Available online at www.jrors.ir

Journal Of Radar and Optical Remote Sensing

JRORS 3 (2021) 37–49

A Heuristic Model for SQL Injection Attacks Prevention in GIS Web Application

Mohammad Ali Arasteh[a], Fahimeh Parsaei[b*]

[a]*Head of GIS Group, Yazd Water and Wastewater Company, Ph.D. Department of Information Technology, University of Qom, Iran*

[b]*Regulatory Center of the Iranian National Taz Administration (INTA), Tehran, Ph.D. Candidate, Department of Cognitive Neuroscience, University of Tabriz, Iran*

Received 28 March 2021; revised 18 May 2021; Accepted 1 August 2021

Abstract

By increasing the development of Geographical Information Systems (GIS) providing electronic map data exchange with internet and mobile applications, related problems such as keeping secure map information, safe transactions, and assured broadcast services are necessary. Every year millions of attacks on financial and data information will be caused a series of problems in the world. One of the most critical attacks on the application level is SQL injection into the Web database. This paper tried to present a model for preventing SQL injection into GIS applications, which leads to fetching and manipulating the map information and data from a database. It also provides solutions for IT managers to keep the GIS website secure. The model security steps were tested on one of the GIS portals of Iranian organizations. To evaluate the performance of the proposed model, the security of an Iranian web GIS was checked before and after the announcement of the instructions, and the test results of the vulnerability checking with Acunetix and DVWA. The result showed that the website was completely safe and the model's instructions for various stakeholders, including programmers, administrators, and GIS experts can significantly prevent this attack.

***Keywords**:* SQL Injection, Web GIS Application-Level Vulnerabilities, Authentication and Authorization, Data Integrity, Application Security Scanner

1. Introduction

The Internet has changed the business and conditions of sending and receiving data, products, and business relationships. In the meantime, new problems have been created that did not exist before. Although information technology is globalizing in all societies, the development and operation, development of communication infrastructure, knowledge related to infrastructure, and the safe and legal environment are also interdependent, one of the most fundamental problems is the security of information exchange. Because of the increasing development of GIS websites and web services, the various devices and methods of attack are increasing rapidly. Anyone around the world can attack and damage a country's cyberspace. Having the ability to detect and have the necessary equipment, its

* Corresponding author Tel: +98-9132201169.
Email address: fahim.parsaei@gmail.com.

expansion and operation depend on the expansion of communication bases and databases related to knowledge and security. Therefore, security in GIS has particular importance among those who are expanding websites. This matter can be secured by rebuffing the vulnerability of the GIS web applications and facing threats to web applications precisely and correctly. Each year, every government and GIS web applications manager should spend some of their budgets to assure website security. However, the number of attack reports and the amount & of damage caused is increasing each year. One of the main security concerns is the weakness of GIS web application programmers and the clumsy expansion of web applications. SQL injection attacks are one of the primary attacks on web application databases and extracting its information. For the first time, SQL piggybacking or SQL injection attack was brought up in late 1998 (Halfond, Viegas, and Orso, 2006).

This paper tries to investigate SQL injection attacks on GIS websites, especially the portals of GIS, and it examines ways of avoiding and contrasting them. Accordingly, the details of SQL injection are explained first. The vulnerability of GIS websites is then examined. A model is provided to prevent vulnerabilities of the GIS website from SQL injection. Then, for evaluating the model the security of an Iranian organization was reviewed. Finally, conclusions and future trends are presented.

2. SQL Injection

A web application is a software system that provides its users with an interface via a web browser on any operating system. Despite the increasing popularity of web applications, the security threat in web applications has become more diverse, resulting in more severe damage. In poorly designed web applications, malware attacks, particularly SQL injection attacks, are common. This vulnerability has been known for over two decades and remains a source of concern to this day (Johny, Nordin, Lahapi, and Leau, 2021). SQL injection happened in electronic records in databases and it still existed even after two decades since it first happened. Most web-based applications are still vulnerable to SQL injection attacks. Although technology had improved a lot during these past years, hackers still can find holes to perform the SQL injection. There are many methods for this SQL injection to be performed by the hackers and there is also plenty of prevention for the SQL injection to happen. The vulnerability to SQL injection is very big and this is a huge threat to the web-based application as the hackers can easily hack their system and obtains any data and information that they wanted anytime and anywhere (Yunus et al., 2018). It is based on this fact that hackers may fill entry parts of a website with appropriate SQL characters, or they may inject some words in SQL statements and define a new structural statement to change the main SQL commands or affect their execution and get access to the database information in different ways (Hu, Zhao, and Cui, 2020). In such cases, the hacker is presented with the chance of changing and defining the information. Among the effects of SQL Injection on web Applications are defining the data in a database (Insert, Update, Delete), executing organizational efforts on a database (such as shutting down DBMS), regaining the content of files that are presented for DBMS, and in some cases commanding the operating system. Stealing emails, database credits, and users' other important information are among the destructive aims of this attack. SQL injection attacks present different techniques to penetrate websites. Each of them can be used through various mechanisms of SQL injection attacks.

2.1. Injection Mechanisms

The web application can be penetrable in different ways:

Inject to user input: SQL is a DML (Data Manipulation Language); therefore, these different sentences such as select, insert, update, delete and select union can be changed and be applied to websites in the title of users' login and can gain different information from databases (Boudraa, 2019). In many SQL attacks, user inputs come from form submissions sent to the web application by HTTP get or post request (Chen, Yan, Wu, and Zhao, 2021).

Attacks on cookies: Cookies are those files that contain a person's user account on a website and are

stored by the browser on the local computer. When the user logs in to the same website for the second time, his information can be read from the cookies and set. If a web application uses cookies content to build SQL queries, the attacker can easily manage an attack by adding some codes to the cookies (Li, Li, Wang, and Cheng, 2019).

Inject into server variables: A set of variables, including environmental variables, HTTP network, and Headers, are called server variables. Web applications use these variables in different ways, including logging usage statistics and defining the browsing process. If these variables enter the database without evaluation, SQLIA vulnerabilities are caused (Kausar, Nasar, and Moyaid, 2019). In this way, hackers can put their queries in HTTP and network headers, and thus when queries are logged to the server variables issued to the database, the attack in this header happens (Vyamajala, Mohd, and Javaid,2018).

Second-order injection: The aim of this attack is different from the aim of regular injection attacks. This mechanism is an indirect SQLIA type since the hacker's login to the system and the database may lead to an attack not just in the same minute, but it happens while this log is used later by the system (Liu and Wang, 2018). In some cases, it is possible to have some limitations (such as escape, type check, and filtering the login). Be maintained by the programmer of the web application to stop the hackers, and in this way, he makes sure about the safety of the program; But when the data is used in another meaning or when different queries are made, some of these methods can be cheated. For instance, if ' is omitted from received logins by the programmer in a way that each " "is replaced with " ' ' " ' in a query, while the hacker gets a username called admin using the previous method and enters admin' --, according to the limitation set by the programmer cannot succeed in penetrating the system. In this case, to penetrate, he can register a user as admin' -- in the registration part, and then the following query can be entered into the login page, and the password of the admin user can be changed:

SQL= "update users set password=' "+ new password" 'where username=' "+ RSO ("username"). +" "

Since the injection point is different from the point where the attack can be recognized, second-order injection attacks cannot be easily found and avoided.

2.2. The Aim of the Attack

Attacks on GIS programs can be divided into the following types based on the four main characteristics of web applications:

Attack on reliability: when the SQL database keeps important data, losing its reliability causes many problems in SQL injection vulnerability (Nasereddin, ALKhamaiseh, Qasaimeh, & Al-Qassas).

Attack on authentication: If SQL commands which are used to evaluate the usernames and passwords are not assured enough, a hacker can penetrate the system as a user without knowing its password (Das, Sharma, and Bhattacharyya, 2019).

Attack on authorization: if the authorization information is stored in the database, this information can be changed due to the appropriate use of SQL injection vulnerability (Das, Sharma, and Bhattacharyya, 2019).

Attack on data integrity: If the integrity of a database is not assured, it is possible to read and change its data using a SQL injection attack. SQL injection attack aims are mentioned based on their objectives. In each case, the processes that should be done in the attack are stated. Finding Inject able parameters: The hacker should find users' login fields and those vulnerable parameters to SQLIA to Attack (Gupta et al., 2019).

Perform database fingerprinting: A hacker used to know the version and type of the web application's database to inject his queries and be able to make his attacks based. Since each database reacts to queries differently, a hacker should know the type and version of the web application's database to inject his queries and then make his attacks based on that database. A simple way is using multi grammar (Ali, Adil, and Ebrahim, 2020).

Determine the database schema: To gain true information from a web application database, hackers

usually need to know some information about the qualities of the database, such as tables' names, columns' names, and the type of columns' data (Ali, Adil, and Ebrahim, 2020).

Extract data: Many attacks aim to find the amount and values from a database. Based on the web application type, this information can be sensitive and appropriate for the attack (Keshri et al., 2022).

Add and modify data: This attack aims to add or change the database information (Keshri et al. 2022).

Denial of service: This kind of attack is used to shut down the database or lock and drop tables of the web application database. So, they destroy users' accessibility and the possibility to service them (Bhateja, Sikka, and Malhotra, 2021).

Evade detection: These attacks are used to avoid checking up and recognizing the mechanisms of protecting the system (Tang et al. 2020).

Bypass authentication: This kind of attack aims for the database to pass the authentication mechanisms in web applications and databases and change the users' level of accessibility. These attacks cause the hacker to have complete accessibility to a particular user (Das, Sharma, and Bhattacharyya, 2019).

Execute remote commands: These kinds of attacks perform arbitrary commands, like store procedures or the functions in the database (Weinfurter et al. 2021).

Perform privilege escalation: These attacks are used to limit accessibility by abusing errors or executing special logical errors in the database. To face the attacks to pass getting an identity, these attacks focus on exploiting users' privilege of the database (Weinfurter et al. 2021).

3. GIS Web Application Vulnerable

There are lots of GIS web applications that are working for online maps. GIS web applications are active to communicate with maps, either directly or as a third party. Apart from the kind of activity done by the web application, the way it interacts with the user, and its user interface, there are some main principles for these communications. Among them, we can mention parts related to registering a user name, their entrance to the GIS web application, selecting layers, sending data, WMS, WFS, etc. These activities need a permanent connection to the database. This kind of attack aims at today's most essential databases, including SQL server, my SQL, oracle, etc. (Sabou and Maiorescu, 2020) analyses in the scientific literature the smart city challenges, focusing on smart governance and potential security issues threatening this component. Starting from these aspects, it proposes one smart governance web GIS (Geographical Information Systems) application designed for civic engagement In Bucharest, describing both the usability and security challenges it must answer to (Giribabu et al., 2018) presents the architecture of WebGIS environment, the role of networking components, and traits of Cybersecurity and portrays various defense mechanisms that aid in Cybersecurity in the WebGIS environment. (Hayslett, 2019) examine and explore the concept of a national Geographic Information System used for academic research, and national data sharing techniques. It takes a deep dive into how to mitigate threats to an organization’s data, and information technology resources.

In this part, the most important parts of the GIS web application, which are connected to the database, are mentioned, and then, in the case of existence, the main techniques of SQL injection will be mentioned. In each part, first, the mechanism of the attack will be presented, then the aim of the attack will be explained, and finally, some cases of SQL injection attacks, which act in this way, will be stated.

3.1. Registration, Entrance, Editing Information

GIS profiles have a part for the user's contacts, including the user's registration, entrance form, editing information, etc. These parts are designed for the website to produce satisfaction, contacts management, special offers, etc. The most vulnerable parts of the GIS web applications are these parts, and attacking them is among the attacks on the clients. One of the significant damages to SQL injection will be done the same way. Approximately every GIS website has a login page, and everybody can use it. However, it is one of the most vulnerable parts of the GIS web application, and mostly all techniques of SQL

injection attacks happen to it. Since this page has access to the user's accounts, the attacks on it are hazardous.

Register and edit information: Injection mechanism: injection to the user's entrance, second-order injection, by the use of this mechanism, as it was said in the previous section, through registering a user using special methods in this part, some of the limitations that the programmer has designed can be defrauded. The attack aim: recognition of injection parameters, detection of the schema of the database, addition or definition of the data, bypassing authentication, and executing remote commands. Some cases of attacks: piggybacked query, union query, functions, etc.

Enter user information: Injection mechanism: inject second-order injection to the user's entrance. The attack aim: All the mentioned in the previous section. Some cases of the attack: All the attacks can happen, especially those which use inference.

3.2. Interaction on the Websites with Web Services

In most GIS websites, there is a possibility to connect to another website with web services or SQL queries. In such cases, the website firstly should get users' account information and then contact the destination, and after passing some processes, the same amount of data will be sent. As a person's vital information is put in this part, destructive penetrators and hackers try more to get information and penetrate the website; this part is one of the sensitive and important parts that need more protection against damages. Its safety is crucial (İlker and Aydos, 2019). Since the data is given from the website database, SQL injection can attack it. The injection mechanism is changing users' entrance, second-order injection. Attack aim is recognizing injection parameters, detecting the schema of the database, adding or defining data, bypassing authentication, executing the remote command, and evading detection. Some cases of attacks are piggybacked queries and union queries: functions, Illegal/ logically Incorrect Queries, etc.

3.3. Cookies

Website cookies are often used to identify past customers, user convenience, or record other information. Cookies are good places for different kinds of attacks. By having access to the user's cookies, the hacker can have access to his information. Since cookie information is the same as having access to the person's user account information and thus leads to interaction with the database, SQL injection is possible while working with cookies (Vyamajala, Mohd. And Javaid, 2018). The injection mechanism is attacking cookies if they contain private information. Attack aim is to perform privilege escalation, bypass authentication, and get the amount of data. Some cases of the attacks are piggybacked queries, tautology, etc.

3.4. URL

It is not necessary to mention that all of the web- sites in the world need their URL to be used to make the connection between other pages. Sometimes, the database reads its information from the URL and writes on it. In fact, in this case, the query string is used. Lots of the attacks aim at URLs. Among these attacks, there are SQL injection attacks. The injection mechanism is injected into the server's variables. Attack aim is to understand the database version, fingerprint database, the database schema, recognize injection parameters, get the amount of data, performing denial of service. Some attack cases are tautology, union query, legal/ logically Incorrect Queries, piggyback queries, and using time delays (Manyumwa et al., 2020).

3.5. Feature Selection Attack

Lots of Web-GIS to comfort the user and the possibility to choose several items, arrange GIS layers

and add each item to his profile. A disabled and enabled item done in interaction with the database can be one of the aims of an attack. An injection mechanism can be done at the user's entrance. Attack aim is to add and change commands, executive remote commands, and define injectable parameters. Some cases of the attacks are union queries, functions, and store procedures (SaiSindhuTheja and Shyam, 2021).

3.6. Offer, Comment, Contact Us, Discuss Subjects, and Private Messages

On many websites, including the GIS website, users can contact website managers by putting in some comments or contacting them. There is also the possibility to comment about a particular item in many others. Sometimes the possibility is also added. Since the information is recorded in the database, there is the possibility of executing another command of the database destructively. The injection mechanism would be injection to the user's entrance. Attack aim is adding and changing data and executing remote commands. Some cases of the attack are functions and stored procedures.

3.7. Searching for an Item

GIS web applications for the comfort of their users, and helping them find subjects, have search, in which by entering subjects keyword, concerned Maher is found. Since this search is done in the database, SQL queries can be entered from this part. An injection mechanism has been done to the user's entrance. Attack aim is to detect the database schema, get the amount of the data, and perform denial of service. Some cases of the attacks are union queries, piggybacked queries, using time delays known names.

4. A Heuristic Model

To encounter SQL injection attacks, various methods should be considered. The best way to avoid these attacks is to prevent them. Since the attack on the GIS site is irreparable damage, Maintaining the security requirements is more important, Because GIS web applications without security, privacy, and protection of sensitive information (of) sender and receiver is impossible.

This model has contained 6 levels:

1. IT policy
    - User accessibility
    - Network accessibility
    - Service accessibility
    - ISMS

2. GIS web app security
    - Authentication and login control
    - Authorization
    - URL passing
    - Cookies
    - Multi-layer infrastructure
    - Object-oriented and modular coding
    - Automatic services

3. GIS distributed/mono database security
    - Fetching data and features
    - Process mining
    - Data accessibility

4. Server security
    - Antivirus and firewalling

- OS pack
- Updating
- System file accessibility

5. Network and internet security
   - Ad-hoc
   - Port scan
   - DOS, DDOS
   - WAF services
   - Firewalling (UTM, SIEM)
   - Internet accessing
   - Proxies

6. GIS availability and exchanging services
   - Web services
   - WFS, WMS, and other map services
   - User feature accessibility

Figure 1 shows the relationship among these 6 levels. Figure 2 shows the security steps of the GIS web app model and the person/staff who is/are responsible for it. then try to identify all types of SQL Injection attacks and prevent them. As shown in Figure 1, the Components of this model include different levels. At each level, Methods to prevent SQL Injection attacks again are described.

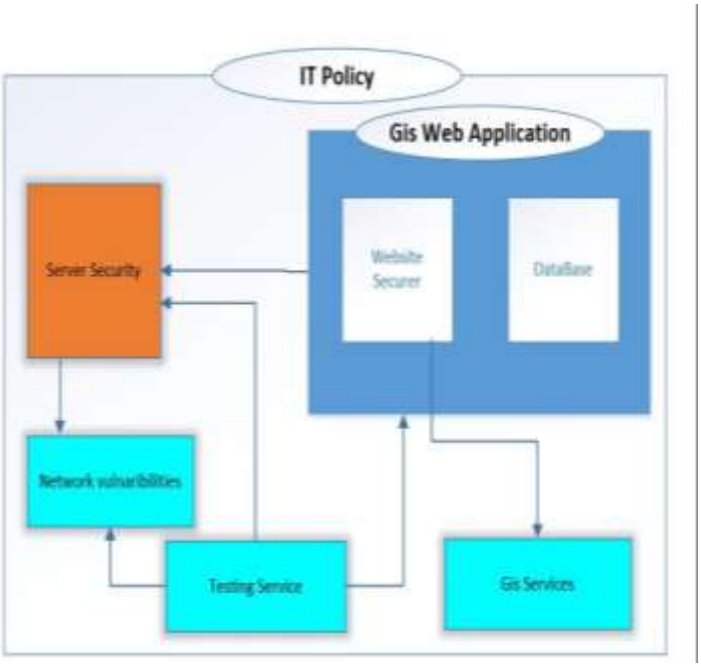

Figure 1. The relationship between different levels in the SQL injection prevention model

4.1. Database Management

Put a safe database driver: This attack can be obtained by putting a safe database driver between the web application and the underlying connection of the database management system. To recognize an

attack, the driver uses SQL queries and the following stack to create an SQL statement, and finally, it can realize an injectable query from the correct query. The driver is not dependent on a particular web application and can be added to every system.

Lockdown in SQL server: There is a list of things to do when creating an SQL Server:

1. Connecting servers' method: It should be checked that just using active network libraries. To be that network utility can be helpful.
2. User accounts existence: user accounts with a low level of access can be created to use the programs. Those user accounts which are unnecessary should be deleted.
3. Using a strong password: lots of the extended stored procedures can be omitted without any damage. Those DLLs containing the codes of extended stored procedures should also be omitted if it is done. All of the sample data banks and examples which are presented in the program as presupposed should be omitted.
4. The user account permissions: a user account that uses a program to access the database should necessarily have the least permission to access its needed objects.
5. Server patches level: There are several attacks of buffer overflow and format string type and several security problems in the patches themselves. So, the server should always be kept updated from the viewpoint of patches.
6. Logged in/out checking and what happens.

4.2. Connect to Database

Use prepared statements: Many programmers think using a managed code can prevent SQL injection vulnerability, but this idea is not right. One of the best ways to avoid this vulnerability is using a prepared statement. These statements have a static structure and usually are designed to improve the database connection. At a low level, those statements separate user data from SQL commands. So, in the case of appropriate use, the user's login can never be transferred as a SQL command.

Error message compatibility: A message should be shown when a database error occurs, but it is possible not to show the whole SQL error message. When a web application is faced with an error, the program should answer the error with an ordinary page or return to a standard place. Thus, debugging information or other details can be hidden to prevent possible hackers.

Use stored procedures: Processes in a database should only be done by the stored procedures. These procedures should have the least level of accessibility and should utilize a parameterized API that dramatically helps with the sanity checking of user data.

4.3. Web-GIS Programming

In this step most systematic and principled way to act is that encryption and programming standards that need to do is this case and should be applied. Visits can be a code for identifying each species that will be useful to the vulnerable. Lots of SQL injection vulnerabilities are a result of invalid input validation. Most of the SQL injections can be recognized by checking the user login, which can be prevented. Different methods of creating validity can be classified as follows: The efforts to process the data and modify them to become valid. In this case, SQL statements can be written so that instead of , using characters like (=, &, space, >, <) equivalent cases be used. Rejecting and refusing those logins which are known as harmful logins. To do that, the user can prevent the entrance of some character link (, &, space, …) or some keywords like (insert, drop shutdown, select union, …) the user. Limiting data type; for example, if a login should be in numeral type, the user should not be able to enter un numeral . values. Limiting the entrance length according to the needs

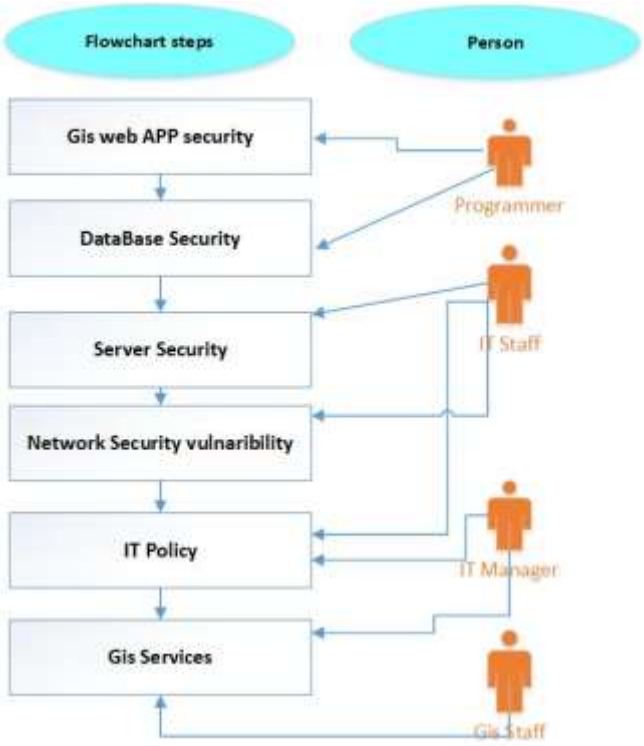

Figure 2. Steps to secure GIS websites

4.4. Access Management to the Database

Links with the least accessibility: To safeguard the database the website manager, should use user accounts with the least necessary accessibility for applications and never use systemic or managerial user accounts like "SA," "dba", " Admin, "and the like.

Not accessing systemic files: The managers should limit SQL server accessibility to systemic files and interactive commands like cmd. Exe.

Disable Ad hoc: Must disable ad hoc report through OLE DB from SQL server. Ad-hoc from OLED provider controls by determining Disallow Ad hoc Access in the registry.

4.5. Process Test and Analyst

One of the efforts which can be made by a web – site manager is using different devices to recognize vulnerable parts of the web – site. For instance, SQL-brute (it can identify the vulnerable parts in blind attacks), SQL-ninja, SQL-bf (the device to check SQL server username), SQL -exec (For system commands which use XP-cmd-shell), absinthe, Acunetix web vulnerability scanner.

4.6. GIS Web Application Managing

It should be noted that complete security is not guaranteed at any time. Therefore, Managers of GIS web application sites must adhere to another principle. The most important recommendation is to ensure that any SQL Injection vulnerability does not exist because even if all the problems are identified and solved, new problems are created daily. To prevent SQL injection attacks, it is recommended to use

parametric reports. Also, suggest knowing the new protocol of electronic web services and updating their usage protocol. Also, they have to figure out the security of web services. In addition, being familiar with legal rules can, in many cases, cause damage to compensate for the attack. As of last prevention and precaution, configuration and testing of the firewall filters to block out unnecessary traffic control. Do not only cause the databases to be more secure but the entire network is safe.

5. Result

Vulnerability testing tools were used to evaluate the model before and after applying the model. Acunetix and DVWA are two tools for this purpose. Figures 3 and 4 show the results of a normal, blind SQL injection attack on one of the Iranian company's GIS Web sites.

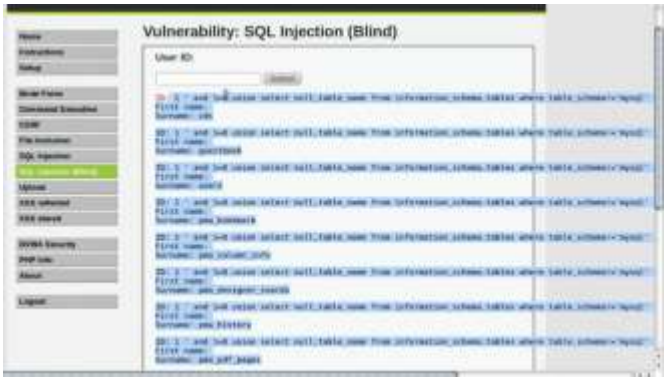

Figure 3. The blind SQL injection attack on the Iranian WebGIS with the DVWA tool.

Figure 4. The SQL injection attack on the Iranian Web GIS with the DVWA tool.

Then the measures and steps mentioned in the model were performed and once again vulnerability testing tools were used to check the situation.

Figure 5. A pic of the DVWA tool that tests an Iranian WebGis with SQL injection attack after applying the model

To confirm the results, the Acunetix vulnerability testing tool was used, the results of which are shown in Figure 6. As it is shown, the WebGIS does not have any high threat (all SQL vulnerabilities are considered as high risk.

Figure 6. A pic of the Acunetix tool that tests the Iranian Web GIS with SQL injection attack after applying the model.

6. Conclusion

Fast and ever-increasing changes in IT and connections, in addition to significant gains including GIS, have caused new threats to the national securities of the countries; therefore, the discussion about securing the virtual world, especially GIS web applications, has gained enormous importance so that whole regular and legal possibilities should be used together with technical possibilities to prevent crimes. Most of the penetrations which happen in a web application are a result of the weakness and gaps in programming and also weakness in the database. In recent years the primary attacks on databases are about those GIS web applications with web services possibility of SQL injection attacks. Thus, this paper

has tried to introduce SQL injection attack mechanism and aim to attack the GIS programs databases, and also briefly mentioned several techniques of this attack, vulnerable parts of a GIS website in connection with its database, and those attacks that in each part can be performed were investigated, and a model to prevent these attacks was presented to be used by managers and web developers of GIS web sites. The main advantage of this model over other research is its simplicity and practicality. It has also identified clear and distinct solutions for IT managers, programmers, and IT security staff that can be used step-by-step to prevent SQL attacks.

As future suggestions, it is recommended to test the model on other GIS sites and other web applications. The same model also needs to be developed for other application-layer attacks, including XSS, DDOS, and path traversal.

References

Ali, I., Adil, S. H., & Ebrahim, M. (2020). Intrusion Detection Framework for SQL Injection. *arXiv preprint arXiv:2009.13868*.

Bhateja, N., Sikka, S., & Malhotra, A. (2021). A Review of SQL Injection Attack and Various Detection Approaches. *Smart and Sustainable Intelligent Systems*, 481-489.

Boudraa, Y. (2019). *New Approach for Detecting SQL Injection Vulnerability in Web application*. University Mohamed Boudiaf-M'Sila Faculty of Mathematics and Computer.

Chen, D., Yan, Q., Wu, C., & Zhao, J. (2021). SQL injection attack detection and prevention techniques using deep learning. In *Journal of Physics: Conference Series* (Vol. 1757, No. 1, p. 012055). IOP Publishing.

Das, D., Sharma, U., & Bhattacharyya, D. (2019). Defeating SQL injection attack in authentication security: an experimental study. *International Journal of Information Security, 18*(1), 1-22.

Giribabu, D., Pandey, K., Rao, K., Bothale, V. M., Reddy, S., PVV, P. R., & Chowdhury, S. (2018). Cybersecurity in WebGIS Environment, International Journal of Computer and Internet Security, 10(1), pp. 11-34.

Gupta, H., Mondal, S., Ray, S., Giri, B., Majumdar, R., & Mishra, V. P. (2019). Impact of SQL Injection in Database Security. In *2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*.

Halfond, W. G., Viegas, J., & Orso, A. (2006). A classification of SQL injection attacks and countermeasures. In *Proceedings of the IEEE international symposium on secure software engineering*.

Hayslett, C. (2019). *Today's Cyberthreat Landscape and the GIS*.

Hu, J., Zhao, W., & Cui, Y. (2020). A survey on SQL injection attacks, detection, and prevention. In *Proceedings of the 2020 12th International Conference on Machine Learning and Computing* (pp.483-488).

İlker, K., & AYDOS, M. (2019). Detection and Analysis of Attacks Against Web Services by the SQL Injection Method. In 2019 *3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)* (pp. 1-4). IEEE.

Johny, J. H. B., Nordin, W. A. F. B., Lahapi, N. M. B., & Leau, Y.-B. (2021). SQL Injection Prevention in Web Application: A Review. In *International Conference on Advances in Cyber Security* (pp. 568-585). Springer, Singapore.

Kausar, M. A., Nasar, M., & Moyaid, A. (2019). SQL Injection Detection and Prevention Techniques in ASP .NET Web Application. *International Journal of Recent Technology and Engineering (IJRTE), 8*(3), 7759-7766.

Keshri, A. K., Sharma, A., Chowdhury, A., Rawat, S. S., & Kiran, K. (2022). SQL–Attacks, Modes, Prevention. *International Journal of Research in Engineering, Science and Management, 5*(1), 162-165.

Li, Q., Li, W., Wang, J., & Cheng, M. (2019). A SQL injection detection method based on adaptive deep forest. *IEEE Access, 7*, 145385-145394.

Liu, M., & Wang, B. (2018). A web second-order vulnerabilities detection method. *IEEE Access, 6*, 70983-70988.

Manyumwa, T., Chapita, P. F., Wu, H., & Ji, S. (2020). *Towards Fighting Cybercrime: Malicious URL Attack Type Detection using Multiclass Classification*. Paper presented at the 2020 IEEE International Conference on Big Data (Big Data).

Nasereddin, M., ALKhamaiseh, A., Qasaimeh, M., & Al-Qassas, R. (2021). A systematic review of detection and prevention techniques of SQL injection attacks. *Information Security Journal: A Global Perspective*, 1-14.

Sabou, G. C., & Maiorescu, I. (2020). Cybersecurity challenges in Smart Cities–a Smart Governance Perspective.

SaiSindhuTheja, R., & Shyam, G. K. (2021). An efficient metaheuristic algorithm-based feature selection and recurrent neural network for DoS attack detection in cloud computing environment. *Applied Soft Computing, 100*, 106997.

Tang, P., Qiu, W., Huang, Z., Lian, H., & Liu, G. (2020). Detection of SQL injection based on artificial neural network. *Knowledge-Based Systems, 190*, 105528.

Vyamajala, S., Mohd, T. K., & Javaid, A. (2018). A real-world implementation of SQL injection attack using open source tools for enhanced cybersecurity learning. In *2018 IEEE International Conference on Electro/Information Technology (EIT)* (pp. 0198-0202). IEEE.

Weinfurter, V., Kirmaier, A. S., Brune, P., & Bergande, B. (2021). Raising Awareness for IT Security in Higher Education-A Teaching Experiment on SQL Injection for Non-Computer Science Majors. In *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 2* (pp. 619-620).

Yunus, M. A. M., Brohan, M. Z., Nawi, N. M., Surin, E. S. M., Najib, N. A. M., & Liang, C. W. (2018). Review of SQL injection: Problems and prevention. *JOIV: International Journal on Informatics Visualization, 2*(3-2), 215-219.