35th International Electric Vehicle Symposium and Exhibition (EVS35)
Oslo, Norway, June 11-15, 2022

STEVE: Security Testing Framework for EV Charging Environments

Jonghyuk Song[1], Jaejo Lee[2], Hyunwoong Kim[3], Jaeson Yoo[1], Kiho Joo [1]

[1]*jhsong@autocrypt.io, jyoo@autocrypt.io, khjoo@autocrypt.io*

[2]*Korea Electrotechnology Research Institute, 111, Hanggual-ro, Sangnok-gu, Ansan-si, Gyeonggi-d, South Korea, jjlee@keri.re.kr*

[3]*Gridwiz. 25 Sanun-ro, 208beon-gil, Bundang-gu, Seongnam, Gyeonggi, South Korea, teddy@gridwiz.com*

Summary

ISO 15118 is an international standard for charging electric vehicles, specifically focusing on communications between Supply Equipment Communication Controllers (SECC) and Electric Vehicle Communication Controllers (EVCC). However, this standard has not been tested against cyber-attacks because no security testing methods currently exist for the industry. In this paper, we propose STEVE, a security testing framework for electric vehicle (EV) charging environments, to find security vulnerabilities in EVCC and SECC that have been established with ISO 15118. We have implemented STEVE and have performed experiments that discovered several vulnerabilities in RISE-V2G. We expect that STEVE can be used to verify the security vulnerabilities of SECC and EVCC.

Keywords: charger, charging, V2G (vehicle to grid), testing processes

1 Introduction

The number of electric vehicles has rapidly increased in recent years. Consequently, a substantial amount of research on charging protocol has been published, including those that focus on cyber threats for EV and charging infrastructure [2-8]. Such studies have defined assets that can be targeted and have illustrated various scenarios with which EV charging communications can be attacked. They have also analyzed potential threats in all components of the charging ecosystem, which include not only EVs and charge points, but also backend servers and the grid itself. However, these works focus on the threats for the physical layer rather than the application layer. Most importantly, they do not provide implementation and experiment data.

In this study, we propose STEVE, a security testing framework for EV charging environments. We focus on analyzing attacks in ISO 15118-2, which specifies network and application protocol requirements. We define not

only ISO 15118-2 attacks proposed in previous studies, but also new potential attacks that were discovered during the course of this research. Finally, we prescribe countermeasures against these attacks, including how to verify the effectiveness of such countermeasures. We have implemented STEVE in Python3 and Java. We have evaluated STEVE with RISE-V2G (Reference Implementation Supporting the Evolution of the Vehicle-2-Grid communication interface ISO 15118) which is the most popular opensource implementation of ISO 15118 communication protocol [1]. During our evaluation period, STEVE discovered several vulnerabilities which can cause unaddressed exceptions in RISE-V2G.

2 Potential Cyberattacks in ISO 15118-2

This section describes potential security attacks in ISO 15118-2 environments. STEVE has verified the following root cause of these threats in order to detect vulnerabilities in advance.

2.1 Eavesdropping (A1)

Eavesdropping attack refers to the gathering of information such as the identification number, charging status, and tariff information. This attack can result in privacy leaks, or lead to other attacks.

2.2 Manipulation of Charging Data (A2)

Malicious EVs can try to fabricate the charging parameters or meter data. By doing so, the attacker can receive more energy than for which she paid.

2.3 Masquerading (A3)

Attackers can masquerade as an EV or a charging station, made possible by stealing valid certificates, manipulating identification numbers, or hijacking authenticated sessions.

2.4 Denial of Service (DoS) Attack (A4)

Network DoS attacks have the potential to take down charging services altogether. There are many DoS attack techniques such as flooding attack, SSL/TLS exhaustion attack, thc-ssl-dos and Slowloris.

2.5 XML Injection Attack (A5)

ISO 15118-2 transfers charging data in XML/EXI format. As a result, the data can be vulnerable to XML injection attacks that compromise the logic of the target by injecting unintended XML contents or structures.

2.6 Invalid Data Attack (A6)

Attackers can transmit inappropriate data or values in the charging parameters to raise exceptions on the target. Since ISO 15118-2 exchanges data in XML formats, it is necessary to verify whether the XML parser handles unexpected data properly.

3 Testing Methods to Verify Countermeasures

Table1: Attacks and Countermeasures

Countermeasures	Attacks
Correct use of TLS	A1, A2, A3, A4, A6
Validation on charging parameters	A2
Validation on identification parameters	A3
DoS attacks prevention	A4
XML injection prevention	A5
Data validation	A6

3.1 Correct Use of TLS

TLS can prevent most of the attacks mentioned in the previous section. Therefore, the EVCC should verify that TLS has been implemented in the SECC, and should validate PKI certificates to confirm the SECC's authenticity. STEVE has been designed to check three different items to verify TLS support. First, STEVE checks that TLS is enabled. If TLS is not enabled in SECC, all communications are vulnerable to eavesdropping attacks, which can lead to privacy leaks or other attacks. Second, STEVE verifies that updated TLS version are being used. Currently, the National Security Agency (NSA) recommends that only TLS 1.2/1.3 be used, and that TLS 1.0/1.1 be avoided. Third, STEVE confirms whether or not the certificate has expired. STEVE can verify the above the items by using OpenSSL, an open-source command tool that provides functions related to TLS/SSL [9]. This tool can help stakeholders identify various information about TSL/SSL of the server, such as TLS version or the expiry status of certificates.

3.2 Validation on Charging Parameters

The SECC should validate charging parameters of ISO 15118-2 protocol messages to prevent manipulation attacks from malicious EVCC components. To prevent such attacks, STEVE can transmit the parameters that violate the restrictions specified in ISO 15118-2, in order to check whether the SECC can successfully filter them. This process also checks whether the SECC handles exceptions, so that the stability of SECC can be validated as well.

Table2: Parameters of ISO 15118-2 Messages and Message Types

Message	Name	Type (restriction)
SupportedAppProtocolReq	ProtocolNameSpace	String (max length: 100)
	VersionNumberMajor	unsignedInt
	VersionNumberMinor	unsignedInt
	SchemaID	unsignedByte
	Priority	unsignedByte (1 to 20)
ServiceDiscoveryReq	ServiceScope	String (max length: 64)
	ServiceCategory	Enumeration
PaymentServiceSelectionReq	SelectedPaymentOption	Enumeration
ChargeParameterDiscoveryReq	RequestedEnergyTransferMode	Enumeration
	DepartureTime	unsignedInt
	Multiplier	Byte (-3 to +3)
	Unit	Enumeration
	Value	Short
PowerDeliveryReq	ChargeProgress	Enumeration
	ChargingProfileEntryStart	unsignedInt
	ChargingProfileEntryMaxNumberOf PhasesInUse	Byte (1 to 3)

Table 2 shows examples of the parameters validated by STEVE and describes the types and restrictions of each parameter, as specified in ISO 15118-2. Table 3 illustrates examples of malicious values used in the charging parameters via STEVE. The exceptional values are used in the test such as large, long, negative, and null values.

Table3: Malicious Values for Parameter Validation Tests

Malicious Values	Description
(empty)	Empty value for null/empty value check
“A” x 1024	Long value for max length check
0	Zero value for null/empty value check
-1	Negative values for checking unsigned parameters
4294967297	Unsigned integer max + 1 for checking integer overflow (32bit)
18446744073709551617	Unsigned integer max + 1 for checking integer overflow (64bit)

3.3 Validation on Identification Parameters

Malicious EVCCs can execute masquerading attacks by manipulating identification parameters. As such, the SECC should filter for fabricated parameters. STEVE sends malicious identification parameters to check whether the SECC can detect such fabricated parameters. Table 4 shows examples of identification parameters that were validated by STEVE. As in the previous section, STEVE transmits the malicious data, such as those found in Table 3, to verify that the SECC can detect them.

Table4: Identification Parameters

Message	Name	Type (restriction)
SessionSetupReq	SessionID	hexBinary (max length: 8)
	EVCCID	hexBinary (max length: 6)
PaymentServiceSelectionReq	ServiceID	unsignedShort
	ParameterSetID	Short
PowerDeliveryReq	SAScheduleTupleID	Short

3.4 DoS Attack Prevention

The SECC must provide reliable charging services to the EVCC at any time, signifying the importance of DoS attack prevention. Therefore, STEVE can execute DoS attacks via traditional techniques (such as flooding attacks) to check that the server can prevent them in the future. In order to utilize existing tools, IPv4 connection must be modified to IPv6, since ISO 15118-2 is based on IPv6.

3.5 XML Injection Prevention

Both the SECC and EVCC should validate incoming XML data to prevent attacks, namely because ISO 15118-2 exchanges data in XML format. STEVE can determine whether the SECC/EVCC are vulnerable to XML/XXE injection attacks by transmitting malicious XML data. Table 5 shows examples of malicious XML data that STEVE can insert into normal XML messages.

Table5: Malicious XML Values

Malicious values	Description
<, >, </>, <!--	Invalid XML parsing
<!ENTITY, <!DOCTYPE, <!ELEMENT	XML/XXE injection

In addition to injection attacks, STEVE also transmits invalid XML data by inserting special characters that cause parsing errors, in order to verify that the XML parser can handle exceptions correctly.

3.6 Data Validation

STEVE can transmit invalid V2GTP messages to check whether the SECC/EVCC handle exceptions properly. Figure 1 illustrates the V2GTP message header structure, while Table 6 describes each V2GTP header fields as specified in the ISO 15118-2 document.

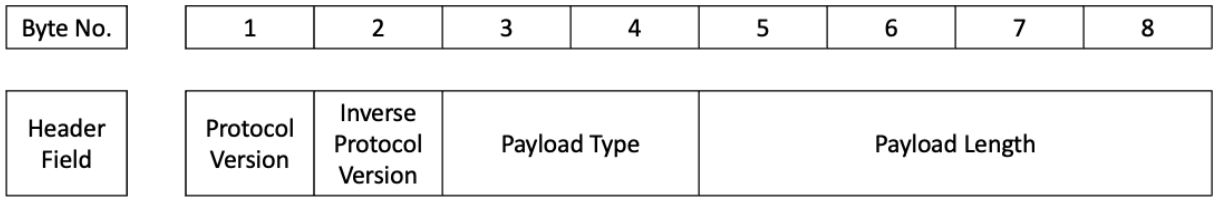

Figure 1: V2GTP message header structure in ISO 15118-2

STEVE generates invalid V2GTP headers by inserting random values into each field that are referenced to Figure 1 and Table 6. For example, some messages are generated with reserved values that have not been used in ISO 15118-2 (e.g., 0xFF in Protocol Version field). After the header has been created, complete V2GTP messages are generated along with a random payload. Among them, some messages are generated with a value of the payload length field that is different from the actual payload length. STEVE generates and sends a large amount of the invalid V2GTP messages, then checks if these errors are detected.

Table6: V2GTP Header Description in ISO 15118-2

Header field	Description	Value	Description
Protocol Version	Protocol version of V2GTP messages	0x01	V2GTP version 1
		0x00, 0x02-0xFF	Reserved
Inverse Protocol Version	Bit-wise inverse value of the protocol version (<Protocol_Version> XOR 0xFF)	0xFE	V2GTP version 1
Payload Type	Information about how to decode the payload	0x8001	EXI encoded V2G Message
		0x9000	SDP request message
		0x9001	SDP response message
		0x0000 – 0x8000, 0x8002 – 0x8FFF, 0x9002 – 0x9FFF	Reserved
		0xA000-0xFFFF	Manufacture specific use
Payload Length	Length of the V2GTP message payload	0 – 4294967295	

4 Test Procedures

STEVE validates the test target in three stages as follows. Exception has been made for the TLS test and the DOS attack test.

1. Preparation process

   Normal communication processed until the parameter to be tested has been reached.

2. Malicious data transmission

   During the test stage, STEVE sends malicious data to the target.

3. Availability check

   STEVE verifies that the test target is operating normally after malicious data has been transmitted.

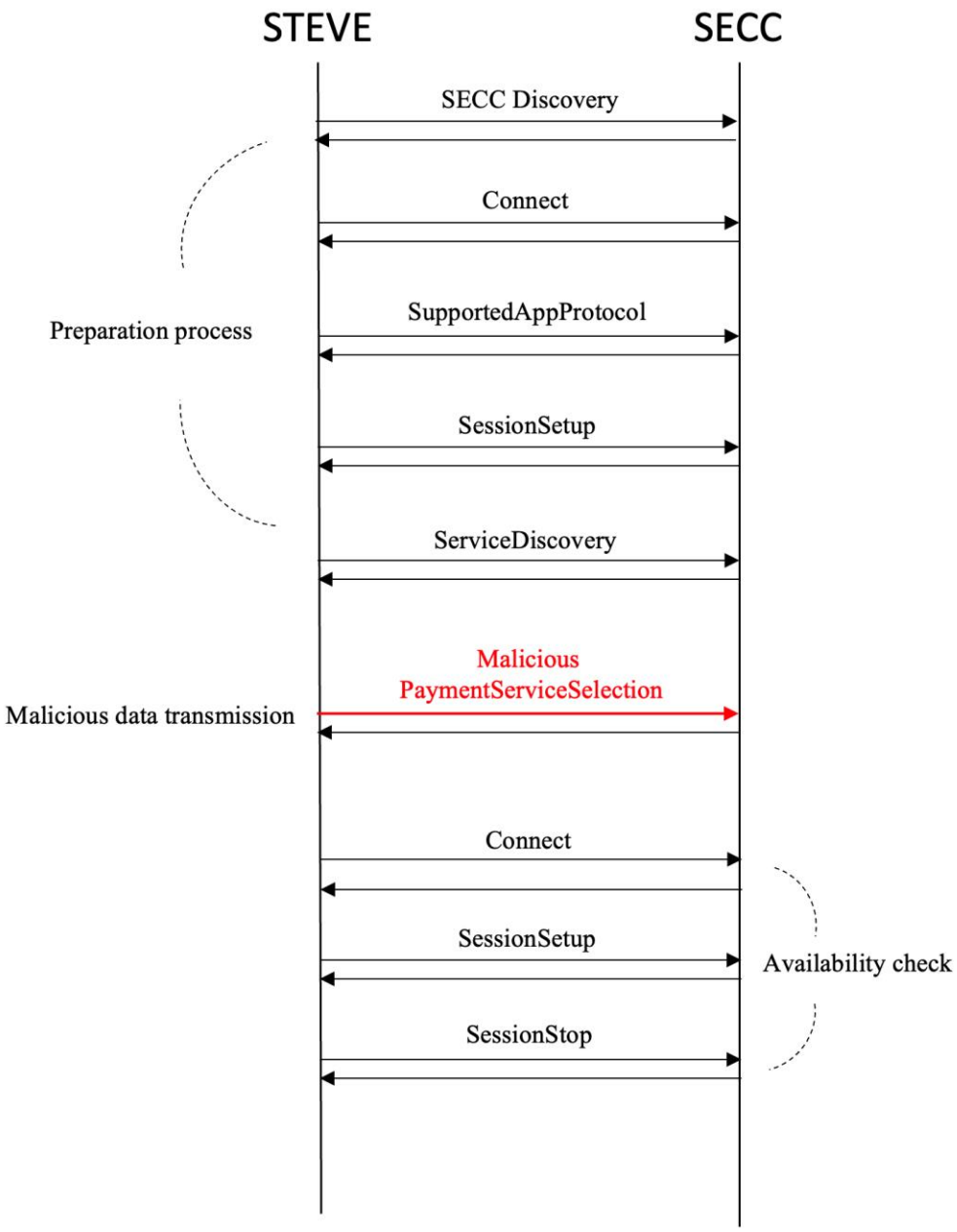

Figure 2: Example procedure for testing *"PaymentServiceSelection"*

Figure 2 is an example procedure for testing the *"PaymentServiceSelection"* stage. First, STEVE conducts normal communications with SECC until the *"PaymentServiceSelection"* stage has been reached. Second, STEVE generates a malicious *"PaymentServiceSelection"* request by using data in Table 3 or Table 5 and transmits it to SECC. Finally, STEVE identifies that SECC is available. If it is not available, STEVE reports that a vulnerability has been discovered by the malicious data.

5 Implementation and Experiments

STEVE is implemented in Python3, except the EXI encoder/decoder, which is implemented in Java. This is because no stable library exists for EXI in Python3. The source code is available at Github [10]. We have evaluated STEVE with RISE-V2G and found several exceptions that were not processed by RISE-V2G. Table 7 shows some examples of the exceptions occurred during the test. When RISE-V2G-SECC raises exceptions, the SECC either transmits a negative response to the EVCC or do not transmit a response altogether, resulting in a timeout. However, the SECC does not crash and continues to run, so the exception was noted without affecting operations. However, if a new normal EVCC is connected and transmits a *"SupportedAppProtocolReq"*, the SECC returns *"Failed_NoNegotiation"* or invalid EXI encoded messages to the EVCC, even though the request has been labeled as legitimate.

Table7: Exceptions in RISE-V2G

Exceptions	Description
java.lang.IllegalArgumentException	hexBinary needs to be even-length: 0 contains illegal character for hexBinary: -1
java.lang.NumberFormatException	Not a number: < For input string: "18446744073709551617"
java.lang.NullPointerException	-

6 Conclusions

This paper proposes a security testing framework to find security vulnerabilities in the SECC and EVCC that have been established with ISO 15118-2 communications. We define potential cyberattacks in ISO 15118-2 and prescribes countermeasures to prevent them. We have implemented STEVE to verify the threats and have discovered unaddressed exceptions in RISE-V2G. In the future, STEVE will be further developed and used for SECC/EVCC verification.

Acknowledgments

This work was supported by the Energy Efficiency & Resources program of the Korea Institute of Energy Technology Evaluation and Planning (KETEP) grant funded by the Korea government Ministry of Knowledge Economy (No. 20209810400030).

References

[1] V2G Clarity. 2020, *Reference Implementation Supporting the Evolution of the Vehicle-2-Grid communication interface ISO 15118*, https://github.com/SwitchEV/RISE-V2G

[2] R. Falk et.al., *Electric Vehicle Charging Infrastructure – Security Considerations and Approaches,* INTERNET 2012, ISBN: 978-1-61208-204-2, pp.58-64, 2012.

[3] S. Fries et.al., *Securely connecting Electric Vehicles to the Smart Grid*, International Journal On Advances in Internet Technology, vol. 6, no. 1 and 2, pp. 57–67, 2013, ISSN: 1942-2652.

[4] S. Lee et.al., *Study on Analysis of Security Vulnerabilities and Countermeasures in ISO_IEC 15118 based Electric Vehicle Charging Technology*, 2014 International Conference on IT Convergence and Security (ICITCS).

[5] K. Bao et.al., *A threat analysis of the vehicle-to-grid charging protocol ISO 15118*, Comput Sci - Res Dev, 33 (2018), pp. 3-12.

[6] R. Baker et.al., *Losing the car keys: Wireless PHY-layer insecurity in EV charging*, USENIX Security, 2019

[7] S. Acharya et.al., *Cybersecurity of smart electric vehicle charging: A power grid perspective*, IEEE Access, vol. 8, pp. 214 434–214 453, 2020

[8] J. Antoun et.al., *A Detailed Security Assessment of the EV Charging Ecosystem*, IEEE Network, 2020.

[9] OpenSSL. https://www.openssl.org/

[10] STEVE. https://github.com/freest4r/STEVE

Authors

Jonghyuk Song is lead for Autocrypt's Red Team. His current tasks are security testing for automotive including fuzzing, penetration testing, and vulnerability scanning. He researches security issues in not only in-vehicle systems, but also V2G and V2X systems. Jonghyuk received his Ph.D. in Computer Science and Engineering at POSTECH, South Korea in 2015. He has worked in Samsung Research as an offensive security researcher, where his work included finding security issues in smartphones, smart home appliances and network routers.