Received November 17th 2025; accepted December 24th 2025. Date of publication December 31st 2025
Digital Object Identifier: https://doi.org/10/25047/jtit.v12i2.460

Mitigating Distributed Denial of Service Attacks on IoT Systems Using Gemstone Architecture

MOHAMMAD ROBIHUL MUFID, YOGI PRATAMA, ARNA FARIZA, SANIYATUL MAWADDAH, MUCH CHAFID, AGUS WIBOWO

Politeknik Elektronika Negeri Surabaya, Jl. Raya ITS, Sukolilo, Surabaya, Indonesia

CORESPONDING AUTHOR: MOHAMMAD ROBIHUL MUFID (email:mufid@pens.ac.id)

ABSTRACT One of the problems in the Internet of Things (IoT) system is the Distributed Denial of Service (DDoS) attack on the information technology infrastructure in the internet network. This is because the IoT device system does not have a gateway portal configuration that is able to provide the required security and privacy protection. This study focuses on the mechanism for reducing the impact of the http flood type DDoS attack on the framework layer using the Gemstone architecture. Gemstone architecture is a PHP-based framework integrated with PHP Swoole. PHP Swoole utilizes event-driven which provides several features to access the transport layer on onConnect so that it can be used to implement initial security such as access control lists, connection concurrency management, and server performance optimization. The methodology used is to develop a TCP connection filtering algorithm by implementing a simple queue system by accepting 67% of connections to be forwarded to the next layer and 33% of connections will be queued with a 2-second timeout. The results of this study indicate that the server can minimize the impact of DDoS and handle traffic specifically for http requests with an average latency of 871.8 ms.

KEYWORDS: Internet of Things, PHP Swoole, TCP, DDoS, event-driven.

I.INTRODUCTION

IoT involves millions of connected devices, making it an attractive target for Distributed Denial of Service (DDoS) attackers to exploit vulnerabilities in devices that often have low security. Many IoT devices are designed with limited resources (such as computing power and memory), making it difficult to implement strong security protocols, making the devices more vulnerable to attacks. IoT is often used in critical sectors such as transportation, healthcare, and energy. DDoS attacks on these systems can cause serious disruptions, such as service outages that compromise public safety [1], [2].

Distributed denial of service attacks are a type of attack carried out with the aim of flooding traffic from web services by sending large-scale requests from the http, tcp, udp protocols with the aim of causing the server to collapse and be unable to process a request [3], [4]. As Salim et al. [5] highlight in their comprehensive survey, the DDoS attack landscape and preservation mechanisms are continually evolving, necessitating ongoing research and mitigation strategy development.

DDoS attack detection and mitigation models vary greatly because they have several types of attacks, types of protocols used, volume of attacks, and unique attack patterns. Diaba et al. [6] the importance of developing robust information metrics for accurate attack detection. In the context of cloud computing environments. This research focuses on detecting low and high-level DDoS attacks by identifying attack parameters, attack patterns, and the combination of protocols used. Their evaluation results provide informative insights into the challenges of accurately identifying various types of DDoS attacks. Alshahrani et al. [7] discusses how Software-Defined Networking (SDN) can be leveraged to combat DDoS attacks, highlighting the potential for innovative approaches in network security.

With the development of security technology on web servers, there are many third-party features that make it easier for users to customize and combine DDoS handling as offered by web hosting

providers in their services [8], [9]. However, to get this feature you need to pay for a premium package which is quite financially draining considering that the need for DDoS protection on some medium to lower scale websites does not require special protection to carry out DDoS mitigation [10]. Danta et al. [11] provides a taxonomy of these attacks, underscoring the need for dedicated defence mechanisms at the application layer. This research aims to provide solutions related to handling DDoS attacks at the framework layer to reduce the impact of http flood type attacks using the PHP and Swoole languages. From their research, it is relevant to the objectives of this research with a direct focus on handling at the application layer. This insight provides several insights into this research approach in designing a mitigation strategy that can effectively handle high-volume attack patterns carried out by one valid address with a lot of request concurrency in slow patterns.

Swoole is an extension of the PHP language that offers additional capabilities used to build high-performance web and server applications [12], [13], [14]. Yang et al. [15] compared Swoole with other PHP asynchronous frameworks, showing its efficiency in handling connections and request management in concurrent environments so that this technology can be used as a tool to build a basic security system at the framework layer that has the capabilities and features to access higher OS layers. In their research regarding the characteristics of Swoole's performance, it was used as a basis used as a reference for developing a flow mitigation and mitigation system for http flood attacks using the event-driven feature because in the trial it could handle IP connections when there were requests and handle high request traffic.

This research, based on references in the previous sub-chapter, specifically focuses on a model to reduce the impact of HTTP Flood type DDoS using the Swoole extension which provides additional features to PHP programming in building its own server with event-driven capabilities and lower-level OSI layer access. In previous work, handling DDoS focused on a separate layer with different technology to focus on the network layer in filtering connections and monitoring anomalies related to packets sent at layer 4, detecting attack patterns, and increasing the efficiency of traffic distributed to servers [16], [17]. In this research, the focus is on the mechanism for reducing the impact of flood type DDoS attacks implemented in a framework called Gemstone-subprocess using Swoole technology. By utilizing Swoole technology which has the capability to directly filter TCP connections at Layer 4 using event-driven OnConnect and several other functions, you can reduce the use of other technologies to build your own firewall at the application layer level [18]. Then Swoole provides an asynchronous and concurrency environment, so it is very relevant to the approach in this research which combines several elements such as connection filtering, queuing systems, and real-time connection management to create a reduction layer to reduce the impact of DDoS Flood Http. The algorithm used in this research is to filter TCP connections which implements a queuing system by accepting 67% of connections to be processed immediately at the next layer and 33% will be queued for 2 seconds as the queue time limit before the connection is closed. Then, each connected IP address will have concurrency limitations to avoid exploitation attacks by utilizing 1 IP to carry out several concurrency processes which can drain resources on the server.

To describe the measurement method in the previous paragraph using metrics such as measuring average latency, request handling capacity, error rate on sockets. This research aims to provide a solution to simplify the DDoS mitigation package at the framework layer which aims to minimize the use of tools to overcome medium to low scale DDoS.

II.METHOD

In this section, the design of the framework that will be built using the gemstone architecture will be explained. Where the system design process starts from creating a folder architecture, gemstone process flow, framework execution model flow, and Xgen query feature flow.

1. Folder Architecture Design

Figure 1 is a design of the proposed framework's folder architecture consisting of devise, public, setup, tempSTR, and migration folders.

Devise, is a directory that contains a framework for building a website. The Basedata sub-directory is used to handle database-related data queries, Display functions as a directory to store codes that will be displayed on the user side, Service is a directory that contains code logic to integrate, manage data, and transmit data to other program classes as needed. Public is a directory that functions to store utility files such as css, images, JavaScript code. In addition, the public folder also functions as a directory that stores special files to publish the code to the user [19].

Setup is a directory that contains the bootstrap program, server config, database config, logger config, and command runner config. TempSTR is a directory that functions as a container for storing files on the submission form in image and document formats. Migration contains the table configuration of the database used for migration.

FIGURE 1. Folder architecture design in framework.

2. Gamestone process flow

In figure 2 is a gemstone sub-process that performs 3 stages of the process on the onConnect event that filters and limits connections from a tcp/ip that must not exceed a certain threshold and if a connection exceeds the number of concurrencies set then the connection according to the first threshold will be accepted and forwarded to the onRequest event while those that do not will have their connections closed. The algorithm applied in this study filters TCP connections by implementing a queuing mechanism, where 67% of incoming connections are forwarded directly for immediate processing at the next layer, while the remaining 33% are temporarily placed in a queue for up to 2 seconds before being terminated if not processed. In the Onrequest event gemstone performs several stages such as rate limiter, cors validation, csrf token validation on post/put requests and the last is to sanitize the data so that when the data is called in the service layer the data is ready to use.

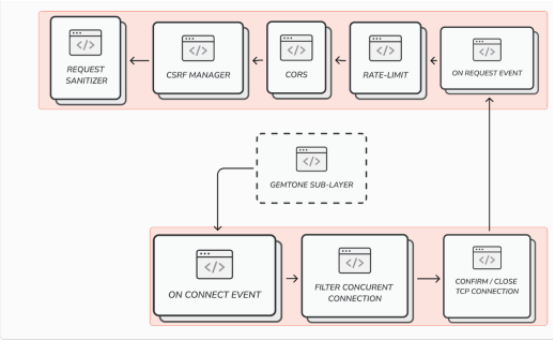

FIGURE 2. Gamestone process flow.

3. Framework execution model flow

Figure 3 is an abstraction architecture on the framework that will start from stage 1, namely after the server is initiated and receives a request, it will be forwarded to the onConnect event that controls the tcp connection from the ip request that will be filtered concurrently on each user so as not to exceed a certain limit. Then valid connections will be forwarded to the onRequest event for the gemstone stage 2 process, namely validating the csrf token, cors, rate-limiting. Then continue with the routing process involving the dispatch router and middleware process to validate requests from clients that will reach the service layer with data that has been sanitized by default to clean tags or scripts that have the potential to be malicious.

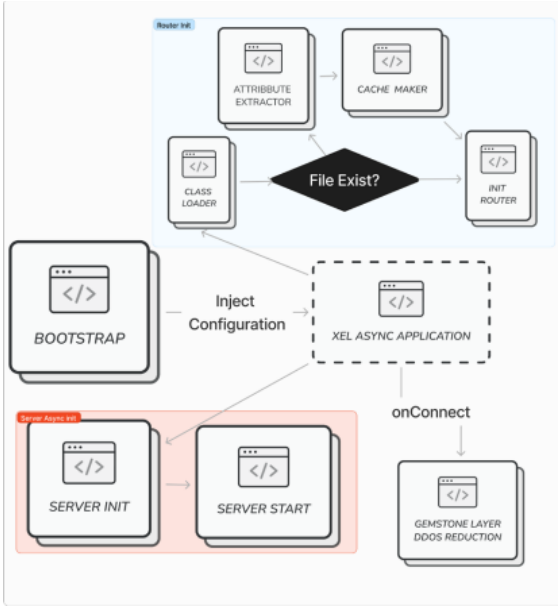

FIGURE 3. Framework execution model flow.

4. Xgen Query Feature Flowchart

Figure 4 is a boot model of the Xgen feature which in its implementation will start from creating a pool connection on the manager that runs the instance and forwarded to the gemstone layer which will eventually be used by the service layer. In Xgen query which instance will be filled with data, then the data will be bound first and prepared in the prepared statement for later the function will be triggered to manipulate table data in the database.

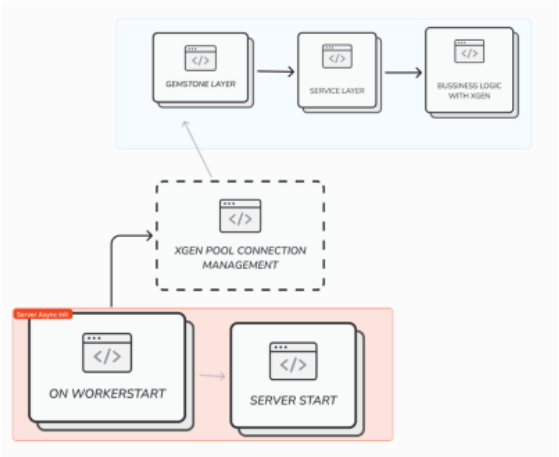

FIGURE 4. Xgen Query Feature Flowchart.

III.RESULT AND DISCUSSION

This section will explain the framework implementation process based on the system design created. At this stage, the framework is prepared with the aim of being used and meeting the targets related to the design that has been made. This design will involve several processes related to the preparation of folder structures, installation of certain libraries that have been specifically designed with the Framework environment and adding setups to be integrated as bootstrap classes as entry points for the framework system in running its functions.

1. Implementation of folder structure design

In figure 5 is the base folder structure that has been implemented and there are slight changes in the structure to reduce the number of folders to be more efficient. The first is removed is tempSTR which is used as a place to store files merged to the public and can freely determine its storage, the removal of the router folder because the use of the router has been implemented dynamically, and the last is the addition of a writeable folder used to store logs, static cache that can only be written by the framework.

```
root@DESKTOP-HP8OI7D:/home/xel/app-test# ls -la
total 144
drwxr-xr-x  9 xel root  4096 Apr 23 23:07 .
drwxr-x---  7 xel xel   4096 Apr 23 00:19 ..
-rw-r--r--  1 xel root   144 Apr 23 00:03 .env
-rw-r--r--  1 xel root    27 Apr 23 00:03 .gitignore
drwxr-xr-x  2 xel xel   4096 Apr 23 23:09 .idea
-rw-r--r--  1 xel root  1053 Apr 23 00:03 composer.json
-rw-r--r--  1 xel root 86304 Apr 23 00:19 composer.lock
drwxr-xr-x  7 xel root  4096 Apr 23 00:03 devise
drwxr-xr-x  2 xel root  4096 Apr 23 00:03 migration
drwxr-xr-x  3 xel root  4096 Apr 23 00:03 public
drwxr-xr-x  6 xel root  4096 Apr 23 00:03 setup
-rwxr-xr-x  1 xel root  1116 Apr 23 00:03 tailwind.config.js
drwxr-xr-x 19 xel root  4096 Apr 23 00:19 vendor
drwxr-xr-x  5 xel root  4096 Apr 23 00:03 writeable
-rw-r--r--  1 xel root   349 Apr 23 00:03 xel
root@DESKTOP-HP8OI7D:/home/xel/app-test#
```

FIGURE 5. Implementation of folder structure design.

2. Server Configuration Setup

In Figure 6, show about the server configuration. In the additional setup, by default, it activates worker Num which refers to the number of threads that will be run by the framework according to the number provided by the computer specifications, task_worker_num relates to the number of workers that are isolated and dedicated to running tasks asynchronously, document_root as the root folder identification of the framework, and enable_static handler which is used to activate the server to render static content.

```php
<?php
return [
    'api_server' => [
        'host' => 'http://localhost',
        'port' => 9501,
        'mode' => 1,
        'options' => [
            "enable_static_handler" => true,
            "document_root" => dirname(__DIR__, 2),
            'worker_num' => swoole_cpu_num(),
//            'task_worker_num'=>16,
            'task_enable_coroutine' => true, // optional to turn on task coroutine support
//            'daemonize' => 1,
//            'http_gzip_level' => 9,

            /**Enable it when use mode 2*/
            'dispatch_mode' => 1,

            /**Optional Config*/

            'open_tcp_nodelay'      => true,
            'reload_async'          => true,
            'max_wait_time'         => 60,
            'enable_reuse_port'     => true,
            'enable_coroutine'      => true,
            'http_compression'      => true,
            'buffer_output_size'    => swoole_cpu_num() * 1024 * 1024,
        ],
    ],
];
```

FIGURE 6. Implementation of configuration.

3. Database Configuration Setup

In Figure 7 is the configuration of the database using PHP Document Object and the main driver recommended to use is MySQL. Unlike most php frameworks, the database connection feature in PHP by default can use pool mode to define more than 1 connection and increase the effectiveness of performing high-intensity query activities without sacrificing performance. The model from the pool mode will not be destroyed but will be returned to the pool so that it can be reused by other processes without having to create a connection from scratch.

```php
<?php
return [
    'driver' => 'mysql',
    'host' => 'localhost',
    'charset' => 'utf8mb4',
    'username' => 'root',
    'password' => 'Todokana1ko!',
    'dbname' => 'sample',
    'options' =>[
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
        PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
    ],

    'pool' => 10,
    'poolMode' => true,

    'migration' => "\\Xel\\Migration",
    'pathMigration' => __DIR__."/../../migration"
];
```

FIGURE 7. Implementation of database configuration.

4. Gemstone configuration setup

In figure 8 is the default configuration of the Gemstone sub process which starts with defining rate-limiting and Ip filter using the sliding size window algorithm which functions to limit requests per ip address at a certain interval. In default mode, it will use request restrictions only at a certain interval and renew the access quota at the next interval and the filter mode will block the ip when an ip uses a request more than the second threshold and at the next interval the ip will not be able to be used to access the destination address/resource. The second is the CORS origin restriction which functions to filter related addresses that can access the server, headers, cookies and requests that are allowed to access the server. and the third is the activation of CSRF protection.

```
/*
 * DOS Protection
 *
"gemstone_limiter" => [
    "condition" => false, // default // ip_based_limit
    "max_token" => 100, // max token fill in bucket
    "interval" => 60, // "in second"

    // ? additional for DDOS to block service when pass the second threshold
    // ? to disable it leave black array, and it will use regular limiter
    // ? if already used and need to disable it, please clear the loaded black listed IP on Gemstone_log
   // "block_ip" => [],
    "block_ip" => [200, __DIR__."/../../../writeable/Gemstone_log/black_list.php"],
], // implemented and underdeveloped

/*
 * Secure Data Protection (CORS)
 *
"securePost" => [
    'condition' => true,
    'cors' => [
        'allowOrigin' => "http://localhost",
        'allowMethods' => ['GET', 'POST', 'PUT', 'DELETE', 'OPTIONS'],
        'allowHeaders' => ['Content-Type', 'Authorization', 'Origin', 'X-Requested-With,', 'X-CSRF-Token'],
        'maxAge' => 86400,
        'allowCredentials' => true,
    ],
],
```

FIGURE 8. Implementation of gemstone configuration.

5. Implementation in website

In Figure 9 is a simulation page of the applied DDoS attack which has 2 sections, namely the description section which explains the DDoS type of flood http which targets flooding http requests on a specific destination to drain server resources until the server collapses and can no longer handle requests. And in the second section is a DDoS trial with gemstone DDoS reduction protection which will be compared with technology without DDoS.

FIGURE 9. Implementation of website for DDoS instruction.

In this section, several test scenarios will be explained that will be carried out on the framework being developed. Starting from framework latency testing, framework reliability testing against DDoS attacks, and framework testing against Cross-site scripting (XSS) attacks.

A. Gemstone framework analysis with other frameworks

In the benchmark test, 2 types of tests were carried out, namely restful API testing that returns "hello world" in Json form and returns query values containing id, name, email data in Json form on the proposed Framework and Xpress JS. The comparison parameters taken are the average latency per framework, and average requests per second. Figure 10 shows the average latency between the proposed frameworks compared to Express JS. Where the results show that from 10 trials, the latency of the proposed framework has a lower value or a faster time process. Figure 11 shows the average request from each framework. Which results also show the number of requests from the proposed framework has a higher value compared to Express JS.

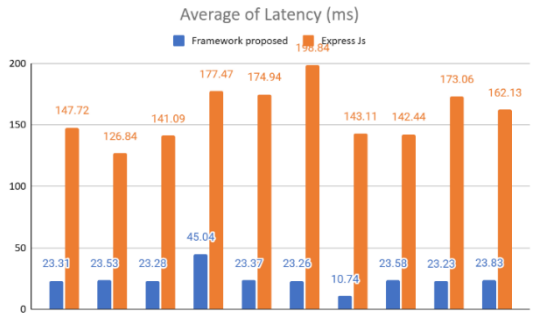

FIGURE 10. Average of latency.

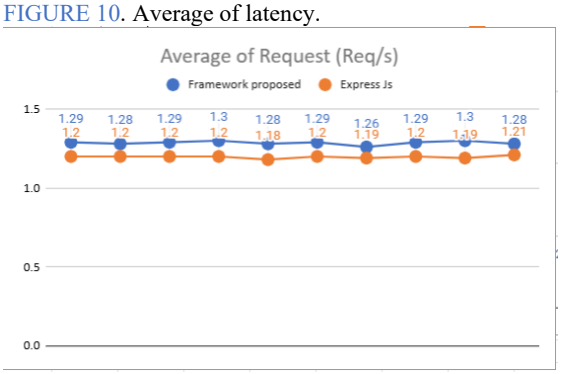

FIGURE 11. Average of Request.

B. Testing against DDoS attacks

Based on the mechanism testing at the beginning of chapter 3, get the result data in table 1 about the comparison of average latency, average requests per second, and total requests in 5 minutes during the attack. Based on the performance results, the empty server shows better performance to handle throughput with low latency, followed by the use of gemstone with non-blocking mode which results below the empty server because of the additional layer to do filter connections and the third test has completed the goal that will block connections continuously when tcp/ip clients have more concurrent requests compared to the threshold. In table 2 provide additional results for socket error performance after the attack which shows about handling progress in the three mechanism tests, namely

- Without Gemstone Layer, the system doesn't have any socket errors but has high number of timeout indicating the server get overwhelmed during the attack.
- When using Gemstone Layer without blocking mode, there is have significant reduction about timeout with tradeoff several error on read and write process which is indicated this layer can handle properly about flood attack.
- And finally, when use full featured Gemstone sub-process including blocking mode, all timeout being eliminated and have trade of increasing high on read error and several error

on write which is with this pattern the layer will take aggressive way to minimize impact of DDoS with rejecting client connection when their IP get listed on acl in connect state.

TABLE 1. Average Performance Comparison

Method	*Avg Latency*	*Avg RPS*	*Total Request*
Not use Gemstone Layer	703.2ms	22.23	11921
Use Gemstone Layer with disbale blocking mode	871.8ms	17.12	11850
Use Gemstone Layer With Limit Concurent and Blocking mode	Nan%	Nan%	Nan%

TABLE 2. Socket Error Comparison

Method	*Error Read*	*Error Write*	*Timeout*
Not use Gemstone Layer	0	0	10956
Use Gemstone Layer with disbale blocking mode	553	636	1323
Use Gemstone Layer With Limit Concurent and Blocking mode	12489	957	0

Figure 12 shows the analysis of the Gemstone framework's performance in handling DDoS compared to previous research using native PHP. The analysis metrics used were latency, including average latency, standard deviation latency, and maximum latency. The measurements were performed using 1,000 requests and three threads. The results show that the Gemstone framework has lower latency analysis compared to subsequent research using native PHP.

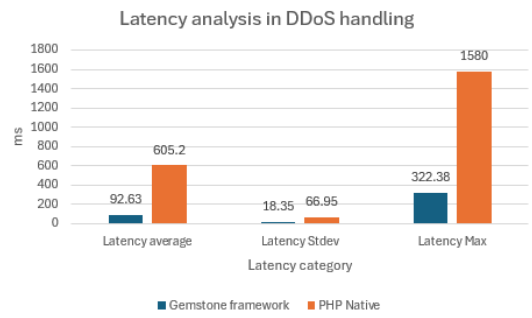

FIGURE 12. Latency analysis in DDoS handling in Gemstone framework and PHP Native.

C. Testing against Cross-site scripting (XSS) attacks

In the XSS attack testing scheme using the stored xss injection method which is carried out on the input form and request url param encoded with UTF-8. Figure 13 is the result of testing the XSS attack via uri param. Where in the test on the uri param using the server port: 9501 targeting the url / crud / blog / {param} which is filled with UTF 8 encoding of the js script.

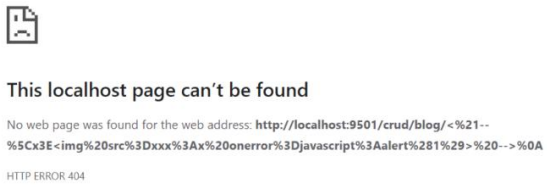

FIGURE 13. XSS attack testing via url param.

Figure 14 is an illustration of XSS injection on a form. And figure 15 is response from testing. In the test on the form injection, the payload hook from beef is used which is injected directly into the input, then the results will be displayed on the website. The purpose of beef injection is used as a test method to spy on website activity and can be done for several attacks such as phishing.

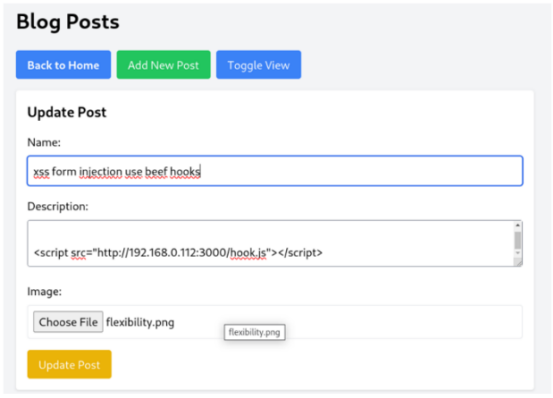

FIGURE 14. XSS Attack form injection.

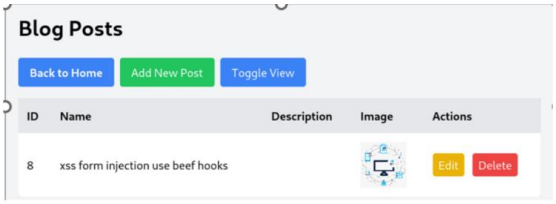

FIGURE 15. Response XSS attack.

IV. CONCLUSION

This study addresses the challenges of Distributed Denial of Service (DDoS) attacks, specifically HTTP flood attacks targeting Internet of Things (IoT) systems, which often lack robust gateway configurations to ensure adequate security and privacy. This study focuses on mitigating these attacks using the PHP Swoole framework, which is

event-based and provides access to the transport layer during the connection event (onConnect). This study uses a TCP connection filtering algorithm by implementing a simple queuing system. The results of tests conducted ranging from benchmark analysis, DDoS attack analysis, and XSS attack analysis have been carried out and compared with several other frameworks or other technologies, indicating that the Gemstone framework has better performance and can run well for monitoring IoT systems. For further research, it is recommended to conduct more experiments on the Gemstone framework with more varied scenarios to see better performance.

ACKNOWLEDGMENT

The researcher gratefully acknowledges the support of Politeknik Elektronika Negeri Surabaya, which has generously sponsored this local research through its Center for Research and Community Service.

REFERENCE

[1] P. Kumari and A. K. Jain, "A comprehensive study of DDoS attacks over IoT network and their countermeasures," *Comput Secur*, vol. 127, p. 103096, Apr. 2023, doi: 10.1016/J.COSE.2023.103096.

[2] R. Vishwakarma and A. K. Jain, "A survey of DDoS attacking techniques and defence mechanisms in the IoT network," *Telecommun Syst*, vol. 73, no. 1, pp. 3–25, Jan. 2020, doi: 10.1007/S11235-019-00599-Z/METRICS.

[3] Y. Al-Hadhrami and F. K. Hussain, "DDoS attacks in IoT networks: a comprehensive systematic literature review," *World Wide Web*, vol. 24, no. 3, pp. 971–1001, May 2021, doi: 10.1007/S11280-020-00855-2/METRICS.

[4] M. Snehi and A. Bhandari, "Vulnerability retrospection of security solutions for software-defined Cyber–Physical System against DDoS and IoT-DDoS attacks," *Comput Sci Rev*, vol. 40, p. 100371, May 2021, doi: 10.1016/J.COSREV.2021.100371.

[5] M. M. Salim, S. Rathore, and J. H. Park, "Distributed denial of service attacks and its defenses in IoT: a survey," *Journal of Supercomputing*, vol. 76, no. 7, pp. 5320–5363, Jul. 2020, doi: 10.1007/S11227-019-02945-Z/METRICS.

[6] S. Y. Diaba, M. Shafie-khah, and M. Elmusrati, "On the performance metrics for cyber-physical attack detection in smart grid," *Soft comput*, vol. 26, no. 23, pp. 13109–13118, Dec. 2022, doi: 10.1007/S00500-022-06761-1/FIGURES/10.

[7] M. M. Alshahrani, "A Secure and Intelligent Software-Defined Networking Framework for Future Smart Cities to Prevent DDoS Attack," *Applied Sciences 2023, Vol. 13, Page 9822*, vol. 13, no. 17, p. 9822, Aug. 2023, doi: 10.3390/APP13179822.

[8] A. Praseed and P. Santhi Thilagam, "DDoS attacks at the application layer: Challenges and research perspectives for safeguarding web applications," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 1, pp. 661–685, Jan. 2019, doi: 10.1109/COMST.2018.2870658.

[9] Y. Feng, J. Li, and T. Nguyen, "Application-Layer DDoS Defense with Reinforcement Learning," *2020 IEEE/ACM 28th International Symposium on Quality of Service, IWQoS 2020*, Jun. 2020, doi: 10.1109/IWQOS49365.2020.9213026.

[10] S. Black and Y. Kim, "An Overview on Detection and Prevention of Application Layer DDoS Attacks," *2022 IEEE 12th Annual Computing and Communication Workshop and Conference, CCWC 2022*, pp. 791–800, 2022, doi: 10.1109/CCWC54503.2022.9720741.

[11] F. S. Dantas Silva, E. Silva, E. P. Neto, M. Lemos, A. J. Venancio Neto, and F. Esposito, "A Taxonomy of DDoS Attack Mitigation Approaches Featured by SDN Technologies in IoT Scenarios," *Sensors 2020, Vol. 20, Page 3078*, vol. 20, no. 11, p. 3078, May 2020, doi: 10.3390/S20113078.

[12] Y. Wang, C. Chen, and W. Zhang, "Design of Smart Home Control System Based on Internet of Things," *ITNEC 2023 - IEEE 6th Information Technology, Networking, Electronic and Automation Control Conference*, pp. 1680–1683, 2023, doi: 10.1109/ITNEC56291.2023.10082440.

[13] D. Tsindeliani, Y. Povstyana, N. Lishchyna, and A. Yashchuk, "Latency Reduction in Real-time GPS tracking in Android and the Web-based GPS Monitoring System," *Proceedings of the 2022 IEEE 12th International Conference on Dependable Systems, Services and Technologies, DESSERT 2022*, 2022, doi: 10.1109/DESSERT58054.2022.10018609.

[14] J. He, "Design and Implementation of Network Teaching Platform of Ideological and Political Education in Colleges and Universities Based on Laravel Framework," pp. 1383–1388, Jun. 2023, doi: 10.2991/978-94-6463-172-2_148.

[15] J. Yang, Z. Shan, and Z. Chen, "Research and practice of swoole asynchronous multithreading design method," *2018 IEEE 4th International Conference on Computer and Communications, ICCC 2018*, pp. 2163–2169, Dec. 2018, doi: 10.1109/COMPCOMM.2018.8780934.

[16] A. Praseed and P. Santhi Thilagam, "DDoS attacks at the application layer: Challenges and research perspectives for safeguarding web applications," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 1, pp. 661–685, Jan. 2019, doi: 10.1109/COMST.2018.2870658.

[17] S. Kaur, A. K. Sandhu, and A. Bhandari, "Investigation of application layer DDoS attacks in legacy and software-defined networks: A comprehensive review," *International Journal of Information Security 2023 22:6*, vol. 22, no. 6, pp. 1949–1988, Aug. 2023, doi: 10.1007/S10207-023-00728-5.

[18] M. Robihul Mufid *et al.*, "Design MicroServer Framework Library with Swoole for Real-time Application Development," *Jurnal Teknologi Informasi dan Terapan (J-TIT)*, vol. 11, no. 2, pp. 2580–2291, Dec. 2024, doi: 10.25047/JTIT.V11I2.5673.

[19] M. R. Mufid, Y. Pratama, A. Fariza, and S. Mawaddah, "Modification MVC Architecture in PHP using Basedata Service Display Pattern," pp. 4–18, Feb. 2024, doi: 10.2991/978-94-6463-364-1_2.

Mohammad Robihul Mufid is an academic and lecturer at the Surabaya State Electronics Polytechnic (PENS) specializing in control technology, signal processing, and machine learning. I completed my formal education in informatics and computer engineering, focusing on control systems and intelligent technology. I focus on the development of science, especially in the integration of artificial intelligence into control and automation systems. Outside of academics, I am active in various research and technology development projects that support industrial progress, such as automation and artificial intelligence applications in the manufacturing sector.

Yogi Pratama, is an academic affiliated with the Surabaya State Electronics Polytechnic (PENS), one of the best vocational education institutions in Indonesia. He is actively involved in teaching, research, and technology development, especially in the fields of informatics and electronics. PENS is known as a center of excellence in technology, including informatics engineering, electrical engineering, and multimedia technology, where its lecturers contribute significantly to various innovations and national-level research projects..

Arna Fariza is a lecturer and researcher from Gresik who successfully earned a doctorate from the Sepuluh Nopember Institute of Technology (ITS). Her research focuses on the

development of an automatic age estimation system based on panoramic radiography for forensic odontology. This system utilizes dental images to identify a person's age with an accuracy of up to 60.93%. The research aims to help the identification process in situations such as mass accidents or natural disasters, as well as contribute to the field of forensic medicine and rescue teams..

Saniyatul Mawaddah, is a lecturer and researcher in the field of Informatics Engineering at the Surabaya State Electronic Polytechnic (PENS), especially at the Lamongan Regency Campus. She has made significant contributions to technology-based research, one of which is on the classification of rhizome images using the Support Vector Machine (SVM) method, which was published in an international scientific conference in 2022..

Much Chafid is a lecturer at the Politeknik Elektronika Negeri Surabaya (PENS), Indonesia, actively involved in the fields of Information and Communication Technology, particularly in network systems and educational technology implementation. He has contributed to academic and applied research, including the use of cloud-based Virtual Private Network (VPN) systems in educational environments and the enhancement of ICT infrastructure in vocational schools. His work often focuses on integrating technology into school networks to improve digital literacy and connectivity, especially in rural or developing regions. In addition to his academic duties, Much Chafid has participated in various community outreach and student recruitment initiatives at PENS, including activities at its Lamongan campus extension.

Agus Wibowo, is a lecturer at the Surabaya State Electronic Polytechnic (PENS) who faculty member in the D3 Multimedia Broadcasting Technology program at Politeknik Elektronika Negeri Surabaya (PENS), where he specializes in digital image processing and multimedia systems. His contributions extend beyond teaching; in 2024 he participated in a community service project that implemented a web-based fish marketing information system in Rejosari Village, Lamongan

.