Threat Modelling and Vulnerability Assessment for IoT Solutions: A Case Study

Krasen Anatoliev Parvanov*
Department of Computer Science and Engineering
University of Gothenburg
Gothenburg, Sweden
krasenparvanov@gmail.com

Chrysostomos Tsagkidis*
Department of Computer Science and Engineering
University of Gothenburg
Gothenburg, Sweden
chrysostomos.tsagkidis@gmail.com

Francisco Gomes de Oliveira Neto
Department of Computer Science and Engineering
Chalmers University of Technology and the University of Gothenburg
Gothenburg, Sweden
francisco.gomes@cse.gu.se

ABSTRACT

Security in Internet of Things (IoT) systems is increasingly critical due to their growing adoption and complexity. This study investigates how threat modelling and vulnerability assessment are applied in industrial IoT development. We conducted a single-case study with an IoT consultancy company, combining semi-structured interviews with practitioners and analysis of development artefacts from a real-world IoT project. Our findings show that while threat identification is practiced, formal methodologies like STRIDE are only used selectively, and integration into the development process is informal and ad hoc. Vulnerability assessment also lacks systematic approaches for discovery, classification, and testing, often relying on expert judgment and external tools. We identify key attack surfaces and common security weaknesses, and highlight gaps in documentation and testing integration. The study contributes practical recommendations for improving security practices and provides a curated list of open-source penetration testing tools. These insights support more structured and proactive security strategies in IoT development.

KEYWORDS

Vulnerability Assessment, Threat Modelling, Penetration Testing Tools

1 Introduction

The rapid expansion of the Internet of Things (IoT) has introduced significant security concerns in the cyber-physical domain [29, 34, 35]. As more devices are deployed, the attack surface grows, exposing vulnerabilities that malicious actors may exploit [19, 39]. A key contributor to this issue is the limited amount of standardization in IoT development, alongside the complexity and heterogeneity of deployed devices and services [10, 28, 38].

IoT systems are integrated into critical domains such as automotive, medical, smart homes, and industrial automation, each requiring robust security mechanisms [22]. Ensuring confidentiality, integrity, and availability is paramount, especially as many devices can be accessed both physically and remotely. To build secure systems, it is essential to identify potential threats early—highlighting the importance of integrating threat modelling into the software development process. Threat modelling helps uncover attack surfaces and supports the implementation of preventative measures.

*Both authors contributed equally to this research.

Alongside threat modelling, vulnerability assessment is central to cybersecurity for IoT. It involves identifying, measuring, and prioritizing system weaknesses for mitigation [45]. Penetration testing plays a complementary role, enabling practitioners to discover and verify vulnerabilities using dedicated tools [7, 25].

Various frameworks support threat modelling, including STRIDE [3], CORAS [17], and PASTA [42], some of which have been applied to IoT contexts [12, 21]. Vulnerability assessment techniques have also been examined in prior work [10, 24], with the OWASP IoT Top Ten serving as a key industry reference [2]. However, gaps remain in standardization across the IoT lifecycle [10, 28, 38], secure architectural patterns [35, 48], and full-stack security analysis across IoT layers [35]. Notably, the OWASP IoT list has not been updated since 2018 [2].

This paper investigates how practitioners conduct threat modelling and vulnerability assessment in real-world IoT development. We focus on the challenges they face and the tools they use, particularly open-source penetration testing tools. The case study also explores how security considerations shape architectural decisions and influence design trade-offs. The main contributions of this paper are:

- A practitioner-focused analysis of how threat modelling is integrated into IoT software development, highlighting real-world challenges and practices.
- A categorized list of common IoT vulnerabilities and attack vectors, informed by industry experience and complemented by architectural insights.
- An evaluation of open source penetration testing tools relevant to IoT security, along with practical guidelines for their effective use.

These contributions offer a grounded perspective on the intersection of threat modelling, vulnerability assessment, and tool use in IoT. By bridging practitioner insights with research gaps, the paper supports the development of more secure IoT systems and moves the state-of-the-art forward through empirical findings, actionable practices, and tool-based recommendations.

2 Background and Related Work

The Internet of Things (IoT) refers to a network of devices embedded with electronics, software, and sensors connected to collect and exchange data [47]. Due to the diversity of its components, the IoT ecosystem presents significant security challenges [10, 22, 28, 39,

49]. The distributed nature of IoT assets increases the complexity of securing them across the system architecture.

A key issue is the fragmented focus in research, where studies often address only specific layers of the IoT architecture, overlooking critical levels such as the Network (L3) or Application (L4, L5, L7) layers [35]. Security challenges vary across application domains such as smart homes, industrial systems, automotive, and healthcare. Some of those domains involve safety-critical components or privacy-sensitive data [31, 32, 39]. To address these risks, threat modelling and vulnerability assessment are essential practices.

Despite the growing number of proposed methods for securing IoT systems, there is a lack of empirical evidence on how these methods are adopted and applied in real-world industrial settings. In particular, little is known about the practical challenges practitioners face and the effectiveness of existing guidelines and tools in supporting secure development. Our case study contributes to addressing this gap by providing empirical insights into how current IoT security practices are implemented by our industry partner.

2.1 Threat Modelling in IoT

Threat modelling helps developers identify and document potential security threats in a systematic way [14, 18, 23]. It plays a key role in secure software development lifecycles, and various frameworks have been proposed to support this activity [41, 46]. Among the most used threat modelling frameworks are:

- **STRIDE** [3]: Classifies threats into spoofing, tampering, repudiation, information disclosure, denial of service, and privilege escalation.
- **PASTA** [42]: A 7-step, risk-based process for identifying threats through attack simulation.
- **OCTAVE** [9]: Focuses on organizational risk assessment using asset and threat evaluation.
- **CORAS** [17]: A model-driven risk assessment approach using visual models and structured analysis.
- **DREAD** [26]: A risk rating methodology based on damage, exploitability, affected users, and discoverability.

While these frameworks are widely used, most were not designed with the specific constraints and characteristics of IoT in mind. To address this gap, Bugeja et al. [16] extended the OWASP Software Assurance Maturity Model (SAMM) [6] with IoT-specific practices based on industry data. Yet, the lack of guidance on integrating such frameworks into real-world development processes—especially in ways that influence architectural and design decisions—remains a challenge [35].

In terms of standards, the ISO/IEC 27000 series offers general security and risk management guidance. However, domain-specific coverage for IoT is still emerging, with ISO/IEC DIS 27400 under development [1]. Researchers also note that minimal security standards are not widely enforced during IoT product development [10, 13, 49]. Tools like ENISA's Good Practices for IoT [5, 11, 19] provide useful checklists, but how these are applied in practice is still unclear—underscoring the need for empirical studies on practitioner adoption.

2.2 Vulnerability Assessment and Penetration Testing Tools

Vulnerability assessment involves identifying, ranking, and prioritizing system vulnerabilities for mitigation [45]. It is a critical component in securing IoT solutions, providing developers with essential insights into weaknesses across devices and systems. Common frameworks like CVSS [4] and the OWASP IoT Top 10 [2] are frequently referenced, though many studies apply them generically, without addressing the specific constraints of IoT environments [11, 31, 34]. Several works link vulnerabilities to specific attack surfaces and vectors in IoT systems [20, 27, 30, 36]. However, OWASP's IoT framework, last updated in 2018, has not kept pace with the evolving landscape—leaving a gap for more current and domain-tailored approaches.

Penetration testing complements vulnerability assessment by simulating attacks to uncover real-world weaknesses [8, 44]. Tools such as NESSUS, NMAP, Burp Suite, Metasploit, and those within the Kali Linux distribution are widely used [10, 33, 43], though many of these tools are either generic or proprietary. As a result, there is a need for updated evaluations of open source penetration testing tools that are effective in the context of IoT.

3 Research Methodology

The objective of our study is to get an understanding of how threat modelling (TM) and vulnerability assessment are being applied by practitioners, and explore possible open source penetration testing tools that can be used based on the identified vulnerabilities. Therefore, we aim to answer the following research questions:

- ***RQ1.*** What threat modelling methodologies and standards do practitioners use for developing IoT solutions?
  - ***RQ1.1.*** How do practitioners integrate threat modelling within the software development process?
  - ***RQ1.2.*** How threat modelling frameworks can help identify security flaws in the software architecture design of IoT solutions?
- ***RQ2.*** What security vulnerabilities do practitioners identify within IoT infrastructures?
  - ***RQ2.1.*** What are the main attack vectors that practitioners identify, and how are they classified?
  - ***RQ2.2.*** How are vulnerabilities assessed and tested?
  - ***RQ2.3.*** What open-source penetration testing tools are available and can be used for discovering vulnerabilities?

RQ1 addresses IoT-related threat modelling techniques and standards and their application in the industry. Particularly, RQ1.1 explores further on how practitioners integrate threat modelling, standards and best practices into the software development process, while RQ1.2 looks at how threat modelling can help practitioners find possible architectural design flaws in IoT solutions.

In turn, RQ2 addresses the need for further exploration of vulnerabilities within the IoT sector by looking at the issues encountered in the industry. RQ2.1 demonstrates and classifies the main attack surfaces and the corresponding attack vectors that practitioners recognize. RQ2.2 dives into how the process of vulnerability assessment and testing is applied in the industry. Finally, RQ2.3 provides

a list of penetration testing open source tools that can address the discovery of vulnerabilities. Our goal with RQ2.3 is to complement the qualitative findings with recommendations to practitioners searching for tools for their specific domain.

3.1 Case Study and Industry Partner

To answer our research questions, we conducted a case study with an industry partner. This involved qualitative analysis of two data collection steps: (i) an interview-based study to capture the experiences and practices of professionals working in IoT security, and (ii) an inspection of IoT related tools and artefacts used in industry and open source.

The software development company chosen for this case is QRTECH, a consultancy company based in Sweden and they specialise in the embedded and IoT domain with more than 20 years on the market. The company's experience in multiple IoT domains will allow for diverse observations that are not limited to a specific IoT area. Within this study we are going to focus on their IoT team, their process, and projects, in terms of artefacts such as requirements, diagrams, and IoT solutions that they provide. In addition, practitioners involved in the security-oriented research and development division of the company were included as part of this study, as well as external cyber-security experts that they cooperate with.

Figure 1 outlines the steps of our study. Step 1 involved the initial planning phase, including discussions with the company, defining the study's goal, and formulating the research questions based on our literature review (Step 2). Steps 3 to 10 describe the planning and execution of the *interview study* and the thematic analysis. These steps included designing the interview instrument, conducting the interviews, and summarizing key findings through an iterative thematic analysis process, detailed later in this section.

Steps 11 to 15 cover the *artefact analysis*, which was based on repository data provided by the company. We developed an artefact analysis template to guide the review of requirements, bug reports, tests, and diagrams related to the IoT infrastructure. The analysis process included data extraction, individual review of components, and evaluation of findings to ensure consistency. In Step 16, we performed an exploratory search and review of open source penetration testing tools. Finally, after synthesizing the data from interviews, artefacts, and tool analysis, we presented our results to company practitioners to validate our findings.

3.2 Data collection

To answer our research questions, we used multiple data sources and collection techniques, ensuring data triangulation and reducing threats to validity. Our qualitative data stems from two categories: process-related and product-related. Process-related data was collected through interviews focused on various stages of the IoT software development life cycle (SDLC), particularly threat modelling and vulnerability assessment. Product-related data includes artefacts such as tests, requirements, merge requests, bug reports, architecture documents and diagrams, and source code. We also complemented our interview data with information on open source penetration testing tools gathered from literature and online sources.

Table 1: Overview of our interviewed participants. Experience is reported in years.

No.	Role	Exp.
P1	Software Architect and Project Manager	15
P2	Project Manager	20*
P3	Development Engineer (R&D)	10
P4	Embedded Developer	6
P5	Development Engineer (R&D) and Project Manager	6.5
P6	Software Engineer	14
P7	Cyber-security expert	28
P8	Solution architect & full-stack developer	7

*12 within software engineering and management, and 8 in sales

Interviews. We conducted first-degree data collection through semi-structured interviews with practitioners at the company. All interviewees were asked the same core questions, with follow-up questions adjusted based on their roles and responses. The interview guide was designed using insights from the literature review and aimed to gather practitioner perspectives relevant to our research questions (available in our supplementary material).

We interviewed eight practitioners with roles across development, management, and cyber-security (Table 1). Participants were selected via convenience sampling within the same company, due to feasibility and relevance as they were directly involved in developing the IoT project under analysis. This allowed us to gain diverse perspectives and insights across the SDLC.

The interviews were recorded using Audacity[1] and transcribed by authors for further data analysis.

Artefact. Following the interviews, we conducted third-degree data collection on a project's artefacts. The selected project, developed by the company for an industrial client, was chosen based on its representativeness and completed status, allowing for a comprehensive examination of development activities and outcomes. The data was stored in a private GitLab repository, including source code, architecture documentation, issue tracking, and test artefacts. We extracted data such as requirements (as GitLab issues), bug reports, merge requests, and architectural diagrams using the GitLab API[2]. Documentation and code repositories were cloned locally for analysis. The artefacts analysed are described in Table 2

For the penetration testing tools, we gathered data from trusted sources including: (i) the OWASP Foundation, (ii) academic publications, (iii) white papers, and (iv) practitioner input. The inclusion criteria were open source availability and relevance to vulnerability discovery described in the tool's documentation. The data was compiled in Section 4, where we list each tool's name, description, and application domain.

[1]https://www.audacityteam.org/
[2]https://docs.gitlab.com/ee/api/

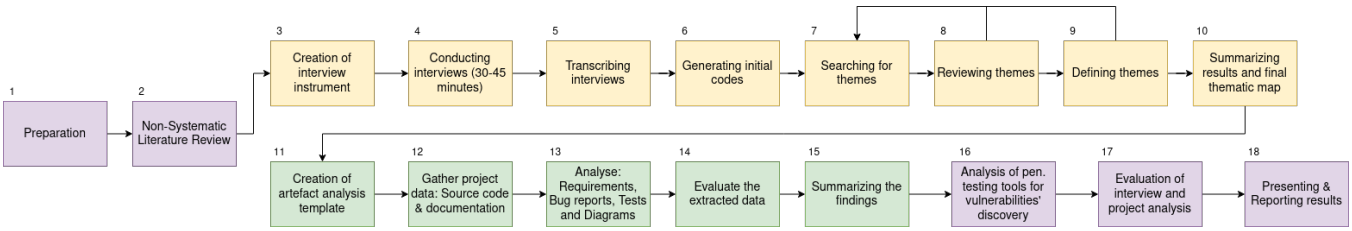

Figure 1: Research process overview. The steps taken during the study are shown in order (1-18). Interview related activities are highlighted in yellow, artefact analysis activities shown in green, whereas the main workflow in purple.

Table 2: Description of analysed artefact components.

Components	Description	IDs
Database communication	Database related components responsible for accessing and communicating to the database from the cloud infrastructure	DC
Web application	Client-server components responsible for the user interaction and the communication with the rest of the cloud infrastructure	WA
Cloud infrastructure deployment	Components responsible for handling the deployment of the services needed for the instantiation and management of the cloud infrastructure	CID
Device OS	Components responsible for building the OS for the devices	Dev-OS
Device configuration	Components related to configuring and managing the device	Dev-Conf
Device packaging	Component responsible for creating a baseline image used for production	Dev-PKG
Device communication	Components responsible for handling different communication channels from and to the device	Dev-Comm

3.3 Data analysis

We use thematic analysis to analyze our qualitative data (e.g., interviews and analysis of artefacts). We conducted the 6-phase approach presented by Braun and Clarke[15].

(1) **Familiarizing with the data:** We engaged with the data while transcribing and reviewing the interviews together.
(2) **Generating initial codes:** Both authors jointly coded all transcripts using an inductive, open coding approach. This ensured shared understanding, minimized disagreements, and helped avoid bias.
(3) **Searching for themes:** Codes were transferred to Miro[3] as sticky notes, allowing us to iteratively group and label them, which led to the emergence of themes.
(4) **Reviewing themes:** We held review sessions to refine themes, resolve inconsistencies, and begin forming sub-themes and higher-level categories.
(5) **Defining and naming themes:** Through four iterations, we finalized descriptive names for the high-level themes and sub-themes, continuing until theme saturation was reached.
(6) **Producing the report:** A thematic map was created based on the final themes, presented in Section 4.

For the artefact analysis, we focused on components developed or customized by the company. This scope was defined collaboratively with company representatives to avoid assumptions or bias. Low-level components like the Linux kernel and firmware were excluded. The analysis covered the IoT device, frontend application, and backend microservices, including the database, monitoring service, communication relays, and web server.

Documentation analysis was guided by a predefined evaluation template based on our research questions [40]. Each researcher independently reviewed assigned components, extracted data using the template. We then jointly reviewed and categorized the data by relevant security domains. The template and component list are included in our data analysis package in Zenodo. Afterward, we compared artefact findings with the thematic analysis to identify potential overlaps or patterns.

4 Results

We reached theme saturation after a total of four iterations of the coding both the interviews and artefacts. The results of the themes that emerged are as follows: 7 main themes, 40 sub-themes, some of which were broken further into specific levels. From the artefacts (e.g., issues, bug reports, merge requests, and architectural documentation) we identified 13 categories related to security threats, mitigation techniques, vulnerabilities, and attack vectors. These categories align with key themes from the interview data, particularly *Threat identification* and *Best practices and recommendations*, which were strongly supported by evidence from both sources. For example, artefacts revealed how practitioners addressed secure communication, device-level protections, and authentication and authorisation mechanisms—topics that were also frequently raised during the interviews and thus reflected in the final thematic map.

A detailed description of the main themes can be found in Table 3. We divided our theme map into two parts focusing respectively on: (i) threat modelling, standards and guidelines; and (ii) vulnerabilities, and attack vectors. For brevity, we omit the remaining themes. Nonetheless, all themes, sub-themes and corresponding

[3]https://miro.com/

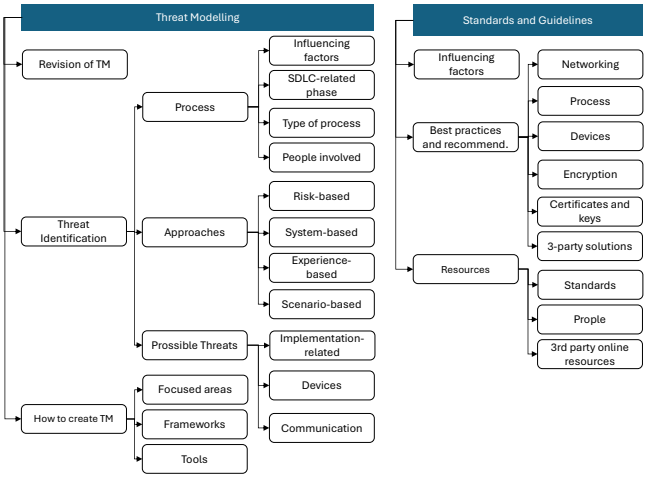

Figure 2: Theme map with sub-themes of the data related to threat modelling, standards, and guidelines reported by practitioners.

quotes from the interviews/artefacts are made available in our data package.

4.1 RQ1: Threat Modelling in Practice

Figure 2 shows the theme map with sub-levels related to threat modelling. Practitioners involved in IoT development reported that no formal **threat modelling methodology** is consistently applied during the development process. Instead, they rely on intuition and professional experience.

> "*There's no strict procedure, it's just common sense I would say.*" - Partic. 4

Despite the absence of formal methods, **threat identification** is regularly performed. Practitioners described it as an iterative process that occurs throughout development and is influenced by project roles, team composition, and the organizational context.

> "*If I'm a part of the team that are implementing, that is the architecture, I would in case I have that role, if I am a tech lead or software responsible somehow, I would try to iteratively enforce it definitely.*" - Partic. 8

Threat identification approaches varied across cases and included risk-driven, scenario-based, structural, and experience-based techniques, often depending on the system's complexity and team expertise.

> "*I mostly think of threats from a technical perspective, well, when you design a system... When we design the system, we try to think holistically.*" - Partic. 1

In cases requiring deeper security scrutiny, external experts are brought in to guide the creation of detailed threat models.

> "*And then I would probably have to bring in a colleague or expertise, because I am not an expert...*" - Partic. 8

Among those who do create structured threat models, the STRIDE framework is preferred, particularly by security specialists. It is valued for its clarity and ability to support secure development practices such as defining "done" criteria.

> "*So for development teams, I usually use STRIDE, because I find that this is way to manage. People tend to understand it, and it also provides a good way to for the engineering team for a Definition of Done.*" - Partic. 7

Some practitioners use supporting tools like the Microsoft Threat Modelling Tool to formalize and visualize threats.

> "*Me and [name of co-worker], who just had begun working with this problems, we felt that the best way, for example, to model threats is to use the software, like Microsoft Threat Modelling tool.*" - Partic. 3

The adoption of **security standards** is largely context-dependent. Some practitioners rely solely on their own expertise, while others follow formal standards in safety-critical environments, such as ISO 26262, ISO 27000, ISO 15408, IEC 62433, or guidelines from OWASP and SAE.

> "*We don't directly look at standards... The security aspect of our products is normally from the individuals, and their experience and expertise rather than following standards.*" - Partic. 1

> "*We are trying to follow the standards, typical available standards, for cyber security from ISO and SAE.*" - Partic. 3

RQ1: What threat modelling methodologies and standards do practitioners use for developing IoT solutions?
Practitioners do not follow a specific threat modelling methodology but rely on risk-, scenario-, structural-, and experience-driven approaches. For security-sensitive solutions, STRIDE is commonly used. Security standards (from the ISO family) are applied in safety-critical cases.

4.2 Integrating Threat Modelling into the Development Process

Threat identification and mitigation activities are integrated into development, particularly during the requirements and design phases. These are reflected in GitLab issues, code reviews, and architectural decisions. However, the process lacks consistency and detailed documentation.

> "*I mean we make sure that this, the decisions, had traceability, so. You know, you could track what we did and why, but there was no heavy documentation.*" - Partic. 4

Security considerations focus on areas like authentication, authorization, communication, and credential management.

> "*The stage is always in specification and design phase... are the two main parts where I think about security implications.*" - Partic. 1

> "*Usually in architectural phase you should take those decisions[security] fairly well.*" - Partic. 8

Table 3: Description of main themes

ID	Theme	Description
MT-1	Attack surfaces	Covering possible attack surfaces of IoT systems that can be vulnerable to attack including people.
MT-2	Attack vectors	Possible ways that an attacker could gain access to the IoT systems
MT-3	Vulnerabilities	Weaknesses in IoT systems that can be exploited
MT-4	Vulnerability assessment	Steps in the process of assessing IoT solutions and the vulnerabilities in terms of discovering, testing and classification
MT-5	Threat modelling	Including techniques, tools and frameworks assisting practitioners in identifying and documenting potential threats
MT-6	Standards and Guidelines	Covers security related standards, recommendations, guidelines and best practices
MT-7	Security challenges and Issues	Identified security related issues and problems that practitioners face throughout their work

Threat identification is collaborative and iterative, involving clients, developers, managers, and experts depending on the use case and project context.

> "*I would say you have a discussion with the client in the beginning and then you have a discussion with the team internally during the architectural phase.*" - Partic. 2

RQ1.1: How do practitioners integrate threat modelling within the software development process?

Threat analysis and mitigation are carried out iteratively across the SDLC, mainly during the requirements and design phases. Efforts focus on authentication, authorization, communication, and credential management, involving collaboration across roles but with limited documentation.

Practitioners use frameworks like STRIDE when needed, especially in security-critical contexts. These frameworks support system decomposition, identify attack paths, and help define security requirements.

> "*I use only STRIDE for application decomposition and I try to educate the teams because it's easy to understand and easy to implement, and also provides a Definition of Done... I'm trying to put requirements into place to the development organization.*" - Partic. 7

They also serve as educational tools for raising awareness of implicit assumptions and overlooked security decisions during design.

> "*Then they realize that they have so many implicit assumptions... they are just accepting it.*" - Partic. 7

Despite being seen as effort-intensive, experienced practitioners emphasize that threat modelling helps surface critical security aspects and improves IoT system design.

> "*...threat modelling is an activity that is kind of painful but you would like to have a light touch on your existing architecture... and just try to list what is important.*" - Partic. 7

RQ1.2: How threat modelling frameworks can help identify security flaws in the software architecture design of IoT solutions?

Threat modelling frameworks, such as STRIDE, support architectural decomposition, reveal potential attack paths, and help reduce implicit assumptions. While applying them can be challenging, they offer clear benefits for identifying and addressing security flaws in IoT system design.

4.3 RQ2: Vulnerabilities and Attacks

Figure 3 presents the themes and sub-themes related to vulnerabilities and attacks. Our analysis of interview and artefact data reveals that practitioners identify vulnerabilities across various areas: configuration, human factors, devices, privileges, protocols, encryption, certificates and keys, endpoints, and design decisions. Other mentioned sources include zero-days, backdoors, and deployment flaws.

Configuration issues were commonly reported, such as weak or hard-coded passwords, open ports (e.g., SSH), misconfigured cloud services, and insecure handling of credentials or third-party orchestration tools.

> "*Then typically all the things with your cloud infrastructure that you are missing that you are not doing pen test on your cloud infrastructure, you have open ports, you have open buckets*" - Partic. 7

Human factors and organizational tools are also seen as significant vulnerability sources. Practitioners pointed to weaknesses in communication and data storage tools, and emphasized human error as a high-risk area.

> "*In my opinion, I think humans are the most likely source of security breach, so I would probably tend to look more at the human.*" - Partic. 8

Certificate and key management issues were also noted—especially the reuse of certificates across devices, which can lead to widespread compromise. Device-level vulnerabilities include open debug pins,

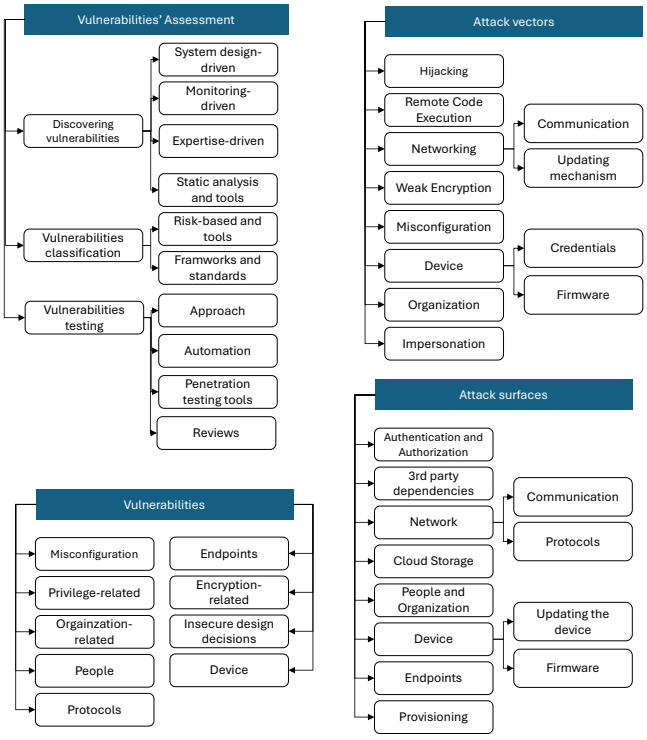

Figure 3: Theme map with sub-themes of the data related to attack vectors, surfaces as well as vulnerabilities.

lack of physical hardening, and insecure data storage on the device itself.

> "*By making sure that you have a per device unique pre-shared key, for instance, you can know that breaking one device does not break any more, which is a common mistake in IoT.*" - Partic. 4

Access control and privilege escalation issues were observed in both interviews and artefacts, with practitioners highlighting bugs that allow unauthorized role elevation.

> "*Privilege escalation problems are due to bugs that allow users to assume a privileged role which they should not be able to to assume.*" - Partic. 6

Protocol and encryption vulnerabilities were also reported. These include weak API implementations, insecure endpoints, limited encryption due to hardware constraints, and outdated cryptographic practices in legacy systems.

> "*There are [legacy systems] still like using max 8 character passwords over a non encrypted TCP connection.*" - Partic. 6

Finally, practitioners acknowledged the risk of zero-day vulnerabilities and backdoors, which are difficult to detect and can be exploited by external attackers.

RQ2: What security vulnerabilities do practitioners identify within IoT infrastructures?

Practitioners highlight vulnerabilities related to weak or hard-coded passwords, misconfigured ports, credential and secret management, and third-party configurations. Additional risks include human factors, device-level weaknesses, insecure protocols, poor certificate handling, encryption flaws, and access control issues.

The attack vectors identified by practitioners fall into the following categories: organization, credentials, networking, device, weak encryption, misconfiguration, remote code execution, hijacking, and impersonation.

Organizational attack vectors involve social engineering, industrial espionage, and weaknesses in internal processes—particularly in how credentials and sensitive information are handled.

> "*There's a lot of processes at the organization that are open to attack.*" - Partic. 1

Networking-related attacks target communication protocols and update mechanisms. Practitioners noted insecure communication links, data injection, data leakage, and weak protocol implementations (e.g., MQTT, MODBUS, BACnet) as common entry points.

> "*I see quite often that..., is this fieldbuses, MODBUS and BACnet, and so on, or as of the few last few years MQTT has somehow made its way to automation and some of those protocols they don't really have any advanced ways of enforcing access.*" - Partic. 8

Device-level attacks were frequently mentioned and supported by artefact analysis. These include exploiting debug ports, firmware vulnerabilities, OS-level flaws, fault injection, and physical tampering.

> "*The entire software stack of Linux on the device which is an IoT device it's actually a lot of technical debt just in the OS.*" - Partic. 6

Credential-related vectors involve poor key management, including reused or shared certificates. Practitioners stressed the importance of unique credentials for each device and user.

> "*So, one attack vector would be if the people designing the system doesn't have unique credentials for each device and for each user, that's definitely vector of attack.*" - Partic. 1

Additional vectors include remote code execution on cloud servers, misuse of default passwords, and weak or custom-built cryptographic solutions.

> "*Another huge mistake is inventing your own crypto solutions.*" - Partic. 4

Practitioners also highlighted risks of hijacking and impersonation, where attackers pose as trusted entities. Other potential threats include backdoors, zero-day exploits, and classical injection attacks such as buffer overflow and SQL injection.

> "*Data that being inserted from the client side or the customer side, has to be protected from, you know, overflow. You know, buffer overflows, bad data injection, so you don't have you know SQL injection attacks.*" - Partic. 1

RQ2.1: What are the main attack vectors that practitioners identify, and how are they classified?
Practitioners classify attack vectors into organizational, credential, networking, device, encryption, and configuration-related categories. Key threats include social engineering, compromised credentials, firmware exploitation, and insecure communication protocols.

Our findings on **vulnerability assessment** focus on how practitioners identify, classify, and test for system weaknesses. Three main approaches to identifying vulnerabilities emerged: system design-driven, monitoring-driven, and expertise-driven.

In the design-driven approach, practitioners examine system architecture, data flow, and component interactions. The *expertise-driven* approach relies on guidance from external cybersecurity professionals, often through discussions or consultations.

> "*We don't have a process or tool sets to ourselves find these vulnerabilities in our systems, so we rely on, what would you call IT security professionals of the world to let us know.*" - Partic. 1

The monitoring-driven approach involves continuous tracking of dependencies and the use of vulnerability management systems. Some practitioners also use scenario-based assessments and reference the OWASP IoT Top 10 checklist. Static code analysis tools are used to detect code-level weaknesses and manage risks across the software dependency tree.

> "*You can do things like static code analysis to make sure that things like your C code is not completely broken and will allow buffer overflow and what not... those are practices in coding and practices in managing your software releases.*" - Partic. 6

When it comes to classifying vulnerabilities, no unified framework is used by development teams, with many relying on individual expertise.

> "*We don't have any frameworks for classification. Again, we rely on people's experience and expertise to do that classification for us.*" - Partic. 1

However, cybersecurity experts supporting the teams report using a combination of risk-based approaches, including CVSS and ISO standards, depending on the threat context.

> "*I use CEM 3.0 [ISO 15408], typically, and for remote attacks I used CVSS when needed... because I also consider risk assessment, also on the product.*" - Partic. 7

In terms of testing, no systematic validation or verification process was identified. Instead, practitioners perform architectural and design reviews informally and on an ad-hoc basis.

> "*Solution reviews [performed in the organisation] where you would have to motivate some of your choices regarding how you would solve authentication and network partitioning and other things.*" - Partic. 6

Security testing is often outsourced or considered unnecessary when using off-the-shelf hardware.

> "*If you have a low number of devices you probably want to use some kind of off-the-shelf hardware and you want to customize the software more and not have to worry too much about doing extensive security testing.*" - Partic. 6

Some practitioners highlighted the value of using threat modelling to identify high-risk areas for targeted testing and test case generation. Penetration testing was also viewed as a useful complementary activity.

> "*The threat modelling and threat analysis helps to see which area of the system is more vulnerable to test it. And then we start to write test cases.*" - Partic. 5

RQ2.2: How are vulnerabilities assessed and tested?
Vulnerabilities are identified using system design reviews, monitoring tools, and external expertise. Static code analysis supports detection in code and dependencies. While no standard classification framework is used, cybersecurity experts apply risk-based methods with CVSS and CEM 3.0. Testing is informal, relying on design walkthroughs and reviews rather than a systematic process.

4.4 RQ2.3: Penetration Testing Tools

Practitioners that conduct research in the IoT domain and the cybersecurity expert reported that in their work they have been using **open source penetration testing tools** for discovering vulnerabilities found in the Kali Linux Distribution. Some notable tools found in the distribution that were mentioned by practitioners are Wireshark, and the Metasploit framework.

> "*We are using mostly tools which are available in Kali Linux...this[pen-testing tool] was wireshark, which I think is one of the most common... for monitoring the network.*" - Partic. 3

A list of open source penetration testing tools in regards to discovering vulnerabilities in IoT solutions can be found in Table 4. The presented tools are grouped in different domains, in order to address different aspects of the IoT solutions, namely the device, web services, mobile applications, networking, as well as people oriented risks.

RQ2.3: What open source penetration testing tools are available and can be used for discovering vulnerabilities?
Practitioners use various tools from Kali Linux for penetration testing. Additionally, our analysis identified open source tools covering multiple IoT domains, including firmware, networking, mobile, web applications, and social engineering, as listed in Table 4.

5 Discussion

This study contributes to a deeper understanding of how threat modelling and vulnerability assessment are practiced in real-world

Table 4: Penetration Testing Tools

Pen-Tool	Description	Domain
OWASP Dependency-Check	Dependency checking framework for vulnerabilities in dependencies	Application dependency
RetireJS	Detects dependency versions with known vulnerabilities	Application dependency
Metasploit framework	Platform that includes various tools spanning from discovering vulnerabilities to exploitation	Broad domain
Flashrom	Utility for flash chips identification, reading, writing, and verification	Firmware
Binwalk	Tool for firmware oriented analysis and reverse engineering	Firmware
Firmadyne	Tool used to perform emulation and dynamic evaluation on Linux-Based firmware	Firmware
Firmwalker	Script for discovering firmware file-system related issues	Firmware
Firmware analysis toolkit	Toolkit for identifying and analysing vulnerabilities in IoT and embedded devices	Firmware
Binary Analysis Tool (BAT)	Tool used for binary code analysis for finding issues	Firmware
Mob-SF	Mobile security framework for static, dynamic, and malware analysis as well as pen testing	Android
Wireshark	Detailed network protocol analyser	Network
nmap	Utility used for network discovery and auditing	Network
sslscan2	Scanner that tests SSL/TLS enabled services	Network
Ettercap	Comprehensive platform for MiTM attacks	Network
Sslstrip	Utility assisting in MiTM attacks	Network
DNSRecon	Script for DNS-oriented analysis and security assessment	Network
DNSChef	DNS proxy tool for conducting application network traffic analysis	Network
Spiderfoot	Open source intelligence tool that gathers information such as IP addresses, network subnet etc	OSINT
The Social-Engineer Toolkit	SET is a toolkit for performing social engineering penetration testing	People
OWASP Zap	Web application security scanner	Web
w3af	Web application attack and auditing framework	Web
Recon-ng	Comprehensive framework for web-based reconnaissance	Web
Kismet	Wireless networking sniffer for device detection, wireless intrusion, and wardriving	Wireless

IoT development. By combining interview insights with artefact analysis, our findings reveal a clear gap between the frameworks and tools proposed in research and their adoption in industry. Below, we highlight three key implications that emerge from our results.

1. Need for systematic integration of threat modelling into development workflows. Despite the availability of frameworks such as STRIDE, PASTA, and CORAS [3, 17, 42], practitioners rarely apply them systematically in IoT projects. This confirms previous concerns about the lack of integration between threat modelling theory and practice [35]. Our findings show that while threat identification does occur (often during the design phase) it is informal and fragmented. Moreover, participants also use their personal experience or intuition to complement the systematic approaches and consider domain-specific needs. Our findings reinforces the need for lightweight, context-aware methods that are easy to adopt and integrate throughout the software development life cycle (SDLC), particularly given the complexity and layered nature of IoT systems [10, 39].

2. Lack of structured vulnerability assessment practices. Practitioners use a mix of ad hoc strategies, including static code analysis, expert consultation, and general checklists (e.g., OWASP IoT Top 10), but do not follow a structured framework for identifying, classifying, or testing vulnerabilities. This is consistent with earlier studies that highlighted the absence of standard vulnerability assessment approaches tailored for IoT environments [20, 31]. While some experts rely on frameworks such as CVSS and ISO 15408 for classification, these are not uniformly adopted across development teams. This inconsistency limits traceability and weakens the ability to prioritize mitigation efforts—underscoring the need for better tooling and integration of vulnerability assessment into CI/CD pipelines.

3. Fragmentation in tooling and limited coverage of IoT-specific security layers. Our study found that most penetration testing tools in use come from general-purpose collections such as Kali Linux, with limited tailoring to IoT-specific contexts. While prior work has emphasized the importance of simulating realistic threats [43, 44], there remains a need for open source tools that align with IoT's diverse stack—from firmware to cloud. Moreover,

security efforts in practice tend to focus on specific architectural layers, particularly communication and credential management, leaving other critical layers (e.g., device firmware, network transport) underexamined—mirroring gaps identified in research [35].

These implications collectively point to a maturity gap in how security activities are adopted in practice. While research has provided valuable frameworks and tools, there is a pressing need for empirical guidance on how to adapt, simplify, and embed these approaches into industrial workflows. As IoT systems grow in complexity and risk, addressing these gaps is essential for improving the security posture of future deployments.

5.1 Threats to Validity

Construct validity. This threat concerns whether we measured what we intended to. One risk is that interviewees may have interpreted questions differently than intended, due to ambiguity or unfamiliar terminology. To reduce this risk, we designed overlapping questions that probed similar topics from different angles, helping us identify inconsistencies or misunderstandings. Additionally, we explicitly linked each interview question to the research questions to ensure focused data collection. Triangulation across interviews, documentation, and artefacts further strengthened construct validity by allowing us to cross-verify insights.

Internal validity. Although internal validity typically applies to causal inference, it remains relevant in qualitative studies where contextual factors can influence responses. In our case, organizational policies, norms, or culture might have shaped how participants responded—particularly on sensitive topics like security practices. To reduce this bias, we included participants with diverse roles, seniority levels, and backgrounds, which allowed for multiple perspectives and helped balance individual or departmental influences.

External validity. As this study is based on a single case, the findings are inherently context-specific and limited in generalisation. The company operates in the IoT domain and has a long-standing presence across several industries, but its practices may not reflect those of other organizations with different constraints, sizes, or sectors. Nevertheless, the details and depth of our analysis may offer transferable insights for similar IoT settings, especially those dealing with embedded systems and distributed infrastructures.

Reliability. Reliability refers to the consistency of the study process and the potential for replication. Qualitative research involves interpretive judgment, and our backgrounds and perspectives may have influenced how we conducted interviews and analyzed data. To address this, we followed established case study guidelines from Runeson and Höst [37], documented all steps of the research process, and held regular peer debriefings to discuss coding decisions, emerging themes, and interpretations. This transparent approach increases the study's dependability and supports reproducibility by other researchers.

6 Conclusion

In this study, we investigated how threat modelling and vulnerability assessment are applied in industrial IoT development. Through interviews and artefact analysis of a complete IoT solution, we explored the frameworks practitioners use, how these are integrated into their workflows, and how vulnerabilities are assessed and tested in practice.

Our findings show that although threat modelling and vulnerability assessment are well-researched areas, they remain loosely integrated into the software development lifecycle. Practitioners often identify threats informally and assess vulnerabilities without a structured methodology, even when supporting frameworks and tools are available. While architectural evaluations and ad hoc reviews are common, security testing and penetration testing are not consistently incorporated into development workflows. To address this, we compiled a set of open-source penetration testing tools relevant to key IoT domains, including network, device, web, mobile, dependency, and social engineering attack surfaces.

To support systematic adoption, we proposed practical recommendations such as using lightweight frameworks like STRIDE, leveraging documentation tools, and aligning security efforts with architectural decomposition and risk prioritization early in the development process.

While this study provides a holistic view of how security is approached in the context of one IoT organization, further research focusing on specific IoT components or layers—such as edge computing, firmware, or connectivity—could yield more detailed insights. Future work may also explore how mitigation strategies and intrusion detection mechanisms can be effectively incorporated into development and deployment practices.

ARTIFACT AVAILABILITY

We make our data available through Zenodo to support transparency and reproducibility in the link: https://zenodo.org/records/15283436. Due to non-disclosure agreements with our industry partner, we cannot share the artefacts used in the qualitative analysis. However, we provide access to the interview guide, data collection templates, and the thematic map along with the corresponding interview quotes generated during our thematic analysis.

ACKNOWLEDGMENTS

We would like to thank everyone at QRTECH for providing us with the means to conduct this study and their valuable support. We are particularly grateful to our study's participants for sharing their knowledge and experience.

REFERENCES

[1] [n. d.]. ISO/IEC FDIS 27400. https://www.iso.org/standard/44373.html accessed 2022-03-08.
[2] [n. d.]. OWASP Internet of Things Project - OWASP. Wiki.owasp.org [Online]. https://wiki.owasp.org/index.php/OWASP_Internet_of_Things_Project accessed 2022-02-26.
[3] 2009 [Online]. The STRIDE Threat Model. Docs.microsoft.com [Online]. https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20) accessed 2022-02-26.
[4] 2022. CVSS v3.1 Specification Document. FIRST — Forum of Incident Response and Security Teams. https://www.first.org/cvss/specification-document accessed 2022-03-08.
[5] 2022. Enisa.europa.eu. https://www.enisa.europa.eu/topics/iot-and-smart-infrastructures/iot/good-practices-for-iot-and-smart-infrastructures-tool accessed 2022-03-08.
[6] 2022. Software Assurance Maturity Model (SAMM). owasp.org. https://owasp.org/www-pdf-archive/SAMM_Core_V1-5_FINAL.pdf accessed 2022-03-08.

[7] Palak Aar and Aman Sharma. 2017. Analysis of Penetration Testing Tools. *International Journal of Advanced Research in Computer Science and Software Engineering* 7 (10 2017), 36. doi:10.23956/ijarcsse.v7i9.408

[8] Ahmad Salah Al-Ahmad, Hasan Kahtan, Fadhl Hujainah, and Hamid A. Jalab. 2019. Systematic Literature Review on Penetration Testing for Mobile Cloud Computing Applications. *IEEE Access* 7 (2019), 173524–173540. doi:10.1109/ACCESS.2019.2956770

[9] Christopher Alberts and Audrey Dorofee. 2002. *Managing Information Security Risks: The OCTAVE Approach.* Addison-Wesley Professional.

[10] Pooja Anand, Yashwant Singh, Arvind Selwal, Mamoun Alazab, Sudeep Tanwar, and Neeraj Kumar. 2020. IoT Vulnerability Assessment for Sustainable Computing: Threats, Current Solutions, and Open Challenges. *IEEE Access* 8 (09 2020). doi:10.1109/ACCESS.2020.3022842

[11] Pooja Anand, Yashwant Singh, Arvind Selwal, Pradeep Kumar Singh, Raluca Andreea Felseghi, and Maria Simona Raboaca. 2020. IoVT: Internet of vulnerable things? threat architecture, attack surfaces, and vulnerabilities in internet of things and its applications towards smart grids. *Energies* 13 (2020). Issue 18. doi:10.3390/en13184813

[12] Peter Aufner. 2020. The IoT security gap: a look down into the valley between threat models and their implementation. *International Journal of Information Security* 19 (02 2020). doi:10.1007/s10207-019-00445-y

[13] Ahmed Banafa. 2016 [Online]. IoT Standardization and Implementation Challenges. IEEE Internet of Things. https://iot.ieee.org/newsletter/july-2016/iot-standardization-and-implementation-challenges.html

[14] A.O. Baquero, Andrew Kornecki, and Janusz Zalewski. 2015. Threat modeling for aviation computer security. *CrossTalk* 28 (01 2015), 21–27.

[15] Virginia Braun, Victoria Clarke, Nikki Hayfield, and G. Terry. 2019. *Thematic analysis.* Springer, Singapore, 843–860. doi:10.1007/978-981-10-5251-4_103

[16] Joseph Bugeja, Bahtijar Vogel, Andreas Jacobsson, and Rimpu Varshney. 2019. IoTSM: An End-to-end Security Model for IoT Ecosystems, In 2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops). *2019 IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2019.* doi:10.1109/PERCOMW.2019.8730672

[17] F. den Braber, I. Hogganvik, M. S. Lund, K. Stølen, and F. Vraalsen. 2007. Model-based security analysis in seven steps - A guided tour to the CORAS method. *BT Technology Journal* 25 (2007). Issue 1. doi:10.1007/s10550-007-0013-9

[18] Danny Dhillon. 2011. Developer-driven threat modeling: Lessons learned in the trenches. *IEEE Security and Privacy* 9 (2011). Issue 4. doi:10.1109/MSP.2011.47

[19] Amir Djenna, S. Harous, and Djamel Eddine Saidouni. 2021. Internet of Things Meet Internet of Threats: New Concern Cyber Security Issues of Critical Cyber Infrastructure. *Applied Sciences* 11 (05 2021), 4580. doi:10.3390/app11104580

[20] Pietro Ferrara, Amit Kr Mandal, Agostino Cortesi, and Fausto Spoto. 2021. Static analysis for discovering IoT vulnerabilities. *International Journal on Software Tools for Technology Transfer* 23 (2 2021), 71–88. Issue 1. doi:10.1007/s10009-020-00592-x

[21] Massimo Ficco, Daniele Granata, Massimiliano Rak, and Giovanni Salzillo. 2021. Threat Modeling of Edge-Based IoT Applications. In *International Conference on the Quality of Information and Communications Technology*. Springer, 282–296.

[22] Mario FRUSTACI, Pace Pasquale, Gianluca Aloi, and Giancarlo Fortino. 2017. Evaluating Critical Security Issues of the IoT World: Present and Future Challenges. *IEEE Internet of Things Journal* PP (10 2017), 1–1. doi:10.1109/JIOT.2017.2767291

[23] Maxime Frydman, Guifré Ruiz, Elisa Heymann, Eduardo César, and Barton P. Miller. 2014. Automating risk analysis of software design models. *Scientific World Journal* 2014 (2014). doi:10.1155/2014/805856

[24] Gemini George and Sabu Thampi. 2019. Vulnerability-based risk assessment and mitigation strategies for edge devices in the Internet of Things. *Pervasive and Mobile Computing* 59 (08 2019), 101068. doi:10.1016/j.pmcj.2019.101068

[25] Aaron Guzman and Aditya Gupta. 2017. *IoT Penetration Testing Cookbook: Identify vulnerabilities and secure your smart devices.* Packt Publishing Ltd.

[26] M. Howard and D. LeBlanc. 2002. *Writing Secure Code* (second ed.). Microsoft Press.

[27] Xingbin Jiang, Michele Lora, and Sudipta Chattopadhyay. 2020. An Experimental Analysis of Security Vulnerabilities in Industrial IoT Devices. *ACM Transactions on Internet Technology (TOIT)* 20, 2, Article 16 (05 2020), 1–24 pages. doi:10.1145/3379542

[28] Nickson Karie, Nor Sahri, and Paul Haskell-Dowland. 2020. IoT Threat Detection Advances, Challenges and Future Directions. In *2020 Workshop on Emerging Technologies for Security in IoT (ETSecIoT).* 22–29. doi:10.1109/ETSecIoT50046.2020.00009

[29] Roger Kwon, Travis Ashley, Jerry Castleberry, Penny McKenzie, and Sri Nikhil Gupta Gourisetti. 2020. Cyber Threat Dictionary Using MITRE ATT&CK Matrix and NIST Cybersecurity Framework Mapping. In *2020 Resilience Week (RWS).* 106–112. doi:10.1109/RWS50334.2020.9241271

[30] Gurjan Lally and Daniele Sgandurra. 2018. Towards a framework for testing the security of IoT devices consistently, In International workshop on emerging technologies for authorization and authentication. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 11263 LNCS, 88–102. doi:10.1007/978-3-030-04372-8_8

[31] Xing Liu, Cheng Qian, William Grant Hatcher, Hansong Xu, Weixian Liao, and Wei Yu. 2019. Secure Internet of Things (IoT)-Based Smart-World Critical Infrastructures: Survey, Case Study and Research Opportunities. *IEEE Access* 7 (2019). doi:10.1109/ACCESS.2019.2920763

[32] Mobasshir Mahbub. 2020. Progressive researches on IoT security: An exhaustive analysis from the perspective of protocols, vulnerabilities, and preemptive architectonics. *Journal of Network and Computer Applications* 168 (2020). doi:10.1016/j.jnca.2020.102761

[33] Yasamin Mahmoodi, Sebastian Reiter, Alexander Viehl, Oliver Bringmann, and Wolfgang Rosenstiel. 2018. Attack surface modeling and assessment for penetration testing of IoT system designs, In 2018 21st Euromicro Conference on Digital System Design (DSD). *Proceedings - 21st Euromicro Conference on Digital System Design, DSD 2018.* doi:10.1109/DSD.2018.00043

[34] Sibin Mohan, Mikael Asplund, Gedare Bloom, Ahmad-Reza Sadeghi, Ahmad Ibrahim, Negin Salajageh, Paul Griffioen, and Bruno Sinipoli. 2018. Special Session: The Future of IoT Security. In *2018 International Conference on Embedded Software (EMSOFT).* 1–7. doi:10.1109/EMSOFT.2018.8537206

[35] Tanusan Rajmohan, Phu Nguyen, and Nicolas Ferry. 2022. A decade of research on patterns and architectures for IoT security. *Cybersecurity* 5 (01 2022). doi:10.1186/s42400-021-00104-7

[36] Syed Rizvi, R. J. Orr, Austin Cox, Prithvee Ashokkumar, and Mohammad R. Rizvi. 2020. Identifying the attack surface for IoT network. *Internet of Things (Netherlands)* 9 (2020). doi:10.1016/j.iot.2020.100162

[37] Per Runeson and Martin Höst. 2009. Guidelines for conducting and reporting case study research in software engineering. *Empirical software engineering* 14, 2 (2009), 131–164.

[38] Amar Seeam, Ochanya S. Ogbeh, Shivanand Guness, and Xavier Bellekens. 2019. Threat Modeling and Security Issues for the Internet of Things. In *2019 Conference on Next Generation Computing Applications (NextComp).* 1–8. doi:10.1109/NEXTCOMP.2019.8883642

[39] Astha Srivastava, Shashank Gupta, Megha Quamara, Pooja Chaudhary, and Vidyadhar Aski. 2020. Future IoT-Enabled Threats and Vulnerabilities: State of the Art, Challenges and Future Prospects. *International Journal of Communication Systems* 33 (08 2020). doi:10.1002/dac.4443

[40] Christoph Treude and Margaret-Anne Storey. 2011. Effective Communication of Software Development Knowledge through Community Portals. In *Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering* (Szeged, Hungary) *(ESEC/FSE '11).* Association for Computing Machinery, New York, NY, USA, 91–101. doi:10.1145/2025113.2025129

[41] Katja Tuma, Gul Calikli, and R. Scandariato. 2018. Threat Analysis of Software Systems: A Systematic Literature Review. *Journal of Systems and Software* 144 (06 2018). doi:10.1016/j.jss.2018.06.073

[42] T. Ucedavélez and M. M. Morana. 2015. *Risk Centric Threat Modeling: Process for Attack Simulation and Threat Analysis.* John Wiley & Sons.

[43] Prashant Vats, Manju Mandot, and Anjana Gosain. 2020. A Comprehensive Literature Review of Penetration Testing & Its Applications, In 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO). *ICRITO 2020 - IEEE 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions),* 674–680. doi:10.1109/ICRITO48877.2020.9197961

[44] Vasaka Visoottiviseth, Phuripat Akarasiriwong, Siravitch Chaiyasart, and Siravit Chotivatunyu. 2017. PENTOS: Penetration testing tool for Internet of Thing devices. In *TENCON 2017 - 2017 IEEE Region 10 Conference.* 2279–2284. doi:10.1109/TENCON.2017.8228241

[45] Ryan Williams, Emma McMahon, Sagar Samtani, and Mark Patton. 2017. Identifying vulnerabilities of consumer Internet of Things (IoT) devices: A scalable approach. In *2017 IEEE International Conference on Intelligence and Security Informatics (ISI).* 179–181. doi:10.1109/ISI.2017.8004904

[46] Wenjun Xiong and Lagerström Robert. 2019. Threat Modeling – A Systematic Literature Review. *Computers & Security* 84 (03 2019), 53–69. doi:10.1016/j.cose.2019.03.010

[47] Yuchen Yang, Longfei Wu, Guisheng Yin, Lijie Li, and Hongbin Zhao. 2017. A Survey on Security and Privacy Issues in Internet-of-Things. *IEEE Internet of Things Journal* 4 (2017). Issue 5. doi:10.1109/JIOT.2017.2694844

[48] Omerah Yousuf and Roohie Naaz Mir. 2019. A survey on the internet of things security: State-of-art, architecture, issues and countermeasures. *Information & Computer Security* (2019).

[49] Nan Zhang, Soteris Demetriou, Xianghang Mi, Wenrui Diao, Kan Yuan, Peiyuan Zong, Feng Qian, Xiao Feng Wang, Kai Chen, Yuan Tian, et al. 2017. Understanding iot security through the data crystal ball: Where we are now and where weare going to be. *Scanning Electron Microsc Meet at* (2017).