Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.063289

ARTICLE

Comprehensive Black-Box Fuzzing of Electric Vehicle Charging Firmware via a Vehicle to Grid Network Protocol Based on State Machine Path

Yu-Bin Kim, Dong-Hyuk Shin and Ieck-Chae Euom*

System Security Research Center, Chonnam National University, Gwangju, 61186, Republic of Korea

*Corresponding Author: Ieck-Chae Euom. Email: iceuom@jnu.ac.kr

Received: 10 January 2025; Accepted: 19 May 2025; Published: 03 July 2025

ABSTRACT: The global surge in electric vehicle (EV) adoption is proportionally expanding the EV charging station (EVCS) infrastructure, thereby increasing the attack surface and potential impact of security breaches within this critical ecosystem. While ISO 15118 standardizes EV-EVCS communication, its underspecified security guidelines and the variability in manufacturers' implementations frequently result in vulnerabilities that can disrupt charging services, compromise user data, or affect power grid stability. This research introduces a systematic black-box fuzzing methodology, accompanied by an open-source tool, to proactively identify and mitigate such security flaws in EVCS firmware operating under ISO 15118. The proposed approach systematically evaluates EVCS behavior by leveraging the state machine defined in the ISO 15118 standard for test case generation and execution, enabling platform-agnostic testing at the application layer. Message sequences, corresponding to valid and mutated traversals of the protocol's state machine, are generated to uncover logical errors and improper input handling. The methodology comprises state-aware initial sequence generation, simulated V2G session establishment, targeted message mutation correlated with defined protocol states, and rigorous response analysis to detect anomalies and system crashes. Experimental validation on an open-source EVCS implementation identified five vulnerabilities. These included session integrity weaknesses allowing unauthorized interruptions, billing manipulation through invalid metering data acceptance, and resource exhaustion vulnerabilities from specific parameter malformations leading to denial-of-service. The findings confirm the proposed method's capability in pinpointing vulnerabilities often overlooked by standard conformance tests, thus offering a robust and practical solution for enhancing the security and resilience of the rapidly growing EV charging infrastructure.

KEYWORDS: Internet of Things (IoT) security; risk assessment; data privacy; fuzzing test; electric vehicle charger security

1 Introduction

The rapid proliferation of electric vehicles (EVs) is transforming the automotive ecosystem, with projections indicating that over 50% of all vehicles will soon be electric [1]. According to Statista, the global EV charging infrastructure market is expected to grow from USD 14.495 billion in 2021 to approximately USD 128.135 billion by 2030, at a compound annual growth rate of 27% [2]. Additionally, *Markets & Markets* reported that this market was valued at USD 11.9 billion in 2022 and is projected to reach USD 72.6 billion by 2027, with an annual growth rate of 45% [3]. A robust charging infrastructure, comprising EV charging stations (EVCSs), charge-point operators (CPOs), and electric mobility service providers (e-MSPs), is essential for delivering effective EV charging services. EVCSs facilitate communication between EVs and CPOs through ISO 15118 [4] and the open charge point protocol (OCPP) [5]. The following security issues exist concerning the EV charging infrastructure.

Copyright © 2025 The Authors. Published by Tech Science Press.
This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Recent security issues include power grid disruption attacks through charging networks, where hackers manipulate multiple charging stations to simultaneously initiate or halt charging, destabilizing the power grid [6]. Additionally, attacks exploiting vulnerabilities in the backend servers of charging stations are increasing, allowing remote manipulation of charging sessions or unauthorized access to user data [7]. In shared EV systems, targeted attacks on charging stations in specific areas could disrupt the entire network. Another concern is user privacy violations through charging data analysis. Studies have shown that even by analyzing current signals exchanged during charging, it is possible to identify specific EV models and track their movement patterns [8]. To mitigate this risk, security monitoring using Intrusion Detection Systems and security assessments at the individual charging station level are being actively researched [9]. Strengthening the security of charging protocols like ISO 15118 remains a critical challenge. Passive testing systems have been proposed to analyze data exchanged during charging and verify compliance with security requirements [10]. Additionally, discussions are ongoing regarding blockchain-based security solutions and payment system protections [11]. Research also suggests that a dedicated security model is necessary for fleet charging infrastructure [12]. As EV charging infrastructure expands rapidly, continuous improvements in security measures are essential [13,14]. To address these security issues, conducting security research on EVCS communication protocols is essential.

Since its establishment in 2010, the Open Charge Alliance has continuously updated the OCPP to enhance security, and active research on security improvements is ongoing. In contrast, ISO 15118 governs communication between EVs and EVCSs and is less secure than OCPP. Since its inception in 2013, ISO 15118 has not undergone significant revisions in its communication and security standards revisions, rendering its security insufficient. While the ISO 15118 standard includes security mechanisms like Transport Layer Security (TLS) and authentication, they are optional rather than mandatory, leading to inconsistent implementation. Manufacturers may simplify communication structures, and CPOs might omit security measures due to cost or network concerns, leaving charging interfaces vulnerable to attacks. The key issue is not the absence of security in the standard but the lack of enforcement mechanisms and the varying implementation levels across different operators, resulting in inconsistent security capabilities and increased security risks. Studies have shown that approximately 90% of EVSEs do not implement TLS encryption, further highlighting the critical gap in security enforcement [15]. Additionally, the standard lacks explicit security guidelines and sufficient implementation requirements, making it susceptible to exploitation. Attackers can target the communication interface between EVs and EVCSs, potentially compromising charging stations or launching widespread attacks on the charging infrastructure. Several security threats related to this interface have already been identified [16]. This study evaluates whether EVCSs correctly implement the ISO 15118 standard and proposes a black-box fuzzing tool to identify additional vulnerabilities at the application layer. Black-box fuzzing enables the assessment of system functionality without requiring access to internal code structures, logic, or implementation details. By employing diverse fuzzing strategies, this approach systematically explores security vulnerabilities in protocol interactions.

The primary contributions of this study are as follows:

1. A black-box fuzzing method is introduced to evaluate multiple ISO 15118 implementations without prior knowledge of the system's internal workings. This approach allows for a functional assessment without analyzing the underlying code structure, logic, or implementation details.
2. Various fuzzing techniques are employed to identify and exploit security vulnerabilities embedded within ISO 15118 protocol interactions. This research highlights fuzzing as a proactive security measure, offering valuable insights into detecting and mitigating weaknesses in the protocol.

3. The proposed fuzzing tool is used to identify vulnerabilities in EVCS implementations. Based on these findings, this study suggests measures to enhance the security of the ISO 15118 standard.

2 Background and Related Work

This section provides an overview of the EV charging infrastructure and the communication protocols that support it. It examines existing threat-analysis studies related to charging infrastructure security and explores research on vulnerabilities specific to EVCSs. Reviewing these areas establishes a foundational understanding of the security challenges facing EVCSs and highlights the need for robust protective measures against potential threats.

2.1 Electric Vehicle Charging Infrastructure

The EV charging infrastructure consists of various components designed to facilitate seamless charging for EV users, forming the foundation of EV charging services. As illustrated in Fig. 1, this infrastructure encompasses EV users, EVs, EVCSs, and a network of operational entities that manage and support charging services. The core components of this infrastructure are EVs and EVCSs. EVCSs transfer electrical power to EV batteries through chargers, which act as physical intermediaries. Chargers are categorized based on charging speed into slow, fast, and ultra-fast types, providing power levels tailored to different user needs and EV specifications.

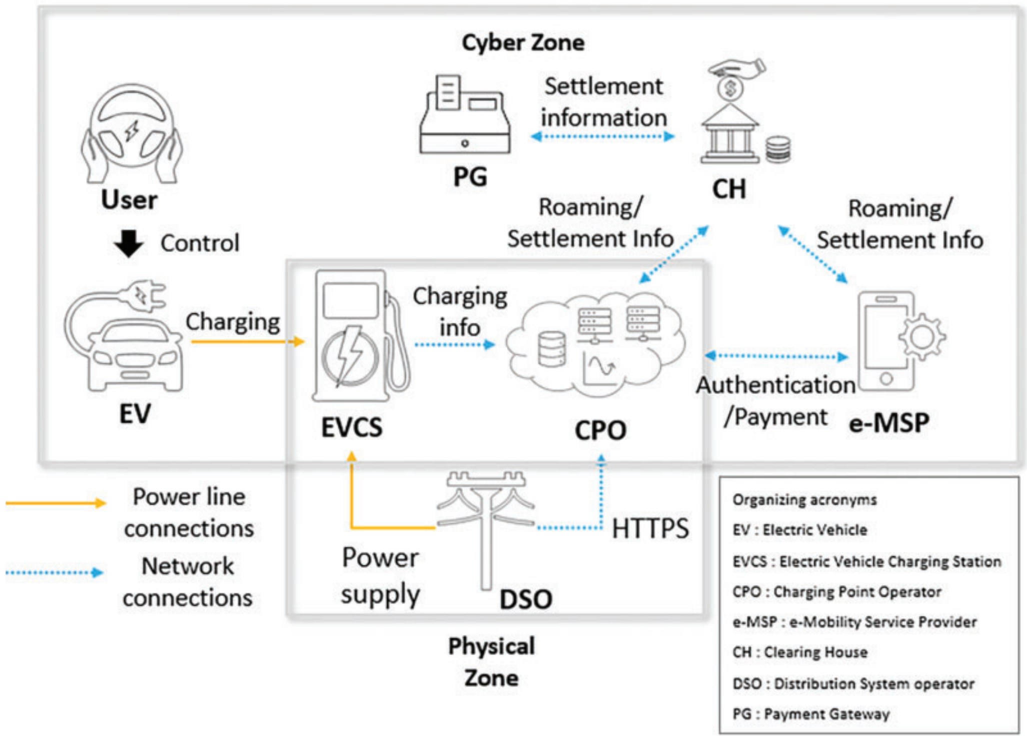


Figure 1: Electric vehicle (EV) charging infrastructure

CPOs manage the installation, operation, and maintenance of chargers, ensuring their functionality and availability. The e-MSPs offer platforms that allow users to access charging services, often integrating multiple payment methods and roaming services for seamless transactions [17]. Another critical component is the clearinghouse, which ensures interoperability among multiple charging networks. This allows users to charge their vehicles at EVCSs without being restricted to a single provider network. Additionally, payment gateways facilitate secure transactions by processing user payments and managing settlement information.

The EV charging infrastructure is also closely integrated with DSOs, which maintain grid stability and ensure a reliable electricity supply to EVCSs and EVs. DSOs play a crucial role in balancing power demand and distribution to meet EV charging needs efficiently. These components interact to create a seamless and user-friendly charging experience. For instance, when a user searches for an available EVCS and makes a reservation via a mobile application, CPOs and e-MSPs coordinate to confirm the reservation and initiate the charging process. Once charging is completed, the clearinghouse and payment gateway handle the transaction and settlement procedures. This interconnected system extends beyond a simple energy supply, evolving into an advanced infrastructure that enhances the overall user experience.

2.2 Electric Vehicle Charging Station Security Threat

The EVCSs rely on two primary communication interfaces: one with EVs and another with CPOs. Research has identified security vulnerabilities in these interfaces, emphasizing the need for robust protections to prevent unauthorized access and control over the central controller [18]. Additionally, studies indicate that EVCSs introduce vulnerabilities in data transmission protocols, further highlighting security concerns. In the United States, threat modeling has been conducted to assess security risks in EVCS infrastructure, identifying multiple attack vectors and potential threats [19]. Similar studies in the United Kingdom have focused on analyzing vulnerabilities and attack surfaces in EVCS systems [20]. Meanwhile, Canadian research has extended beyond vulnerability identification to propose a framework for security risk assessment risks [21]. However, these studies have not extensively examined vulnerabilities specific to individual communication interfaces and protocols.

The CPO-EVCS interface has been standardized since Version 1.0 and widely adopted in Version 1.6 through the OCPP [22]. Research indicates that OCPP 1.6 has inadequate security measures, making it susceptible to denial-of-service (DoS), man-in-the-middle (MITM), and code injection attacks [23,24]. Additionally, vulnerabilities in OCPP can be exploited for power grid attacks and data breaches [25,26]. A detailed analysis of injection vulnerabilities in CPO implementations has demonstrated that manipulated input values can facilitate various attacks [27]. Some studies have used fuzzing techniques during WebSocket handshakes to test potential OCPP backend exploits, while others have developed malicious CPO tools to identify vulnerabilities by inputting malicious data into the EVCS [28]. Building on these findings underscore the importance of continuous security, research highlighted how OCPP, focusing on both protocol vulnerabilities can be leveraged for MITM, DoS, replay, masquerading, and FDI attacks, ultimately impacting power grid stability [29]. Additionally, another study explored switching attacks and proposed an edge-based AI detection system to mitigate these threats and enhance grid resilience [30]. Therefore, research on the OCPP has encompassed a wide range of efforts, from identifying vulnerabilities to discovering specific weaknesses in CPO implementations.

The EV-EVCS interface relies on the ISO 15118 protocol for communication. However, this standard lacks robust security requirements, making it vulnerable to MITM attacks due to power-line communication (PLC) [31] and vulnerabilities have been identified that allow session hijacking via relay attacks [32]. Notably, ISO 15118 does not mandate TLS, and while guidelines exist for TLS implementation, they lack precise requirements [33].

Furthermore, insufficient authentication mechanisms enable session hijacking and MITM attacks [34], even when ISO 15118 is fully implemented [35]. The use of PLC in ISO 15118 also makes it susceptible to wireless attacks that can disrupt charging sessions [36]. Compared to OCPP, research on vulnerabilities arising from input manipulation in ISO 15118 is limited. Studies suggest that input validation is critical, as ISO 15118 lacks built-in protections for confidentiality, integrity, and availability against data manipulation [37]. While OCPP security measures are continuously improved by the Open Charge Alliance, ISO 15118 remains insufficiently secure [38]. This highlights the urgent need for further research on securing communication interfaces between EVs and EVCSs.

A comparison of existing threat studies (Table 1) shows that OCPP has been the focus of more extensive security and input validation research than ISO 15118. The lack of research on ISO 15118 suggests a higher likelihood of vulnerabilities due to improper input validation, underscoring the necessity for further investigations in this area.

Table 1: Comparative analysis of related work in EVCS threat

Reference	Target protocol	Threat	Consideration of input validation
[22]	Open charing point protocol	MITM, session hijacking	×
[23]		DoS, MITM, RCE	×
[24]		MITM, session hijacking	×
[25]		DoS, MITM, replay attack	×
[26]	Open charing point protocol, ISO 15118	MITM, sniffing	×
[27]	Open charing point protocol	MITM, DoS, data poisoning	○
[28]		DoS, code injection	○
[29]		MITM, DoS, masquerading	×
[30]		Switching attack	×
[31]	ISO 15118	Sniffing, MITM	×
[32]	ISO 15118	Relay attack	×
[33]	ISO 15118	TLS-based attack	×
[34]	ISO 15118	MITM, masquerade attacks	×
[35]	ISO 15118	MITM, masquerade attacks	×
[36]	ISO 15118	Wireless disruption	×
[37]	ISO 15118	MITM, sniffing	×
[38]	ISO 15118	MITM, DoS, replay	×

2.3 Vehicle-to-Grid Communication

ISO 15118, Introduced in 2013, enables vehicle-to-grid (V2G) communication between EVs and EVCSs. This standard defines communication protocol implementation across all seven layers of the open systems interconnection (OSI) model, from the physical layer to the application layer. Additionally, ISO 15118 includes test cases to verify compliance with its specifications. Fig. 2 illustrates the relationship between the ISO 15118 standard and the OSI seven-layer model. The ISO 15118 standard operates in conjunction with IEC 61851, a traditional which is traditionally used for EV charging standards [39]. Communication between EVs and EVCSs is facilitated via the control-pilot line, which determines the selection of digital communication protocols.

The primary communication standards within ISO 15118 include ISO 15118-2 [40] and ISO 15118-3 [41]. ISO 15118-3 defines the physical and data link layer requirements, specifying data transmission methods using PLC. These communications rely on the HomePlug Green PHY standard [42]. In contrast, ISO 15118-2 defines network-to-application layer requirements, detailing the messages exchanged between EVs and EVCSs, including IP identification and session termination. These messages are structured using XML schemas and transmitted in EXI format.

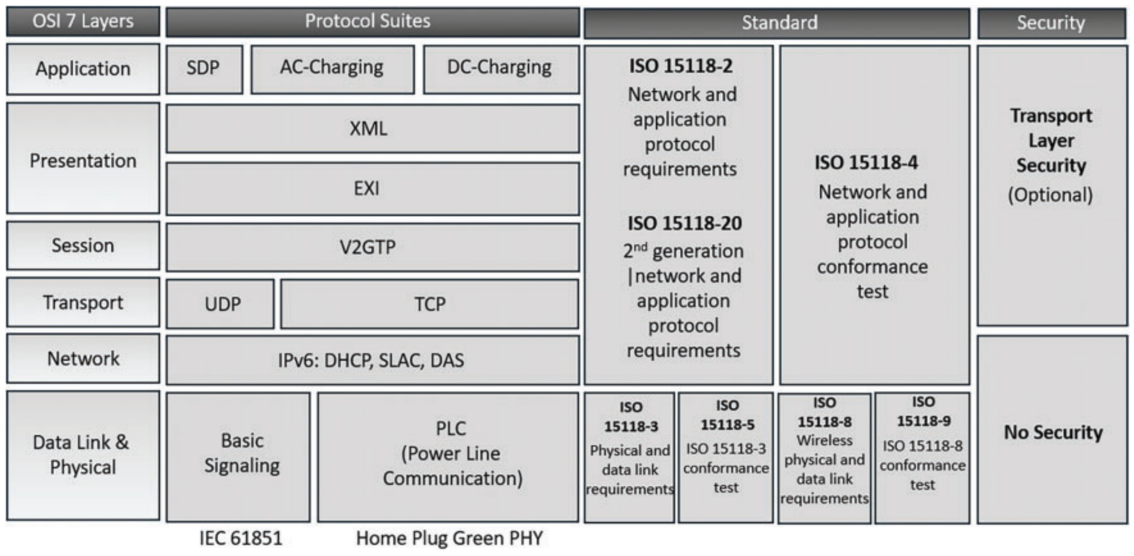

OSI 7 Layers
Protocol Suites
Standard
Security
Application
SDP
AC-Charging
DC-Charging
Presentation
XML
EXI
Session
V2GTP
Transport
UDP
TCP
Network
IPv6: DHCP, SLAC, DAS
Data Link & Physical
Basic Signaling
PLC (Power Line Communication)
IEC 61851
Home Plug Green PHY
ISO 15118-2 Network and application protocol requirements
ISO 15118-20 2nd generation |network and application protocol requirements
ISO 15118-4 Network and application protocol conformance test
ISO 15118-3 Physical and data link requirements
ISO 15118-5 ISO 15118-3 conformance test
ISO 15118-8 Wireless physical and data link requirements
ISO 15118-9 ISO 15118-8 conformance test
Transport Layer Security (Optional)
No Security

Figure 2: ISO 15118 and its relationship with the OSI seven-layer model

Additionally, ISO 15118-4 [43] and ISO 15118-5 [44] provide conformance testing for ISO 15118-2 and ISO 15118-3, respectively. These tests use predefined input-output conditions to ensure compliance with the standard.

In summary, ISO 15118 is critical for enabling secure and efficient V2G communication. Its comprehensive requirements and testing protocols enhance interoperability and reliability in EV charging, supporting broader EV integration into the power grid.

2.4 Vehicle-to-Grid Fuzzing Tests

This section examines fuzzing techniques used to identify security vulnerabilities in EVCSs arising from improper input validation. Before vulnerability identification through fuzzing, protocol conformance testing was conducted to verify standard compliance by analyzing input values and system responses. Some studies integrate fuzzing into conformance testing to assess protocol security more thoroughly. Fuzzing has proven effective in uncovering vulnerabilities in various protocols, including OCPP. Research has explored conformance testing for OCPP on EVCSs [45] and further extended fuzzing techniques to identify security weaknesses in CPO implementations [46]. However, comparable research on ISO 15118 remains limited, with only a few studies focusing on its conformance testing.

ISO 15118 lacks comprehensive implementation guidelines, highlighting the necessity for additional research. Previous studies have leveraged ISO 15118-4 for developing compliance assessment tools [47]. Another study identified TTCN-3 as a suitable programming language for implementing a test framework based on ISO 15118-4 [48]. However, these studies primarily focus on verifying minimal communication requirements rather than identifying broader security vulnerabilities.

Existing fuzzing studies on EVCSs have primarily targeted TLS vulnerabilities [49]. However, since TLS communication between EVCSs and EVs is relatively limited, fuzzing tests focused solely on TLS have a narrow scope. As a result, existing research has significant gaps in identifying vulnerabilities related to input manipulation. Table 2 shows a comparison of these studies and our proposed method. This study proposes a combined approach integrating conformance testing and vulnerability identification to enhance the security and reliability of EV-EVCS communications under ISO 15118.

Table 2: Research on EVCS fuzzing tests

Paper	**Protocol**	**Conformance testing**	**Vulnerability detection**	**Layer**	**Mutation**	**State machine**
[45]	Open charging point protocol	○	×	Application	×	×
[46]	Open charging point protocol	○	○	Application	○	○
[47]	ISO-15118	○	×	All	×	×
[48]	ISO-15118	○	×	All	×	×
[49]	ISO-15118	×	○	Transport	×	×
Proposed method	ISO-15118	○	○	Application	○	○

3 Proposed Fuzzing Method

This section outlines proposed fuzzing method, providing a detailed explanation of its steps. The method is designed to assess the compliance of electric vehicle chargers with established standards while detecting potential vulnerabilities.

3.1 Overview

The motivation for developing the proposed fuzzing method arises from the need to address the challenges posed by the diversity of firmware and architectures in EVCS. Black-box network fuzzing was chosen for this method because it allows the testing of external communication interfaces without the need for source code or firmware access, ensuring its applicability across various EVCS models. The method follows a black-box approach to testing ISO 15118 implementations, making it difficult to obtain detailed internal execution flow information, such as code paths. Since the only available feedback consists of protocol messages and their corresponding responses, measuring traditional "code coverage" is virtually impossible. Instead, this study introduces the concept of "protocol coverage" to assess how extensively the fuzzing process explores state transitions and message sequences defined by ISO 15118. The core idea is that if a specific function or state within the ISO 15118 protocol is deeply embedded in the message flow, reaching that function requires input that sequentially satisfies the protocol's constraints and procedures. In other words, if intermediate message formats, parameter ranges, or authentication steps are not correctly followed, further transitions cannot be reached, making it difficult to verify whether the function is properly implemented. Therefore, increasing "protocol coverage" in black-box fuzzing is a key strategy for effectively validating the full scope of the protocol. Additionally, the key to expanding protocol coverage is to perform fuzzing across all state machines of the EV charger. Fig. 3 illustrates the fundamental logic for fuzzing all state machines, and this concept serves as a core principle in the methodology design.

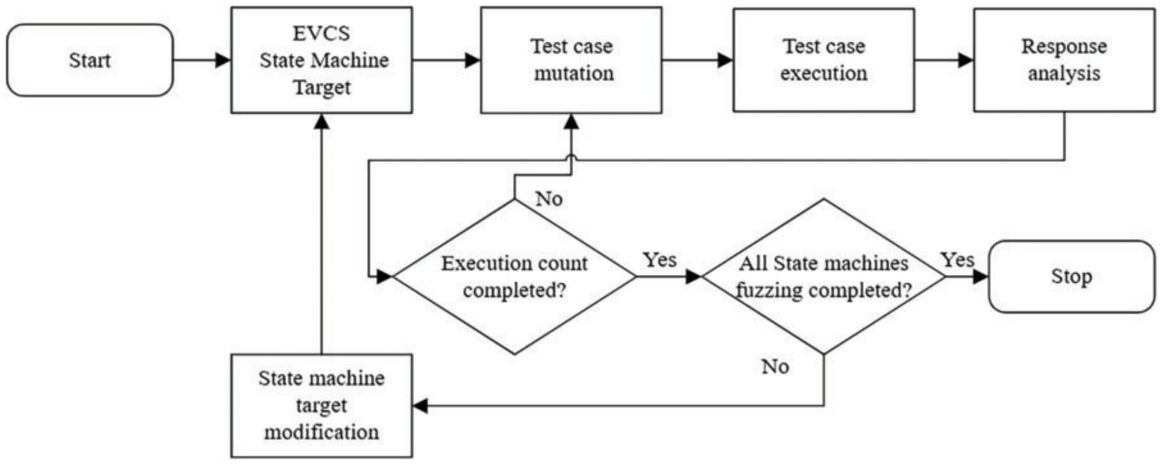

Start
EVCS State Machine Target
Test case mutation
Test case execution
Response analysis
No
Execution count completed?
Yes
All State machines fuzzing completed?
Yes
Stop
No
State machine target modification

Figure 3: EVCS state machine fuzzing flow chart

This paper focuses on the application layer of the ISO 15118 protocol, targeting logical flaws and implementation issues in EVCS communication. The communication layers of an EV charger follow the same principles as traditional IT systems for Layers 1 to 6, but the seventh layer, the application layer, is specifically designed to reflect the unique characteristics of EV charger communication. Additionally, a significant portion of ISO 15118-based interactions is governed by ISO 15118-2, making it a critical component of the overall communication process. Therefore, performing fuzzing at the application layer allows for the discovery of vulnerabilities specific to EV charger communication. Additionally, this method combines compliance-based fuzzing, which ensures adherence to ISO 15118 standards, with violation-based fuzzing, which injects malformed inputs to evaluate exception handling and ensure robust communication.

The proposed fuzzing method largely follows the network fuzzing methodology for IoT devices. To perform protocol fuzzing on EV chargers, a communication module was designed using a top-down approach based on the ISO 15118 standard, enabling interaction with EVCSs. This method is designed to simulate an EV and communicate directly with the EVCS. This approach helps identify issues that may arise during the communication process between EVs and EVCSs and enables the detection of potential vulnerabilities that could be exploited if an attacker impersonates an EV and sends abnormal input values to the EVCS.

The fuzzing method consists of four distinct stages: initial sequence generation based on state machine paths, V2G session establishment, message sequence mutation targeting specific state machine states, and analysis of EVCS responses. As shown in Fig. 4, the method is categorized into three primary components: preprocessing, fuzzing input, and fuzzing response analysis. Preprocessing involves all tasks that must be completed before initiating the fuzzing process. Once preprocessing is finished, the method moves on to the fuzzing input and response analysis phases to conduct the fuzzing tests. The following subsections provide a detailed description of each step.

3.2 Initial Sequence Generation Based on the State Machine Path

The application layer of the EVCS operates according to a state machine defined by the ISO 15118 standard, which governs the processing of messages within each state. The state machine specifies the permissible messages that can be received in particular states, as well as the conditions for state transitions. Appendix A (Fig. A1) presents the state machine defined in the standard. Based on the defined state machine, it was assumed that the EV charger follows the state machine model. The EV charger state machine was modeled using the state machine framework and state transitions defined in ISO 15118. These state transitions occur through a sequence of messages. In this paper, predefined message sequences are used to navigate the state machine and serve as the foundation for fuzzing tests. The EVCS state machine comprises 14 distinct states, with the fuzzing tool developed in this study specifically targeting 11 of the most critical states.

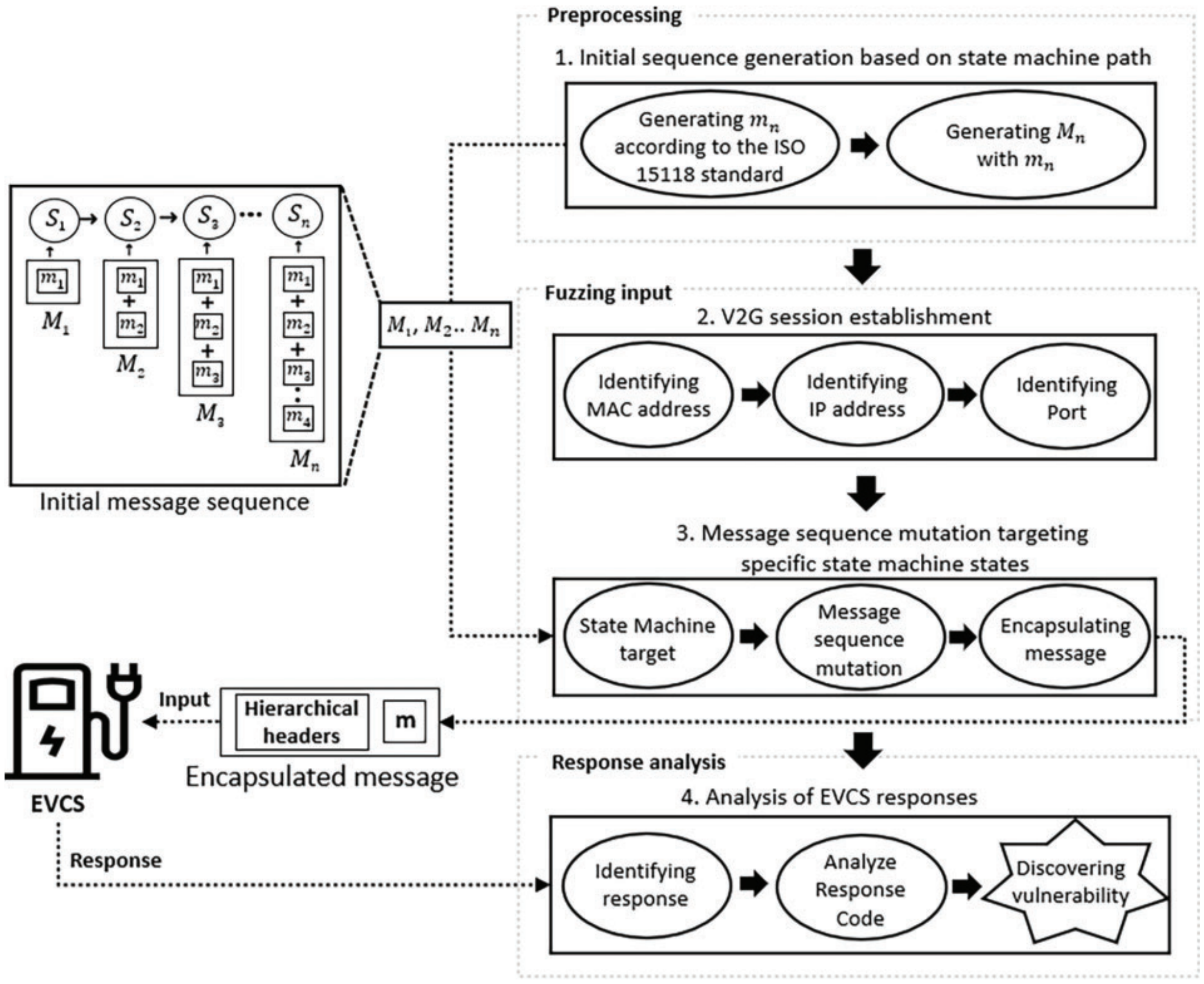


Figure 4: Proposed fuzzing method

Each message is labeled as m_n, and a sequence of messages is represented as M_n. The state in which the message m_n is received is denoted as S_n. The state machine includes states S_1 through S_{11}, with S_1 being the initial state, where a session is established and receives the first message m_1. To transition from state S_1 to state S_2, message m_1 must be received in state S_1. When targeting a specific state, the complete sequence of messages required to reach that state must be provided. For instance, to reach state S_4, the message sequence m_1, m_2, and m_3 must be predefined and represented as M_3. If additional messages, such as m_4, are necessary for fuzzing in state S_4, the final message sequence becomes $M_4 = \{m_1, m_2, m_3, m_4\}$.

The message sequences, based on the state machine paths, were defined according to the ISO 15118 standard specifications. This systematic modeling of the EVCS state transition process enables the effective design of fuzzing tests targeting specific states. These sequences also serve as the initial seeds for subsequent message mutations, which are crucial for conducting state machine-based testing.

Thus, the generation of message sequences based on state machine paths serves as a foundational step in the fuzzing process. This approach improves the preparation and efficiency of the fuzzing tests, providing a structured framework for evaluating compliance with the ISO 15118 standard and identifying potential security vulnerabilities in the EVCS.

3.3 Vehicle-to-Grid Session Establishment

In this paper, the fuzzing tool initiates communication with the EVCS by emulating an EV, a critical process for conducting fuzzing tests. This allows the fuzzing tool to exchange messages at the application layer. V2G session establishment follows the signal level attenuation characterization (SLAC) and service discovery protocols defined in the ISO 15118 standard, ensuring a secure physical connection and facilitating network information exchange between the EV and EVCS.

The SLAC protocol is used to verify the physical connection between the EV and the EVCS. The fuzzing tool begins the SLAC process by sending a CM_SLAC_PARM.REQ message, emulating the role of an EV. Upon receiving a CM_SLAC_PARM.CNF message from the EVCS, the fuzzing tool transmits the CM_SLAC_ATTEN_CHAR.IND message three times to prepare for signal attenuation measurements. This setup allows the EVCS to measure signal attenuation based on the transmitted signals.

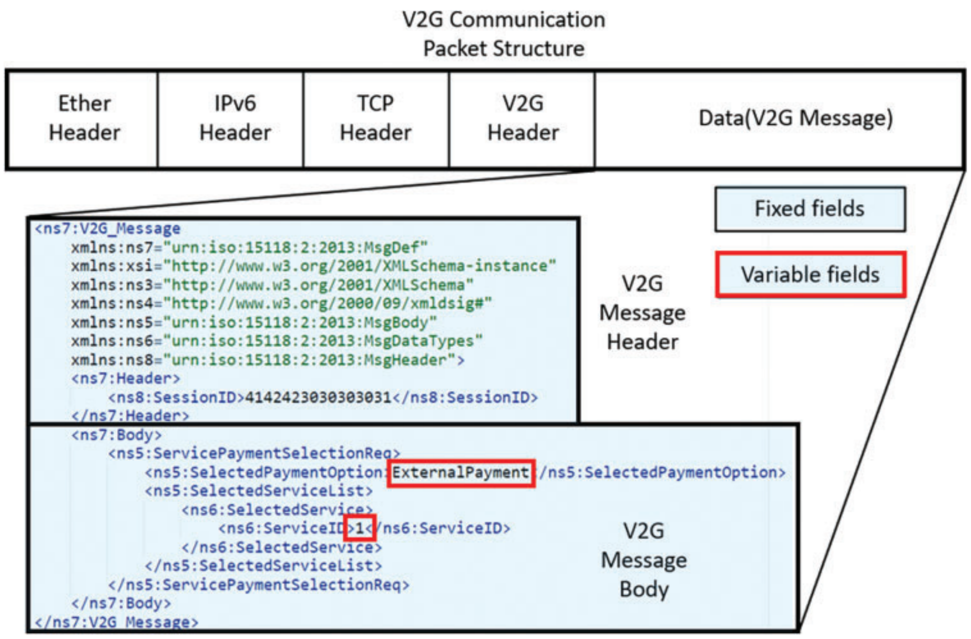
The fuzzing tool continuously sends CM_SLAC_SOUND.IND messages and receives responses from the EVCS, which include the measured attenuation values. Once the attenuation measurements are complete, the fuzzing tool verifies the response with a CM_ATTEN_CHAR_RSP message and selects the EVCS with the lowest attenuation value. This is followed by sending a CM_SLAC_MATCH.REQ message to request the final matching. The physical connection is finalized upon receiving a CM_SLAC_MATCH.CNF message from the EVCS. Throughout the SLAC process, the fuzzing tool fully emulates the EV, interacting with the signal attenuation verification process of the EVCS to prepare for subsequent communication.

After the SLAC protocol, the fuzzing tool uses the service discovery protocol to exchange network information. The fuzzing tool sends a request for the IP address and port number of the EVCS to the local link multicast address FF02::1. Upon receiving the response from the EVCS, the fuzzing tool establishes the transmission control protocol and TLS sessions, completing the network connection required for communication.

3.4 Message Sequence Mutation Targeting Specific Machine States

Following the V2G session establishment, fuzzing is conducted based on the EVCS state machine. This involves targeting each state (S_1–S_{11}) and mutating the messages in these states for testing. Fuzzing begins in state (S_1) by altering the message m_1 and progresses through S_2 to S_{11} by sequentially mutating and transmitting the necessary messages for each state. The modification of these messages adheres to the XML format specified by the ISO 15118 standard, ensuring that the XML structure and syntax remain intact.

Importantly, only the values of the message elements are altered (Fig. 5). The messages exchanged by EV chargers follow the XML format and consist of Elements, Sub-Elements, and Element Values. The Elements are fixed in communication and are classified as either mandatory or optional, while the Element Values vary with each transmission. Therefore, in this study, only the Element Values were mutated during the fuzzing process. This approach minimizes the likelihood of the EVCS rejecting messages due to XML syntax errors, allowing the focus to remain on identifying vulnerabilities through variations in input values. By preserving the integrity of the XML format, the method ensures that the fuzzing process can effectively probe for security weaknesses without being hindered by format-related issues, thereby improving the reliability and accuracy of vulnerability detection in the EVCS communication protocols.



```
<ns7:V2G_Message
    xmlns:ns7="urn:iso:15118:2:2013:MsgDef"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ns3="http://www.w3.org/2001/XMLSchema"
    xmlns:ns4="http://www.w3.org/2000/09/xmldsig#"
    xmlns:ns5="urn:iso:15118:2:2013:MsgBody"
    xmlns:ns6="urn:iso:15118:2:2013:MsgDataTypes"
    xmlns:ns8="urn:iso:15118:2:2013:MsgHeader">
    <ns7:Header>
        <ns8:SessionID>4142423030303031</ns8:SessionID>
    </ns7:Header>
    <ns7:Body>
        <ns5:ServicePaymentSelectionReq>
            <ns5:SelectedPaymentOption>ExternalPayment</ns5:SelectedPaymentOption>
            <ns5:SelectedServiceList>
                <ns6:SelectedService>
                    <ns6:ServiceID>1</ns6:ServiceID>
                </ns6:SelectedService>
            </ns5:SelectedServiceList>
        </ns5:ServicePaymentSelectionReq>
    </ns7:Body>
</ns7:V2G_Message>
```

Figure 5: Message variable fields in V2G packets

This fuzzing process is carried out across all states (S_1–S_{11}) of the state machine. After designating a target state, the required messages for testing that state are modified. For example, in the state S_1, message m_1 is required, so fuzzing is performed by modifying each message field. When transitioning to a state S_2, the same strategy was applied to the message m_2, and this process continues for all subsequent states. If the altered messages result in unexpected responses, error messages, or software crashes upon input into the EVCS, these incidents are recorded and analyzed to identify potential vulnerabilities.

The design of the mutation operators used for this process takes a more diverse and efficient approach than traditional random mutation methods (e.g., [50,51]). These operators were inspired by the binary-based alteration mechanisms employed in tools like American fuzzy lop (AFL) [52] and other similar studies [53]. While traditional methods mutation techniques operate at the bit or byte level, XML is stored in a textual format, so directly applying binary mutations is not ideal. To address this challenge, the alteration operators were designed as string-based mechanisms to ensure compatibility with the XML format. One of the operators, the *delete value* operator, avoids generating null values by preventing deletions when the string length is 1. The *insert value* operator introduces a 10% probability of adding two values instead of one, increasing the size of the value field.

Messages are modified using two approaches: mutating within the range that complies with the defined constraints or altering them beyond the constraints to cause violations. This dual approach helps verify whether the EVCS correctly processes inputs that conform to the standards, ensuring compliance. It also aims to uncover potential vulnerabilities that may arise even when the EVCS is operating within standard-compliant scenarios. Mutating messages within the constraints checks whether the EVCS handles valid values appropriately while violating the constraints tests the EVCS's ability to manage abnormal inputs. Specifically, constraint violations are designed to test the robustness of the EVCS in handling exceptions and to identify potential security weaknesses due to insufficient or missing exception-handling mechanisms.

3.5 Analysis of the Electric Vehicle Charging Station Response

The EVCS communicates the results of processing input messages through the response code element in its response messages. These response code values indicate the status of message processing. Under normal conditions, a successfully processed message triggers an “OK” response, while errors or exceptional conditions during message processing result in a “FAILED” response. This section analyzes these response messages to assess how the EVCS handles input messages, ensuring compliance with the ISO 15118 standard and identifying potential security vulnerabilities. Additionally, Fig. 6 illustrates the methodology used to detect vulnerabilities and crashes in the EVCS.

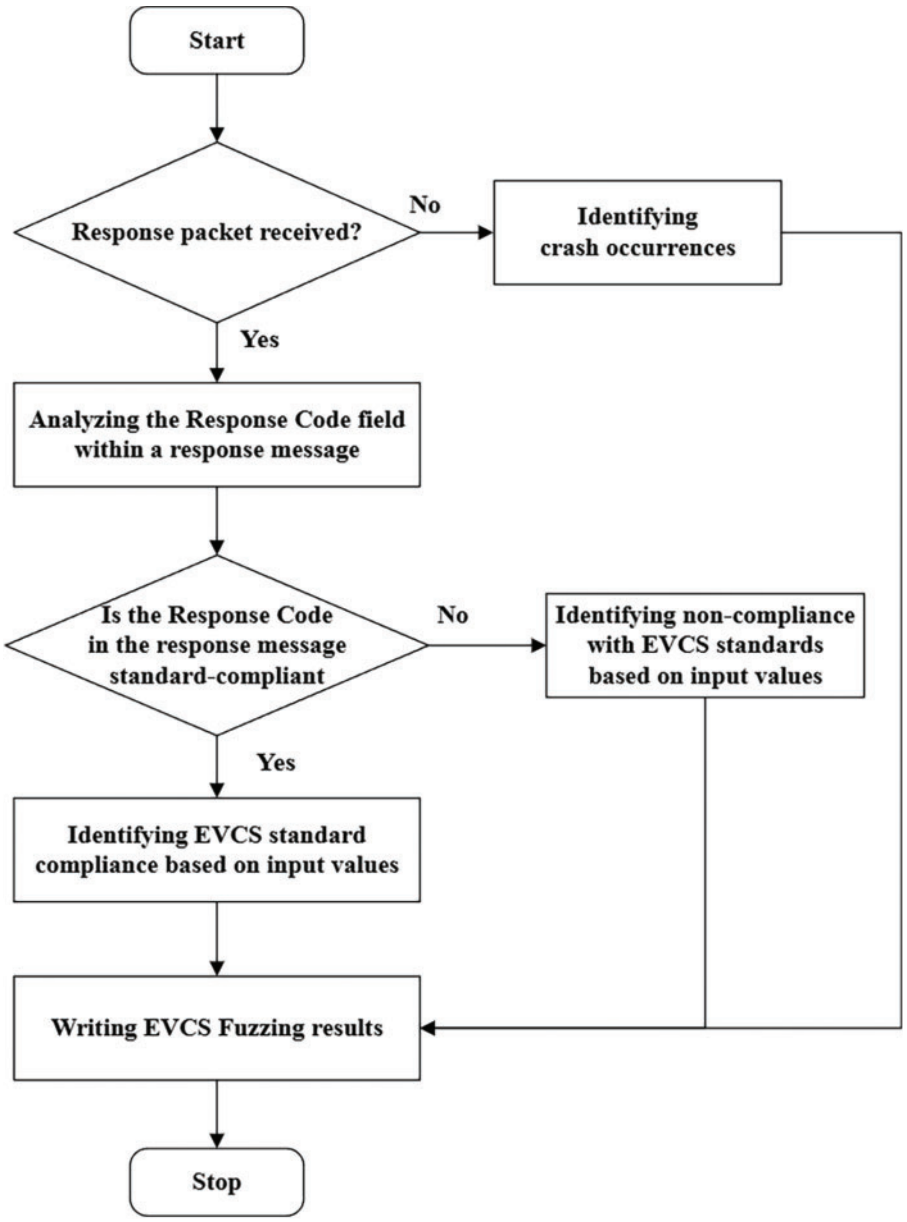

Start
Response packet received?
No
Identifying crash occurrences
Yes
Analyzing the Response Code field within a response message
Is the Response Code in the response message standard-compliant
No
Identifying non-compliance with EVCS standards based on input values
Yes
Identifying EVCS standard compliance based on input values
Writing EVCS Fuzzing results
Stop

Figure 6: Identifying EVCS vulnerabilities

During fuzzing, if an EVCS responds with "FAILED" despite receiving a standard-compliant message, or if it responds with "OK" to a message that violates the standard, it suggests that the EVCS implementation does not fully adhere to the ISO 15118 standard. Such instances of noncompliance may expose additional vulnerabilities, which are detected and recorded during the response analysis phase. Additionally, if a crash occurs or a response that does not conform to the standard is observed, the user can manually review the input values and re-execute the test to verify whether it was a false positive.

Moreover, if no response message is received or if the session terminates abnormally, this could indicate that the EVCS firmware has encountered an error, potentially leading to a system crash. Crashes present significant risks to software stability and the security of the EVCS. Therefore, it is crucial to analyze the specific characteristics of the messages that triggered the crash, as well as the EVCS's response. In particular, recording the field values of the messages that caused the crash, along with the applied mutation patterns, provides valuable insights into the vulnerabilities of the EVCS and helps identify potential attack vectors.

Response analysis facilitates a comprehensive evaluation of both the normal and abnormal operations of the EVCS. This dual approach not only verifies compliance with the ISO 15118 standard but also helps detect and analyze security weaknesses, such as message processing errors or crashes. The findings from this analysis provide essential data to improve the security and stability of the EVCS.

4 Implementation and Evaluation Metrics

This section outlines the architecture of the implemented fuzzing tool and the evaluation metrics used to assess the fuzzing tests. These metrics not only help identify potential vulnerabilities but also evaluate compliance with the relevant standards. They are designed to pinpoint specific messages where vulnerabilities may occur and assess how well the implementation aligns with ISO 15118.

4.1 Electric Vehicle Charging Station Fuzzing Architecture

The fuzzing tool developed for testing the EVCS consists of five key components: the device connector, error detector, constraint manager, message mutator, and state machine manager (Fig. 7). The tool follows a black-box approach, meaning fuzzing is performed using only the input and output values of the EVCS without requiring knowledge of its internal implementation. This design ensures compatibility across different programming languages and operating systems used to implement the ISO 15118 protocol in EVCS systems.

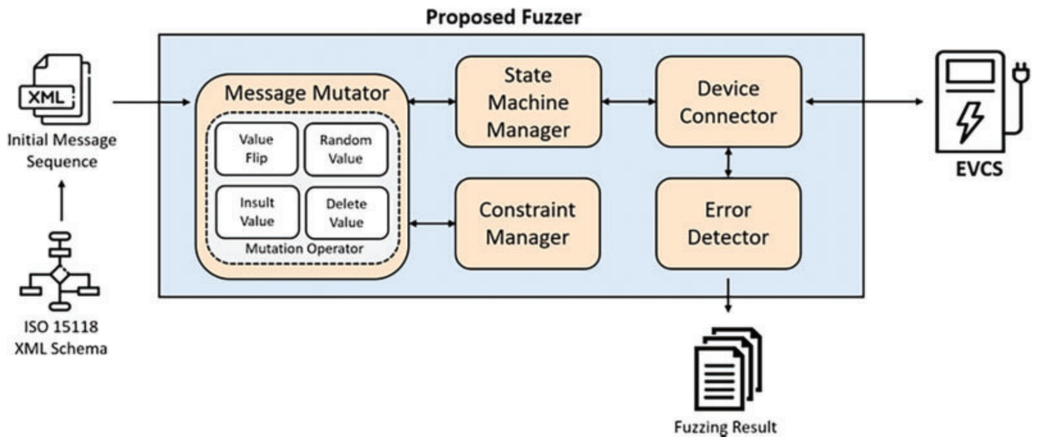


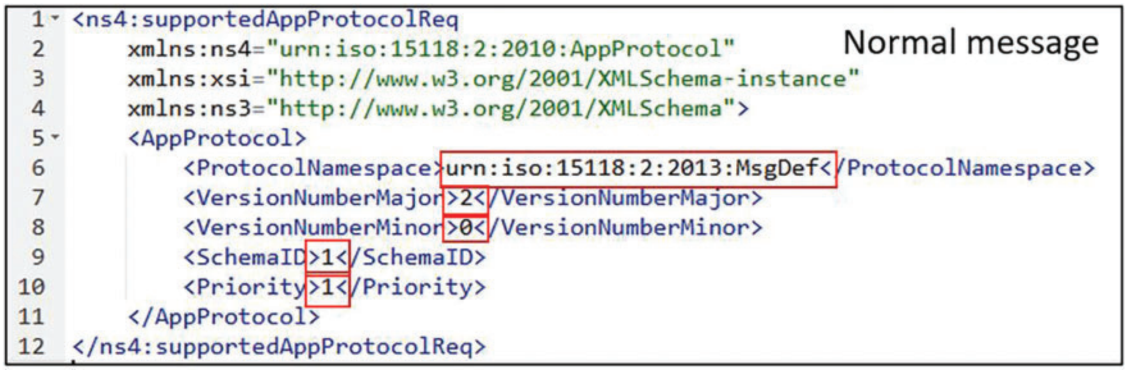
Figure 7: Electric vehicle charging station (EVCS) fuzzing architecture

This architecture ensures comprehensive and consistent testing across different EVCS models, regardless of their specific implementations. By focusing on external communication interfaces and strictly following ISO 15118, the fuzzing tool effectively detects security vulnerabilities while verifying compliance. The integration of these components facilitates a systematic exploration of the EVCS state machine, enhancing the security and reliability of EV charging infrastructure. Additionally, the EVCS fuzzer has been made available on GitHub [54].

4.2 Evaluation Metrics

The specific metrics were adapted by benchmarking those used in the OCPP Storm study introduced in related research. Unlike other fuzzing metrics, the metrics used in this study focus on assessing the extent to which the standard has been implemented, making them particularly relevant for evaluating protocol compliance. Therefore, this aspect was incorporated into our research. Five primary evaluation metrics were selected to assess the results of the fuzzing tests. The total number of messages sent to the EVCS is denoted as M_T, which includes both standard-compliant and non-compliant messages. In this study, messages that adhere to the constraints defined in the standard are classified as normal, while those that do not are classified as abnormal.

As shown in Fig. 8, the SupportedAppReq message serves as an example of a normal message. Before initiating communication, the EV transmits this message to the EVCS to verify the supported communication versions. This message consists of five fields: ProtocolNameSpace, VersionNumberMajor, VersionNumberMinor, SchemaID, and Priority. Table 3 provides a detailed overview of the constraints associated with these fields. A message is considered normal only if all field values comply with constraints regarding type, length, and value limits.



```
<ns4:supportedAppProtocolReq
    xmlns:ns4="urn:iso:15118:2:2010:AppProtocol"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ns3="http://www.w3.org/2001/XMLSchema">
    <AppProtocol>
        <ProtocolNamespace>urn:iso:15118:2:2013:MsgDef</ProtocolNamespace>
        <VersionNumberMajor>2</VersionNumberMajor>
        <VersionNumberMinor>0</VersionNumberMinor>
        <SchemaID>1</SchemaID>
        <Priority>1</Priority>
    </AppProtocol>
</ns4:supportedAppProtocolReq>
```

Figure 8: SupportedAppReq normal message

Table 3: SupportedAppProtocolReq message elements

Element	Type	Length	Data description
ProtocolNameSpace	String	100	Specific protocols supported by the EVCS
VersionNumberMajor	UnsignedInt	–	Major version number of the protocol
VersionNumberMinor	UnsignedInt	–	Minor version number of the protocol
SchemaID	UnsignedByte	–	SchemaID assigned to the protocol
Priority	UnsignedByte	1–20	Protocol priority

Under normal conditions, the EVCS processes both normal and abnormal messages according to the standard. If a message complies with the standard, the EVCS responds with an "OK" response code, which is classified as a normal response. Conversely, if the EVCS encounters an error or an exceptional condition, it responds with a "FAILED" response code. The response code field in the response message, which follows an enumeration format with 26 possible values, indicates the processing status. Typically, normal messages elicit an "OK" response, while abnormal messages elicit a "FAILED" response, except in 24 specific cases where different standard-defined responses apply.

By analyzing these response values, the fuzzing tool determines whether the EVCS implementation adheres to ISO 15118. The tool verifies compliance by ensuring that normal messages receive "OK" responses and abnormal messages receive "FAILED" responses. Additionally, the tool monitors cases where response messages are not received or sessions terminate abnormally, which may indicate firmware errors leading to system crashes. Such crashes pose significant risks to the security and stability of the EVCS, necessitating a detailed analysis of the messages that triggered the crash and the system's response. The following sections introduce the evaluation metrics used to assess the EVCS based on its input and response messages.

4.2.1 Correct Response Rate

The correct response rate measures the proportion of messages that elicited standard-compliant responses. A well-implemented EVCS, compliant with ISO 15118, should achieve a correct response rate of 100%.

A response is considered correct if:

1. The EVCS returns a normal response upon receiving a normal message.
2. The EVCS returns an error response upon receiving an abnormal message.

The number of input messages that receive correct responses is denoted as M_{C}. The correct response rate is calculated by dividing M_{C} by the total number of messages M_{T}, as follows:

$$Correct\,Response\,Rate = \frac{M_C}{e_T}. \tag{1}$$

4.2.2 Incorrect Response Rate

The incorrect response rate represents the proportion of messages that received responses that do not comply with the standard. It is the inverse of the correct response rate, meaning that the sum of the correct and incorrect response rates should equal 100%.

According to the standard:

- Normal messages should receive normal responses.
- Abnormal messages should receive error responses.

However, if the EVCS returns incorrect response values, the number of such instances is denoted as M_{IC}. A higher incorrect response rate indicates lower compliance with ISO 15118 and a greater potential for vulnerabilities. The incorrect response rate is calculated as follows:

$$Incorrect\,Response\,Rate = \frac{M_{IC}}{M_T}. \tag{2}$$

The incorrect response rate can be further categorized into three sub metrics: valid request-causing error rate, non-error-causing fuzz rate, and crash-causing rate.

4.2.3 Valid Request-Causing Error Rate

The valid request-causing error rate quantifies how often the EVCS incorrectly processes a normal message, responding with an error response instead of a normal response. Among the incorrectly processed messages (M_{IC}), the subset of normal messages that received a non-compliant response is denoted as M_{CE}. This metric is calculated as follows:

$$Valid\,Request\text{–}Causing\,Error\,Rate = \frac{M_{CE}}{M_T}. \tag{3}$$

A higher valid request-causing error rate indicates poor implementation of EVCS communication protocols. If an EVCS frequently rejects normal communication attempts, it suggests difficulty in maintaining standard-based communication, potentially disrupting EV charging operations. Even a single occurrence of M_{CE} can indicate a significant issue in standard compliance.

4.2.4 Nonerror-Causing Fuzz Rate

The non-error-causing fuzz rate represents the proportion of abnormal messages that the EVCS incorrectly processes as normal messages, returning an "OK" response instead of an error response. Among the incorrectly processed messages (M_{IC}), the subset of abnormal messages that received an incorrect normal response is denoted as M_R. This metric is calculated as follows:

$$Nonerror\text{–}Causing\,Fuzz\,Rate = \frac{M_R}{M_T}. \tag{4}$$

A high non-error-causing fuzz rate suggests that the EVCS is failing to properly handle invalid messages, which may expose vulnerabilities. Abnormal messages should be properly rejected according to the standard, and failure to do so may indicate exploitable weaknesses in the system.

4.2.5 Crash-Causing Rate

The crash-causing rate measures the proportion of normal and abnormal messages that trigger software failures, causing the EVCS to crash and become unresponsive.

A crash occurs when a software error forces the EVCS system to stop functioning. The number of input messages that cause such crashes is denoted as M_{CC}. The crash-causing rate is calculated by dividing M_{CC} by the total number of messages M_T, as follows:

$$Crash\text{–}Causing\,Rate = M_{CC}/M_T. \tag{5}$$

This metric is crucial for assessing the stability and security of the EVCS. System crashes can lead to severe issues, including downtime, DoS attacks, and security vulnerabilities such as buffer overflows or null pointer dereferences. Identifying messages that trigger crashes helps mitigate critical security risks.

5 Experimental Evaluation

The proposed fuzzing method and tool were validated through an experiment. This section presents the fuzzing tests conducted as a case study on open-source software that simulates ISO 15118 communications, followed by an analysis of the results. The primary objective is to demonstrate the effectiveness of the fuzzing tool in testing open-source software for EVCS communication and to assess its potential applicability to real-world electric vehicle charging stations.

5.1 Experiment Setup

This section outlines the fuzzing target and the experimental environment in which the fuzzing method was applied.

5.1.1 Experimental Target

We explored open-source EV chargers for testing and found that EV@Scale, a consortium in the United States, is actively researching EV charging infrastructure. In particular, Idaho National Laboratory is focusing on EV charger security research. Idaho utilizes the AcCCS [55] open-source platform to analyze real-world EV charger vulnerabilities, making it a highly relevant testing environment. Given this, we determined that testing with AcCCS would provide the most realistic assessment and conducted our experiments accordingly. However, AcCCS does not fully replicate real-world environments, which remains a limitation of this paper.

For this experiment, the hardware dependencies of AcCCS were removed, and the software was modified to enable communication without requiring physical hardware. The study assumed that AcCCS software functioned as the communication firmware installed in an actual EVCS and conducted fuzzing tests accordingly to detect potential vulnerabilities in real-world EVCS communication. The primary goal was to evaluate whether AcCCS accurately implements the ISO 15118-2 standard.

5.1.2 Experimental Environment

A fuzzing tool was developed for the EVCS, and the experimental environment was designed to logically resemble a scenario where an EV and EVCS are physically connected. While PLC was not implemented, a virtualized local network was used to replicate the communication structure.

The fuzzing test results from this experimental setup demonstrated effectiveness comparable to tests conducted on actual EVCSs connected via power-line networks. These results validate the applicability of the experimental findings to real-world settings.

As illustrated in Fig. 9, the experimental environment consisted of two Linux virtual machines (VMs) running on VMware Workstation. One VM hosted the fuzzing tool, emulating an EV, while the other hosted AcCCS, simulating an EVCS. The virtual machines were configured as follows:

- Fuzzing Tool (EV emulator)

MAC address: 00:1e:c0:f2:6c:a1

Ipv6 address: fe80::21e:c0ff:fef2:6ca1

- AcCCS (EVCS emulator)

MAC address: 00:1e:c0:6c:a0

Ipv6 address: fe80::21e:c0ff:fef2:6ca0.

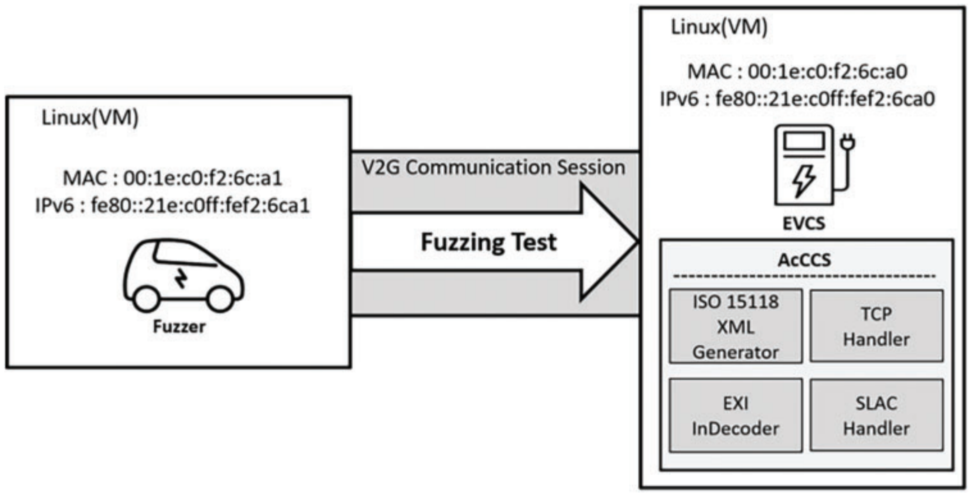


Figure 9: Experimental environment

Both VMs were connected to the same local network, allowing the fuzzing tool to send test messages to AcCCS. Each VM was allocated four processors and 2 GB of RAM to ensure stable execution of the fuzzing tests and packet processing. The network adapters were set to bridge mode to replicate real-world PLC-based networks, ensuring that the Ipv6-based communication closely mirrored actual EVCS communication environments.

5.2 Analysis by Metrics

Fuzzing tests were conducted on AcCCS within the configured experimental environment, and fuzzing was performed on AcCCS, with the results presented in Fig. 10. In the experimental environment, approximately 100,000 test cases were executed. While a larger number of test cases could have been applied, the test set was limited to 100,000 due to the significant increase in testing time when considering the various input parameters. The correct response rate, representing the actual testing process took approximately seven hours, with additional time required due to the need for manual verification of inputs and responses, as well as re-execution in cases of crashes or errors. The proportion of input messages that received standard-compliant responses was 49.53%. This value is significantly lower than the ideal 100%, which would indicate full compliance with ISO 15118. Conversely, 50.47% of input messages elicited noncompliant responses, categorized into the following:

- Valid request-causing error rate: 0%;
- Nonerror-causing fuzz rate: 49.51%; and
- Crash-causing rate: 0.96%.

The results indicate that AcCCS responded correctly to standard messages at a high rate, demonstrating its ability to communicate effectively with EVs under normal conditions. However, the system failed to generate error responses for abnormal messages, instead issuing standard responses, revealing a weakness in its exception-handling mechanisms. While this behavior may enhance flexibility in communication, it also introduces security risks by leaving potential vulnerabilities unaddressed. Additionally, specific message fields triggered crashes, regardless of whether the messages were correctly formatted. Among the 11 tested message types, only one fully complied with the standard. The primary reason for this noncompliance was improper handling of abnormal messages rather than issues with normal message processing. Fig. 11 shows the ratio of Correct and Incorrect rates for each input message.

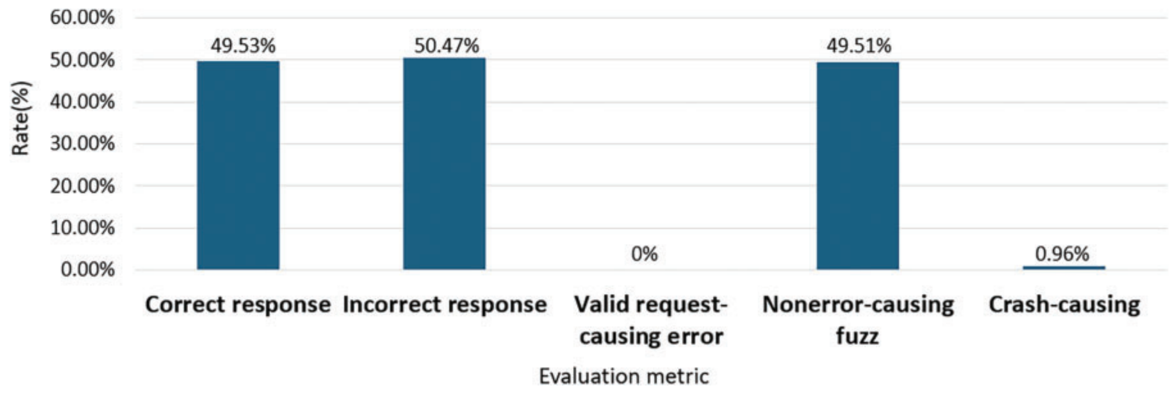


Figure 10: Fuzzing test metrics

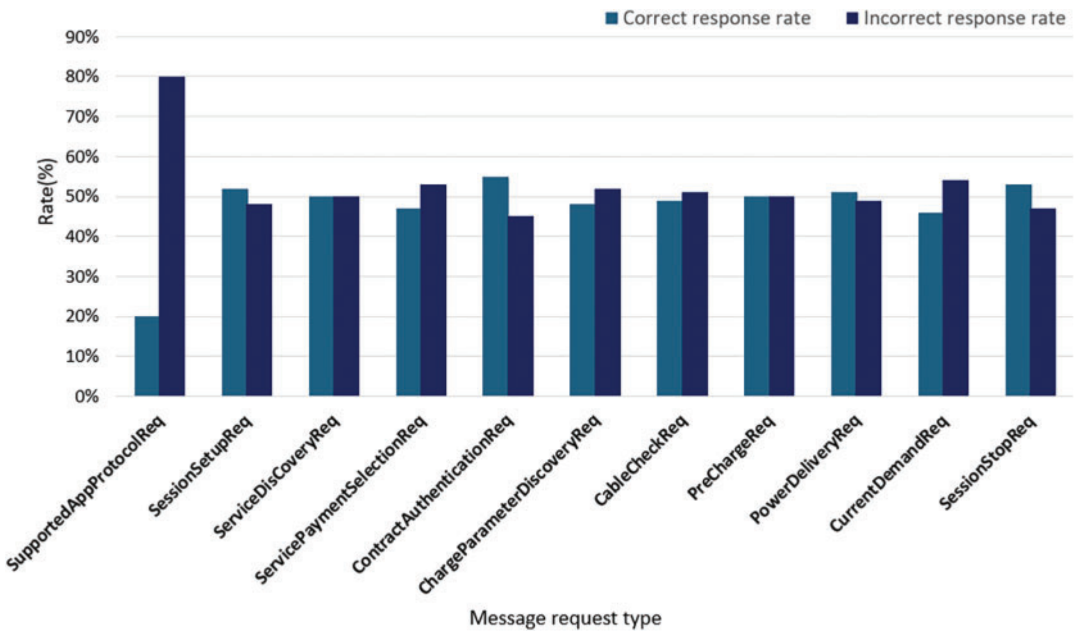


Figure 11: Evaluation results of standard conformance rates

Notably, the SupportedAppProtocolReq message exhibited vulnerabilities that were not detected through existing conformance tests. Similarly, the CurrentDemandReq message lacked test cases for incorrect values, as existing tests only verified responses for appropriate voltage and current levels. This finding highlights the effectiveness of the proposed fuzzing tool in uncovering vulnerabilities that conventional conformance tests overlook. The identified vulnerabilities are not structural flaws within the ISO 15118 protocol itself but rather issues arising from the incorrect adherence to the protocol's specified constraints and inadequate exception handling during implementation. As a result, the fuzzing tool identified five distinct vulnerabilities in AcCCS. Table 4 summarizes these findings, detailing standard compliance, vulnerable elements, the number of identified vulnerabilities, and whether existing conformance tests could detect them.

Table 4: List of identified vulnerabilities by message

Message	Standard compliance	Vulnerable elements	No. of vulnerabilities	Identifiable by conformance tests
SupportedAppProtocolReq	×	Protocol NameSpace	3	×
SessionSetupReq	×	Session ID	1	○
ServiceDisCoveryReq	×	–	0	–
ServicePayment SelectionReq	×	–	0	–
Contract AuthenticationReq	○	–	0	–
ChargeParameter DiscoveryReq	×	–	0	–
CableCheckReq	×	–	0	–
PreChargeReq	×	–	0	–
PowerDeliveryReq	×	–	0	–
CurrentDemandReq	×	EVTargent Current, EVTarget Voltage	1	×
SessionStopReq	×	–	0	–

5.3 Vulnerability Analysis

Several vulnerabilities in AcCCS were identified in messages categorized under the non-error-causing fuzz rate. These vulnerabilities stemmed from the system's failure to handle abnormal messages properly, exposing significant security risks. Some vulnerabilities resulted in system crashes, where specific input values caused AcCCS to become unresponsive, indicating a critical flaw in its ability to process unexpected or malformed inputs. To assess the risks associated with the identified vulnerabilities, a risk assessment was conducted using the evaluation framework shown in Table 5. The assessment includes Impact and Exploitability as key evaluation criteria. Impact measures the extent of the vulnerability's effect by determining whether it affects only a single charger or extends to the entire infrastructure. The classification is divided into Low, Medium, and High based on the scope of potential damage. Exploitability evaluates how the vulnerability can be accessed, considering whether the attack requires physical, local, or remote access. The evaluation results based on these criteria are presented in Table 6. The following subsections provide a detailed description of each vulnerability.

Table 5: Risk assessment metric

Assessment metric	L (Low)	M (Medium)	H (High)
Impact	Affects only an EVCS	Affects multiple EVCS in a localized area.	Affects multiple EVCSs, entire charging networks, or national infrastructure.
Exploitability	Physical access	Local access	Remote access

Table 6: Risk assessment of identified vulnerabilities

Vulnerability	Type	Impact	Exploitability
Weak session authentication and integrity	Improper authentication	M	M
Billing discrepancies due to inaccurate metering data	Improper input validation	H	L
Parameter manipulation causing runtime errors	Resource exhaustion	M	L

5.3.1 Weak Session Authentication and Integrity

In this study, AcCCS failed to properly validate session IDs, which are essential for identifying and managing charging sessions between EVs and EVCSs. Session IDs enable the EVCS to track active charging sessions and ensure communication integrity. If a session ID changes during communication, the EVCS should respond with the response code "FAILED_UnknownSession" to indicate an issue. However, AcCCS did not enforce this Validation and instead treated altered session IDs as normal, responding as if the communication were valid.

This lack of proper session ID validation introduces several serious vulnerabilities. One major risk is session ID collisions, which could allow a session termination command from one EV to interrupt the charging session of another. Additionally, session ID conflicts could lead to log entries overwriting one another in the EVCS, undermining log integrity. Typically, each EV is assigned a unique session ID to track its charging session, such as EV-1 using session ID 01 and EV-2 using session ID 02. The EVCS relies on these IDs to manage charging interruptions, session terminations, and log records. However, due to AcCCS's inadequate validation, multiple EVs could end up sharing the same session ID.

For instance, as illustrated in Fig. 12, if two EVs, such as EV-1 and EV-2, are assigned the same session ID, unintended consequences may occur. For instance, if EV-1 sends a session termination message, it could inadvertently stop EV-2. This flaw not only disrupts charging sessions but also creates the potential for incorrect billing, as multiple users might be charged under the same session. Additionally, session log entries could collide due to ID conflicts, compromising the integrity of the charging records. These issues highlight the critical need for robust session ID validation to prevent session mismanagement, service interruptions, and security vulnerabilities in EVCSs.

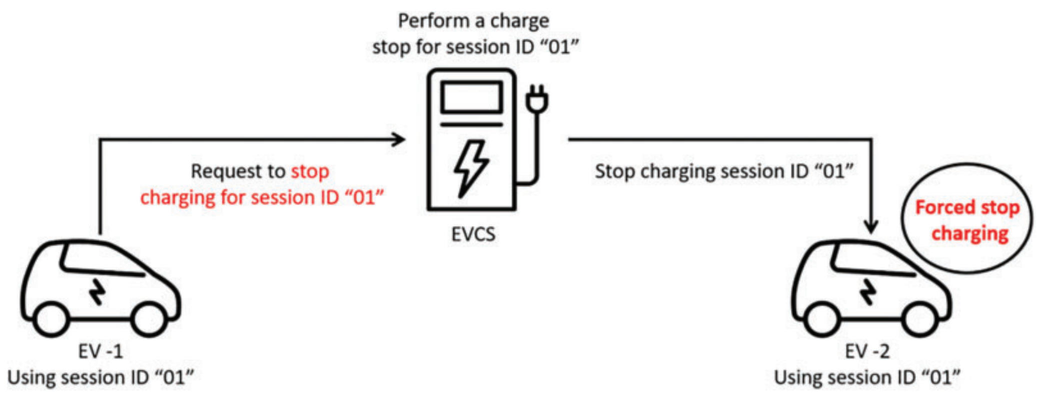

Perform a charge stop for session ID "01"
Request to stop charging for session ID "01"
EVCS
Stop charging session ID "01"
Forced stop charging
EV -1
Using session ID "01"
EV -2
Using session ID "01"

Figure 12: Session interruption vulnerability due to weak session authentication

5.3.2 Billing Discrepancies Due to Inaccurate Metering Data

When Evs receive power from an EVCS, they send a CurrentDemandReq message specifying the requested current and voltage values. In this message, the EVTargetCurrent and EVTargetVoltage fields represent current and voltage levels, respectively. The EVTargetCurrent field is measured in amperes, with a defined range of 0 to 400 A, while the EVTargetVoltage field is measured in volts, with a range of 0 to 100 V.

In this study, AcCCS failed to handle abnormal messages that violated these constraints. When messages containing values outside the defined limits were input, AcCCS accepted them without proper exception handling. This included accepting negative values for both the current and voltage fields. Since billing information in the EVCS is generated based on power consumption, calculated as voltage multiplied by current, and applied to the measured rate per kilowatt-hour (kWh), processing negative values resulted in negative power calculations, leading to incorrect billing.

This vulnerability suggests that an attacker could intentionally send messages with negative current and voltage values to manipulate billing records. Furthermore, improper handling of the EVTargetCurrent and EVTargetVoltage fields may introduce additional vulnerabilities within the EVCS software. These flaws could be exploited to alter billing information or disrupt the charging process, posing significant security risks to both EVCS operators and EV users.

5.3.3 Parameter Manipulation Causing Runtime Errors

Fuzzing tests on AcCCS identified Table 7 details the three types of crashes: memory exhaustion, index overflow, and parsing errors, all of which could disrupt AcCCS operations and potentially lead to DoS attacks. The primary cause of these crashes was linked to the ProtocolNameSpace element in the AppProtocolSupportedReq message, while other components did not exhibit similar issues. Although ProtocolNameSpace was designed as a string element with a maximum length of 100 characters, AcCCS failed to properly handle special characters within the string, leading to parsing errors. During XML data processing, special characters such as <, > triggered parsing failures, and the absence of adequate exception handling caused the program to terminate unexpectedly.

Table 7: List of runtime errors

Error type	Element	Cause
Index of error	Protocol NameSpace	Element value $\geq$ 30 characters
Parse error		Special symbols
Out of memory		Mass data input

An index overflow error occurred when the ProtocolNameSpace value exceeded 30 characters, despite the specification allowing up to 100 characters. This issue originated from an implementation flaw in AcCCS that failed to enforce the correct length constraint.

Additionally, memory exhaustion errors were caused by continuous data input, which overwhelmed the system's memory resources. This suggests that the EXI decoding module of AcCCS was incapable of reliably processing sustained input. These error types were attributed to insufficient exception handling and flawed implementation, even though AcCCS formally adhered to standard constraints. Similar errors were also observed during fuzzing tests conducted without constraints, indicating that even implementations meeting formal specifications may contain latent vulnerabilities.

6 Limitations and Discussion

This study introduces a state machine-based fuzzing technique for identifying vulnerabilities in ISO 15118 implementations of EVCSs. While the proposed method effectively detects security flaws, several limitations must be acknowledged, along with areas for future research.

First, the experiments were conducted using a single open-source implementation, AcCCS, which limits the validation of the method's effectiveness. The absence of testing on commercial charging firmware or a broader range of open-source projects restricts the demonstration of its comprehensive applicability. Future research should incorporate experiments on both commercial firmware and additional open-source implementations to ensure broader validation.

Second, although fuzzing was applied across all state machines, vulnerabilities were analyzed individually within each state machine, overlooking potential security risks arising from interactions between states. Issues occurring during state transitions due to complex program logic were not fully explored. The black-box nature of the approach further complicates the detection of such interactions, highlighting the need for complementary techniques to uncover these vulnerabilities.

Finally, although this study conducted testing using black-box fuzzing to ensure platform and implementation independence, the identified vulnerabilities do not necessarily represent all potential security flaws. Therefore, integrating other testing techniques or comparing the proposed approach with alternative methodologies could further enhance and refine the effectiveness of security testing for ISO 15118 implementations.

7 Conclusion

This study presents a state machine-based fuzzing technique for detecting security vulnerabilities in ISO 15118 implementations of EVCSs. By designing tests based on EVCS communication protocol characteristics and applying message alterations at the application layer, the proposed method effectively identified compliance issues and security vulnerabilities. Given the complexity of ISO 15118 and the state machine structure of EVCSs, this fuzzing technique enabled efficient testing across different states. The tool was designed to function independently of the programming language and operating system in EVCS implementations by adopting a black-box approach, making it applicable across various platforms. The method successfully identified vulnerabilities related to non-compliant message processing, crashes, and insufficient exception handling. Fuzzing tests conducted on an open-source implementation identified five vulnerabilities, demonstrating the effectiveness of the proposed approach in uncovering security flaws and improving the robustness and reliability of EVCS infrastructure. However, since this fuzzing technique follows a black-box approach, conducting an in-depth vulnerability analysis may be challenging, and detecting vulnerabilities deeper in the code could be difficult. Therefore, further improvements to the methodology are necessary, and this remains a subject for future research. Regarding future research and adaptability, the structural design and mechanism of the fuzzing tool have been developed to ensure sufficient scalability as the standard evolves or new specifications such as ISO 15118-20 are introduced. However, additional research will be necessary to fully enable this adaptability.

Acknowledgement: Not applicable.

Funding Statement: This research was conducted with the support of the Korea Internet & Security Agency (KISA)—Information Security Specialized University Support Project (50%) and supported by a grant from the Korea Electric Power Corporation (R24XO01-4, 50%) for basic research and development projects starting in 2024.

Author Contributions: The authors confirm their contributions to the study as follows: Conceptualization, Yu-Bin Kim and Dong-Hyuk Shin; methodology, Yu-Bin Kim and Ieck-Chae Euom; validation, Dong-Hyuk Shin and Ieck-Chae Euom; investigation, Yu-Bin Kim and Dong-Hyuk Shin; writing—original draft preparation, Yu-Bin Kim, Dong-Hyuk Shin, and Ieck-Chae Euom; writing—review and editing, Yu-Bin Kim and Ieck-Chae Euom; project administration, Ieck-Chae Euom; funding acquisition, Ieck-Chae Euom. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The tools proposed in this study are available at https://github.com/kingyoubin/EVC-Fuzzer-project/ (accessed on 24 December 2024).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

Appendix A

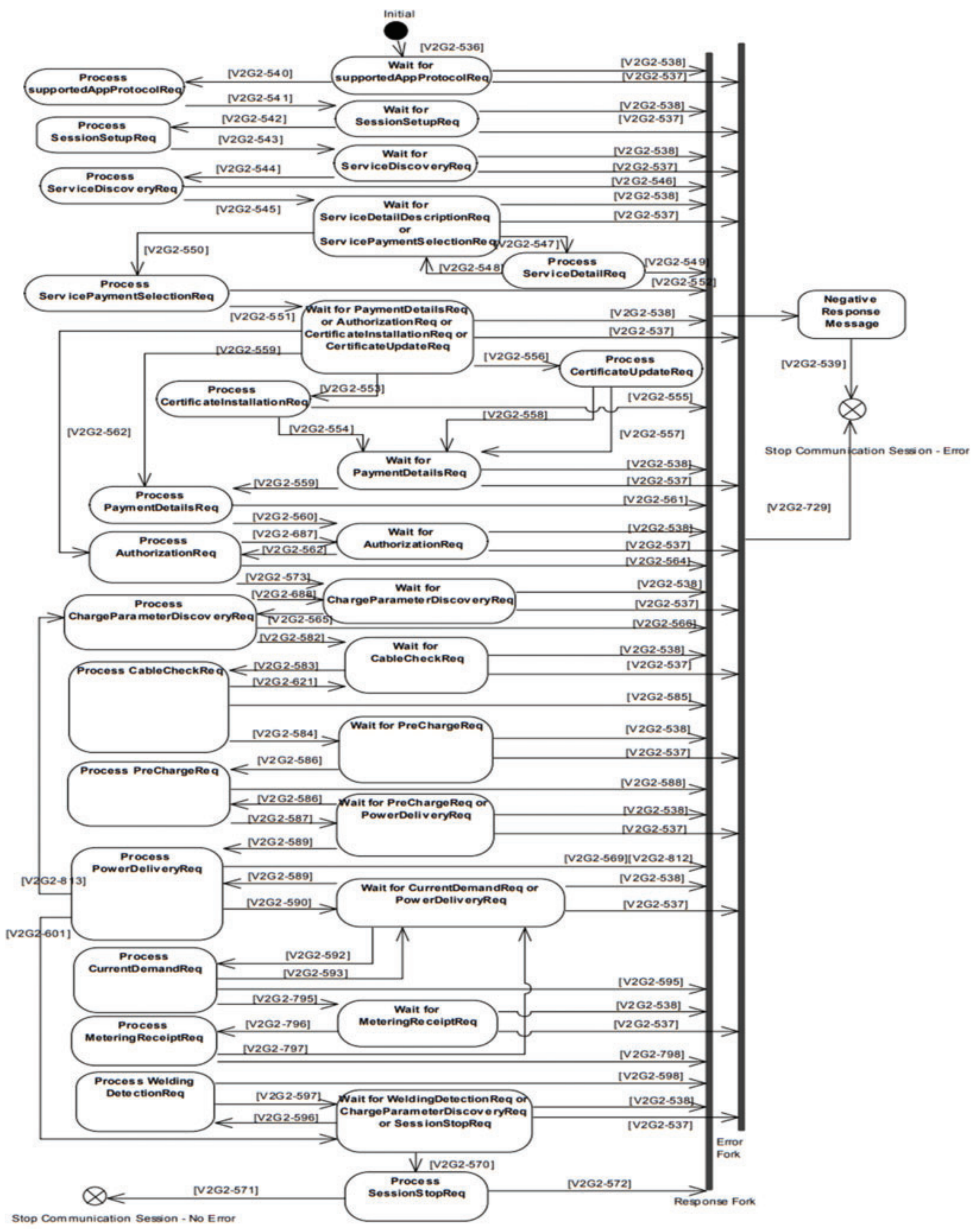


Figure A1: Electric vehicle charging station (EVCS) state machine

References

1. National Geographic. Electric cars may rule the world's roads by 2040 [Internet]. [cited 2024 Dec 21]. Available from: https://www.nationalgeographic.com/news/2017/09/electriccars-replace-gasoline-engines-2040.
2. Statista. Global public electric vehicle charging station market size between 2016 and 2022, with a forecast through 2028 [Internet]. [cited 2024 Dec 21]. Available from: https://www.statista.com/statistics/1359545/global-ev-charging-market-volume-forecast.
3. MarketsandMarkets. [cited 2024 Dec 21]. Available from: https://www.marketsandmarkets.com.
4. ISO 15118-1:2013. Road vehicles—vehicle to grid communication interface—part 1: general information and use-case definition. Geneva, Switzerland: International Organization for Standardization; 2013.
5. Open Charge Alliance. Open charge point Protocol 1.6 (OCPP 1.6). The Hague, The Netherlands: Open Charge Alliance; 2015.
6. Kern D, Krauß C, Hollick M. Attack analysis and detection for the combined electric vehicle charging and power grid domains. In: Proceedings of the 19th International Conference on Availability, Reliability and Security; 2024 Jul 30–Aug 2; Vienna, Austria. doi:10.1145/3664476.3664512.
7. Sarieddine K. Bolstering EV charging ecosystem infrastructure resilience and unraveling threats—a comprehensive study [dissertation]. Montreal, QC, Canada: Concordia University; 2024.
8. Wang F, Zhuge C, Chen A. Data-driven vulnerability analysis of shared electric vehicle systems to cyberattacks. Transp Res Part D Transp Environ. 2024;135:104379. doi:10.1016/j.trd.2024.104379.
9. Hu X, Jiang X, Zhang J, Wang S, Zhou M, Zhang B, et al. Electric vehicle charging network security: a survey. J Syst Archit. 2025;159:103337. doi:10.1016/j.sysarc.2025.103337.
10. Brighente A, Conti M, Donadel D, Turrin F. EVScout2.0: electric vehicle profiling through charging profile. ACM Trans Cyber-Phys Syst. 2024;8(2):1–24. doi:10.1145/3565268.
11. Lee H, Shin M. TestShark: a passive conformance testing system for ISO 15118 using wireshark. Energies. 2024;17(23):5833. doi:10.3390/en17235833.
12. Sagdullaev M. Fleet charging infrastructure resilience to cybersecurity threats [master's thesis]. Cincinnati, OH, USA: University of Cincinnati; 2024.
13. Acharya S, Khan HAU, Karri R, Dvorkin Y. MaDEVIoT: cyberattacks on EV charging can disrupt power grid operation. In: 2024 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT); 2024 Feb 19–22; Washington, DC, USA. doi:10.1109/ISGT59692.2024.10454199.
14. Vailoces G, Keith A, Almehmadi A, El-Khatib K. Securing the electric vehicle charging infrastructure: an in-depth analysis of vulnerabilities and countermeasures. In: Proceedings of the Int'l ACM Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications; 2023 Oct 30–Nov 3; Montreal, QC, Canada. p. 31–8. doi:10.1145/3616392.3623424.
15. Szakály M, Köhler S, Martinovic I. Current affairs: a measurement study of deployment and security trends in EV charging infrastructure. arXiv:2404.06635. 2024. doi:10.48550/arXiv.2404.06635.
16. Antoun J, Diop S, El-Mougy A, Ibnkahla M. A detailed security assessment of the EV charging ecosystem. IEEE Netw. 2020;34(3):200–7. doi:10.1109/MNET.001.1900348.
17. Sultana Z, Basha CH, Irfan MM. Communication protocols for electric vehicles: a comprehensive analysis. In: 2024 4th International Conference on Sustainable Expert Systems (ICSES); 2024 Oct 15–17; Kaski, Nepal. p. 127–33. doi:10.1109/ICSES63445.2024.10763347.
18. Idaho National Laboratory. EV SALaD 2023 demonstration best practices and mitigations for protecting EVSE infrastructure [Internet]; 2024. [cited 2024 Dec 21]. Available from: https://inldigitallibrary.inl.gov/sites/STI/STI/Sort_130145.pdf;.
19. Johnson J, Anderson B, Wright B, Quiroz J, Berg T, Graves R, et al. Cybersecurity for electric vehicle charging infrastructure. Albuquerque, NM, USA: Sandia National Laboratories (SNL-NM); 2022. Report No.: SAND2022-9315.
20. Hamdare S, Kaiwartya O, Aljaidi M, Jugran M, Cao Y, Kumar S, et al. Cybersecurity risk analysis of electric vehicles charging stations. Sensors. 2023;23(15):6716. doi:10.3390/s23156716.

21. Shirvani S, Baseri Y, Ghorbani A. Evaluation framework for electric vehicle security risk assessment. IEEE Trans Intell Transp Syst. 2024;25(1):33–56. doi:10.1109/TITS.2023.3307660.
22. Terrance W, Kouadio K, Youssef T. Understanding open charge point protocol. In: SoutheastCon 2023. Orlando, FL, USA. 2023 Apr 1–16. p. 559–64. doi:10.1109/SoutheastCon51012.2023.10115127.
23. Elmo D, Fragkos G, Johnson J, Rohde K, Salinas S, Zhang J. Disrupting EV charging sessions and gaining remote code execution with DoS, MITM, and code injection exploits using OCPP 1.6. In: 2023 Resilience Week (RWS). Vol. 31. 2023 Nov 27–30; National Harbor, MD, USA. doi:10.1109/RWS58133.2023.10284654.
24. Rubio JE, Alcaraz C, Lopez J. Addressing security in OCPP: protection against man-in-the-middle attacks. In: 2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS); 2018 Feb 26–28; Paris, France. doi:10.1109/NTMS.2018.8328675.
25. Alcaraz C, Lopez J, Wolthusen S. OCPP protocol: security threats and challenges. IEEE Trans Smart Grid. 2017;8(5):2452–9. doi:10.1109/TSG.2017.2669647.
26. Sanghvi A, Markel T. Cybersecurity for electric vehicle fast-charging infrastructure. In: 2021 IEEE Transportation Electrification Conference & Expo (ITEC); 2021 Jun 21–25; Chicago, IL, USA. p. 573–6. doi:10.1109/itec51675.2021.9490069.
27. Sarieddine K, Ali Sayed M, Torabi S, Attallah R, Jafarigiv D, Assi C, et al. Uncovering covert attacks on EV charging infrastructure: how OCPP backend vulnerabilities could compromise your system. In: Proceedings of the 19th ACM Asia Conference on Computer and Communications Security; 2024 Jul 1–5; Singapore. p. 977–89. doi:10.1145/3634737.3644999.
28. Gebauer L, Trsek H, Lukas G. Evil SteVe: an approach to simplify penetration testing of OCPP charge points. In: 2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA); 2022 Sep 6–9; Stuttgart, Germany. doi:10.1109/ETFA52439.2022.9921430.
29. Bonadonna M, Zhang J, Vu T. Analysis of EVCS switching attack impact on power grid operation. In: 2024 IEEE PES Innovative Smart Grid Technologies Europe (ISGT EUROPE); 2024 Oct 14–17; Dubrovnik, Croatia. doi:10.1109/ISGTEUROPE62998.2024.10863707.
30. Sarieddine K, Ali Sayed M, Torabi S, Attallah R, Assi C. Edge-based detection and localization of adversarial oscillatory load attacks orchestrated by compromised EV charging stations. Int J Electr Power Energy Syst. 2024;156:109735. doi:10.1016/j.ijepes.2023.109735.
31. Dudek S, Delaunay JC, Fargues V. V2G injector: whispering to cars and charging units through the power-line. In: Proceedings of the SSTIC (Symposium sur la sécurité des technologies de l'information et des communications); 2019 Jun 5–7; Rennes, France. p. 5–7.
32. Conti M, Donadel D, Poovendran R, Turrin F. Evexchange: a relay attack on electric vehicle charging system. In: European Symposium on Re-Search in Computer Security; 2022 Sep 26; Copenhagen, Denmark. p. 488–508. doi:10.1007/978-3-031-17140-6_24.
33. Kilic A. TLS-handshake for plug and charge in vehicular communications. Comput Netw. 2024;243:110281. doi:10.1016/j.comnet.2024.110281.
34. Kim Y, Shin D, Euom I. Research on countermeasures against man-in-the-middle attacks through analysis and demonstration of V2G communication protocol security threats. J Inf Secur. 2024;24(4):87–97. doi:10.33778/kcsa.2024.24.4.087.
35. Bao K, Valev H, Wagner M, Schmeck H. A threat analysis of the vehicle-to-grid charging protocol ISO 15118. Comput Sci Res Dev. 2018;33(1):3–12. doi:10.1007/s00450-017-0342-y.
36. Köhler S, Baker R, Strohmeier M, Martinovic I. Brokenwire: wireless disruption of CCS electric vehicle charging. 2022. doi:10.48550/arXiv.2202.02104.
37. Lee S, Park Y, Lim H, Shon T. Study on analysis of security vulnerabilities and countermeasures in ISO/IEC 15118 based electric vehicle charging technology. In: 2014 International Conference on IT Convergence and Security (ICITCS); 2014 Oct 28–30; Beijing, China. doi:10.1109/ICITCS.2014.7021815.
38. Johnson J, Berg T, Anderson B, Wright B. Review of electric vehicle charger cybersecurity vulnerabilities, potential impacts, and defenses. Energies. 2022;15(11):3931. doi:10.3390/en15113931.

39. IEC 61851-1. Electric vehicle conductive charging system—part 1: general requirements. Geneva, Switzerland: International Electrotechnical Commission; 2017.
40. ISO 15118-2. Road vehicles—vehicle to grid communication interface—part 2: network and application protocol requirements. Geneva, Switzerland: International Organization for Standardization; 2014.
41. ISO 15118-3. Road vehicles—vehicle to grid communication interface—part 3: physical and data link layer requirements. Geneva, Switzerland: International Organization for Standardization; 2015.
42. HomePlug Powerline Alliance. HomePlug Green PHY 1.1: the standard for in-house smart grid powerline communications: an application and technology overview [Internet]. [cited 2024 Dec 21]. Available from: https://www.homeplug.org/media/filer_public/98/4e/984e8b2e-5b8e-4f5a-8e5e-2b4e5e5e5e5e/homeplug_gp_1_1_whitepaper.pdf.
43. ISO 15118-4. Road vehicles—vehicle to grid communication interface—part 4: network and application protocol conformance test requirements. Geneva, Switzerland: International Organization for Standardization; 2018.
44. ISO 15118-5. Road vehicles—vehicle to grid communication interface—part 5: physical layer and data link layer conformance test requirements. Geneva, Switzerland: International Organization for Standardization; 2018.
45. Priyasta D, Hadiyanto, Septiawan R, Herminawan F, Bayu H. Ensuring compliance and reliability in EV charging station management systems: a novel testing tool for OCPP 1.6 messages conformance. J Eur Des Systèmes Autom. 2023;56(1):121–9. doi:10.18280/jesa.560116.
46. Coppoletta G, Gjomemo R, Kaur A, Valizadeh N, Rana O, Venkatakrishnan V. OCPPStorm: a comprehensive fuzzing tool for OCPP implementations. In: Proceedings Symposium on Vehicle Security & Privacy; 2024 Feb 26; San Diego, CA, USA. doi:10.14722/vehiclesec.2024.23069.
47. Shin M, Kim H, Kim H, Jang H. Building an interoperability test system for electric vehicle chargers based on ISO/IEC 15118 and IEC 61850 standards. Appl Sci. 2016;6(6):165. doi:10.3390/app6060165.
48. Hänsch K, Pelzer A, Komarnicki P, Gröning S, Schmutzler J, Wietfeld C, et al. An ISO/IEC 15118 conformance testing system architecture. In: 2014 IEEE PES General Meeting | Conference & Exposition; 2014 Jul 27–31; National Harbor, MD, USA. doi:10.1109/PESGM.2014.6938863.
49. Schoneberger T. FUZZ testing the ISO 15118 protocol stack [Internet]. [cited 2024 Dec 21]. Available from: https://cdn.vector.com/cms/content/events/2021/vSES21/vSES21_Slides07_Schoeneberger_Vector.pdf.
50. Kim S, Woo S, Lee H, Oh H. Poster: IoTcube: an automated analysis platform for finding security vulnerabilities. In: Proceedings of the 38th IEEE Symposium on Security and Privacy; 2017 May 22–24; San Jose, CA, USA.
51. Park H, Nkuba CK, Woo S, Lee H. L2Fuzz: discovering bluetooth L2CAP vulnerabilities using stateful fuzz testing. In: 2022 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN); 2022 Jun 27–30; Baltimore, MD, USA. p. 343–54. doi:10.1109/DSN53405.2022.00043.
52. Zalewski M. American fuzzy lop [Internet]. [cited 2024 Dec 21]. Available from: http://lcamtuf.coredump.cx/afl/.
53. Ahn P, Jang Y, Woo S, Lee H. BloomFuzz: unveiling bluetooth L2CAP vulnerabilities via state cluster fuzzing with target-oriented state machines. In: Garcia-Alfaro J, Kozik R, Choraś M, Katsikas S, editors. Computer security—ESORICS 2024. Cham, Switzerland: Springer; 2024. p. 110–29. doi: 10.1007/978-3-031-70896-1_6.
54. GitHub. EVC-Fuzzer Project [Internet]. [cited 2024 Dec 21]. Available from: https://github.com/kingyoubin/EVC-Fuzzer-project.
55. GitHub. IdahoLabResearch/AcCCS [Internet]. [cited 2024 Dec 21]. Available from: https://github.com/IdahoLabResearch/AcCCS.