, provided by the manufacturer, and its interfaces is executed and documented.

The research aims to contribute both academically and practically by documenting the testing process in a structured format, allowing reproduction of the results. This includes identifying strengths and limitations of the ISTG framework, evaluating its applicability in real-world situations, and determining whether it offers sufficient coverage across all areas such as hardware, firmware, communication, and application layers.

In addition to evaluating the framework itself, a practical testing guide is produced that can be reused or adapted by other researchers and security professionals working with similar consumer IoT devices. This allows for independent validation and reproduction of the results and offers scientific guidance.

To achieve these goals, the following research questions are addressed:

1. How effectively does the OWASP ISTG framework support a structured and reproducible penetration testing process for consumer IoT devices?
2. Which strengths and limitations of the ISTG framework become apparent when applied in a real-world black-box penetration test?
3. To what extent can the documented testing process serve as a reusable reference or manual for practitioners working with similar IoT devices?

1.4 Scope and Limitations

The subject of the case study of this thesis is a commercially available consumer IoT baby health monitor, hereafter referred to as BHM (Baby Health Monitor). It consists of two devices – a wearable sensor and a base station. The latter also functions as charging station for the sensor and handles the communication to the cloud service. The manufacturer provides a corresponding mobile application, which is tested on an emulated device. The penetration test is conducted as a so called “black box” penetration test [12], meaning that no internal documentation, customized access or source code has

been provided by the manufacturer. This is used to simulate an external attacker with no provided information. The testing covers all aspects of the BHM and is separated into hardware, firmware, local communication, the mobile application, and the cloud service.

The OWASP IoT Security Testing Guide is used as the primary methodology for the penetration test execution process. The framework provides structured testing across multiple layers of the system, such as the ones described above. Privacy aspects of the results of the security assessment will be analysed if relevant personal data is exposed.

Due to the ISTG covering all kinds of IoT products, some test cases may not apply to the BHM. The results are therefore not fully generalizable, as they are specifically related to the BHM. For non-applicable test cases, a short recommendation is included to create a full practical reference.

Security testing conducted during this research respects ethical boundaries. All interactions with back-end services or personal data have been performed with explicit consent of the company and under strict contractual agreements.

1.5 Scientific Contribution

This thesis is one of the first published academic evaluations of the OWASP ISTG framework. It is therefore an important scientific contribution to OWASP and security professionals alike. The results grant feedback into limitations, applicability, and coverage to OWASP, while security professionals are provided with a practical reference to apply the ISTG in their own research or work. In contrast to classical penetration testing, this work evaluates the practical and methodological performance of the ISTG, rather than focusing on the security of the device itself.

The practical study additionally results in the first penetration test of the BHM. By using a structured, reproducible method the results will provide important initial insight into the security and development of the device itself. This not only gives the manufacturer the opportunity to make their device more secure but also creates a baseline for further penetration tests of the BHM.

The resulting thesis bridges the gap between structured scientific examinations and practical penetration testing. It supports academics and practitioners alike by providing a

hybrid approach of technical and practical steps and reproducibility through the structured methodology.

1.6 Thesis Structure

The thesis is structured in nine chapters. Chapter 1 provides an introduction into the general topic and describes the motivation, scope, and objectives of this work. Chapter 2 presents the background and related work, analysing the IoT security landscape, existing security assessment frameworks, and relevant academic papers. Chapter 3 describes the methodology used in this research, including evaluation metrics and the principles the author followed while designing this study. Chapter 4 presents the technical methodology for the preparation and structure of the conducted security assessment. Chapter 5 then documents the empirical execution of the case study, the testing process, observed behaviours, and collected evidence. Chapter 6 evaluates and analyses these findings and assesses ISTG's suitability through discussing its strengths and limitations. Chapter 7 then synthesizes the analysed information and presents proposed recommendations. Chapter 8 describes possible future work suggested by the author. Lastly, Chapter 9 summarizes the objectives and observations, answers the research questions and underlines significant conclusions.

2 Background and Related Work

The conceptual foundation of this thesis is provided by the understanding of the security landscape of consumer Internet of Things (IoT) devices and current methodologies used to test them. This chapter outlines common device architectures and threat vectors first, followed by IoT-specific challenges hindering the reproducibility of penetration tests in consumer IoT-environments. Then different security-testing frameworks are reviewed such as the Penetration Testing Execution Standard (PTES) [13], the Open Source Security Testing Methodology Manual (OSSTMM) [14], and the Information System Security Assessment Framework (ISSAF) [4]. This is supported by comparative evaluations from recent studies and academic works. Lastly, other related academic work and the research gaps are highlighted that create the basis for this thesis.

2.1 IoT Device Security Landscape

IoT devices often combine embedded hardware, wireless communication, cloud services, and mobile applications, which result in complex ecosystem with multiple security-relevant layers. The following sections describe typical architectures and attack vectors of IoT ecosystems to provide context for understanding their corresponding security risks.

2.1.1 Common Architectures in Consumer IoT

Ecosystems of consumer Internet of Things devices generally consist of a multilayered architecture, integrating device, network, cloud and application components [15]. The device layer usually integrates sensors and actuators on embedded hardware together with microcontrollers with specific firmware. For communication of the collected data, different technologies are used such as Bluetooth Low Energy (BLE), Wi-Fi or Zigbee. Data exchange between the device and different types of gateways is handled by the network layer, using protocols such as the Message Queuing Telemetry Transport (MQTT) or Constrained Application Protocol. The devices themselves usually only

[4] https://untrustednetwork.net/files/issaf0.2.1.pdf

collect data – the aggregation, analysis and storage are often done in the cloud layer, offered as platform-as-a-service models or self-hosted. To control or update the devices, the application layer provides user interfaces such as mobile applications or web interfaces [16]. This data flow is summarized in Figure 1 below.

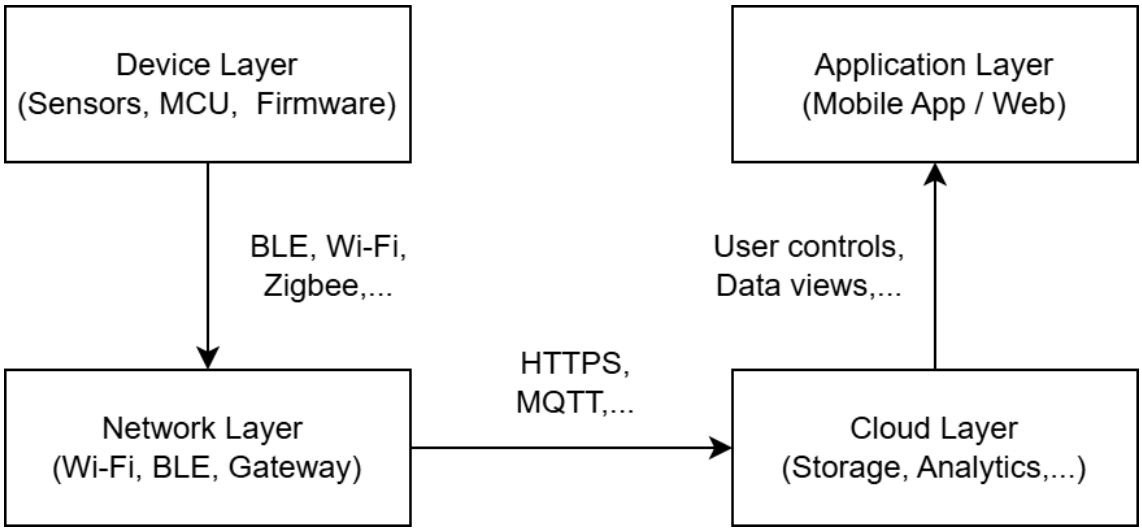


Figure 1. Typical Data Flow and Communication Channels of IoT devices.

Although this layered design creates a flexible and scalable infrastructure, it also introduces a specific complexity to the security. Every system is as strong as the weakest link – in this case the compromise of one layer, could lead to exploitation of the whole infrastructure. Therefore, each layer must be configured and protected properly. The IEEE IoT Ecosystem Study emphasizes, that widespread adoption of IoT devices requires consistent standards, ensuring interoperability and efficiency of devices and services [16]. However, unified security management is complicated by the use of proprietary ecosystems with very limited transparency by most consumer IoT products.

2.1.2 Typical Attack Surfaces and Threats

The diversity of consumer IoT infrastructures creates an unusually broad attack surface. According to a report published by Fortinet [17], any of these layers can lead to vulnerabilities:

1. **Hardware**: accessible debugging interfaces (UART, JTAG), unprotected storage or generally insecure electronical design.
2. **Firmware**: plaintext credentials, vulnerable libraries or unvalidated input.
3. **Network**: unencrypted traffic, weak authentication or insecure pairing processes.

4. **Cloud / Application**: misconfigurations, insecure databases/APIs or inadequate access control.

Multiple studies show that these vulnerabilities are widespread. Williams et al. reported in a large-scale survey, that almost 13% of analysed devices showed at least one vulnerability [18]. The same study also shows concerns regarding data transmission, with webcams and smart-TV modules having exposed unencrypted services like Telnet or SNMP. [18]

While industrial IoT-devices are often combined with firewalls and other security installations, low-cost consumer devices are especially at risk. Murat et al. found that low-cost IoT devices often implement basic security mechanisms to save development and production cost, which leads to vulnerabilities affecting consumers [19]. This leads to a structural rather than accidental shift in insecurity for consumers. Because of the big variety and dependency of vulnerabilities across all layers, security testing must consist of the entire stack to be effective. This creates a challenge, addressed in later sections of this chapter.

2.2 Challenges in Consumer IoT Penetration Testing

As conventional IT or web-application penetration testing mostly focus the cyber space, IoT security assessments differ significantly through the complexity of the products and the connection between cyber space and physical space. The combination of different hardware modules, specific firmware and hybrid network connections creates technical and methodological obstacles complicating reproducibility and scientific testing. This section highlights these challenges and forms the grounds of why a structured framework-based methodology such as the OWASP IoT Security Testing Guide is necessary.

2.2.1 Technical Barriers

Most consumer IoT devices contain a varied architecture and proprietary ecosystems hindering consistent assessment processes. Using microcontrollers with limited resources and storage running product-specific operating systems, manufacturers make it difficult to use standardized security or forensic tools. The firmware is usually stored in read-only flash memory and further secured through secure-boot mechanisms. This forces

researchers or testers to extract the binary firmware by accessing low-level hardware interfaces like universal asynchronous receiver-transmitter (UART)[5] or Serial Wire Debug (SWD)[6] [20]. This access is mostly only possible by physically tearing down the device, which can lead to irreversible damage and further hinders testing.

If it is possible to extract the firmware, it commonly contains custom bootloaders or binary blobs lacking debug information or symbols. Therefore, disassembly and analysis cannot rely on documented interfaces and often depends on heuristics making it more difficult to recognize third-party libraries and custom code. Additionally, these analyses require specific technical knowledge by the tester.

Another layer adding complexity is the network aspect of IoT devices. Typical consumer products use lightweight communication protocols like MQTT, Zigbee or Bluetooth Low Energy (BLE) that either do not support robust authentication or encryption at all or do not use it by default [21]. The default MQTT broker configurations may transmit data unencrypted or even allow unauthenticated connections [21]. Testing different protocols also required specialized tools and transceivers, further complicating accessible and repeatable testing methodologies.

Lastly, connectivity to cloud services adds another layer of complexity. Used APIs are often provided through global cloud services bound to the manufacturer's user credentials. Security testing must be conducted in a way to simulate legitimate user behaviour but also preventing accidental disruptions to cloud services.

2.2.2 Methodological Limitations

The second considerable limitation concerns the lack of standardisation and reproducibility of IoT testing methodologies. In comparison to web or network security, almost no processes or evaluation methods are universally recognized for IoT penetration testing. In practice, security experts are adapting existing frameworks such as the PTES or OSSTMM to IoT infrastructures, even though these methodologies are focused on logical rather than embedded systems. As Sarker and Deraman observed, most security

[5] https://www.analog.com/en/resources/analog-dialogue/articles/uart-a-hardware-communication-protocol.html
[6] https://docs.nordicsemi.com/bundle/nwp_037/page/WP/nwp_037/swd_if.html

assessment teams used self-developed methodologies, highlighting the need for a standardized approach [22].

Because of missing unified methodologies, reporting and tool usage is inconsistent between different security testers. Shanley and Johnstone demonstrated that a significant number of proposed frameworks were either labelled incorrectly or provided only limited conceptual basis and coverage, which resulted in restricted applicability [20]. These inconsistencies lead to a lack of reproducibility as two researchers testing the same device may use completely different processes and tools, resulting in different outcomes. In 2009, J. Frankland wrote in an article that when security assessments are conducted without clear methodological structures, they may lead to fewer identified vulnerabilities, which results in unreliable conclusions about the security of the assessed system [23]. This statement shows that complexity and non-standardized testing is not only complicated to execute but may also impact the reliability of the results. Without a clear system of IoT-specific testing phases, there is a risk of academic studies devolving into subjective reports rather than scientifically valid experiments.

A different methodological difficulty is the balance between testing depth and device safety, as aggressive exploitation could permanently disable IoT devices. As a result, security professionals often limit their testing to passive or non-destructive techniques such as network sniffing, firmware extraction and static analysis. These methods provide limited visibility of the systems runtime security. It remains a current problem in IoT research to balance the depth of testing, the device integrity and vendor cooperation. [19]

Lastly, there is also a lack of standardised scoring or classification schemes for risks. While most security professionals rely on the Common Vulnerability Scoring System (CVSS), its properties (e.g. attack vector, privileges required) are not covering the complex infrastructure of IoT devices and its physical-layer or embedded risks. IoT vulnerabilities are often chained and multi-layered, which cannot properly be expressed with CVSS scoring. As a result, analysis across different case studies and quantitative assessments of the effectiveness of frameworks are hindered.

2.2.3 Legal and Ethical Limitations

IoT penetration testing also involves the consideration of legal and ethical boundaries of permissible research. As many consumer IoT infrastructures contain vendor-managed

cloud services processing personal or other sensitive data, penetration tests of these environments without explicit authorization may violate data-protection laws. Therefore, it is important to integrate the ethical principle of responsible disclosure into methodologies. The Internet Engineering Task Force (IETF) published the RFC 9116 as a result. It contains a standardized, machine-parsable method of reporting vulnerabilities to vendors [24]. Additionally, the ISO 29147 standard defines a Coordinated Vulnerability Disclosure (CVD) encouraging collaboration between manufacturers and security professionals [25].

It is also imperative to gain explicit consent for a specific testing scope. Many established frameworks such as the PTES or OSSTMM begin with a pre-engagement phase, defining authorised targets, allowed techniques and how to handle data and discovered vulnerabilities [13]. In the case of IoT penetration testing this is especially important, as many devices are relying on third-party infrastructure. For example, testing a vendor's API endpoint or database may accidentally lead to access of customer data or an impact on availability for other services. Therefore, all methodologies should include a written letter of authorisation (LoA), which specifies the scope of the penetration test and cooperation with the manufacturer regarding non-disclosure agreements (NDAs).

2.3 Frameworks for Security Testing

To enable repeatable, transparent und structured penetration testing processes, many different frameworks were published in recent years. These frameworks provide testers with workflows, vocabulary and sometimes even reporting templates [13]. With these elements it is possible to create reproducible scientific results instead of ad-hoc penetration tests. In this section four of the most widely used frameworks and standards are reviewed: the Penetration Testing Execution Standard (PTES), the Open-Source Security Testing Methodology Manual (OSSTMM) [14], the PETIoT framework [26] and the OWASP IoT Security Testing Guide (ISTG) [11]. Analysis of their completeness and domain coverage is provided with comparative studies from Shanley & Johnstone [20] and Sarker & Deraman [22]. Their suitability for consumer IoT environments is also discussed.

2.3.1 Penetration Testing Execution Standard (PTES)

The PTES was published in 2009 as a way to standardise the process of professional security assessments across different organizations [13]. It defines seven separate but consecutive phases:

1. Pre-Engagement Interactions
2. Intelligence Gathering
3. Threat Modelling
4. Vulnerability Analysis
5. Exploitation
6. Post-Exploitation
7. Reporting

The authors included objectives, deliverables and quality indicators in each phase promoting uniform testing. In practice, the PTES is valued for its flexibility and general guidance but was never intended to be used for IoT infrastructures. Shanley & Johnstone note, that due to a lack of necessary attributes, incomplete documentation, or loose structure it can neither be considered a methodology nor a framework [20]. The standard is based on the assumption that access to traditional network endpoints, operating systems or user accounts is possible, which is rarely the case in consumer IoT ecosystems. Therefore, using the PTES for IoT devices would require specific adaptation, especially in the area of intelligence-gathering, including hardware and firmware specific procedures. These could contain interface enumeration, binary reverse engineering and disassembly and UART analysis.

While these limitations exist, PTES still remains a relevant source of methodology guidance, as its pre-engagement and reporting phases are outlining strong communication practices. In this thesis, PTES is serving as a reference for the process maturity of ISTG, comparing completeness and reproducibility.

2.3.2 Open Source Security Testing Methodology Manual (OSSTMM)

The Institute for Security and Open Methodologies published the first OSSTMM in 2000, with the current version being released in 2010. It is a comprehensive manual for operational security assessments of information systems [14]. The scope of the OSSTMM

is separated in three classes: Telecommunications and Data Networks security, Physical and Human Security and Wireless Security, all separated into five channels respectively. In comparison to PTES, which focuses on a process flow, the OSSTMM is defining quantitative metrics, the so-called Operational Security Metrics (OSM). They measure authenticity, confidentiality, integrity, availability and non-repudiation within each of the channels.

Shanley and Johnstone describe the OSSTMM as a primarily audit-focused approach offering less depth than the ISSAF[20]. Additionally, they note its usefulness as a security assessment reference while it does not provide concrete processes, which leads to the need for interpretation by the tester [20]. For the context of IoT devices, OSSTMM's focus on auditing and assessor skill shows a strong dependence on expertise. The physical and wireless classes conceptually align with consumer IoT infrastructures, but the general scope is oriented onto enterprise-contexts, limiting its applicability to embedded systems. As a result, OSSTMM provides valuable measurement principles but lacks guidance for processes of consumer-IoT penetration testing.

2.3.3 Penetration Testing for the Internet of Things (PETIoT)

In 2023 Bella et al. have published the PETIoT framework, specifically addressing the need for an IoT assessment methodology [26]. The authors position PETIoT as a new kill-chain-based methodology, specifically for IoT security assessments. Their methodology is separated into six phases aligned with IoT-specific components: experiment setup, information gathering, traffic analysis, vulnerability assessment, exploitation and fixing.

In their research the authors conducted an empirical test on an IP camera and discovered three vulnerabilities, which were responsibly disclosed and fixed by the manufacturer [26]. With this case study, they provided first proof of the empirical applicability of PETIoT. Similar to one of the goals of this study, PETIoT contains practical guidance including commands and tools to further enable reproducibility and offer guidance for other professionals.

2.3.4 OWASP IoT Security Testing Guide (ISTG)

The Open Worldwide Application Security Project (OWASP) has published various testing guides for different areas of cybersecurity such as the Web Security Testing Framework [7] or the API Security Testing Framework[8]. In 2024 they published the IoT Security Testing Guide (ISTG), recognising the importance of an IoT-specific methodology [11]. The ISTG targets the full stack of IoT devices: hardware, firmware, communication, and application. They provide a list of test cases for each category, including objectives, required access levels and remediation steps. For example, the firmware category includes a test case for insufficient firmware update signatures, usage of outdated software or disclosure of user data. The full categorization of domains is presented in Figure 2 below.

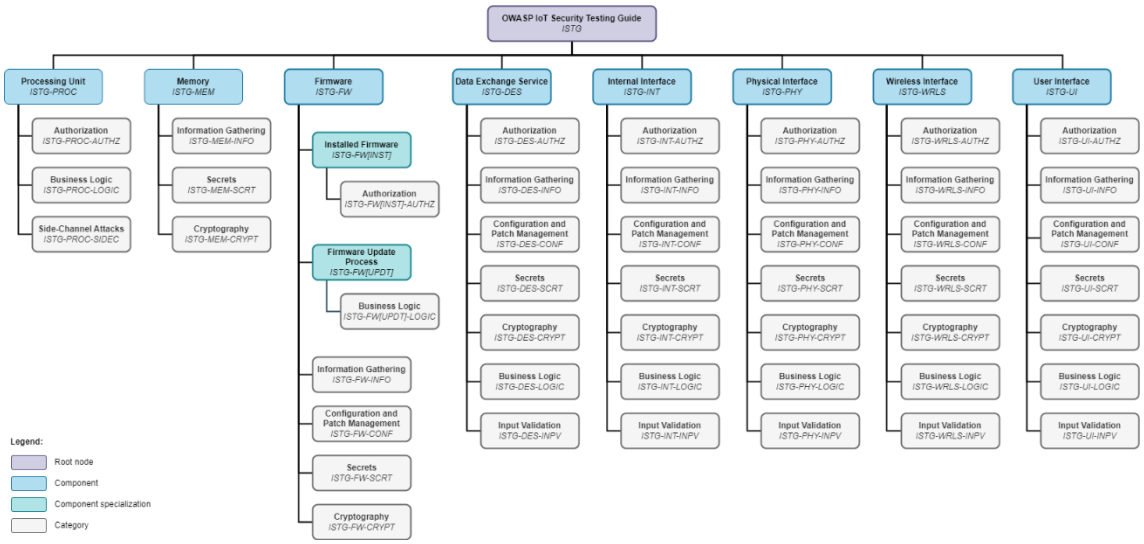


Figure 2. Overview of ISTG's categories. [11]

The authors note that the guide is not a complete, prescriptive manual for penetration testing IoT devices, but rather functions as a set of test cases, which cover a wide range of IoT domains and components [11]. Their focus lies on first creating a modular framework that can be applied to many different IoT ecosystems to ensure comparability. They also acknowledge, that in the future the guide is supposed to grow and encompass more detailed sections for specific technologies. [11]

[7] https://owasp.org/www-project-web-security-testing-guide/
[8] https://owasp.org/www-project-api-security-testing-framework/

As it was published just a year before the start of this research, it lacks comparative evaluations or peer-reviewed case studies. Therefore, this work will contribute to academic understanding and practical improvement of the ISTG framework by evaluating ISTG through a case study.

2.4 Related Work

With research on IoT penetration testing being focused on isolated device categories such as smart home appliances, industrial controllers or healthcare systems, the scientific landscape stays fragmented without a unified approach. The complexity and diversity of hardware, firmware and communication interfaces have led to most security experts using ad-hoc methodologies adjusted to the specific device. This chapter reviews current academic and industry practices highlighting common vulnerabilities and illustrates limitations in methodologies that motivate this thesis.

2.4.1 Early Large-Scale Analyses

One of the first large-scale vulnerability assessments of consumer IoT devices was published by Williams et al. in 2017 [18]. In this research, the authors discovered that "20,237/156,680 (12.92%) consumer IoT devices in our collection have 'Critical', 'High', 'Medium', or 'Low' risks". While webcams and printers accounted for most of the high-risk ('Critical', 'High' and 'Medium) vulnerabilities, smart TVs mostly showed medium level findings. These results show that insecurities in IoT devices are not limited to isolated categories of products but rather a systematic issue of weak default configurations and short update timeframes. [18]

In 2016 the famous Mirai botnet has infected up to 600.000 systems, mostly IoT devices, further demonstrating the real-world consequences of insecure and mass-produced IoT environments [27]. After its discovery researchers detected that the botnet infected around 65.000 IoT devices during the first day of an attack [28], by exploiting insecure Telnet credentials. The botnet was reportedly used for Distributed Denial of Service (DDoS) attacks, causing large-scale outages at domain name resolution (DNS) services, telecom providers and game servers, totalling an estimated 15.000 attacks. Further research of the malware source code and other traces by Antonakakis et al. showed that primarily low-cost consumer devices from notable manufacturers such as home routers or cameras were

targeted. This demonstrates that even big manufacturers often lack security mitigations to defend against attacks like Mirai [27].

These studies collectively highlight why reproducible vulnerability-assessment processes and methodologies are needed, so they can be applied systematically before IoT devices reach the global market.

2.4.2 Consumer IoT and Smart-Home Case Studies

With the raised awareness for IoT security, research shifted towards assessing specific device categories. Murat et al. [19] analysed a variety of low-budget smart-home devices finding a range of vulnerabilities including unencrypted Wi-Fi communication, unprotected firmware update mechanisms and insecure cloud APIs. The authors concluded that further hardware and firmware security testing should be integrated into production and certification processes. [19]

Another empirical study published by Zhou et al focused on the interactions between IoT devices, their mobile applications and associated cloud services in smart-home contexts. They acknowledged that prior research of smart home security was done in a fragmented way, analysing aspects separately. The authors mention, the lack of systematic investigations into the complex interactions between IoT ecosystem components such as devices, cloud services, and mobile applications [29]. Their findings revealed widespread security flaws in authentication processes between devices and cloud services allowing hijacking of IoT systems through cloud interfaces. Even though this study was one of the earliest comprehensive analyses of IoT infrastructures, it mainly focused on the discovery of vulnerabilities rather than the development of a reproducible assessment methodology. This further highlights the current ad-hoc approach instead of a formalised framework.

As mentioned in **Error! Reference source not found.**, Bella et al. introduced the PETIoT f ramework, which they empirically validated on a TP-Link Tapo C200 camera [26]. As a result, three zero-day vulnerabilities were discovered and responsibly disclosed. PETIoT signifies a valuable step towards structured IoT penetration testing methodologies, aligning testing activities with the kill-chain concept. Yet, it still depends on individual expertise and awaits further scientific and independent validation.

2.4.3 Healthcare and Wearable IoT Devices

Besides smart-home devices, healthcare and consumer-health devices form a growing proportion in the IoT market. However, their sensibility of processed data in addition to potential health risks lead to an even stronger need for security. Islam et al. analysed the landscape of IoT for health care and argued that healthcare IoT devices may be an attractive target for attackers [30]. The authors also highlight the importance of addressing confidentiality, integrity, authentication and availability, which is further acknowledged by Nezhad et al. [31]. In their research, they identified fourteen security requirements marked as prerequisites for ensuring the CIA (confidentiality, integrity, availability) triad. They include aspects such as resilience, access control, accessibility and key management, which can all be verified by thorough security assessments [31].

In 2020 Somasundaram and Thirugnanam published a paper analysing security challenges in healthcare IoT security [32], specifically focusing on implantable and wearable devices. They found that multiple categories of medical IoT devices such as implantable devices or radio frequency identification tags are exposed to significant security risks [32]. Using their empirical risk assessment methodology, the authors presented a 95% risk factor of distributed-denial-of-service (DDoS) attacks on implantable cardioverter-defibrillators and a 55% risk factor through authentication vulnerabilities of wireless insulin pumps. They argue that device-level protection such as cryptography, secure firmware updates and mutual authentication remains a critical requirement for safe usage of medical IoT products. [32]

A case study published by Rapid7, an US-based cybersecurity company, discusses the security assessment of a wide range of IoT baby monitors. They disclosed ten previously unknown vulnerabilities and conclude that all tested devices contained several vulnerabilities [33]. These include cleartext API communication, unencrypted video storage, and ways to bypass authentication and gain system access through insecure default credentials or UART access. Hilding and Nordström analysed the Motorola MBP855Connect baby monitor, also finding UART access and hard-coded credentials in the corresponding mobile application [34]. These papers illustrate that no area of IoT devices is exempt from common vulnerabilities and expose sensitive environments and information.

2.4.4 Automation and Machine-Learning Approaches

With the rise of artificial intelligence, efforts have been made to automate vulnerability assessments. Kaksonen et al. researched the potential of automation for the ETSI TS 103 701 specification [35] using open-source tools. Their proof-of-concept revealed that around half of the selected tests can be conducted with basic network security tools, while advanced tooling can result in almost three quarters of the test set being automated [36]. According to the authors for a full coverage of tests it is necessary to develop custom tooling. Empirical testing of a commercial IoT gateway achieved also almost 50% coverage of network-specific test cases and determined that automated checks can also verify compliance during product development. [36]

A different approach of automation is continuous fuzzing, a technique where invalid or unexpected data is provided to applications, and errors or crashes are monitored. Google's OSS-Fuzz project [37], launched in 2016, resulted in over 10.000 resolved vulnerabilities in over 1000 projects [37]. These numbers illustrate the impact of long-term automated testing, especially for common libraries such as libpng[9] or mBedTLS[10], responsible for handling PNG files and TLS encryption respectively. However, as fuzzing requires extensive computational resources and mainly targets software components, its practicality for embedded devices is limited.

As comprehensive IoT penetration testing requires physical interaction with the assessed device, automation attempts are limited to performing analyses with specific auditing tools, but practitioners will still be required to disassemble, analyse and connect the device for an automation to be executed. Therefore, the actual benefit of these efforts is reduced, due to the need for expert knowledge to complete the conditions required for automated assessments.

[9] http://www.libpng.org/pub/png/libpng.html
[10] https://github.com/Mbed-TLS/mbedtls

2.5 Literature Gap and Justification

The previous sections are demonstrating that a significant gap exists in methodologies for vulnerability discovery and development of standardized frameworks. While many studies analyse specific devices, few are assessing the methods used and whether they are scientifically comprehensive and reproducible. This section combines the insights from previous scientific work and explains the limitations this thesis aims to address.

2.5.1 Fragmented and Inconsistent Methodologies

Analyses of existing literature shows that IoT security assessments remain unstandardised. Security professionals use various definitions of test phases, heterogeneous tools, and individual reporting procedures [22]. Such diversity prevents scientific comparison between different studies and limits a collective scientific progress.

While one researcher might include the enumeration of physical interfaces and the teardown of the device in the reconnaissance phase, another defines this phase exclusively as network scanning. Results cannot be independently replicated without a common vocabulary and taxonomy. Subsequently, research in IoT security provides many device-specific findings while contributing little towards a validated methodology. The OWASP ISTG provides a structured and general catalogue of test-cases, addressing this limitation by proposing standard layers.

2.5.2 Limited Empirical Validation of Frameworks

Another gap concerns the lack of framework-level evaluation. Most research in IoT security focuses on discovering vulnerabilities within devices, rather than analysing the methodology and how effectively it guides the testing process. For example, the PETIoT framework shows a promising prototype for an IoT-security methodology but was only demonstrated by the authors themselves and independent replication is missing. As the ISTG framework was published in 2024, almost no empirical validation exists, making its real-world applicability one of the research questions of this thesis.

Without validation, professionals cannot determine if ISTG's proposed structure actually improves scientific outcomes or is simply a collection of best practices already in use. With the application of ISTG to a selected consumer IoT device and the systematic

documentation of each test case and other necessary steps during a penetration test, this thesis provides necessary evidence to evaluate usability and completeness of the ISTG.

2.5.3 Neglect of Reproducibility and Documentation Quality

The lack of comprehensive documentation of methodologies is another persistent issue limiting reproducibility. While some IoT security studies provide code or tools used during the assessment, many skip intermediate steps such as the extraction of firmware or the configuration of tools.

The adapted ISO 25010 quality model proposed by Shanley and Johnstone [38] identifies six key attributes of any methodology such as usability, domain coverage and reliability. By applying these attributes to IoT security assessments the importance of clear guidance, modular processes and transparent reporting are highlighted. This thesis aims to implement this idea, resulting in comprehensive documentation and thereby contributing to the reproducibility of IoT security research.

2.5.4 Inadequate Integration Across IoT Layers

With most prior research targeting individual aspects or components of IoT devices such as firmware, network or cloud interfaces, only a few studies are analysing the complete system environment at once. The PETIoT and other frameworks focus mainly on network and firmware layers, while security assessments of healthcare devices are dealing with application and privacy aspects. Almost none are covering the entire ecosystem from hardware interfaces to cloud interactions. As IoT devices rely on the interconnection between all aspects of its system, vulnerabilities in any interface can lead to a loss of security for the whole product.

This thesis aims to adopt a full-stack perspective on IoT security assessments, integrating hardware, firmware, communication, and application components. Therefore, it aligns with ISTG's design and highlights how a multi-layer methodology can lead to discovery of vulnerabilities that focused penetration tests might overlook.

2.5.5 Ethical and Legal Considerations

In addition to a complete technical analysis, it is also important to establish a norm for ethical and legal frameworks for penetration testing. While many studies mention

responsible disclosure procedures, few discuss a direct methodology in combination with a practical study. A common approach to vulnerability disclosure can only be established by both, researchers and manufacturers.

The PTES and OSSTMM are providing specific phases for the definition of the scope, clarify authorisation, and reporting of results. This thesis will integrate these aspects into the ISTG methodology to ensure a strong ethical basis for the vulnerability assessment. The process of determining formal authorisation, the scope and limitations, and the reporting procedure together with the device vendor are documented. This demonstrates that ethical and legal considerations can and should be included in technical scientific studies.

2.5.6 Summary of Literature Gaps

The identified gaps hinder scientific progress and assessment quality alike. Table 1 provides an overview of the most relevant limitations discovered in existing literature and describes how the author of this thesis addresses them through methodological design and empirical evaluation strategies.

Table 1. Summary of Literature Gaps and their Proposed Mitigations.

Identified Gap	**Implication**	**Proposed Mitigation**
Fragmented, inconsistent IoT testing methodologies	Results not comparable or reproducible across studies	Applying the structured OWASP ISTG and documenting each test case systematically
Lack of empirical validation of frameworks	No evidence that frameworks improve testing outcomes	Performing a full empirical evaluation of ISTG through a real-world consumer device case study
Poor reproducibility and incomplete documentation	Prevents replication and peer verification	Publishing detailed methodological records
Testing limited to isolated layers (firmware or network)	Misses cross-layer vulnerabilities and dependencies	Conducting full-stack testing across hardware, firmware, communication, and application layers
Insufficient consideration of legal and ethical scope	Risk of non-compliance and reputational damage	Implementing pre-engagement authorisation and responsible disclosure procedures

By addressing these gaps, the thesis is not only justified on scientific but also practical grounds. The scientific value is provided by it being one of the first empirical evaluations of OWASP ISTG, bridging the separation between theoretical guidelines and reproducible real-world research. The results are providing empirical data on the

performance of ISTG, creating the possibility for future comparison to established frameworks such as the PTES or OSSTMM. As this thesis can also be used as a testing guide for other security professionals it also delivers a practical result that can be adopted for future assessments of consumer IoT products.

By applying the ISTG framework to a real-world device, while documenting each step, and evaluating its outcomes critically, the ISTG is validated and refined. This also promotes methodological reliability and transparency in IoT penetration testing. The following chapter describes the methodological design of this study.

3 Research Methodology

This chapter describes the methodology used in this research to evaluate the OWASP IoT Security Testing Guide (ISTG) as a framework providing structure for penetration testing of consumer IoT devices. The object of evaluation in this thesis is the framework itself, while in later chapters it is applied to a specific device through a practical security assessment. The methodology is designed to define the evaluation strategy, the assessment criteria, and measures necessary for producing rigorous, valid, and reproducible scientific work. While the specific aspects are described in the following sections, Figure 3 presents a high-level overview of the developed research methodology.

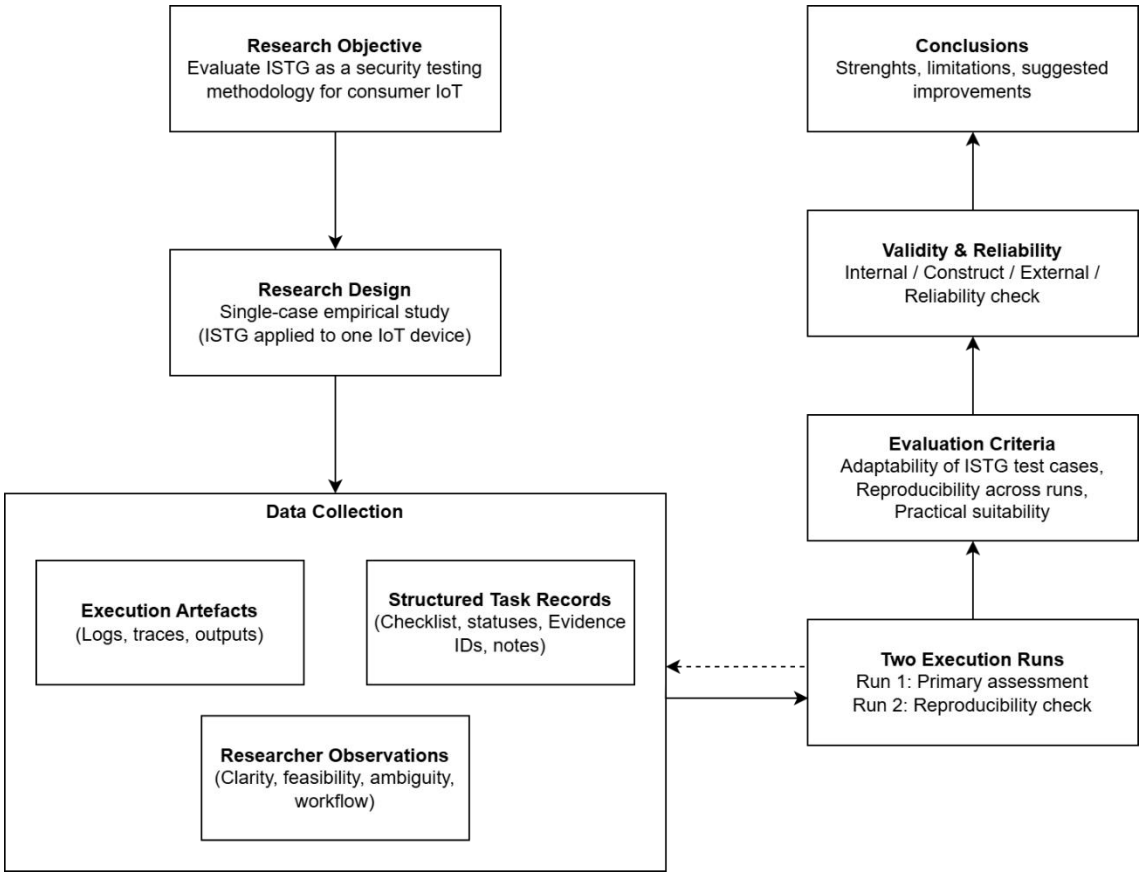


Figure 3. Overview of Research Methodology.

3.1 Research Design

An embedded single-case, empirical evaluation of the ISTG assesses its suitability as a foundation for methodological security assessments. The framework is applied in a controlled and authorized testing conditions. The capacity of the framework to support a structured, comprehensive and reproducible testing process is determined.

The goal of the case study is not to generate security relevant findings about the specific device but rather to validate:

- If the ISTG enables comprehensive and structured penetration testing processes,
- If the results of a security assessment supported by the ISTG are reproducible, and
- The behaviour of the ISTG when applied to a multi-component consumer IoT architecture.

To compare ISTG with general established penetration-testing methodologies, specific aspects are selectively evaluated against the Penetration Testing Execution Standard (PTES). The PTES is not applied directly during the security assessment but rather provides a baseline for the process structure and workflow completeness.

3.2 Empirical Data Collection

During the practical assessment, several types of data are collected, which in turn are used for the evaluation of the ISTG and are described in the following paragraphs. This data includes methodological interpretations and observations about the frameworks usability and are not limited to technical aspects.

3.2.1 Execution Artefacts

These artefacts represent relevant outputs gained during the assessment. They are used as an objective record of the behaviour during the testing steps and are the basis for evaluating if the framework supported obtaining meaningful insights. Execution artefacts include:

- Logs from communication interfaces (BLE, UART),

- Firmware update traces,
- Tool outputs from interactions, scanning or enumeration of the device, and
- Observations about the behaviour of the device under specific conditions.

Evidence IDs are used to link artefacts to the corresponding test cases, allowing transparent and traceable interpretation of the results, while providing other researchers with grounds to independently verify this author's conclusions.

3.2.2 Structured Task Documentation

The test cases provided by the ISTG are converted into a task checklist before testing. All test cases already contain multiple attributes, such as required access levels or remediation suggestions, which are included in the checklist. To create comprehensive tasks, the author adds some additional information for each test case, which are highlighted in Figure 4.

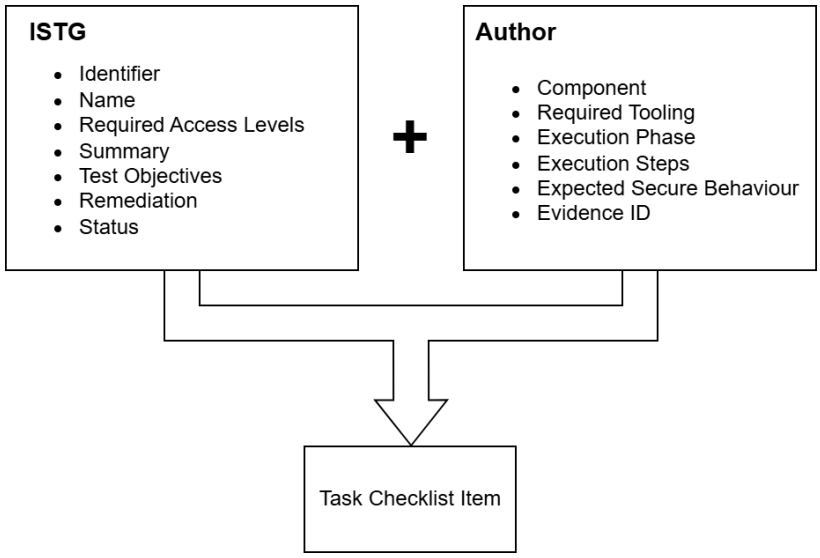


Figure 4. Attributes of Task Checklist Items.

3.2.3 Researcher Observations

Observations of the researcher's perspective are documented during the assessment. They capture subjective aspects, that cannot be described through the test case status alone, including:

- Clarity of provided test cases,
- Missing instructions or ambiguities,
- Necessary domain knowledge,
- Unrealistic test situations, and
- Limitations of the practical workflow.

As a component of qualitative evaluation, this collected information enables an interpretative evaluation of ISTG's practicality.

3.3 Evaluation Criteria

To ensure transparent assessment of the ISTG and comprehensive interpretation of the results, the evaluation is based on three dimensions, which will be described further in the following sections.

3.3.1 Adaptability

With this criterion, the author describes the degree of how well the ISTG framework can be adjusted to all attackable aspects of the chosen device. It is separated into two aspects, applicability and coverage, as the ISTG provides a set list of test cases possibly not corresponding to the infrastructure of consumer IoT devices.

Applicability

Applicability is defined as the proportion of how many ISTG test cases can be conducted on the device. With the diversity of IoT products on the market, different architectures and ecosystems may lead to irregular amounts of test cases fitting the assessment's conditions. The formula used to calculate this score is shown below in Equation (1).

$$\text{Applicability } (\%) = \frac{(\#\text{ of applicable ISTG test cases})}{(\#\text{ of total ISTG test cases})} \quad (1)$$

High applicability suggests compatibility between the frameworks and the device's structure, while a low percentage indicates that the device's infrastructure or testing conditions may not match large parts of the ISTG's test cases.

Coverage

To assess how well the ISTG covers actual components of the device, the applicability metric is combined with a coverage metric. To capture the complex, multi-layered, and manufacturer-specific device architecture, the coverage metric is introduced, which evaluates if the ISTG provides test cases for each assessable component in the device model. Components not covered by the ISTG are identified as coverage gaps. It is important to separate these gaps from the applicability calculation, as these cannot be evaluated quantitatively with an infinite amount of possible test cases. As a result, this metric is interpreted as qualitative limitation of ISTG's support for universal consumer IoT penetration tests.

In combination the two defined metrics result in a comprehensive and balanced evaluation of the ISTG's adaptability. While applicability measures practical relevance, coverage defines the completeness in relation to consumer IoT infrastructures. This allows assessing the framework from its own perspective and from the device's.

3.3.2 Reproducibility

Reproducibility is assessed by executing two independent runs of the practical assessment under the same constraints. It measures how well the ISTG supports a consistent application even under complex conditions of consumer IoT infrastructures by capturing the clarity of instructions provided by the ISTG and the stability of the task structure. Any identified differences between Run 1 and Run 2 support the discovery of areas where the ISTG may be not specified enough, too open for interpretation, or dependent on intuition instead of providing guidance.

3.3.3 Practical Suitability

This aspect evaluates how well the ISTG operates when applied by a security tester to a realistic workflow. It contains observations about clarity, required expertise, feasibility, and if it can be integrated into common practices. If the framework is too abstract it may be more difficult for practitioners to apply, while being very detailed may negatively impact the aforementioned adaptability. Therefore, practical suitability evaluates ISTG's balance between flexibility, efficient instructions, and methodological guidance.

3.4 Validity and Reliability

Multiple validity considerations supporting the credibility of the evaluation are implemented. Additionally, these measures specify the boundaries and limitations of the security assessment.

Construct Validity

Construct Validity is achieved through the combined evaluation of the coverage metric, reproducibility metric, and qualitative observations, ensuring a systematic assessment of the framework. Each of the considered metrics is representing an aspect of ISTG's purpose of guiding comprehensive security assessments, providing structure, and supporting consistency and repeatability. Through the creation of the Task Checklist and the Coverage Matrix, the author verifies that the executed procedures align with the constructs to be evaluated.

Internal Validity

This type of validity is ensured through the use of consistent assessment conditions, the documentation of processes and workflows, and static tooling configurations. Potential inconsistencies are identified and remediated in the pilot run, and environmental differences are limited through the reset phase between Run 1 and Run 2. These measures are used to ensure that different observed behaviour between runs can be linked to the device or inconsistencies in the framework design, instead of to other external conditions.

External Validity

As the evaluation is based on just one selected consumer IoT device, it cannot be used as a general result for all device types, in contrast to ISTG's design, which is meant to be applied to any IoT device. By creating transparent and documented workflows and models, this study allows other researchers to transfer the methodology to comparable devices and conditions.

Reliability

Similar to the internal validity, reliability is supported by structured documentation and reproducible workflows. Through the use of the Task Checklist, the Evidence Log, and

the Coverage Matrix, each test case is formalised, expected actions are specified, and observations are documented in a consistent way. The second execution (Run 2) allows for direct evaluation of the consistency of observations under comparable conditions, supporting the reliability metric.

3.5 Methodological Limitations

The defined research methodology is bound by certain limitations which must be acknowledged while interpreting the results. The single-case design leads to restricted generalisability of the results. Even though the chosen device represents a common modern IoT architecture, the ISTG's behaviour may be different for different hardware structures or ecosystems. Additionally, the access conditions defined in the LoA limit the security assessment by not permitting testing of private cloud services. Finally, parts of the evaluation are relying on the author's judgement, as the ISTG does not provide aspects such as execution order, testing depth, or prioritisation. Overall, these limitations do not weaken the validity of this study but define boundaries of the interpretation of the results.

4 Technical Methodology

This chapter describes the methodology used to apply the OWASP IoT Security Testing Guide to the practical assessment. In contrast to chapter 3, which outlines the research methodology, this chapter focuses on the testing workflow, the processes applied to the test cases, and the practical preparation for the assessment execution. It aims to provide a systematic and understandable documentation of how the ISTG was adapted to the chosen device, how test cases were selected, and how evidence was collected and organized.

The technical methodology is based on the structure of the ISTG and its components and inspired by the PTES phase-based workflow. Where necessary, additional components were introduced by the author to support the execution and evaluation such as the task checklist and the evidence log. These added aspects do not interfere with the principle of black-box IoT security testing and provide support for traceable and reproducible results.

4.1 Scoping and Legal / Ethical Setup

A legal and ethical process is used to ensure that all executed activities are authorized and compliant with responsible research practices. Before any practical activities were carried out, a Letter of Authorization (LoA) was defined and signed by all parties involved. This document sets the boundaries of allowed testing activities, the management of sensitive information, and the responsibilities of the researcher. Full physical access is explicitly permitted, including possible access to all interfaces found on the device hardware, while ensuring that all actions are executed in a non-destructive matter. Therefore, all described activities of this research are conducted within the authorized scope.

In accordance with the interests of the manufacturer and to comply with the defined confidentiality requirements, all identifying information is redacted throughout this research. The name, model, other identifiers, and network infrastructure endpoints of the device are anonymized by replacing them with generic placeholders. While screenshots, packet captures, or log snippets containing identifying information are redacted in this thesis, by agreement, the tester provides a separate and unredacted report with full details

of the results. Possible future works or publications of the results by the tester will only be conducted with explicit approval from the manufacturer.

The scoping phase defines which components of the infrastructure are included in the security assessment. The resulting scope includes the hardware of the device, its firmware, all communication interfaces, and the respective mobile application, if they are legally and technically accessible. Cloud infrastructure and other back-end services are also included in the scope if directly connected or used by the device or the mobile application. Systems or endpoints discovered but not accessible through legitimate customer access (e.g. through brute force attacks) are excluded.

In conclusion, the methodological design is based on ethical principles of cybersecurity, following practices of responsible disclosure, minimisation of damage, and avoidance of undefined and unnecessary intrusion. A structured and documented scoping and legal setup is deemed crucial for ensuring ethically sound and replicable research and allows future professionals to follow the same principles.

4.2 Preparation and Instrumentation

In this phase the technical and methodological requirements are established. The definition of the test perspective, the formalisation of the device and attacker models, the construction of the testing fragments, and the preparation of the testing environment are documented.

4.2.1 Test Perspective

The definition of the test perspective establishes the conditions under which all methodological choices are taken. It specifies which categories of the ISTG test cases can be taken into consideration, influences which assumptions about the behaviour of the system are suitable, and which attack scenarios are feasible.

The practical assessment follows a black box testing perspective with physical access to the device, which means that no internal information, such as schematics or source code, is known or provided. The analysis is based solely on external observations of behaviour and all direct interactions possible for a general consumer or an attacker with possession of the product. The chosen simulated situation depicts realistic adversarial conditions of

consumer IoT products and provides a practical usage scenario of the ISTG without assuming privileged or special access to the system or to information.

4.2.2 Device and Attacker Model

The methodology follows the ISTG in the creation of device and attacker models [11], which in combination determine the general scope and applicability of test cases. This is the first direct application of the ISTG framework in this methodology. The two models are described in further detail in the following paragraphs.

Device Model

The ISTG offers a technology-focused structure for the modelling of IoT devices. It differentiates between device-internal elements, found inside of the physical enclosure, and interfaces, enabling interaction and communication internally or with external components. Typically, internal elements include a processing unit, memory, firmware, and services for data exchange, which are all treated as separate testable categories. Interfaces can be divided into internal interfaces such as UART or SWD, or external interfaces such as physical or wireless connections (e.g. USB cables, Bluetooth, or Wi-Fi) and user interfaces (e.g. displays, mobile applications).

Through mapping the device infrastructure, the system is broken down into smaller components, enabling the tester to select relevant ISTG test cases according to the discovered elements and interfaces. By creating a graphical representation of this model, a comprehensive and clear basis is created, further enhancing reproducibility and comparability with other devices. An example graphical model can be found below in Figure 5.

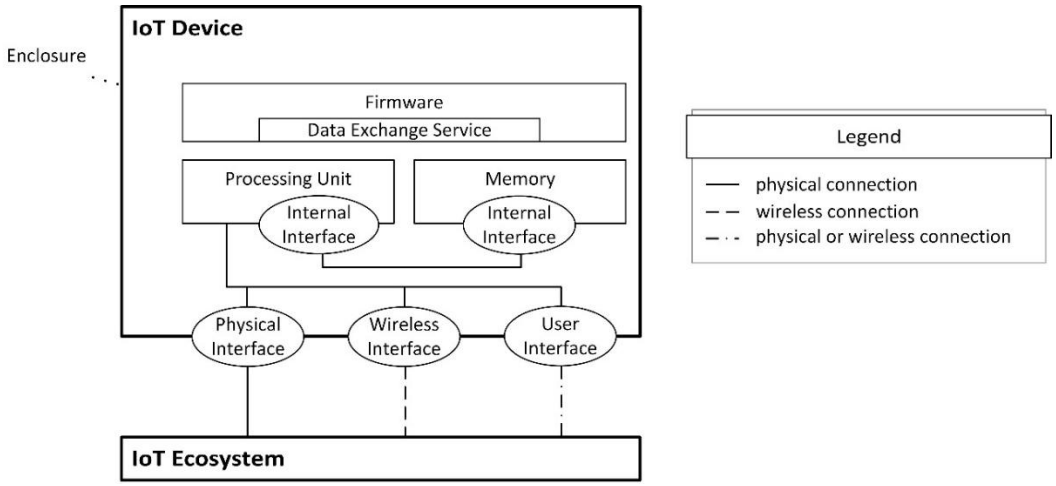


Figure 5. Example Graphical Device Model. [39]

Attacker Model

In addition to the device model, an attacker model is created as defined by the ISTG [11]. It categorizes the conditions of malicious actors by two metrics: physical access level (PA) and authorization access level (AA).

Physical access is separated into four levels, ranging from PA-1 (remote interaction) to PA-4, which describes full invasive access to the hardware. Similarly, authorization access has four levels as well. Unauthorized interaction is described as AA-1 while privileges on the level of the manufacturer are AA-4.

Each provided test case in the ISTG has requirements specifying the minimum PA and AA levels necessary to conduct them. Therefore, these metrics are defining which of the mapped components can be tested under the test perspective chosen for the security assessment. Due to possible exploitation of discovered vulnerabilities, the authorization level may change during the penetration test, allowing additional test cases to be included.

4.2.3 Task Checklist Construction

The methodology used in this thesis implements the list of ISTG test cases into a task checklist. Each item of this checklist is based on a test case, grouped by ISTG categories, and expanded into the following structure (see Figure 4):

- ISTG test case identifier
- Name of the test case
- Applicable component(s)

- Access level requirements
- Tooling requirements
- Summary of the test case
- Execution phase
- Objectives
- Execution steps
- Security-relevant findings
- Expected secure behaviour
- Remediation suggestions
- Evidence IDs
- Status (success / partial / blocked / not applicable)

This list is used to enable consistent execution and documentation of every test case, resulting in systematic analyses and references. Additionally, reproducibility is supported by allowing other security professionals to replicate the testing process under equivalent conditions.

4.2.4 Coverage Matrix and Evidence Log

To further support a structured analysis and evaluation, two complementary documentation elements are formed:

- Coverage Matrix – This figure links all tasks from the checklist to the respective ISTG category, recording the applicability and status such as completed, partially completed, blocked, or not applicable. This results in the basis for the metric measuring ISTGs applicability for the selected device. A separate coverage metric is evaluated as defined in chapter 3.
- Evidence Log – This log includes records of all evidence collected during the security assessment, such as serial logs, screenshots, packet captures, and other outputs. Every entry is linked to the Task Checklist by a unique Evidence ID and is used for the final analysis. Sensitive, personal, or identifiable information is redacted in all publicly available versions of this document but got provided to the manufacturer.

In combination, these two elements are enabling the creation of a comprehensive audit trail of the whole penetration testing process and allow the qualitative and quantitative evaluation of the ISTG framework.

4.2.5 Environment Setup

Before the start of the practical assessment, a controlled and documented testing environment is set up to ensure consistency during multiple runs and to allow for exact reproduction of the testing conditions. The environment is consists of the workstation that is used for analysis and tool execution, the required hardware interfaces for the interaction with the device, and the documentation of the process of preparing the device itself. This setup aims to reduce uncontrolled variables such as automatic updates or different behaviour on multiple assessments. The three primary aspects of this setup are described in the following paragraphs.

Workstation Environment

To allow for consistent results and processes, a dedicated machine is configured for the security assessment. This system provides the required tools with fixed versions (if possible) and ensures that the configuration is as identical as possible across all phases of this research. The goal of this step is the creation of an analytical basis that can be reproduced independently.

Hardware Interaction Environment

As some IoT devices commonly use external or internal interfaces that require special equipment such as adapters or connectors, these are prepared and documented during this step. Through documentation of connections and configurations, the author further creates a reproducible setting, allowing for identical conditions between executions and for future academic works.

Operational Device Environment

The device is connected to the real production ecosystem, as there are no available separated environments for the communication with the cloud infrastructure. During the security assessment, the device is analysed in unconfigured, out of the box condition, and is then set up through the intended process of a real customer. As the cloud infrastructure

cannot be isolated and the communication via mobile networks cannot be simulated or interfered with by federal regulation, the chosen methodology aims to create a consistent environment for local testing, without controlling the backend conditions. All interactions with the aforementioned uncontrollable aspects remain within the authorized limitations of the LoA and national legal regulation.

4.3 Pilot Run

Before the main run of the security assessment is executed, a pilot run is used to validate the technical and methodological setup. The pilot run does not aim to discover vulnerabilities, but rather to verify the consistency, feasibility, and clarity of the proposed workflow and its documentation elements such as the Task Checklist, Coverage Matrix, Evidence Log, and environmental setups. This phase provides additional assurance to create a coherent, structured and reproducible application of the main assessment.

A small selection of test cases is selected for the pilot run, representing each ISTG category and executed on a primarily validation-focused level. The selected test cases represent the main ways of interactions with the device that will be used during the main run such as physical access or tasks based on the communication level. The results of the pilot support the tester in identifying possible ambiguities in aspects such as the checklist definitions, assumed requirements and access levels, and necessary tooling. This limits the necessary adjustments during the main phase, resulting in a more efficient and transparent security assessment.

4.4 Main Execution (Run 1)

During this phase, the ISTG is applied comprehensively to the device through a PTES-inspired workflow. It is the primary practical assessment, during which all previously selected and applicable ISTG test cases are executed in a structured manner. The results of Run 1 are used as a complete set of observations, later used to analyse the coverage, reproducibility, and suitability. No interpretation of findings or conclusions will be taken at this stage. The main execution is structured in ten high-level steps developed by the author, which are described below. An overview of the individual steps is shown in Figure 6.

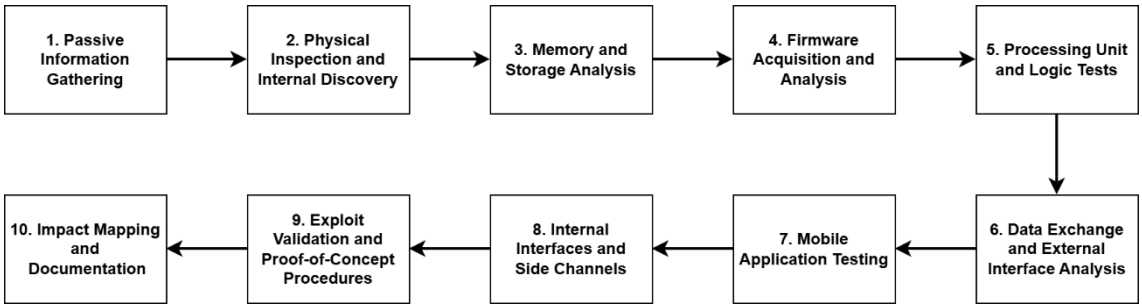


Figure 6. Overview of Main Execution Steps.

As the ISTG does not describe a specific workflow sequence, the defined categories were reorganised into a coherent process order reflecting the authors logical process and common penetration-testing practices such as the PTES. The Task Checklist provides the basis for the execution of each task and results are documented in the Coverage Matrix and Evidence Log.

4.4.1.1 Passive Information Gathering

At the beginning of the main execution initial reconnaissance is conducted, primarily through discovery and analysis of public information. While during this phase the device is not modified or its behaviour influenced, initial network and wireless analysis is performed and used in later phases. The primary goal is the creation of an understanding of the device and its observable components, which is used as a baseline for the subsequent phases.

4.4.1.2 Physical Inspection and Internal Discovery

A thorough inspection of the device enclosure and internal components, such as chips and connectors, is conducted to determine which interfaces are available and accessible. This phase defines, which internal interfaces of the device model can be tested within the authorised scope.

4.4.1.3 Memory and Storage Analysis

Identified memory or storage components are analysed to the extent allowed by the LOA during this phase. By examining and identifying memory chips, the plausibility of access is determined through cross-referencing publicly available datasheets. This preparation provides the necessary information for the following firmware analysis.

4.4.1.4 Firmware Acquisition and Analysis

If found feasible in the previous step, authorised methods are used to acquire the firmware of the device, which is then prepared for both, static and dynamic analysis. This phase is crucial for evaluating aspects such as the integrity of the firmware, default configurations, update mechanisms, and the presence of sensitive data such as credentials or API keys.

4.4.1.5 Processing Unit and Logic Tests

During this phase all test cases focusing on the processing unit are executed. These include tests concerning processing-unit behaviour, implementation of business logic, and the handling internal instruction. If applicable these tasks are aiming to analyse if the processor handles data as intended, especially focusing on malformed or unexpected input.

4.4.1.6 Data Exchange and External Interface Analysis

All available communication channels between the device and external systems are assessed in this step. Depending on the device, these channels can include wireless, physical, and network interfaces, The test cases focus on how data is transmitted, processed, or received, and evaluates if appropriate security mechanisms are implemented at corresponding components.

4.4.1.7 Mobile Application Testing

The accompanying mobile application is assessed statically and dynamically by using an emulated mobile device, intercepting network traffic or analysing the publicly accessible installation files. As no separate isolated backend environment is available, the assessment is limited to cloud endpoints that are discovered during legitimate user behaviour. Primary aspects of this phase are the interaction between the device and the phone, potential exposure of sensitive information, and security mechanisms for communication to the cloud environment.

4.4.1.8 Internal Interfaces and Side Channels

The tests executed during this phase evaluate all internal interfaces defined in the LoA, such as UART, SWD or JTAG. Additionally, potential side-channel attacks[11] are assessed, which could be exploited to break cryptographic mechanisms. Altogether, the

[11] https://csrc.nist.gov/glossary/term/side_channel_attack

accessibility and security of low-level components are evaluated and indicators for side-channel attacks are explored.

4.4.1.9 Exploit Validation and Proof-of-Concept Procedures

For all previously analysed test cases, which showed signs of potential exploitation, proof-of-concept procedures are created and executed. These are used to validate the validity and reproducibility of the potential exploit. Due to the missing separation between the testing and production system, all proof-of-concept executions are conducted under strong precautions of the tester, and ethical and legal guidelines, as to not disrupt any live services.

4.4.1.10 Impact Mapping and Documentation

This administrative step includes the documentation of all executed tasks in the Task Checklist and Coverage Matrix, adding the status of the task and potential Evidence IDs. Additionally, impact categories (e.g. confidentiality, integrity, availability) are assigned preliminarily enabling later analysis. No assessments of the severity are taken or conclusions drawn during this phase. The aim of this organisational process is to provide a complete, transparent and prepared basis for Run 2 and the final evaluation.

4.5 Reset and Replication (Run 2)

After finalising and documenting Run 1, the testing environment is restored to its original conditions. The device itself is reset as far as made possible by the manufacturer. These steps are taken to create comparable conditions to the start of the first run. Depending on the possibilities of the device reset mechanisms, the initial setup process is repeated. The aim of this phase is to create an environment with minimised differences to the initial conditions that could negatively impact the validity of the results discovered in the second execution.

To evaluate the reproducibility of the results, Run 2 includes a subset of test cases which are executed. For this subset tasks get selected by the impact level or security considerations of the corresponding findings. In addition, test cases that are expected to produce inconsistent results are chosen as well. The decision to only repeat part of the task checklist is made due to technical limitations and time constraints. Each selected task

is executed following the procedure described in Run 1, with new Evidence IDs being used for observations, enabling future comparison.

The execution of the replication run allows a systematic evaluation of the consistency of discovered behaviours under comparable conditions. This supports the reproducibility metric of this thesis and assesses the ISTG's methodological robustness.

5 Practical Execution

In this chapter, the methodology defined in Chapter 3 is practically executed and documented by applying the workflow of Chapter 4. The process follows the same structure as the methodology and presents the definition of the scope, the device and attacker model, the applied task structure, and the actual environment setup. To enable other security professionals to apply the same workflow to other devices, this chapter aims to provide a transparent, reproducible, and comprehensive documentation of the security assessment execution.

5.1 Practical Scoping & Legal Setup

As defined in section 4.1, the LoA permits the tester to assess the physical device, internal components, firmware, wireless and physical interfaces, the mobile application, and cloud communication. Internal interfaces are tested under the condition that no intentional damage is done to the components. Access to the cloud infrastructure and APIs is limited to domains owned by the manufacturer and all endpoints found communicating with the BHM, though only non-destructive testing was permitted, as no separate testing environment was available. Testing of the mobile application and cloud services was limited to test-case relevant contexts and were not performed as standalone assessments. The tester did not attempt to interfere with any services impacting other customer's experience or information. To provide a reference for future works by other researchers, a redacted version of the LoA is found in Appendix 2 and may be used as a template for comparable conditions.

5.2 Practical Preparation and Instrumentation

All steps taken during the practical preparation of the security assessment are documented in this section. It describes the process and results of how the device model, attacker model, task checklist, coverage matrix, and testing environment were created and applied.

5.2.1 Device Model

As described in the methodology (see 4.2.2), a device model was developed for both, the sensor unit and the base station of the BHM. As a result, the individual models represent the internal structure and interfaces discovered by analysing the hardware of the device. The device model is not an accurate or comprehensive representation of the BHM but rather an abstraction, enabling understanding of the system and selection of applicable test cases. The diagrams shown in Figure 7 and Figure 8 are describing the internal components, wireless and physical interfaces, and inaccessible internal machine-to-machine communication interfaces. To gather the necessary information the BHM components were inspected and carefully torn down minimising the risk of damage. With access to the bare circuit boards, it was possible to identify the components such as the main control unit (MCU) and or exposed debugging interfaces individually.

Sensor Unit Device Model

The manufacturer decided to use a separate small and battery-powered sensor unit to collect the necessary health and motion data. This data then gets transmitted to the base station, that is described below. The ISTG device model defines a processing unit component, which was identified as an ISP1507-AX[12] module, which is based on the nRF52 system-on-a-chip (SoC) family manufactured by Nordic Semiconductor[13]. No additional memory chip was identified during the inspection, which got confirmed by referencing the respective datasheet by Insight SiP [40]. As a result, the firmware, configuration, and temporary data are all stored in internal memory of the MCU, making direct access without further deconstruction impossible.

The custom firmware running on the ISP1507 implements all necessary functionalities such as reading sensor data and communication with the base station through BLE. After analysis of available debugging interfaces, potential functionality for over-the-air (OTA) firmware updates was identified but could not be directly accessed and will be examined further at the firmware analysis phase.

[12] https://www.insightsip.com/products/bluetooth-le-modules/isp1507
[13] https://www.nordicsemi.com/Products/nRF52840

As mentioned before, the sensor unit implements a BLE interface as the data exchange service with the base station. Internally, the MCU communicates with a digital accelerometer via the Inter-Integrated Circuit protocol (I2C)[14].

The ISTG defines sensors and actors as physical interfaces rather than components, as they are handling interactions of external factors, such as movement or temperature, and the MCU. In the case of the sensor unit, the tester therefore defined three physical interfaces: an infrared temperature sensor, an accelerometer, and an inductive charging coil. The ISP1507 contains a BLE transceiver, which is identified as the only wireless interface of the sensor unit. Additionally, three internal interfaces were discovered during the teardown of the part. A Serial Wire Debug interface through a layout of 3x2 headers for pogo pins was found, as well as a set of five blank copper pads, most likely used for sensor testing purposes by the manufacturer. Through analysing the rest of the installed components, the author deduced that the accelerometer only supported communication via I²C, which is the third internal interface. In contrast to the base station, the sensor unit does not include any user interface such as LEDs or buttons. The finished device model is shown below in Figure 7.

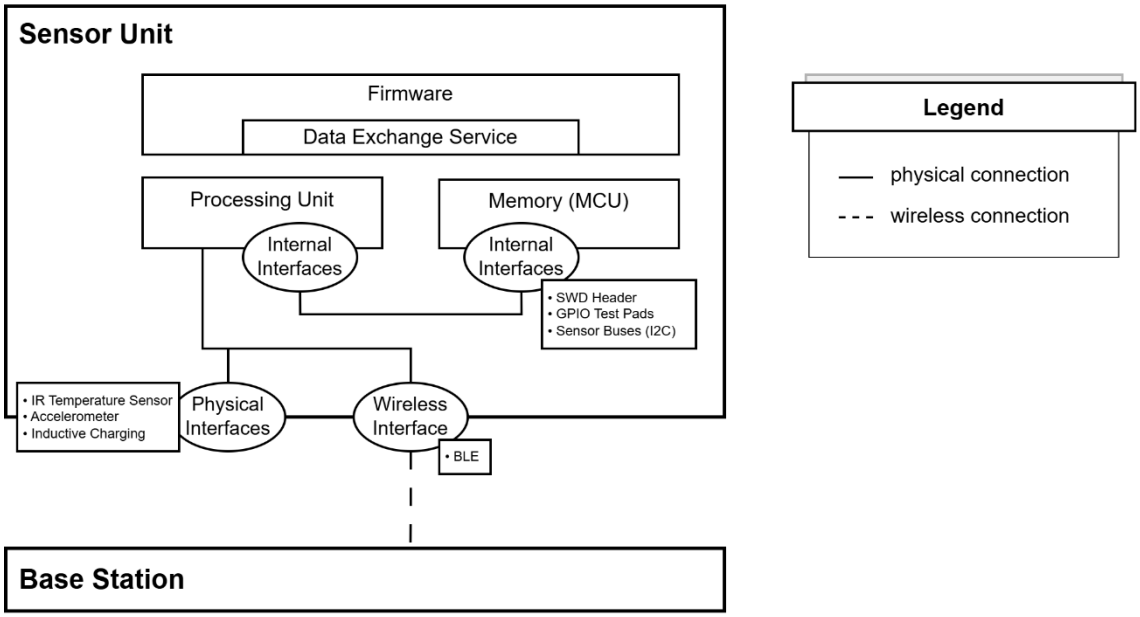


Figure 7. Device Model of the Sensor Unit.

[14] https://onlinedocs.microchip.com/oxy/GUID-61202E6F-BC03-4D81-AAB9-269E87F9B49C-en-US-22/GUID-90719A31-E8B5-4556-8025-477678405B3A.html

Base Station

The main purpose of the base station is to relay the sensor data to the cloud infrastructure, split between two processing units. A nRF52840 microprocessor, also from Nordic Semiconductor, handles the operational logic and BLE communication with the sensor unit. By using internal UART channels, it is connected to a Quectel EG915U-EU[15] modem for cellular connectivity. The two MCUs both rely only on individual integrated memory; thus, no external memory modules were discovered.

Additionally, to omit the necessity of physical SIM cards, the manufacturer includes a Samsung S3FW9FJ secure element[16]. This SoC manages authentication mechanisms for LTE, based on the embedded Universal Integrated Circuit Card (eUICC) standard[17], allowing remote management and configuration. As no public datasheet of this component is available, and physical interference may lead to permanent damage or data deletion, the S3FW9FJ was defined as out of scope for this assessment.

Both MCUs are flashed with individual firmware packages but share a common firmware update mechanism. The steady internet connection provided by the LTE module allows the manufacturer to update the product as a whole over-the-air and without user intervention. The data exchange services comprise of the LTE connectivity enabled by the EG915U, the BLE communication provided by the nRF52840, and the internal UART channel between the two MCUs.

Exposed interfaces on the base station are limited. Similar to the sensor unit, its internal interfaces include SWD pads connected to the nRF52840, UART headers for debugging information of the modem, and the internal UART communication. The manufacturer implemented a USB-C port to power the base station as the only physical interface. Attempts to use this USB connection for communication failed, as the USB data pins are not connected to the SoC. The tester defined two wireless interfaces: BLE between the two physical devices and LTE for interaction with the cloud service. In contrast to the sensor unit, the base station contains two LEDs, which represent power and LTE connectivity states. The final device model of the base station is found in Figure 8 below.

---

[15] https://www.quectel.com/product/lte-cat-1-bis-eg915u-series/
[16] https://semiconductor.samsung.com/security-solution/ese-esim/part-number/s3fv9rr/
[17] https://csrc.nist.gov/glossary/term/embedded_universal_integrated_circuit_card

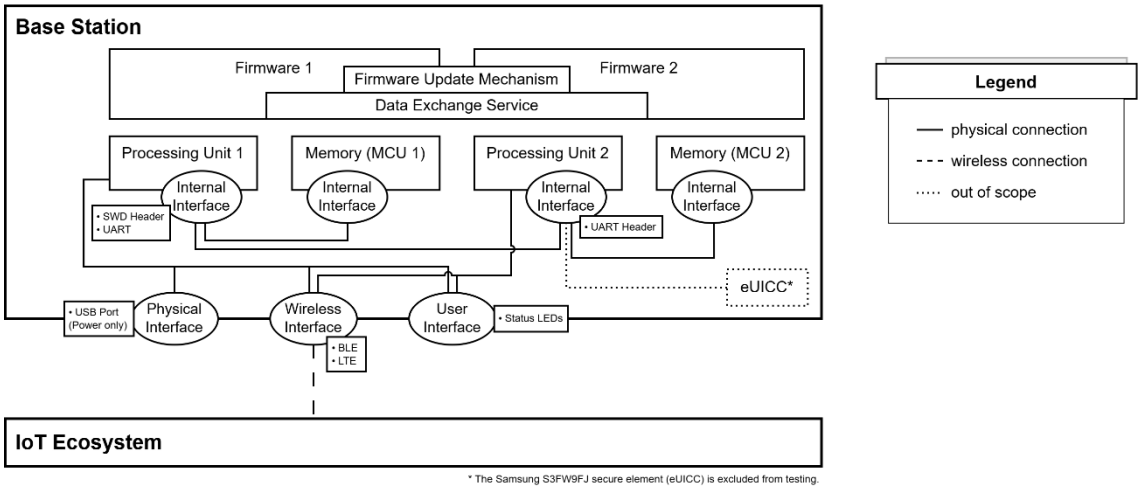


Figure 8. Device Model of the Base Station.

5.2.2 Attacker Model

The aim of the attacker model is to describe the access conditions of the tester during the practical security assessment. This model follows the structure provided by the ISTG and is founded by the assumption that the tester represents a realistic adversary in possession of the device and without any privileged access or information. [41]

Physical Access Level (PA)

For this security assessment, the tester was able to gain invasive physical access for both devices of the BHM. The electronics of the base station were enclosed in a plastic frame, accessible by removing three screws. The sensor unit was encased in silicone by the manufacturer but could simply be pulled off carefully after an initial cut was made. As a result, the internal interfaces were analysed, and components probed directly. In the context of the ISTG, this corresponds to the highest level of physical access possible:

> *Invasive access (PA-4): There is no physical distance between an individual and the device and the individual can directly access device-internal elements in a physical manner (i.e., open the device enclosure). [41]*

With most test cases already requiring PA-4 and all lower access levels being included in PA-4, the security assessment will not be limited in this regard.

Authorization Access Level (AA)

There are no authenticated interfaces available to the tester at device level. Both of the devices do not expose any authenticated local services, and the BLE communication between the devices can be sniffed but not interacted with by the tester. Even though debugging interfaces such as UART and SWD were discovered, all attempts to access them directly failed, as they were read-only logs and locked by the MCU respectively. As a result, for the physical analysis of the security assessment, the authorization level corresponds to the lowest level defined in the ISTG:

> *Unauthorized access (AA-1): An individual can get anonymous access to the device component. Attackers with anonymous access can be any unregistered user. [41]*

During normal customer setup of the mobile application, the tester becomes a legitimate authenticated user of the cloud service and the application. As the functionalities of the mobile application are fully separated from the devices, only communicating with the cloud infrastructure, the elevated AA-2 level only applies to cloud-endpoints. With most ISTG test cases interacting with the devices and not the cloud, the attacker model of this thesis is defined with AA-1, but if applicable AA-2 is considered.

Resulting Attacker Model

With both access levels defined, the model representing the attackers' perspective during this security assessment is described by the author as:

A realistic malicious actor having full invasive physical access to the BHM (PA-4). Access to the devices' internal components is unauthorized (AA-1), and realistic customer access (AA-2) to cloud functionalities is set up.

This perspective enables all non-destructive test cases of the hardware, firmware, and interfaces, while still being bound to the LoA. The access levels and their corresponding aspects are summarized in Table 2 below.

Table 2. Defined Access Levels for Practical Assessment.

Aspect	**Access Level**	**Description**
Physical Access	PA-4	Internal interfaces (UART, SWD, test pads) probed and components analysed.
Device Authorization	AA-1	No authenticated local services; tester treated as anonymous actor.

Cloud / App Authorization	AA-2	Standard customer setup; no interaction with the devices.

5.2.3 Task Checklist

All test cases defined in the ISTG are included in the practical task checklist. The component field defines for which component (sensor unit, base station, mobile application, or cloud service) the test was executed. The execution phase field was assigned before the security assessment was conducted, which led to some changes during the execution, noted in the same field. The Evidence IDs are defined in the form of 'BHM-2025-E' followed by a sequential number, linking task entries to their corresponding evidence. A practical excerpt of the checklist is shown in Table 3, while all applicable Task Checklist items are found in Appendix 3. Non-applicable test cases are described and published via GitHub[18].

Table 3. Example Task Checklist Item.

ISTG Test Case Identifier	**Name**
ISTG-WRLS-INFO-001	Disclosure of Implementation Details
Access Requirements	**Component**
PA-2 / AA-1	Sensor Unit — BLE Interface
Execution Phase	**Tooling**
Phase 6	nRF Sniffer, Wireshark, BLE-capable adapter
Status	**Evidence ID**
Completed	BHM-2025-E01
Summary of Test Case	
This test case evaluates whether the wireless interface (BLE) unintentionally discloses implementation details during advertising, connection establishment or normal operation. Such details may include device name conventions, firmware identifiers, service UUID structure or diagnostic metadata.	
Objectives	
▪ Identify information leaked through BLE advertisements, scan responses or connection parameters. ▪ Assess whether the BLE GATT structure exposes internal implementation details. ▪ Determine whether metadata allows fingerprinting of firmware version or hardware characteristics.	
Execution Steps	

[18] https://github.com/tfankhauser/ISTG-evaluation

1. Perform passive BLE scanning using nRF Sniffer and Wireshark. 2. Record advertisement packets, scan responses and initial connection exchanges. 3. Analyse GATT service enumeration using a BLE inspection tool. 4. Document any implementation-specific metadata (e.g., model identifiers, version strings, vendor-specific UUIDs). 5. Cross-reference leaked details against publicly known device identifiers to assess fingerprinting risk.
Security-relevant Findings
The advertisements on the BLE channel revealed minimal metadata such as the MAC addresses and non-descriptive UUIDs. No specific versions of services or firmware information were discovered. Analysis of the GATT packages showed custom data which could not be interpreted.
Expected Secure Behaviour
BLE advertisements should disclose only minimal necessary information (MAC address, generic flags, and optional anonymised service identifiers). No firmware version, hardware identifiers, internal service naming or diagnostic information should be observable.
Remediation Suggestions
▪ Minimise metadata included in BLE advertisement frames. ▪ Remove or obfuscate diagnostic GATT services from release firmware builds. ▪ Implement BLE privacy extensions to reduce device fingerprinting risk.

5.2.4 Environment Setup

The workstation used during the assessment was running Kali Linux 2025.3, a penetration testing specific Linux distribution. It was set up as a virtual machine as to allow the creation of snapshots to save and restore the state before the first execution. The workstation was used to extract and analyse the firmware, capturing and inspecting the network traffic, monitoring wireless data, and the documentation of all findings.

Interactions on hardware level were performed using multiple cheap adapters. The UART interface was accessed with a USB adapter[19] based on the CH340G chip. As the pin headers the board were in a non-standard layout, small tweezer clips[20] were used to connect to the adapter. For the SWD debugging interface, a ST-LINK/V2 adapter[21] was used in combination with STM32 Cube Programmer software. Through 3D-printing and manual inspection of the pins, the author created a custom connector for the SWD header

[19] https://amzn.eu/d/2pi4iR7
[20] https://amzn.eu/d/47g2DL4
[21] https://amzn.eu/d/2Z5JSQB

layout of the PCBs. Additionally, a logic analyser[22] was used to inspect and identify protocols. Lastly, Bluetooth Low Energy traffic was captured and analysed with the help of an nRF52840 Dongle USB adapter[23]. An overview of the used hardware tools can be seen in Figure 9 below.

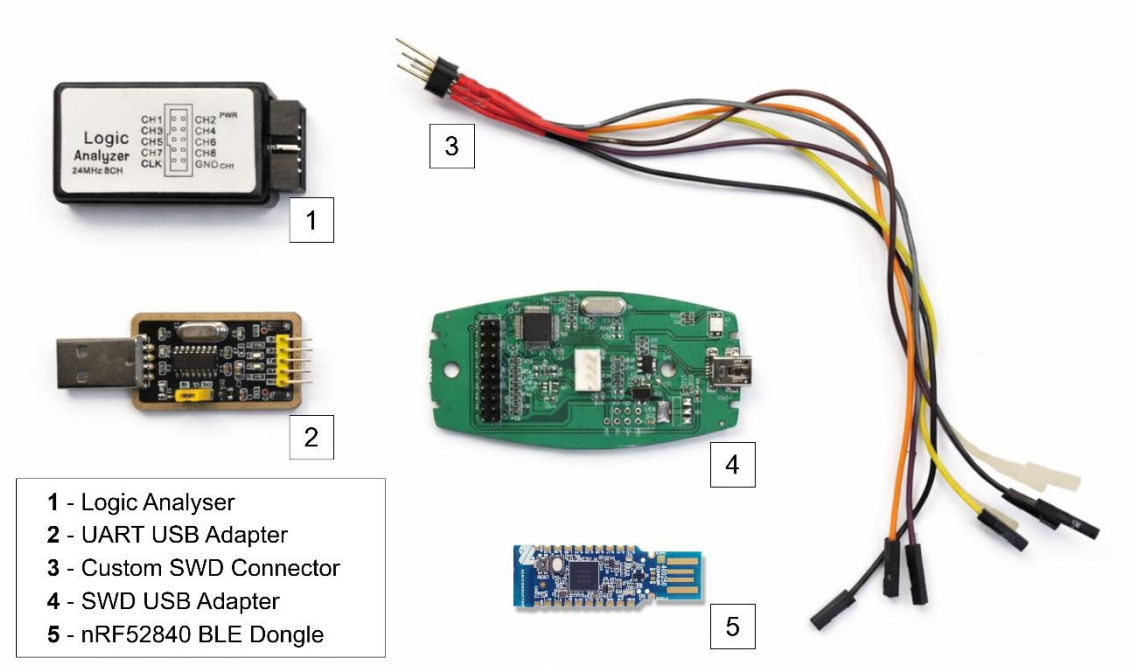


Figure 9. Overview of Used Hardware Tools.

A collection of established and mostly open-source software was used to execute the ISTG test cases. While the primary tools include Wireshark, Burp Suite and Visual Studio Code, a full list of utilized software tools and their corresponding version can be found in Appendix 5.

5.3 Practical Pilot Run

As described in the Technical Methodology (see 4.3), the pilot run aims to validate the workflow of the security assessment, the structure of the task checklist, and the

[22] https://amzn.eu/d/1nn2iKq
[23] https://amzn.eu/d/dw9WvoX

environment setup before conducting the main runs. By executing selected test cases, it ensures that the process, documentation, and testing artefacts are functionally sound.

5.3.1 Scope of the Pilot Run

A small sample of test cases was selected from early phases of the ISTG execution. They include test cases from different categories and are listed in Table 4.

Table 4. Selected Test Cases for Pilot Run.

Summary	**Corresponding ISTG Identifier**
Physical hardware inspection	ISTG-PHY-INFO-001/002
Wireless Discovery (BLE)	ISTG-WRLS-INFO-001/002, ISTG-WRLS-AUTHZ-001
Firmware fingerprinting and static string analysis	ISTG-FW-INFO-002/003
Initial overview of cloud endpoints based on analysis of firmware	ISTG-DES-INFO-001/002

These tasks did not require full interaction with the BHM's external infrastructure and were conducted using only the physical hardware and discovered firmware binaries. The pilot run was not executed to result in security findings. Therefore, it did not include physical probing of debugging interfaces, attempted active exploitation, or other activities requiring full environment access and interaction.

5.3.2 Validation of the Methodology

After conducting the pilot run, the author confirmed that the template and workflow used for the task checklist works effectively and did not need to be adjusted. The items contain a clear separation of the execution and observations, and the evidence linking works as expected.

The general process for evidence handling, file naming, and the references between evidence in the appendix and chapter content were validated. Standardised file names were adjusted minimally to ensure consistent traces between task checklist items and documented evidence.

The results of the pilot run confirmed that the phase structure defined in the methodology can be mapped onto the BHM's architecture. Regardless, only early phases were

validated, and later phases were verified during the main run. If adjustments were needed in these stages, they are explicitly mentioned in the corresponding documentation.

5.3.3 Observations and Implications

As mentioned before, the goal of the pilot run was not to discover security vulnerabilities but rather to validate the methodology defined in chapter 3 and 4. The results of the applied test cases showed that the defined workflow based on the ISTG, and the chosen tooling were suitable for assessing the BHM, even though the author discovered multiple practical considerations that influenced the main execution. These implications are described in more detail in the following paragraphs.

The analysis of the physical hardware confirmed that the inspection of external and internal hardware of the BHM is possible without destructive disassembly. A teardown is necessary but does not impact the devices functionality and can be fully reversed by the tester or the manufacturer. As a result, the pilot run confirmed that the planned order of execution phases, with an early hardware inspection phase, can be executed without modification and impact on the other testing phases.

Early wireless discovery of the BLE communication showed that the device uses some mechanisms to only allow a connection with the bound base station. As a result, some wireless test cases cannot be conducted without adjustments. For the main runs this led to the preparation of alternative methods for testing of the wireless interface, especially considering limitations based on the binding mechanism.

Basic static analysis of the firmware resulted in the extraction of component identifiers and communication endpoints, which confirmed the initial understanding of the infrastructure and device architecture. It verified the selected tooling and analysis sequence, not requiring any modifications.

Finally, the pilot run highlighted inconsistencies in file naming and timestamping of evidence. While some evidence files automatically included timestamps, others were lacking references of the time of execution, which would negatively impact transparency of the sequence of actions. As a solution, the author introduced a fixed naming scheme for any collected evidence, including the timestamp of the execution.

In conclusion, the pilot run was a valuable tool to validate the applicability of the ISTG-based workflow before the main execution. It resulted in the confirmation of the overall structure, tooling, and processes for collecting and storing evidence and led to minimal adjustments. The evidence collected during the pilot run was not stored persistently, as it was not security-relevant and the test cases are executed again in the main execution. The state of the device was not changed, and the workstation was reset to initial conditions. The following chapter describes the systematic execution of the security assessment with the completed adjustments and the validated methodology.

5.4 Main Execution (Run 1)

The first main execution contained all test cases that were deemed relevant and feasible for the assessment of the BHM. Each test case execution was based on the task checklist item, providing structure for the definition of objectives, requirements, and execution steps. The results were documented with corresponding evidence, linked in the checklist item and added either in the appendix or the public GitHub repository, depending on file size and format. With the workflow structure defined in chapter 4.4, the assessment began with general passive reconnaissance and hardware analysis, and skipped the memory and storage inspection, as no external memory was discovered. The firmware, processing unit, external interfaces and mobile application got assessed, and ending the assessment by analysing the internal interfaces. Discovered possible vulnerabilities were exploited and proof-of-concepts documented before finishing the first run by finalizing the documentation.

As per the LoA, the scope of this security assessment was limited to the BHM itself and its corresponding interfaces. Active security testing of the cloud infrastructure was excluded, with the exception of already discovered endpoints. The constraints identified by the pilot run, such as BLE binding behaviour, locked access to debugging interfaces, and no usable trigger to start a firmware update, were implemented into the execution plan.

All executed task checklist items and their respective results are described in further detail in the corresponding sections. Non-applicable test cases are included in the complete task checklist available on GitHub[24] and are not discussed in this chapter.

5.4.1 Passive Information Gathering

As a starting point, this phase aimed to gain an understanding of the targeted system and included only passive reconnaissance. This involved no specific ISTG test cases and instead focused on the device architecture, observable components, and passive communication discovery.

The collection of information was limited to externally visible data, including BLE advertisements from the gateway and the sensor, publicly available product information, and mobile application metadata. With the help of a BLE sniffer (nRF52840 Dongle), the collected BLE broadcasts showed the presence of the two BHM devices (sensor and base station), their MAC (medium access control address) addresses and basic service identifiers, which confirmed Bluetooth Low Energy as the communication channel between the devices. Through analysing the base station's behaviour, the tester was able to assume that it connects to the mobile network without any necessary user configuration.

In addition, the manufacturer's website, user documentation, and support articles were analysed for further contextual details. The information gathered provided insights into the user setup, capabilities for users, and general intended functionality of the device. In combination with the device model, this allowed the tester to gather an understanding of the functionality and architecture of the BHM and its infrastructure. As this phase was designed to be used solely as preparation for following phases, all observations remained passive and non-intrusive.

5.4.2 Physical Inspection and Internal Discovery

The second phase involved a full analysis of the BHM, first in its enclosure and then torn down to the bare PCBs. During this stage, the goal was to identify available interfaces, connectors, or hardware components to evaluate in later phases, bound by the restrictions

[24] https://github.com/tfankhauser/ISTG-evaluation

set forth in the LoA. No destructive activities were conducted and the BHM was deconstructed only to the permissible extent. Therefore, no components were removed or modified in any way.

As a result, the internal components such as the processing unit (MCU), wireless modules, and accessible connectors were identified and documented. The tester also searched for possible UART, SWD, or other exposed internal interfaces, which resulted in finding one UART connector (base station's LTE module) and a SWD header on each of the two devices connected to the MCU. Through referencing the components' identifiers with the corresponding public datasheets, the exact models for relevant chips were matched and possibly interesting pins and traces were examined under the microscope. The USB-C port of the base station was assessed to determine if it can be used as a data interface, which was not successful.

The task checklist contains multiple test cases, that focus on identifying and categorizing internal and physical interfaces. An overview of applied test cases can be found in Table 5 below and its corresponding evidence is linked through the task checklist. The results of this phase showed that the only accessible interfaces were either non-interactive (UART debug-log of LTE module) or securely locked (SWD access on both devices).

Table 5. Executed Test Cases of Phase 2.

Test Case Identifier	**Topic**
ISTG-PHY-AUTHZ-001	Unauthorized Interaction with Physical Ports
ISTG-PHY-INFO-003	Disclosure of User Data
ISTG-PHY-CONF-001	Usage of Outdated Software (physical)
ISTG-PHY-CONF-002	Presence of Unnecessary Functionalities
ISTG-INT-AUTHZ-001	Unauthorized Access to the Interface
ISTG-INT-INFO-001	Disclosure of Implementation Details

5.4.3 Memory and Storage Analysis

During this phase, the tester focused on assessing the memory architecture of both devices trying to determine if accessible non-volatile storage could be analysed. If external memory chips are used, an attacker could possibly use special adapters to read data stored in the memory during runtime, exposing binaries or configurations. In the case of the BHM, both the sensor unit and the base station are based on the nRF52840 SoC, which

by default uses integrated flash storage, resulting in no external modules being available for direct probing.

This was also confirmed by visual inspection of the boards, as no external chips or other storage modules such as SD-cards were found. The internal flash of the SoC contains all persistent data such as firmware and configurations and could not be analysed within the permissions of the LoA. As an alternative, the tester built an adapter for the exposed SWD interfaces to try and access the MCU directly, but the interfaces were found to be locked by default on nRF52840 deployments.

After exhausting and evaluating all possible options, no source code, internal configuration, or binaries could be gathered during this phase. These observations resulted in all memory-related ISTG test cases being marked as non-applicable within the allowed access-level. All execution steps and partial findings were documented in the corresponding task checklist item.

5.4.4 Firmware Acquisition and Analysis

The fourth stage of Run 1 was focusing on the initial acquisition of the firmware running on both devices of the BHM. If firmware would have been found, the corresponding test cases would have been executed. Regardless, initially no firmware extraction was possible due to the lack of accessible interfaces. The SWD access was locked on both devices and the UART connector on the base station only read boot log messages and was non-interactive. As a result, all test cases regarding firmware acquisition and initial analysis were considered non-applicable.

While conducting phase 7, mobile application testing, the tester did find valid credentials in the network traffic of the mobile application. These credentials allowed authenticated access to the cloud API and further analysis showed, that the application sends a request to a specific endpoint, which returns all available firmware updates. Using this authenticated route, a firmware package was retrieved while remaining in the limitations of the LoA.

As the package seemed to only be a partial update of the devices' firmware, the tester was not able to extract any binaries or source code but rather analysed the binary file statically. This resulted in information about the development platform (nRF Connect SDK with

Zephyr RTOS[25]) and the implemented bootloader (MCUboot[26]). Extraction of strings revealed names of internal modules, assumed subsystem initialisation procedures, BLE binding logic, and references to communication channels such as MQTT. It was also possible to extract embedded certificates, suggesting that the device verifies the firmware package with a signature, while no signs of encryption were discovered. Configuration values seem to be controlled dynamically via the cloud service. Therefore, no hardcoded credentials, user data, or other sensitive information could be extracted.

After the unexpected acquisition of a firmware package, the results of this stage provided valuable insight into BHM's architecture and interactions with the cloud service. They also enabled the tester to confirm assumptions about implementation details from other phases. The executed test cases can be found in Table 6 below and corresponding evidence is attached and linked to the corresponding tasks in the Task Checklist.

Table 6. Executed Test Cases of Phase 4.

Test Case Identifier	**Topic**
ISTG-FW-INFO-001	Disclosure of Binaries or Source Code
ISTG-FW-INFO-002	Disclosure of Implementation Details
ISTG-FW-INFO-003	Disclosure of Ecosystem Details
ISTG-FW-SCRT-001	Discovery of Hardcoded Secrets in Public Storage
ISTG-FW-SCRT-003	Usage of Hardcoded Credentials
ISTG-FW[UPDT]-AUTHZ-001	Unauthorized Update of Firmware
ISTG-FW[UPDT]-CRYPT-001	Firmware Update Signature Insufficient
ISTG-FW[UPDT]-CRYPT-003	Insecure Transmission of Firmware Updates
ISTG-FW[UPDT]-LOGIC-001	Rollback Protection Insufficient

5.4.5 Processing Unit and Logic Tests

During this test phase, the MCUs of both devices were assessed, specifically focusing on possible disruptions or misuse of their internal logic. As the tester was not able to gain direct interaction with the SoCs, due to locked SWD interfaces and non-interactive

[25] https://www.zephyrproject.org/
[26] https://docs.mcuboot.com/

UART, all tests of this stage were limited to observation-based analysis and minimal interaction through accessible external interfaces of sensors.

The examination confirmed the expected behaviour of the processing units, delivering reliable handling of sensor data and BLE communication. While the tester attempted to interfere with the usual conditions or to modify sensor communication, no abnormal behaviour or crashes were observed. The debugging boot-log of the LTE module showed usual subsystem startup, without revealing any possibilities for influencing the devices' behaviour.

Side-channel attacks were conceptualized, as a safe and non-destructive exploitation could not be guaranteed, forcing the tester to deem practical execution out of scope by the LoA. During research, a possible vulnerability in earlier revisions of the nRF52840 was discovered, which unlocked the SWD interface through voltage glitching[27]. This flaw was fixed in later versions of the SoC, including the one integrated in both devices. Without further deconstruction, prohibited by the LoA, it was not possible to identify other power-analysis aspects or timing artefacts, which could be used for potential side-channel exploits. As a result, no manipulation, direct or indirect, of the MCUs' logic was possible with the given level of access and authorization. The attempted test cases can be found in Table 7.

Table 7. Executed Test Cases of Phase 5.

Test Case Identifier	**Topic**
ISTG-PROC-AUTHZ-001	Unauthorized Interactions with the Processing Unit
ISTG-PROC-LOGIC-001	Insecure Instruction Handling
ISTG-PROC-SIDEC-001	Insufficient Side-Channel Protection

5.4.6 Data Exchange and External Interface Analysis

The sixth stage of the planned assessment methodology involves analysing the interaction between the devices and external services such as the cloud infrastructure. As the LoA prohibited active testing of the cloud service, except discovered legitimate endpoints, this phase included observation of real data flows and structures through the mobile

[27] https://docs.nordicsemi.com/bundle/IN/resource/in_133_v1.0.pdf

application, firmware strings, and network traffic captures. BLE communication between the base station and the sensor unit were also analysed in this phase.

Cloud Communication

During static firmware analysis, the tester discovered MQTT endpoints used by the base station to send and receive data from and to the cloud service. This data not only included the measured sensor data, but also telemetry and configurations, the latter sent by the cloud service to provision the BHM remotely. Additionally, hostnames, MQTT structures, and a public certificate used for TLS were identified in the firmware. Through the use of a proxy server, the network traffic of the mobile application provided insights into API authentication, structures, and administration procedures. The cloud API was never manipulated, fuzzed[28], or actively exploited in any way.

The base station was assumed to be using secure channels to communicate with the cloud service by using Transport-Layer-Security-based (TLS) encryption methods, which prevents an attacker from directly inspecting data packages. This was not verified, as interference with LTE networks is prohibited by national legislation in the country where the assessment is taking place. The MQTT structures found in the firmware strings showed a predictable telemetry data model and suggested the use of the ThingsBoard IoT platform[29], but no sensitive data was found embedded in the firmware itself. The observation of mobile application traffic revealed general implications about authentication handling in the assessed IoT ecosystem, but will be discussed in stage 7, mobile application testing, in more detail.

Wireless Interface (BLE)

The tester used a nRF52840 Dongle to passively scan and sniff the BLE communication between the two units. The sensor unit was identified as the primary client, advertising its presence for the base station to connect to. These advertisement packets contained a public MAC address and a short identifier specific for the manufacturer of the BHM. No

[28] https://developer.mozilla.org/en-US/docs/Glossary/Fuzzing
[29] https://thingsboard.io/

unnecessary or sensitive information such as the firmware version or subsystem structures were discovered. As the devices are shipped to the consumer with an already established binding between the two units, and there was no access to interactively disconnect this pairing, the initial connection process could not be examined.

As a result, Generic Attribute Profile (GATT)[30] packets were captured and interpreted. It was discovered that the communication was not encrypted and assumptions of the data were made but could not be confirmed. Different attempts to connect to either of the devices with the nRF52840 Dongle were rejected, as the existing binding state was enforced. Replaying captured packets did not show any effects and were not relayed to the cloud service by the base station.

The executed tasks and collected evidence form a detailed understanding of the communication between the BHM and the cloud service without actively testing the backend itself, and the BLE channel between the two local devices. This stage contained the biggest number of executed test cases, which can be found in x.

Table 8. Executed Test Cases of Phase 6.

Test Case Identifier	**Topic**
ISTG-DES-AUTHZ-001	Unauthorized Communication via DES
ISTG-DES-AUTHZ-002	Privilege Escalation in Data Exchange Services
ISTG-DES-INFO-001	Disclosure of Implementation Details
ISTG-DES-INFO-002	Disclosure of Ecosystem Information
ISTG-DES-INFO-003	Disclosure of User Data
ISTG-DES-SCRT-001	Access to Sensitive Data
ISTG-DES-CRYPT-001	Insecure Cryptography Algorithms
ISTG-DES-LOGIC-001	Impacts on Intended Business Logic
ISTG-DES-INPV-001	Insufficient Input Validation
ISTG-DES-INPV-002	Command Injection or Code Injection
ISTG-WRLS-AUTHZ-001	Unauthorized Interaction via Wireless Interfaces
ISTG-WRLS-INFO-001	Disclosure of Implementation Details
ISTG-WRLS-INFO-002	Disclosure of Ecosystem Details

[30] https://www.bluetooth.com/bluetooth-resources/intro-to-bluetooth-gap-gatt/

Test Case Identifier	Topic
ISTG-WRLS-INFO-003	Disclosure of User Data
ISTG-WRLS-CONF-002	Identified Unnecessary Software
ISTG-WRLS-SCRT-001	Access to Sensitive Data
ISTG-WRLS-CRYPT-001	Insecure Cryptography Algorithms
ISTG-WRLS-LOGIC-001	Impacts on Intended Business Logic

5.4.7 Mobile Application Testing

As the ISTG does not explicitly include specific test cases for mobile applications of IoT devices but rather references other OWASP Testing Guides, the author decided to include the application as an auxiliary component of the Data Exchange Services. The interactions with the cloud infrastructure were analysed, used API endpoints discovered, and authentication mechanisms examined. The companion application was set up on an emulated Android device and network was proxied through the Burp Suite. This allowed the tester to collect, intercept, manipulate and repeat all network requests sent by the application.

The examined traffic exposed multiple implementation details corresponding to the ISTG-DES category such as hardcoded authentication credentials for cloud API endpoints, MQTT client credentials for the base station, and API calls revealing internal device details. The application primarily communicated with two different domains, one dealing with user specific data like account creation and the other for device data such as measured temperature and other interpreted data generated by the BHM. Both channels used hard coded credentials either via Basic Authentication Header (Base64 encoded) or plaintext in a POST request body and could therefore be extracted and validated. They provided admin-level access to both endpoints, allowing an attacker to fully take over the infrastructure. The MQTT client credentials could be used to imitate the corresponding base station and send invalid sensor data to the cloud.

Even though the mobile application is not considered part of the ISTG categories, it contributed directly to the context of information disclosure, access-control, and the handling of secrets. Therefore, it allowed further understanding and theoretical examination of the cloud infrastructure and led to the discovery of external security-relevant behaviours. The test cases used for this phase are seen in Table 9.

Table 9. Executed Test Cases of Phase 7.

Test Case Identifier	Topic
ISTG-DES-AUTHZ-001	Unauthorized Access to DES
ISTG-DES-INFO-001	Disclosure of Implementation Details
ISTG-DES-INFO-002	Disclosure of Ecosystem Details
ISTG-DES-INFO-003	Disclosure of User Data
ISTG-DES-SCRT-001	Access to Sensitive Data
ISTG-DES-CRYPT-001	Insecure Cryptographic Algorithms
ISTG-DES-INPV-001	Insufficient Input Validation

5.4.8 Internal Interfaces and Side-Channel Analysis

The last analytical testing phase focused on the examination of the internal interfaces discovered in phase 2, and possible side-channel or fault-injection attacks. The accessible communication interfaces were assessed for missing access-control, disclosure of hardware information or other sensitive data, and unintended behaviour through interaction. As defined in the LoA, no destructive or invasive analysis was performed, and all test cases were conducted with the exposed test pads and debugging headers.

With the help of specific low-cost adapters, the UART pads found on the base stations PCB were probed and low-level debugging output of the LTE module was discovered during startup, though only limited information was logged. This included messages about the initialization of internal UART interfaces, directly connected to the nRF52840 MCU, and module identifiers. The interface was read-only and therefore could not be interacted or tampered with. The discovered SWD interface was tested using a 3d-printed connector, due to the implementation of a special pin layout and the high cost of the corresponding connector. After initial connection attempts failed, further research led to the assumption, that the interface was locked by default on the nRF52840 SoCs in production settings. Therefore, no connection via SWD was possible to either of the devices. The circuit board of the sensor unit contained five exposed pads, whose purpose could not be determined.

Side-channel attacks were not executed practically, but rather considered conceptually, as the device's hardware design and implementations did not allow safe, non-destructive analysis. As mentioned in 5.4.5, historical vulnerabilities were no longer present in the

integrated MCUs. Generally, no unintentional disclosure of information or behaviour was observed. The corresponding test cases can be found in Table 10.

Table 10. Executed Test Cases of Phase 8.

Test Case Identifier	Topic
ISTG-INT-AUTHZ-001	Unauthorized Interaction with Internal Interfaces
ISTG-INT-INFO-001	Disclosure of Implementation Details
ISTG-PROC-AUTHZ-001	Unauthorized Access to MCUs
ISTG-PROC-LOGIC-001	Insecure Instruction-Handling
ISTG-PROC-SIDEC-001	Possibility of Side-Channel Attacks

5.4.9 Exploit Validation & Proof-of-Concept Procedures

During this phase it was determined if any of the identified vulnerabilities could be developed into an exploit within the constraints of the LoA. As defined in the methodology, this step aims to validate assumptions from previous phases instead of performing possibly damaging or intrusive exploitation. Each vulnerability was assessed on the required conditions, technical feasibility, and the potential impact on confidentiality, integrity and availability.

The discovered MQTT credentials suggested possibly impersonation of the BHM, which was validated by connecting and publishing a telemetry entry to the broker. The structure of the data was verified with the strings found in the firmware analysis phase, so that the risk of possible negative impact on the cloud service was minimized. The connection was successfully established, and the published data was displayed in the cloud interface.

Additionally, the BLE communication was actively assessed for potential exploitation, but the devices' pairing and binding mechanism did not allow any connections from an unexpected device. It was, however, possible to passively sniff the unencrypted communication between the base station and the sensor unit. After multiple failed attempts to understand the collected data, the tester was able to create a possible interpretation after all. This cannot be verified within the context and limitations of this security assessment but is highly likely to parse the data into real measured temperature and accelerometer data, which would result in lowered confidentiality and potential privacy issues.

The physical connectors such as the debugging interfaces and test pads were also actively interacted with during this phase. Multiple serial adapters and configurations were attempted but concluded with no possible manipulation of the devices. After electrically analysing the testing pads with a logic analyser and physical examination of the corresponding traces on the PCB through a microscope, no explicit use case could be conducted or exploited.

Finally, the hardcoded credentials leaked through the application traffic were verified by logging in to their respective service. While one served a web interface, which provided full administrative access to all devices and configurations through these credentials, no further interaction was conducted, as the service was out of scope. For the other one, no web interface was available, and validation was done by enumerating undiscovered endpoints and sending a corresponding request including the discovered username and password. As this was successful this exploitation was deemed verified, even though no further meaningful exploitation was found.

Combined, these attempts showed that while some attack paths did not lead to active exploitation or were limited by the defined scope, they were technically feasible and possible under different conditions. Therefore, this phase documented the viability of discovered vulnerabilities, even if not performed. All conclusions and evidence are found in the corresponding task checklist item and evidence log and an overview of the test cases that led to attempted exploits can be found in Table 11.

Table 11. Test Cases Corresponding to Attempted Exploits.

Test Case Identifier	**Topic**
ISTG-DES-SCRT-001	Unauthorized Interaction with DES
ISTG-DES-LOGIC-001	Unintended Interaction with Business Logic
ISTG-PROC-AUTHZ-001	Unauthorized Access to MCU
ISTG-PROC-LOGIC-001	Insecure Instruction-Handling
ISTG-WRLS-AUTHZ-001	Unauthorized Access to Wireless Interface
ISTG-WRLS-INFO-003	Disclosure of User Data

5.4.10 Impact Mapping and Documentation

As a last step of the main run the author combined all results generated by previous test stages into an impact-focused overview of the BHM's security. This phase did not

generate new results itself but consolidated findings and mapped them against ISTG categories, ensuring transparency of necessary conditions and context for general risk and affected components.

Both, the likelihood and practical viability of each finding, were considered during the mapping process. As documented in earlier phases, protections on hardware level secured the processing unit from direct compromise, and a secure bootloader prevented firmware tampering. Communication over BLE possibly exposed the measured data due to missing encryption but pairing binding restricted active interaction effectively. Through the usage of LTE as the communication channel between the BHM and the cloud services, the manufacturer successfully prevents simple interception or manipulation, as the complexity of an attack is strongly increased and handled as a criminal offense in many countries. In conclusion, no possible compromises of the BHM itself were discovered during the assessment.

Nevertheless, the tester identified multiple implementation weaknesses in the remaining ecosystem, especially the handling of credential management. Hardcoded credentials with administrative authorization and static MQTT authentication details leaked through network traffic of the mobile application could enable widespread disclosure of sensitive customer information and device takeover. Although it was not technically exploited during this assessment due to the defined scope, the discovery itself leads to a theoretical impact that has to be taken into account.

During the documentation stage, these security-relevant findings were categorized according to the ISTG and linked to the corresponding task checklist items to ensure transparency. The resulting mapping, seen in Table 12, builds the basis for the discussion and recommendations in later chapters.

Table 12. Impact Mapping of Discovered Vulnerabilities.

ISTG Category	**Relevant Area**	**Notes**	**Impact Level**
ISTG-PHY	Physical Access & Hardware Exposure	Appropriate protections in place.	Very Low
ISTG-INT	Internal Interfaces	UART non-interactive; SWD locked.	Very Low

ISTG-PROC	Processing Unit & Logic	No direct access; side-channel attack unlikely.	Low
ISTG-FW	Firmware & Update Mechanisms	Modification possible through administrative access.	Medium
ISTG-WRLS	BLE Interface	Sniffing possible; missing encryption.	Low
ISTG-DES	Data Exchange Services	Disclosure of administrative credentials.	Critical

5.5 Reset of Environments

As defined in the methodology, the tester attempted to return the BHM's ecosystem to initial conditions. This process did not alter the device firmware or internal state, as no available mechanisms are implemented by the manufacturer. The mobile application was fully removed and the emulated device reset to the same state as before Run 1, deleting any data and stored authentication. This ensured that this component of the ecosystem was fully reset.

In addition, the application provided the functionality to clear the pairing state of the sensor unit and the base station. This is done by configuring the stored MAC address of the sensor unit in the base station to an empty MAC address, which forces the two devices to establish a new connection on the next startup. As no other reset mechanisms were discovered, this concluded the reset procedures and set the baseline for Run 2.

5.6 Replication Run / Main Execution Run 2

This stage aimed to verify the reproducibility of a subset of selected test cases executed in Run 1. The author chose tests, that produced either informative or security-relevant results, and no new test cases were included during this phase. The objective was to follow the documented steps of the test cases of Run 1 to confirm the reproducibility of the outcomes of Run 1. The results of this stage support the reliability of the assessment.

Run 2 was conducted with comparable baseline conditions as Run 1. The tester used the same tooling, access levels and configurations established in the main execution. If the devices' behaviour deviated from the documented observations, the differences were recorded and their respective impact analysed. Overall, the test cases executed during this stage showed consistent behaviour to the expected results and therefore strengthened the confidence in the findings.

5.6.1 Chosen Test Cases for Replication

The following Table 13 presents the test cases that were chosen to be executed during the replication run due to their importance to the overall security of the BHM and the clarity of the general results.

Table 13. Chosen Test Cases for Replication Run.

Test Case Identifier	**Topic**	**Reason**
ISTG-WRLS-INFO-001	Disclosure of Implementation Details	Missing encryption of BLE communication and pairing binding stability needed confirmation.
ISTG-DES-SCRT-001	Access to Sensitive Data	Verification that MQTT and API credentials are consistently disclosed.
ISTG-FW-INFO-002	Implementation Details Disclosure from Firmware	Validation of firmware string analysis and OTA package structure remained identical.
ISTG-PROC-AUTHZ-001	Unauthorized Access to MCUs	Ensured that SWD/UART remain inaccessible.

The author selected these test cases, as they collectively represent the whole ecosystem, including wireless interfaces, cloud interaction, the mobile application, firmware, and hardware access interfaces. Additionally, these test cases generated the most distinct results in the main execution, which enables stricter and more precise comparison. In combination, they provide an effective and representative selection for evaluating methodological and technical reproducibility.

5.6.2 Comparative Results of Test Cases

The results of all replicated test cases stayed consistent with those generated during Run 1. There was no observable deviation of the devices' behaviour, information disclosure, debug interface access, or of the cloud API's responses.

- **Wireless Interface (ISTG-WRLS-INFO-001):**
  The content of BLE advertisements, refusal of connection attempts and the GATT structure were identical to the observations of the main execution. No additional data or packets were identified.
- **Exposure of Cloud Credentials (ISTG-DES-SCRT-001):**
  Analysis of the application's network traffic revealed the same set of authentication data as observed in Run 1. Additionally, the discovered API endpoint consistently disclosed the same static MQTT credentials when queried with the authentication details found in application traffic.
- **Firmware Transparency (ISTG-FW-INFO-002):**
  As the firmware version did not change in the time between the two executions, the image included exactly the same strings, API endpoint references, bootloader metadata, and SDK identifiers as found in the main run.
- **Processing Unit Access (ISTG-PROC-AUTHZ-001):**
  Attempts to interact with the LTE module's UART interface remained unsuccessful. Each of the two MCUs did not allow connections via the SWD headers as well, consistent with Run 1.

No general anomalies or other inconsistencies were observed during the replication run. These results suggest a high reliability of the documented findings and confirm stable device behaviour. It is important to note, that the validity may be impacted by the lack of reset mechanisms for the BHM and a missing second set of devices for comparison.

5.7 Practical Summary

The results of the practical security assessment indicate, that the BHM's hardware, firmware and wireless interfaces are secured effectively, with only a few possible low-impact passive attacks. In contrast, the surrounding ecosystem, including the mobile application and cloud APIs, showed vulnerabilities in the handling of credentials and

access control. The following Table 14 links the components and executed ISTG categories to create a structured overview of the results.

Table 14. Executed ISTG Categories per Component.

■ = tested (vulnerability found) □ = tested (no vulnerability discovered) × = not applicable								
	PROC	**MEM**	**FW**	**DES**	**INT**	**PHY**	**WRLS**	**UI**
Sensor Unit	□	×	□	×	□	□	■	×
Base Station	□	×	□	×	□	□	□	×
Cloud Services	×	×	×	■	×	×	×	×
Mobile Application	×	×	×	■	×	×	×	×

With the practical security assessment completed and documented, the following chapter is focusing on the evaluation of the ISTG as a framework for IoT penetration testing. While the author did not find high impact vulnerabilities, the goal of the practical assessment was not to generate a high number of findings, but rather to systematically apply the ISTG framework. The results documented in this chapter build the basis for a comprehensive analysis of the ISTG's structure and guidance.

6 Analysis and Evaluation

This chapter evaluates the strengths, limitations, and applicability of the OWASP ISTG on the basis of the collected empirical results. With chapters 4 and 5 describing the technical methodology and execution, the goal of this section is the assessment of the ISTG itself as a methodological and reproducible basis for IoT penetration testing. As defined in chapter 3, adaptability, reproducibility, and practical suitability are used as evaluation criteria and evidence from both execution runs will be referenced, as well as the task checklist and coverage matrix.

This chapter starts with the discussion of strengths and limitations of the ISTG, followed by the adaptions that were necessary to apply the ISTG to a real-world scenario. The author then compares the framework with the PTES, highlighting differences of their approaches. Lastly, the contributions to the academic and practical field are presented, before concluding with the limitations specific to this single-case evaluation.

6.1 Strengths of ISTG

The conducted practical assessment suggests that the OWASP IoT Security Testing Guide offers a valuable structural basis for penetration testing of multi-layered consumer IoT devices. The framework enabled a systematic breakdown of the whole ecosystem, when applied to the BHM, and supported comprehensive coverage across components. Additionally, the test-case based design allowed a reproducible workflow and documentation.

As for the adaptability defined in chapter 3.3.1, the categories of the ISTG aligned well with the BHM's architecture. The device model was created following the ISTG's distinction between internal components such as the processing unit, memory, and external or physical interfaces. This structure was applied naturally to the two main devices of the BHM, allowing the selection of relevant test cases for each identified component.

The coverage aspect is shown in the summary of executed ISTG categories per component in Table 14. While not every category of the ISTG could be applied to every part of the BHM's ecosystem, it was possible to link each major component to at least one relevant ISTG category. Hardware-focused categories such as physical and internal interfaces (PHY, INT), processing unit (PROC), and firmware (FW) could be applied to the base station and the sensor unit, BLE communication was covered by wireless-related test cases (WRLS), and interactions with the cloud service or mobile application were assessed with the data exchange services (DES) category. This suggests that the internal taxonomy of the ISTG is appropriately representative of a realistic consumer IoT ecosystem, while not requiring custom categories.

Using applicability metric defined in section 3.3.1, 36 out of the 89 ISTG test cases were applicable to the BHM, resulting in an applicability score of 40.45%. This proportion represents the degree to which the provided test-case catalogue could be used to assess the BHM under the scope and constraints defined in the LoA. The result suggests a sufficiently large subset of relevant and applicable test cases to assess the ecosystem meaningfully, while several ISTG categories were left out due to situational limitations.

One of the main strengths, from a methodological perspective, is the individual structure of test cases. With each provided test case already including attributes such as the required physical access and authorization levels, objectives, and possible remediation steps, it enabled the author to directly adapt them for the task checklist. This simplified the conversion of abstract test cases into concrete, executable tasks. While the author did extend the provided test cases with test-specific execution steps, required tooling, and expected secure behaviour, the task checklist was built on top of a clearly defined structural basis provided by the ISTG. This resulted in a consistent application and execution of the task checklist in Run 1 and 2.

Scoping, constraining and documenting the test perspective was greatly enhanced by the device and attacker models defined through the ISTG. The definition of available access levels for physical components (PA) and authorization levels (AA) by the ISTG, it was possible to precisely specify that the simulated attacker has full physical access (PA-4) while not having any authorized access to the device (AA-1) and legitimate customer access rights (AA-2) to the mobile application and cloud. This resulted in a defined and realistic attack scenario, which enabled the selection of relevant test cases for the

permissions and limitations set by the LoA. With the provided access level requirements of each ISTG test case, a transparent and clear selection of applicable tests was possible without ad-hoc adjustments during the assessment.

A further strength of the ISTG is its ability to complement an external process model resulting in a full assessment workflow. As the framework does not provide an order of execution or prioritization, the categorised structure was mapped easily to the logical assessment procedure used in this study, which was derived from the PTES. While the ten phases executed in the empirical case-study were not imposed by the ISTG, the underlying test-case catalogue was directly mapped onto the chosen workflow. Therefore, it was possible to cover all aspects of the BHM's ecosystem defined in the Device Models and even extend to the mobile application and cloud services. The resulting distribution of applied test cases across all layers suggests that the categories defined by the ISTG are flexible enough to be integrated with external process structures, while still supporting a comprehensive multi-layered assessment.

The ISTG also positively contributes to reproducibility of security assessments, as indicated by this study. With an independent second execution of selected test cases, it was observed that behaviours documented during the first execution stayed consistent across the reproduced execution as well. Static firmware analysis resulted in identical metadata and discovered strings, BLE communication exposed the same information, while the cloud API and debugging interfaces showed matching behaviour too. These results suggest that ISTG's test cases can be reapplied reliably when supported by the task checklist and evidence log. The stability of the observations during this assessment implies that the ISTG can be implemented and used for reproducible penetration-tests for comparable consumer IoT devices.

Lastly, comprehensive documentation and traceability was supported by the ISTG as well. Through combining the task checklist and evidence log a clear link was created between conducted test cases, respective executed steps, and observed behaviours. As highlighted by the reviewed literature, IoT security studies sometimes lack reproducibility and documentation quality. By combining the ISTG's test case structure with extended attributes, this study demonstrates that the ISTG is not only a collection of technical checks but rather can be used as a foundation for transparent and verifiable IoT security assessments.

6.2 Limitations of ISTG

Although the structured basis provided by the ISTG has a positive impact on IoT security assessments, several limitations were discovered while applying the framework to the BHM. These concern the adaptability of test cases, the lack of practical guidance, and dependencies on specific conditions. Together, these limitations influence the overall practical suitability defined in chapter 3.3.3. Where relevant, introduced adaptions by the author are documented for each discovered shortcoming.

The first constraint involves the applicability of ISTG categories focused on hardware. Multiple test cases of the memory or processing unit categories could not be executed as external memory chips were not implemented or because the MCUs' debugging interfaces were locked. This led to multiple individual tests and an entire category of test cases being considered not applicable. While the origin of this limitation is the BHM's architecture rather than a deficient ISTG, it demonstrates that the framework is dependent on assumptions that might not hold for many consumer IoT products. As a result, the applicability of some ISTG categories to commercial IoT devices may be lower than the framework designers intended.

Test cases of the data exchange service (DES) category were discovered to be limited as well, especially those requiring privileged access to these services. Several test cases could not be conducted as access was constrained by the assessment conditions and the LoA. As mentioned in chapter 3.3.1, the general applicability of the framework is not only influenced by architectural diversity but also by attack permissions. No fallback strategies or alternative testing approaches are provided by the ISTG, leading to complete exclusion of test cases where the required privilege cannot be obtained. As a result, practical suitability is reduced in real-world assessments where contractual or regulatory boundaries limit given attack paths, even though vulnerabilities theoretically may exist.

Another limitation is linked to gaps in the coverage of BHM's ecosystem-level behaviour. The ISTG offers test categories for individual components but does not explicitly define a structure to address workflows involving interconnected subsystems, such as the chain of the sensor unit, the base station and the cloud services. Some specific behaviours of this chain, such as provisioning of the devices, had no associated test case and would have

required the author to interpret and adapt existing categories. This emphasizes the absence of ecosystem-oriented test cases as a structural gap of the ISTG.

The lack of procedural guidance, already mentioned in section 6.1, is also deemed a limitation, affecting practical suitability and reproducibility. As the ISTG does only provide a catalogue of test cases and does not define an order of execution or prioritisation, the PTES-inspired workflow used in this study was developed by the author. Without a provided process structure, the framework relies heavily on individual and independent operationalization by the tester to create a safe, legal, and effective execution workflow. As a result, this leads to an increased dependency on the expertise of the security professional, which in turn reduces consistency and reproducibility across penetration tests.

Similar to the missing procedural guidance, the author defined the level of abstraction of test cases provided by the ISTG as a limitation as well. The descriptions of individual tests only offer high-level explanations without operational guidance. Therefore, many test cases are requiring the practitioner to interpret and translate the given information into concrete execution steps. This leads to reduced practical suitability, as defined in section 3.3.3, due to deficient clarity and feasibility. For the practical assessment conducted in this thesis, the author addressed this by extending the ISTG test cases with custom attributes to create an executable, traceable and repeatable workflow. Consequently, the need for these adaptions suggests that the ISTG currently may not fully support security professionals without domain specific knowledge or assessors operating under time constraints typical of commercial penetration tests.

The last limitation discovered by the tester was the lack of evidence handling processes. OWASP does provide their own checklist, which includes test case identifiers and names, extended with a status field and a column for notes. These attributes are not sufficient for creating a transparent and repeatable assessment documentation. Even though most commercial security practitioners may use their respective reporting template, it creates an individualized approach, reducing independent verification and reproduction. For this reason, the author introduced the evidence log, which documents and captures all relevant information such as tool outputs or screenshots. This enables transparent results and serves as a point of reference for other security professionals.

In conclusion, these limitations suggest that while the ISTG provides a strong structural basis, its usability in real-world IoT security assessments is limited by device-specific architectures, legal constraints, minimal procedural guidance, and the requirement of individual interpretation. On the basis of the discussed limitations, the next section discusses the adaptions of the ISTG undertaken by the author for the assessment.

6.3 Comparison with PTES

To further enhance the evaluation of the ISTG, this section compares the framework with the Penetration Testing Execution Standard (PTES), which is one of the most widely used process-oriented methodologies for security assessments [42]. Even though both frameworks aim to enable structured penetration tests, their scope, level of abstraction, and intended use differ substantially. Through this research, these differences illustrate how ISTG can be used as a complementary methodology but not as a replacement for a process-driven standard such as the PTES.

While the ISTG offers a test-case catalogue without specific ordering or execution conditions, the PTES provides a structure based on phases, which define clear processes for each phase of the security assessment. As mentioned in section 6.2, the author constructed an independent assessment procedure inspired by PTES, due to the lack of practical guidance by the ISTG. A fundamental distinction is emphasized: while the PTES controls how an assessment is conducted, the ISTG governs the scope of what is tested. This distinction is illustrated below in Table 15 by presenting the structural components of the ISTG and PTES side by side.

Table 15. Structural Comparison of ISTG and PTES.

Dimension	**ISTG**	**PTES**
Primary Function	Provide structured IoT-specific test cases	Define penetration testing phases and workflow
Structure	Independent categories	Sequential phases
Abstraction Level	Technical test definitions	High-level procedural guidance
Device Focus	IoT ecosystems	Generic IT systems
Order of Execution	Not defined	Defined
Requirements for Use	Test interpretation & high expertise	Process understanding & moderate expertise

Dimension	ISTG	PTES
Evidence Guidance	Not provided	Provided

This comparison highlights the different layers of operation for both frameworks. For the assessment conducted in this research, PTES provided the basis for the procedural structure, enabling ordering and prioritization of actions, while the ISTG contributed the technical aspects for component-specific evaluation. As a result, the practical approach used in this thesis turned into a hybrid methodology, with a workflow inspired by the PTES and technical tasks defined by the ISTG.

The two frameworks also build on different expectations of available expertise of the tester. The ISTG assumes deep domain-specific knowledge, due to the necessary interpretation of test cases in the context of IoT devices, whereas the PTES requires the practitioner to understand, interpret, and execute general penetration testing concepts such as reconnaissance or exploitation. As discussed in 6.2, this leads to reduced practical suitability for security experts with limited IoT-specific knowledge. Based on the experience gained during the conducted assessment, the cognitive load required from the author appeared to be substantially different between the two frameworks. While the PTES reduced mental load by providing structural aspects, the ISTG added cognitive overhead of interpretation and operationalization of each test case. It is important to note, that this observation is subjective and specific to this case study and tester.

Finally, in the context of completeness, PTES's generality is complemented by ISTG's IoT-specific component coverage. The assessment conducted during this research demonstrates that neither framework alone is sufficient for a comprehensive consumer IoT security test, as the ISTG lacks process structure, and the PTES is missing IoT focus. However, in combination these frameworks offer a robust security assessment methodology, with PTES enabling a clear workflow, and ISTG ensuring technical coverage.

6.4 Contribution to Research & Practice

This study aims to contribute to two areas of cybersecurity. The first is general academic research on methodologies for IoT security testing, while the second concerns the real-world application of penetration testing frameworks. As described in chapter 3, the

defined research questions focus on the assessment of the strengths and limitations of the ISTG framework as a foundation for an assessment methodology, its potential value for security experts, and its suitability for reproducible and systematic security tests. The conducted assessment and evaluation address these attributes and results in several contributions, which are described in more detail in this chapter.

The first contribution is the empirical validation of the ISTG framework. Even though other OWASP frameworks are widely adopted in the industry, the ISTG has only been academically evaluated to a limited extent. At the time of writing, no studies examining its applicability in a multi-layered IoT ecosystem context. Therefore, through the application of the ISTG to the BHM's comprehensive security assessment, the study provides the first documented results of its performance across interconnected, heterogeneous components. Future comparative research on IoT testing methodologies or frameworks is supported by the calculation of the applicability score as a quantitative reference. Through this validation, the author addresses the academic gap of empirical research on IoT security methodologies in real-world settings.

Another contribution consists of the identification of operational and structural limitations affecting the usability of the ISTG. These include a lack of coverage on the ecosystem level, the required interpretation of provided test cases, and missing procedural guidance and evidence handling. As highlighted in section 6.2, the ISTG may be deficient in some areas when applied to consumer IoT devices, especially the tester is bound to legal and architectural constraints. By evaluating the findings through the methodological criteria specified in chapter 3, this study provides a reproducible, transparent and structured model for framework evaluations.

The third contribution is the development of methodological adaptions deemed necessary during the assessment. In combination, the task checklist, evidence log, and custom execution step documentation form a comprehensive layer to operationalize the ISTG, compensating the lack of procedural guidance. The conceptual test-catalogue provided by the ISTG therefore gets translated into an applicable and repeatable testing methodology. As demonstrated through the repeated execution of Run 2, the developed adaptions enable consistent results and provide a foundation for traceable evidence handling. These artefacts can further be used by security experts as references or

templates while applying the ISTG to comparable devices, resulting in increased usability and reduced operational cognitive load described in section 6.3.

Additionally, the comparison between the ISTG and the PTES contributes to general cybersecurity research, as it demonstrated the integration of both, process-oriented and test-case-oriented frameworks. The hybrid workflow developed and applied during the assessment offers a procedural structure combining ISTG's IoT-specific test cases with PTES's workflow structure. Researchers and practitioners designing comprehensive IoT testing methodologies without depending solely on one framework may use this insight as a reference. It also suggests that IoT penetration testing requires multi-layered methodologies addressing both technical and process structure.

Finally, the author contributes to industry practice by demonstrating possible challenges that practitioners can face when conducting consumer IoT security assessments. The observed limitations, such as restricted access or locked interfaces, prove why careful scoping, realistic attacker models, and multi-layered frameworks are required for IoT penetration tests. This thesis therefore may support security professionals in planning, execution, and documentation of similar security assessments by providing the adapted methodology and templates for documentation. The summarized contributes are listed in Table 16.

Table 16. Overview of Contributions of This Work.

Contribution Area	**Contribution Summary**	**Evidence in Thesis**
Framework Evaluation	First practical assessment of ISTG across full consumer IoT ecosystem	Chapter 5-6
Methodology	Hybrid ISTG-PTES workflow; evaluation metrics	Chapter 3-4, 6
Practical Evidence	Task Checklist; Evidence Log	Appendix & GitHub
Academical Gaps	Practical evaluation of IoT security methodologies	Section 6.4
Industry Relevance	Reference for constrained IoT security assessments	Chapter 5, Section 6.2

6.5 Limitations of This Study

Even though the author attempted to minimize the limitations of this study, several constraints of the generalisability of the findings remained. These originate from methodological choices, external restrictions shaping scope and depth of the assessment, and characteristics specific of the BHM. While the validity of the evaluation is not negated, these limitations construct the boundaries within which the results of this study should be interpreted.

The first constraint involves the single-case study design. The author uses one consumer IoT device with a specific multi-layered architecture to evaluate the ISTG framework. As mentioned in chapter 3, while case studies offer practical depth, external validity is limited. The calculated applicability score of 40% and discovered strengths and weaknesses of the ISTG may vary substantially for devices with other architectures, such as industrial IoT systems or devices with more accessible interfaces. By evaluating multiple device types, future academic works would allow for stronger claims about the generalised performance of the ISTG.

A second limitation is related to the conditions of the assessment defined by the LoA. Certain test cases and ecosystem interactions could not be evaluated, as the interaction to the cloud services was restricted and all testing activities needed to remain non-destructive. As discussed in section 6.2, no fallback strategies are provided by the ISTG for situations where required access levels are not obtainable. As a result, this study indicates ISTG's performance under realistic but constrained testing conditions, with regulatory and contractual restrictions preventing full execution of the whole test catalogue.

The BHM's architectural constraints result in another limitation, specifically the inaccessible debugging interfaces and the lack of external memory modules. The execution of the whole memory test category (MEM) and some processing unit test cases (PROC) was prevented, which negatively influenced the applicability score. Even though these restrictions are widespread in modern consumer IoT devices, and generally have a positive impact on overall security, they limit the degree of the generalisation of conclusions related to ISTG's hardware categories.

The overall academic validity is also limited by this thesis' reliance on just one assessor, whose expertise, domain knowledge, and interpretation influenced the method of operationalising the ISTG. As described in chapter 3, repeated execution of a selected subset of test cases was used to evaluate reproducibility, but as Run 2 was conducted by the same tester, reproducibility between different assessors could not be examined. Due to the required interpretation of high-level test cases into actual execution steps, outcomes may vary depending on the tester's expertise and preferences.

Another limitation concerns the temporal stability of the conducted assessment and corresponding results. As the execution was performed within a defined timeframe, the observations only capture a snapshot of the current state of the BHM's ecosystem. Updates to the firmware, cloud endpoints, mobile application or hardware composition could lead to variation in applicability, observed behaviours, and discovered vulnerabilities in the future. The reproducibility metric defined in this thesis therefore only represents stability within the time window of the penetration test.

Finally, the author excluded regulatory and privacy analyses from the scope of this study, focusing entirely on testing the BHM's security. While this works for the defined research objectives and the ISTG, it reduces the extent to which the interpreted results can be applied to broader evaluations of IoT environments.

7 Conclusion

This final chapter summarizes the methodological and empirical results collected during this study and establishes them in the broader context of evaluating the applicability, strengths, and limitations of the OWASP ISTG framework for penetration testing of consumer IoT devices. The findings from the theoretical analysis, the structured methodology, and the practical assessment of the BHM are collected and the effectiveness of the ISTG in supporting a reproducible, systematic, and technically feasible security assessment is reiterated. Additionally, implications for other security experts stemming from these results are discussed, before describing recommended future works and framework refinements.

7.1 Summary of Findings

The aim of this thesis was to evaluate how effectively the ISTG can support a reproducible and structured security assessment workflow for consumer IoT devices. By applying the framework to the BHM, it was demonstrated that the ISTG offers a comprehensive starting point, but practical limitations, gaps in coverage, and the requirement of tester interpretation were discovered.

From the empirical perspective, the BHM showed strong hardware and firmware security throughout the security assessment. Debugging interfaces were not accessible, firmware extraction was not possible from the device itself, and other data exposure was limited to minimal information. The only security-relevant insecure finding was the missing encryption of the BLE communication, which allowed passive collection of sensor data. These findings suggest a good basis of embedded security for the BHM, even though the absence of attack vectors highlighted the dependency of ISTG test categories on the availability and accessibility of specific components, such as debugging interfaces or external memory modules.

The surrounding ecosystem of the BHM posed more significant observations. Throughout the mobile application and cloud services, weak credential handling, missing

authorization boundaries, and information disclosure were identified, demonstrating that security vulnerabilities often do not originate from the IoT device itself but from the surrounding connected ecosystem. These findings underline the importance and difficulty of multi-layered IoT security assessments, while simultaneously revealing the limits of ISTG guidance, whereafter it becomes abstract or insufficient in practice.

Methodologically, the IoT Security Testing Guide scored an applicability rating of 40.45% by providing 36 applicable test cases within the defined scope. While this suggests that the ISTG is widely representing real-world IoT architectures, it also shows that a significant amount of test cases cannot be used without specific conditions. Through the methodological extension with the introduction of the task checklist and evidence logs, the author enabled consistent replication conditions for selected test cases, supporting ISTG's reproducibility when applied systematically.

In combination, the results of this thesis demonstrate that the ISTG is most effective as a technical foundation rather than a complete methodology for security assessments. While it ensures depth and systematic coverage, it requires complementary structures such as a PTES-based workflow, defined evidence handling processes, and tester-specific execution to allow for scientific reproducibility and real-world applicability.

7.2 Answers to Research Questions

This section summarises how the observations and results of this case study answer the research questions described in section 1.3.

RQ1: How effectively does the OWASP ISTG framework support a structured and reproducible penetration testing process for consumer IoT devices?

The results show that the ISTG framework offers clear and organised structures improving consistency and transparency of IoT security assessments. Through the categorization of test cases into categories and the definition of predefined attributes it was possible to systematically transform them into the Task Checklist, which enabled a traceable and documented process. The methodologically limited assessment of reproducibility suggested stable results when tests were conducted under controlled conditions.

However, as the reproducibility relied strongly on interpretation of the tester and on external factors, it does not eliminate the individual impact of the conducting practitioner. Additionally, an external process flow was required to create a comprehensive assessment procedure, increasing subjective decision making on how to execute the given test cases.

RQ2: Which strengths and limitations of the ISTG framework become apparent when applied in a real-world black-box penetration test?

The author observed several strengths during the practical assessment. The overall coverage of the ISTG includes all major components of an IoT device, reducing the chance of overlooking attack surfaces. The modular design allowed for efficient mapping of test cases to device components, and the provided test case information enabled flexible extension to other parts of the device ecosystem.

At the same time, notable limitations were discovered as well. As the test cases do not include execution details, the framework depends on the testers domain knowledge to fill conceptual gaps and define execution steps. This increases cognitive effort and may lead to inconsistencies in the application between different security professionals. Furthermore, the ISTG does not include guidance for multi-component systems or chained vulnerabilities.

RQ3: To what extent can the documented testing process serve as a reusable reference or manual for practitioners working with similar IoT devices?

The produced structured documentation such as the Task Checklist and the Evidence Log demonstrate that the ISTG can be used as a foundation to build a reproducible and transparent testing workflow around it. These artefacts create a practical reference for other researchers or practitioners to reuse or adapt them for comparable IoT testing conditions.

The methodology used in this thesis also demonstrates that the identified gaps of the ISTG can be bridged by implementing additional steps and conditions. While the results of this case study cannot be generalized, due to the specific device context, the documentation and workflow are transferable to other IoT devices. The author contributed to this field by showing the operationalization of the ISTG in a real-world security assessment,

resulting in a concrete and reusable template which may be applied, extended, or refined by other security specialists.

7.3 Future Work

This thesis highlights multiple possible directions for future development of methodologies and research in the IoT security assessment field. With the limitations discussed in section 6.5, further academic research is necessary to evaluate the ISTG's generalizability across a wider range of different IoT devices and infrastructures.

The first identified direction involves the validation of ISTG's applicability in other IoT industry contexts, such as healthcare or industrial IoT ecosystems. These ecosystems should contain various architectures and complexity to allow for a broad and systematic comparison of coverage, reproducibility, and adaptability of the ISTG.

Future research could also explore the combination and integration of the ISTG and other frameworks or standards. As the ISTG is structured in components, it could benefit from procedural guidance sourced from methodologies such as the PTES. To generally strengthen operational clarity a selective implementation of elements from other OWASP standards or guides could be evaluated, while defining clear boundaries and mappings between the ISTG and the integrated framework.

Expanding on the results of this thesis, the operational guidance of the ISTG itself could also be refined through future academic work. While the Task Checklist (see Appendix 3) offers execution steps, these are limited to the executed assessment and the author's interpretation. Future research could develop general execution templates or attack paths, supporting testers in planning concrete assessment actions. By submitting these works as proposed extensions to the authors of the ISTG, researchers would support ISTG's maturation and usability.

Lastly, quantitative evaluation methods for assessment methodologies, such as reproducibility or coverage, could be investigated. Based on the research process of this study, similar experiments could be conducted involving multiple testers applying the same framework to the same device. This could result in additional insights into the dependency on individual expertise in contrast to methodological guidance.

7.4 Final Remarks

The OWASP IoT Security Testing Guide was evaluated as a methodological basis for penetration testing of a consumer IoT device through an empirical case study. This research focused on examining the effectiveness of ISTGs support for structured security assessments, its coverage across realistic IoT environments, and its reproducibility of findings, rather than directly on the security of the assessed device.

The results confirm that the ISTG offers a comprehensive structure of IoT components and corresponding security concerns, which supports testers in systematically identifying relevant attack surfaces across most parts of the ecosystem. However, it is shown that the operational guidance is limited, which creates the reliance on prior expertise to translate the test cases into practical penetration testing steps. As a result, the author implemented different methodological steps to achieve reproducible findings and efficient usability of the ISTG.

Through the empirical application and critical evaluation, this thesis contributes insights that extend beyond practical security assessments. While underlining the necessity of integration with procedural methodologies or additional execution guidance, the findings are also positioning the ISTG as a framework well suited for structuring IoT penetration tests.

In conclusion, the importance of evaluating frameworks and methodologies for security assessments empirically, rather than solely on a conceptual level, is highlighted through this work. As the complexity and societal impact of IoT systems continue to increase, empirical assessments are critical to enable further improvement of applicable and technically comprehensive penetration testing methodologies.

References

[1] "Glossary:Internet of Things (IoT) - Statistics Explained - Eurostat." Accessed: Dec. 08, 2025. [Online]. Available: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Glossary:Internet_of_Things_(IoT)

[1] "Glossary:Internet of Things (IoT) - Statistics Explained - Eurostat." Accessed: Dec. 08, 2025. [Online]. Available: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Glossary:Internet_of_Things_(IoT)[2] Transforma Insights, "Number of Internet of Things (IoT) connected devices worldwide from 2019 to 2034, by use case (in millions) [Graph]," https://www.statista.com/statistics/1194701/iot-connected-devices-use-case/.

[3] Statista, "Consumer IoT - Worldwide," https://www.statista.com/outlook/tmo/internet-of-things/consumer-iot/worldwide.

[4] "Internet of Things: market data & analysis | Statista." Accessed: Dec. 08, 2025. [Online]. Available: https://www.statista.com/study/109197/internet-of-things-market-outlook-report/

[5] S. Vashi, J. Ram, J. Modi, S. Verma, and C. Prakash, "Internet of Things (IoT): A vision, architectural elements, and security issues," in *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, IEEE, Feb. 2017, pp. 492–496. doi: 10.1109/I-SMAC.2017.8058399.

[6] "Regulation (EU) 2016/679 of the European Parliament and of the Council on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) - Article 4." Accessed: Dec. 08, 2025. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32016R0679&from=EN#d1e1489-1-1

[7] Dimitris Paraskevopoulos, "Challenges with IoT product launches: Why time-to-market has increased." Accessed: Dec. 08, 2025. [Online]. Available: https://iot-analytics.com/challenges-iot-product-launches-why-time-to-market-has-increased-80-percent-in-4-years/

[8] Verizon, "2023 Mobile Security Index white paper," 2023. Accessed: Dec. 08, 2025. [Online]. Available: https://www.verizon.com/business/resources/Tc61/reports/mobile-security-index-report.pdf

[9] “Internet Of Babies - When Baby Monitors Fail To Be Smart - SEC Consult.” Accessed: Dec. 08, 2025. [Online]. Available: https://sec-consult.com/blog/detail/internet-of-babies-when-baby-monitors-fail-to-be-smart/

[10] “Vulnerabilities Identified in Nooie Baby Monitor.” Accessed: Dec. 08, 2025. [Online]. Available: https://www.bitdefender.com/en-us/blog/labs/vulnerabilities-identified-in-nooie-baby-monitor

[11] OWASP, “OWASP IoT Security Testing Guide.” Accessed: Dec. 08, 2025. [Online]. Available: https://owasp.org/owasp-istg/

[12] V. C. Hu, R. Kuhn, and D. Yaga, “Verification and Test Methods for Access Control Policies/Models”, doi: 10.6028/NIST.SP.800-192.

[13] “The Penetration Testing Execution Standard.” Accessed: Aug. 20, 2025. [Online]. Available: http://www.pentest-standard.org/index.php/Main_Page

[14] Pete Herzog, “OSSTMM 3 - The Open Source Security Testing Methodology Manual,” *ISECOM*, Accessed: Aug. 20, 2025. [Online]. Available: https://www.isecom.org/OSSTMM.3.pdf

[15] I. Butun, P. Osterberg, and H. Song, “Security of the Internet of Things: Vulnerabilities, Attacks, and Countermeasures,” *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 616–644, 2020, doi: 10.1109/COMST.2019.2953364.

[16] IEEE, “Internet of Things (IoT) Ecosystem Study,” Jan. 2015. Accessed: Dec. 08, 2025. [Online]. Available: https://iot.ieee.org/images/files/pdf/iot_ecosystem_exec_summary.pdf

[17] Fortinet, “Top IoT Device Vulnerabilities: How To Secure IoT Devices.” Accessed: Dec. 08, 2025. [Online]. Available: https://www.fortinet.com/resources/cyberglossary/iot-device-vulnerabilities

[18] R. Williams, E. McMahon, S. Samtani, M. Patton, and H. Chen, “Identifying vulnerabilities of consumer Internet of Things (IoT) devices: A scalable approach,” in *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*, IEEE, Jul. 2017, pp. 179–181. doi: 10.1109/ISI.2017.8004904.

[19] K. Murat *et al.*, “Security Analysis of Low-Budget IoT Smart Home Appliances Embedded Software and Connectivity,” *Electronics (Basel)*, vol. 13, no. 12, p. 2371, Jun. 2024, doi: 10.3390/electronics13122371.

[20] A. Shanley and M. N. Johnstone, “Selection of penetration testing methodologies: A comparison and evaluation,” in *Australian Information Security Management Conference, AISM 2015*, SRI Security Research Institute, Edith Cowan University, 2015, pp. 65–72. doi: 10.4225/75/57b69c4ed938d.

[21] A. Rizzardi, S. Sicari, and A. Coen-Porisini, “Analysis on functionalities and security features of Internet of Things related protocols,” *Wireless Networks*, vol. 28, no. 7, pp. 2857–2887, Oct. 2022, doi: 10.1007/s11276-022-02999-7.

[22] K. U. Sarker, F. Yunus, and A. Deraman, “Penetration Taxonomy: A Systematic Review on the Penetration Process, Framework, Standards,

Tools, and Scoring Methods," *Sustainability*, vol. 15, no. 13, p. 10471, Jul. 2023, doi: 10.3390/su151310471.

[23] J. Frankland, "The importance of standardising methodology in penetration testing," *Database and Network Journal*, vol. 39, p. 13, 2009.

[24] Foudil and Shafranovich, "RFC 9116," Internet Engineering Task Force (IETF). Accessed: Nov. 05, 2025. [Online]. Available: https://www.rfc-editor.org/rfc/rfc9116.html

[25] "ISO/IEC 29147:2018 - Information technology — Security techniques — Vulnerability disclosure." Accessed: Dec. 08, 2025. [Online]. Available: https://www.iso.org/standard/72311.html

[26] G. Bella, P. Biondi, S. Bognanni, and S. Esposito, "PETIoT: PEnetration Testing the Internet of Things," Feb. 2023, Accessed: Aug. 19, 2025. [Online]. Available: http://arxiv.org/abs/2302.04900

[27] M. Antonakakis *et al.*, "Understanding the Mirai Botnet", Accessed: Nov. 02, 2025. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis

[28] T. Alladi, V. Chamola, B. Sikdar, and K.-K. R. Choo, "Consumer IoT: Security Vulnerability Case Studies and Solutions," *IEEE Consumer Electronics Magazine*, vol. 9, no. 2, pp. 17–25, Mar. 2020, doi: 10.1109/MCE.2019.2953740.

[29] W. Zhou *et al.*, "Discovering and Understanding the Security Hazards in the Interactions between IoT Devices, Mobile Apps, and Clouds on Smart Home Platforms."

[30] S. M. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, and K. S. Kwak, "The internet of things for health care: A comprehensive survey," *IEEE Access*, vol. 3, pp. 678–708, Jun. 2015, doi: 10.1109/ACCESS.2015.2437951.

[31] M. Zahedian Nezhad, A. J. J. Bojnordi, M. Mehraeen, R. Bagheri, and J. Rezazadeh, "Securing the future of IoT-healthcare systems: A meta-synthesis of mandatory security requirements," *Int J Med Inform*, vol. 185, p. 105379, May 2024, doi: 10.1016/J.IJMEDINF.2024.105379.

[32] R. Somasundaram and M. Thirugnanam, "Review of security challenges in healthcare internet of things," *Wireless Networks*, vol. 27, no. 8, pp. 5503–5509, Nov. 2021, doi: 10.1007/S11276-020-02340-0/TABLES/5.

[33] M. Stanislav and T. Beardsley, "Hacking IoT: A Case Study on Baby Monitor Exposures and Vulnerabilities," 2015. [Online]. Available: https://www.rapid7.com/disclosure.jsp

[34] M. Hilding and M. Nordström, "Exploring Ethical Hacking by Identifying Vulnerabilities in Motorola BabyMonitor MBP855CONNECT(4855)," Stockholm, 2021.

[35] Cyber, "TS 103 701 - V1.1.1 - CYBER; Cyber Security for Consumer Internet of Things: Conformance Assessment of Baseline Requirements," 2021. [Online]. Available: https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

[36] R. Kaksonen, K. Halunen, M. Laakso, and J. Röning, “Automating IoT Security Standard Testing by Common Security Tools,” in *Proceedings of the 10th International Conference on Information Systems Security and Privacy*, SCITEPRESS - Science and Technology Publications, 2024, pp. 42–53. doi: 10.5220/0012345900003648.

[37] Google, “ OSS-Fuzz,” https://google.github.io/oss-fuzz/. Accessed: Nov. 05, 2025. [Online]. Available: https://google.github.io/oss-fuzz/

[38] A. Shanley and M. N. Johnstone, “Selection of penetration testing methodologies: A comparison and evaluation,” *Australian Information Security Management Conference*, pp. 65–72, Jan. 2015, doi: 10.4225/75/57b69c4ed938d.

[39] “IoT Device Model - OWASP IoT Security Testing Guide.” Accessed: Dec. 08, 2025. [Online]. Available: https://owasp.org/owasp-istg/02_framework/device_model.html

[40] “ISP1507 Series - Bluetooth 5.0 Modules with Antenna.” Accessed: Dec. 08, 2025. [Online]. Available: https://www.insightsip.com/products/bluetooth-le-modules/isp1507#doc

[41] “Attacker Model - OWASP IoT Security Testing Guide.” Accessed: Dec. 08, 2025. [Online]. Available: https://owasp.org/owasp-istg/02_framework/attacker_model.html

[42] N. J. Van den Hout, “Standardised Penetration Testing? Examining the Usefulness of Current Penetration Testing Methodologies,” 2019. Accessed: Dec. 12, 2025. [Online]. Available: https://www.researchgate.net/publication/335652869_Standardised_Penetration_Testing_Examining_the_Usefulness_of_Current_Penetration_Testing_Methodologies

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis[31]

I, Thomas Fankhauser

1 grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis “Evaluating the OWASP IoT Security Testing Guide: A Case Study on Consumer IoT Penetration Testing”, supervised by Shaymaa Mamdouh Khalil

1.1 to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;

1.2 to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.

2 I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.

3 I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

[31] The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.

Appendix 2 – Letter of Authorization Template

Letter of Authorization (LoA)

Between:

Authorizing Organization
[ORGANIZATION NAME]
[ORGANIZATION ADDRESS]
[CITY, POSTAL CODE, COUNTRY]

and

Authorized Tester / Researcher
[RESEARCHER NAME]
[RESEARCHER ADDRESS]
[CITY, POSTAL CODE, COUNTRY]

---

1. Purpose / Zweck

The Authorizing Organization hereby authorizes the Authorized Tester to perform security testing and penetration testing activities for the purpose of academic or scientific research, conducted as part of a thesis, study, or equivalent research project.

2. Scope of Testing

The following systems and components are **within scope**:

- Hardware, firmware, and software of the [device]
- Mobile application (current App Store version at the time of testing and available older versions)
- Cloud infrastructure (reachable via device analysis, black-box testing)
- APIs and servers discovered during testing (only non-destructive testing)
- Physical access to the device

Out of Scope:

- Any domains outside the ownership of [domain]
- Brute-force attacks outside of the **local** system environment
- Denial of Service (DoS) attacks (except for potential **local** disruptions of device communication without operational impact)

3. Testing Process

Once [tester] receives the physical device, an initial local testing phase will begin. This includes all testing and analysis that can be performed locally, without sending data to [company]'s infrastructure. This phase also functions as reconnaissance and supports modelling and planning of attack paths.

Based on these findings, [tester] will prepare a detailed execution plan for any external attack paths, including the exact commands to be run. This plan will be submitted to [company] for review and approval.

Only after explicit approval will a specific timeframe be agreed upon for executing the external testing phase.

This staged process ensures that no unintended damage occurs and that every action outside of the local system environment is pre-approved by [company].

4. Timeframe

Local, physical Tests can be executed at any time. All external tests will only be executed in the specifically defined timeframe.

5. Conduct

All testing activities shall be conducted in a responsible, professional, and non-destructive manner. [tester] will not intentionally cause harm to systems, services, or data integrity.

Errors or disruptions caused by legitimate testing activities are acknowledged as acceptable, provided no operational damage occurs to [company].

[tester] assumes no liability for any direct or indirect damages, loss of revenue, data loss, or other adverse effects that may occur during or as a result of the authorized testing. However, he commits to exercising the utmost care and minimizing any potential risk to [company] as far as possible.

6. Reporting and Confidentiality

[tester] will deliver a full report of findings to [company] no later than 14 days after the conclusion of the testing period.

A separate confidentiality agreement (NDA) will be signed by both parties.

7. Contact

Primary contact and emergency contact for this engagement:

[contact name]
[position]
[email / phone]

8. Legal Jurisdiction

This agreement is governed by the laws of [country].

Signatures

For [company]	**For [tester]**
Name: _____	Name: [tester]
Position: _____	
Date: _____	Date: _____
Signature: _____	Signature: _____

Appendix 3 – Task Checklist

ISTG Test Case Identifier	**Name**
ISTG-PHY-AUTHZ-001	Unauthorized Interaction with Physical Ports
Access Requirements	**Component**
PA-4 / AA-1	Sensor Unit & Base Station — Physical Int.
Execution Phase	**Tooling**
Phase 2	UART Adapter, SWD Adapter, Multimeter, Logic Analyser, Serial Monitor
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case evaluates whether the physical interfaces allow unauthorized interaction. This could include access to internal data, manipulation, or code execution. Physical ports are identified and interactively tested.	
Objectives	
▪ Identify exposed physical interfaces (UART, SWD, …). ▪ Assess if interaction is possible without authorisation. ▪ Assess if access enables reading data or device manipulation.	
Execution Steps	
1. Visual inspection of the PCB to identify exposed connectors or pins. 2. Identify protocols of connectors via traces or continuity testing. 3. Connect corresponding adapter and monitor output via any serial monitor. 4. Attempt interaction by sending commands and assess response.	
Security-relevant Findings	
Physical interfaces are present on both components. No interactive interface was accessible and no sensitive information was disclosed. UART interface logs boot information and SWD interfaces are locked by MCU.	
Expected Secure Behaviour	
Physical interfaces should be protected through appropriate access control mechanisms. Interactive debugging interfaces should be disabled or locked on shipped devices. Any accessible output should be limited to non-sensitive information.	
Remediation Suggestions	
▪ Disable or restrict physical interfaces for production devices. ▪ Remove unnecessary pin headers to prevent non-destructive probing.	

ISTG Test Case Identifier	**Name**
ISTG-PHY-INFO-003	Disclosure of User Data

Access Requirements	**Component**
PA-4 / AA-1	Sensor Unit & Base Station — Physical Int.
Execution Phase	**Tooling**
Phase 2	UART Adapter, SWD Adapter, Multimeter, Logic Analyser, Serial Monitor
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case evaluates if physical interfaces disclose user data at runtime.	
Objectives	
▪ Determine if user data can be accessed without authorization.	
Execution Steps	
1. Visual inspection of the PCB to identify exposed connectors or pins. 2. Identify protocols of connectors via traces or continuity testing. 3. Connect corresponding adapter or logic analyser and monitor output via serial monitor. 4. Monitor for meaningful disclosed data.	
Security-relevant Findings	
Physical interfaces did not disclose any user data.	
Expected Secure Behaviour	
User data should not be accessible through physical interfaces.	
Remediation Suggestions	
▪ Collected or processed user data should only be accessible by authorized users or processes.	

ISTG Test Case Identifier	**Name**
ISTG-PHY-CONF-001	Usage of Outdated Software (Physical)
Access Requirements	**Component**
PA-4 / AA-1	Sensor Unit & Base Station — Physical Int.
Execution Phase	**Tooling**
Phase 2	UART Adapter, SWD Adapter, Multimeter, Logic Analyser, Serial Monitor
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case identifies outdated software components through physical interaction with the device.	

Objectives
▪ Identify software versions through physical interfaces. ▪ Determine if disclosed data contains software information.
Execution Steps
1. Visual inspection of the PCB to identify exposed connectors or pins. 2. Identify protocols of connectors via traces or continuity testing. 3. Connect corresponding adapter or logic analyser and monitor output via serial monitor. 4. Monitor for software version information.
Security-relevant Findings
No indicators of outdated software were discovered through physical interfaces.
Expected Secure Behaviour
Production devices should not output any detailed software version information through physical interfaces. Accessible output should be limited to necessary information.
Remediation Suggestions
▪ Ensure that physical interface output does not disclose any software version information.

ISTG Test Case Identifier	**Name**
ISTG-PHY-CONF-002	Presence of Unnecessary Functionalities
Access Requirements	**Component**
PA-4 / AA-1	Sensor Unit & Base Station — Physical Int.
Execution Phase	**Tooling**
Phase 2	UART Adapter, SWD Adapter, Multimeter, Logic Analyser, Serial Monitor
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case determines if unnecessary functionalities are present and accessible via physical functionalities, that are not required and could increase the attack surface.	
Objectives	
▪ Identify physical interfaces or components that are not required. ▪ Determine if such functionalities could be exploited. ▪ Assess if the physical attack surface is appropriately minimized.	
Execution Steps	
1. Visual inspection of the PCB to identify exposed connectors or pins. 2. Identify protocols of connectors via traces or continuity testing. 3. Connect corresponding adapter or logic analyser and monitor output via serial monitor. 4. Monitor for unused features and possible interaction.	

Security-relevant Findings
No unnecessary functionalities were discovered due to limited access to the interfaces. All assessed physical features appeared to be consistent with the intended requirements of the device.
Expected Secure Behaviour
Production devices should have minimized physical functionality exposure. Any necessary interfaces should be appropriately restricted.
Remediation Suggestions
▪ Remove unnecessary connectors or headers for production devices. ▪ Protect remaining interfaces with access controls.

ISTG Test Case Identifier	**Name**
ISTG-INT-AUTHZ-001	Unauthorized Interaction with Internal Interfaces
Access Requirements	**Component**
PA-4 / AA-1	Sensor Unit & Base Station — Internal Int.
Execution Phase	**Tooling**
Phase 2 & 8	UART Adapter, SWD Adapter, Multimeter, Logic Analyser, Serial Monitor
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case evaluates if internal interfaces allow for unauthorized interaction providing access to internal functionalities or sensitive information.	
Objectives	
▪ Assess if internal interfaces can be interacted with without authentication. ▪ Assess if authorisation methods can be bypassed.	
Execution Steps	
1. Identify internal interfaces by inspecting traces or measuring continuity. 2. Assess possible connections and interactions with the corresponding adapters. 3. If access control is present, identify possible bypasses.	
Security-relevant Findings	
No interaction with internal interfaces was possible, as connections were integrated into the PCB directly and no non-destructive connection methods were identified.	
Expected Secure Behaviour	
Only necessary internal interfaces should be enabled and protected by access control mechanisms. Non authorised interaction should not be possible.	
Remediation Suggestions	

▪ Ensure the use of lock mechanisms for internal interfaces. ▪ Ensure that communication channels between PCB components are not easily accessible.

ISTG Test Case Identifier	**Name**
ISTG-INT-INFO-001	Disclosure of Implementation Details
Access Requirements	**Component**
PA-4 / AA-1	Base Station — Internal Int.
Execution Phase	**Tooling**
Phase 2 & 8	UART Adapter, Serial Monitor
Status	**Evidence ID**
Executed	BHM-2025-E01
Summary of Test Case	
This test case assesses if internal interfaces disclose any information specific to implementation details. This information could be used by an attacker to further understand the device's architecture or configuration.	
Objectives	
▪ Identify if internal interfaces expose implementation details at any point. ▪ Evaluate if exposed information can be weaponized.	
Execution Steps	
1. Connect to identified UART pins with the UART adapter. 2. Start monitoring with the serial monitor and power up the device to capture boot messages. 3. Analyse discovered output from boot and runtime for any useful information.	
Security-relevant Findings	
The UART interface of the LTE module on the base station disclosed minimal implementation details, such as the UART baud rate used for internal communication and references to "FOTA" (firmware-over-the-air) mechanisms. No sensitive information was exposed.	
Expected Secure Behaviour	
Internal debugging output should be limited in production devices and should not expose any configuration data.	
Remediation Suggestions	
▪ Reduce information output of exposed UART interface for production devices. ▪ If possible, disable debug UART interface completely when shipping the device.	

ISTG Test Case Identifier	**Name**
ISTG-FW-INFO-001	Disclosure of Binaries or Source Code

Access Requirements	**Component**
PA-1-4 / AA-1-4	Firmware – Static Image
Execution Phase	**Tooling**
Phase 4	Binwalk, strings, file, hex editor
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case assesses if the attacker can gain access to firmware binaries or source code.	
Objectives	
▪ Identify if firmware image includes binaries. ▪ Evaluate if firmware includes source code or debugging symbols.	
Execution Steps	
1. Obtain the firmware image. 2. Use binwalk to identify file structures and artefacts. 3. Use strings to extract any meaningful strings in the firmware image. 4. Analyse outputs of binwalk and strings for source code or extractable binaries.	
Security-relevant Findings	
No source code or binaries were found in the firmware image. All extractable data consisted of unusable compiled binaries and no source-level artefacts.	
Expected Secure Behaviour	
Production firmware on the device should not contain any source code or other plain artefacts. Compiled binaries should be packaged and stripped to protect against reverse-engineering.	
Remediation Suggestions	
▪ Continue the current practice of firmware distribution.	

ISTG Test Case Identifier	**Name**
ISTG-FW-INFO-002	Disclosure of Implementation Details
Access Requirements	**Component**
PA-1-4 / AA-1-4	Firmware – Static Image
Execution Phase	**Tooling**
Phase 4	Binwalk, strings, file, hex editor
Status	**Evidence ID**
Executed	BHM-2025-E02, BHM-2025-E03
Summary of Test Case	
This test case evaluates if the firmware discloses implementation details, that could be weaponised by an attacker.	

Objectives
▪ Identify implementation details inside of the firmware image. ▪ Assess the risk of the exposed details.
Execution Steps
1. Perform static analysis of the firmware with binwalk and strings. 2. Inspect outputs for meaningful information. 3. Identify implementation details or references to modules or mechanisms.
Security-relevant Findings
The firmware image included implementation details such as module names, service identifiers and references to communication and update mechanisms. No sensitive information such as credentials were disclosed.
Expected Secure Behaviour
Firmware images should have limited exposure of implementation details. Any identifiers should be abstracted or obfuscated to reduce the value to an attacker.
Remediation Suggestions
▪ Reduce exposure of internal identifiers where feasible. ▪ Obfuscate necessary information in the firmware image.

ISTG Test Case Identifier	**Name**
ISTG-FW-INFO-003	Disclosure of Ecosystem Details
Access Requirements	**Component**
PA-1-4 / AA-1-4	Firmware – Static Image
Execution Phase	**Tooling**
Phase 4	Binwalk, strings, hex editor
Status	**Evidence ID**
Executed	BHM-2025-E02
Summary of Test Case	
This test case evaluates if the firmware image discloses details about the device's ecosystem, such as backend services, API endpoints, or dependencies.	
Objectives	
▪ Identify references to backend services, API endpoints or other third-party dependencies. ▪ Assess the value of discovered references and if they can be weaponised.	
Execution Steps	
1. Extract strings or other information from firmware. 2. Identify references to API endpoints or other backend services. 3. Assess the value of discovered information.	

Security-relevant Findings
The firmware image included strings representing API endpoints and communication services such as MQTT references. No credentials or other sensitive information was discovered. The discovered information allows further understanding and mapping of the ecosystem but does not enable direct access.
Expected Secure Behaviour
Firmware images should not disclose unnecessary details. Where possible, references to endpoints or services should be abstracted or obfuscated to reduce the value to an attacker.
Remediation Suggestions
▪ Minimize plaintext / hardcoded identifiers and references where possible. ▪ Avoid embedding descriptive strings in the firmware (obfuscation).

ISTG Test Case Identifier	**Name**
ISTG-FW-SCRT-001	Discovery of Secrets in Public Storage
Access Requirements	**Component**
PA-1-4 / AA-1-4	Firmware – Static Image
Execution Phase	**Tooling**
Phase 4	Binwalk, strings, hex editor
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case evaluates if firmware contains hardcoded credentials or secrets stored in publicly accessible locations.	
Objectives	
▪ Identify credentials, tokens, or keys embedded in the files of the image. ▪ Assess if exposed secrets could be weaponised for unauthorized access.	
Execution Steps	
1. Extract all possible information from the firmware image. 2. Analyse extracted strings or binaries for embedded secrets. 3. Assess if discovered secrets could be used maliciously.	
Security-relevant Findings	
The firmware image did not include any credentials or keys. Extracted data did not include sensitive credentials in plaintext or identifiable encryption.	
Expected Secure Behaviour	
Firmware images should not include plaintext secrets in files which could be extracted through static analysis of the image. Necessary credentials should be provisioned through protected channels during runtime.	
Remediation Suggestions	

▪ Continue the communication of secrets via secure channels during runtime. ▪ Ensure periodical checks for unintended artefacts in the firmware.

ISTG Test Case Identifier	**Name**
ISTG-FW-SCRT-003	Usage of Hardcoded Credentials
Access Requirements	**Component**
PA-1-4 / AA-1-4	Firmware – Static Image
Execution Phase	**Tooling**
Phase 4	Binwalk, strings, hex editor, Ghidra
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case evaluates if the firmware source code includes plaintext credentials, that could be extracted and used for unauthorized access.	
Objectives	
▪ Identify hardcoded usernames, passwords, or other credentials in the firmware. ▪ Evaluate if discovered credentials can be used maliciously.	
Execution Steps	
1. Extract strings or decompile firmware image with Ghidra. 2. Analyse if the source code includes hardcoded credentials. 3. Assess if identified credentials are valid and can be used to gain unauthorized access.	
Security-relevant Findings	
No hardcoded credentials were discovered during static analysis and decompilation.	
Expected Secure Behaviour	
Firmware images should not include plaintext secrets in the source code which could be extracted through static analysis of the image. Necessary credentials should be provisioned through protected channels during runtime.	
Remediation Suggestions	
▪ Continue the communication of secrets via secure channels during runtime. ▪ Ensure periodical checks for unintended artefacts in the firmware.	

ISTG Test Case Identifier	**Name**
ISTG-FW[UPDT]-AUTHZ-001	Unauthorized Update of Firmware
Access Requirements	**Component**
PA-1-4 / AA-1-3	Base Station - Firmware Update Mechanism

Execution Phase	**Tooling**
Phase 4	Burp Suite, Wireshark, UART adapter, Serial Monitor
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case assesses if the firmware update mechanism can be executed or manipulated without required authorization.	
Objectives	
▪ Identify how firmware updates are conducted and communicated. ▪ Assess if unauthorized triggering of the update mechanism is possible.	
Execution Steps	
1. Analyse the firmware for references to update mechanisms. 2. Analyse network traffic for references to update endpoints or mechanisms. 3. Evaluate if this information can be used to trigger an unauthorized firmware update.	
Security-relevant Findings	
It was not possible to identify a firmware update mechanism through firmware analysis. Network traffic between the base station and the cloud could not be analysed due to legal regulation according LTE traffic.	
Expected Secure Behaviour	
Firmware updates should only be triggered through authorized and secure channels. Any attempts to update the firmware outside of the controlled mechanisms should be rejected by the device.	
Remediation Suggestions	
▪ Ensure that update triggers cannot be started by an attacker. ▪ Continue the implementation of strict and secure update mechanisms.	

ISTG Test Case Identifier	**Name**
ISTG-FW[UPDT]-CRYPT-001	Firmware Update Signature Insufficient
Access Requirements	**Component**
PA-1-4 / AA-1-4	Firmware – Static Image
Execution Phase	**Tooling**
Phase 4	Binwalk, strings, hex editor
Status	**Evidence ID**
Executed	BHM-2025-E04
Summary of Test Case	
This test case evaluates if the firmware mechanism validates the authenticity of the update package.	

Objectives
▪ Assess if firmware updates are cryptographically signed. ▪ Evaluate if this signature gets verified before the update gets applied.
Execution Steps
1. Analyse the firmware for references to cryptographic verification mechanisms. 2. Identify signatures packaged with the firmware update. 3. Monitor if verification mechanism is correctly executed or if it can be bypassed.
Security-relevant Findings
Static analysis showed references to signature verification mechanisms. Binwalk also showed certificates packaged within the firmware, indicating proper signature procedures. Possible bypasses could not be analysed due to limitations of permitted scope.
Expected Secure Behaviour
Firmware updates should be signed by appropriate cryptographic means and verified before any changes are executed. Invalid firmware signatures should be rejected completely.
Remediation Suggestions
▪ Ensure signature verification cannot be bypassed. ▪ Continue with strong cryptographic signature and validation processes.

ISTG Test Case Identifier	**Name**
ISTG-FW[UPDT]-CRYPT-003	Insecure Transmission of Firmware Updates
Access Requirements	**Component**
PA-1-4 / AA-1-4	Base Station - Firmware Update Mechanism
Execution Phase	**Tooling**
Phase 4	Wireshark, Burp Suite
Status	**Evidence ID**
Executed	BHM-2025-E05
Summary of Test Case	
This test case assesses if firmware updates are only transmitted through secure channels.	
Objectives	
▪ Identify the transport channel for firmware updates. ▪ Assess if the discovered channel uses appropriate encryption.	
Execution Steps	
1. Analyse firmware for references to update transmission protocols or endpoints. 2. Assess observable network traffic. 3. Evaluate if observed traffic is encrypted properly or can be intercepted.	
Security-relevant Findings	

No firmware update traffic could be observed on permitted protocols. This indicates the firmware updates are transmitted over LTE, which is deemed secure, in combination with discovered references for encryption of the LTE channel.
Expected Secure Behaviour
Firmware updates should only be transmitted through encrypted and authenticated channels. This is done to prevent interception or tampering during transmission.
Remediation Suggestions
▪ Maintain current transport mechanisms. ▪ Ensure that no insecure fallback mechanisms are implemented that could be exploited.

ISTG Test Case Identifier	**Name**
ISTG-FW[UPDT]-LOGIC-001	Rollback Protection Insufficient
Access Requirements	**Component**
PA-1-4 / AA-1-4	Base Station - Firmware Update Mechanism
Execution Phase	**Tooling**
Phase 4	Strings
Status	**Evidence ID**
Executed	BHM-2025-E06
Summary of Test Case	
This test case assesses if appropriate firmware update rollback mechanisms are in place. These ensure that the device's functionality does not get impacted by a failed firmware update.	
Objectives	
▪ Identify firmware update logic. ▪ Assess if rollback mechanisms are working correctly by triggering a failed update.	
Execution Steps	
1. Analyse firmware for update mechanism rollback references. 2. If possible, analyse behaviour after power outage during update process.	
Security-relevant Findings	
Firmware strings indicate that rollback mechanisms and error handling are in place and lead to a reset of the device. No active attacks could be attempted due to legal limitations.	
Expected Secure Behaviour	
Appropriate rollback mechanisms should be in place, that reject and prevent failed installation processes.	
Remediation Suggestions	
▪ Ensure that rollback mechanisms are properly enforced.	

ISTG Test Case Identifier	**Name**
ISTG-PROC-AUTHZ-001	Unauthorized Access to the Processing Unit
Access Requirements	**Component**
PA-4 / AA-1	Sensor Unit & Base Station – MCU
Execution Phase	**Tooling**
Phase 5 & 8	SWD adapter, UART adapter, serial monitor, STM32CubeProgrammer
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case is used to evaluate if unauthorized access to the processing unit is possible. This could include debugging, memory access or execution control.	
Objectives	
▪ Identify exposed or accessible MCU interfaces. ▪ Assess access control of identified interfaces.	
Execution Steps	
1. Identify MCU through visual inspection. 2. Locate potential interfaces by analysing traces and referencing corresponding data sheets. 3. Connect the corresponding adapter and attempt interactive connections.	
Security-relevant Findings	
SWD debugging interfaces were discovered on both, the base station and the sensor unit. These could not be interacted with due to access control implemented on chip level.	
Expected Secure Behaviour	
Production devices should enforce strict access control on any interactive interfaces. Unauthorized interaction should not lead to readable memory or execution control.	
Remediation Suggestions	
▪ Periodically verify access control to prevent accidental unlocking through firmware settings.	

ISTG Test Case Identifier	**Name**
ISTG-PROC-LOGIC-001	Insecure Instruction Handling
Access Requirements	**Component**
PA-4 / AA-1-4	Sensor unit & base station – MCU
Execution Phase	**Tooling**
Phase 5 & 8	SWD adapter, STM32CubeProgrammer
Status	**Evidence ID**
Executed	-

Summary of Test Case
This test case evaluates if the MCU's instruction handling can be influenced by an attacker. This could lead to unintended behaviour or malfunctions.
Objectives
▪ Assess if instructions can be misused. ▪ Identify error handling.
Execution Steps
1. Observe MCU behaviour through possible channels. 2. Analyse firmware image for references to error handling. 3. Use accessible channels to influence instructions. 4. Evaluate if MCU can be influenced directly.
Security-relevant Findings
No insecure instruction handling was observed. Due to locked SWD interfaces, it was not possible to directly influence MCU's behaviour.
Expected Secure Behaviour
The MCU should have robust error handling and execution control, preventing direct influence of instructions from interactive channels.
Remediation Suggestions
▪ Ensure continuous error handling and input validation. ▪ Continue with locking interactive debugging interfaces.

ISTG Test Case Identifier	**Name**
ISTG-PROC-SIDEC-001	Insufficient Side-Channel Protection
Access Requirements	**Component**
PA-4 / AA-1-4	Sensor unit & base station – MCU
Execution Phase	**Tooling**
Phase 5 & 8	SWD adapter, STM32CubeProgrammer, oscilloscope
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test evaluates if the MCU shows observable characteristics for side-channel attacks, that could possibly leak sensitive information. This could include timing patterns or power variations.	
Objectives	
▪ Identify observable characteristics of MCU. ▪ Assess potential correlations between behaviour and characteristics.	
Execution Steps	

1. Observe boot and runtime power consumption and behaviour. 2. Correlate characteristics with information found in firmware. 3. Evaluate if patterns exist and could be interpreted.
Security-relevant Findings
Both MCUs did not present any characteristics for side-channel attacks. Observed behaviour and patterns was consistent and could not be correlated to specific operations.
Expected Secure Behaviour
MCUs should not produce observable characteristics while processing specific operations, such as security-relevant instructions.
Remediation Suggestions
▪ Ensure that specific side-channel protection techniques are implemented. ▪ Maintain implementation practices limiting observable characteristics.

ISTG Test Case Identifier	**Name**
ISTG-DES-AUTHZ-001	Unauthorized Interaction with Data Exchange Services
Access Requirements	**Component**
PA-1-4 / AA-1	Data Exchange Services – Cloud Services
Execution Phase	**Tooling**
Phase 6 & 7	Android Studio, Burp Suite Proxy
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case evaluates if the Data Exchange Services enforce proper access control.	
Objectives	
▪ Identify access control mechanisms of DES. ▪ Assess if there are ways to bypass them.	
Execution Steps	
1. Identify DES endpoints and channels of the ecosystem by monitoring network traffic. 2. Attempt to interact with discovered endpoints without authentication. 3. Assess if access control is implemented correctly and if it could be bypassed.	
Security-relevant Findings	
Authorization control was implemented properly and no unauthenticated or unauthorized interaction was possible. No bypass for these mechanisms was discovered.	
Expected Secure Behaviour	
Communication channels and endpoints should only be accessible with correct authentication and authorization, verified by the recipient.	

Remediation Suggestions
▪ Maintain current access control mechanisms. ▪ Ensure that newly added functionalities also implement these mechanisms.

ISTG Test Case Identifier	**Name**
ISTG-DES-AUTHZ-002	Privilege Escalation in DES
Access Requirements	**Component**
PA-1-4 / AA-2-3	Data Exchange Services – Cloud Services
Execution Phase	**Tooling**
Phase 6	Android Studio, Burp Suite Proxy
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case assesses if the DES implement authorization separation between privilege levels.	
Objectives	
▪ Identify possible privilege levels. ▪ Assess if privileged interaction is restricted to privileged roles. ▪ Evaluate if it is possible to manipulate the separation mechanisms to execute higher-privileged functionalities.	
Execution Steps	
1. Identify DES endpoints and channels and observable privilege levels. 2. Attempt to interact with them in a lower-privileged context. 3. Evaluate if there are ways to gain a higher privilege context by manipulating requests or communication.	
Security-relevant Findings	
No privilege escalation vulnerabilities were discovered, due to the lack of identified authorization levels. Only one authenticated context was observed; unauthorized interaction was assessed in ISTG-DES-AUTHZ-001.	
Expected Secure Behaviour	
Data exchange services should implement strict separation between authorization roles. It should not be possible to perform operations meant for higher-privileged contexts.	
Remediation Suggestions	
▪ Ensure that authorization separation is implemented in all channels. ▪ Avoid a “one-role-only” permission management.	

ISTG Test Case Identifier	**Name**

ISTG-DES-INFO-001	Disclosure of Implementation Details
Access Requirements	**Component**
PA-1 / AA-1-4	Data Exchange Services – Cloud Services
Execution Phase	**Tooling**
Phase 6 & 7	Android Studio, Burp Suite Proxy
Status	**Evidence ID**
Executed	BHM-2025-E07 TODO (auth mechanisms)
Summary of Test Case	
This test case evaluates if data exchange services disclose information that can aid a malicious actor in discovering used technologies or internal service architecture.	
Objectives	
▪ Identify if implementation details are disclosed in DES communication. ▪ Assess observed information for security-relevant details.	
Execution Steps	
1. Intercept network traffic of DES. 2. Inspect traffic for implementation details. 3. Identify and correlate discovered information to services or structures. 4. Assess if this information increases the attack surface of the ecosystem.	
Security-relevant Findings	
Some implementation specific details were discovered in DES network traffic, which could be used for understanding the ecosystem and planning further attacks.	
Expected Secure Behaviour	
DES communication should only disclose technically necessary information. Internal metadata should not be revealed during normal communication.	
Remediation Suggestions	
▪ Evaluate which technical information is required for functionality and minimize exposed details.	

ISTG Test Case Identifier	**Name**
ISTG-DES-INFO-002	Disclosure of Ecosystem Information
Access Requirements	**Component**
PA-1 / AA-1-4	Data Exchange Services – Cloud Services
Execution Phase	**Tooling**
Phase 6 & 7	Android Studio, Burp Suite Proxy
Status	**Evidence ID**
Executed	-
Summary of Test Case	

This test case evaluates if data exchange services expose information about the ecosystem itself. This might include URLs, IP addresses, or other infrastructure components.
Objectives
▪ Identify exposed information about endpoints or other components. ▪ Assess if this exposure is technically necessary and if it could be weaponized.
Execution Steps
1. Analyse network traffic of DES for endpoints, hostnames, etc. 2. Correlate discovered information to known components and functionalities. 3. Assess if newly discovered details could be used maliciously.
Security-relevant Findings
No indicators of ecosystem detail disclosure was discovered within the assessment scope.
Expected Secure Behaviour
Communication of DES should have limited exposure of ecosystem details. Only functionally required information should be disclosed.
Remediation Suggestions
▪ Abstract or obfuscate infrastructure-related information. ▪ Review DES communication for unnecessarily exposed metadata.

ISTG Test Case Identifier	**Name**
ISTG-DES-INFO-003	Disclosure of User Data
Access Requirements	**Component**
PA-1 / AA-1-4	Data Exchange Services – Cloud Services
Execution Phase	**Tooling**
Phase 6 & 7	Android Studio, Burp Suite Proxy
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case evaluates if the communication of the data exchange services exposes user specific data in an unauthorized context.	
Objectives	
▪ Determine if user information is exposed to unauthorized actors. ▪ Assess if separation between data of different users is separated correctly.	
Execution Steps	
1. Intercept DES communication and inspect for user-related data. 2. Attempt to access user data outside of authorized context via request manipulation. 3. Assess if exposed data should be accessible.	

Security-relevant Findings
The DES did not show any unauthorized disclosure of user-related data. This information was only accessible in an authorized context.
Expected Secure Behaviour
Authorization controls must be in place, preventing unauthorized access to user data. These access control mechanisms should be enforced on all related endpoints.
Remediation Suggestions
▪ Review access control mechanisms for user-specific data endpoints. ▪ Maintain current implementation for future development.

ISTG Test Case Identifier	**Name**
ISTG-DES-SCRT-001	Access to Sensitive Data
Access Requirements	**Component**
PA-3 / AA-2	Data Exchange Services – Cloud Services
Execution Phase	**Tooling**
Phase 6 & 7	Android Studio, Burp Suite Proxy, mosquitto
Status	**Evidence ID**
Executed	BHM-2025-E08, BHM-2025-E09
Summary of Test Case	
This test case evaluates if the cloud API exposes confidential data, such as credentials, through authenticated endpoints accessible by the mobile application. The scope includes verifying whether unintended sensitive information is retrievable by clients.	
Objectives	
▪ Determine if API endpoints return credentials. ▪ Assess if MQTT credentials are exposed. ▪ Identify if the API enforces separation between user and device privileges.	
Execution Steps	
1. Start Burp Suite and Android Studio Virtual Device set up with Burp Proxy 2. Launch the BHMs mobile application on the emulator 3. Monitor HTTP(S) traffic during legitimate usage of the application 4. Identify a request to an endpoint which returns MQTT credentials for base station 5. Extract the MQTT username, password, and client_id 6. Verify credentials using: `mosquitto_pub -d -q 1 -h <mqtt_broker> -p 8883 -t v1/devices/me/telemetry -i "<client_id>" -u "<username>" -P "<password>" -m "{temperature:25}"`	
Security-relevant Findings	

The API exposed static device specific MQTT credentials for the base station to any authenticated request. The credentials provided full publish/subscribe permissions and could allow a user to impersonate the base station and possibly influence the cloud's behaviour. This results in a significant information disclosure vulnerability.
Expected Secure Behaviour
Device-level secrets should never be accessible by client applications. API endpoints must enforce strict privilege separation and authentication.
Remediation Suggestions
▪ Replace static credentials with short-lived tokens bound to device identity. ▪ Prevent user-facing clients from querying device-level authentication data. ▪ Implement role-based access control on API endpoints.

ISTG Test Case Identifier	**Name**
ISTG-DES-CRYPT-001	Insecure Cryptographic Algorithms
Access Requirements	**Component**
PA-1 / AA-1-4	Data Exchange Services – Cloud Services
Execution Phase	**Tooling**
Phase 6 & 7	Android Studio, Burp Suite Proxy
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case evaluates if data exchange services are using weak or insecure cryptographic algorithms during communication. This could lead to an attacker cracking encryption.	
Objectives	
▪ Identify implemented cryptographic algorithms. ▪ Assess if they are still secure and modern.	
Execution Steps	
1. Intercept network traffic to analyse TLS parameter, protocol versions and possible implemented ciphers. 2. Assess if the discovered configurations are still considered secure or deprecated.	
Security-relevant Findings	
No weak or deprecated encryption mechanisms were discovered in the DES. Identified configurations were consistent with secure and modern best practices.	
Expected Secure Behaviour	
All communication channels of the DES should employ strong and modern cryptographic mechanisms to ensure confidentiality and integrity. Parameter and configurations should be chosen according to current best practices.	
Remediation Suggestions	

▪ Maintain current configurations and encryption methods. ▪ Regularly review best practices for necessary changes.

ISTG Test Case Identifier	**Name**
ISTG-DES-LOGIC-001	Circumvention of the Intended Business Logic
Access Requirements	**Component**
PA-1-4 / AA-1-4	Data Exchange Services – Cloud Services
Execution Phase	**Tooling**
Phase 6	Android Studio, Burp Suite Proxy
Status	**Evidence ID**
Executed	BHM-2025-E10
Summary of Test Case	
This test case assesses if the DES validates and enforces constraints of the intended business logic. It evaluates if authenticated requests can be used in unintentional ways.	
Objectives	
▪ Identify implemented business logic constraints. ▪ Determine if these are validated and if they could be misused.	
Execution Steps	
1. Analyse normal workflows as a baseline. 2. Identify possible variables that could be altered and misused. 3. Observe system behaviour when replaying altered requests.	
Security-relevant Findings	
It was found that DES accepted altered requests. Logic rules were not consistently enforced. It was possible to interact with the DES in unintended ways, but no exploitation was discovered in the permitted scope.	
Expected Secure Behaviour	
Data exchange services should validate not only authorization and authentication but also operational contexts. Unexpected communication should be rejected.	
Remediation Suggestions	
▪ Ensure that validations of request contexts are implemented. ▪ Review intended business logic and infer expected workflows and operational states. ▪ Adjust error handling and validation to prevent misuse.	

ISTG Test Case Identifier	**Name**
ISTG-DES-INPV-001	Insufficient Input Validation
Access Requirements	**Component**

PA-1-4 / AA-1-4	Data Exchange Services – Cloud Services
Execution Phase	**Tooling**
Phase 6 & 7	Android Studio, Burp Suite Proxy
Status	**Evidence ID**
Executed	BHM-2025-E11
Summary of Test Case	
This test case evaluates if the DES are validating received input correctly before processing it.	
Objectives	
▪ Assess if DES enforce validation of expected formats and ranges. ▪ Evaluate if unexpected input or malformed requests are rejected. ▪ Determine if system behaviour can be impacted due to insufficient validation.	
Execution Steps	
1. Identify input parameters in captured network traffic. 2. Modify and replay requests with unexpected or malformed values. 3. Observe system behaviour for missing validation and analyse impact.	
Security-relevant Findings	
It was found that some DES endpoints did not validate malformed input correctly. Unexpected values were accepted and most likely processed, though it could not be further tested and exploited in the given scope.	
Expected Secure Behaviour	
All DES endpoints should enforce validations of expected input values. Failed validation should lead to rejected requests and error handling.	
Remediation Suggestions	
▪ Review current input validation mechanisms of all DES endpoints. ▪ Ensure correct enforcement of expected formats, ranges, and types. ▪ Unexpected inputs should be consistently rejected and non-technical errors returned.	

ISTG Test Case Identifier	**Name**
ISTG-DES-INPV-002	Command or Code Injection
Access Requirements	**Component**
PA-1-4 / AA-1-4	Data Exchange Services – Cloud Services
Execution Phase	**Tooling**
Phase 6	Android Studio, Burp Suite Proxy
Status	**Evidence ID**
Executed	-
Summary of Test Case	

This test case evaluates if assessed DES are vulnerable to injection attacks. These would allow an attacker to inject code or commands due to insecure input handling.
Objectives
▪ Assess if injected code or commands are executed or processed. ▪ Determine if input is properly sanitized and handled.
Execution Steps
1. Identify possible attack vectors in DES endpoints through network analysis. 2. Inject command or code payloads into discovered parameters. 3. Observe system behaviour, responses and error handling. 4. Evaluate if injected payloads are executed, processed, or rejected.
Security-relevant Findings
No injection vulnerabilities were discovered in the DES within the permitted scope. Attempted payloads were consistently rejected, and no evidence of execution or processing was identified.
Expected Secure Behaviour
DES processes should validate and sanitize all user input before processing and should never be interpreted as commands or code.
Remediation Suggestions
▪ Maintain current input handling and sanitization. ▪ Ensure “Zero-Trust” principle for all user input in future development.

ISTG Test Case Identifier	**Name**
ISTG-WRLS-AUTHZ-001	Unauthorized Interaction
Access Requirements	**Component**
PA-2-4 / AA-1-4	Sensor unit & base station – BLE interface
Execution Phase	**Tooling**
Phase 6	nRF52840 Dongle, Wireshark
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case evaluates wireless interfaces allow unauthorized interaction. It assesses if an unauthorized actor can perform operations that are intended for authorized contexts.	
Objectives	
▪ Assess if wireless interfaces are accessible without authorization. ▪ Determine if access control of wireless interfaces is enforced.	
Execution Steps	

1. Identify wireless interfaces by visual inspection or general scanning of common radio protocols. 2. Attempt interaction with discovered interfaces without any pairing process or authentication. 3. Assess if operations are permitted and evaluate their impact.
Security-relevant Findings
Identified wireless interfaces (BLE) were correctly enforcing pairing states. It was not possible to force a pairing process or directly interact with the devices.
Expected Secure Behaviour
All wireless interfaces should implement authorization and authentication constraints. Direct interaction or other operations should not be possible and be rejected.
Remediation Suggestions
▪ Maintain current implementation of pairing constrictions. ▪ Review pairing process to ensure protection against impersonation.

ISTG Test Case Identifier	**Name**
ISTG-WRLS-INFO-001	Disclosure of Implementation Details
Access Requirements	**Component**
PA-2-4 / AA-1-4	Sensor unit & base station – BLE interface
Execution Phase	**Tooling**
Phase 6	nRF52840 Dongle, Wireshark
Status	**Evidence ID**
Executed	BHM-2025-E12, BHM-2025-E13
Summary of Test Case	
This test case assesses if wireless communication discloses information about implementation-specific details. This information could be used by an attacker to understand internal functionalities and plan further attacks.	
Objectives	
▪ Assess if implementation details are exposed through wireless channels. ▪ Evaluate if disclosed information exceeds necessary data. ▪ Determine the potential impact of this data.	
Execution Steps	
1. Monitor wireless communication with Wireshark and the BLE adapter. 2. Analyse captured packages for implementation-specific details. 3. Assess the impact of discovered information.	
Security-relevant Findings	
Captured BLE traffic exposed minimal implementation details, which could be used for further understanding of the functionalities, although the impact was deemed very low.	

Expected Secure Behaviour
Wireless interfaces and communication should only expose necessary data.
Remediation Suggestions
▪ Review information exposed by wireless communication. ▪ Redact unnecessary information from the package flow.

ISTG Test Case Identifier	**Name**
ISTG-WRLS-INFO-002	Disclosure of Ecosystem Details
Access Requirements	**Component**
PA-2-4 / AA-1-4	Sensor unit & base station – BLE interface
Execution Phase	**Tooling**
Phase 6	nRF52840 Dongle, Wireshark
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case evaluates if the wireless interfaces disclose any information that could lead to a broader understanding of the devices' ecosystem. This could include backend services or device roles.	
Objectives	
▪ Assess if wireless communication discloses ecosystem-related information. ▪ Evaluate if discovered information exceeds functionally necessary data.	
Execution Steps	
1. Monitor wireless traffic with Wireshark and the BLE adapter. 2. Analyse captured packages for ecosystem-related information, such as backend descriptors or endpoints. 3. Evaluate the potential impact of discovered data and if it is technically required.	
Security-relevant Findings	
No ecosystem-related information was found in the captured BLE packages.	
Expected Secure Behaviour	
Wireless interfaces should not disclose any unnecessary information about the surrounding ecosystem. Only details that are required for normal operation should be transmitted.	
Remediation Suggestions	
▪ Review BLE communication for unnecessary references. ▪ Maintain restrictive information transmission.	

ISTG Test Case Identifier	Name
ISTG-WRLS-INFO-003	Disclosure of User Data
Access Requirements	**Component**
PA-2-4 / AA-1-4	Sensor unit & base station – BLE interface
Execution Phase	**Tooling**
Phase 6	nRF52840 Dongle, Wireshark
Status	**Evidence ID**
Executed	BHM-2025-E14

Summary of Test Case

This test case evaluates if wireless communication exposes user-related data outside of authorized contexts. This data could include account information, other identifiable data, or sensor values.

Objectives

- Assess if wireless channels expose data that can be correlated or interpreted as user data.
- Evaluate if disclosed data can be linked to specific users or user actions.

Execution Steps

1. Monitor wireless traffic with Wireshark and the BLE adapter.
2. Analyse captured packages, focusing on data exchange packets.
3. Evaluate if data can be interpreted meaningfully and correlates to user-related information.

Security-relevant Findings

It was discovered that the data exchange between the two devices is unencrypted and interpretation of the content was attempted but not verified. If correct, sensor measurements including body temperature and movement could be exposed without any restrictions and could be passively sniffed.

Expected Secure Behaviour

User-related data should not be exposed to unauthorized and unauthenticated actors. Appropriate measures should be in place to encrypt information and access control should limit exposure of data.

Remediation Suggestions

- Implement BLE encryption between sensor unit and base station using current best practices.
- If encryption is not possible, data should be obfuscated to obstruct interpretation.

ISTG Test Case Identifier	Name
ISTG-WRLS-CONF-002	Identified Unnecessary Software
Access Requirements	**Component**
PA-2-4 / AA-1-4	Sensor unit & base station – BLE interface

Execution Phase	**Tooling**
Phase 6	nRF52840 Dongle, Wireshark
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case assesses if the devices expose any unnecessary wireless services.	
Objectives	
▪ Enumerate all exposed wireless services. ▪ Evaluate if these services include any characteristics that are not in use. ▪ Determine the impact on the attack surface through their exposure.	
Execution Steps	
1. Use Wireshark and the BLE adapter to passively monitor wireless traffic. 2. Identify all available services and characteristics. 3. Assess their operational relevance through comparison to expected behaviour.	
Security-relevant Findings	
Both devices did not expose any unnecessary wireless services through the analysed wireless channels. All exposed data was required for correct operation and communication.	
Expected Secure Behaviour	
Wireless interfaces should only expose services that are required for intended functionality.	
Remediation Suggestions	
▪ Maintain minimized exposure of wireless services.	

ISTG Test Case Identifier	**Name**
ISTG-WRLS-SCRT-001	Access to Sensitive Data
Access Requirements	**Component**
PA-2-4 / AA-1-4	Sensor unit & base station – BLE interface
Execution Phase	**Tooling**
Phase 6	nRF52840 Dongle, Wireshark
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case evaluates if the wireless interfaces disclose sensitive data such as credentials, tokens, or cryptographic keys.	
Objectives	

▪ Identify if wireless interfaces transmit security-relevant information. ▪ Evaluate if it is possible to extract secrets from wireless traffic.
Execution Steps
1. Monitor wireless traffic with Wireshark and the BLE adapter, including the pairing process and data exchange. 2. Analyse packet data for sensitive data such as tokens or credentials. 3. Assess if pairing process is implemented insecurely, e.g. through repeated usage of the same secret.
Security-relevant Findings
The analysed wireless traffic did not show any sensitive information, credentials, tokens, or keys. No information was discovered, that could lead to a compromise of authentication or encryption.
Expected Secure Behaviour
Security-relevant data should not be exposed through wireless interfaces without proper protection mechanisms in place, such as encryption.
Remediation Suggestions
▪ Maintain current implementation of pairing process.

ISTG Test Case Identifier	**Name**
ISTG-WRLS-CRYPT-001	Insecure Cryptographic Algorithms
Access Requirements	**Component**
PA-2-4 / AA-1-4	Sensor unit & base station – BLE interface
Execution Phase	**Tooling**
Phase 6	nRF52840 Dongle, Wireshark
Status	**Evidence ID**
Executed	BHM-2025-E15
Summary of Test Case	
This test case assesses if wireless interfaces and communication use appropriate cryptographic methods to maintain the confidentiality and integrity of the transmitted data.	
Objectives	
▪ Identify implemented cryptographic mechanisms of wireless communication. ▪ Compare identified encryption methods against current cryptographic best practices.	
Execution Steps	
1. Capture wireless traffic with Wireshark and BLE adapter. 2. Analyse captured packets for indicators of encryption. 3. Identify encryption methods by comparing data structures.	
Security-relevant Findings	

During analysis it was found, that the BLE communication of the devices was not employing any encryption method. The transmitted plaintext data was captured passively, and interpretation attempts were made but could not be verified.
Expected Secure Behaviour
Wireless interfaces should implement strong and modern cryptographic mechanisms for data exchange.
Remediation Suggestions
▪ Adjust the data transmission implementation to apply modern and secure encryption methods. ▪ Review best practices for wireless data transmission. ▪ If encryption is not technically feasible, data should be obfuscated to prevent passive sniffing and interpretation.

ISTG Test Case Identifier	**Name**
ISTG-WRLS-LOGIC-001	Circumvention of intended Business Logic
Access Requirements	**Component**
PA-2-4 / AA-1-4	Sensor unit & base station – BLE interface
Execution Phase	**Tooling**
Phase 6	nRF52840 Dongle, Wireshark
Status	**Evidence ID**
Executed	-
Summary of Test Case	
This test case assesses if wireless interfaces enforce constraints protecting the intended business logic, or if misuse leads to unintended behaviour.	
Objectives	
▪ Analyse intended logic workflows. ▪ Evaluate if replayed, altered, or invalid requests lead to unintended device behaviour.	
Execution Steps	
1. Capture normal wireless traffic between the devices with Wireshark and the BLE adapter. 2. Identify attack vectors for possible value modification or indicators of intended packet order. 3. Replay altered or re-ordered packets and observe device behaviour.	
Security-relevant Findings	
The tester could not identify any flaws in the business logic of the wireless data exchange. Any attempts to send modified or altered packets got rejected by the receiving device.	
Expected Secure Behaviour	
Constraints should be implemented, that reject any unexpected or invalid interactions.	

Remediation Suggestions
▪ Review LTE transmission implementation for possible business logic flaws (out of scope). ▪ Maintain current logic handling implementation.

Appendix 4 – Evidence Log

BHM-2025-E01 – Implementation Details disclosed in UART Boot Log

BHM-2025-E02 – Implementation & Ecosystem (MQTT Endpoints) Details disclosed in Firmware Strings

```
Broker requests an acknowledgment, send it.
MQTT already connecting...
check MQTT - not connected, reconnect
QoS MQTT event %d publish msg: id=%d, notified_count=%d, type=%d, flags=%d
v1/devices/me/telemetry
MQTT try to connect.
MQTT broker for SIM 1: ip=%s, port=%u
MQTT broker for SIM 2: ip=%s, port=%u
MQTT User %s
MQTT connect error: %d
Subscribe err: %d
Subscribing to "v1/devices/me/attributes/response/+"
v1/devices/me/attributes/response/+
Subscribing to "v1/devices/me/rpc/request/+" for FOTA
v1/devices/me/rpc/request/+
Image confirmation failed. Old image will be booted after next restart.
Image confirmed successful. This image is permanent now.
```

BHM-2025-E03 – Implementation Details (MCU Type) disclosed in Firmware Strings

```
client finished
server finished
master secret
key expansion
 00000000000000
NRFX_GPIOTE
NRFX_PPI
NRFX_PWM
NRFX_RTC
NRFX_WDT
Terminal
TTR REGGES
Unknown error
>>> ZEPHYR FATAL ERROR %d: %s on CPU %d
Fault during interrupt handling
Current thread: %p (%s)
```

BHM-2025-E04 – Binwalk Output showing Embedded Signature

```
DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
382856        0x5D788         DES SP2, little endian
383112        0x5D888         DES SP1, little endian
385380        0x5E164         SHA256 hash constants, little endian
400302        0x61BAE         Certificate in DER format (x509 v3), header length: 4, sequence length: 910
404107        0x62A8B         PEM certificate
405070        0x62E4E         AES S-Box
421072        0x66CD0         AES Inverse S-Box
421328        0x66DD0         AES S-Box
```

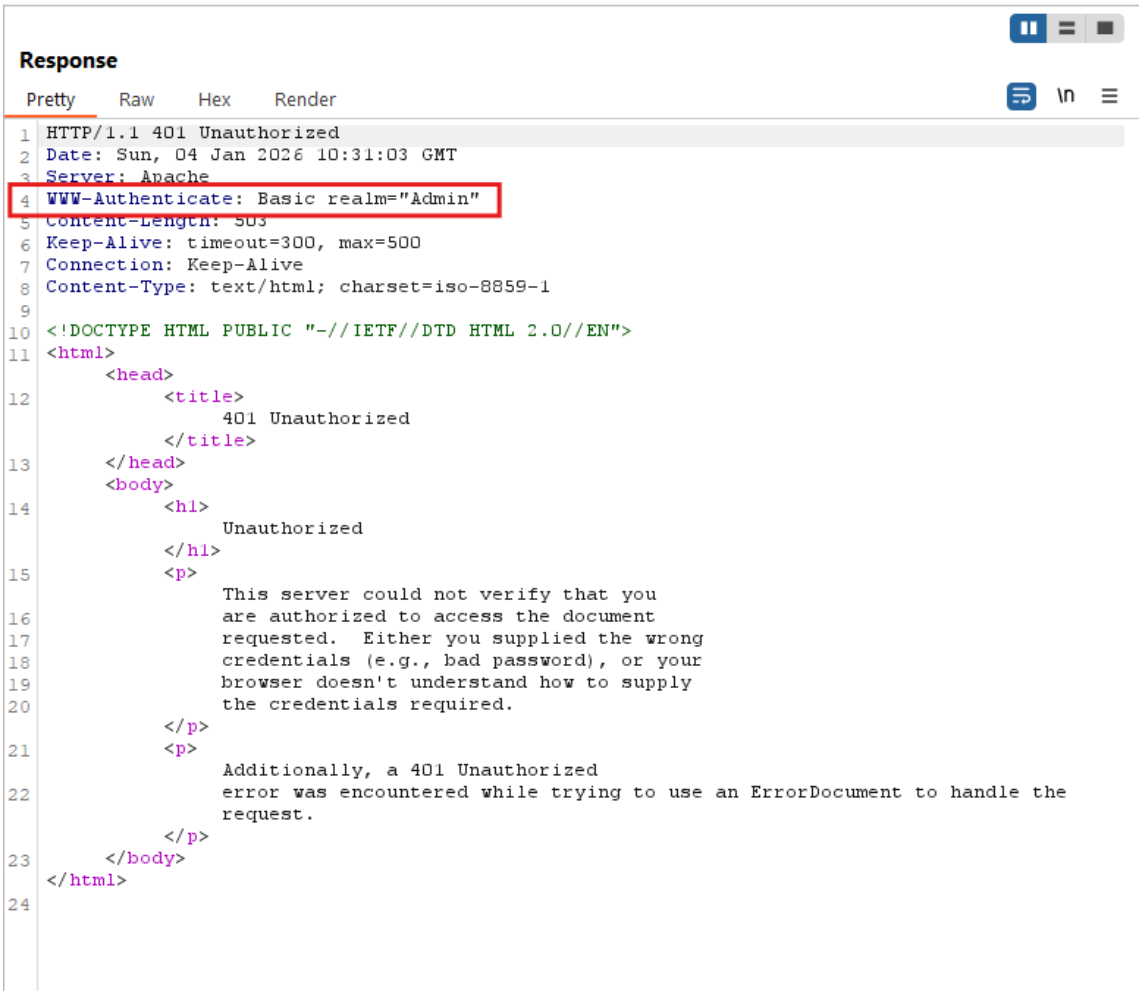
BHM-2025-E05 – Firmware Strings indication Appropriate Encryption

```
5386    -----BEGIN CERTIFICATE-----
5387    Invalid TLS context
5388    Deallocating unused TLS context
5389    Failed to allocate TLS context
5390    TLS handshake timeout
5391    TLS handshake error: -%x
5392    TLS reset error: -%x
5393    Failed to set session for %p
5394    Failed to obtain session for %p
5395    Failed to allocate session buffer.
5396    Failed to serialize session, err: 0x%x.
5397    Failed to save session for %p
```

BHM-2025-E06 – Error Handling & Rollback Mechanisms indicated by Firmware Strings

```
5238    DOWNLOADING
5239    SOTA %s %d%%, send_progress=%d
5240    write to flash err: %d
5241    FAILED_FLASH
5242    Download of %u bytes (time=%lld [ms], speed= %u B/s)
5243    cannot release download resources, err: %d
5244    UPDATING
5245    FAILED_UPDATE
5246    schedule update err: %d
5247    perform a system reset in 3s
5248    Error %d during download
5249    FAILED_ERROR
```

BHM-2025-E07 – Authentication Method disclosed in HTTP Response



Response

Pretty Raw Hex Render

```
1  HTTP/1.1 401 Unauthorized
2  Date: Sun, 04 Jan 2026 10:31:03 GMT
3  Server: Apache
4  WWW-Authenticate: Basic realm="Admin"
5  Content-Length: 503
6  Keep-Alive: timeout=300, max=500
7  Connection: Keep-Alive
8  Content-Type: text/html; charset=iso-8859-1
9
10 <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
11 <html>
       <head>
12         <title>
               401 Unauthorized
           </title>
13     </head>
       <body>
14         <h1>
               Unauthorized
           </h1>
15         <p>
               This server could not verify that you
16             are authorized to access the document
17             requested.  Either you supplied the wrong
18             credentials (e.g., bad password), or your
19             browser doesn't understand how to supply
20             the credentials required.
           </p>
21         <p>
               Additionally, a 401 Unauthorized
22             error was encountered while trying to use an ErrorDocument to handle the
               request.
           </p>
23     </body>
   </html>
24
```

BHM-2025-E08 – MQTT Credentials exposed as Plaintext in HTTP Response

Request

Pretty Raw Hex

```
1 GET /getDevCred.php?q=2.                    8 HTTP/1.1
2 User-Agent: Embarcadero Rest Client/1.0
3 Accept-Charset: utf-8, *;q=0.8
4 Accept: application/json, text/plain; q=0.9, text/html;q=0.8,
5 Authorization: Basic
6 Host: iot.sticklett.com
7 Connection: keep-alive
8 Accept-Encoding: gzip, deflate, br
9
10
```

Response

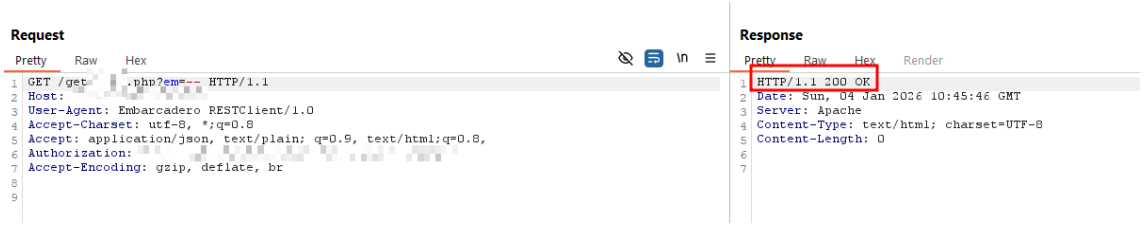
Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 Date: Thu, 04 Dec 2025 02:00:29 GMT
3 Server: Apache
4 Keep-Alive: timeout=300, max=500
5 Connection: Keep-Alive
6 Content-Type: text/html; charset=UTF-8
7 Content-Length: 107
8
9 Verbindung erfolgreich: Werte: 90      92 89(      128 86      0 -- Datei gespeichert.
10
```

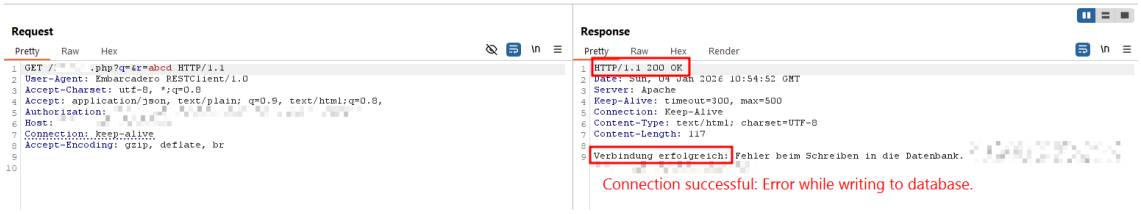
BHM-2025-E09 – Mosquitto Output verifying Valid Credentials

```
thomas@DESKTOP-93HTEE8: $ mosquitto_pub -d -q 1                                    -p 8883 -t v1/devices/m
e/telemetry -i "                " -u "                " -P "                " -m "{temperature:25}"
Client 866069065838730 sending CONNECT
Client 866069065838730 received CONNACK (0)
```

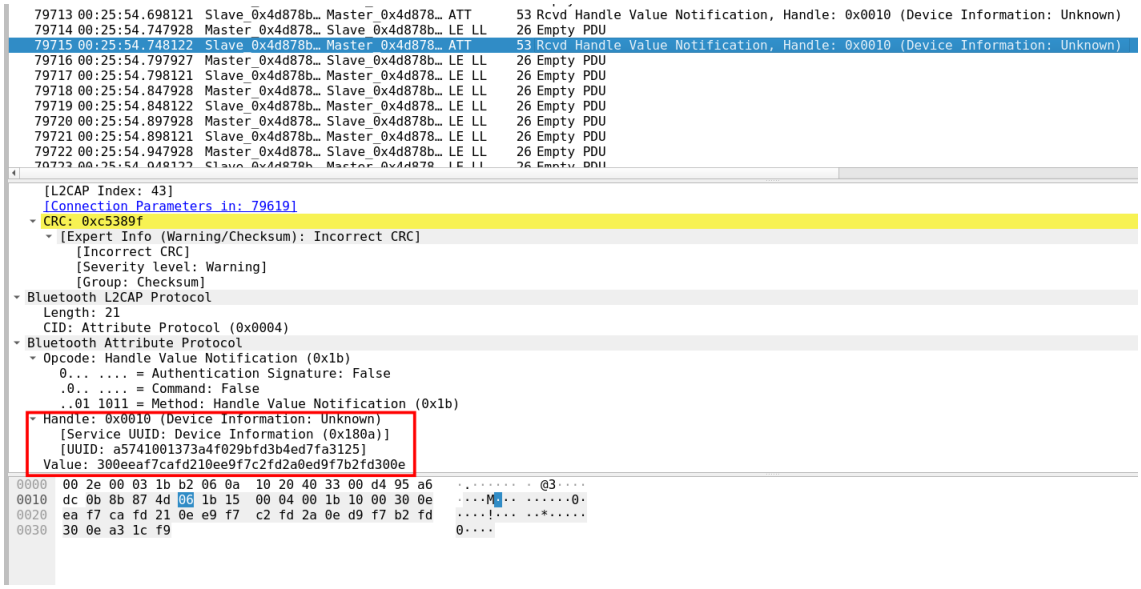
BHM-2025-E10 – Malformed Request Parameter is Processed Successfully



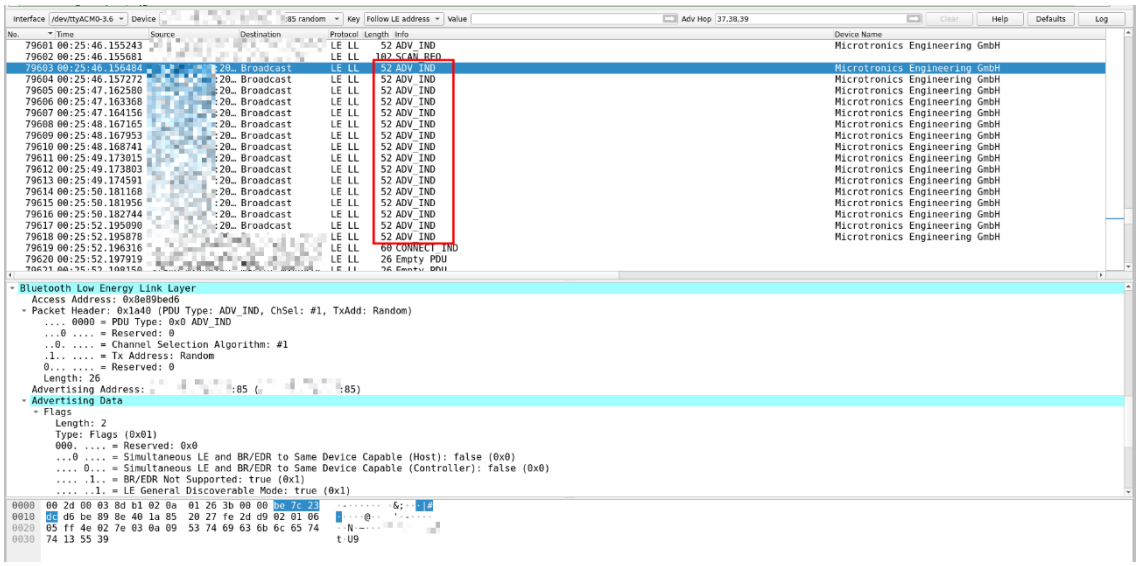
BHM-2025-E11 – Invalid Parameter Input gets Processed in Database



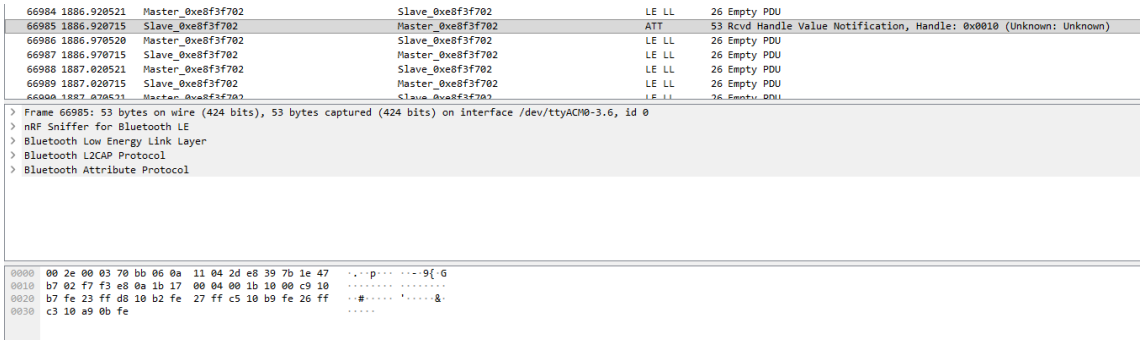
BHM-2025-E12 – Service Identifiers in BLE Packets



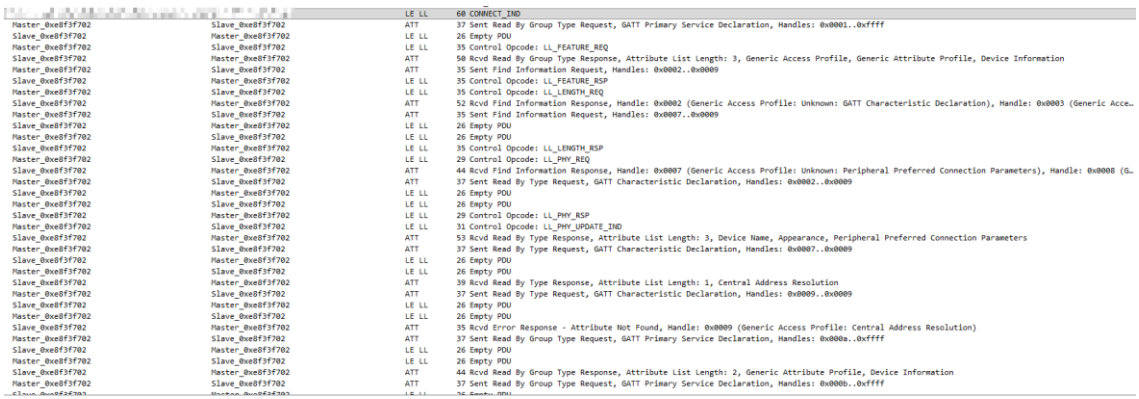
BHM-2025-E13 – Broadcast Packets of Sensor Unit



BHM-2025-E14 – Plaintext BLE Data Transmission



BHM-2025-E15 – BLE Pairing Process is missing Encryption Setup



Appendix 5 – Software-Tools

Software	Version	Use-Case
Kali Linux	2025.3	Base platform for security assessment
Windows 11	25H2	Host machine OS for Kali virtual machine
VirtualBox	7.0.14	Virtualization software for Kali VM
Wireshark	3.6.2	Monitoring wireless protocols (WiFi, BLE)
nRF Sniffer Firmware	4.1.1	Firmware for BLE sniffing
Burp Suite	2025.4.4	HTTP network proxy & repeater
Binwalk	3.1.0	Firmware analysis & extraction tool
Ghidra	11.4.2	Firmware reverse engineering
Strings	2.34	Firmware string extraction
File	5.38	Firmware file type analysis
Nmap	7.80	Network scanner
Android Studio	2025.1.1	APK analysis & Android emulation base
Android Emulator	35.6.11	Emulator for virtual Android devices
Android	16	Operating system of virtual Android device
Visual Studio	1.104	File analysis & Serial Monitor
STM32CubeProgrammer	2.20.0	SWD interface software
Mosquitto	2.0.22	MQTT message broker