Hindawi
Security and Communication Networks
Volume 2021, Article ID 5523806, 11 pages
https://doi.org/10.1155/2021/5523806

Research Article

Poor Coding Leads to DoS Attack and Security Issues in Web Applications for Sensors

Khuda Bux Jalbani,[1] **Muhammad Yousaf**,[1] **Muhammad Shahzad Sarfraz**,[2] **Rozita Jamili Oskouei**,[3] **Akhtar Hussain**,[4] **and Zojan Memon**[5]

[1]*Riphah Institute of Systems Engineering, Riphah International University, Islamabad 44000, Pakistan*
[2]*Department of Computer Science, National University of Computer and Emerging Sciences, Islamabad, Pakistan*
[3]*Department of Computer Science and Information Technology, Islamic Azad University, Mahdishahr Branch, Mahdishahr, Iran*
[4]*Department of Information Technology, Quaid-E-Awam University of Engineering, Science and Technology, Nawabshah 67450, Pakistan*
[5]*Department of Information Technology, University of Sufism and Modern Sciences, Bhitshah 70140, Pakistan*

Correspondence should be addressed to Rozita Jamili Oskouei; rozita2020j@gmail.com

Received 21 January 2021; Revised 1 March 2021; Accepted 8 May 2021; Published 20 May 2021

Academic Editor: Shah Nazir

As the SQL injection attack is still at the top of the list at Open Web Application Security Project (OWASP) for more than one decade, this type of attack created too many types of issues for a web application, sensors, or any similar type of applications, such as leakage of user private data and organization intellectual property, or may cause Distributed Denial of Service (DDoS) attacks. This paper focused on the poor coding or invalidated input field which is a big cause of services unavailability for web applications. Secondly, it focused on the selection of program created issues for the WebSocket connections between sensors and the webserver. The number of users is growing to use web applications and mobile apps. These web applications or mobile apps are used for different purposes such as tracking vehicles, banking services, online stores for shopping, taxi booking, logistics, education, monitoring user activities, collecting data, or sending any instructions to sensors, and social websites. Web applications are easy to develop with less time and at a low cost. Due to that, business community or individual service provider's first choice is to have a website and mobile app. So everyone is trying to provide 24/7 services to its users without any downtime. But there are some critical issues of web application design and development. These problems are leading to too many security loopholes for web servers, web applications, and its user's privacy. Because of poor coding and validation of input fields, these web applications are vulnerable to SQL Injection and other security problems. Instead of using the latest third-party frameworks, language for website development, and version database server, another factor to disturb the services of a web server may be the socket programming for sensors at the production level. These sensors are installed in vehicles to track or use them for booking mobile apps.

1. Introduction

With the growing number of mobile app users, everyone is trying to develop their business apps as soon as possible. So these mobile apps are used to track the user's activities, getting information regarding vehicle locations, and tracking of logistics. For tracking vehicles, different types of mobile apps or sensors are used. For the full functionality of these apps or sensor devices, support software is required such as Personal Home Page (PHP) for the backend server, MySQL for data storage, and NodeJS for other required functionality as per requirements of clients or devices. Too many different types of open-source frameworks are used for backend functionality if the old version of these third-party tools is used. Then they may have a well-known vulnerability that can be exploited by an attacker or adversaries.

All of the above Structured Query Language (SQL) injection attacks are more dangerous for the web application or for other devices (like mobile apps or sensors) which are using it as web services. This attack is at the top of all

injection-type family attacks or web application attacks. In this attack, the weakness of input fields is exploited by the attackers. It is performed by inserting the SQL query command into the input or the query will be appended with the targeted Uniform Resource Locator (URL). These SQL queries are transformed into SQL code which is inserted by an attacker [1, 2]. This injection vulnerability is the main point of web application security exploitation by an attacker. These loopholes in web application remain because the testing of input boxes is sanitized properly [1]. If the PHP old version is used during the development and testing phase, it will also make web applications vulnerable. A web application developed on a local system with the latest version of PHP and the old version installed on the production server may lead to the unavailability of web application services for users. This may disturb other applications during the upgradation of the PHP version if there is a shared server to save the operational cost of hosting or any old version of PHP framework such as WordPress in use that can be also more dangerous for web application services [3]. With these vulnerable frameworks, the attacker can delete databases or ask for a ransom to restore those databases or encrypted code. Another issue for the performance and security of web applications is the sockets used for communication between the web server and sensors. The nature of sensors is heterogeneous; due to this, the use of the same protocol for communication is not possible [4]. These sensors are more vulnerable to be exploited due to low computing and power storage. These sensors are used for the different types of services such as, in the health system for monitoring conditions of patients [5], in vehicles to trace the location, and in water management for checking the level of water at rivers, etc. As the usage of IoT sensors is growing in every area of life, the number of attacks also increase. These attacks are performed on different layers of IoT sensors to stop services for legitimate users, or to forward fake information. This research paper has used sensors in the vehicles for tracking them. The sensors are supported by socket-based communication for sending and receiving information from clients to the server. With the use of these sensors, it creates easiness for the monitoring of taxis and getting the information regarding peak hours more business areas for taxi services. As the number of requests increases and socket connections are going to backlog, then the server starts to stop binding port of sockets with Internet Protocol (IP) Address. To fix the error of port binding with IP address, this backlog should be cleared with two methods. First, in this paper, we have to kill all backlog connections manually or the second option is to restart the service of the socket connection program. As these two methods are performed, these services of sensor connections will be down for the users. It can be said it is a self-created Denial of Service DoS attack on services [6]. Due to this type of SQL injection attack, not only specific application is disturbed, but this complete database server is crashed so that all other applications are not accessible to valid users, which is the cause of DDoS attacks on databases servers. Another issue with sockets is that systems firewalls will be applied to make a socket with the server. It will be a more critical issue with the security of a web server because everyone is allowed to make a socket connection with the server and this can be exploited by an attacker for malicious activities.

This paper is described as follows: In Section 2, the related work is described for the most threatening web application attacks, sensor-based devices usage, and security issues, and WebSocket related problems. In Section 3, we described the proposed methodology. In Section 3.1, we will define how the database server is crashed with poor coding. In Section 4.2, we mentioned a few major issues related to WebSockets by using PHP programming. Section 4 will describe the results and discussions. In Section 5, this paper will be concluded.

2. Related Work

For one and half decade, the SQL injection attack was at top of all attacks on the web application. Every attacker targets the vulnerability of SQL injection in a web application to exploit them and take control of all services related to the webserver. This attack is performed by appending or inserting malicious SQL queries with a legitimate query. The author has proposed WAVES to test the vulnerabilities of SQL injection in web applications based on the black-box method [7]. This will find out every entry point of SQL injection in web applications by using the web crawling method. After that, it will use a predefined number of methods and attack techniques on those vulnerabilities for the SQL injection attack. In the last step, WAVES will monitor the web application traffic for checking the reaction of that attack, and for more betterment of the attack method, the machine learning method is used. The researcher has proposed the method for tautology-based SQL injection queries based on the application layer-created queries and should be analyzed with a static method by combining it with an automated process [8]. But it is limited to tautology-based SQL injection and will not detect or prevent attacks related to this. The AMNESIA model has been proposed by an author and it is based on static and dynamic analysis of the queries [9]. In the static analysis process, the AMNESIA looks into the application generated legitimate queries with every connection to the database; on this condition, it will create a prototype of queries. In the second process of dynamic analysis, AMNESIA interprets all queries; after that, these will be forwarded to the database and then it will be comparing every query with a static prototype model already created. If any query is out of scope from this model, then these queries will be considered as SQL injection attacks and these queries will not be executed on a database server. But this model has more ratio of false positive or false negative if queries are encrypted by developers. The ARDILLA tool has been proposed by an author for the detection of SQL injection and Cross-Site Scripting attacks in real-time [10]. The ARDILLA is developed for the PHP scripts input testing only and sessions are not handled by this tool. The Web Application SQL injection Protector (WASP) has been proposed by the researcher, and this tool is used to detect SQL injection attacks from stored procedures with real-time configuration [11]. But this tool needs much

more improvement for the protection of web applications from SQL injection and XSS attacks.

As the demand for easy life increased due to this usage, IoT devices or sensors are also increased. Some people need to know about their business, such as tracking their goods, vehicles, Cab services and monitoring of patient's health conditions, etc. The latest version of Homecare is known as E-Homecare services which have functions of injection timings, diet management, a routine of exercise, and monitoring of health conditions [12]. "SmartPill" Wireless container is utilized to transmit intraluminal pH, pressure, and temperature information at standard interims to SmartPill GI Monitoring framework [13]. Titan implantable hemodynamic sensor (IHM) is a gadget having a size of a pencil eraser that can be embedded in the core of a patient to quantify basic factors like temperature, and afterward, remotely transmit this information to a protected database [14]. Intelligent vehicle: An arrangement of mechanical applications to gather data on the position, kinematics, and elements of the vehicle, the condition of nature, and the condition of the driver and the traveler, to survey such data and settle on choices dependent on it. It is fit for duplex correspondence with a side of the road foundation and different vehicles, to utilize computerized map applications and satellite situating frameworks, it has a functioning web association and its physical location [15]. A Smart Sustainable City (SSC) using Information and Communication Technologies (ICTs) for the creative city will give a better life, productivity of urban facilities, and competitiveness in between them. With this, current and future needs can be met as per economically, socially, and environmental changes [16]. The scholar in [17] proposed a 2-pivot MAG for distinguishing vehicle driving direction. A high discovery pace of 99% was seen when making a trip vehicles pass near the sensor. Execution corrupted to 89% as the signal-to-noise ratio (SNR) decreased. A two-edge, four-state machine calculation was presented in [18] for vehicle discovery utilizing 3-hub MAG.

The WebSocket protocol was created as a major aspect of the HTML 5 activity to encourage communications channels over TCP. WebSocket is neither a request/response nor a distribute/subscribe in the protocol. In WebSocket, a customer introduces a handshake with a server to set up a WebSocket session. The handshake itself is like HTTP, so web servers can deal with WebSocket sessions just as HTTP associations through a similar port [19]. As the WebSocket connection is established between client and server, they can send or receive data to each other with half-duplex. This connection will remain active with unlimited time and can be closed by the client or server as they want [20]. The Websocket Application Program Interface (API) provides great functionality to websites to establish a connection and transmit data to any server [21]. Due to this functionality, it is easy and effortless for a developer to work on WebSockets in websites for transmitting data. The major drawback of WebSocket that it does not add the HyperText Transfer Protocol (HTTP) header along with the connection. Due to this, the policy of origin resource verification does not provide a secure connection anymore because these origins can be spoofed [22]. As per the author, another security issue is cache poisoning with Websockets; to protect them from this vulnerability, the protocol working group introduced the method of frame-masking [21, 23]. With the addition of frame-masking, the cross-site scripting injection attack has been blocked, but the information of WebSocket cannot be transferred in plain text between client and server. The frame-masking has been used with WebSockets for the protection from cache poisoning attack, but it makes it harder the detection of malicious data via firewalls and other virus detection tools [24]. The firewalls can be bypassed by the attackers to compromise the targeted user browser and create that as a WebSocket proxy between him and the targeted organization network [25]. It is also vulnerable to another more common attack type of Denial of Service (DoS). In this case, attackers are trying to overwhelm the clients or server with bursts of information or maybe too many numbers of connections request; due to this, the legitimate users will not be able to complete their requests. In any case, on the web applications that use WebSockets, the XSS vulnerabilities open up a few new dangers. For example, with an XSS defencelessness, the attack might have the option to supersede the callback elements of a WebSocket association with custom ones [22]. This methodology permits the attacker to sniff the traffic, control the information, or actualize a man-in-the-middle attack against WebSocket associations.

When InnoDB is applied with MySQL creating too many issues related to SQL injection attacks which may lead to a complete crash of the database server. Therefore, there is a need for a solution for the prevention of this type of SQL injection attack.

3. Proposed Methodology

This research will describe the practical deployment of WebSockets for the tracking of a vehicle installed sensors in them. The complete deployment scenarios of the vehicle tracking application are defined in this section. To take care of major constraints regarding low power storage, these sensors have been implemented in idle state condition or can say it in passive mode sensors. This web application and MySQL server are deployed on Ubuntu 19.04 along with all updates of operating systems. And all other tools are also up-to-date as per deployment of this application, which is implemented about six months ago. Furthermore, in this research proposed method the latest PHP version 7.3, Laravel framework 5.4.36, MySQL 5.7, and NodeJS v13.3.0 are installed for the proposed application (see Figure 1).

The aim is to avoid the well-known vulnerabilities regarding the operating system, web application framework, database server, and version of PHP used for WebSockets. To avoid fake sensors, the authentication process has been implemented with the help of the Laravel framework web application. In this process, the drivers or vendors of vehicles need to be registered at web application along with their personal information and sensor identification number, which may be its serial number. As the constraint of sensors, these are accepting WebSockets only for communication

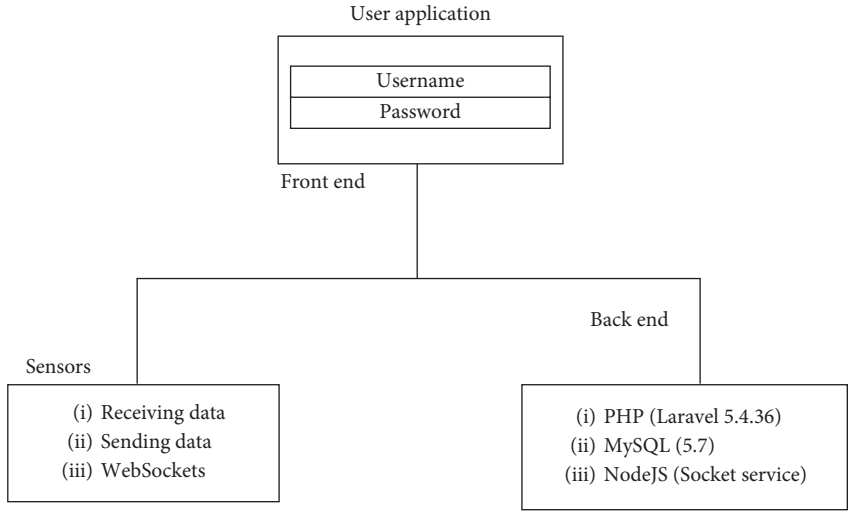

Figure 1: Vehicle tracking system overview.

instead of any API. So for this communication between vehicles and servers, the WebSocket program is developed in custom PHP, which is defined in the results and discussions section. For the security of web applications at the operating system level, iptables or Uncomplicated Firewall (UFW) has been used to block unauthorized users. For more protection at the system level, the version of the apache web server and operating system is configured as hidden in apache2.conf (see Figure 2) with red circle options.

The user's login information, sensor details, and movement of vehicles are stored in the MySQL database, which is also hosted on the same server along with a web application. For the optimization of the MySQL database, in this research, the InnoDB has been used for the stored procedure, which gives the functionality of foreign key relationships between tables. The more features and drawbacks of InnoDB and MyISAM stored procedure are explained in the next section of the crashing database server. All critical information regarding users, sensors, and vehicles is stored in a database, so the implemented security of it at the system level such as disallow remote login on a database for the root or normal users from any IP address. The default databases in MySQL have been removed, and the database users have been created with complex password authentication to avoid the brute-force attacks on MySQL databases. And for the protection from cache poisoning or man-in-the-middle attack, encryption has been implemented for the WebSocket communication between sensors and with a web application server. To compare the WebSocket issues regarding performance and backlog closing of connections with sensors, the NodeJS has been implemented for it.

3.1. Crashing Database Server. The most critical part of any web application is its databases because it is the main source of information storage regarding its users, user sessions, integrated third-party applications information, financial information, locations of users or vehicles tracking information, and much more. As per the last two-decade research

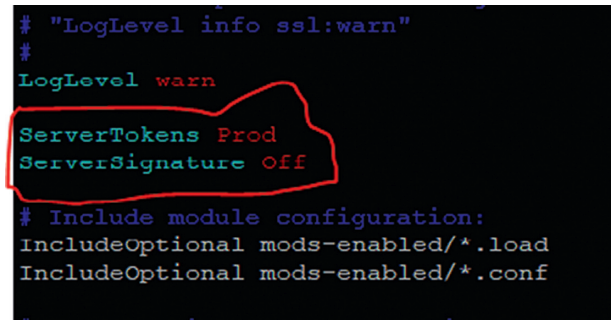

Figure 2: Hiding the operating system and Apache version.

work regarding web application attacks and OWASP top 10 attack reports of 2013 and 2016 [26, 27], the SQL injection attack is at top of all. This attack is more dangerous for web applications in the form of information stealing, DoS attack [28], system crashing, alteration in database records to insert the fake information, traffic redirection, and getting root rights of system. This attack is easy to be performed by attackers with little effort. That is why everyone is trying to exploit this injection vulnerability. The SQL injection attack is performed on web applications that have the vulnerability of weak validation on input fields such as login form. These input fields are not sanitized properly. For the better performance and optimization of MySQL database, two types of stored procedures are used, namely InnoDB and MyISAM. These methods' usage is based on the requirements of web applications. The advantages and disadvantages are explained as follows.

3.2. InnoDB Store Procedure. For the transactional database or relationships of tables, the InnoDB stored procedure is used [29]. This is used for more write operations into databases such as insert and update. This stored procedure is used for solving the issue of table-locking weakness. The InnoDB is used in applications where data integrity is in high demand for the users, and this is achieved with the help of

relationship and transaction functionality. It is used for faster write operations into databases because it supports locking the tables at row-level for better integrity. It is the most fitting stored procedure for high-simultaneousness and high-exchange remaining tasks at hand.

3.3. MyISAM Store Procedure. The default stored procedure for MySQL is MyISAM used for the high usage of reading operations. But another issue with this is that the less transactional and low level of concurrent write operations are supported. If any application needs big-size tables and fewer changes are required, then MyISAM stored procedure is used as a priority [29]. If anyone wants to use it as transactional, then they need to add an extra MySQL SQL extension of Lock Table and Unlock Tables. It is used for the high speed read and simple in implementation due to this most popular for general-purpose stored usage.

In this research paper, we have used the InnoDB stored procedure for the vehicle tracking application. In this application, the relationships between tables and more write operations are needed. The user login information, sensor information, vendors' details, and the location tracking of vehicles are stored in this database. As per the previous study of the SQL injection attack, sanitized input fields for malicious query protection used the latest version of framework and MySQL. But still, the database server has been crashed with one wrong value entry at the user login page. That wrong password value with special characters is in bold (see Figure 3).

In 2008 or early for bypassing HTTP communication, a new method for two-way communication has been developed. Maltreatment of HTTP for bidirectional correspondence prompts imperfect utilization of HTTP connections, causing superfluous issues for correspondence parties. For the solution of this issue, it has been added into working draft 10 for the HTML5 in June 2008 and that program function was named TCP connection which is based on Transmission Control Protocol (TCP) socket API [30]. The TCP connection was renamed WebSocket in late July 2008. Originally the WebSocket was created by the World Wide Web Consortium (W3C) and the WHATWG group, but it was transferred to Internet Engineering Task Force (IETF) for further development in February 2010. As the too many numbers revisions, IEFT published the final version as a WebSocket protocol with Request For Comments (RFC) 6455 in December 2011 [31]. The communication methods are used [32] given below.

3.4. Request or Response Method. It is a system where the customer sends a solicitation to the server and gets a reaction. This procedure is driven by some cooperation, for example, the snap of a catch on the website page to invigorate the entire page. At the point when Asynchronous JavaScript and XML (AJAX) entered the image, it made the website pages' dynamic through the use of JavaScript mechanization and aided in stacking some piece of the page without stacking the entire page once more. When InnoDB is applied with MySQL creating too many issues related to

{"id":3,"name":"mobile","email":"mobile@admin.com","password":"70U79Ee\/PFHu2","role_id":2,"remember_token":null,"status":1,"updated_at":"2020-01-27 08:02:43","created_at":"2019-11-04 00:00:00"}

Figure 3: Wrong value entered in password field.

SQL injection attacks, it may lead to a complete crash of the database server. Therefore, there is a need for a solution for the prevention of this type of SQL injection attack.

3.5. Polling Method. It is a system for situations where information should be invigorated without client collaboration, for example, the score of a football coordinate. In surveying, the information is brought after a set timeframe and it continues hitting the server, whether or not the information has changed or not. This makes superfluous solicitations the server, opening an association and afterward shutting it without fail. It is related to WebSockets that shows how they handle the request of users.

3.6. Long Polling Method. It is an instrument mishandling Request/Response where the association is kept open for a specific timeframe. At the point when the customer utilizes long surveying, the server reacts to the customer simply after the information is fit to be sent, which contrasts with the conventional Request/Response strategy where the reaction is sent to the customer directly after the solicitation. This is one of the approaches to accomplish constant correspondence. However, it works just with known time interims.

This research has used the PHP custom program for the WebSocket communication between vehicle sensors and web application server. This connection is used for the tracking of vehicles to get more information regarding peak hours of passengers for taxis and movement of vehicle information for their vendors. The connection between the web server and vehicle sensors has no time limit to close that connection as the few logs of the CLOSE_WAIT state are given (see Figure 4).

As in the above log entries regarding CLOSE_WAIT state or it is known as long-polling of WebSocket connection are given there are too many more connections in this state. It is causing too many problems for the webserver. The main issue of this the IP address cannot be bind with port 25001 of WebSocket for new connection requests for sensors. And sending or receiving of information from vehicle sensors is also stopped due to this issue of IP address binding. The Websocket connection creation code and temporary solution to this custom PHP program and permanent solution of this problem are discussed in the next section of results and discussion.

4. Results and Discussion

This section will discuss the issue of wrong entry from the user into SQL databases which crashed its InnoDB store procedure, how the WebSocket connections are created in the PHP custom program, and the issue of CLOSE_WAIT state of those connections for an unlimited time. The temporary solution to this problem is to apply the timer for unused opened connections to closing those WebSocket

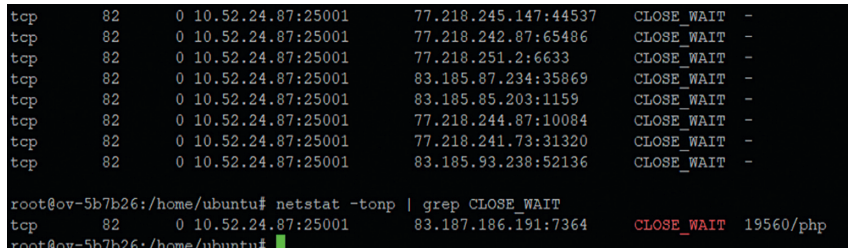

Figure 4: WebSocket CLOSE_WAIT state.

connections. A permanent solution to this problem is the usage of NodeJS-based application for the unlimited WebSocket connections without any overhead on the server. As per best knowledge, this research has used the latest software and tools for this web application of the vehicle's tracking system to avoid known vulnerabilities of SQL injection, PHP frameworks, Apache web server, and any other vulnerability related to the operating system. First, we will discuss how the MySQL database server has been crashed with a single entry at the login page.

4.1. InnoDB Crashed. The InnoDB store procedure is used for the transactional operations and relationships between tables. It is used in web applications on which write operations are performed more frequently with the support of table lock for the integrity of data. But database server has been crashed with a single wrong entry into the database at the login page as described in Figure 3. Due to that wrong entry, the InnoDB store procedure has been crashed. The MySQL InnoDB crash is shown in Figure 5.

As in Figure 5, it is crashed due to the wrong value that has been inserted into databases. It was the main reason behind the crash of it. The value of key_buffer_size is inserted in large size from its normal value. Because of this reason, the InnoDB has been crashed. Figure 6 is given for more details regarding the database crash.

To recover from this issue, we have applied two methods: First, this research proposed changing the store procedure from InnoDB to MyISAM, and secondly, changing the value of my.cnf file to recovery mode for InnoDB. As the database structure changed from InnoDB to MyISAM, all relationships between tables have been deleted with this operation and transactional operations for insertion are also disturbed. This process has taken down web service for the sensors to track the location of vehicles and it is a shared hosting server. Due to this, other web application are also unavailable for the users. This is known as a self-DoS attack on organization's web services to clients. As in Figure 7, the recovery mode entry in my.cnf file has been added for a temporary solution to crashed InnoDB.

The InnoDB has been set as in recovery mode to fix the crashing issue of the store procedure. But due to these changes, the insertion, updating, and deletion operations have been locked into databases. All databases with the InnoDB store procedure have been disturbed due to these changes. The users of those web applications are unable to write data into a database. To recover from this issue, the MySQL database server has been reinstalled after getting a backup of all databases on this server. As experienced, the single wrong value has been entered by a user at the login page that has created this problem. To protect the database from this issue again, we have applied a limit of value on that field of the login page. And go through again for validation of each input filed of web application against any malicious record entry.

4.2. WebSocket Long Polling Issue. As in this research, as earlier mentioned in section V regarding states of WebSockets for communication between client and server. The first two states of request/response and poling are working fine in the PHP custom program for connection between vehicle sensors and web application for tracking. The code section for WebSocket connection creation between sensors and web applications (see Figure 8).

If a WebSocket connection is already created with the binding of the same port with IP address, then it shows us the message of WebSocket cannot be created. The connection creation time has been set to 300 seconds in decremented order. As the connection is successful, the host IP address and port will be a bind. Communication will be started between sensors and web servers for the storage of information regarding the tracking of vehicles or insurance details, etc.

We have faced issues at the long-polling connection. They are going into the backlog for an unlimited period. As in this research shown in Figure 3 in Section 6, there are too many connections in the state of CLOSE_WAIT and due to this, the new connection cannot be created and old connections are unable to send or receive required data. This problem occurs as more than 10 users are trying to connect with the webserver at the same time. This is not good for the production servers as in real-time, there are 0.3 million users who will have to use this service. For sharing their location information, insurance details, and other required details for the security of users. The issue of IP address binding with port occurred as the number of users is increasing as in Figure 9. We have just given a single error message of IP address bind, but there are too many numbers of the same type of error messages regarding binding.

To avoid this, IP address binding with the port of WebSocket has been applied a temporary solution. In this research, we have created a service for WebSocket connections (see Figure 10).

This service has been added into crontab job scheduling and this service has been restarted after every two hours to kill the backlog of WebSocket connection (see Figure 11).

By doing this, the service of WebSocket connection to users is unavailable because there is a need to kill all the

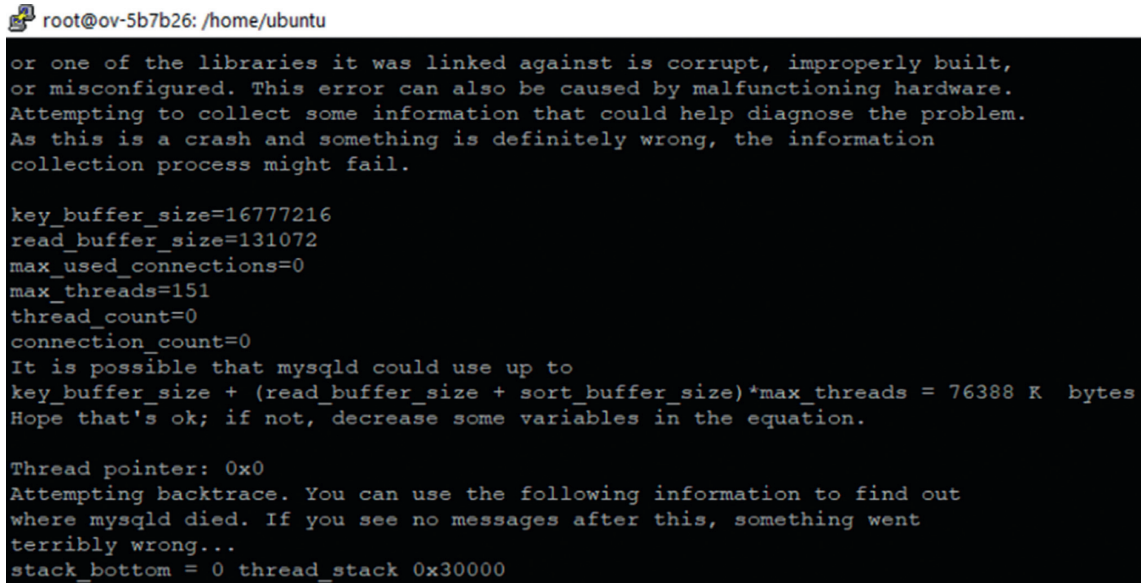

Figure 5: InnoDB crashed with wrong entry.

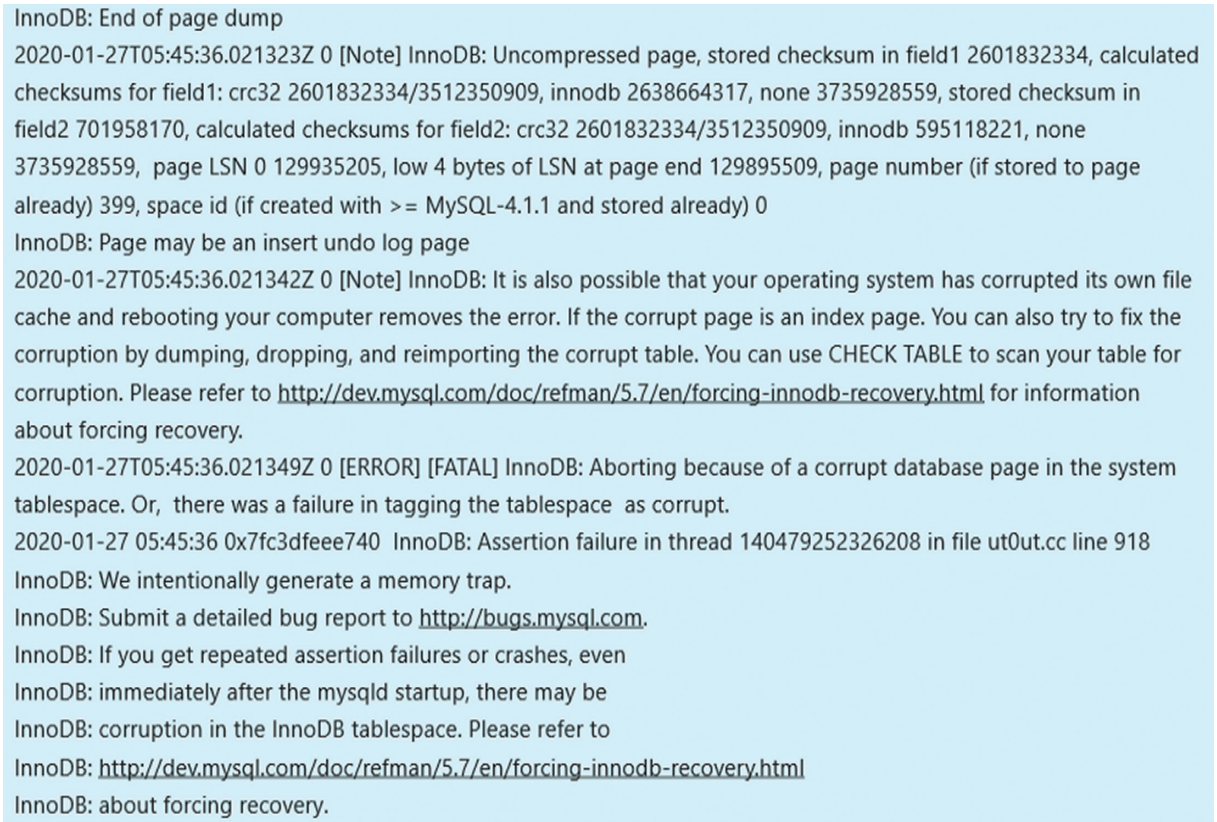

Figure 6: InnoDB crashed detailed logs.

processes related to these connections for sensors. This is not good for production servers or applications that the service of Websocket will be restarted after every few hours and critical for the real-time application, it is not considered as good practice in the case of the vehicle tracking system.

4.3. Permanent Solution for WebSocket Connections. To solve this issue, we have implemented WebSocket connections between sensors and webserver by using NodeJS 13.3.0 version. This research has faced the issue of the backlog in the state CLOSE_WAIT for too many number connections

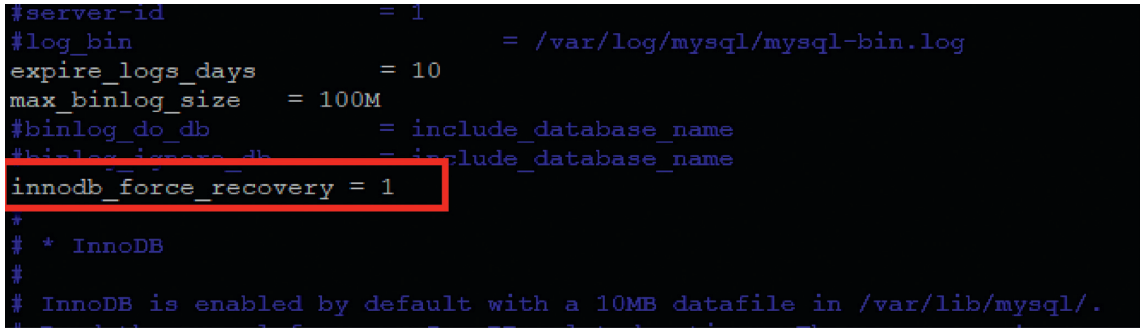

Figure 7: InnoDB recovery mode settings.

```
//$socket = socket_create (AF_INET, SOCK_STREAM, SOL_TCP) or die ("Could not
create socket\n");
    if (! ($socket = socket_create (AF_INET, SOCK_STREAM, 0))) {
        $errorcode = socket_last_error ();
        $errormsg = socket_strerror ($errorcode);

        die ("Couldn't create socket: [$errorcode] $errormsg");v
    }
    $timeout = array ('sec'=>3000, 'usec'=>0);
    $try = socket_set_option ($socket, SOL_SOCKET, SO_RCVTIMEO, $timeout);
    // bind socket to port
    $result = socket_bind ($socket, $host, $port) or die ("Could not bind to socket\n");
    // start listening for connections
    //$result = socket_listen ($socket, SOMAXCONN) or die ("Could not set up socket listener\n");
    $result = socket_listen ($socket, 4096) or die ("Could not set up socket listener\n");
```

Figure 8: Code in PHP for WebSocket connections.

```
[Mon Dec 16 14:34:09.096356 2019] [php7:warn] [pid 31393] [client
58.65.132.206:16043] PHP Warning: socket_bind(): unable to bind address [98]:
Address already in use in /home/benin/public_html/serverUpEncyption.php
```

Figure 9: WebSocket binding error.

due to which new connection request is not completed. And old connections are unable to send or receive data for tracking the vehicles. The connection code in NodeJS is as shown in Figure 12.

In the above connection, the function has been created for the WebSocket for the sensors installed in vehicles. With the help of NodeJS, the autotest of more than 100K requests for the WebSocket connection has been created without any issue of backlog or error of IP address binding with the port (see Figure 13).

And in production currently, almost 5K requests are handled by the server without any error of binding port with IP address. The long-polling connections are not opened for the unlimited time between sensors and webserver. As the data regarding vehicle location tracking, its insurance details, and vendor details are shared or inserted into the database, the connection will be closed.

The confusion matrix has been given in Table 1 for the proposed methodology in this research paper. The second name of the confusion matrix is the error matrix which is used for the quantity-based analysis of static data. The proposed system to change from the structure of the MySQL database from InnoDB to MyISAM is best against the attack mentioned in the above sections. The overall accuracy of this proposed method is 96.154%.

```
[Unit]
Description=Benin Encrypt service
After=network.target
StartLimitIntervalSec=0
[Service]
Type=simple
Restart=always
RestartSec=1
User=ubuntu
ExecStart=/usr/bin/env php /home/exam/public_html/serverUpEncryption.php

[Install]
WantedBy=multi-user.target
~
```

Figure 10: WebSocket connection service.

```
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
0 */12 * * * systemctl restart benin
5 */12 * * * systemctl restart beninen
~
```

Figure 11: WebSocket connection service restart.

```
var server = net.createServer (function (connection) {
    console.log ('client connected');
    connection.on ('data', function (data) { /*Logic Here */
                var det = data.toString (); // Decrypting Data
                det = decrypt (det);
                var sql = "INSERT INTO checkrequest (entriessent, timestamp) VALUES (?, ?)";
                var todayDate = getTodayDate ();
                con.query (sql, [det, todayDate]);
            console.log ('client data received: '+data);
                console.log ("Decrypted by Server: "+det);
                if (det == null){
                        var responseToBeSent = "#####$1;1, 1, 1, 1, 86400;";
                        responseToBeSent = encrypt (responseToBeSent);
                        console.log ("Response Sent To Client: "+responseToBeSent);
                        connection.write (responseToBeSent);
                        console.log ("Response Sent");
        connection.destroy ();
                }
                else { /*data explode */
                    var decodedVal = det.split (", ");
                        if (decodedVal.length < 8 || decodedVal.length>8){
                                var responseToBeSent = "#####$1;1, 1, 1, 1, 86400;";
                                responseToBeSent = encrypt (responseToBeSent);
                                console.log ("Response Sent To Client: "+responseToBeSent);
                                connection.write (responseToBeSent);
                                console.log ("Response Sent");
                                connection.destroy ();
                        }
}
```

Figure 12: WebSocket connection in NodeJS.

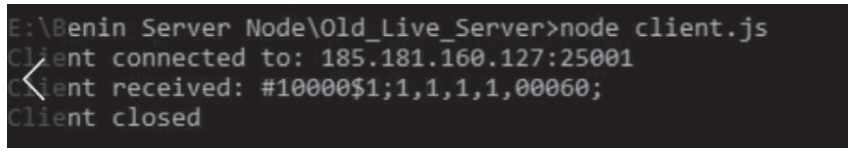

Figure 13: Autotest of WebSocket connections.

Table 1: Confusion matrix for the proposed method.

	Normal traffic	SQL injection	DDoS attacks	Classification overall	Precision
Normal traffic	600	0	0	600	100%
SQL injection	0	800	30	830	96.386%
DDoS attacks	0	50	600	650	92.308%
Truth overall	600	850	630	2080	
User accuracy	100%	94.118%	95.238%		

Overall accuracy = 96.154%.

5. Conclusions

For ease of business and to facilitate the customer's everyone wants his existence on web applications and mobile apps. As the trend of monitoring increased for security reasons and to get more data for traffic jams, peak hours for customers are hiring taxis, delivery services, health services, or online education, etc. Due to this, the usage of IoT devices is also increased. With the use of these devices, some existing security and some new issues have been arising such as for communication, WebSockets has been introduced in back 2008 and existing type of attacks is SQL injection. As have experienced just the latest tools, frameworks or operating system is not a solution to security breaches for a web application or sensors devices. But there is another factor with service unavailability, which is poor coding and selection of poor programming software solution of WebSocket connections between sensors and the webserver. As the high-security risk to the web applications is an SQL injection attack, it is performed on web applications that are not sanitized properly for input fields as we have seen that just a single wrong value entered at the login page crashed the MySQL InnoDB store procedure. Due to this, the services vehicles tracking to clients remain unavailable for a long time. To come online again temporarily, the stored procedure has been changed from InnoDB to MyISAM. But with this change, the performance of transactional operation decreased and relationships between tables also deleted. There is a need to change the mode MySQL for InnoDB into recovery mode. Due to this, other hosted websites also go down. For the protection from the injection type of attacks on web applications, the input fields need to be sanitized so that malicious users should be unable to insert malicious scripts into targeted web applications. Secondly, the WebSocket connection program was written in the PHP custom program, which has created another issue of the binding IP address with ports. The new connections of WebSocket are not created and old connections are also unable to send or receive data. For a temporary solution, we have implemented WebSocket connection service restart in CronJob of Ubuntu. And this was not a good solution for the production server. So, this research has changed the program for WebSocket connections which is NodeJS as a permanent solution to this issue. Now webserver can handle 100K+ requests without any problem of IP address binding with port numbers.

Data Availability

Data used to support this study are available from the corresponding author upon request via email (bux.khuda@gmail.com).

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] W. G. Halfond, J. Viegas, and A. Orso, "A classification of SQL-injection attacks and countermeasures," in *Proceedings of the IEEE International Symposium on Secure Software Engineering*, vol. 1, pp. 13–15, Washington, DC, USA, March 2006.

[2] A. Tajpour, M. Z. Heydari, M. Masrom, and S. Ibrahim, "SQL injection detection and prevention tools assessment," in *Proceedings of the 2010 3rd ICCOINS International Conference*

on Computer Science and Information Technology IEEE, vol. 9, pp. 518–522, Chengdu, China, July 2010.

[3] J. Ruohonen, "A demand-side viewpoint to software vulnerabilities in WordPress plugins," in *Proceedings of the Evaluation and Assessment on Software Engineering*, pp. 222–228, Copenhagen, Denmark, April 2019.

[4] C. Tian, X. Chen, D. Guo, J. Sun, L. Liu, and J. Hong, "Analysis and design of security in the internet of things," in *Proceedings of the 8th International Conference on Biomedical Engineering and Informatics (BMEI)*, pp. 678–684, IEEE, Shenyang, China, October 2015.

[5] M. Talal, A. A. Zaidan, B. B. Zaidan et al., "Smart home-based IoT for real-time and secure remote health monitoring of triage and priority system using body sensors: multi-driven systematic review," *Journal of Medical Systems*, vol. 43, no. 3, p. 42, 2019.

[6] D. Surbhi and K. Deepak, "Analysis of tree-based classifiers for web attack detection," in *Advances in Signal and Data Processing*, pp. 421–428, Springer, Singapore, Asia, 2021.

[7] Y. W. Huang, S. K. Huang, T. P. Lin, and C. H. Tsai, "Web application security assessment by fault injection and behavior monitoring," in *Proceedings of the 12th International Conference on World Wide Web*, pp. 148–159, Budapest, Hungary, May 2003.

[8] G. Wassermann and Z. Su, "An analysis framework for security in web applications," in *Proceedings of the FSE Workshop on Specification and Verification of Component-Based Systems (SAVCBS)*, pp. 70–78, Newport Beach, CA, USA, October 2004.

[9] W. G. Halfond and A. Orso, "AMNESIA: analysis and monitoring for neutralizing SQL-injection attacks," in *Proceedings of the 20th International Conference on Automated Software Engineering IEEE/ACM*, pp. 174–183, Long Beach, CA, USA, November 2005.

[10] A. Kieyzun, P. J. Guo, K. Jayaraman, and M. D. Ernst, "Automatic creation of SQL injection and cross-site scripting attacks," in *Proceedings of the 31st International Conference on Software Engineering IEEE*, pp. 199–209, Vancouver, Canada, May 2009.

[11] N. S. Ali and A. S. Shibghatullah, "Protection web applications using the real-time technique to detect structured query language injection attacks," *International Journal of Computer Applications*, vol. 149, no. 6, pp. 26–32, 2016.

[12] G. Demiris and J. Tan, *Rejuvenating Home Health Care and The-Home Care. E-Health Care Information Systems: an Introduction for Students and Professionals*, Jossey-Bass, San Francisco, CA, USA, 2005.

[13] S. Maqbool, H. P. Parkman, and F. K. Friedenberg, "Wireless capsule motility: comparison of the smartPill GI monitoring system with scintigraphy for measuring whole gut transit," *Digestive Diseases and Sciences*, vol. 54, no. 10, pp. 2167–2174, 2009.

[14] R. L. Benza, A. Raina, W. T. Abraham et al., "Pulmonary hypertension related to left heart disease: insight from a wireless implantable hemodynamic monitor," *The Journal of Heart and Lung Transplantation*, vol. 34, no. 3, pp. 329–337, 2015.

[15] F. Duchoň, P. Hubinský, J. Hanzel, A. Babinec, and M. Tölgyessy, "Intelligent vehicles as robotic applications," *Procedia Engineering*, vol. 48, pp. 105–114, 2012.

[16] S. N. Kondepudi, V. Ramanarayanan, A. Jain et al., *Smart Sustainable Cities: an Analysis of Definitions; the ITU-T Focus Group for Smart Sustainable Cities*, International Telecommunication Union (ITU), Geneva, Switzerland, 2012.

[17] N. Wahlström, R. Hostettler, F. Gustafsson, and W. Birk, "Classification of driving direction in traffic surveillance using magnetometers," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1405–1418, 2014.

[18] K. Liu, H. Xiong, and H. He, "A new method for detecting traffic information based on anisotropic magnetoresistive technology," in *Proceedings of the Fifth International Conference on Machine Vision (ICMV 2012)*, Wuhan, China, March 2013.

[19] V. Karagiannis, P. Chatzimisios, F. Vazquez-Gallego, and J. Alonso-Zarate, "A survey on application layer protocols for the internet of things," *Transaction on IoT and Cloud Computing*, vol. 3, no. 1, pp. 11–17, 2015.

[20] A. Wessels, M. Purvis, J. Jackson, and S. Rahman, "Remote data visualization through WebSockets," in *Proceedings of the Eighth International Conference on Information Technology: New Generations IEEE*, pp. 1050-1051, Las Vegas, NV, USA, April 2011.

[21] L. S. Huang, E. Y. Chen, A. Barth, E. Rescorla, and C. Jackson, "Talking to yourself for fun and profit," in *Proceedings of the W2SP*, pp. 1–11, Oakland, CA, USA, May 2011.

[22] J. P. Erkkilä, *Websocket Security Analysis*, pp. 2-3, Aalto University School of Science, 2012.

[23] L. Fette and A. Melnikov, *RFC 6455, "The WebSocket Protocol,"* 2011.

[24] M. Shema, S. Shekyan, and V. Toukharian, *Hacking with WebSockets*, BlackHat USA, London, UK, 2012.

[25] S. Shah, *Html5 Top 10 Threats Stealth Attacks and Silent Exploits*, BlackHat Europe, London, UK, 2012.

[26] "What is OWASP? what are the OWASP top 10?" 2020, https://www.owasp.org/index.php/Top_10_2013-Top_10.

[27] "What is OWASP? what are the OWASP top 10?" 2020, https://www.cloudflare.com/learning/security/threats/owasp-top-10/.

[28] A. Bhardwaj, V. Mangat, R. Vig, S. Halder, and M. Conti, "Distributed denial of service attacks in cloud: state-of-the-art of scientific and commercial solutions," *Computer Science Review*, vol. 39, Article ID 100332, 2021.

[29] A. Tomic, "NoSQL: a relational database using NoSQL technology," Master's thesis, USI, Valhalla, NY, USA, 2011.

[30] "W3C. 2008. HTML5-A vocabulary and associated APIs for HTML and XHTML," 2020, https://www.w3.org/TR/2008/WD-html5-20080610/single-page/.

[31] I. Hickson, *The WebSocket Protocol Draft-Hixie-the WebSocket Protocol-76*, Google. Inc, Menlo Park, CA, USA, 2010.

[32] V. Chopra, *WebSocket Essentials–Building Apps with HTML5 WebSockets*, Packt Publishing Ltd, Birmingham, UK, 2015.