What do you assume? A Theory of Security-Related Assumptions

Sophie Corallo*
Thomas Weber*
Lars König*
Karlsruhe Institute of Technology
Karlsruhe, Baden-Württemberg, Germany

Kathrin Leonie Schmidt†
Frederik Reiche*
Anne Koziolek*
Karlsruhe Institute of Technology
Karlsruhe, Baden-Württemberg, Germany

ABSTRACT

Assumptions play a significant role in software engineering. Especially for security, implicit, inconsistent, or invalid assumptions on the system can have a high impact. Even though there are several approaches for managing assumptions in security engineering, most of them are highly specific for their domain and phase in software development. However, for holistic assumption management, a general understanding of security-related assumptions is needed. Funded on a Grounded Theory-based approach, including nine interviews with security researchers and a literature review of 53 scientific publications on assumptions, we propose a first definition of security-related assumptions.

CCS CONCEPTS

• **Security and privacy → Software security engineering**; • **Software and its engineering → Requirements analysis**.

KEYWORDS

assumption management, security assumptions

ACM Reference Format:
Sophie Corallo, Thomas Weber, Lars König, Kathrin Leonie Schmidt, Frederik Reiche, and Anne Koziolek. 2024. What do you assume? A Theory of Security-Related Assumptions. In *2024 IEEE/ACM 46th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion '24), April 14–20, 2024, Lisbon, Portugal.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3639478.3643070

1 INTRODUCTION

Assumptions play a significant role in various aspects of software engineering, as evident from prior research [9, 16, 18]. Regrettably, many assumptions are impromptu and left implicit. The repercussions of implicit, inconsistent, and invalid assumptions can be profound, causing requirement violations, miscommunication, and many system issues, including security vulnerabilities [16]. Many of these problems can be prevented by assumption management. Assumption management for software systems is the systematic development and maintenance of explicit assumptions in software systems. This covers identifying, describing, evaluating, maintaining, tracing, monitoring, reusing, and organizing assumptions [16].

In systems with elevated security standards, explicit assumption management becomes crucial. Failures in such systems, such as those used in cars, medical systems, or power supplies, can severely affect user privacy or safety. Security engineering faces numerous challenges with many issues arising from undocumented and implicit assumptions that give rise to vulnerabilities [17]. An example is the Equifax data breach from 2017 [8]. Systematic uncovering and management of the assumed anchors of a system's security could help with these challenges. Indicators can be found when investigating the relation of assumptions and threats [2, 10, 12, 13]. Therefore, an understanding of security-related assumptions, their inclusion in the software engineering process, and their relations to other artifacts is needed.

Even though *assumption* is a widely used term, it is often not well-defined [16]. In a systematic mapping study on assumptions and their management, Yang et al. categorize and describe several types of assumptions in software engineering. The most frequent types are *context*, *trust*, *architectural*, and *early architectural* assumptions. Especially, *trust assumption* is a widely but differently used term in security [5, 6, 14]. In general, a problem with the existing notions is that assumptions are only defined for specific contexts and software development phases. However, assumptions have a dynamic nature and have a flexible usage [13]. This requires the opportunity to define and refine assumptions regarding contexts, software development phases, and viewed artifacts. However, to our knowledge, no definition or concept addresses this problem.

In this work, we investigate assumptions that impact directly the security of a system, so-called security-related assumptions, and raise the research question: *"What is a security-related assumption, and how is it linked to other software development concepts?"*

2 APPROACH

To answer our research question, we decided on a Constructivist Grounded Theory-based approach [3]. The main goal of Grounded Theory is to inductively generate theory from data by immediate and continuous data analysis. Everything can be used as data, e.g., interviews, literature, or media. To build the theory, exemplary assumptions would work the best. However, publicly available, explicit assumptions on security are rare to find. Therefore, we conducted structured interviews with nine security researchers from a German university. In each interview, we presented the Dutch smart grid as a common scenario [11]. The interviewees

*firstname.lastname@kit.edu
†kathrin.schmidt@student.kit.edu

ICSE-Companion '24, April 14–20, 2024, Lisbon, Portugal

ACM ISBN 979-8-4007-0502-1/24/04
https://doi.org/10.1145/3639478.3643070

were then asked to reflect on assumptions and requirements that occurred during their work as well as their impact on the security of the system. After every two to three interviews, we coded and analyzed the gathered data to adapt our view on security-related assumptions.

We increased the amount of analyzable data by an extensive literature review of 53 scientific publications. Due to an observed similarity between assumptions and requirements, we clustered the literature regarding requirements engineering activities, e.g., elicitation and verification and highlighted activities, e.g., threat models, code, and design, to support the coding process.

Further details and information can be found on https://secdragon-dev.github.io/WhatDoYouAssume/.

3 RESULTS

In the conducted interviews, we observed several commonalities and discrimination criteria: Security-related assumptions seem to have a probability of violation that seems to be useful for security estimations. Moreover, they have consequences, e.g., legal, conceptual, and physical aspects, as well as an impact on the system. Security-related assumptions are specific for a certain perspective, described by a subject and the viewed information, e.g., an UML class diagram. Thereby, an assumption targets the outer world, the context, or the inner of a system. In both of the latter cases, assumptions influence decisions on the system. Independent of the view or the target space, assumptions, like requirements, can be formulated on different levels of abstraction. Thereby, assumptions can have sub-assumptions that describe the first on a more fine-grained level.

In our literature review, most work aligned with our picture of assumptions, even though some extended our view. The most remarkable change was regarding the relation between requirements and assumptions. Whereas some works do not distinguish between assumptions and requirements [1, 15], some make assumptions on design decisions [4].

In conclusion of our observations, we provide the following definition. As it relies on limited data, it is formulated as a theory.

Theory. *A security-related assumption is a statement that:*

(1) is taken for true,
(2) can be violated,
(3) is formulated from a certain perspective [1]

Security-related assumptions seem to have several properties:

(4) They can be interpreted differently from different perspectives,
(5) they can be refined horizontally (on the same abstraction level), as well as vertically (across different abstraction levels),
(6) they have a probability to be violated, they have a risk for the security of the system, as well as consequences of their violation,
(7) they occur when a design decision is made,
(8) they transform into a requirement if they are supposed to be realized by a design decision.

To see whether the definition and properties would help to explicate security-related assumptions, we conducted a applicability study based on a worked example. For this, we used risk assessment documents of the *Corona Warn App* [7], a contact tracing system commissioned by the German government. However, for further validation, we plan to conduct a questionnaire and ask software engineers globally, whether they do agree with our theory or not. We also want to investigate whether our definition of security-related assumptions also fits assumptions in general and whether any hard discriminators between assumptions and requirements deny the reuse of requirement-based approaches for assumptions. Otherwise, the reuse of established approaches could improve and extend the landscape in assumption management.

[1] A perspective characterizes a view that a certain subject has based on a certain view type, e.g., a software architecture or uml class model

4 ACKNOWLEDGMENTS

This work was supported by funding from the topic Engineering Secure Systems of the HGF and by KASTEL Security Research Labs and the pilot program Core Informatics at KIT (KiKIT) of the Helmholtz Association (HGF).

REFERENCES

[1] Brandon Broadnax, Pascal Birnstill, Jörn Müller-Quade, and Jürgen Beyerer. 2016. Eliciting and refining requirements for comprehensible security. In *Security Research Conference : 11th Future Security, Berlin, September 13-14, 2016. Ed.: O. Ambacher*. Fraunhofer Verlag, 323–329. 46.12.03; LK 01.
[2] Prabuddha Chakraborty, Jonathan Cruz, Christopher Posada, Sandip Ray, and Swarup Bhunia. 2022. HASTE: Software Security Analysis for Timing Attacks on Clear Hardware Assumption. *IEEE Embedded Systems Letters* 14, 2 (2022).
[3] Kathy Charmaz. 2014. *Constructing grounded theory* (2 ed.). SAGE Publications, London, England.
[4] Hassan El-Hadary and Sherif El-Kassas. 2014. Capturing security requirements for software systems. *Journal of Advanced Research* 5, 4 (2014), 463–472. Cyber Security.
[5] C.B. Haley, R.C. Laney, J.D. Moffett, and B. Nuseibeh. 2004. The effect of trust assumptions on the elaboration of security requirements. In *Proceedings. 12th IEEE International Requirements Engineering Conference, 2004.* 102–111. https://doi.org/10.1109/ICRE.2004.1335668
[6] Charles B Haley, Robin C Laney, JD Moffett, and B Nuseibeh. 2003. Using Trust Assumptions in Security Requirements Engineering. In *Second Internal iTrust Workshop On Trust Management In Dynamic Open Systems*. Citeseer, 15–17.
[7] Robert Koch Institute. 2020. Open-Source Project Corona-Warn-App. https://www.coronawarn.app/en/. Accessed: 2023-07-25.
[8] Ilya Kabanov and Stuart Madnick. 2020. A Systematic Study of the Control Failures in the Equifax Cybersecurity Incident. (2020).
[9] Md Abdullah Al Mamun and Jörgen Hansson. 2011. Review and challenges of assumptions in software development. In *Second Analytic Virtual Integration of Cyber-Physical Systems Workshop (AVICPS)*.
[10] V. Page, M. Dixon, and I. Choudhury. 2007. Security risk mitigation for information systems. *BT Technology Journal* 25, 1 (01 Jan 2007), 118–127.
[11] P. Van Aubel and E. Poll. 2019. Smart metering in the Netherlands: What, how, and why. *International Journal of Electrical Power & Energy Systems* 109 (2019).
[12] Dimitri Van Landuyt and Wouter Joosen. 2020. A Descriptive Study of Assumptions Made in LINDDUN Privacy Threat Elicitation. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing* (Brno, Czech Republic) *(SAC '20)*. Association for Computing Machinery, New York, NY, USA, 1280–1287.
[13] Dimitri Van Landuyt and Wouter Joosen. 2021. A descriptive study of assumptions in STRIDE security threat modeling. *Software and Systems Modeling* (2021).
[14] John Viega, Tadayoshi Kohno, and Bruce Potter. 2001. Trust (and Mistrust) in Secure Applications. *Commun. ACM* 44, 2 (feb 2001), 31–36.
[15] Xiaowei Wang, John Mylopoulos, Giancarlo Guizzardi, and Nicola Guarino. 2016. How software changes the world: The role of assumptions. In *2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS)*. 1–12. https://doi.org/10.1109/RCIS.2016.7549327
[16] Chen Yang, Peng Liang, and Paris Avgeriou. 2018. Assumptions and their management in software development: A systematic mapping study. *Information and Software Technology* 94 (2018), 82–110.
[17] Chen Yang, Peng Liang, Paris Avgeriou, Ulf Eliasson, Rogardt Heldal, and Patrizio Pelliccione. 2017. Architectural Assumptions and Their Management in Industry – An Exploratory Study. In *Software Architecture*, Antónia Lopes and Rogério de Lemos (Eds.). Springer International Publishing, Cham, 191–207.
[18] Pamela Zave and Michael Jackson. 1997. Four Dark Corners of Requirements Engineering. *ACM Trans. Softw. Eng. Methodol.* 6, 1 (jan 1997), 1–30.