Journal of Engineering Innovations & Technology

journal homepage: www.journal-eit.online

Paper Type: Original Article

RDDT-MS: Real-Time Decryption and Detection of Malicious Traffic Using Man-in-the-Middle Proxy and SnortML

Lulu Liu[1], Liangbin Yang[1]

1.University of International Relations, Beijing, China

Abstract

In network system construction, the issue of network traffic security is of paramount importance. Against the backdrop of strengthening national cybersecurity, traffic inspection plays a significant role in continuously enhancing the security capabilities of network systems. However, as more and more traffic is transmitted using the HTTP protocol over SSL/TLS encryption, this not only provides protection for legitimate communications but also facilitates covert attacks by malicious actors. Traditional traffic detection systems (such as Snort and Suricata) struggle with the challenge of inspecting encrypted traffic, and their detection rules are typically configured based on known attack patterns, making them less effective against evolving new types of attacks.

To address the aforementioned issues, this paper innovatively proposes a comprehensive traffic detection frame. The main contributions of this work are summarized in three parts:

1.Real-time Traffic Decryption and Detection Framework: We have designed and implemented an efficient real-time traffic decryption and detection architecture that cleverly integrates man-in-the-middle proxy technology to enable real-time decryption of HTTPS encrypted traffic.

2. Model Training and Validation Using an SQL Injection Dataset: To validate the effectiveness of the framework, we utilized a dataset containing SQL injection attacks to train the detection model and applied it within the framework to evaluate its capability to detect unknown attack traffic.

3. For the development and testing of the SnortML plugin: To enable real-time traffic feature extraction in SnortML, we added feature extraction capabilities to the SnortML detector and used the tcpreplay tool to simulate complex network traffic scenarios.

Through our research, we have achieved automatic detection and efficient defense against real-time, encrypted, and potentially unknown attack traffic. Experimental result clearly demonstrate that the framework has significantly improved the accuracy of encrypted traffic detection, providing strong technical support and innovative ideas for building a safer and more robust network environment.

Keywords: Traffic detection, Traffic Decryption, Men-In-The-Middle, SnortML, Machine Learning

Introduction

Research background

In today's era of rapid internet development, network traffic, as the lifeblood of the information society, is of paramount importance. With the increasing frequency of information exchange, especially the transmission of sensitive data over networks, cybersecurity issues have risen to unprecedented heights, becoming a focal point of societal concern. In President Xi Jinping's discussions on national security[1], cybersecurity holds a pivotal position, as he explicitly states: "Without cybersecurity, there is no national security." This statement profoundly elucidates the symbiotic relationship between cybersecurity and the development of information technology, emphasizing that both must advance together, like the dual wings of a bird or the twin wheels of a vehicle, to propel progress forward.

Given this context, China has been steadily increasing its investment in cybersecurity in recent years, aiming to align with international standards and build an impregnable cybersecurity barrier. According to the latest data from the "2023 Analysis Report on China's Cybersecurity Industry[2]" the size of China's cybersecurity market reached approximately 63.3 billion RMB in 2022, with an annual growth rate of 3.1%. It is predicted that this industry will continue to expand at an average annual compound growth rate exceeding 10%, reaching a market size of over 80 billion RMB by 2025.

Simultaneously, the emergence of new technologies such as cloud computing, big data, and the Internet of Things (IoT) has significantly enhanced social productivity and convenience, while also presenting new challenges to cybersecurity. These technologies act like double-edged swords, expanding application boundaries on one hand and significantly increasing potential attack surfaces on the other, making network threats increasingly complex and diverse. The "2023 Cybersecurity Vulnerability Situation Report" shows that 29,039 cybersecurity vulnerabilities were discovered in 2023, a 16.6% increase compared to 2022, setting a new record. Among these, Web application vulnerabilities accounted for 40.3%, ranking first and highlighting their prevalence and harmfulness.

This trend clearly reflects the surge in demand for cybersecurity, with both enterprises and various sectors actively seeking more effective security measures. For example, the national government has enacted a series of policies and regulations to strengthen cybersecurity management, such as the "Cybersecurity Law of the People's Republic of China," to ensure the

safety and controllability of cyberspace. Additionally, companies and organizations are actively seeking more effective security measures, including advanced firewall technologies, intrusion detection systems (IDS), intrusion prevention systems (IPS), encryption technologies, and identity authentication mechanisms.

Therefore, in the face of increasingly severe cybersecurity challenges, strengthening cybersecurity defenses and enhancing threat detection and response capabilities, particularly for web services, is crucial. This is not only necessary for national security but also essential for protecting personal information and promoting the healthy development of the digital economy.

Research Status

Since the birth of the Internet, Web services have been among the most important and widely used applications on the web. The HTTP protocol, as the primary channel for transmitting Web service data, has become one of the most critical and common application-layer protocols on the Internet. However, with the rise of e-commerce activities such as online transactions and internet banking, security issues related to Web services have become increasingly prominent. As a result, the security requirements for the HTTP protocol, which serves as the transmission channel for Web services, have reached a new level of importance.

Although the HTTP protocol is widely used, from a cybersecurity perspective, it has some significant security flaws. One major issue is the transmission of data in plaintext form, meaning that all data is sent without encryption or digital signatures. Instead, the data is transmitted in clear text, providing attackers with opportunities to easily intercept sensitive information such as usernames, passwords, ID numbers, credit card numbers, and more through various means of attack, leading to serious security incidents.

To address these issues, the HTTPS protocol was introduced, which combines the HTTP protocol with the SSL/TLS protocol. The SSL/TLS protocol provides confidentiality, data integrity, data authenticity, non-repudiation, and identity verification through digital certificate authentication for the entire communication process. Consequently, HTTPS has become the most widely used encrypted transmission protocol.

However, this encrypted transmission method also provides cover for malicious actors to conduct covert attacks. Attackers may exploit the characteristics of HTTPS connections to hide their malicious activities, making it more difficult to detect and prevent these attacks.

For instance, Fangfang Jia[3] noted that detecting encrypted trojans based on HTTPS has become more challenging, while Xie J[4] proposed a method for detecting HTTP trojans, which, however, is not applicable to HTTPS encrypted traffic. Intrusion detection systems (IDS) are another commonly used detection method. An IDS processes network traffic data to quickly distinguish between normal and anomalous data, enabling it to determine whether a behavior

constitutes an attack. J.P. Anderson[5] first introduced the concept of intrusion detection in his book "Computer Security Threat Monitoring and Surveillance" and provided a detailed explanation. Zalbina et al[6] proposed a detection method based on IDS, but this approach cannot be applied to HTTPS encrypted data. Meanwhile, Feng Leng et al[7] analyzed attacks through the examination of Snort rules, but this method can only target known attacks. Given the constantly evolving nature of attack methods, relying solely on rule matching is no longer sufficient. Therefore, there is an urgent need for a new approach that can effectively detect both known and unknown encrypted traffic attacks.

Methodology

This paper addresses the aforementioned issues by proposing a real-time encrypted traffic detection framework based on a man-in-the-middle proxy and SnortML. The framework decrypts encrypted traffic via a man-in-the-middle proxy, saves it as a .pcap file, and uses tcpreplay to convert the .pcap file into a stream of data. The SnortML plugin for secondary development is then used to perform real-time traffic detection. Additionally, machine learning is employed by training an SQL injection detection model and loading it into SnortML to detect unknown threats. The overall structural framework is illustrated in Figure 1.

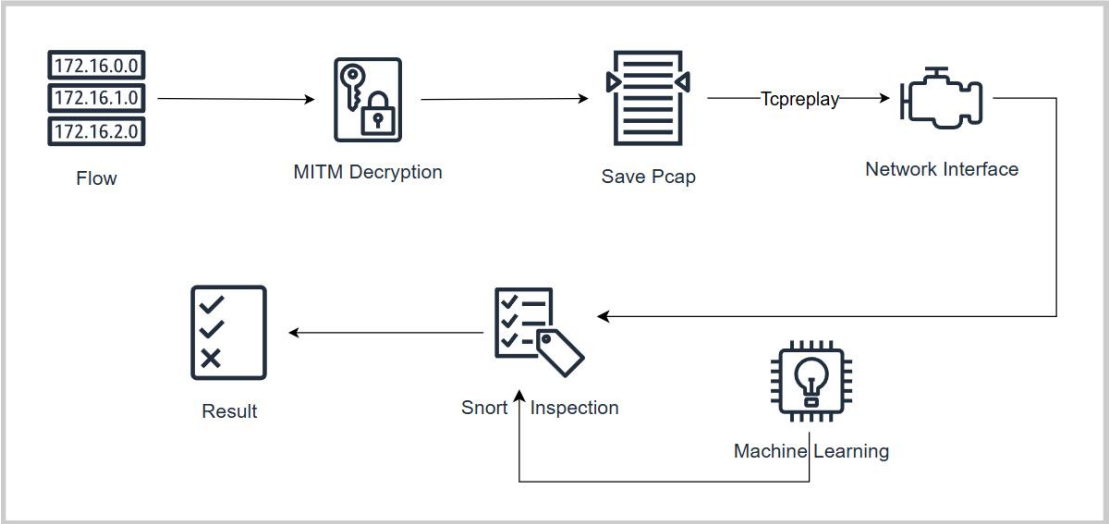


Figure 1 RDDT-MS

MITM

During normal access, when a client establishes an encrypted HTTPS connection with a server, the process involves certificate exchange, certificate validation, key negotiation, and encrypted communication to achieve traffic encryption. This process is shown in Figure 2.

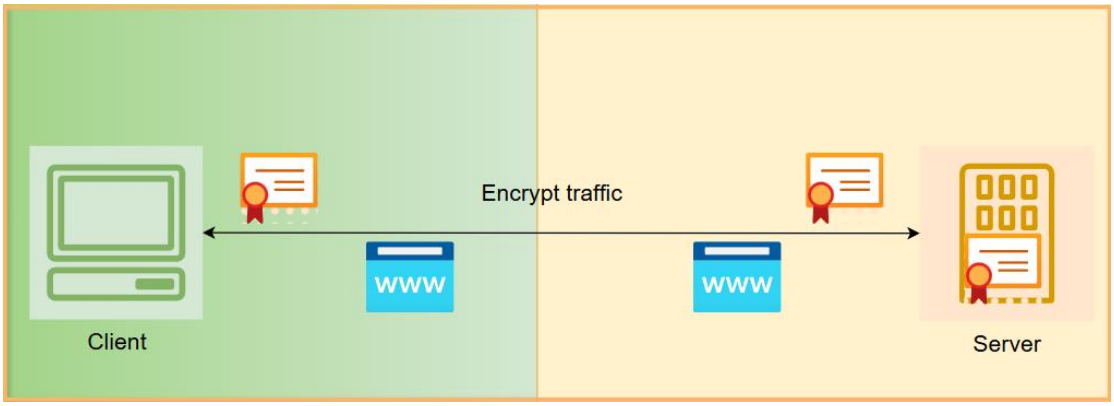


Figure 2 Normal Access

A Man-in-the-Middle (MITM) broker[8] is an indirect method for decrypting traffic. This method involves placing a proxy server virtually between the two communicating parties so that the proxy server can establish separate connections with the original client and the server, thereby maintaining the information exchange between the original client and the server. In this configuration, both the original client and the server believe they are communicating with legitimate counterparts. At the same time, the MITM broker can read and decrypt the information passing between them, transforming the originally encrypted data into plaintext. Through this approach, we can obtain plaintext traffic for subsequent security inspections. As Figure 3.

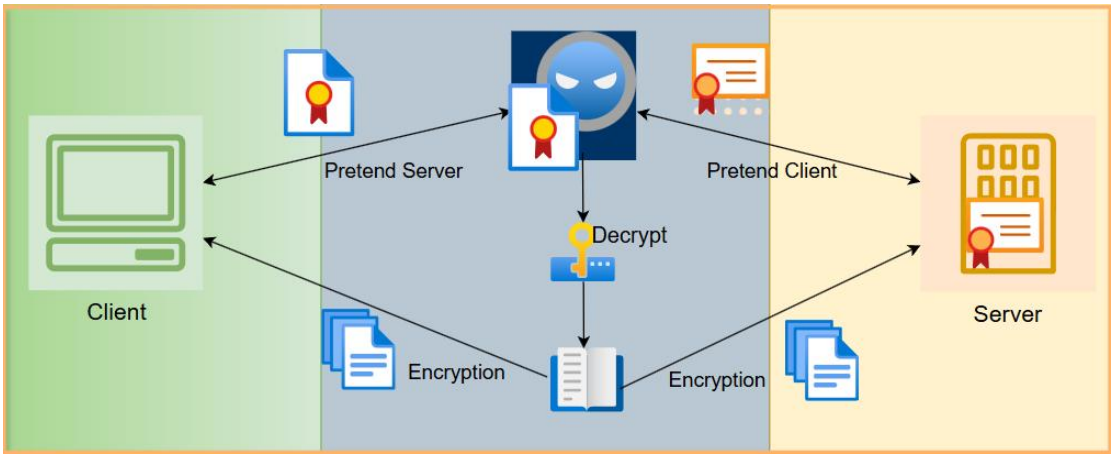


Figure 3 Men-In-The-Middle Attack

This paper uses Polarproxy as the MITM tool for HTTPS decryption. It is important to note that the client must be configured to trust the certificate generated by Polarproxy. This is because Polarproxy generates a self-signed certificate to decrypt HTTPS traffic, which needs to be added to the client's trusted certificate list to ensure smooth.

SnortML

Gibert's D[9] analysis mentions the use of machine learning for analyzing malicious behavior, even when such behaviors are unknown, providing us with a good line of thought. Machine

learning algorithms can learn patterns from large amounts of historical data and automatically adjust their parameters to adapt to new input data. This makes machine learning particularly suitable for detecting novel attacks that do not have clearly defined characteristics.

Snort points out in the blog[10], SnortML is a machine learning-based detection engine for the Snort intrusion prevention system. There are two components to this new detection engine. The first component is the snort_ml_engine, which loads pre-trained machine learning models, instantiates lassifiers based on these models and then makes the classifiers available for detection. The second is the snort_ml inspector, which subscribes to data provided by Snort service inspectors, passes the data to classifiers, and then acts on the output of the classifiers.

We have added feature extraction functionality in the snort_ml inspector module. After obtaining traffic data, real-time traffic features such as character length and symbol count are extracted and predicted by the model together with the original data.

SQL injection attacks

SQL injection attacks are a common cybersecurity threat, exploiting incorrect handling of user input by applications, allowing attackers to execute malicious SQL queries[11]. This can lead to serious consequences such as data leaks, data tampering, and system crashes. Methods typically used to carry out SQL injection attacks include:

1. Closing Quotes: By adding extra quotes or comments to "close" string literals in SQL statements, attackers can alter the structure of the SQL statement.

2. Using Comments: Employing comments within SQL statements to "ignore" parts of the query.

3. Exploiting Stored Procedures: If the database supports stored procedures and the application's SQL queries call them, attackers might inject specific parameters to invoke different stored procedures or modify how they execute.

4. Union-Based SQL Injection: When the result set of an SQL query can be "united" with another query's result set, attackers can use UNION SELECT statements to retrieve additional data.

5. Boolean Blind SQL Injection: In cases where the application does not return the results of the SQL query but instead provides different responses (such as success or failure) based on whether the SQL statement executes successfully, attackers can infer the database structure or data by sending different queries and observing the application's response.

This article, based on the methods of SQL injection attacks, conducts a feature analysis of the

SQL injection dataset to identify data that shows clear classification effects. Through a thorough analysis of the different techniques used in SQL injection attacks, this study performs an exhaustive feature analysis of the SQL injection dataset. By uncovering unique patterns and traces left by these attack methods in the dataset, we aim to identify data features that effectively distinguish and classify SQL injection attacks. This will provide strong support for building more accurate and efficient SQL injection defense mechanisms.

Experimental

Experimental Environment

The experimental environment for this study is TensorFlow 2.9, Python 3.9, running in Jupyter Notebook, with the operating system being Ubuntu 24.04 LTS, and Snort version 3.3.0.

To simulate a real network environment, we used the command 'sudo ip link add test type dummy' to create a virtual network interface named 'test' in Ubuntu and configured Snort to listen on this 'test' interface. We used the tcpreplay tool to convert decrypted and saved pcap files into actual data streams sent to the 'test' interface, simulating Snort's real-time traffic inspection capabilities.

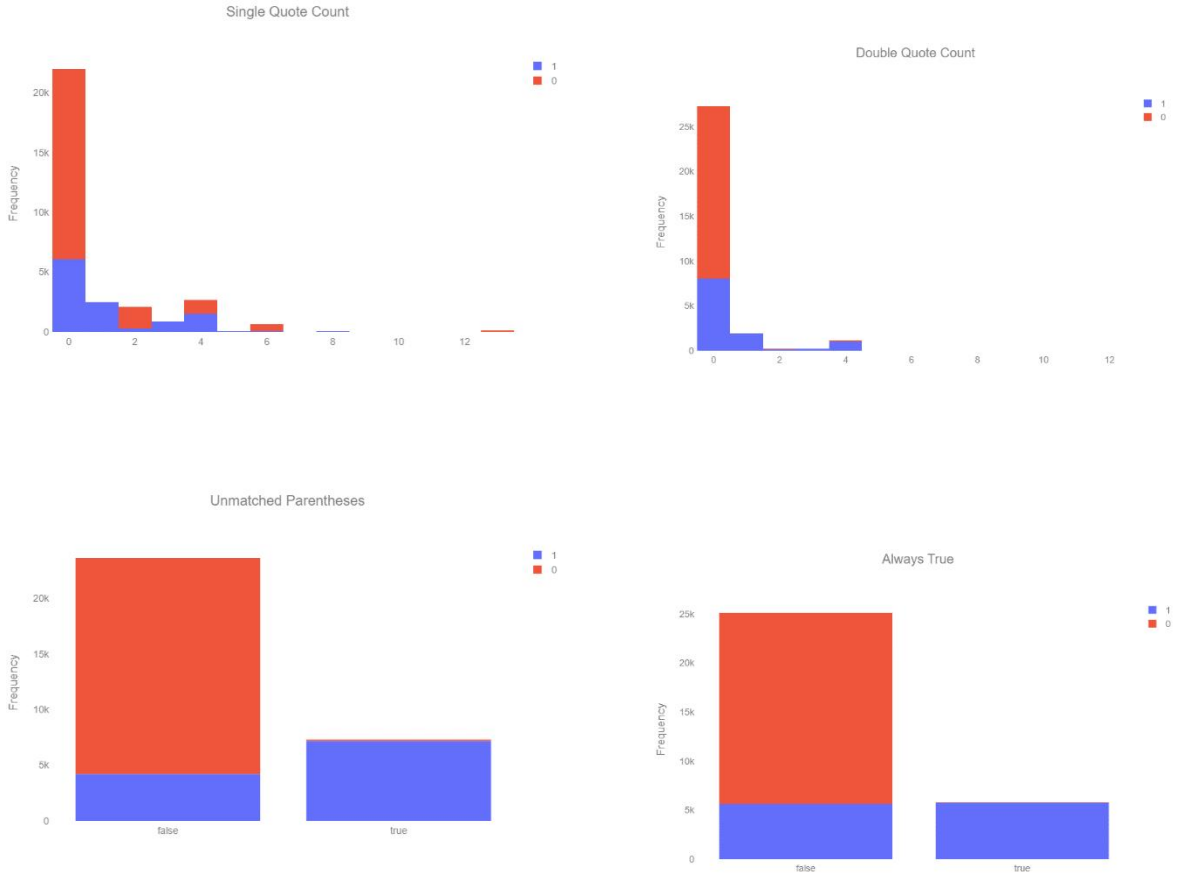
Dataset analysis

The dataset used in this article is from Kaggle Datasets[12]. To ensure the quality and accuracy of the data, we performed preprocessing on the raw data, specifically including deduplication and removal of null values. Below is the information about the data before and after processing.As Table 1 .

Table 1 Dateset

	Malicious	Benign
Before processing	11382	19537
After processing	11378	19529

Based on the characteristics of SQL injection, we analyzed four key features: single quotes, double quotes, parentheses matching, and whether the statement is always true, as shown in the following Figure 4 .



Through the analysis of SQL injection attack characteristics, we noticed that the feature indicating whether the statement is always true had a better effect in distinguishing the data. However, the other four features—occurrences of single quotes, double quotes, and parentheses and their matching—did not show a significant distinguishing effect. Therefore, we further explored improvement strategies for these features.

To better utilize these features for distinction, we considered the typical construction of statement closure through symbols in SQL injection attacks. Based on this, we computed whether single quotes, double quotes, and parentheses appear in pairs (whether they occur in even numbers) as new features. As shown in the following Figure 5 , the distinguishing effect is better.

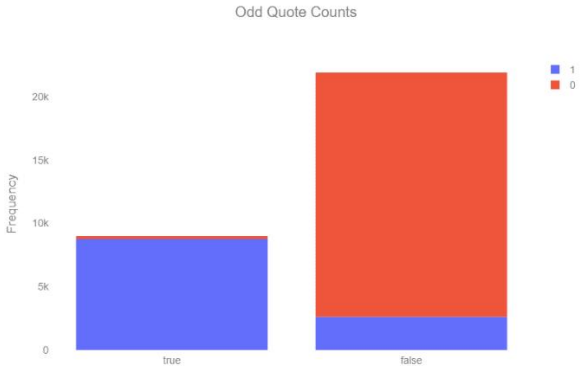


Figure 5 Odd and Even Numbers

Experimental

This article uses the Decision Tree Classifier machine learning algorithm and sets up ablation experiments to prove the effectiveness of the features identified in this study.

Ablation Experiment 1: Only the SQL statement, without any features.

Ablation Experiment 2: The SQL statement along with four key features: single quotes, double quotes, parentheses matching, and whether the statement is always true.

Ablation Experiment 3: The SQL statement along with single quotes, double quotes, parentheses matching, whether the statement is always true, and whether they occur in even numbers.

The experimental results are as Table 2:

Table 2 Experimental Result

	F1	Recall
Ablation Experiment 1	0.76	0.62
Ablation Experiment 2	0.88	0.79
Ablation Experiment 3	0.92	0.91

Comparing Ablation Experiment 2 with Ablation Experiment 1, it can be observed that the four features we examined—single quotes, double quotes, parent matching, and the condition of always being true—are indeed effective.

Comparing Ablation Experiment 3 with Ablation Experiment 2, it can be observed that the feature we examined—whether occurrences happen in even numbers—is indeed effective.

CONCLUSION

In this paper, we first propose the RDDT-MS, which can detect encrypted traffic in real-time and identify unknown threats. We analyze SQL injection statements to identify features that are

effective for classification, which are then used to train our model. Furthermore, we extend the SnortML Inspector to extract feature information from real-time traffic. In future work, we will explore how to update the model in real-time and discover additional data features.

ACKNOWLEDGMENTS

This work was supported by Student Academic Research Training Project of University of International Relations (No. 3262024SYJ19).

References

[1] xinhua net. The network security and information leading group held the first meeting [EB/OL]. http://www.cac.gov.cn/2014-02/28/c_126205866.htm.2014-02-28.

[2] H2C. Cybersecurity Vulnerability Situation Report 2023.2024. https://www.h3c.com/cn/d_202402/2056604_30003_0.htm

[3] Jia Afang, Chen Shi, Wu Shuang, et al. Encryption Trojan Detection Method based on HTTPS Covert tunnel [J]. Journal of Information Engineering University, 2019, 4.

[4] Xie J, Li S, Yun X, et al. Hstf-model: An http-based trojan detection model via the hierarchical spatio-temporal features of traffics[J]. Computers & Security, 2020, 96: 101923.

[5] Anderson J P. Computer security threat monitoring and surveillance[J]. Technical Report, James P. Anderson Company, 1980.

[6] Zalbina M R, Stiawan D. HTTP Attack Detection System Based on HTTP Inspect Preprocessor and Rule Options[J]. Available on academia. edu, last accessed April, 2023, 10.

[7] Leng Feng, Zhang Cuiling, Chen Wanyu, et al. Analysis of attacks from Protocol Information of Snort rules [J]. Computer Applications, 2022.

[8] Mallik A. Man-in-the-middle-attack: Understanding in simple words[J]. Cyberspace: Jurnal Pendidikan Teknologi Informasi, 2019, 2(2): 109-134.

[9] Gibert D, Mateu C, Planes J. The rise of machine learning for detection and classification of malware: Research developments, trends and challenges[J]. Journal of Network and Computer Applications, 2020, 153: 102526.

[10] Snort. Talos launching new machine learning-based exploit detection engine. 2024.03.15. https://blog.snort.org/2024/03/talos-launching-new-machine-learning.html

[11] Halfond W G J, Viegas J, Orso A. A Classification of SQL Injection Attacks and Countermeasures[C]//ISSSE. 2006.

[12] Kaggle. biggest-sql-injection-dataset. https://www.kaggle.com/datasets/gambleryu/biggest-sql-injection-dataset/data