

Automotive Security Testing—The Digital Crash Test

Stephanie Bayer, Thomas Enderle, Dennis-Kengo Oka
and Marko Wolf

Abstract Modern vehicles consist of many interconnected, software-based IT components which are tested very carefully for correct functional behavior to avoid safety problems, e.g. the brakes suddenly stop working. However, in contrast to safety testing systematic testing against potential security gaps is not yet a common procedure within the automotive domain. This however could eventually enable a malicious entity to be able to attack a safety-critical IT component or even the whole vehicle. Several real-world demonstrations have already shown that this risk is not only academic theory [1]. Facing this challenge, the paper at hand will first introduce some potential automotive security attacks and some important automotive security threats. It then explains in more detail how to identify and evaluate potential security threats for automotive IT components based on theoretical security analyses and practical security testing.

Keywords Automotive security analysis · Automotive security testing · Automotive penetration testing

S. Bayer (✉) · T. Enderle · M. Wolf
ESCRYPT GmbH, Leopoldstraße 244, 80807 Munich, Germany
e-mail: stephanie.bayer@escrypt.com

T. Enderle
e-mail: thomas.enderle@escrypt.com

M. Wolf
e-mail: marko.wolf@escrypt.com

D.-K. Oka
ETAS K.K., Queens Tower C-17F, 2-3-5, Minatomirai, Nishi-Ku,
Yokohama 220-6217, Japan
e-mail: dennis-kengo.oka@etas.com

1 Introduction

Suddenly car drivers all over the world witness spooky behavior of their Internet-enabled car infotainment units over the past few days. Out of the blue their navigation system jumps to another route, the unit calls service numbers on its own, or the display shows skulls and laughing, white masks. A quick analysis by security experts shows that the reason behind this behavior is a critical security breach of the Internet-enabled GSM/LTE interface that enables unauthorized persons to access the software of the vehicle infotainment unit. But, how is it possible that this vulnerability had been missed, as the infotainment unit passed numerous tests before going into series production? The answer is quite simple; even though there were several tests focusing on the functionality and safety, a systematic security evaluation containing theoretical analysis and practical tests had not been accomplished.

Luckily, this is only a potential scenario (yet) and not a real case but the above scenario clearly exposes a critical gap in automotive IT testing, which is not yet covered by existing functional testing procedures that are already well-established and have been conducted for several decades. However, in contrast to functional testing, the systematic evaluation of automotive IT components regarding their IT security is still in a very early stage. At the same time there is a strong need for security testing as a part of the automotive engineering procedure, not only due to scenarios like the introductory example but also as a result of corresponding automotive IT security research activities [2, 3], and increasing demands made by public authorities [4].

The paper is structured as follows. First we discuss recent automotive IT security threats ranging from odometer manipulations up to remote controlling of the steering system. Then, we give an overview and a short introduction to the different aspects of automotive security evaluations. We discuss theoretical security analyses and explain practical security testing, especially regarding automotive onboard IT components.

2 Automotive Attack Motivations and Threats

This section presents an overview of potential automotive security threats. To understand the types of threats, one must first have a basic understanding of various automotive functionalities which can be targeted. Obviously there are direct security threats on driving safety such as manipulating the steering wheel or brakes, and indirect security threats such as distracting the driver by triggering odd vehicle behavior. Security researchers have successfully demonstrated that it is already possible to engage or disable the brakes or manipulate the steering wheel [1–3, 5] by maliciously injecting the relevant messages on the vehicle CAN bus. Indirect

safety issues are possible by injecting CAN messages to, for example, disable wipers or turning off the headlights when it is raining and dark outside.

Another type of security threat is targeting authoritative functionalities in the vehicle. For instance, the odometer logs the traveled distance and, when selling a used car, it is attractive for an attacker to reduce the value of the odometer to increase the value of the car. In Germany, according to police investigations, around 2 million cars are subject to odometer manipulation per year with an average loss per vehicle of around 3000 € resulting in total losses of around 6 billion € per year [6]. Furthermore, critical data are stored in the ECUs such as crash data, data for insurances, or warranty indicators. Such data are also very attractive for malicious manipulations. For example, data such as vehicle speed, seat belt status, brake pedal position etc. are typically recorded in the seconds before a crash. A driver who has been involved in an accident could be motivated to change the recorded data to indicate that the brakes were applied when they really were not.

Moreover, since vehicles are becoming ubiquitously interconnected, private data such as vehicle location, credentials to online services, or mobile payment data becomes increasingly stored in the vehicle as well. Attackers may be interested in stealing such data and misuse them directly or as a stepping stone to launch further attacks, for instance, attacking an online service by stealing the respective credentials stored in the vehicle. These private data could be extracted wirelessly by exploiting security vulnerabilities in services provided by communication interfaces such as Bluetooth and Wi-Fi or through physical access to the OBD port, a USB port or the ECU itself.

Another type of threat is theft of vehicles or valuable vehicle components such as airbags or head units, e.g. by abusing diagnostics commands to reprogram a new key. This functionality is typically used by workshop dealers when replacing a lost key, but can also be exploited by attackers to program a “thief key” for the vehicle that they are stealing [7]. There are other cases where attackers are able to send control messages to a vehicle to disable the alarm and unlock the doors, resulting in attackers being able to gain physical access to the interior of a vehicle [8].

3 Automotive Security Evaluations

As discussed in the previous section, modern vehicles can be open to various security risks. By applying in-depth security evaluations for an automotive IT system, for instance an ECU, potential security weaknesses can be identified and countered before an attacker can exploit this weakness in the field and causing real financial or even safety damages. The earlier such a security evaluation is done within the developer cycle the less costly and time-consuming it is and such security weaknesses can be found and be closed effectively.

Automotive security evaluations can be done in theory as well as in practice. Theoretical security evaluation can (and should) be done during virtually all steps of the automotive development cycle, ideally already from the very beginning,

when only a description of the vehicular IT system is available. Subsequent security evaluations for the next product development iterations can then be done very efficiently based on the results of the previous evaluation. In fact, the need for theoretical and practical security evaluations does not end with series production of the corresponding IT system. Even in field the IT component might need a security re-evaluation (and eventually also new countermeasures) due to ongoing development of new attacks or new results from security research.

Practical security testing, of course, can only be conducted on an implementation of the target system, for instance with a first prototype.

It is important to note that security evaluation can support but not replace mandatory security protection measures such as security by design or security engineering.

4 Theoretical Automotive Security Analyses

Theoretical security analyses are becoming more and more common in the automotive context [9] and are applied to identify and understand the security weaknesses of an automotive IT system based on a paper-based evaluation of the corresponding system specifications and documentations. Depending on the level of scrutiny and the documents available, we differentiate between a more high-level *design analysis* and a more in-depth *threat and risk analysis*.

To conduct a **design analysis** of an automotive system, only a theoretical description of the system is needed. Depending on the level of detail of these descriptions, e.g. high-level protocol descriptions up to explicit specifications, the depth and accuracy of the analysis vary. The design analysis can identify systematic flaws in the system even in an early state in the development since high level descriptions can be adequate for a design analysis. Secondly, the results can establish trust in the soundness of the system's architecture. To achieve these goals, the documents are inspected for potential attack points, for instance, weak cryptographic algorithms, insufficient key lengths, or possible attacks due to bad interaction of different standard protocols.

To categorize the identified vulnerabilities further and to detect the most important flaws of the system that need to be fixed first, a **threat and risk analysis** can be applied to the system. Starting from the documents available, the system is analyzed and possible attacks identified similar to the design analysis. Additionally, the difficulty of the corresponding attack procedure is rated for each of the attacks identified. The rating takes into account amongst others the required time, the needed expertise of the attacker, the equipment, and the needed access level. Furthermore, the potential damage of a successful attack is estimated in terms of safety, operational, and financial impacts. Both values, the attack difficulty and the potential attack damage, result in an overall risk for a certain attack. Security vulnerabilities that result in attacks with a high risk are then critical candidates that should be fixed first.

Nonetheless, theoretical security analyses can neither find any implementation flaws or deviations of the implementation from the specification, nor detect vulnerabilities that are part of insufficiently documented specifications or flaws hidden in supplied components from third parties. To guard the system against such implementation issues, secure software development measures should be applied to the whole vehicular development process [10, 11]. But especially practical security testing, as described in the following section, can be used to identify possible vulnerabilities and cover the gap.

5 Practical Automotive Security Testing

Practical security testing can find implementation errors that could be exploited by an outside attacker, but also unspecified functionality and discrepancies to the specifications. Therefore, a thorough practical security test helps to establish trust in the soundness of the implementation. Furthermore, practical security tests help to estimate the actual difficulty of an attack against the target system. In general, practical security testing consists of at least four different steps as described in the following paragraph.

In the first step, **functional security testing**, tests all security-related functions inside the test system for correct behavior and robustness. This step is similar to general functional testing but with focus on security functionality. A careful execution of this test can find implementation errors, discrepancies to the specification, and especially unspecified functionality that all might result in a potential security weakness. The next step, **vulnerability scanning**, tests the system for already known common security vulnerabilities, for instance, known security exploits or (security) configurations with known weaknesses. **Fuzzing** goes even further and tries to find new vulnerabilities of an implementation by sending systematically malformed input to the target system to check for unknown, potentially security-critical system behavior. To test the security of the whole system, i.e. software and hardware, highly individual **penetration tests** can be applied in a last step. During a penetration test a “smart human tester” tries to exploit all the vulnerabilities which were found in the earlier steps in a “sophisticated way” based on many years of “hacking” experience with the aim to change the behavior of the target system.

For all the approaches the tester needs to have access to the actual software and hardware of the target system. Furthermore, for functional testing also the specification of the system is needed, and for all other methods supporting software, for instance remaining bus simulation, may be needed to run the hardware device. Moreover, special testing hardware and software is needed, for example, JTAG-debuggers or special signal generators. However, practical security testing, especially fuzzing and penetration testing, cannot give any assertion on completeness. Depending on the time and resources it is possible to miss larger

systematic flaws. Hence, practical security testing cannot replace theoretical security analyses and should always be complemented by a theoretical analysis to identify possible attack paths.

After presenting a first overview about practical security testing, the following subsection gives some further details about practical security testing in the automotive domain. Even though practical security testing is still relatively new to the automotive world, there is a strong need to establish a testing process [12].

5.1 Functional Automotive Security Testing

Functional automotive security testing ensures the general compliance to specifications and standards of the implemented security functionality, for instance, encryption algorithms, authentication protocols, of a vehicular IT system. However, the algorithms are not only tested for correct behavior according to the specification but also for robustness. Furthermore, performance of (often computationally intense) security algorithms is tested to identify potential bottle necks that might affect the overall security performance. As a result, functional security testing ensures dependable security functionality and that a functional weakness does not create any exploitable security threats.

In many cases, standard implementations such as OpenSSL [13] are not suitable for use in the automotive domain due to various constraints, and therefore, a much wider spectrum of cryptographic and security relevant implementations are in use. Performance or size limitations play a role here but also safety standards such as MISRA-C [14] must be fulfilled. Furthermore, a wide range of automotive specific security protocols are in use, such as secure flash algorithms or secure communication, secure OBD, theft protection, and upcoming vehicle-to-x (V2X) communication. It is vital that those security implementations are subject to thorough functional security testing.

Functional security is usually achieved by testing official test vectors (if available) and running lengthy tests against reference (if available) or independent implementations. Many cryptographic algorithms contain specific corner cases which cannot be caught by this kind of testing but could lead to security vulnerabilities, for example subtle flaws in numeric implementations that trigger only in one out of 4 billion random cases. Many modern cryptography and security implementations further rely on secure random number generators. In order to gain trust into the security of such a random number source, extensive statistical testing is required. Finally, in the highly performance- and cost-sensitive automotive environment, performance testing can help with correct dimensioning of hardware or enable an optimal choice of security algorithms and parameters.

5.2 *Automotive Vulnerability Scans*

Vulnerability scans are used to examine all relevant applications, source codes, networks, and backend infrastructures of an automotive system for known security weaknesses from a continuously updated database of known automotive security vulnerabilities.

There are numerous different variations of vulnerability scanning. Firstly, the code of the software/firmware running on the system can be scanned, identifying, for example, buffer overflows and heap overflows by using static and dynamic analyses of the source code and compiler settings. This must be done on source level or on binary level. Secondly, the system can be scanned for open ports and interfaces and also for available services running on the interfaces. In automotive systems, this encompasses classical IT interfaces such as IP communication on Ethernet, Wi-Fi, or cellular internet. Scanning is especially valuable due to the fact that a whole range of operating systems, network stacks, applications, and libraries is re-used, where a large base of vulnerabilities is already known and can be tested automatically, as done in OpenVAS [15]. This concerns reconnaissance port scans, as well as deep scans of specific vulnerabilities.

The automotive bus systems such as CAN have no equivalent in classical IT and are a specialty in the automotive environment but they are highly standardized. This means that automatic scanning tools are well-suited to give a first overview. In this context, scans of diagnostic functionality are notable, as those are likely to contain weakly documented security critical functionality, such as development or debugging functionality. As a third form of vulnerability scanning, the configuration for the whole system can be analyzed to identify security gaps, e.g. access to critical functions for everybody without authentication or automated check for authentication. Vulnerability scanning ensures that a system is secure against known attacks, which can easily be tried out by attackers and therefore are very likely attacks.

5.3 *Automotive Fuzzing*

Fuzzing is a technique used for a long time to test software and IP networks by exposing the implementation to unexpected, invalid, or random input with the hope that the target will react in an unexpected way, and thereby, to discover new vulnerabilities. The reaction of the target can range from strange output over unspecified behavior up to crashes. However, fuzzing as a testing technique for automotive target systems is relatively new although modern vehicles have many similarities to common computer networks. In fact, ECUs can be viewed as small computers running different software, that are connected by different network types such as CAN, FlexRay, or MOST. Hence, it is quite natural to come up with the

idea to apply fuzz testing also to automotive target systems as part of the security testing process.

In general, fuzzing consists of three different steps: firstly the creation of the input for the target, secondly the delivery of the input to the target and lastly the monitoring of the target system to detect errors in the program flow. Since fuzzing is widely used in the computer world, fuzzing tools such as Peach [16] already exist. Peach has a powerful fuzz generator that can be adapted individually for different protocols, e.g. UDS. The input generated by the fuzz generator is then delivered to the target using the required transport protocol. The target system is now monitored to detect possible vulnerabilities. The monitoring process can range from inspection of the return values up to the usage of debuggers which observe the internal status of the target device. In the end, all identified unusual behavior has to be analyzed by an expert to detect exploitable vulnerabilities. Examples for such exploitable bugs are insufficient input validation or undocumented functionality, e.g. open debug or configuration interfaces. One famous example for insufficient input validation is the Heartbleed Bug [17] of OpenSSL, which allows reading out critical data since length parameters are not double-checked.

In the automotive context, fuzzing can be applied to diagnosis protocols, such as UDS, or to automotive network protocols, e.g. CAN, FlexRay, MOST or LIN. However, classical fuzzing targets, i.e. IP based networks, play an increasing role in modern vehicles. Hence, automotive security testing also benefits from experiences made in fuzz testing of classical protocols and modern software applications, e.g. cellphone apps.

5.4 Automotive Penetration Testing

Penetration tests start with general reconnaissance, which includes enumerating interfaces, determining components and their connections on the PCB, specifications available to a hypothetical attacker, and, in general, any information that can be helpful in further attacks. Using the information acquired in the first step, further attacks can be planned. A second step may include attacks of local external interfaces such as the CAN bus, Ethernet, USB, serial ports, or attacks on the hardware itself.

Such invasive hardware penetration tests are motivated by either IP protection or authoritative functionalities that rely on the integrity of the ECU against interests of physically present persons. Examples are theft protection, component protection, protection from odometer manipulations, feature activation, protection from false warranty claims from “tuned” vehicles, or safety functionality. One common method for a tester is to find overlooked or undocumented debug access interfaces or to gain access to ECU-internal interfaces such as memory buses. More advanced methods require etching open chip packages and accessing the actual silicon chip.

The specific forms of penetration testing are black-box testing, white-box testing, and grey-box testing.

For black-box testing, the tester is provided with practically no documentation or specifications, except information that could also be acquired by a real world attacker. The advantage of this method is that it results in a very realistic simulation of an attack. As a disadvantage, the penetration tester must spend a lot of time on basic reverse engineering and there is a good chance that deeper attack paths are not discovered because the tester did not circumvent easier first-line defense mechanisms, whereas an attacker may later break such defense mechanisms, by luck, because more information got public, because the state-of-the-art has advanced, or because he invested more resources than the tester.

For a white-box test, the tester is provided with full specifications and documentation for the device under test. This means he is able to specifically target weaknesses and has more resources available, which he does not have to spend on gaining information. Both reasons improve the efficiency of the test. The disadvantage of white-box testing is that the conditions are not nearly as realistic as in black-box tests giving a less reliable estimation of attack difficulty and likelihood.

Grey-box-tests represent a middle ground between black- and white-box testing. For a grey-box test, the tester receives partial information, concerning a specific sub-system that is in focus or information that a specific attacker such as an insider could have acquired. A step-wise approach is also possible, where the tester receives more information or access after having shown the basic presence and exploitability of vulnerability without having to fully develop the attack itself. This optimizes the ratio of test efficiency and realism.

6 Conclusion and Outlook

This paper showed the strong need for systematic automotive security evaluation and discussed different methods for theoretical and practical security evaluation of automotive IT components. We focused especially on practical security testing for automotive components such as automotive penetration testing since practical security testing is still relatively new to the automotive domain.

We believe theoretical and practical security evaluations will become a standardized and mandatory procedure, for instance as we know it from today's safety testing in ISO26262, to fulfill state-of-the-art product liability requirements and to protect various upcoming automotive business models (e.g. on pay-per-use basis). This of course requires serious efforts from OEMs, suppliers, and security experts to establish necessary automotive security testing expertise, testing standards, and testing infrastructures.

References

1. Miller C, Valasek C (2013) Adventures in automotive networks and control units. In: DEFCON 21 Hacking Conference
2. Koscher K, Czeskis A, Roesner F, Patel S, Kohno T, Checkoway S, McCoy D, Kantor B, Anderson D, Shacham H (2010) Experimental security analysis of a modern automobile. In: 2010 IEEE symposium on security and privacy (SP)
3. Checkoway S, McCoy D, Kantor B, Anderson D, Shacham H, Savage S, Koscher K, Czeskis A, Roesner F, Kohno T (2011) Comprehensive experimental analyses of automotive attack surfaces. In: USENIX security, San Francisco, CA, USA
4. Markey E (2013) As wireless technology becomes standard, Markey queries car companies about security, privacy. In: Press release of the US senator for Massachusetts, Massachusetts, USA, 23 Dec 2013
5. Miler C, Valasek C (2014) A survey of remote automotive attack surfaces. In: Blackhat
6. Thiemel AV, Janke M, Steurich B (2013) Speedometer manipulation—putting a stop to fraud. ATZ elektronik worldwide Edition, 2013–02
7. ExtremeTech, Hack the diagnostics connector, steal yourself a BMW in 3 min. <http://www.extremetech.com/extreme/132526-hack-the-diagnostics-connector-steal-yourself-a-bmw-in-3-minutes>
8. ExtremeTech, Hackers can unlock cars via SMS. <http://www.extremetech.com/extreme/91306-hackers-can-unlock-cars-and-meddle-with-traffic-control-systems-via-sms>
9. Wolf M, Scheibel M (2012) A systematic approach to a quantified security risk analysis for vehicular IT systems. In: Automotive—safety & security, Karlsruhe
10. CERT Secure Coding Standards. <http://www.cert.org/secure-coding/research/secure-coding-standards.cfm?>
11. SAFECODE (2011) Fundamental practices for secure software development
12. Alliance of Automobile Manufacturers (2014) Auto cyber-security: continual testing, checks and balances. <http://www.autoalliance.org/auto-innovation/cyber-security>
13. The OpenSSL Project. <https://www.openssl.org/>
14. M. I. S. R. Association and M. I. S. R. A. Staff (2013) MISRA C: 2012: Guidelines for the use of the C language in critical systems, Motor Industry Research Association, 2013
15. OpenVAS—Open Vulnerability Assessment System. <http://www.openvas.org>
16. DEJA VU SECURITY (2014) Peach Fuzzer platform. <http://peachfuzzer.com/products>
17. The Heartbleed Bug (2014) <http://heartbleed.com/>
18. Wolf M (2009) Security engineering for vehicular IT systems—improving trustworthiness and dependability of automotive IT applications, Vieweg+Teubner Verlag



<http://www.springer.com/978-3-319-19817-0>

Energy Consumption and Autonomous Driving
Proceedings of the 3rd CESA Automotive Electronics
Congress, Paris, 2014

Langheim, J. (Ed.)

2016, XIII, 245 p. 122 illus., 109 illus. in color.,

Hardcover

ISBN: 978-3-319-19817-0