Collaborative detection and response framework against SQL injection attacks in IoT-based smart grids

Chahira Boukhari[1], Abdelouahid Derhab[2], Mohamed Guerroumi[3], Nadia Nouali[1], Abdelaziz Babakhouya[1], and Abdelkarim Meziane[1]

[1] Research Center for Scientific and Technical Information, Algeria.
[2] Center of Excellence in Information Assurance, King Saud University, Saudi Arabia.
[3] USTHB University, Algeria

Abstract. In this paper, we propose a collaborative detection and response framework against SQL injection attacks in IoT-based smart grids. The framework is composed of a set of host-based detection systems; each of which is deployed at a smart meter, in addition, at the data management server. When an attack at one host is detected, the network administrator is notified and remotely patches the other hosts. The detection engine is lightweight as each smart meters analyzes the log file associated with its network traffic. Hence, the framework is sacalable to large IoT-based smart grids as the detection task is performed by each smart meter and does not rely on a single component. Prelimary results are promising in terms of true positive and false positive rates.

Key words: IoT-based Smart grid, SQL injection, log file, collaborative detection.

1 Introduction

The Smart Grid (SG) can be seen as one of the largest deployment of internet of things (IoT).

The energy grid chain, which starts from the energy power generators to the electricity consumers, and comprises transmission and distribution power networks, are deployed with additional intelligence devices that have computational and communication functionalities, in order to efficiently monitor and control the energy grid [1].

The IoT-based SG is composed of billions of smart objects/things including: sensors, smart appliances, smart meters, as well as multiple communication components within the smartgrid infrastructure. The main goal of the smart grids is to provide the consumers with updated information regarding their energy consumption. This information must be managed correctly to avoid indesirable consequences. As the web browsers can be found in any desktop machine, laptop, or smartphone, web applications are playing a crucial role in smart grid implementations. Therefore, data that transit through web applications play a major role in the correct functionality of smart, as they should be accurate. Hence, appropriate security controls are required to ensure the integrity property, and that the collected data are not tampered [2]. The consumers could be a threat/victim to/of the utility companies. For example, invalid operations or execution of malicious programs might result in indesirable effects. Even incorrect handling of a safe and valid input or code might cause unexpected results. Structured Query Language (SQL) injection is one of the most dangerous attacks [3], as it might tamper with the utility databases (i.e., pricing database), and hence has devastating impact on the correct functionality of the smart grid. In order to protect consumers from the threats and to secure web applications against the risk of SQL injections, countermeasures must be taken into account. Detection of SQL injection has been widly investiged in the literature [4][5][6][7][8][9][10]. To the best of our knowledge, no work has studied the SQL injection issue in smart grid and how to cope with the limitation capacity of smart meters. Two major approaches are used to detecte SQL injection: static analysis which examines the web application code, and dynamic analysis, which monitors the web application's behavior. Although static analysis can detect the attack beforehand, but it cannot uncovers all the attacks. In dynamic analysis, it is difficult for the attacker to hide his/her malicious activities. For this reason, we adopt in this paper the dynamic analysis approach. As the detection system is intended to be deployed on resource-constraind smart meters, we use the regular expressions for analyzing the log file. The main contributions of this paper are as follows:

- We propose a framework for the detection of SQL injection attacks. The framework is composed of a set of host-based detection systems, each of which is deployed on a smart meter to capture and analyze web traffic related to one consumer. Additionally, a detection system is deployed on the Data Management Server (DMS), which stores the users' database, i.e., the set of detectors are deployed at all the possible entry points that can be exploited by the SQL injection attacks.
- We propose a collaborative scheme among the detectors in order to share the generated alerts. In this way, detecting a vulnerable web application at one host might prevent the compromise of other hosts by the same vulnerability.
- The framework is enhanced by a response system, which blocks the traffic generated from the source of the attack. It notifies the network administrator at the DMS, in order to remotely patch the web applications at the other hosts with updated web components.
- The detection engine that is used by the detectors is lightweight as it only analyzes the log file using regular expressions. Additionally, the entries in the log file are replaced in FIFO order in order to cope with the storage limitation of smart meters. There is no replacement policy at the DMS, which is characterized by high-volume data storage.

The rest of the paper is organized as follows: Section 2 presents the Smart Grid architecture. Section 3 describes in detail the collaborative detection and response framework. Section 4 provides experimental results. We finally end with a conclusion in section 5.

2 Smart grid architecture

The standard architecture of the smart grid was proposed by the The National Institute of Standards and Technology (NIST), which is composed of seven domains: Bulk Generation, Transmission, Distribution, Customer, Markets, Service Provider and Operations as shown in Fig. 1. The first four domains represent the bidirectional power and information flows. The last three domains represent the information collection and the power management in the Smart grid.

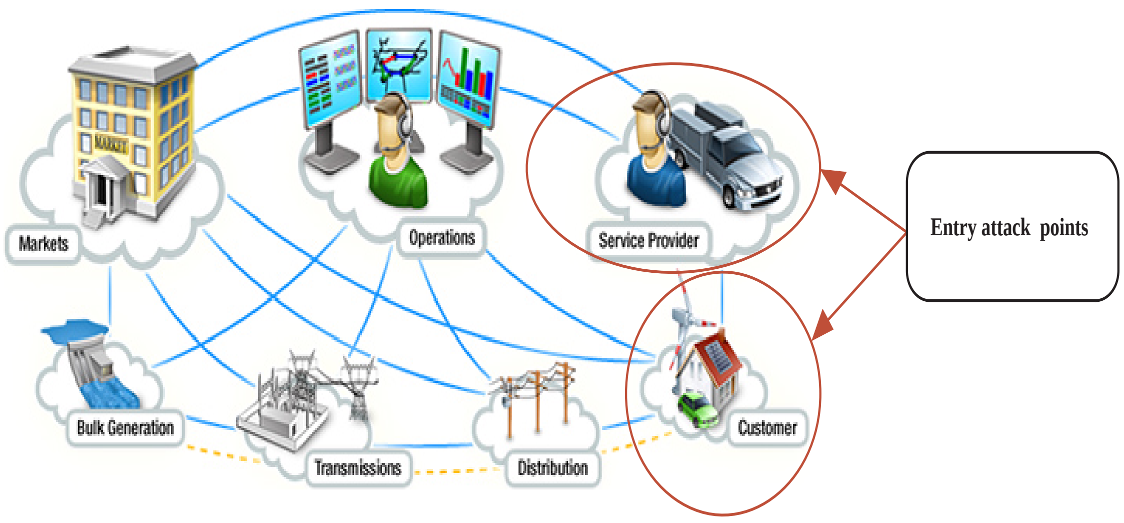

Fig. 1. Domains of a smart grid [NIST].

In our work, we focus on the two major components in smart grid: Service provider domain and customer domain as they are the entry attack points used in SQL injection attacks. Service Providers make contracts with users to give electricity for home appliances as well as individual devices. They also interact with the internal devices via messages flowing through the smart meter. To implement such interactions, the service providers should register with the energy utility and get digital certificates to compute public key and ensure secure communications with the users. Customer domain contains millions of smart meters, which periodically store the updated power consumption and transmit it to the utility server. They also perform some operation such as: connecting or disconnecting a customer power supply, and sending alarms in case of an anomaly. Some smart meters have relays that are interfaced with smart home devices to control them. For example, turning off the air conditioner during peak consumption periods. In addition, the smart meter might be utilized in demand side control [11].

3 Framework description

3.1 Framework architecture

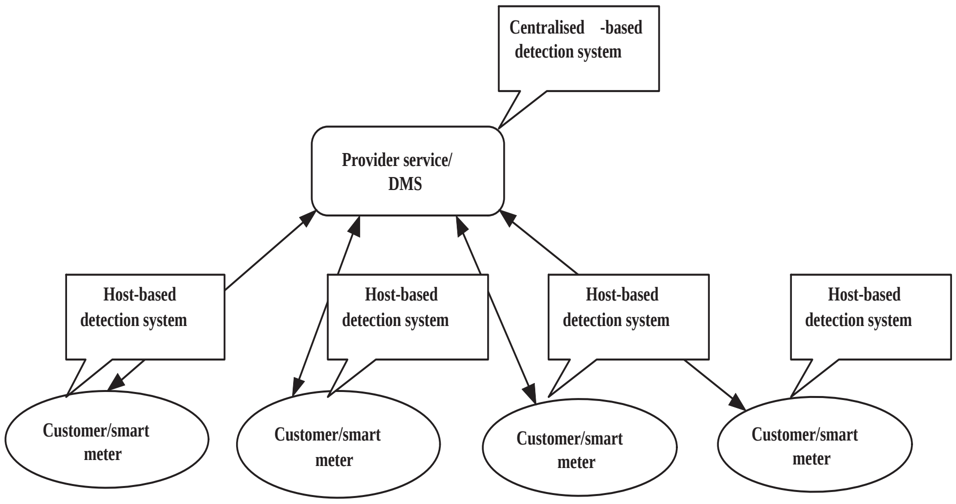
Centralised -based detection system
Provider service/ DMS
Host-based detection system
Host-based detection system
Host-based detection system
Host-based detection system
Customer/smart meter
Customer/smart meter
Customer/smart meter
Customer/smart meter

Fig. 2. Framework architecture

As smart grid might consist of millions of smart meters. The centralized architecture means that DMS has to analyze huge amount of traffic that is generated by these smart meters, and hence this architecture is not scalable with respects to large nomber of subscribers (i.e, number of smart meters). For this reason, we adopt in this paper, a decentralized architecture, in which a lightweight detector is deployed on each smart meter. As shown in Fig. 2 we deploy additionally, a heavyweight detection system at the service provider.

If an attack is detected at one host, the network administrator at the DMS identifies the vulnerable web component, corrects it, and remotly patches the web applications that are deployed in the network. The advantage of this architecture is the ability to share attacks among the different hosts detectors, which hepls preventing the compromise of other hosts by the same vulnerability.

3.2 Detection system description

The detection system is based on the analysis of the Apache web server log file. The Apache HTTP Server provides a variety of different mechanisms for logging everything that can happen within the web server, from the initial request, through the mapping process URLs (Uniform Resource Locator) until closing connection where each line contains a set of information about each HTTP request [12]. Our detection system automatically detects SQL injection attacks regardless of the used application by tracking the attack in the Apache log file. Our system adopts the signature-based approach for the detection of attacks.

Detection mechanism The detection system uses a signature-based model to detect SQL injection attacks. The detection mechanism requires prior knowledge of SQL injection attack techniques that hackers use to create or deduce typical scenarios. It relies on a signature database or the rules of attacks to generate the alerts. In this detection scheme, the malicious parameters obtained from the log analysis are compared to the signatures of SQL injections. If the input (malicious parameters) corresponds to one of the predefined signatures, all the information about it are stored in the database and the network administrator is alerted of a possible attempt. In addition, the signatures must be chosen carefully because we want to minimize the number of returned false positives. To be able to describe our different signatures, we used regular expressions that are an ideal tool for filtering data. They allow analyzing the different strings, and ensure that this string matches an attack signature [13].

Detection system workflow The overall workflow of the detection system, as shown in Fig. 3 is composed of the following steps:

- Support standard log format (common, combined, etc.).
- Analyze the web log to detect SQL injection using regular expressions.
- Detect SQL injection web attacks according to the signatures-based.
- Enable notification: the remote administrator is alerted of a possible attempt SQL injection attack.
- The detected attacks are recorded in the attack table.
- Add a new SQL injection signatures in signature table.

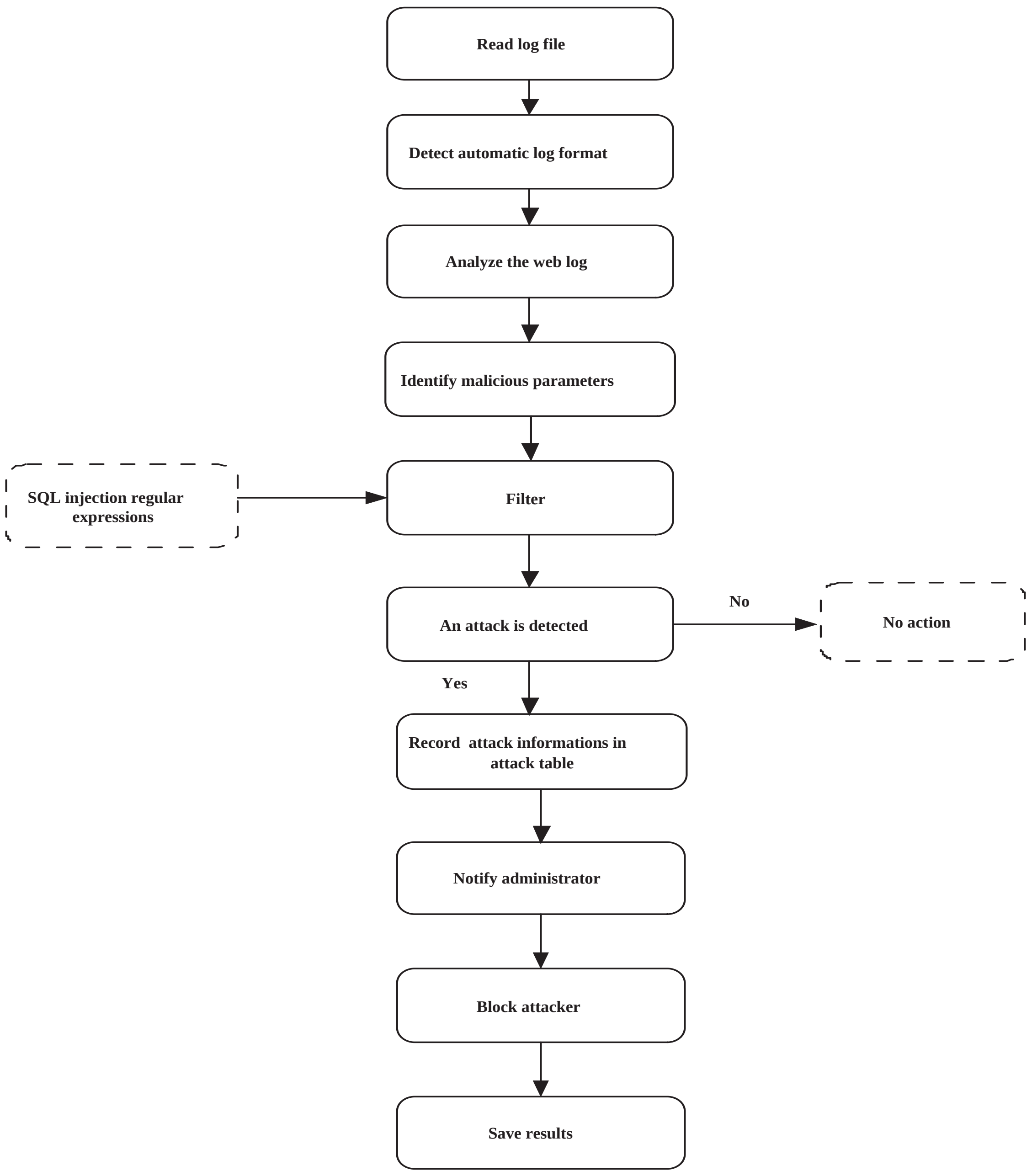
Read log file
Detect automatic log format
Analyze the web log
Identify malicious parameters
SQL injection regular expressions
Filter
An attack is detected
No
No action
Yes
Record attack informations in attack table
Notify administrator
Block attacker
Save results

Fig. 3. Detection System workflow.

3.3 Response system

The administrator is notified about the attacks. Once an attack is discovered, the administrator will be alerted by message, containing all the information about the attack. Then, he/she identifies the web compoment to be updated, and patchs all the web application in the network. Additionnely, this patching operation is performed when an attack or vulnerability is found at the web application of the DMS.

4 Evaluation

A smart meter is characterized by limited resources in terms of computational and memory capacities, and it has also very limited storage capacity (8 KB RAM and 120 KB Flash memory) [14]. Our work aims to satisfy the storages requirements. So the stored data structures should not exceed the storage capacity of the smart meter. In this paper, we adopt the web enabling smart meter technology, which implements the core feature of the web architecture, as described in [15][16]. In addition, the smart meter stores the following data structures:

– Tiny table of signatures.
– Tiny code the detection system.
– Limited size of log file with FIFO replacement policy.

To be fully operational, the detection system has to pass through the testing and validation phase. This step allows to test detection tool and validate samples of SQL injection attacks. The objective of this phase is the evaluation system performance.
The system is tested by submitting a variety of requests of several types of SQL injection attacks. The attack scenarios consist of requests sent to a website in order to make an unlawful action. Table 1 shows the detection results of our IDS against different types of SQL injection attack.

Type attack	Pattern	Detection
Tautology	Or $a \neq b$ - -	✓
Tautology	Or $a \neq b$ - -	✓
Logically Incorrect	'	✓
Union	union delete from users	✓
Union	UNION select * from users	✓
PiggyBacked	10; DROP TABLE users - -	✓
PiggyBacked	; shutdown ;- -	✓

Table 1. Presentation of Results according to SQLI Attack Type.

As shown the Table 1 the system is able to detect all types of SQL injection attacks, and the system has blocked the attacks successfully.
To test the efficiency of the detection system, we used a testing scenario that

consists of 100 submissions of which 55% of web SQL injections are attacks. The obtained results are shown in Table 2.

Detection	normal	Attack
Normal	55	0
Attack	5	40
TPR	88,88%	
FPR	0%	

Table 2. Test Results.

$$TPR = \frac{TP}{TP+FN} \qquad FPR = \frac{FP}{FP+TN}$$

The test results do not raise any warning for the 55 normal queries and raise 40 alerts out of 45 illegitimate requests. As the most important quality in a detection system is to ensure a very good true positif rate TPR with a false positif rate FPR as low as possible. So the obtained results are promising in the sense that our system correctly detected SQL injection attacks (TPR = 88,88%), while avoiding false alarms (FPR = 0%).

5 Conclusion

In this paper, we have proposed a collaborative detection and response framework against SQL injection attacks in IoT-based smart grids. The framework is composed of a set of host-based detection systems; each of which is deployed at a smart meter as well as a centralized data management server: When an attack at one host is detected, the network administrator is notified and remotely patches the other hosts. The detection engine is lightweight as each host-based analyzes the log file associated with its traffic. Experiment results are promising in terms of detection accuracy. As futur work, we plan to enhance the proposed detector engine by a behavioral-based component, in order to improve detection accuracy.

References

1. Chakib Bekara: Security Issues and Challenges for the IoT-based Smart Grid. International Workshop on Communicating Objects and Machine to Machine for MissionCritical Applications COMMCA(2014)
2. Flick T., Morehouse J.: Securing the Smart Grid: Next Generation Power Grid Security. Syngress, (2010)
3. The Open Web Application Security Project, OWASP TOP 10 Project, `http://www.owasp.org`, Accessed 10 March 2017 (2017)

4. Y. Kosuga, K. Kono, M. Hanaoka, M. Hishiyama, Y.Takahama : Sania: Syntactic and semantic analysis for automated testing against SQL injection. Twenty-Third Annual Computer Security Applications Conference (ACSAC). DOI: 10.1109/ACSAC.2007.20 (2007)
5. William G.J. Halfond, Jeremy.Viegas, and Alessandro.Orso: Classification of SQL Injection Attacks and Countermeasures. International Symposium on Secure Software Engineering (2006)
6. Yu-Chi Chung , Ming-Chuan Wub, Yih-Chang Chen b., Wen-Kui Chang b.: A Hot Query Bank approach to improve detection performance against SQL injection attacks. Computers & Security, Elsevier, Volume 31, Issue 2, pp. 233248 (2012)
7. Inyong Lee a., Soonki Jeong b., Sangsoo Yeoc, Jongsub Moond: A novel method for SQL injection attack detection based on removing SQL query attribute values. Mathematical and Computer Modelling, Elsevier, Volume 55, Issues 12, pp. 5868 (2012)
8. Stephen W. Boyd and Angelos D. Keromytis: SQLrand: Preventing SQL Injection Attacks. International Conference on Applied Cryptography and Network Security, ACNS 2004: Applied Cryptography and Network Security, pp. 292-302 (2004)
9. Diallo Abdoulaye Kindy and Al-Sakib Khan Pathan: A Detailed Survey on Various Aspects of SQL Injection: Vulnerabilities, Innovative Attacks, and Remedies. 15th International Symposium on Consumer Electronics (ISCE), pp. 468-471 (2011)
10. William G.J. Halfond and Alessandro Orso: Preventing SQL Injection Attacks Using AMNESIA. The 28th international conference on Software engineering, pp. 795-798. Doi: 10.1145/1134285.1134416 (2006)
11. Fadi Aloula, A. R. Al-Alia , Rami Al-Dalkya, Mamoun Al-Mardinia, Wassim El-Hajjb: Smart Grid Security: Threats. Vulnerabilities and Solutions, International Journal of Smart Grid and Clean Energy for architecture. 1(1), pp. 1-6 (2012)
12. `https://www.apache.org`, Accessed 15 December 2016 (2016)
13. `http://www.regular-expressions.info`, Accessed 25 February 2017(2017)
14. Ganesh Kumar Venayagamoorthy: A Survey of Electric Power Synchrophasor Network Cyber Security. 5th IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)(2014)
15. Andreas.Kamilaris1, Vlad.Trifa, and Dominique.Guinard: Building web-based Infrastructures for Smart meters. First Workshop on Energy Awareness and Conservation through Pervasive Applications, Helsinki, Finland (2010)
16. Fabio Clarizia, Daniele Gallo, Carmine Landi, Mario Luiso and Raffaele Rinaldi : Smart Meter Systems for Smart Grid management. IEEE International Conference on Instrumentation and Measurement Technology Conference Proceedings (I2MTC). DOI: 10.1109/I2MTC.2016.7520565 (2016)