# Implementation and Analysis of BREACH Attack System

LIU Jin-jian, GUO Zi-yun, YANG Zong-qi, LI Jun-le and XU An-jun

***Abstract*—In the transmission of network information, compression algorithms such as DEFLATE are commonly used to minimize bandwidth consumption. However, these algorithms pose a risk of secret information leakage. The Browser Reconnaissance and Exfiltration via Adaptive Compression of Hypertext (BREACH) attack exploits pre and post compression changes in the HTTP response information body to steal information, displaying typical SSL/TLS attack characteristics. This project aims to raise awareness and educate users about the BREACH attack. It involves developing a user-friendly interface and conducting a comprehensive simulation of the BREACH attack, analyzing its characteristics and mechanisms, and comparing it with similar attacks to gain a deeper understanding. The project's findings will be summarized, and prevention suggestions will be provided to enhance network security awareness.**

***Index Terms*—compression, BREACH attack, SSL/TLS, security**

## I. INTRODUCTION

Compression is a useful technique for saving the bandwidth and widely used in various systems, while acting as a side-channel[1] without any preventive measures. However, when plaintext data is compressed before encryption, the length of the resulting ciphertext can inadvertently reveal information to potential attackers. This vulnerability has led to the emergence of attacks that exploit this side-channel information.

The BREACH attack is one of them, as a compression side-channel attack, which targets information compressed in HTTP response bodies. This attack can be used to extract login credentials, anti-CSRF tokens and other sensitive, personally identifiable information from SSL-enabled websites.

This paper primarily focuses on give a completely sight in the system we implement, in addition to that, the paper also providing a comprehensive analysis of the BREACH attack, in comparison to other similar attacks. The aim of the paper is to present a detailed examination of the complete BREACH attack, highlighting its intricacies and implications.

### *A. Overview of CRIME and BREACH*

CRIME (Compression Ratio Info-leak Made Easy) and BREACH (Browser Reconnaissance and Exfiltration via Adaptive Compression of Hypertext) are two security attacks related to SSL (Secure Sockets Layer) and TLS (Transport Layer Security).

The CRIME attack, introduced by Rizzo and Duong in 2012[2] targets SSL/TLS and exploits the compression ratio of HTTP responses to discover session tokens or other sensitive information. It can be triggered when a user visits a website controlled by an attacker, and the attacker manipulates the victim into sending requests to a specific URL, leading to significant information leakage and potential loss[4].

BREACH, on the other hand, is another attack on SSL/TLS that was discovered in 2013[3]. It leverages the compression of HTTPS responses to extract sensitive information, such as CSRF tokens or authentication credentials, from the compressed responses. The attack takes advantage of the way compression algorithms work by analyzing the variations in the compressed data size.

The project mainly focus on BREACH and the process and mechanism in detail will be discussed in the subsequent sections.

L. JJ is a student at the College of Software, Sichuan University, Chengdu, China.

G. ZY is a student at the School of Cyber Science and Engineering, Sichuan University, Chengdu, China.

Y. ZQ is a student at Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China.

L. JL is a student at School of Cybersecurity, Northwestern Polytechnical University, Xian, China.

X. AJ is a student at College of Informatics, Huazhong Agricultural University, Wuhan, China.

### B. *Our contribution*

Our project team's objective is to present the characteristics and mechanisms of the BREACH attack for educational purposes. We have successfully implemented the BREACH attack model and constructed a WEB platform with user-friendly pages, considering educational aspects, to make the system accessible and rewarding for any user. Furthermore, we conducted research to analyze and compare the BREACH attack with other popular SSL tools, and we have compiled a comprehensive summary of our findings

## II. BREACH ATTACK

### A. *DEFLATE Compression Algorithm*

The BREACH attack is mainly focus on DEFLATE compression algorithm, which is a lossless data compression file format that uses a combination of LZ77 and Huffman coding.

For LZ77, it basically Using slide window contains data, moving forward as progress. Main process are as follows[5]:

First, the algorithm begins with an empty buffer and an empty output stream. Then, it searches for patterns of data within the sliding window that match the current input data being processed. When a match is found, it represents it as a pair (length, distance), where "length" indicates the number of matching characters and "distance" shows the offset from the current position to the start of the match within the sliding window.

Next, the algorithm outputs the (length, distance) pairs for matches and the literal characters that do not match any previous data. This output is designed to be stored or transmitted efficiently. As new data is processed, the sliding window slides forward, discarding the old data and adding the new data to the window.

The process repeats itself as the algorithm continues to search, encode matches, output data, and update the sliding window with each iteration.

For Huffman coding, which is the process of finding or using such a Huffman code, which is a particular type of optimal prefix code that is commonly used for lossless data compression.

*Mechanism*

Breakthrough point：The length of the compressed data is still visible after compressing with the DEFLATE algorithm,

The attacker needs to Inject his guesses of the secrets (using JavaScript) into the HTTP response bodies at the first time, then observe the time when responses would be highly compressed, and the output length differs, means the guess matches, after that, it's time to detect the secret information, and finally the complete secret has been extracted.

## III. IMPLEMENTATION

### A. *Overview*

In this part, we mainly simulate the attack and display it in webpage, adding functions so that users can easily interact with it.

### B. *Implement the Web*

We use html, css, javascript to develop the webpage and interactive functions, the main steps and figures related are shown as follows.

Firstly, we trying to design and finish the layout of the webpage.


Figure 1. Layout of the webpage

Then, design the style of different elements and add appearance elements on the page, find the appropriate background image, then fill in the text and make some adjustments.

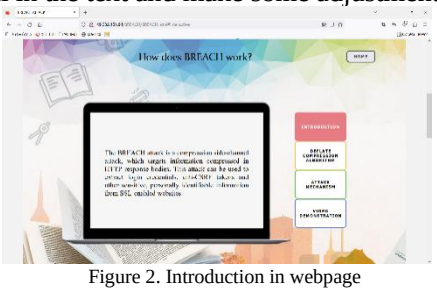


Figure 2. Introduction in webpage

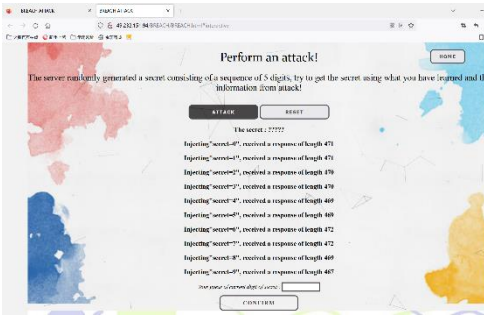


Figure 3. Interaction design in webpage

In order to show more information and raise the concern of prevention, we also give some advice of preventing such an attack.

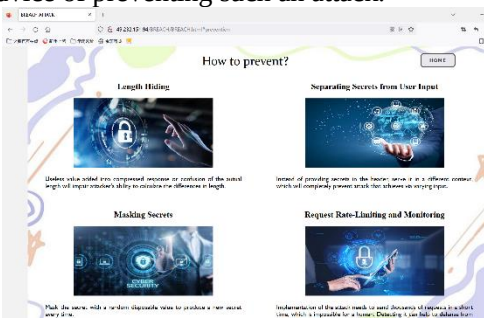


Figure 4. Preventing advice shown in webpage

### C. *Implement the Attack*

This lab system is based on the Docker virt ualization platform and Tencent Cloud lightw eight servers. The Docker virtualized containe rs are deployed on Tencent Cloud lightweight servers, simulating a victim host within one Docker container, while using the cloud serve r as the attacking host to launch attacks agains t the Docker container.

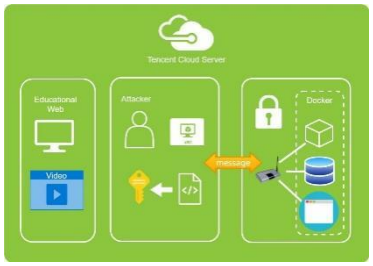


Figure 5. Attack implementation architecture

In detail, we first to gain control of the victim's network. In this system, the attacker is able to inject code to the victim's machine for execution, and note that we primarily assume that we have already gained control of the victim's host network and successfully injected attack script into the victim's host, gaining partial control over the victim's network.

The attack script initiates multiple requests to the target website, which are intercepted and analyzed by the attacker. Since the attacker script runs in a different context from the target website, it is bound by the same-origin policy and unable to access the plaintext or encrypted responses directly. However, the encrypted requests and responses are accessible to the sniffer through direct network access.

By comparing the lengths of the encrypted data, the sniffer can deduce information about the corresponding plaintext lengths and their relationships.

A successful attack completely decrypts a portion of the plaintext. The portion of the plaintext which the attack tries to decrypt is the secret. That portion is identified through an initially known prefix which distinguishes it from other secrets. Each byte of the secret can be drawn from a given alphabet, the secret's alphabet.

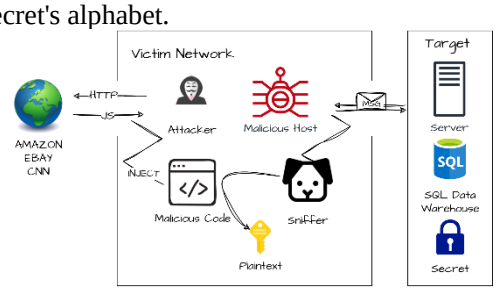


Figure 6. Attack schematic diagram

## IV. RESULT AND ANALYSIS

### A. *The result of our system*

The system successfully worked. During user operations, the occurrence of BREACH attacks and web page interactions can occur correctly

### B. *Analysis*

The analysis of BREACH attack mainly focuses on the advantages and disadvantages compared to other attacks related to it, and given to the characteristic of this attack, methods of preventions are also provided [6]

Advantages of the BREACH Attack are shown as follows:

- Effective Information Extraction: The BREACH attack can effectively extract sensitive information from vulnerable web applications. By manipulating the size of compressed responses, attackers can infer specific parts of the plaintext, enabling them to extract valuable data.
- Low-Cost Attack: Implementing the BREACH attack does not require

sophisticated tools or substantial computing resources. Attackers can use relatively simple scripts or tools to perform the attack, making it accessible to a wide range of adversaries.

• Difficult to Detect: Detecting a BREACH attack is challenging since it leverages legitimate TLS/SSL communications. As a result, traditional intrusion detection systems may struggle to identify the attack, allowing it to remain undetected for extended periods.

Disadvantages of the BREACH Attack are shown as follows:

• Limited Targets: Not all web applications are vulnerable to the BREACH attack. Several conditions must be met for a successful attack, such as the presence of specific types of sensitive data and the usage of HTTP response compression.

• Mitigations Available: The BREACH attack received significant attention upon its discovery, leading to various mitigation strategies being developed. For example, disabling HTTP compression, randomizing secrets, or implementing anti-CSRF tokens can help protect against this attack.

• Complex Execution: While the attack concept is relatively simple, carrying out a successful BREACH attack can be more challenging in practice. Attackers need to find and exploit vulnerable web applications and manipulate the victim's browser to send cross-origin requests.

### *C. Prevention:*

The available prevention methods against this attack are currently limited. The article briefly introduces these methods without delving into extensive details, and they primarily include the following:

The first is length hiding: This method involves adding random padding or extra characters to the plaintext data before compression, making the resulting ciphertext lengths less predictable and harder for attackers to infer sensitive information.

Disabling compression is also a useful method: By disabling compression in the communication protocols, the vulnerability exploited by the BREACH attack can be eliminated. However, this may lead to increased bandwidth usage.

Furthermore, separating secrets from user input can also be useful: Keeping sensitive information, such as session tokens or authentication credentials, separate from user-controllable input can help reduce the attack surface and make it more challenging for attackers to exploit the compression vulnerabilities.

For more approaches, are masking secrets and request rate-limiting and Monitoring: the pre approach involves obfuscating the sensitive data in a way that makes it challenging for attackers to recognize or distinguish specific patterns related to the secrets. For the post one, implementing rate-limiting measures on incoming requests and monitoring for suspicious activity can help detect and mitigate BREACH attacks in real-time.

Up to 2021, the BREACH attack is still very effective, although exact numbers have not been found, there are not a few websites with weaknesses in this regard, and the protection methods are mostly limited to the few described above, which has research prospects.

### *D. Attacks comparison*

This section provides a discussion on some attacks that most of which are stated above. Our discussion is based on specific criteria[7] that are selected to evaluate the current state of the art, which are:

• Weakness: Identifies the vulnerability of the system that has been attacked.

• Effect: Identifies the action that has been enforced by the attack.

• Limitation, to find a limitation on the current solution for the attack.

Such discussions on these criteria are shown in TABLE 1 as follow.

TABLE1: ATTACKS COMPARRISON

| Attack | Effects | Weakness |
|---|---|---|
| BEAST | Recover the cookies | Predictable IV in CBC mode |
| CRIME | Discover session token or other secret information | Compressed size of HTTP requests |
| TIME | Infer the compressed payload's size | Compressed size of HTTP response |
| BREACH | Extracting the secrets behind the HTTP response | Compressed size of HTTP response |
| Lucky 13 | Get the decrypted message without key | Pad is not include in MAC |

## V. CONCLUSION

BREACH attack is a typical compression-based side-channel attack that has been widely discussed in the internet security community. Our work is to implement the attack and take it as an educational use for all of the users have a deeply understanding of the attack and hopefully enhancing awareness of network security prevention, which is becoming increasingly important in contemporary times. In addition, our project's analysis and comparison of BREACH attacks can also predict potential forms of future network attacks at a certain level, hoping to be helpful for future research on related network security

### *References*


[1] Kelsey, J: Compression and information leakage of plaintext. In Daemen, J., Rijmen, V., eds.: FSE 2002.

[2] Rizzo, J., Duong, T.: The CRIME attack Presented at ekoparty 12. http://goo.gl/mlw1X1

[3] Gluck, Y., Harris, N., Prado, A.: SSL, gone in 30 seconds: A BREACH beyond CRIME. In: Black Hat USA 2013.

[4] I. Sankalpa, T. Dhanushka, N. Amarasinghe, J. Alawathugoda and R. Ragel, "On implementing a client-server setting to prevent the Browser Reconnaissance and Exfiltration via Adaptive Compression of Hypertext (BREACH) attacks," 2016 Manufacturing & Industrial Engineering Symposium (MIES), Colombo, Sri Lanka, 2016.

[5] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," in IEEE Transactions on Information Theory, vol. 23, no. 3, pp. 337-343, May 1977.

[6] ATTACKS ON SSL A COMPREHENSIVE STUDY OF BEAST, CRIME, TIME, BREACH, LUCKY 13 & RC4 BIASES, Pratik Guha Sarkar, Shawn Fitzgerald, San Francisco, August, 2013.

[7] A. E. W. Eldewahi, T. M. H. Sharfi, A. A. Mansor, N. A. F. Mohamed and S. M. H. Alwahbani, "SSL/TLS attacks: Analysis and evaluation," 2015 International Conference on Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE), Khartoum, Sudan, 2015.