Power Jacking Your Station: In-Depth Security Analysis of Electric Vehicle Charging Station Management Systems

Tony Nasr[a,*], Sadegh Torabi[a,*], Elias Bou-Harb[b], Claude Fachkha[c], Chadi Assi[a]

[a] *The Cyber Security Research Centre, Concordia Institute for Information Systems Engineering, Montreal, Quebec, Canada*
[b] *The Cyber Center for Security and Analytics, University of Texas at San Antonio, San Antonio, Texas, USA*
[c] *College of Engineering and Information Technology, University of Dubai, Dubai, UAE*

Abstract

The demand for Electric Vehicles (EVs) has been exponentially increasing, and to achieve sustainable growth, the industry dictated rapid development of the supporting infrastructure. This requires building a reliable EV charging ecosystem that serves customer demands while ensuring the security of the Internet-enabled systems and the connected critical infrastructure against possible cyber attacks. To this end, we devise a system lookup and collection approach to obtain a representative sample of widely deployed EV Charging Station Management Systems (EVCSMS). Furthermore, we leverage reverse engineering and penetration testing techniques to perform a first-of-a-kind comprehensive security and vulnerability analysis of the identified EVCSMS and their software/firmware implementations. Indeed, our systematic analysis unveils an array of vulnerabilities, which demonstrate the insecurity of the EVCSMS against remote cyber attacks. Considering the feasibility of such attacks, we discuss attack implications against the EV charging stations (EVCS) and their users. More importantly, we simulate the impact of practical cyber attack scenarios against the power grid, which result in pos-

---

[*] Corresponding Authors

Email addresses: `t_asr@encs.concordia.ca` (Tony Nasr), `sa_tora@encs.concordia.ca` (Sadegh Torabi), `elias.bouharb@utsa.edu` (Elias Bou-Harb), `cfachkha@ud.ac.ae` (Claude Fachkha), `assi@encs.concordia.ca` (Chadi Assi)

sible service disruption and failure in the grid. Finally, while we recommend mitigation measures, our discoveries raise concerns about the lack of adequate security considerations in the design of the deployed EVCS, which will motivate vendors to take immediate action to patch their developed systems. Indeed, our communication with the concerned parties resulted in positive responses from some vendors such as Schneider Electric, who acknowledged our findings by reserving 12 CVEs, respectively.

Keywords: Electric Vehicle (EV), EV Charging Station Management System, security analysis, zero-day vulnerabilities

1. Introduction

The global Electric Vehicle (EV) fleet has been rapidly expanding over the last few years, leading to the development and adoption of supporting technologies and infrastructure. According to a recent report published by the International Energy Agency [1], the total number of EVs has increased to a significant 6.3 million, with more than 800 thousand public charging points and an estimated 6.5 million private chargers. More importantly, the projected increase in the adoption of EVs will result in a linear growth in the numbers of deployed electric vehicle charging stations (EVCS), which are required to fulfill client demands and enhance quality of service. In line with that, the Internet-of-Things (IoT) paradigm led to the improvement of EVCS through the implementation of EVCS management systems (EVCSMS) which allow for extended capabilities such as remote monitoring, management, scheduling and user billing, to name a few.

Although, in general, the IoT paradigm has been shown to be beneficial in different aspects of our lives, its insecurity and the wide range of associated vulnerabilities brought security and management challenges as a major concern to the users and network operators (e.g., Mirai botnet [2]). Similarly, in the context of Internet-connected EVCS, the extended remote functionalities may open doors for an array of cyber attacks against the deployed EVCS, their users as well as directly integrated critical infrastructure such as the power grid.

Therefore, in this paper, we examine the security posture of Internet-connected EVCS through in-depth security analysis of their deployed EVCSMS. We evaluate a representative body of 16 EVCSMS including systems developed by globally recognized vendors such as Schneider Electric, who provides a plethora of EV services that are widely deployed in Europe [3]. We conduct a thorough analysis on the underlying firmware-, web-, and mobile-based EVCSMS, and uncover various zero-day vulnerabilities, which demonstrate the insecurity of the deployed systems. In addition, while we discuss practical attack implications against the EVCS and its users, we demonstrate the feasibility of leveraging the identified vulnerabilities to compromise EVCS and perform frequency instability attacks against the power grid using simulation analysis. Indeed, we raise attention towards the vulnerability of the studied systems against cyber attacks, which calls for prompt actions to protect the EVCS ecosystem against them. To do so, we recommend mitigating countermeasures for such vulnerability-driven attacks.

To this end, we frame the contributions of this paper as follows:

- To the best of our knowledge, we are among the first to perform a comprehensive security analysis of the identified EVCSMS which are developed by various vendors. Specifically, we devise a system lookup and collection approach to identify a large number of EVCSMS, then leverage reverse engineering and white-/black-box web application penetration testing techniques to perform a thorough vulnerability analysis.

- We demonstrate the feasibility of cyber attacks against the deployed EVCS by presenting vulnerabilities which can lead to remote EVCS exploitation and manipulation. In our analysis, we uncover a list of high- and critical-severity security issues for the analyzed EVCSMS such as SQL Injection (SQLi), Cross-Site Scripting (XSS), Server-Side Request Forgery (SSRF), and Cross-Site Request Forgery (CSRF), to name some.

- We discuss practical attack implications against the stakeholders namely the EVCS, their users, and the connected critical infrastructure such as the power grid and the communication networks. Moreover, we simulate attack scenarios against the power grid, where an adversary is assumed to leverage compromised EVCS to perform large-scale attacks with the purpose of causing frequency instability, which result in possible power outages and/or denial of service.

- We highlight a major flaw in the design and implementation of EVCSMS, which have been surprisingly overlooked by EVCS manufacturers and system developers, despite the fact that the identified vulnerabilities have already been addressed in other contexts. Furthermore, while we recommend a list of countermeasures to address these current security issues and strengthen the deployed systems against future attacks, we communicate the findings to the respective EVCSMS vendors/developers. Indeed, our findings were acknowledged by some vendors, among which, Schneider Electric assigned 12 common vulnerabilities and exposures (CVE) IDs [4], respectively.

The rest of the paper is organized as follows. We present background information in Section 2. Then, we present a detailed description of our data collection and system analysis approach in Section 3 and provide in-depth vulnerability analysis of the studied systems and possible attack implications in Section 4. Consequently, we discuss mitigating countermeasures in Section 5. Finally, we discuss the results of our analysis and pinpoint future work in Section 6, then present an overview of related work in Section 7, before concluding the paper by summarizing the main takeaways in Section 8.

2. Background

The EV charging ecosystem is a complex environment aggregated of several entities and components that coordinate in-between to operate and deliver a number of functionalities. In what follows, we present the main components of the EV charging ecosystem.

2.1. Physical Infrastructure

The EV charging ecosystem consists of a number of physical entities such as charging pools which encompass the EVCS [5] and the power grid which encompasses the power plant, transmission medium and distribution facilities. EVCS are normally categorized based on their locations as public (i.e., accessible by everyone) or private (i.e., located at residences), or classified into three major classes (Level 1–3) based on the maximum amount of power that they feed to the EV battery from the grid [6]:

- Level 1: This is the basic charger often found in homes and workplaces which provides power through a standard $120\,V$ AC household outlet with a corresponding power output of 1.5-2 kW. It does not require installation of additional equipment and can deliver 4-5 miles of range per hour of charging. This type of charging is restricted to North America, since the rest of the world uses a $220\,V$ electric supply for their plug-in EV [7].

- Level 2: This is the most common charger which provides power through a $240\,V$ residential connection or $208\,V$ for commercial plug. It requires installation of additional equipment, and can deliver 10-20 miles of range per hour of charging. It allows peak power of 19 kW and outputs on average 7.2 kW of power permitting drivers to fully charge their EVs in a couple of hours [8].

- Level 3: This is most often found in public, especially along heavy traffic corridors, and commonly referred to as DC (Direct Current) fast charging. It provides charging through a $480\,V$ AC input with up to $800\,V$ for some plug-in vehicles, with peak power of up to 240 kW and on average 110 kW. This charger requires specialized high-powered expensive equipment, and can deliver 60-80 miles of range in 20 minutes of charging [9].

2.2. EVCS Management System

The main function of EVCS is to deliver electric power to charge EV batteries on demand, however, advanced EVCS have further features such as Internet connectivity for remote management [10]. These capabilities are driven by components such as the EVCS firmware, the EVCSMS and the open charge point protocol (OCPP). EVCS firmware is an embedded computer software designed to provide low-level control over the charging station's hardware and remains unmodified unless explicitly updated by the EVCS administrator. It is often a package containing a minified operating system (OS) with peripheral libraries and binaries. EVCSMS is a specialized software that provides the EVCS user/operator with the interface and tools to remotely control and manage the operations on the EVCS such as scheduling, charge record-keeping, user authentication, to name a few. Although some of these features can also be managed through physical interfaces such as the human machine interface (HMI) implemented on the EVCS hardware, for this work we focus on examining the software components that allow for remote management over the network. The EVCSMS software can be deployed to instrument the EVCS by either embedding its application user interface into the EVCS firmware (i.e., firmware-based) or in the cloud on a web server (i.e., web-based) or through an application programming interface (API) by means of a mobile application (i.e., mobile-based). As for OCPP [11], it is the de-facto standard application protocol developed by the Open Charge Alliance (OCA) for communication between the EVCS and EVCSMS. In this work, we examine the security of the three types of software EVCSMS products and conduct an in-depth vulnerability analysis on each of them. Specifically, we analyze firmware-, web-, and mobile-based EVCSMS software, respectively.

2.3. Open Charge Point Protocol

With the diversity of EVCS operators, various proprietary protocols have been devised to allow for communication between the EVCS and EVCSMS, which is essential for managing and controlling the various operations and functionalities of EVCS. However, in order to standardize the communication between these two entities, the OCA designed and introduced in 2012 the OCPP [11] which got adopted as the de-facto standard application protocol developed for message exchange between the EVCS and EVCSMS. With each new version release, the protocol received improvements and updates to extend its features and capabilities. For instance, version 1.5 only supported Simple Object Access Protocol (SOAP) messaging protocols and had 24 unique message types. Later on, in version 1.6 support got extended to include both SOAP and JavaScript Object Notation (JSON) and added new functionalities such as smart charging. Then, version 2.0 got published in 2018, offering support for only JSON and got extended to have 65 unique message types as well as added several new features mainly:

- EV-Grid standards: The charging process supervised by the ISO-15118 standard can be remotely managed through the EVCSMS (i.e., start or stop the charging process).
- Remote control: version 2.0 allows for full remote control through the EVCSMS, by letting the charging station users and operators monitor and manage the EVCS in real time (e.g., change configurations, start/stop charging, unlocking the connector).

In prior work from the literature, several studies [12, 13] have examined the security of OCPP. Specifically, they presented vulnerabilities that open door for manipulator-in-the-middle (MITM) attacks as elaborated in Section 7 (Related Work). However, despite OCPP being a viable attack surface, in this work we focus on examining the insecurity of EVCSMS which allow an adversary to conduct remote exploitation of the underlying EVCS.

3. Methodology

In this work, we devise a multi-stage approach (Figure 1) to identify 16 Internet-enabled EVCSMS software products (i.e., firmware-, web- and mobile-based) and analyze their security posture. Our objective is to shed light on the insecurity of the deployed EVCSMS by uncovering an array of vulnerabilities and discussing their possible attack implications while exploring practical countermeasures to mitigate future cyber attacks. In particular, we aim at addressing the following research questions (RQs):

1. *Given the growing number of Internet-enabled EVCS, what systematic methodologies can be explored to examine the security of their EVCSMS and identify vulnerabilities that can be used to launch cyber attacks?*

2. *What are the real-life implications of cyber attacks against the EVCS and their users? How can we utilize cybersecurity countermeasures to mitigate them?*

3. *How can adversaries leverage exploited EVCS to attack critical infrastructure such as the power grid? What are the practical implications of such attacks against the power grid and its operations?*

To answer our RQs, we follow the approach presented in Figure 1 to collect a corpus of EVCSMS while utilizing various methods to perform an in-depth security analysis of the identified systems. Our aim is to detect vulnerabilities in these systems and explore real-life attack implications. Furthermore, we setup and conduct simulation experiments to study the feasibility of leveraging a botnet of exploited EVCS to carry out frequency instability attacks against the power grid and its operations. In what follows, we present a detailed description of the approach in terms of system lookup and collection, asset analysis, and vulnerability analysis.

3.1. System Lookup and Collection

As illustrated in Figure 1, the first phase in our approach consists of gathering a multitude of EVCSMS products for analysis. To achieve this, we compile a word-list containing combinations of EV- and EVCS-related keywords (e.g., EV charger, EVCS, charging station), and leverage it to perform a lookup search on each of its content using both web and device search engines. As summarized in Table 1, we identified 16 EVCSMS, which we

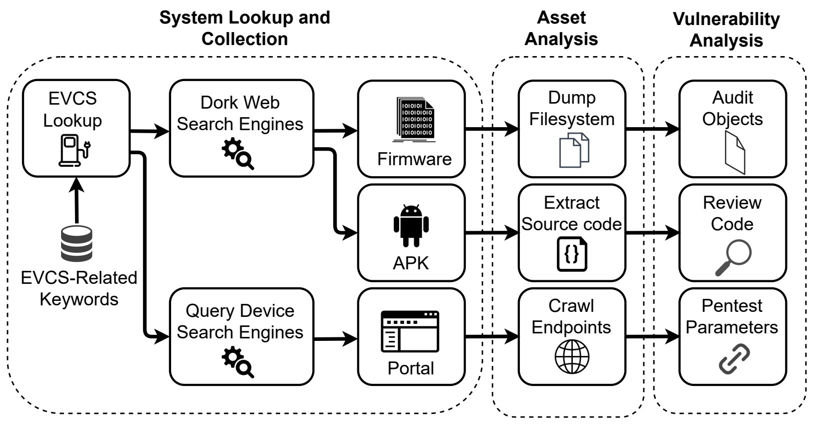


Figure 1: Overall approach for identifying online EVCSMS and analyzing their security.

categorize as firmware, mobile and web apps. In what follows, we elaborate on the different search procedures utilized to obtain each of these systems respectively.

Firmware. Our system lookup resulted in obtaining 5 widely deployed EVCSMS firmware as listed in Table 1. To obtain these firmware, we utilize various dorking techniques [14] on web search engines using the compiled EVCS wordlist to locate indexed websites of popular EVCSMS vendors, then examine their sitemaps to fetch software images from download endpoints. We achieve this by performing automated web crawls on the manufacturers' websites as well as on specialized mirror sites by inspecting each of their corresponding web pages and recursively visiting all embedded hyperlinks and directory paths to find endpoints from which we automatically download firmware and software update packages. For instance, we ran the following dork query `site:www.se.com intext:“charging station”` on Google web engine to search Schneider Electric's main website (i.e., www.se.com) for web pages that contain the string “charging station” in their HTML by explicitly specifying those keywords using the `intext` filter. This allowed us to find an endpoint titled “EVlink Charging Station” on Schneider Electric's website, containing information about the EVlink EVCS product line, which we crawled recursively to locate and download its corresponding firmware update package.

Mobile App. In addition to locating software endpoints, by exploring these websites, we detect external references that point to mobile application stores (e.g., Apple's App Store, Google's Play Store) where the vendor publishes their mobile-based EVCSMS products. For instance, when exploring ChargePoint's main website (www.chargepoint.com) we found a web endpoint (i.e., /en-us/drivers/mobile) where they published the details for obtaining their mobile app. In order to collect and analyze these apps, we automatically visit the EVCSMS app's front page on each respective store and extract all identification details (e.g., app name, version, developer) then search and download the corresponding android application package (APK) from third-party application repositories. In this work, we concentrate on collecting APK instead of iOS app store packages (IPA) since most vendors target product release on the Play Store, due to its less restrictive policies in comparison to the App Store. In total, we identified and collected information for three EVCSMS mobile apps namely ChargePoint, Go Electric, and EV Connect (Table 1).

Web App. Lastly, we query the curated EVCS word-list on different device search engines (e.g., Shodan [15], Censys [16]) through their User Interfaces (UI) and API to find publicly accessible EVCSMS portals. For instance, we ran the following filter query `title:"Electric Vehicle Charging"` on Shodan engine to search for Internet-facing devices having a specific EVCS-related string of keywords (i.e,, "Electric Vehicle Charging") in the HTML title tag element of their HTTP web interface banners. This allowed us to find many EVCSMS product instances belonging to Cornerstone Technologies Limited whose EVCSMS portal has the following title "Login - Electric Vehicle Charging Management System". By filtering relevant and true positive systems based on the returned host data banners, we obtain a list of unique web-based EVCSMS that are connected to public and private charging stations. Furthermore, we reverse-search the discovered portals by extracting special information that they hold (e.g., product name or version, vendor name or logo), to determine their corresponding vendor, as listed in Table 1.

3.2. Asset Analysis

In this section, we discuss various techniques for performing analysis on each of the discovered systems. In general, when analyzing firmware, we dissect them to dump the EVCSMS document collection, then reverse-engineer the corresponding binaries, while for analyzing EVCSMS web apps given that we do not have access to the back-end codebase, we perform black-box penetration testing to evaluate their security and identify weaknesses. As for mobile app analysis, we decompile the collected APKs into Java source

Table 1: List of Identified EVCSMS products along with their vendors.

	Vendor/Developer	**EVCSMS**
Firmware	Schneider Electric	EVlink
	Eaton Corporation	xChargeIn
	Etrel	CSWI Etrel
	Smartfox	Smartfox
	Keba	Keba
Mobile	ChargePoint	ChargePoint
	Go Electric Stations	Go
	EV Connect	EV Connect
Web	Open Access to Sustainable Intermittent Sources	OASIS Portal
	Cornerstone Technologies Limited	BaSE EVMS
	Ensto	Ensto CSI
	Fuzhou Comprehensive Energy Inf. Service	FCEIS
	Bluesky Energy Technology	ICEMS
	Revolution Pi Project	PiControl
	Garo	Garo CSI
	Unicorn Systems	Lancelot

code, which we review to detect vulnerabilities. In what follows, we present a detailed analysis procedure for the identified systems within each category.

Firmware. As listed in Table 1, we select 5 EVCSMS firmware, which belong to top EV product vendors (e.g., Schneider Electric). Each of these firmware packages represent the vendor's developed management system, which operates a distinct set of charging station products designed and provided by their respective manufacturer. These stations include several series, are designed for different application areas, and cover the EV charging needs for public parking (e.g., streets) and private parking (e.g., commercial buildings, domicile). They are equipped with energy metering capabilities, user authentication, report generation, cost allocation, and remote maintenance. In our analysis, we download the latest firmware update releases from each of the vendor's domains and analyze them respectively. To achieve this, we mainly utilize two analysis procedures for the firmware that we collected.

In the first procedure, we uncompress the `ZIP` update archive, and extract its embedded `POSIX` tar (i.e., EPK package), from which we obtain various files amongst them a Linux/ARM OS Kernel Image, `BIN` data, `shell` scripts, `ELF` executables and a `JFFS2` filesystem containing the EVCSMS document collection. To investigate this filesystem, we write a `Bash` script (Listing 1) that creates a temporary device node, loads `mtdblock` and `jffs2` Linux kernel modules, dumps the image's `jffs2` binary `rootfs` to the device node using the `dd` utility [17], then finally mount `jffs2 rootfs` and extracts the

```bash
#!/bin/bash
sudo mknod mtdblock0 b 31 0;
sudo modprobe mtdblock;
sudo modprobe mtdram total_size=65536 erase_size=256;
sudo modprobe jffs2;
sudo dd if=EVCS_base_jffs2.img of=mtdblock0;
mkdir /media/jffs2-extracted;
sudo mount -t jffs2 mtdblock0 /media/jffs2-ext;
cd /media/jffs2-ext;
mkdir -p EVCS_base;
tar zxvf EVCS_base.tgz -C EVCS_base;
```

Listing 1: Bash script for mounting JFFS2 filesystem.

base directory from the output `TGZ` tar archive. In this base directory, we locate the document collection and reverse-engineer the `cgiServer` binary found in the `cgi-bin` sub-directory, which is a Common Gateway Interface (CGI) program used to dynamically generate and manage web content on the EVCSMS. When a request points to the `cgiServer`, the HTTP server sends its standard output to the web client instead of the terminal. The client sends `GET`-based or `POST`-based requests with form data and parameters to the `cgiServer` via standard input, and URL paths, header data as well as additional directories through process environment variables. The `CGI` program can then read those variables and data from standard input, and adapt to the web client's request to generate web pages. When we analyze the `cgiServer` executable, we find a handful of vulnerabilities that we elaborate upon in Section 4. We apply the techniques from this approach to EVlink and CSWI Etrel firmware, while the others required a different approach which we elaborate upon next.

In the second analysis procedure, we download each of xChargeIn, Smartfox and Keba update packages from the official vendor support sites, and extract their respective update `ZIP` archives. Within these archives, we locate several binary data `BIN` files among which `firmware.bin` contains the EVCSMS document collection. To dump their filesystem, we search the binary images for embedded files and executable code then subsequently extract them, using an extraction utility called `binwalk` [18]. These files that we obtain represent the document collection for the base EVCSMS consisting of several entities (e.g., HTML, XML, and PNG) and `zlib` compressed data from which, we recursively recover additional files using `binwalk` (Listing 2). This allows us to extract and examine the full control panel files from the update package. However, while many of the files in the first analysis procedure

```
DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
347193        0x54C39         GIF image data
351628        0x55D8C         HTML document header
353246        0x563DE         HTML document footer
...
```

Listing 2: Extraction of embedded files from a binary image using binwalk.

are obtained in raw format and can be directly reviewed, in this procedure, most of the files are compiled and require disassembly before analysis.

Mobile App. From the mobile app collection phase, we choose 3 mobile apps listed in Table 1 and analyze them by using a special set of reverse engineering techniques. The analysis of each of these apps relies on reversing their `APK`, which is typically an archive package that contains a manifest file, a certificate for the application, a `lib` directory holding cross-platform compiled code and a `DEX` file holding all the compiled classes, which can be interpreted by the Dalvik Virtual Machine (DVM) and the Android run-time environment. This `classes.dex` file is the most important component to conduct the study, and to obtain it, we extract all of the files from each mobile app APK package using `apktool` [19] which disassembles all the resources to a nearly original form. With the extracted classes archive, we use the `dex2jar` utility [20] to convert it into a `JAR` file, which we directly browse through its underlying `Java` source code files using `jd-gui` [21]. This recovered source code represents an almost complete initial development base used for the app compilation, which allows us to perform an accurate code review while searching for bugs.

Web App. In the system collection phase, we select 8 candidate web-based EVCSMS for thorough analysis (e.g., OASIS Portal, Ensto) as listed in Table 1. Given that these applications' complete file setup are closed source and not publicly available for acquisition, we cannot perform white-box testing approaches with them. Therefore, we resort to black-box analysis and penetration testing in order to determine vulnerabilities within them. Initially, we fingerprint open ports on each of these apps and search for their main UI front-ends typically running over `HTTP`/`HTTPS`. This allows us to determine the app's web root, which we then automatically crawl to enumerate all accessible endpoints that require no authentication to inspect. Moreover, we leverage the absence of rate-limit in some of these interfaces to brute-force the URL path using common directory dictionaries, permitting us to find hidden web content and files. With the collected URLs, we investigate (using automated tools) the pages' Document Object Model (DOM) to find `HTTP POST`-based form and embedded `GET` parameters. In addition, we harvest hidden parameters from JavaScript library files, and conduct probes to find hidden `HTTP headers` in all requests. Further, whenever we are able to go beyond the authentication form (e.g., using default credentials), we perform the previous steps again on internal endpoints. By collecting all these entities, we scan each of them systematically to identify misconfigurations and code cleansing issues. For repetitive testing and ease of logging, we also create offline copies of the examined systems by using `HTTrack` [22].

3.3. Vulnerability Analysis

We follow a multitude of methods and techniques to steer our in-depth security analysis of the obtained EVCSMS and their assets, such as the Open Web Application Security Project (OWASP) testing guide [23]. While we explore an array of vulnerabilities across the analyzed systems, in this work, we focus on identifying severe vulnerabilities that can lead to exploiting and controlling the target system such as Cross-Site Scripting (XSS) and Structured Query Language Injection (SQLi), to name a few. Furthermore, we leverage the presented threat model to explore real-world attack implications against the EVCS and their users, while using simulation results to demonstrate the feasibility and implications of cyber attacks against the power grid and its operations.

Threat Model. We assume that an adversary can compromise public and private EVCS by leveraging high severity vulnerabilities within their EVCSMS. Further, the exploitation is carried out remotely through the network, which can be performed in different ways depending on the connectivity/accessibility of the target EVCS, such as whether the EVCSMS is Internet-facing or only locally accessible through the local area network (LAN). In our study, we focus on examining Internet-connected EVCSMS whose exploitation does not require having access to the LAN, therefore making the attack vector very powerful and effective. In this case, to exploit Internet-facing EVCS, the adversary is assumed to perform Internet-wide scanning to search for viable EVCSMS before trying to exploit their vulnerabilities. However, it should be noted that connectivity of the EVCS does not present any difference in terms of the actual exploitation process (i.e., triggering the vulnerabilities). For instance, if the EVCS is not accessible via the Internet, then the adversary is assumed to have access to the LAN where the

EVCS is connected to in order to conduct local, however remote, exploitation. Following these two methods, the adversary is assumed to take control over the underlying EVCS, while being capable of launching various cyber attacks against the vulnerable EVCS (e.g., manipulating the charging process), the corresponding users/operators (e.g., hijacking their user accounts), and the integrated critical infrastructure (e.g., destabilize the power grid). Additionally, the adversary can leverage the compromised EVCS to create a botnet and conduct distributed cyber attacks against other devices (e.g., Distributed DoS).

4. Results: Vulnerability Analysis and Attack Implications

In general, our analysis unveiled a range of vulnerabilities, among which, we highlight 13 severe vulnerability classes across all the analyzed EVCSMS. As summarized in Table 2, we enumerate each vulnerability/weakness using its corresponding common weakness enumeration (CWE) ID, which can be used to lookup further information about the specified weaknesses within the CWE MITRE [24] and OWASP [25] databases (e.g., CWE-79 corresponds to XSS). Note that the analyzed systems were associated with a number of vulnerabilities. For instance, we identified 8 severe vulnerabilities for EVlink (e.g., XSS) while ChargePoint was associated with a single severe vulnerability (Table 2). Despite the variable number of uncovered weaknesses across the analyzed EVCSMSs, it is important to note that our main objective is to identify at least one vulnerability that can lead to remote/local exploits, thus, compromising the target EVCS. In addition to the identified vulnerabilities, we discuss attack implications against various stakeholders within the EV ecosystem. While it is possible to conduct different attacks on various entities within the EV ecosystem, in this work, we focus on investigating large-scale attacks that have severe impact on the compromised charging station (EVCS), its users, and the connected power grid.

4.1. Attacks Against the EVCS

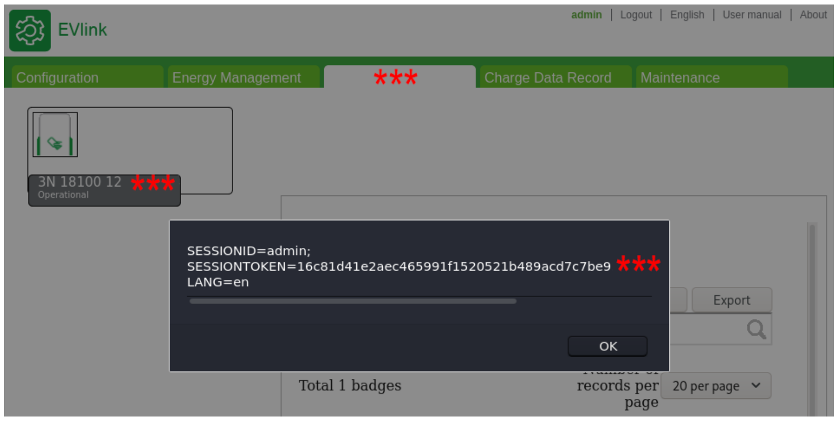
As described in the threat model presented in Section 3.3, an adversary can compromise a target EVCS by exploiting its management system using one or more vulnerabilities. We discuss some of the main attack implications against the EVCS and its operations in the following sub-sections:

Charging Process and Settings Manipulation. Our security analysis unveiled several vulnerabilities across the identified EVCSMS (Table 2)

Table 2: Overview of vulnerabilities discovered within the analyzed EVCSMS.

		CWE-ID / Vulnerability												
		79	89	200	306	321	352	425	798	799	918	942	942	1236
	EVCSMS	Cross-Site Scripting (XSS)	SQL Injection (SQLi)	Information Disclosure	Missing Authentication	Embedded Secrets	Cross-Site Request Forgery (CSRF)	Forced Browsing	Hard-Coded Credentials	Missing Rate Limit	Server-Side Request Forgery (SSRF)	CORS Misconfiguration	FCDP Misconfiguration	CSV Injection (CSVi)
Firmware	EVlink	✓		✓			✓	✓	✓	✓	✓			✓
	xChargeIn				✓					✓			✓	
	CSWI Etrel	✓			✓				✓		✓		✓	
	SmartFox								✓	✓				
	Keba				✓								✓	
Mobile	ChargePoint					✓								
	Go					✓				✓				
	EV Connect					✓				✓				
Web	OASIS Portal	✓					✓							
	BaSE EVMS		✓							✓				
	Ensto CSI				✓					✓				
	FCEIS									✓		✓		
	ICEMS		✓							✓				
	PiControl	✓					✓		✓	✓	✓	✓		
	Garo CSI				✓					✓				
	Lancelot									✓		✓		

that allow an attacker to compromise the EVCS and view its charging schedules while manipulating its operations by initiating, delaying, or stopping any charging process. In general, our analysis indicates that most of the examined EVCSMS lack adequate input sanitization, which is a root-cause of XSS vulnerabilities. For instance, EVlink suffers from several XSS vulnerabilities, which were detected by reversing the `cgiServer` binary and uncovering several endpoints along with their corresponding `GET` parameters that permitted malicious JavaScript injection into the web frame. This was mainly caused by the lack of adequate cleansing and encoding of supplied user-



```
SESSIONID=admin;
SESSIONTOKEN=16c81d41e2aec465991f1520521b489acd7c7be9***
LANG=en
```

Figure 2: Stored XSS on EVlink allows hijacking the administrator's session tokens (the identifiable information from the testing instance has been redacted).

input. Exploiting such XSS allows an attacker to inject malicious JavaScript code into the EVCSMS context to hijack a target user's account session and take many actions such as modifying the account and EVCS settings/configurations. Furthermore, when the compromised target user account has privileged access (e.g., administrator), the attacker gains full control over all of the EVCSMS functionalities and data. For example, we discovered a configuration initialization functionality within EVlink that was vulnerable to Comma-Separated Values injection (CSVi), which can be exploited to embed an XSS payload that gets triggered and stored on the system database when the crafted CSV file is loaded. This vulnerability leads to a stored XSS, which enables privilege escalation by hijacking the administrator's session tokens, as shown in Figure 2. Additionally, this XSS weakness allows persistent access by implanting a web shell to periodically fetch and execute JavaScript from an adversarial remote server.

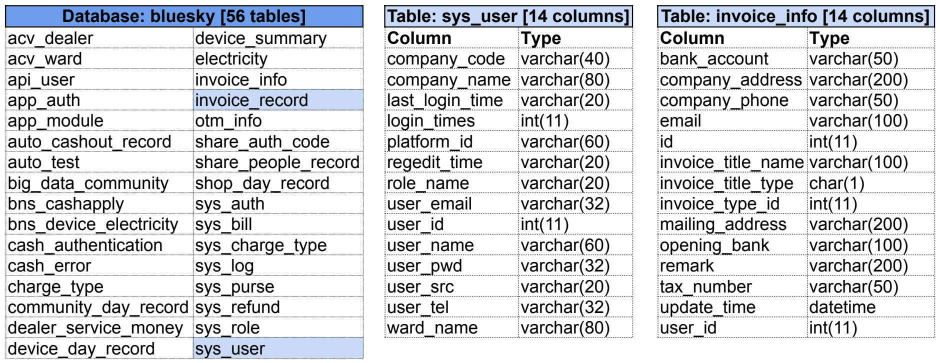
In addition to XSS, we found several EVCSMS that were vulnerable to CSRF weaknesses, which allow attackers to induce target users to perform unintentional actions that lead to gaining control over the user account and manipulating the EVCS settings. Consequently, the attacker can view/-control all of the EVCSMS's data and functionalities when the target user is privileged (e.g., administrator user). For example, we discovered a CSRF flaw on the OASIS administrator panel, which is due to the lack of a CSRF token when submitting data, that can be used to trigger a `POST`-based reflected XSS, allowing an attacker to hijack the user's account. The crafted Proof-of-

```
<html>
  <body>
  <script>history.pushState('', '', '/')</script>
    <form action="http://[host]:[port]/admin.cgi" method="POST">
      <input type="hidden" name="FORM" value="LOGIN&#95;FORM" />
      <input type="hidden" name="EMAIL" value="a&#64;ia&#46;ayu41d&apos;&gt
    ;&lt;script&gt;eval&#40;atob&#40;&quot;YWxlcnQoZG9jdW1lbnQuY29va2llKQ&
    quot;&#41;&#41;&#59;&lt;&#47;script&gt;cg1yi" />
      <input type="hidden" name="SUBMIT" value="Login" />
      <input type="submit" value="Submit request" />
    </form>
    <script>
      document.forms[0].submit();
    </script>
  </body>
</html>
```

Listing 3: CSRF chained with XSS on OASIS can lead to account hijacking via theft of session tokens.

Concept form is presented in Listing 3, where a HTML-encoded/BASE64-encoded XSS payload is embedded into the vulnerable `EMAIL` parameter to trigger a popup alert box showing the account session tokens. Moreover, in PiControl-based EVCSMS, we discovered `POST`-based CSRF weaknesses that can lead to the modification of the EVCS's control panel settings including device information, networking settings, and charging/scheduling configurations. We also discovered a `GET`-based CSRF vulnerability in EVlink which allows attackers to takeover the target user's account by changing the corresponding password value through a vulnerable `GET` parameter.

Firmware Manipulation. In addition to XSS and CSRF vulnerabilities, an attacker can exploit other high severity weaknesses such as SQL injection (SQLi) attacks to gain privileged access to the EVCSMS and perform firmware manipulation. This is typically done by exploiting SQLi vulnerabilities to obtain access to the entire EVCSMS database, which contains user records including high privilege user account information and credentials (e.g., administrator). Indeed, we identified a number of EVCSMS that are vulnerable to SQLi (Table 2). For instance, the authentication forms on BaSE EVMS and ICEMS suffered from boolean-/time-based blind SQLi flaws through their `POST` parameters, which can be utilized as injection points to systematically execute arbitrary SQL queries and dump the stored EVCSMS database tables. For example, as shown in Figure 3, table `sys_user` contains user accounts' information and credentials, including those belonging to the administrator accounts. Moreover, our analysis showed that some of the



Database: bluesky [56 tables]	
acv_dealer	device_summary
acv_ward	electricity
api_user	invoice_info
app_auth	invoice_record
app_module	otm_info
auto_cashout_record	share_auth_code
auto_test	share_people_record
big_data_community	shop_day_record
bns_cashapply	sys_auth
bns_device_electricity	sys_bill
cash_authentication	sys_charge_type
cash_error	sys_log
charge_type	sys_purse
community_day_record	sys_refund
dealer_service_money	sys_role
device_day_record	sys_user

Table: sys_user [14 columns]	
Column	**Type**
company_code	varchar(40)
company_name	varchar(80)
last_login_time	varchar(20)
login_times	int(11)
platform_id	varchar(60)
regedit_time	varchar(20)
role_name	varchar(20)
user_email	varchar(32)
user_id	int(11)
user_name	varchar(60)
user_pwd	varchar(32)
user_src	varchar(20)
user_tel	varchar(32)
ward_name	varchar(80)

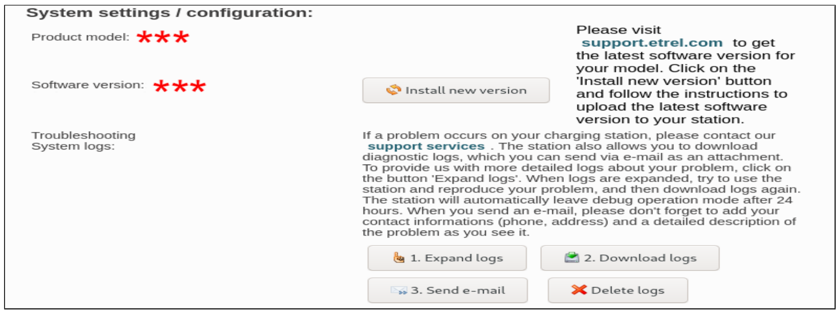
Table: invoice_info [14 columns]	
Column	**Type**
bank_account	varchar(50)
company_address	varchar(200)
company_phone	varchar(50)
email	varchar(100)
id	int(11)
invoice_title_name	varchar(100)
invoice_title_type	char(1)
invoice_type_id	int(11)
mailing_address	varchar(200)
opening_bank	varchar(100)
remark	varchar(200)
tax_number	varchar(50)
update_time	datetime
user_id	int(11)

Figure 3: SQLi on ICEMS allows dumping the database which contains tables with sensitive user account details “sys_user” and billing information “invoice_record”.

tested EVCSMS such as SmartFox and CSWI Etrel have default hard-coded credentials used for inbound authentication. By obtaining these credentials, an attacker can directly access and use the EVCSMS circumventing the implemented security measures and have access to internal functionalities and data.

Subsequently, an attacker can exploit the obtained administrator level of access to alter the deployed EVCS firmware. For instance, the attacker can downgrade the firmware of vulnerable EVCSMS by uploading, through the functionality illustrated in Figure 4, an older and possibly less secure version. It is important to note that such firmware downgrade was mainly possible due to the lack of adequate version checks in the implementation of the system. Additionally, due to insufficient sanity checks on the uploaded firmware package, the attacker can override the checksum hash stored locally within the filesystem and upload a modified firmware with various altered binaries. This is extremely alarming as it enables implanting a rootkit within the EVCS firmware to gain persistent and privileged capabilities while enabling further attacks by controlling the EVCS and performing covert malicious activities.

Billing Manipulation. In addition to firmware manipulation, an attacker can exploit SQLi vulnerabilities to manipulate the billing functions within a compromised EVCSMS and modify charging costs. For instance, the adversary can overwrite the content of the `sys_bill` and `sys_refund` tables within the EVCSMS database, which contain billing and refund information (Figure 3). Consequently, the original system billing values can be


System settings / configuration:

Product model: ***

Software version: ***

Please visit support.etrel.com to get the latest software version for your model. Click on the 'Install new version' button and follow the instructions to upload the latest software version to your station.

Troubleshooting
System logs:

If a problem occurs on your charging station, please contact our support services . The station also allows you to download diagnostic logs, which you can send via e-mail as an attachment. To provide us with more detailed logs about your problem, click on the button 'Expand logs'. When logs are expanded, try to use the station and reproduce your problem, and then download logs again. The station will automatically leave debug operation mode after 24 hours. When you send an e-mail, please don't forget to add your contact informations (phone, address) and a detailed description of the problem as you see it.

Figure 4: Improper check on CSWI Etrel allows an attacker with administrator privilege to downgrade the EVCS firmware (the identifiable information from the testing instance has been redacted).

tampered with to decrease billed values or claim illegitimate refunds. Note that such attacks can be of interest to an external malicious party or legitimate users who want to abuse the EVCSMS to modify or possibly nullify their charging expenses.

Bot Recruitment and Network Proxy. An attacker can recruit a large number of compromised EVCS within a coordinated botnet to launch various cyber attacks such as targeted denial of service (DoS) or Internet probing/reconnaissance activities. To achieve this, an attacker can leverage Server-Side Request Forgery (SSRF) vulnerabilities to use the compromised EVCS as proxies and force them to redirect requests towards internal/external endpoints and perform lateral movement on the network as well as scan third-parties. Our analysis indicates that three EVCSMS suffer from SSRF vulnerabilities (Table 2), which are mainly caused by incomplete validation of the values passed through `GET` and `POST` request parameters. Specifically, by intercepting these requests and altering their default values to inject random domain/IP addresses, an attacker can force the EVCSMS to submit `HTTP`/`DNS` requests to external parties. In addition, the adversary can inject local IP addresses (e.g., 192.168.0.1) to force the station to forward internal requests towards other devices on the LAN, hence enabling local device discovery. Furthermore, SSRF vulnerabilities can be leveraged further to extract information from the EVCSMS by redirecting requests to the `localhost` (i.e., 127.0.0.1), which enables reading arbitrary files and record logs stored on the EVCS's filesystem.

Denial of Service (DoS). An attacker can leverage their control over the EVCSMS to lock the underlying EVCS or disable specific features in

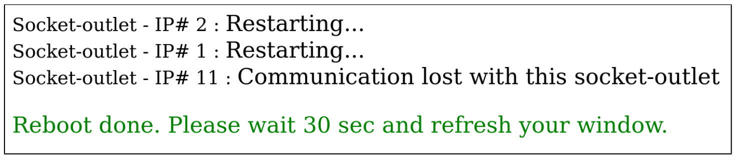
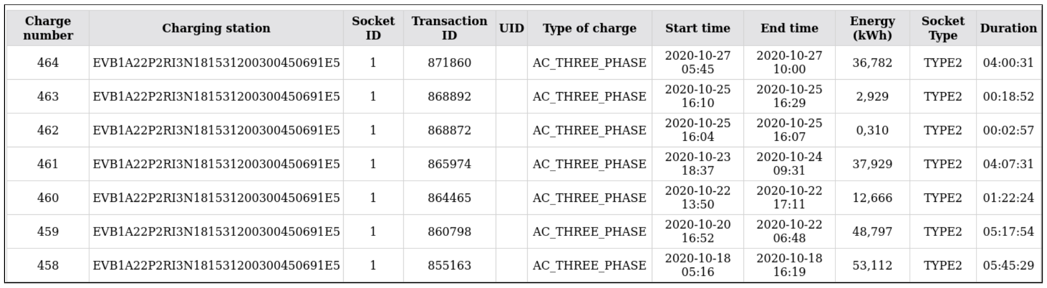

```
Socket-outlet - IP# 2 : Restarting...
Socket-outlet - IP# 1 : Restarting...
Socket-outlet - IP# 11 : Communication lost with this socket-outlet

Reboot done. Please wait 30 sec and refresh your window.
```

Figure 5: Forcing a 30-second restart loop on EVlink through CSRF flaw.

its configurations, denying the legitimate user from physical and virtual access. To perform DoS attacks and prevent legitimate clients from using the EVCS, an attacker needs to initially gain control over the EVCSMS, for instance through XSS or CSRF, and then, tweak the EVCS settings to switch ON/OFF certain features that hinder its usage. For instance, we found that EVlink and OASIS suffer from CSRF flaws, which enable an adversary to hijack functionalities on the EVCS, specifically, to force-restart the EVCS. Consequently, the adversary can trigger the restart functionality repeatedly to keep the EVCS in a continuous restart loop, which disrupts its charging schedules/operations. As illustrated in Figure 5, the CSRF vulnerability on EVlink can cause the EVCS to restart every 30 seconds by continuously triggering this action. This is mainly possible due to the absence of a randomized token to validate the restart action. Furthermore, an attacker can also perform DoS attacks against a target EVCS by flooding the EVCSMS with a massive amount of requests while preventing legitimate users from accessing the management system. Indeed, our analysis indicates that several EVCSMS (e.g., CSWI Etrel, Keba) do not implement rate limiting mechanisms on their essential functionalities such as authentication. This allows an adversary to crash the EVCSMS as well as conduct dictionary attacks against the login form and brute-force the EVCSMS web paths to determine hidden endpoints and resources.

4.2. Attacks Against the User

It is important to realize that the EVCS users represent main stakeholders within the EV ecosystem. Moreover, the EVCSMS application interfaces are primarily developed to provide remote management functionalities and features to facilitate the charging experience for users. Therefore, any vulnerability within the EVCSMS can constitute a direct threat to the users themselves. In what follows, we describe a number of attack scenarios against the EVCS users.



Charge number	Charging station	Socket ID	Transaction ID	UID	Type of charge	Start time	End time	Energy (kWh)	Socket Type	Duration
464	EVB1A22P2RI3N181531200300450691E5	1	871860		AC_THREE_PHASE	2020-10-27 05:45	2020-10-27 10:00	36,782	TYPE2	04:00:31
463	EVB1A22P2RI3N181531200300450691E5	1	868892		AC_THREE_PHASE	2020-10-25 16:10	2020-10-25 16:29	2,929	TYPE2	00:18:52
462	EVB1A22P2RI3N181531200300450691E5	1	868872		AC_THREE_PHASE	2020-10-25 16:04	2020-10-25 16:07	0,310	TYPE2	00:02:57
461	EVB1A22P2RI3N181531200300450691E5	1	865974		AC_THREE_PHASE	2020-10-23 18:37	2020-10-24 09:31	37,929	TYPE2	04:07:31
460	EVB1A22P2RI3N181531200300450691E5	1	864465		AC_THREE_PHASE	2020-10-22 13:50	2020-10-22 17:11	12,666	TYPE2	01:22:24
459	EVB1A22P2RI3N181531200300450691E5	1	860798		AC_THREE_PHASE	2020-10-20 16:52	2020-10-22 06:48	48,797	TYPE2	05:17:54
458	EVB1A22P2RI3N181531200300450691E5	1	855163		AC_THREE_PHASE	2020-10-18 05:16	2020-10-18 16:19	53,112	TYPE2	05:45:29

Figure 6: Charging data record log.

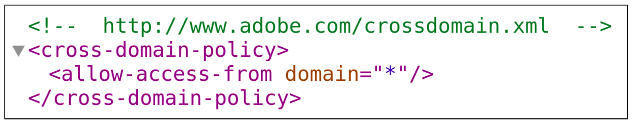
Charging Data/Record Theft. A number of the vulnerabilities presented in Table 2 such as CSRF and SQLi allow the attacker to disguise as a legitimate user and have access to various user information and resources. Some of these resources such as the charging data records and vehicle-specific log data (Figure 6), can delineate user behaviors and charging activities. For instance, an adversary can use such information to infer users' EV charging habits and schedules, which can be abused for several malicious purposes (e.g., surveillance, espionage, property heist, etc.).

Additionally, an attacker can leverage vulnerabilities such as Cross-Origin Resource Sharing (CORS) and Flash Cross-Domain Policy (FCDP) misconfigurations to leak sensitive user information/data and use it to access user accounts. For instance, we found that some EVCSMS such as FCEIS and Lancelot, implemented significantly lenient CORS policies, which can be exploited to remotely access the EVCSMS from external domains. An example of such permissive CORS policy is linked to the specification of a wildcard ("*") origin in the `Access-Control-AllowOrigin` header, as shown in Listing 4. This weakness makes the system vulnerable to cross-domain attacks by dynamically accepting and reflecting origins from cross-domain external connections. Moreover, such CORS misconfiguration allows the adversary to extend the EVCSMS Same-Origin Policy (SOP) to perform further attacks by sending requests to external domains and exfiltrate account session data.

Further, we discovered that some EVCSMS implemented permissive FCDP, which open the door for attacks on the users by allowing arbitrary domains to interact with the EVCSMS. For instance, CSWI Etrel employed an unrestricted cross domain policy (`crossdomain.xml`), which can enable two-way interactions between external domains and the EVCSMS (Figure 7). Consequently, attackers can steal account tokens and exfiltrate data from the

```
HTTP/1.1 200 OK
Content—Type: text/html
Accept—Ranges: bytes
Vary: Accept—Encoding
X—Powered—By: ASP.NET
Access—Control—Allow—Origin: *
Connection: close
```

Listing 4: Over-lenient CORS on FCEIS permits for cross-domain exploitation.



```
<!-- http://www.adobe.com/crossdomain.xml -->
<cross-domain-policy>
  <allow-access-from domain="*"/>
</cross-domain-policy>
```

Figure 7: Insecure FCDP on CSWI Etrel permits arbitrary domain access.

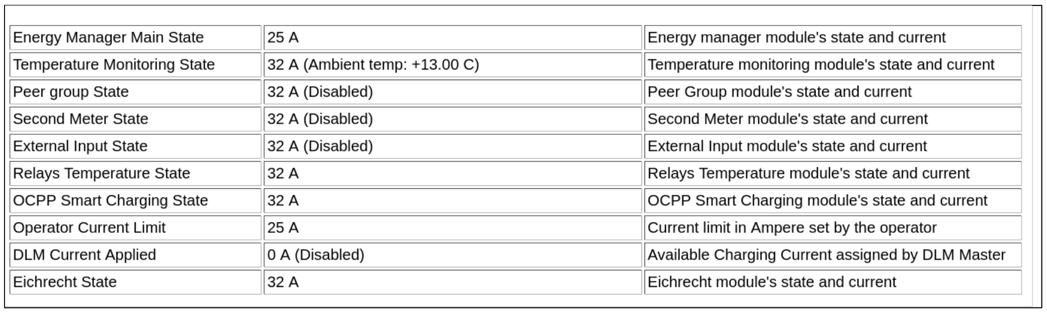
target user session while enabling further attacks using the compromised user account.

Personally Identifiable Information Leakage. In general, EVCSMS products may require users to provide their details as part of their account configuration to facilitate authentication and ensure service legitimacy by incorporating their Personally Identifiable Information (PII) such as name, address, and contact information. Therefore, attackers can exploit EVCSMS vulnerabilities with the intention to compromise users' accounts and obtain their PII, which can be leveraged for consequent attacks against the users such as blackmailing, harassment, and identity theft, to name some.

It is important to realize that PII leakage can occur by different means. For instance, we discovered a maintenance endpoint on EVlink that did not enforce adequate authorization and thus, can lead to forced browsing attacks. In fact, authentication can be bypassed by directly visiting this endpoint, which can grant an attacker access to the maintenance and energy management settings panel, where they will be able to view user information and charging processes details along with other sensitive information about the internal system (e.g., firmware version).

Similarly, other EVCSMS such as Keba and Garo CSI suffered from information disclosure vulnerabilities due to missing authentication on a number of their endpoints. For example, this can enable an unauthenticated adversary to learn information about the state of the underlying EVCS, as illustrated in Figure 8.

From a different perspective, we discovered hard-coded secret keys (e.g.,



Energy Manager Main State	25 A	Energy manager module's state and current
Temperature Monitoring State	32 A (Ambient temp: +13.00 C)	Temperature monitoring module's state and current
Peer group State	32 A (Disabled)	Peer Group module's state and current
Second Meter State	32 A (Disabled)	Second Meter module's state and current
External Input State	32 A (Disabled)	External Input module's state and current
Relays Temperature State	32 A	Relays Temperature module's state and current
OCPP Smart Charging State	32 A	OCPP Smart Charging module's state and current
Operator Current Limit	25 A	Current limit in Ampere set by the operator
DLM Current Applied	0 A (Disabled)	Available Charging Current assigned by DLM Master
Eichrecht State	32 A	Eichrecht module's state and current

Figure 8: Unauthenticated EVCS state information disclosure on Ensto CSI.

cryptographic keys) in the Java source code of three mobile-based EVCSMS namely ChargePoint, Go and EV Connect (Table 2). These secret keys were used for encrypting data/communications through several operations such as account creation/registration, authentication, and information update. Thus, by acquiring them, an adversary can potentially circumvent the encryption schemes exposing users' communications to infer personal information along with other sensitive data (e.g., account credentials).

Payment Fraud. Almost all public EVCSMS are designed with online payment capabilities to handle transactions and charging bills. As described in Section 4.1 (Billing Manipulation), SQLi vulnerabilities can be used to dump information from the EVCSMS database including stored billing records that contain users' payment information (e.g., `invoice_info` table from Figure 3). Alternatively, an adversary can implement an active listener to covertly steal this payment information by exploiting other vulnerabilities such as stored XSS, as illustrated in Figure 2. Thereafter, the attacker can utilize the stolen financial information to execute payment fraud or sell this information to other malicious third-parties who will utilize it to commit malicious activities. Note that our analysis indicates that several EVCSMS products are vulnerable to such attacks by leveraging SQLi and stored XSS vulnerabilities (Table 2).

4.3. Attacks Against the Power Grid

As we demonstrated the insecurity of a number of deployed EVCS by exploiting different vulnerabilities within their management systems, it is worth noting that exploited EVCS might be utilized to perform cyber attacks against the integrated infrastructure such as the power grid [26–28]. In essence, by utilizing the remotely exploitable vulnerabilities (e.g., XSS) that have been discussed throughout this work, an attacker is able to fully take over and control the underlying EVCS that are being managed by the impacted vulnerable EVCSMS. Thus, malicious actors who exploit these vulnerabilities can manipulate the corresponding EVCS by dictating their operations such as the charging and discharging of the connected EVs, ultimately swaying the power consumption flow established with the power grid.

Given that the power grid handles large-scale operations to serve millions of customers, any attacks against such critical infrastructure would consequently result in significant implications. Thus, performing such large-scale cyber attacks would be of interest to various political adversaries and state sponsored actors who seek to cause devastating economical and social damage to their opposition [29]. In fact, power grids are increasingly prone to cyber attacks and according to the U.S. Department of Homeland Security there were 12 times more security threats issued for entities in the U.S. electricity sector in 2018 in comparison with 2010 [30]. Furthermore, with the EV charging ecosystem being a new and wide attack surface, it constitutes an attractive target for exploitation by large-organizations as well as individuals and groups [31] with enough resources to conduct attacks by leveraging this infrastructure.

To this end, we discuss attack implications against the power grid by evaluating the impact of various attack scenarios, in particular, frequency instability attacks which we choose to examine in view of the significant threat that they represent to the power grid's stability [32]. Specifically, these attacks are due to abrupt changes in the power demand/supply which create an imbalance that drives to sudden drop or increase in the power grid's system frequency. To perform our analysis, we assume that the adversary controls a large number of compromised EVCS, which are orchestrated to initiate simultaneous charging/discharging requests with the aim of destabilizing or crippling the power grid.

In order to determine the stability and reliability of the grid, electric supply/demand balance along with the frequency of the system, represented through the speed of the generators, are used as indicators. For instance, when the power grid's electrical demand increases, the speed of the generators is reduced, releasing this kinetic energy into the system and vice versa. In other words, to maintain its stability, the grid has to operate within a specific range of frequencies. If any frequency deviation occurs the system stability and performance is thrown off. Given that it is virtually impossible to investigate the implications of such large-scale attacks on a real-world

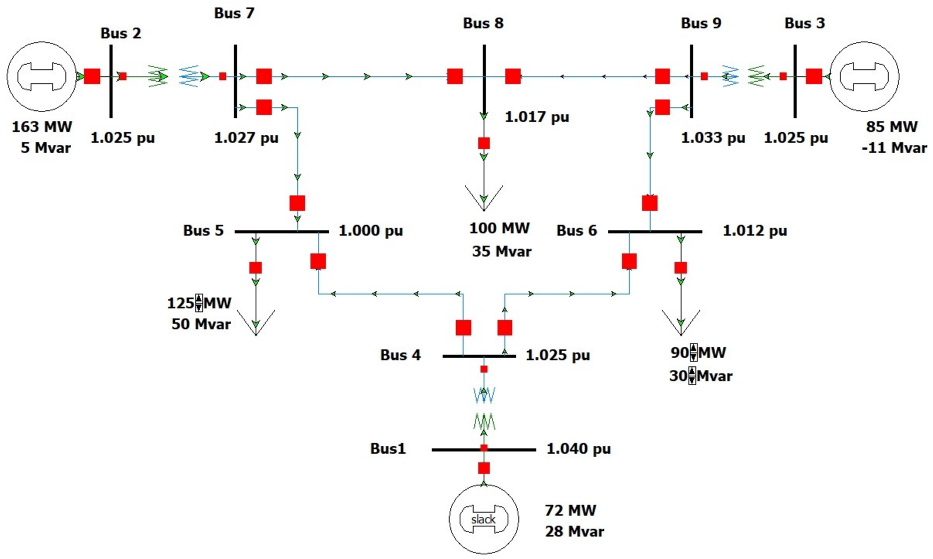
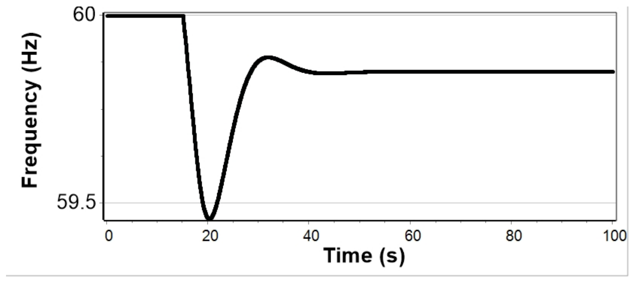


Figure 9: This WSCC system with nine buses and three generators.

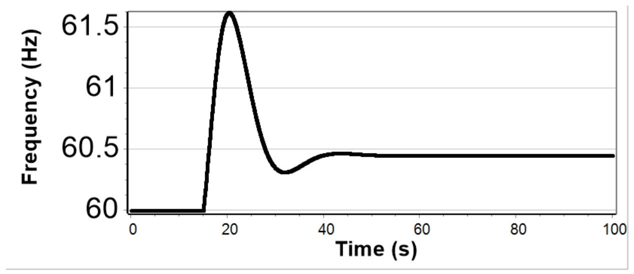
power grid, we leverage simulation analysis using `PowerWorld` [33], which is a widely used industrial-level software suite for testing/analyzing frequency stability of power systems.

For our simulation, we based the analysis on North America's standard for critical region frequency ranges [34]. Specifically, whenever the system's frequency drops below $59.5Hz$ (Demand $>>$ Supply) or raises above $61.5Hz$ (Supply $>>$ Demand) due to massive imbalance between supply and demand, the grid becomes operating in a critical region and any further electric imbalance would lead to shutting down the protection relays and equipment.

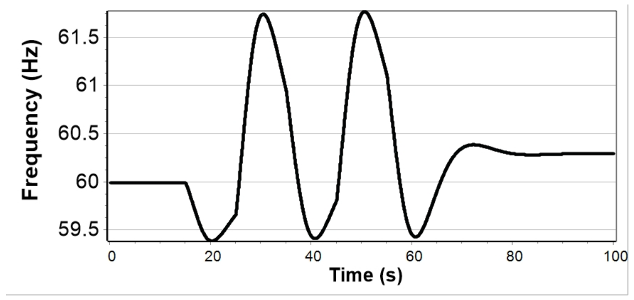
As shown in Figure 9, we use a system approximation provided by the Western System Coordinating Council (WSCC) with 9 buses/lines and a demand equal to $315MW$ [35]. This setup is commonly used as a benchmark for power systems transient stability analysis due to its reasonably small size. Note that buses 5, 6, and 8 are the load buses. Also, there are two generators at buses 2 and 3 with inertia, while the generator at slack bus 1 has no inertia, since it has variable generation to make the power flow equations feasible. For testing purposes, we set generators 2 and 3 to be IEEE type-2 speed-governing model (IEEE-G2) with fixed inertia constants, and we assume that an adversary has compromised EVCS (level 1, 2 and 3) that are scattered across buses 5, 6 and 8.



(a) Sudden growth in demand by mass EV charging requests–critical region ($59.5Hz$)



(b) Sudden growth in supply by mass EV discharging–critical region ($61.5Hz$)



(c) Alternating growth in demand and supply by mass EV charging and discharging

Figure 10: Frequency instability attack scenarios against the power grid.

Increase in Charging Demand. In this attack scenario, an adversary is assumed to leverage a large number of compromised EVCS to launch synchronized charging operations at the same time. The objective is to destabilize the grid through a sudden increase in the charging demands, which can lead to cascading failure in the grid [26, 32, 36]. To emulate this attack scenario, we initially operate the WSCC system at its nominal frequency

($60Hz$) and then, we increase the loads on buses 5, 6 and 8, which represent compromised EVCS, by $7.2MW$ at t = 15s. On average, this load increase corresponds to an estimated 3,000 EVs charging on level 2 EVCS, or 196 EVs charging on level 3 EVCS, or a mixture of about 1,000 EVs charging on level 2 EVCS and 131 EVs charging on level 3 EVCS.

As shown in Figure 10a, the transient stability analysis demonstrates a sudden drop in the simulated system's frequency as a result of this attack. More importantly, the frequency drops below the critical operating region ($59.5Hz$), thus achieving the attacker's goal in terms of destabilizing the frequency and causing power failures in the grid.

Increase in Discharging Supply. In the second attack scenario, the adversary is assumed to reverse the electric flow back to the grid by discharging a large number of connected EVs through the bidirectional power flow feature of compromised EVCS [37]. This feature is enabled by Vehicle-to-Grid (V2G) technology which allows feeding the energy stored in an EV battery back into the power grid. While V2G is intended to support the grid when power is required, an attacker can maliciously exploit it to impair the grid by injecting power to it. The objective is to synchronize large-scale discharging operations to destabilize the power grid by causing a sudden growth in the electric supply disrupting the grid's power demand/supply balance.

To test this attack scenario, we changed bus 5 into a generator to emulate the reverse power flow of the discharging EVs. Although it is not trivial to fully simulate an EVCS which encompasses power converters (AC/DC and DC/DC), power conditioning units such as a power factor corrector, sensors and controllers to direct the power flow, we can approximately imitate its electric behaviour. Specifically, for our simulation setup, we utilize a photovoltaic (PV) REGC_A model generator with REGC_C exciter [38] which closely resembles the power electronics of an EV battery discharging through an EVCS. For testing purposes, we select a Chevy Volt with charging/discharging limit of 3.3 kW as our connected EVs [39]. Furthermore, we simulate the discharging operations by a sudden injection of $51.7MW$ of power at t = 15s, which represents an estimated 15,000 discharging EVs, respectively. As indicated by the simulation results in Figure 10b, the attack was successful as it caused a system instability by pushing the frequency above the critical region ($61.5Hz$).

It should be noted that this discussed attack may not be viable today due to the large amount of manipulated power load. However, having a large number of connected EVs within a city will become a common occurrence in the future. For instance, New York city already has more than 15,000 EVs registered [40]. Therefore, with the incentives that governments are offering to stimulate the purchase of EVs [41], the projected increase in the adoption of EVs and the corresponding linear growth in the numbers of deployed EVCS [1], we expect that this attack will become feasible in the few upcoming years.

Switching Attack. In a switching attack scenario, the adversary combines the capabilities presented in the previous attack scenarios to synchronize large-scale alternating charging and discharging operations among the compromised EVCS and their connected EVs within a short time period. Such attack aims at causing sudden and switching frequency disturbances, that throw off the stability of the power grid, and ultimately leading to cascading failures [42, 43]. Specifically, by forcing the EVs to start charging, the attacker would cause the system frequency to drop, from which the system will attempt to recover by increasing its generation to bring back the frequency to normal. At this point, the attacker will take advantage of the system's response to perform the opposite attack (increase in supply by forcing EVs to discharge), by removing the load they added and instead injecting power to the grid, by leveraging the Vehicle-to-Grid (V2G) feature of EVCS. This would cause the system to have more generation than load, overshooting the frequency to the critical region. Consequently, the system will attempt to recover by reducing its own generation, to which the attacker will respond by increasing the load, causing a drop in frequency and so on. Thus, the attacker does not allow the system to restore its frequency back to normal.

To simulate the effects of this attack on the grid, we switch between the two previous attacks by conducting a charging demand increase at bus 6 followed by a discharging supply increase at bus 5. We start the attack by emulating an adversary who initially forces EVCSs to cause a load surge by increased charging at t = 15s. Then, once the system's frequency restores from its critical peak to the nominal frequency range (within 10s), the adversary supplies the system with $66MW$, representing significant discharging operations from connected EVs. In order to determine the amount of power that the attacker requires to inject or remove to destabilize the power grid, we perform several simulations using transient stability analysis with which we trace the power loads that push the system's frequency into the critical regions. As shown in Figure 10c, these two charging and discharging attacks are repeated consequently, causing the system's frequency to alternate values below (e.g., at t=20s, 40s, and 60s) and above the critical regions (e.g., at t=30s and 50s) within a short time period. As a result, the attacker will be able to destabilize the power grid while possibly causing cascading failures in the operations of the grid.

4.4. Attacks Against Other Entities

While we discuss attack implications against the above mentioned three main stakeholders, it is also possible to perform attacks using the compromised EVCS against other entities within the EV ecosystem. For instance, an attacker can target the connected EVs with the purpose of damaging their batteries by modifying their charging levels and ignoring critical battery conditions through the toleration of high voltages/currents [44]. Nevertheless, discussing these attacks requires further investigations and analysis, which are beyond the scope of this work and will be considered for future work. Additionally, we only focus in our study on the main stakeholders as we believe that attacks that target them have significant impact and implications on users and the connected critical infrastructure.

5. Mitigating Countermeasures

In what follows, we discuss practical countermeasures that can be implemented by system developers/designers to strengthen the deployed EVCSMS and mitigate further cyber attacks against the EVCS, its users, and the connected power grid. As presented in Table 3, we propose a list of countermeasures, which aim at addressing the identified vulnerabilities in Table 2. Additionally, we refer to the documentations available on the CWE MITRE [24] and OWASP [25] for detailed information about known/recommended countermeasures, which can be navigated using the given CWE-ID of each vulnerability. Finally, we provide additional security guidelines and best practices for further protection and future system implementation and deployment.

5.1. Patching the Vulnerabilities

Mitigating the discussed attacks against the main stakeholders (Section 4) requires addressing all the identified high severity vulnerabilities of the EVCSMS, as summarized in Table 3. In what follows, we present further details about the recommended mitigation techniques to patch each vulnerability.

XSS. To prevent XSS vulnerabilities (CWE-79) within EVCSMS, tamperable HTTP parameters have to be strictly filtered based on a pre-compiled

Table 3: Overview of recommended mitigations for the discovered vulnerabilities within the analyzed EVCSMS.

CWE-ID	Vulnerability	Impact	Implication	Mitigation
79	XSS	Code execution	Account hijacking	Sanitize user-controllable input data
89	SQLi	Query execution	Complete takeover	Utilize parametrized queries
200	Information Disclosure	Information exposure	Information leakage	Enforce authentication on all endpoints
306	Missing Auth.	Unauthorized access	Functionality manipulation	Enforce authentication on all functionalities
321	Embedded Secrets	Information exposure	PII leakage	Request cryptographic keys during runtime
352	CSRF	Settings modification	Functionality manipulation	Utilize random tokens with all requests
425	Forced Browsing	Unauthorized access	Functionality manipulation	Enforce better access control mechanisms
798	Hard-Coded Cred.	Unauthorized access	Complete takeover	Enforce credential update policy
799	Missing Rate Limit	Unauthorized access	DoS	Prevent excessive and fast requests
918	SSRF	Network access	Bot recruitment	Sanitize IP/URL addresses on parameters
942	CORS Misconfiguration	Data exfiltration	Data/record theft	Enforce stricter cross-domain policy
942	FCDP Misconfiguration	Data exfiltration	Data/record theft	Enforce stricter cross-domain policy
1236	CSVi	Code execution	Account hijacking	Implement safe parsing for CSV files

list of valid values when possible, and user-controllable input have to be properly cleansed and encoded on output (e.g., HTML tag brackets < and > become &#60; and &#62; with HTML-encoding) to prevent them from being actively rendered as part of the response HTML body [45]. During our analysis, it was observed that such vulnerable fields and parameters within EVCSMS correspond to PII form fields (e.g., user name, station name), system search functionalities as well as authentication form and configurations/settings parameters. Moreover, applying appropriate response headers (e.g., Content-Type and X-Content-Type-Options) and enforcing Content Security Policy (CSP) can reduce the severity and impact of any subtle XSS vulnerabilities that may still occur on the system. By properly patching XSS issues, the developers can mitigate attacks such as charging process and settings manipulation and data/record theft.

SQLi. To mitigate SQLi vulnerabilities (CWE-89) within EVCSMS, the developers have to prevent an adversary from executing SQL queries by abusing string concatenation issues on vulnerable parameters within the EVCSMS authentication forms which incorporate untrusted input treated as data such as the account username and password. In order, to resolve these issues, they have to use parameterized queries to distinguish code from data and prevent misinterpretation of variable data from arbitrary origins. Thus, any external data item should not be trusted and treated as potential threat by completely avoiding the usage of string concatenation in the EVCSMS handling queries. By properly patching SQLi issues which represent critical severity vulnerabilities that provide the adversary with full control over the EVCS, the developers can mitigate all attack scenarios discussed in the previous section such as firmware and billing manipulation as well as the attacks against the power grid. It should be noted that for mitigating firmware manipulation attacks where an adversary gains privileged access to the EVCSMS, the developers should implement stronger firmware version checks in order to prevent firmware downgrades and implement stronger signature checks to prevent firmware alteration by explicitly defining the hashes belonging to accepted firmware builds.

Information Disclosure. To patch information disclosure issues (CWE-200), the developers should enforce authentication on all endpoints such that an adversary can not induce the EVCSMS into unintentionally leaking sensitive information via direct or malformed requests on side endpoints. This can be achieved by compiling a list of critical endpoints and the information that they contain then securing them and implementing proper error handling to restrict the debugging information that gets revealed.

Missing Authentication. Similarly, to mitigate missing authentication vulnerabilities (CWE-306) on EVCSMS, the developers should enforce authentication on all functionalities within the system especially critical features such as configuration update and EVCS power options and restart settings, to prevent an unauthenticated adversary as well as an adversary who hijacked a target user's session from accessing and modifying them.

Embedded Secrets. To mitigate the threats such as PII leakage attacks that arise from embedded secrets (i.e., cryptographic keys) in mobile-based EVCSMS, vendors should instead make the system obtain these keys during runtime by securely connecting to back-end API endpoints that generate unique keys for each client user.

CSRF. To protect against CSRF vulnerabilities (CWE-352) within EVC-

SMS, the developers should secure every form with sensitive actions such as the one for changing user credentials, turning off the EVCS, and downloading charging records. This is achieved by appending unpredictable random token values (e.g., CSRF tokens) with each `GET`- and `POST`-based HTTP request to correlate and validate the corresponding actions and prevent an adversary from crafting malicious requests to override or hijack these actions and cause system modifications. Resolving CSRF vulnerabilities can mitigate several attacks such as DoS and charging process and settings manipulation.

Forced Browsing. To prevent forced browsing vulnerabilities (CWE-425), system developers should ensure that all sensitive endpoints and resources are correctly enforced with authorization models and access control mechanisms. That is by linking the specific endpoints and resources to particular authority-levels, and ensuring that only permitted entities with the corresponding privilege level can obtain access to them through the intended design path. Patching these vulnerabilities can mitigate settings manipulation and PII leakage attacks.

Hard-Coded Credentials. Although, hard-coded credentials are utilized by vendors for ease of deployment and scalability, they give rise to vulnerabilities (CWE-798) and attacks. Therefore, to mitigate these threats, the developers can take several steps to harden their acquisition by an adversary such as by creating complex credentials and hard-coding their corresponding salted hashes within the source code instead of directly placing them in plaintext. Additionally their values and locations can be obfuscated to make it more difficult to extract them. Moreover, the vendors could embed different passwords for each customer EVCSMS installation and accordingly prompting them on initial setup to change those credentials.

Missing Rate Limit. In order to mitigate attacks such as DoS which arise from the absence of rate limit (CWE-799), the developers should implement mechanisms to block excessive and fast requests such as a web application firewall (WAF), as well as insert temporal delays to reduce the frequency of repetitive actions such as brute-force attempts to guess endpoints or account credentials and repetitive attempts to restart the EVCS, which can lead to unauthorized access and directly/indirectly cause damage to the EVCS.

SSRF. To prevent SSRF vulnerabilities (CWE-918), the developers should rely on an alternative logic to replace passing IP/URL addresses information through parameters that can be tampered by the system user. In addition, they should implement checks to validate IP/URL addresses that are passed to the system and reject those that do not conform to a pre-compiled valid list in order to avoid crfated addresses that trick the EVCSMS into loopback. Furthermore, the EVCSMS should store a mapping between valid client-side target addresses and corresponding server-side tokens to prevent tampering attempts. Patching SSRF vulnerabilities can efficiently mitigate severe threats such as bot recruitment and network proxy attacks.

Cross-domain Policy Misconfigurations. To prevent attacks such as data/record theft and account information leakage that arise from lenient cross-domain policy such as CORS and FCDP misconfigurations (CWE-942) vulnerabilities, the developers should explicitly specify trusted origins from where the required sensitive resource can be requested. That is by explicitly specifying the external domains that are allowed to interact with the EVCSMS.

CSVi. To mitigate threats and attacks that arise from exploiting CSVi vulnerabilities (CWE-1236) due to EVCSMS storing and managing charging records in CSV format, the developers should ensure that the EVCSMS safely parses the supplied/stored files and rejects those with malformed and dangerous characters that are used to trigger or execute code. In addition, if the EVCSMS relies on a third-party software to parse the CSV files, the developers should ensure that the parser distribution is up-to-date and patched against the latest bugs.

5.2. Security Measures, Guidelines, and Best Practices

To mitigate mass cyber attacks that target the power grid, the developers have to properly patch all the aforementioned vulnerabilities specifically those with critical and high severity/impact (e.g., SQLi), which give the adversary full control over the EVCSMS in order to effectively prevent remote exploitation and manipulation of the underlying EVCS. It is important for vendors and developers to continuously assess the security of their EVCSMS while implementing the necessary patches. It is also essential to integrate security by design, through finding and addressing such vulnerabilities during the product development stage which will reduce the burden of re-designing and re-assessing the deployed systems while avoiding known security issues [46].

Additionally, to mitigate demand-supply manipulation attacks against the power grid (Section 4.3), there are several countermeasures that aim at preventing charging schedule manipulation. The power grid operators can perform early attack detection by frequently monitoring the charging schedules and the status of the connected EVCS to detect anomalies in the charging behaviour. This process can be automated by leveraging machine learning (ML) models to design an anomaly detection system that constantly monitors the charging records collected from the data streams of EVCS smart meters to learn normal patterns and warn the operators when malicious patterns are detected. This allows the operators to react to anomalies and activate contingency plans to handle attack scenarios. It should be noted that the success of this anomaly-detection strategy implies establishing a trust model between the power grid and EVCS operators in order for them to exchange data. From the literature, Basnet et al. [47] tackled the risks of open communication layer to physical layer of the power grid (e.g., bidirectional communication, smart resource management) by proposing a deep learning-based intrusion detection system (IDS) to detect DoS attacks within EVCS in an attempt to fortify their trustworthiness towards the grid. In addition, Mousavian et al. [48] proposed a mixed integer linear programming model that optimizes security risk within the EVCS communication networks to handle attacks such as malware propagation through infected charging stations.

Moreover, to prevent an adversary from tampering with the EV charging schedules and the EVCS configurations that relate to the EV, the corresponding EVCSMS and EV system can implement a mutual consensus to validate modifications that occur on any of their settings. For instance, to make changes to the EV charging schedules, the EVCSMS would require the EVCS to notify the EV operator/user of this requested change in order for them to approve or decline it. In this way, an adversary who compromised the EVCSMS and gained control over the EVCS can not enforce custom charging schedule configurations without obtaining the approval of the other participating entities such as the EV operator/user.

Furthermore, while it is the EVCSMS developers' primary task to produce secure-by-design systems, the EVCS users also need to properly and securely setup their charging stations in order to prevent some attacks. Thus, we provide some guidelines to raise user-awareness. A first step is to always change the default credentials that are set on the EVCS firmware. Additionally, users should setup remote authentication methods with strong account credentials. These steps can be effective to mitigate automatic and large-scale cyber attacks to compromise online EVCS using default and/or weak credentials (e.g., The Mirai Botnet [2]). Additionally, users must avoid interacting with untrusted websites/emails that masquerade as EV product vendors since attackers typically utilize them to embed malicious code and carry out attacks against the corresponding EVCSMS. Furthermore, private EVCS users can disable public device discovery on their EVCSMS portals to hide them from remote attackers on the Internet and reduce the attack surface. In addition, it is always recommended to configure a firewall by setting different rules, which only allows traffic and connections between trusted parties.

6. Discussion

In the context of the EV charging ecosystem, similarly to the insecurities that exist in the IoT ecosystem [2], the range of extended EVCS remote functionalities open doors to various cyber attacks. However, there is a lack of knowledge about the security of the growing number of deployed EVCS in the wild and the EVCSMS that instrument them, especially when the studies from the literature were limited to theoretical attacks and specific scenarios that require extensive setup. Therefore, in this paper, we perform an in-depth security analysis of 16 EVCSMS developed by globally recognized vendors such as Schneider Electric and we highlight major security flaws in these systems, which have been surprisingly overlooked by the respective developers.

Specifically, we uncovered various zero-day vulnerabilities (e.g., SQLi and XSS), which demonstrate the insecurity of the deployed systems within the EV charging ecosystem. Furthermore, we showed that in practice, adversaries can leverage the identified severe vulnerabilities to perform an array of cyber attacks, which result in compromising the EVCS and impacting its resources, data, operations, and the security of its users. More importantly, we conduct simulation analysis to demonstrate the feasibility of leveraging these compromised charging stations to perform frequency instability attacks against the interconnected critical infrastructure such as the power grid [26, 32, 42, 43]. Indeed, our analysis highlighted several attack scenarios that can utilize compromised EVCS to cripple the operations of the power grid.

It is worth noting that the discussed vulnerability classes are known to the security community [25] and have been examined/addressed in other contexts [49]. Nevertheless, the fact that they have not been addressed in the context of EVCS is alarming and implies the absence of security consideration when deploying EVCS and designing the management systems. We believe that such insecure design/implementation could be linked to several factors. For instance, the EV technologies are relatively new yet rapidly growing. Therefore, vendors might be prioritizing production to keep up with the competition and the significant market demands while overlooking some security requirements by investing less time/effort to conduct in-depth security analysis and evaluation. Despite that, this study raises attention towards the insecurity of the EV charging ecosystem and calls for prompt actions by proposing several countermeasures to protect/patch existing EVCSMS and mitigate future large-scale cyber attacks.

Ethical Consideration and Zero-day Vulnerabilities. We conducted our EVCSMS lookup/collection and security analysis in September 2020, and we communicated our findings to the respective system vendors/providers of the analyzed EVCSMS prior to publishing our results in order for them to take the necessary actions towards securing their products. In fact, a number of vendors (e.g., Cornerstone Technologies, Bluesky Energy, Etrel) have acknowledged the identified vulnerabilities and addressed them. In addition, Schneider Electric reviewed our reported vulnerabilities and reserved 12 CVE-IDs accordingly.

Limitations. While we identified and analyzed 16 EVCSMS in this work, it is worth noting that obtaining information about all available EVCSMS in the wild is an extremely challenging task. This is mainly due to the proprietary nature of some EVCSMS platforms, which are only provided to enterprise-level customers or Charging Point Operators (CPO) with a prepaid subscription. Another limitation is the hardware-dependency of some EVCSMS whose developers do not offer the firmware packages. Hence, making the analysis process dependent on memory dumps that must be collected from actual stations. Additionally, within the web app analysis that we oversee, several systems are only examined from the public-facing front (i.e., authentication form), without internal access beyond the login interface. Therefore, we were unable to download the document collection and closely examine its components. Moreover, while we were restricted to utilize default credentials on these systems, the security of some EVCSMS login forms makes it unfeasible to inspect the post-authentication content, which would possibly accommodate further vulnerabilities.

Future Work. In terms of possible future work, it is important to note that our current approach requires a considerable amount of manual analysis and inspections, which relies on domain knowledge and expertise in the field of security auditing and testing. Nevertheless, given the fact that the majority of the identified vulnerabilities are known to the security community, our approach can be generalized by automating parts of the analysis using custom tools. Moreover, to make the analysis scalable, we plan to leverage AI-based techniques and models to learn structural and contextual characteristics of the analyzed systems, while providing means for automated vulnerability analysis and detection when dealing with new systems. Finally, motivated by the overall insecurity of the analyzed EVCSMS, we aim at expanding our knowledge about deployed EVCSMS by developing an approach for large-scale EVCS device discovery and fingerprinting. While the outcomes of such study will contribute towards identifying other EVCSMS products and vendors, it can be used to quantify the cyber threats associated with the deployed charging stations in the wild.

7. Related Work

Several studies have looked at various aspects of EV charging ecosystem security. However, EVCS firmware and EVCSMS security have received surprisingly little to no academic attention compared to other facets of the EV ecosystem. To the best of our knowledge, this work is the first to conduct security analysis on such systems originating from several vendors and the first to thoroughly present a multitude of vulnerabilities that can be leveraged to attack EVCS in real-world circumstances. In what follows, we give a state-of-the-art review of previous security-oriented research within the context of the EV charging ecosystem.

Firmware and Management System. To the best of our knowledge, there are no studies from the literature that examine the security posture of EVCS and their management systems, however from outside of academia, Kaspersky Lab's team [50] analyzed the security of ChargePoint home charging station and found significant vulnerabilities in its firmware and mobile management app. Furthermore, Schneider Electric [51] released a security advisory for three high severity vulnerabilities affecting its EVlink product line. In our study, we provide a systematic and detailed analysis of 16 EVCSMS including firmware-based, web-based and mobile-based products.

Communication and Protocol. Alcaraz et al. [12] presented weaknesses in the OCPP that could allow for manipulator-in-the-middle (MITM) attacks which lead to interfering with EV resource reservation potentially disrupting the stability of power provisioning networks. While their research did not consider the disposition of EVCS security itself, they examined the imperative component of protocol security which can be leveraged to attack the EV charging ecosystem. The same authors [13], addressed the attacks from their first paper and provided countermeasures. Furthermore, these security issues were mitigated in the latest OCPP releases (versions 2.0 and 2.0.1) by implementing various security measures to harden the protocol such as security profiles for authentication via key management for client-side certificates and secure communication through Transport Layer Security (TLS) [52]. In other protocol-related studies Boe et al. [53] and Lee et al. [54] performed a security analysis of the vehicle-to-grid charging protocol ISO 15118 and presented attack scenarios to compromise the charging process. Baker et al. [55] implemented the first wireless eavesdropping tool tat leverages electromagnetic side-channel attacks against power-line communication (PLC) networks and used it to recover messages and traffic from close-proximity EVCS. The aforementioned studies from the literature provide viable cyber attacks to target and compromise EVCS, however the discussed attack scenarios require extensive setup or close proximity, while in our study, we provide practical vulnerabilities that allow an adversary to remotely compromise a target EVCS.

Electric Vehicle. From the literature, several studies discussed vulnerabilities in electric vehicles which arise from their in-vehicular networks of electronic control units (ECU) [36]. From the literature, several work have discussed vehicular cyber attacks, where an attacker who tampers with the controller area network (CAN) bus or individual ECU can control the vehicle's operations [56–58]. In addition, Rouf et al. [59] presented security flaws in the tire pressure monitoring system (TPMS) that allow an attacker to eavesdrop, reverse engineer and spoof the communication within range of an EV. Checkoway et al. [60] presented the attack surface enabled by the physically accessible ports available within a vehicle (EV included) which could be exploited to perform DoS or side-channel attacks against the ECUs. Furthermore, Zeng et al. [61] discussed cyber attacks that can jam the network and spread misinformation to the vehicle's road-side units (RSU). While the aforementioned studies examine the security of EV which is an important entity in the EV charging ecosystem, the discussed vulnerabilities can not be leveraged to perform remote and wide-scale cyber attacks against the ecosystem, which in our research we are able to perform by discovering and leveraging vulnerabilities in EVCSMS.

Infrastructure. From the literature, several studies centered around EV charging infrastructure and the potential cyber threats that can impact it. Pratt et al. [28] discussed the security of EVCS and electric power grids while identifying different threats and devising security principles to prevent cyber attacks against the EV charging infrastructure. Gottumukkala et al. [62] analyzed the security threats against the EV charging infrastructure with respect to confidentiality, integrity and availability (i.e., CIA triad) then provided mitigations for these attacks. Antoun et al. [63] presented an overview of cyber threats targeting the entities that constitute the EV charging system, classified them according to the CIA triad and presented literature solutions for each. Furthermore, Fraiji et al. [27] surveyed the security of the various entities within the Internet-of-Electric-Vehicles (IoEV) drawing attention to the different attacks that can be used to disrupt the operations in this architecture. Acharya et al. [26] utilized the vulnerabilities pinpointed by Alcaraz et al. [12] to create an attack against the power grid by creating a botnet of compromised high wattage EVCS. However, their work did not discuss the feasibility of such attacks and exploitation in real-life scenarios, in addition to assuming that the utilized EVCS are vulnerable without specifying actual exploitation vectors and techniques, which in our study we present and give details about.

Vehicle-to-Grid Technologies. Several studies have discussed the security of the vehicle-to-grid (V2G) technologies, which handle the process of feeding the energy stored in an EV battery back into the power grid. For instance, Saxena et al. [64] discussed the network security and privacy requirements and challenges of V2G and proposed a privacy-aware scheme with features such as anonymous authentication and fine-grained access control. Zhou et al. [37] proposed a framework for secure V2G energy trading by utilizing blockchain, contract theory, and edge computing. In addition, other works have been conducted to propose privacy and authentication schemes for V2G networks [65–68]. Moreover, Mousavian et al. [48] proposed a mixed integer linear programming model that optimizes security risk within the EV infrastructure to handle an epidemic attack model such as malware propagation through infected devices within the EVCS communication networks. Note that they rely on a purely theoretical attack scenario where the EVCSs are assumed to be infected without discussing the exploitation process. Nevertheless, in this work, we identify actual vulnerabilities that can be leveraged to infect a large body of EVCS through their insecure management systems (e.g, firmware manipulation through XSS and SQLi). Additionally, while the aforementioned studies mainly focus on enhancing the V2G interaction at a high level, our work represents an in-depth analysis of the security posture of deployed EVCS in the wild. Indeed, while this work explores practical attack implications, it also sheds light on effective mitigating countermeasures that aim at addressing the existing security flaws within the EVCSMS and therefore, contributing towards improving the overall security of the EV charging ecosystem.

8. Conclusion

In this work, we give an overview of the EV charging ecosystem and its main physical components as well as its software and protocol constituents. Furthermore, we present our approach in which we collect EVCS firmware, web-based and mobile-based EVCSMS, then conduct a thorough security analysis on each of them. By highlighting our findings, we demonstrate that the EV charging ecosystem — one of the world's most proliferating ecosystems — suffers from critical vulnerabilities within its most fundamental entities, EVCS and their management systems, leaving the overall hierarchy at a high risk of cyber attacks. More importantly, while we discuss practical attack implications against the EVCS and its users, we demonstrate the feasibility of cyber attacks against the operations of the interconnected power grid, leading to possible instabilities and service disruptions. Finally, while we shed light on the feasibility of cyber attacks using the identified vulnerabilities, we recommend a number of practical countermeasures that aim at securing the design and implementation of existing and/or new systems, while providing security guidelines and best practices for developers as well as end users and power grid operators.

Acknowledgments

We thank the reviewers in advance for their time and feedback. This work has been supported by Natural Sciences and Engineering Research Council of Canada (NSERC) and Concordia University. This work was also partially supported by a grant from the U.S. National Science Foundation (NSF), Office of Advanced Cyberinfrastructure (OAC), #1907821.

References

[1] The International Energy Agency, Global EV Outlook 2020, Retrieved from `https://www.iea.org/reports/global-ev-outlook-2020` (Jun. 2020).

[2] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, Y. Zhou, Understanding the mirai botnet, in: 26th USENIX Security Symposium (USENIX Security 17), USENIX Association, Vancouver, BC, 2017, pp. 1093–1110.

[3] C. Galbrun-Noel, How the IONITY and Schneider Electric Partnership Drives eMobility While Expanding Europe's EV Charging Infrastructure, Retrieved from `https://blog.se.com/automotive-mobility/2019/11/07/ionity-schneider-electric-partnership-drives-emobility-europes-ev-charging-infrastructure` (2019).

[4] National Vulnerability Database, Common Vulnerabilities and Exposures (CVE), Retrieved from `https://cve.mitre.org` (2021).

[5] Netherlands Enterprise Agency, Electric Vehicle Charging: Definitions and Explanation, Retrieved from `https://www.rvo.nl/sites/default/files/2019/01/Electric\%20Vehicle\%20Charging\%20-\%20Definitions\%20and\%20Explanation\%20-\%20january\%202019_0.pdf` (Jan. 2019).

[6] Office of Energy Efficiency & Renewable Energy, Vehicle Charging, Retrieved from `https://www.energy.gov/eere/electricvehicles/vehicle-charging` (2020).

[7] I. Rahman, P. M. Vasant, B. S. M. Singh, M. Abdullah-Al-Wadud, N. Adnan, Review of recent trends in optimization techniques for plug-in hybrid, and electric vehicle charging infrastructures, Renewable and Sustainable Energy Reviews 58 (2016) 1039–1047.

[8] R. J. Flores, B. P. Shaffer, J. Brouwer, Electricity costs for an electric vehicle fueling station with level 3 charging, Applied Energy 169 (2016) 813–830.

[9] G. Joos, M. de Freige, M. Dubois, Design and simulation of a fast charging station for phev/ev batteries, in: 2010 IEEE Electrical Power Energy Conference, 2010, pp. 1–5. doi:10.1109/EPEC.2010.5697250.

[10] Hydro-Quebec, Electric Vehicle Charging Stations: Technical Installation Guide, Retrieved from `https://www.hydroquebec.com/data/electrification-transport/pdf/technical-guide.pdf` (2015).

[11] OCPP, OCPP 2.0 Part 2: Specification. Technical Report, Retrieved from `https://smartcharge.com.br/artigos/ocpp/OCPP-2.0_part2_specification.pdf` (2018).

[12] C. Alcaraz, J. Lopez, S. Wolthusen, OCPP Protocol: Security Threats and Challenges, IEEE Transactions on Smart Grid 8 (5) (2017) 2452–2459.

[13] J. E. Rubio, C. Alcaraz, J. Lopez, Addressing Security in OCPP: Protection Against Man-in-the-Middle Attacks, in: 2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Paris, France, 2018, pp. 1–5.

[14] Exploit Database, Google Hacking Database, Retrieved from `https://www.exploit-db.com/google-hacking-database` (2021).

[15] Shodan, The search engine for Internet-connected devices, Retrieved from `https://www.shodan.io` (2021).

[16] Censys, A search engine based on Internet-wide scanning for the devices and networks, Retrieved from `https://censys.io` (2021).

[17] Linux, dd(1) — Linux manual page, Retrieved from `https://man7.org/linux/man-pages/man1/dd.1.html` (2021).

[18] Refirm Labs, Binwalk: Nb 1 Firmware extraction tool in the world, Retrieved from `https://www.refirmlabs.com/binwalk` (2021).

[19] R. Wiśniewski, Apktool: A Tool for Reverse Engineering Android APK Files, Retrieved from `https://ibotpeaches.github.io/Apktool` (2021).

[20] E. Dupuy, JD-GUI: Java Decompiler, Retrieved from `https://java-decompiler.github.io` (2021).

[21] B. Pan, Pdex2jar, Retrieved from `https://github.com/pxb1988/dex2jar` (2021).

[22] X. Roche, HTTrack Website Copier, Retrieved from `https://www.httrack.com` (2021).

[23] The OWASP Foundation, OWASP Testing Guide (4.0), Retrieved from `https://owasp.org/www-project-web-security-testing-guide/assets/archive/OWASP_Testing_Guide_v4.pdf` (May 2021).

[24] The MITRE Corporation, Common Weakness Enumeration (CWE), Retrieved from `https://cwe.mitre.org` (2021).

[25] The OWASP Foundation, Open Source Foundation for Application Security, Retrieved from `https://owasp.org` (2021).

[26] S. Acharya, Y. Dvorkin, R. Karri, Public plug-in electric vehicles + grid data: Is a new cyberattack vector viable?, IEEE Transactions on Smart Grid 11 (6) (2020) 5099–5113. doi:10.1109/TSG.2020.2994177.

[27] Y. Fraiji, L. B. Azzouz, W. Trojet, L. A. Saidane, Cyber Security Issues of Internet of Electric Vehicles, in: 2018 IEEE Wireless Communications and Networking Conference (WCNC), Barcelona, Spain, 2018, pp. 1–6.

[28] R. M. Pratt, T. E. Carroll, Vehicle Charging Infrastructure Security, in: 2019 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 2019, pp. 1–5.

[29] G. Liang, S. R. Weller, J. Zhao, F. Luo, Z. Y. Dong, The 2015 ukraine blackout: Implications for false data injection attacks, IEEE Transactions on Power Systems 32 (4) (2016) 3317–3318.

[30] N. Ferris, S. van Renssen, Cybersecurity Threats Escalate In the Energy Sector, Retrieved from `https://energymonitor.ai/tech/digitalisation/cybersecurity-threats-escalate-in-the-energy-sector` (2021).

[31] Wired Magazine, The Highly Dangerous 'Triton' Hackers Have Probed the US Grid, Retrieved from `https://www.wired.com/story/triton-hackers-scan-us-power-grid` (jun 2021).

[32] S. Soltan, P. Mittal, H. V. Poor, Blackiot: Iot botnet of high wattage devices can disrupt the power grid, in: 27th USENIX Security Symposium (USENIX Security 18), USENIX Association, Baltimore, MD, 2018, pp. 15–32.

[33] PowerWorld Corporation, The PowerWorld Simulator, Retrieved from https://www.powerworld.com (2021).

[34] A. Muir, J. Lopatto, Final report on the august 14, 2003 blackout in the united states and canada: causes and recommendations, Retrieved from https://www.energy.gov/sites/default/files/oeprod/DocumentsandMedia/BlackoutFinal-Web.pdf (apr 2004).

[35] Illinois Center for a Smarter Electric Grid (ICSEG), Power test cases, Retrieved from https://icseg.iti.illinois.edu/wscc-9-bus-system (2021).

[36] S. Acharya, Y. Dvorkin, H. Pandžić, R. Karri, Cybersecurity of smart electric vehicle charging: A power grid perspective, IEEE Access 8 (2020) 214434–214453. doi:10.1109/ACCESS.2020.3041074.

[37] Z. Zhou, B. Wang, M. Dong, K. Ota, Secure and efficient vehicle-to-grid energy trading in cyber physical systems: Integration of blockchain and edge computing, IEEE Transactions on Systems, Man, and Cybernetics: Systems 50 (1) (2020) 43–57. doi:10.1109/TSMC.2019.2896323.

[38] PowerWorld Corporation, Transient Stability Overview: Renewable Energy Generation Modeling (Wind, Solar, Energy Storage, Distributed Photo Voltaic), Retrieved from https://www.powerworld.com/WebHelp/Content/MainDocumentation_HTML/Transient_Stability_Overview_WindModeling.htm (2021).

[39] O. Erdinc, N. G. Paterakis, T. D. Mendes, A. G. Bakirtzis, J. P. Catalao, Smart household operation considering bi-directional ev and ess utilization by real-time pricing-based dr, IEEE Transactions on Smart Grid 6 (3) (2014) 1281–1291.

[40] Input Magazine, NYC will add 100 curbside EV chargers by October 2021, Retrieved from https://www.inputmag.com/tech/nyc-will-add-100-curbside-ev-chargers-by-october (2021).

[41] Gouvernement du Québec, New Electric Vehicle Rebate, Retrieved from https://vehiculeselectriques.gouv.qc.ca/english/rabais/ve-neuf/programme-rabais-vehicule-neuf.asp (2020).

[42] A. K. Farraj, D. Kundur, On using energy storage systems in switching attacks that destabilize smart grid systems, in: 2015 IEEE Power Energy Society Innovative Smart Grid Technologies Conference (ISGT), 2015, pp. 1–5. doi:10.1109/ISGT.2015.7131855.

[43] L. Shan, C. Bo, Z. Takis, K. Deepa, B.-P. Karen, A Coordinated Multi-Switch Attack for Cascading Failures in Smart Grid, IEEE Transactions on Smart Grid 5 (3) (2014) 1183–1195.

[44] F. Sagstetter, M. Lukasiewycz, S. Steinhorst, M. Wolf, A. Bouard, W. R. Harris, S. Jha, T. Peyrin, A. Poschmann, S. Chakraborty, Security challenges in automotive hardware/software architecture design, in: 2013 Design, Automation Test in Europe Conference Exhibition (DATE), 2013, pp. 458–463. doi:10.7873/DATE.2013.102.

[45] M. Weissbacher, W. Robertson, E. Kirda, C. Kruegel, G. Vigna, Zigzag: Automatically hardening web applications against client-side validation vulnerabilities, in: 24th USENIX Security Symposium (USENIX Security 15), USENIX Association, Washington, D.C., 2015, pp. 737–752.

[46] H. Assal, S. Chiasson, Security in the software development lifecycle, in: Fourteenth Symposium on Usable Privacy and Security (SOUPS 2018), USENIX Association, Baltimore, MD, 2018, pp. 281–296.

[47] M. Basnet, M. Hasan Ali, Deep learning-based intrusion detection system for electric vehicle charging station, in: 2020 2nd International Conference on Smart Power Internet Energy Systems (SPIES), 2020, pp. 408–413. doi:10.1109/SPIES48661.2020.9243152.

[48] S. Mousavian, M. Erol-Kantarci, L. Wu, T. Ortmeyer, A risk-based optimization model for electric vehicle infrastructure response to cyber attacks, IEEE Transactions on Smart Grid 9 (6) (2018) 6160–6169. doi:10.1109/TSG.2017.2705188.

[49] O. Alrawi, C. Zuo, R. Duan, R. P. Kasturi, Z. Lin, B. Saltaformaggio, The betrayal at cloud city: An empirical analysis of cloud-based mobile backends, in: 28th USENIX Security Symposium (USENIX Security 19), USENIX Association, Santa Clara, CA, 2019, pp. 551–566.

[50] S. Dmitry, ChargePoint Home Security Research, Retrieved from `https://media.kasperskycontenthub.com/wp-content/uploads/`

sites/43/2018/12/13084354/ChargePoint-Home-security-research_final.pdf (Dec. 2018).

[51] Schneider Electric, Schneider Electric Security Notification: Security Notification – EVLink Parking, Retrieved from https://download.schneider-electric.com/files?p_enDocType=Software+-+Release+Notes&p_File_Name=SEVD-2018-354-01_Security+Notification.pdf&p_Doc_Ref=SEVD-2018-354-01 (May 2018).

[52] Open Charge Alliance, Open Charge Point Protocol 2.0.1, Retrieved from https://www.openchargealliance.org/protocols/ocpp-201 (2021).

[53] K. Bao, H. Valev, M. Wagner, H. Schmeck, A Threat Analysis of the Vehicle-to-Grid Charging Protocol ISO 15118, Computer Science-Research and Development 33 (1-2) (2018) 3–12.

[54] S. Lee, Y. Park, H. Lim, T. Shon, 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), Beijing, China, 2014, pp. 1–4.

[55] R. Baker, I. Martinovic, Losing the car keys: Wireless phy-layer insecurity in EV charging, in: 28th USENIX Security Symposium (USENIX Security 19), USENIX Association, Santa Clara, CA, 2019, pp. 407–424.

[56] S. Woo, H. J. Jo, D. H. Lee, A practical wireless attack on the connected car and security protocol for in-vehicle can, IEEE Transactions on Intelligent Transportation Systems 16 (2) (2015) 993–1006. doi:10.1109/TITS.2014.2351612.

[57] I. Foster, A. Prudhomme, K. Koscher, S. Savage, Fast and vulnerable: A story of telematic failures, in: 9th USENIX Workshop on Offensive Technologies (WOOT 15), USENIX Association, Washington, D.C., 2015, pp. 1–9.

[58] R. Currie, Hacking the can bus: basic manipulation of a modern automobile through can bus reverse engineering, SANS Institute (2017).

[59] I. Rouf, R. Miller, H. Mustafa, T. Taylor, S. Oh, W. Xu, M. Gruteser, W. Trappe, I. Seskar, Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study, in: 19th USENIX Security Symposium (USENIX Security 10), USENIX Association, Washington, DC, 2010, pp. 1–16.

[60] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno, Comprehensive experimental analyses of automotive attack surfaces, in: 20th USENIX Security Symposium (USENIX Security 11), USENIX Association, San Francisco, CA, 2011, pp. 1–16.

[61] K. Zeng, S. Liu, Y. Shu, D. Wang, H. Li, Y. Dou, G. Wang, Y. Yang, All your GPS are belong to us: Towards stealthy manipulation of road navigation systems, in: 27th USENIX Security Symposium (USENIX Security 18), USENIX Association, Baltimore, MD, 2018, pp. 1527–1544.

[62] R. Gottumukkala, R. Merchant, A. Tauzin, K. Leon, A. Roche, P. Darby, Cyber-physical System Security of Vehicle Charging Stations, in: 2019 IEEE Green Technologies Conference (GreenTech), Lafayette, LA, USA, 2019, pp. 1–5.

[63] J. Antoun, M. E. Kabir, B. Moussa, R. Atallah, C. Assi, A Detailed Security Assessment of the EV Charging Ecosystem, IEEE Network 34 (3) (2020) 200–207.

[64] N. Saxena, S. Grijalva, V. Chukwuka, A. V. Vasilakos, Network security and privacy challenges in smart vehicle-to-grid, IEEE Wireless Communications 24 (4) (2017) 88–98. doi:10.1109/MWC.2016.1600039WC.

[65] L. Chen, J. Zhou, Y. Chen, Z. Cao, X. Dong, K.-K. R. Choo, Padp: Efficient privacy-preserving data aggregation and dynamic pricing for vehicle-to-grid networks, IEEE Internet of Things Journal 8 (10) (2021) 7863–7873. doi:10.1109/JIOT.2020.3041117.

[66] N. Saxena, B. J. Choi, Authentication scheme for flexible charging and discharging of mobile vehicles in the v2g networks, IEEE Transactions on Information Forensics and Security 11 (7) (2016) 1438–1452. doi:10.1109/TIFS.2016.2532840.

[67] M. He, K. Zhang, X. S. Shen, Pmqc: A privacy-preserving multi-quality charging scheme in v2g network, in: 2014 IEEE Global Communications Conference, 2014, pp. 675–680. doi:10.1109/GLOCOM.2014.7036885.

[68] H. Liu, H. Ning, Y. Zhang, M. Guizani, Battery status-aware authentication scheme for v2g networks in smart grid, IEEE Transactions on Smart Grid 4 (1) (2013) 99–110. doi:10.1109/TSG.2012.2224387.