Global Journal of Engineering and Technology Advances
eISSN: 2582-5003
Cross Ref DOI: 10.30574/gjeta
Journal homepage: https://gjeta.com/

(RESEARCH ARTICLE)

Check for updates

Intrusion detection system in a hardware in the loop scenario

Berenice Arzate-Rueda, Enrique Reyes-Archundia *, Juan C. Olivares-Rojas, Jose A. Gutiérrez-Gnecchi, Maria del C. García-Ramírez, Marco V. Chávez-Báez and Javier A. Rodríguez-Herrejón

Division of Graduate Studies and Research, National Technological of Mexico / Technological Institute of Morelia, Morelia, Mexico.

Global Journal of Engineering and Technology Advances, 2023, 17(03), 036–044

Publication history: Received on 13 November 2023; revised on 22 December 2023; accepted on 25 December 2023

Article DOI: https://doi.org/10.30574/gjeta.2023.17.3.0247

Abstract

Detecting perturbation in electrical signals is crucial to assure power quality (PQ) and protect equipment connected to an electrical network. Some of the information generated by disturbances analysis can be communicated using Internet of Things (IoT) protocols, but it is vital to verify the integrity of the data. As a result of the lack of security presented by IoT devices, the proposal of this work is based on the implementation of an algorithm based on IDS that allows identifying attacks or failures in the security of an IoT-based PQ disturbance detection system implemented in a BeagleBone Black Wireless. The Sparta and Sqlmap programs were used as tools to perform tests and detect security flaws in the BeagleBone, verifying essential operations of the IDS.

Keywords: IoT; Intrusion detection; Power quality; BeagleBone

1. Introduction

In recent years, technological advances in the energy sector and the need to supply the growing demand for electrical energy, estimated to triple in 2035 [1], have generated essential changes in distributing electrical energy. One of these changes is the integration of the Smart Grid (SG), which combines Information and Communication Technologies (ICT) for the Electrical Grid [2], allowing bidirectional communication between service companies and the end user [3].

The inclusion of Smart Electrical Networks seeks to improve energy management. For this, it may or may not use the Internet of Things (IoT) to connect smart sensors and actuators to the Internet and collect information stored in the cloud and accessible anywhere globally [4].

However, the massive amount of data generated, processed, and exchanged on the smart grid can be susceptible to security and privacy breaches [5], increasing concern and security threats such as cyber threats, for example, Denial of Service, Denial of Service (DoS) Attacks and Distributed DoS, Distributed Denial of Service (DDoS) Attacks that can compromise the availability of systems, False Data Injection, False Data Injection (FDI) attacks which can endanger the integrity of the information and, finally, Man-in-The-Middle attacks, Man in the middle or Man-in-the-middle attack (MiTM) that threaten the confidentiality of the systems [3].

All the problems above must be addressed critically to minimize and avoid possible failures or threats [6] that could affect the security of the devices or the system. As a measure to solve the previous problem, according to [7]-[9], in the field of SG, different algorithms have been tested to detect attacks on the network using Intrusion Detection System (IDS) such as *Bro*, *Snort* and *Suricata* or based on methods such as Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Gradient Boost Decision Trees (GBDT) and Convolutional Neural Networks (CNN).

* Corresponding author: Enrique Reyes-Archundia

In this scenario, there is a concern about evaluating intrusion detection in real-time by using digital devices. This work proposes an algorithm developed from studying the existing IDS, applied to a platform that detects Power disturbances in electrical signals, which generate malfunctions and reduce the useful life of equipment and devices [10].

2. The Snort IDS

Snort is an open-source IDS and Intrusion Prevention System (IPS) that allows real-time network traffic analysis and packet logging and can be used to debug network traffic; it implements an attack detection and port scanning engine that uses a set of rules that define malicious activity and finds data packets that match said activity to generate alerts to users.

It has three modes of operation: The *Sniffer* mode, which reads the packets circulating through the network and displays them in a continuous flow on the console; the *Packet Logger* mode, which logs packets to disk; and the *Network-based Intrusion Detection Systems* mode, which analyzes network traffic and looks for matches with a defined rule [11], [12].

In addition to the three modes of operation, Snort can perform network inspection in passive or online mode with Data Acquisition (DAQ) modules. Passive mode observes and detects traffic on the specified network interface but does not provide the complete traffic-blocking tool allowed in online mode.

2.1. Snort Rules

The search engine uses Rules or signatures to compare received packets and generate alerts if they show a match. In Snort, there are four categories of rules [11], which are:

- Protocol rules. They depend on the protocol being analyzed.
- Generic content rules. They define binary or ASCII mode patterns to search the packet data field.
- Malformed packet rules. They specify the characteristics of the packets and check the headers for inconsistencies or other types of anomalies.
- IP rules. They are applied directly on the IP layer and are checked for each datagram.

The rule header contains the action, protocol, source, destination networks, ports, and the direction of the traffic to which the rule applies. The actions that can be included in the rule are [12]:

- Alert. Generates an alert about a package and logs it. It is the most common action, with most of the rules already defined in Snort.
- Reject. It blocks the packet, registers it, and performs an action depending on the protocol in which the attack occurred.
- Drop. Block and register the package.
- Log. Register the package.
- Pass. Ignore the package.
- Sdrop: Blocks the packet but does not register it.

The necessary dependencies were downloaded and installed to install Snort according to the instructions given in [13].

2.2. Validation tools

To test the essential operation of the IDS, the *Sparta* and *Sqlmap* programs were used to perform tests and detect security flaws in the *BeagleBone; Snort* read a total of 4060 rules and compared them with the packets received when using the two mentioned tools.

2.2.1. Sparta interface

The Sparta is a graphical interface developed in *Python* that allows network audits to be carried out; it performs penetration tests in the scanning and numbering phases of vulnerabilities or weaknesses, for which it uses default credentials and “brute force” to explore the services detected during the scanning [14].

2.2.2. Sqlmap platform

It is an open-source tool used for testing and exploiting SQL injection flaws and taking control of database servers. The program can take passwords from databases and obtain the information stored in them, allowing access to the underlying file system and the execution of commands in the operating system [15].

3. Intrusion Detection System

Firstly, some disturbances were simulated in a Raspberry Pi 4 and sent to BeagleBone. This work does not present the algorithm to simulate disturbances in electrical signals. Instead, the data provided by Raspberry Pi 4 is used to explore the integrity of communication, and security is the concern of this work.

GPIOs 5, 48, and 31 of the BeagleBone were used and connected to GPIOs 17, 27, and 22 of a Raspberry Pi 4 to obtain the information saved in the database, as shown in Figure 1. On the Raspberry Pi, the analysis of different waveforms is carried out, including Power Quality Disturbances (PQD), their processing, and subsequent classification of the disturbances; the results obtained from the classification are sent with a program developed in Python to the BeagleBone, the connected pins share 3 bits of information depending on the disturbance detected, the values sent range from 0 to 7 for the Sine, Sag, Oscillatory, and Swell signals, Flicker, Interruption, Notch, and Harmonics, respectively. The algorithm allows data to be sent every 205 ms without the risk of losing information.

The BeagleBone executes a second code developed in Python that consecutively reads the values received on the pins, converts them from their binary value to their decimal equivalent, and obtains the sum of the three to identify which signal it corresponds to. When the program receives data from a disturbance, it records the start time and continues reading the input from the pins. When the value received changes, the end time is taken, the duration is calculated, and the information is sent to the database.

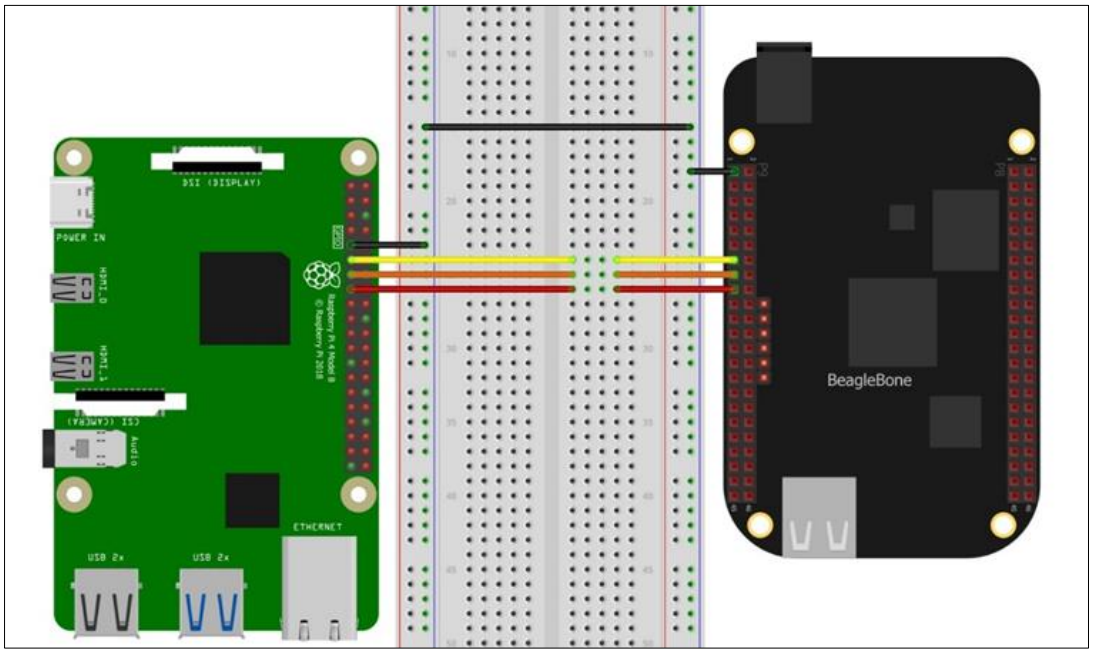
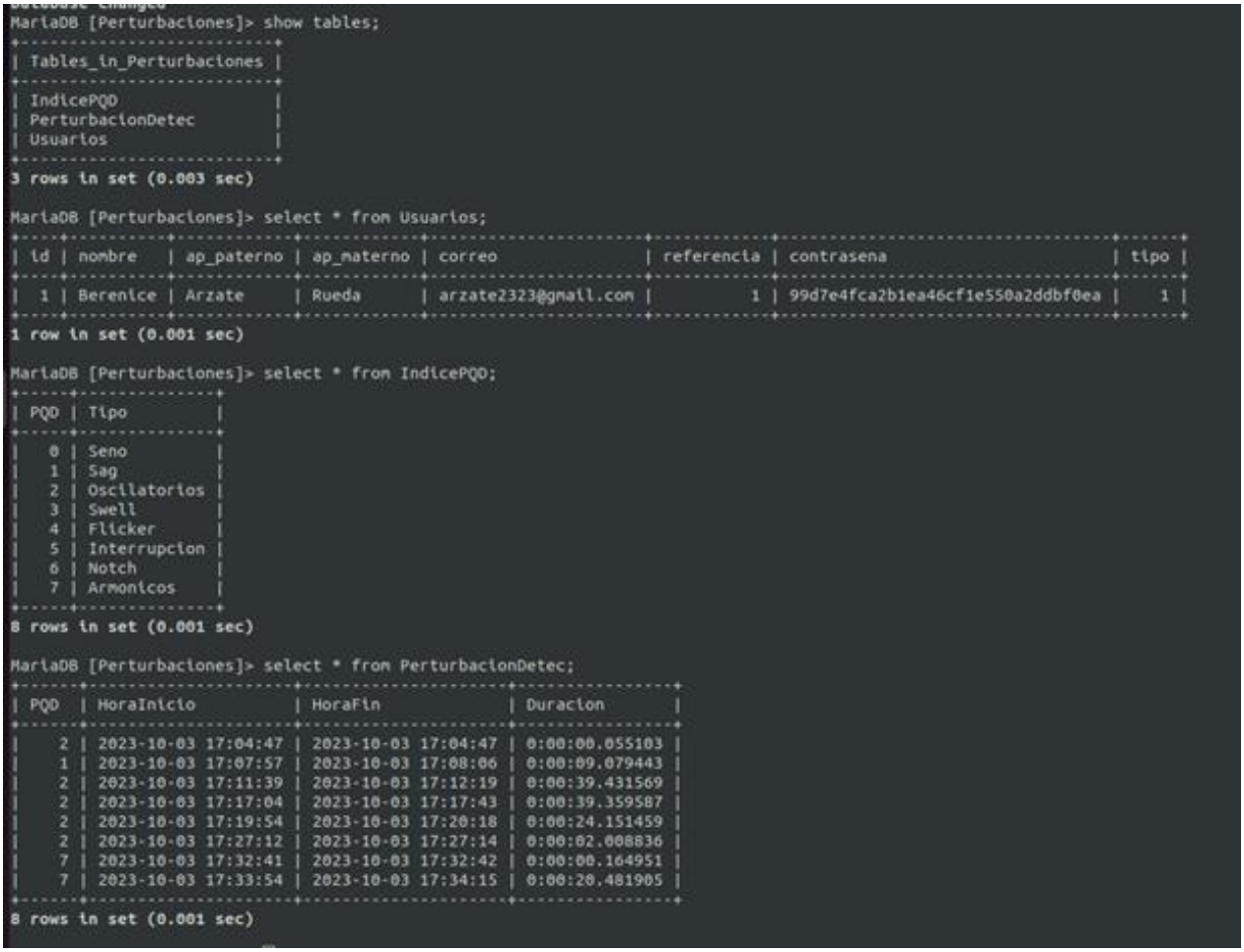


Figure 1 Raspberry Pi 4 and BeagleBone connection

The MySQL database was used to store the data, for which a MariaDB server was installed on the BeagleBone. Subsequently, the Disturbances database was created, the *IndexPQD* table was designated that relates the types of disturbances with their corresponding integer value, and the *PerturbacionDetec* table to receive and store the number corresponding to the detected disturbance, the start time, end time, and duration of the same. In addition, the *Users* table was added, which is used to access the above information from a web page. The tables that make up the database can be seen in Figure 2.



```
MariaDB [Perturbaciones]> show tables;
+-------------------------+
| Tables_in_Perturbaciones |
+-------------------------+
| IndicePQD               |
| PerturbacionDetec       |
| Usuarios                |
+-------------------------+
3 rows in set (0.003 sec)

MariaDB [Perturbaciones]> select * from Usuarios;
+----+----------+------------+------------+---------------------+------------+----------------------------------+------+
| id | nombre   | ap_paterno | ap_materno | correo              | referencia | contrasena                       | tipo |
+----+----------+------------+------------+---------------------+------------+----------------------------------+------+
|  1 | Berenice | Arzate     | Rueda      | arzate2323@gmail.com |          1 | 99d7e4fca2b1ea46cf1e550a2ddbf0ea |    1 |
+----+----------+------------+------------+---------------------+------------+----------------------------------+------+
1 row in set (0.001 sec)

MariaDB [Perturbaciones]> select * from IndicePQD;
+-----+--------------+
| PQD | Tipo         |
+-----+--------------+
|   0 | Seno         |
|   1 | Sag          |
|   2 | Oscilatorios |
|   3 | Swell        |
|   4 | Flicker      |
|   5 | Interrupcion |
|   6 | Notch        |
|   7 | Armonicos    |
+-----+--------------+
8 rows in set (0.001 sec)

MariaDB [Perturbaciones]> select * from PerturbacionDetec;
+------+---------------------+---------------------+----------------+
| PQD  | HoraInicio          | HoraFin             | Duracion       |
+------+---------------------+---------------------+----------------+
|    2 | 2023-10-03 17:04:47 | 2023-10-03 17:04:47 | 0:00:00.055103 |
|    1 | 2023-10-03 17:07:57 | 2023-10-03 17:08:06 | 0:00:09.079443 |
|    2 | 2023-10-03 17:11:39 | 2023-10-03 17:12:19 | 0:00:39.431569 |
|    2 | 2023-10-03 17:17:04 | 2023-10-03 17:17:43 | 0:00:39.359587 |
|    2 | 2023-10-03 17:19:54 | 2023-10-03 17:20:18 | 0:00:24.151459 |
|    2 | 2023-10-03 17:27:12 | 2023-10-03 17:27:14 | 0:00:02.008836 |
|    7 | 2023-10-03 17:32:41 | 2023-10-03 17:32:42 | 0:00:00.164951 |
|    7 | 2023-10-03 17:33:54 | 2023-10-03 17:34:15 | 0:00:20.481905 |
+------+---------------------+---------------------+----------------+
8 rows in set (0.001 sec)
```

Figure 2 Data base proposed

4. Results and discussion

As a result of the work, three types of tests were done. In the first, Sparta was used to scan the service. Second, an attack with SQL was simulated. At last, Snort's performance was verified in an online execution.

4.1. Scanning with Sparta

When running Sparta, a scan is performed in stages, as shown in Figure 3; for instance, the interface reports six open ports, the services running on each port, and the protocol used. It is worth mentioning that if the services are not active during the scan, the tool is responsible for activating them.

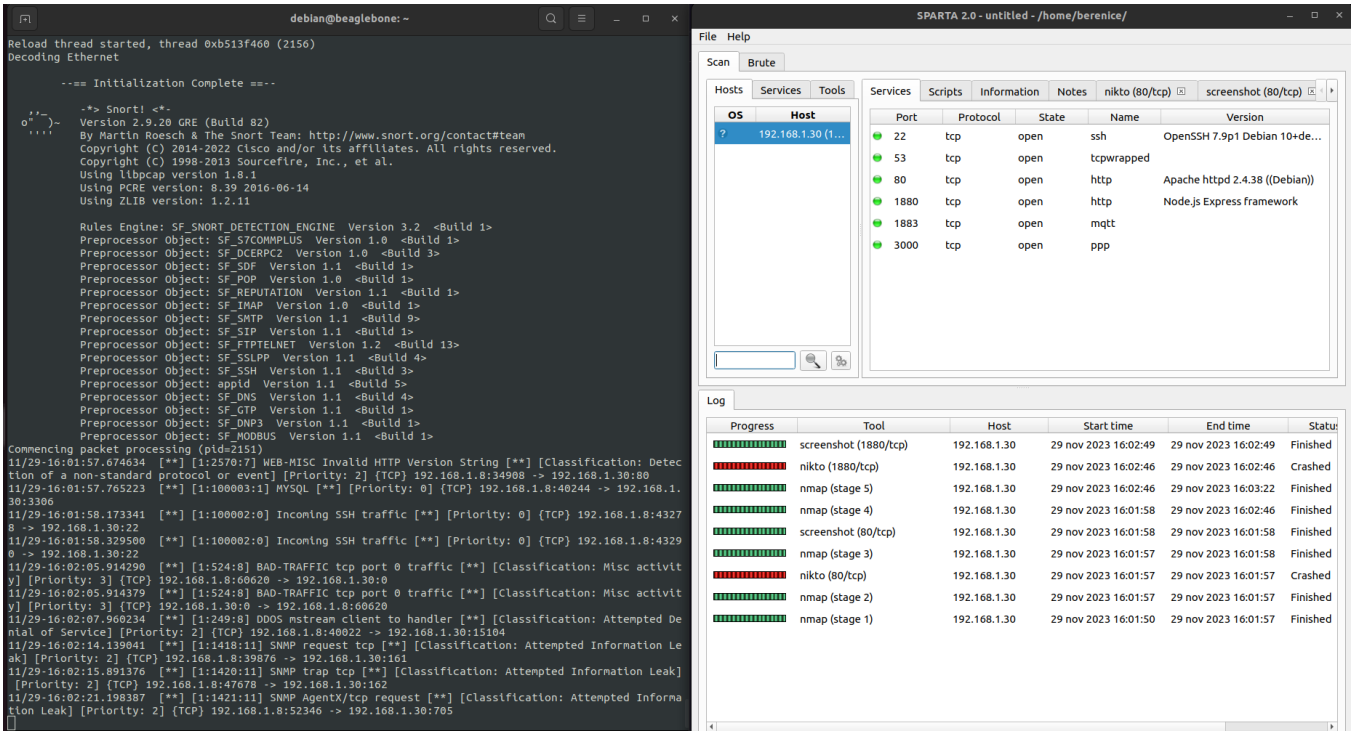


Figure 3 Alerts generated when using Sparta

Before the release of Sparta, the IDS was run with the command

sudo snort -A console -q -c /et- c/snort/snort.conf -i wlan0,

- A console activates console alerts
- q activates the mode silent so that the status report is not displayed
- c indicates the path of the configuration file
- i specifies the port to monitor.

The console shows that the device with IP 192.168.1.8 activated nine alerts for incoming packets to different ports of the address 192.168.1.30 corresponding to the BeagleBone. The first alert warns that an invalid HTTP version string has been detected; this indicates that vulnerabilities can be exploited in applications based on web servers. The alert with the highest detected priority contains the message BAD-TRAFFIC TCP port 0 traffic; this is generated when receiving TCP packets with destination port 0, which does not mean the destination port is 0. Instead, it indicates that it is carrying out a reconnaissance activity and that there is a prelude to an attack.

The Snort command is executed to obtain a complete IDS report, omitting the silent mode -q. In this way, it can be seen in Figure 4 that in the almost 3 minutes Sparta took to perform the scan, a total of 137760 packages of which 99.998% were analyzed.

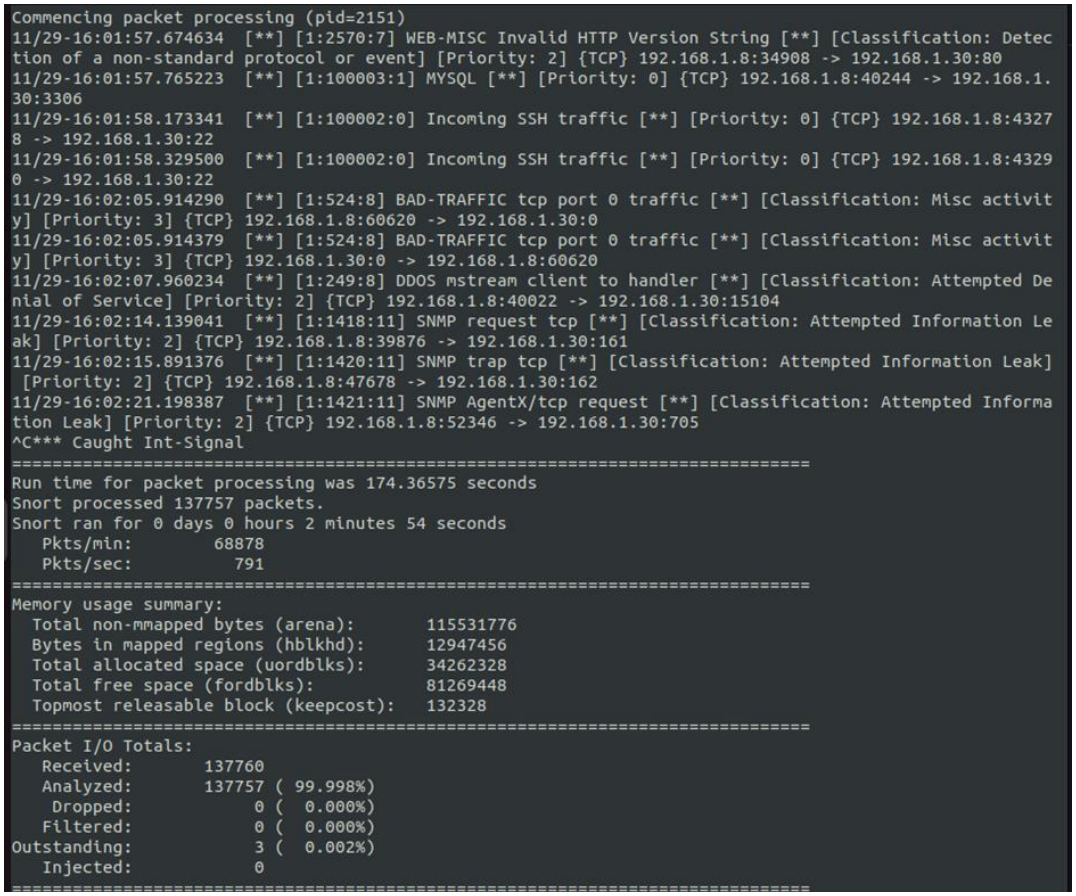


Figure 4 Report generated when using Sparta

4.2. SQL injection attack

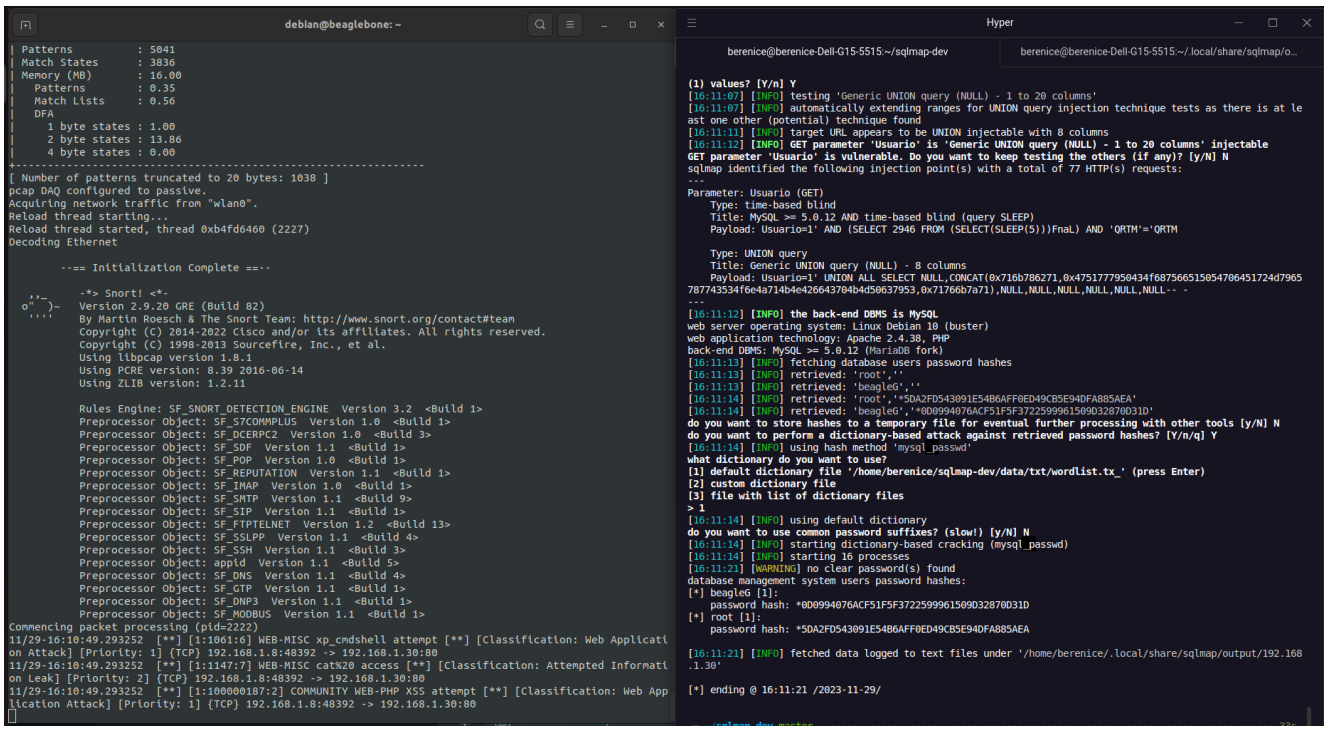


Figure 5 Running Sqlmap to obtain database password

With the *Sqlmap* tool, an SQL injection attack was launched using the web page URL, and the user identifier was sent to the database to log in. The python command *sqlmap.py -u http://192.168.1.30/PWeb/login.php\?Usuario\=1 –batch –passwords* tells the tool to obtain the passwords from the database used by the web; this results in what is shown in Figure 5.

Suppose the tool continues to obtain more information from the database. In that case, no more alarms are activated; only the number of packets received increases, of which, as shown in Figure 6, it is possible to analyze a 99.876%.

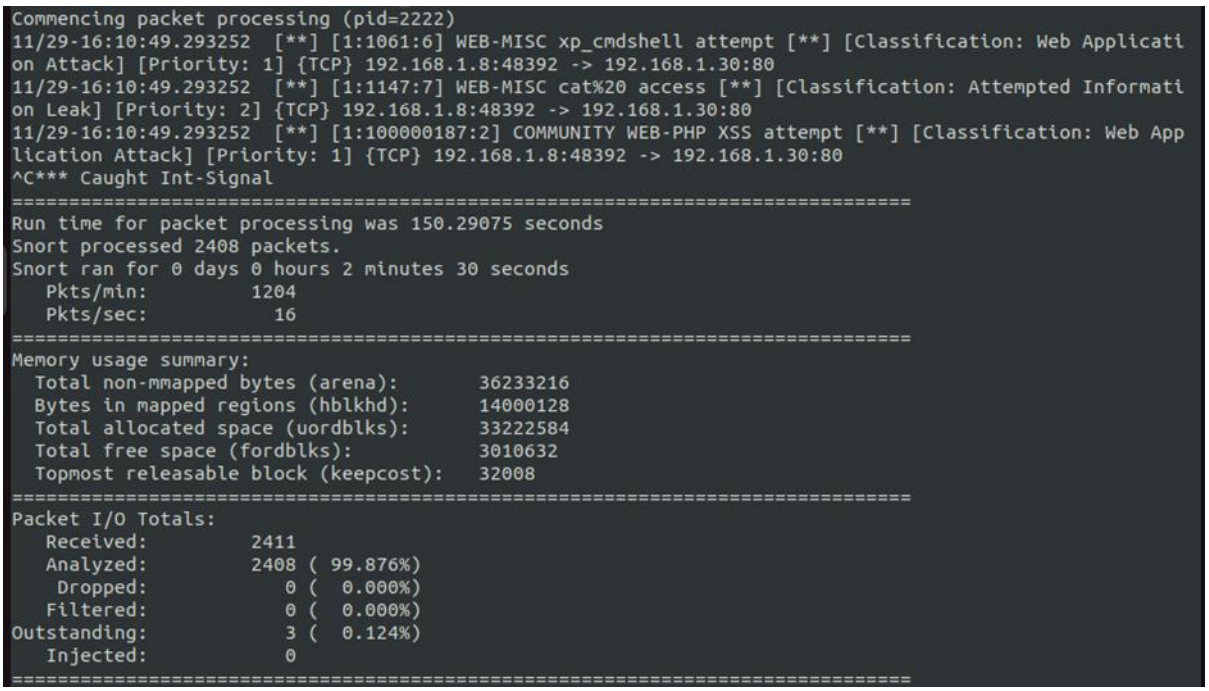


Figure 6 Report generated when using Sqlmap

Regarding the detection made by the IDS, the first alert generated indicates that Snort has detected traffic associated with SQL injection or the presence of other vulnerabilities against SQL-like servers. The second alert warns that the CGI webdist program in SGI IRIX has allowed commands to be executed through shell metacharacters. The latest alert refers to an attack directed at web servers.

In addition to the alerts in the console, a file called *snort.alert.fast* is generated that allows you to review the warnings generated after the execution of Snort. In the same directory, files with a *.log* extension are generated after each IDS release.

4.3. Online execution

Finally, rules were created with essential information to verify the performance of Snort in blocking network traffic in an attempted SQL injection attack.

One of the implemented rules uses the drop action to block and record *Transmission Control Protocol* (TCP) data packets that enter from any port and device and are directed to port 80 of the network of the BeagleBone; the rule should put the SQL drop message. The aforementioned rule was added with the structure:

drop tcp any any ->$HOME NET 80 (msg:"SQL drop"; sid:100005; rev:1;

The results obtained by activating Snort in online mode with rules for blocking packets network can be seen in Figure 7.

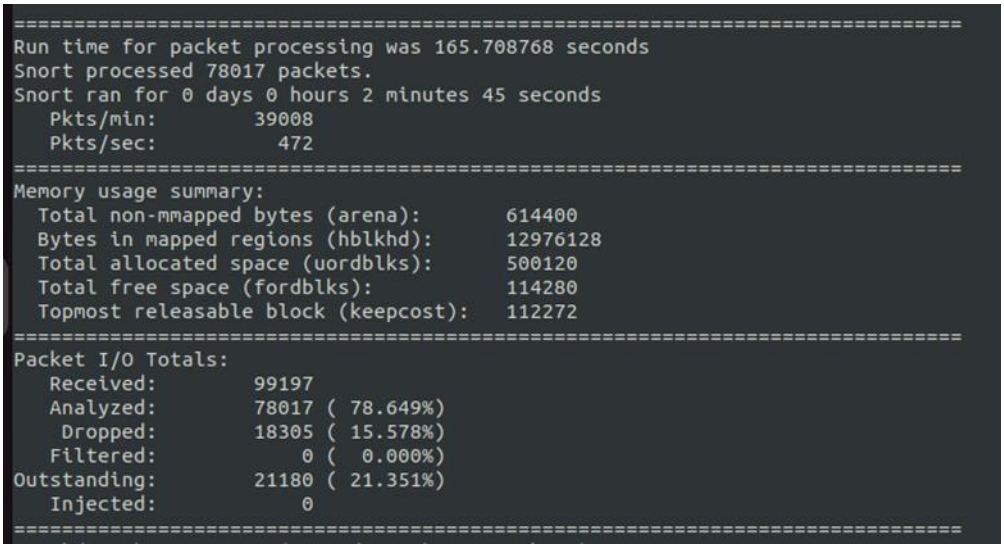


Figure 7 Running Snort online

5. Conclusion

The present work presents an implementation of intrusion detection on a BeagleBone device using the IDS protocol.

The system correctly detects intrusion attempts; however, after operating for a few hours, the BeagleBone presents operating problems and allows, among other things, to view these parameters, generating the possibility that they can be used to launch attacks of SQL injection.

Regarding the performance of the intrusion detection system installed in the BeagleBone, it presents a percentage of more than 99% of packets analyzed when it runs in passive mode and generates alerts correctly.

In online mode, 15.578% of the received packets were blocked; however, only 78.649% of the network traffic was analyzed, and in the end, the SQL injection attack obtained the database information data; the operating problems, in this case, are due to the configuration of the IDS preprocessor. Therefore, its characteristics must be thoroughly reviewed to obtain a higher percentage of packets analyzed and a successful blocking of the attacks.

Compliance with ethical standards

Acknowledgments

Tecnologico Nacional de Mexico supported this work in part under the grant 17144.23-P. CONAHCYT supported this work in part under the scholarship 832270.

Disclosure of conflict of interest

The Authors confirm that the content of this manuscript has no conflict of interest.

References

[1] G. M. Madhu, C. Vyjayanthi and C. N. Modi, Design and Development of a Novel IoT based Smart Meter for Power Quality Monitoring in Smart Grid Infrastructure, in TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON), ISSN: 2159-3450, Oct. 2019, pp. 2204-2209. doi: 10.1109/TENCON.2019.8929689.

[2] J. C. Olivares-Rojas, E. Reyes-Archundia, J. A. Gutiérrez-Gnecchi, I. Molina-Moreno, J. Cerda- Jacobo and A. Méndez-Patiño, A Comparative Assessment of Embedded Databases for Smart Metering Systems, in 2021 IEEE PES Innovative Smart Grid Technologies Conference - Latin America (ISGT Latin America), ISSN: 2643-8798, Sep. 2021, pp. 1-5. doi: 10.1109/ISGTLatinAmerica52371.2021.9543058.

[3] P. I. Radoglou-Grammatikis and P. G. Sarigiannidis, An Anomaly-Based Intrusion Detection System for the Smart Grid Based on CART Decision Tree, in 2018 Global Information Infrastructure and Networking Symposium (GIIS), ISSN: 2150-329X, Oct. 2018, pp. 1-5. doi: 10.1109/GIIS.2018.8635743.

[4] V. P. Singh, V. T. Dwarakanath, P. Haribabu, and N. S. C. Babu, IoT standardization efforts — An analysis, in 2017 International Conference on Smart Technologies for Smart Nation (SmartTechCon), Aug. 2017, pp. 1083-1088. doi: 10.1109/SmartTechCon.2017.8358536.

[5] Z. Guan, Z. Lv, X. Sun, et al., A Differentially Private Big Data Nonparametric Bayesian Clustering Algorithm in Smart Grid, in IEEE Transactions on Network Science and Engineering, vol. 7, No. 4, pp. 2631-2641, Oct. 2020, Conference Name: IEEE Transactions on Network Science and Engineering, ISSN: 2327-4697. doi: 10.1109/TNSE.2020.2985096.

[6] L. P. Bopape, B. Nleya, and P. Khumalo, A Privacy and Security Preservation Framework for D2D Communication Based Smart Grid Services, in 2020 Conference on Information Communications Technology and Society (ICTAS), Mar. de 2020, pp. 1-6. doi: 10.1109/ICTAS47918. 2020.233995.

[7] P. U. Rao, B. Sodhi, and R. Sodhi, Cyber Security Enhancement of Smart Grids Via Machine Learning - A Review, in 2020 21st National Power Systems Conference (NPSC), Dec. 2020, pp. 1-6. doi: 10.1109/NPSC49263.2020.9331859.

[8] A. Subasi, K. Al-Marwani, R. Alghamdi, et al., Intrusion Detection in Smart Grid Using Data Mining Techniques, in 2018 21st Saudi Computer Society National Computer Conference (NCC), Apr. 2018, pp. 1-6. doi: 10.1109/NCG.2018.8593124.

[9] V. K. Singh, H. Ebrahim and M. Govindarasu, Security Evaluation of Two Intrusion Detection Systems in Smart Grid SCADA Environment, in 2018 North American Power Symposium (NAPS), Sep. 2018, pp. 1-6. doi: 10.1109/NAPS.2018.8600548.

[10] B. Arzate-Rueda and J. A. Medina-Molina, Design and Implementation of a Smart Meter based on the Microcontroller NODEMCU ESP8266 and Sensor PZEM-004T, Bachelor Thesis, Technological Institute of Morelia, Morelia, Aug. 2022, 124 pp.

[11] F. Porras-Noriega, Snort setup and usage guide, Aug. 2020. url: https://issuu.com/fatimaabigailporrasnoriega/docs/gu_a_de_configuraci_n_y_uso_de_snort (retrieved on 25-11-2023).

[12] Snort - Network Intrusion Detection & Prevention System. url: https://www.snort.org/ #documents (retrieved on 25-11-2023).

[13] Installing Snort - Snort 3 Rule Writing Guide. url: https://docs.snort.org/start/ installation (11-26-2023).

[14] R. Velasco, SPARTA, an interface to simplify security audits, Redes Zone, Nov. 2015. url: https://www.redeszone.net/2015/11/28/sparta-una-interfaz- para-simplificar-las-auditorias-de-seguridad/ (retrieved on 11-26-2023)

[15] sqlmap: automatic SQL injection and database takeover tool. url: https : / / sqlmap . org/ (retrieved on 11-26-2023)