Threat Modeling Smart Metering Gateways

Armin Lunkeit
OpenLimit SignCubes GmbH
Berlin, Germany
armin.lunkeit@openlimit.com

Tobias Voß
softScheck GmbH
Sankt Augustin, Germany
tobias.voss@softscheck.com

Hartmut Pohl
softScheck GmbH
Sankt Augustin, Germany
hartmut.pohl@softscheck.com

***Abstract*— In the traditional Software Development Lifecycle (SDL), measures aimed at increasing the security level are generally implemented only shortly prior to shipping, and frequently even after the system has been delivered. Because around half of all security-related bugs (vulnerabilities) can be traced back to design bugs, security measures will, however, have to be implemented shortly before or during the design. Costs of removing vulnerabilities are also lowest at this development phase.**

Threat Modeling supports the methodological development of a trustworthy system design and architecture at the design phase of software development.

In this article, the Threat Modeling process is first applied to a Smart Metering Gateway as component of Smart Grid infrastructure. Smart Metering Gateways (SMGW) represent communication between the prosumer with his consuming and generating devices and the Smart Grid with distribution network operators, meter operators, metering service providers, suppliers etc.; they represent the "security core" of the Smart Grid and thus have a significant influence on public perceptions of the entire Smart Grid Infrastructure.

Keywords—fuzzing, smart metering gateway, smart grid infrastructure, smart metering, testing, threat modeling, security, vulnerability

I. OVERVIEW

Wherever energy (electricity, gas, water, heat) needs to be metered for billing with energy suppliers appropriate meters needs to be installed. In the current practice these meters must be read regularly on site, this is an enormous logistical and costly process for energy suppliers.

Added to this there is an increasing decentralized generation of energy. Earlier enterprises or households were pure consumers, energy was provided exclusively by central generators like nuclear or coal-fired power plants. Today, consumers are increasingly producers of energy. Households rely on electricity by photovoltaic, wind power or combined heat and power generation. Excess electricity can be sold into the public electricity grid and thus made available to other consumers. The resulting requirements cannot meet conventional meters and power grids. Smart meters (Smart Metering Gateways) and electricity networks supplemented by information technology (Smart Grids) are required. The SMGW is an ideal entry point for attackers into the power management of private households or even in core areas of the smart grid.

The paper is organized as follows: Section II introduces the formalized security requirements of the SMGW and further security testing procedures. In this paper we will focus on the design phase with threat modeling. In section III to V we explain the threat modeling process with SMGW specific details and examples. The paper closes with conclusions and related work.

II. BASICS

To meet the high security requirements of the SMGW the German Federal Office for Information Security (BSI) has created a Protection Profile (PP) and a Technical Guideline (TR) concerning the SMGW. "Technical Guideline BSI TR-03109" [2] formulates security requirements for the "Smart Metering Gateway (SMGW)" component of the Smart Grid infrastructure. In addition to these security requirements, there will also be need for an evaluation of the SMGW in relation to the "Protection Profile for the Gateway of a Smart Metering System" (Common Criteria) [1].

The common criteria require Threat Modeling in the design phase. The goal here is to identify potential attack paths and the resulting threats and vulnerabilities. Without vulnerabilities [12] no successful attacks are possible and without a rewarding goal (asset) an attack would not be worthwhile.

In addition to the systematic [11], tool-supported process of Threat Modeling for identifying vulnerabilities already at the design phase, the following test steps have to be worked through before a product can be released:

- Static Source Code Analysis: formal testing of the source code.

- Penetration Testing: simulated attacks etc. for testing for known vulnerabilities.
- Dynamic Analysis - Fuzzing: test of executable compiled files – no source code necessary.
- Explorative testing and manual code auditing.
- Backdoor Detection: analysis and identification of covert functions – non-documented, covered and hidden functions.

III. PREPARATORY TESTS

The creation of the Threat Model commences with the inspection and analysis of documentation (if available) – in particular the safety design or the program Data Flow Diagrams (DFD). From this information, the information essential for the Threat Model is extracted. This provides the current state of development, which is then used for the Threat Model.

A. Entry Points

The Entry points are (input) interfaces through which an attacker can access the SMGW. An entry point is to be considered not only as a single physical interface. A single physical entry point can also be distributed among several logical entry points assigned to different services. There follow three examples of entry points of the SMGW:

- Via the Home Area Network (HAN) interface, the SMGW communicates with the Controllable Local Systems (CLS) of the connection.
- The Wide Area Network (WAN) interface allows the Gateway Administrator to send tariff and configuration data or soft- and firmware updates to the SMGW. Even if the Administrator is regarded as trustworthy, communication between SMGW and Gateway Administrator must be confidential, integer and authenticated.
- The Local Metrological Network (LMN) interface is reserved exclusively for the meters. Through this interface metered values gets into the SMGW. The metered values can originate from external-powered or battery-powered meters. One challenge here is the heterogeneous technological basis of the meters. The encryption method used depends on the used meter.

B. Assets

Assets represent the resources of the SMGW to be protected. No assets, no threats: an attacker will always try to obtain access to the assets of the SMGW. Unless adequate security precautions are in place, an attacker could, for example, read or write (manipulate or delete) the customer log, system log or authentication data.

- The customer log contains data of accounting relevance for an end-user and all transactions of the SMGW such as the transmission of meter values or activities of the SMGW administrator. The SMGW is client-compatible, and thus several customer logs can exist for different end-users [3].
- Authentication data – All external entities that make up a communication link with the SMGW must be authenticated by the SMGW.
- Metered values – these are sent from a meter to the SMGW or remain for further processing in the SMGW.
- Timestamp – serves e.g. as basis for tariffing and requires regular synchronization with an external and reliable time source located on the WAN.

C. Trust Level

An end-user has different authorizations in relation to the SMGW than an administrator. A Trust Level accordingly characterizes an external entity (user/technical component) from the point of view of authentication and the access rights granted to it [4]. When the Trust Level has been identified, it can be assigned to Entry Points and Assets. There follow typical examples of entry points of the SMGW:

- CLS Device – A CLS Device sends energy feed-ins from the HAN (e.g. photovoltaic, wind power, …) to the SMGW. In addition the SMGW can request energy feed-ins directly from a CLS Device. The CLS Device must have access to the SMGW built-in proxy service for sending data to an External Market Actor.
- Display Unit – The display unit has access to the customer log. This gives the customer insight into its power consumption and its customer specific transactions from the SMGW.
- External Market Actor – The role of the External Market Actor is subdivided into the necessary custom authorization profiles. An energy supplier can request customers' energy consumption or a Gateway Administrator can modify the configuration of the SMGW and so on.
- Meter – The metered values must be sent to the SMGW, the SMGW can also request metered values directly from an external meter at configurable intervals.

IV. MODELING

A. Scenarios

Taking account of the information hitherto collected, the scope of the Threat Model is established on the basis of scenarios. The Threat Model can also be used including the scenarios for Penetration Tests or for monitoring subsequent implementation [12]. There follow a few examples for scenarios:

- A CLS Device creates a connection to an external market participant in the WAN. The energy provider can thus, for example, set off the feed-in value of a CLS Device against the consumption bill of the end customer.
- An External Market Actor retrieves consumption data from a customer.

- The SMGW receives from an external time server located on the WAN a current timestamp.

B. Data Flow Diagram

The information collected hitherto is implemented in a DFD. The modeling of the entire system takes place here at several levels (Level 0, Level 1, ..., Level n+1) and can be performed by individual technical components down to deeper levels in order to reveal function calls [12]. The depth of the modeling must be determined in advance, to avoid getting lost in details. A DFD enables diagrammatic presentation of the attack paths to the SMGW and thus forms a basis for identifying the threats. The DFD also provides a basis for discussion for the parties involved in developing the SMGW.

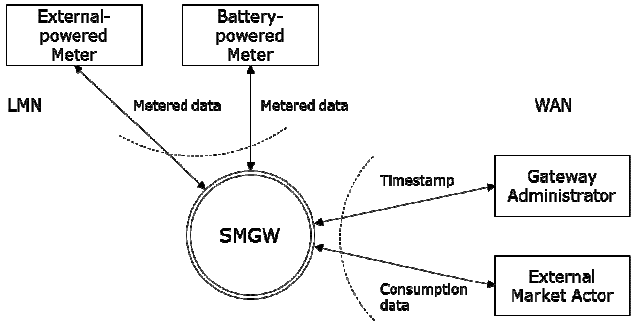

Fig. 1. Section of a Data Flow Diagram (Level 0)

In Figure 1, the SMGW is shown as a central switch between LMN and WAN. The rectangular blocks represent external entities, the arrows represent data flows between the external entities and the SMGW. Each data flow passes through a trust level, marked by the dotted line. The SMGW is shown as black box and is resolved in the next level in the individual functions of the SMGW.

V. THREATS

Threats are systematically collected from the compiled information and the DFD created. A threat can never be removed entirely. As long as the functionality relating to the threat and the relevant asset is part of the SMGW, a threat can only be mitigated. Every threat must be examined from the point of view of potential mitigation.

A. Classification of threats with STRIDE

STRIDE is a system for classifying the identified threats and stands for the following terms and classifications [8]:

- Spoofing Identity
- Tampering with data
- Repudiation (non-assignment of events)
- Information Disclosure
- Denial of Service (restriction on availability)
- Elevation of Privilege (extension of authorizations)

Following the STRIDE classification of threats, they can be subject to a first prioritization by agreement with the designers of the SMGW. For example, the threats of the category Elevation of Privilege could be prioritized for further refinement of the Threat Model.

VI. THREAT RISK ASSESSMENT: ASSESSING DREAD

DREAD is used for numerically determining the risk of a threat according to five independent factors. Each factor receives a predefined weighting. Experience shows that the most simple weighting (e.g. 1 to 3) can be used efficiently. The total of the five DREAD factors forms the risk assessment of a threat according to DREAD. Assessment of the threat is therefore undertaken exclusively on the basis of the identified threats, not in relation to the corresponding mitigation. The five DREAD assessment factors [6] are presented below with a brief explanation:

- Damage Potential

  What damage is caused by the threat being exploited?

- Reproducibility

  To what degree can exploitation of the threat be reproduced?

- Exploitability

  How high is the cost of exploiting the threat?

- Affected Users

  How many users are affected by the threat?

- Discoverability

  How easily can the threat be identified?

After the threats have been evaluated, they can be assigned to different risk classes (e.g. from low to high) [7]. This will enable further prioritizing of the threats from the point of view of development of a more secure and robust design.

A. Vulnerabilities

A non-mitigated threat represents a vulnerability. A vulnerability can be presented in a Threat Tree as (complete and unmitigated) Attack Path. A successful attack is the exploitation of an identified vulnerability and an attacker obtains access to an asset in SMGW deserving protection.

The threats are therefore to be mitigated by the SMGW developer as much as possible. This ensures that a threat can no longer be exploited by currently available means and methods.

A vulnerability can be described graphically in a Threat Tree. This provides not only a clear overview but also facilitates communication with different persons, perhaps less technically oriented, involved in the development of systems.

The threat to be analyzed is located in the roots of the Threat Tree and can be attacked from the leaves out. The leaves contain mitigated or non-mitigated threats. An Attack Path that can be traced through non-mitigated threats down to the roots represents a vulnerability.

VII. Conclusion

Threat Modeling makes a decisive contribution to the development process of a SMGW in terms of early identification and recognition of vulnerabilities. This statement is proven by the experience gained in many projects. In a random sampling of 40 projects, for example, no less than 156 vulnerabilities are identified. If a manufacturer can manage to remove a majority of the identified vulnerabilities, this will increase the level of security and decrease anticipated costs from troubleshooting in developed hardware and software components.

After creation of a Threat Model, a validation must be undertaken of all steps carried out. Only in this way can remaining inconsistencies in the Threat Model be recognized and removed.

Threat Modeling is an iterative process. On completion of the validation, the security gaps in the current design should be closed. New threats for the system can, however, arise when existing security gaps are removed, and for this reason a repeat audit of the design will be necessary. If the Threat Model is carefully configured, repeat audits can be performed more quickly than the first creation of a Threat Model.

When the first executable version of the SMGW software becomes available, further audits going beyond Threat Modeling should be carried out by means of the processes Dynamic Analysis: Fuzzing and Static Source Code Analysis.

VIII. Related Work

Schieferdecker et al. [9] described techniques and related models for security vulnerability testing at a higher level of abstraction that could also be applied to Threat Modeling.

Wang et. al. [13] mentioned in the paper a threat driven approach to MBST. In this approach, Unified Modeling Language (UML) sequence diagrams are used to specify a threat model. The threat model is then used as a basis for code instrumentation. Finally, the instrumented code is recompiled and executed using randomly generated test cases. If an execution trace matches a trace described by the threat model, security violations are reported and actions should be taken to mitigate the threat in the system. In addition to this Jürjens [5] developed UMLsec, a security profile for the UML.

With those approaches a connection could be made from abstract threat modeling to code implementation and automated security vulnerability testing.

In a next step the code implementation should be tested with Dynamic Analysis - Fuzzing. Schneider described [10] techniques for generating fuzz test cases at runtime instead of before execution. Results from already executed test cases are used for further test cases.

References

[1] Bundesamt für Sicherheit in der Informationstechnik (Ed.): Protection Profile for the Gateway of a Smart Metering System, Version 01.01.01 (final draft), Bonn 2011 https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/SmartMeter/PP-SmartMeter.pdf?__blob=publicationFile, August 2012.

[2] Bundesamt für Sicherheit in der Informationstechnik (Ed.): Technische Richtlinie BSI-TR-03109-1, Version 0.50, Bonn 25.05.2012, https://www.bsi.bund.de/ContentBSI/Publikationen/TechnischeRichtlinien/tr03109/index_htm.html, August 2012.

[3] Bundesamt für Sicherheit in der Informationstechnik (Ed.): Technische Richtlinie TR-03109-3 - Kryptographische Vorgaben, Version 0.50, Bonn 25.05.2012, https://www.bsi.bund.de/ContentBSI/Publikationen/TechnischeRichtlinien/tr03109/index_htm.html, August 2012.

[4] Homeland Security (Ed.): Configuring and Managing Remote Access for Industrial Control Systems, Washington, November 2010. http://www.us-cert.gov/control_systems/pdf/Recommended_Practice-Remote_Access_1-6-2011.pdf, S. 14, August 2012.

[5] Jürjens, J.: UMLsec: Extending UML for Secure Systems Development, 2002, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.19.2850, April 2013.

[6] Leblanc, D.: DREADful. Redmond 2007, http://blogs.msdn.com/b/david_leblanc/archive/2007/08/13/dreadful.aspx, August 2007.

[7] Microsoft (Ed.): Improving Web Application Security: Threats and Countermeasures (Step 6: Rate the Threats), Redmond June 2003 http://msdn.microsoft.com/en-us/library/ff648644.aspx#c03618429_011, August 2012.

[8] Microsoft (Ed.): The STRIDE Threat Model, Redmond 2005, http://msdn.microsoft.com/en-us/library/ee823878%28v=cs.20%29.aspx, August 2012.

[9] Schieferdecker, I., Grossmann, J., Schneider, M.: Model-based Security Testing. http://arxiv.org/abs/1202.6118v1, Februar 2012.

[10] Schneider, M., Grossmann, J., Schieferdecker, I., Pietschker, A.: Online Model-Based Behavioral Fuzzing. http://arxiv.org/abs/1202.6118v1, Februar 2012.

[11] softScheck (Ed.): Threat Modeling, Sankt Augustin 2012 http://www.softscheck.com/consulting_threat_modeling.html, August 2012.

[12] Swiderski, F., Snyder W.: Threat Modeling. S. xi, 15, 70, 90, June 2004.

[13] Wang L., Wong E., Xu D.: A Threat Model Driven Approach for Security Testing. In: Proceedings of the Third International Workshop on Software Engineering for Secure Systems, SESS '07, IEEE Computer Society, Washington, DC, USA, pp. 10–. Available at http://dx.doi.org/10.1109/SESS.2007.2.

Armin Lunkeit was born in Germany in 1978. In his capacity as Chief Development Officer he has since December 2007 been member of the management of the OpenLimit Group.

He studied Microsystems Technology at the university of applied science Fachhochschule für Technik und Wirtschaft in Berlin, which he completed in 2002, being awarded the graduate engineer title *Dipl. Ing (FH)*. Armin Lunkeit commenced software development in 2000.

He was employed as developer at Kithara GmbH. In June 2003, Armin Lunkeit was employed by OpenLimit SignCubes GmbH in the area of product development, before taking up his current position.

OpenLimit SignCubes GmbH
Saarbrücker Str. 38a, 10405 Berlin
Tel.: +49 30 400 3510 10
Fax: +49 30 400 3510 41
armin.lunkeit@openlimit.com

Tobias Voß is a qualified IT specialist in system integration. He studied Business Information Systems, Bachelor of Science (B.Sc.) with the main focus on information security and business application systems.

Freelance work in the area of planning and operation of web-content management systems (WCMS) and groupware servers.

Senior Consultant in softScheck GmbH for IT security. Projects in the area of Security Development Lifecycle (SDL): Threat Modeling (Requirements/Design), Static Source Code Analysis (Implementation) and Dynamic Analysis: Fuzzing (Verification/Testing).

softScheck GmbH
Office: Bonner Str. 108, 53757 Sankt Augustin
Tel.: +49 (2241) 255 43 – 0
Fax: +49 (2241) 255 43 – 29
tobias.voss@softscheck.com

Prof. Dr. Hartmut Pohl Software Security: Software Development Lifecycle: Security by Design, Threat Modeling, Static Source Code Analysis, Penetration Testing, Manual Auditing, Dynamic Analysis: Fuzz-Testing in the following sectors:

Commercial Services: Automotive, Chemical/Pharmaceutical Industry, Nutrition/Beverages, Security, Software - Financial Services: Saving Banks, Banking Houses - Public Services: Governmental Departments, Defense, Transports/Logistics, - Infrastructure Services: Utility Companies, Smart Grid (Electricity, Oil, Gas), M2M, Cloud Computing.

Managing Partner softScheck GmbH
Office: Bonner Str. 108, 53757 Sankt Augustin
Tel.: +49 (2241) 255 43 – 0
Fax: +49 (2241) 255 43 – 29
hartmut.pohl@softscheck.com