Lightweight, Low-Rate Denial-of-Service Attack Prevention and Control Program for IoT Devices

Chi-Che Wu[1], Rung-Shiang Cheng[2], Chiung-Wen Hsu[3], Li-Wei Wu[4]

[1] Department of Electrical Engineering, National Kaohsiung University of Sciences and Technology, Taiwan
[2] Department of Information Technology, Overseas Chinese University, Taiwan
[3] Department of Information Management, National Kaohsiung University of Sciences and Technology, Taiwan
[4] Department of Information Business, Tunghai University, Taiwan
1101405110@gm.kusa.edu.tw, rscheng@mail.ksu.edu.tw, sandrahsu33@kuas.edu.tw, lwwu@thu.edu.tw

Abstract

As information technology has become more advanced, the Internet of things (IoT) has evolved from being a mere concept to becoming a part of everyday life. IoT-based home appliance applications have matured, and numerous relevant software programs have been made commercially available. Therefore, IoT-created security issues have become an issue that must be addressed.

Hypertext transfer protocol (HTTP) transmission is one of the main transmission methods used in IoT-based communication. As HTTP evolves from the original HTTP/1.0 to the current HTTP/2, transmission efficiency and security have undergone considerable improvements. However, despite these improvements, HTTP/2 remains exposed to various risks, one of the most common of which is low-rate denial-of-service attacks (DoS attacks). Using this type of attack, hackers can paralyze target hosts. This hinders the target hosts' ability to respond promptly, causing substantial damage to their systems.

Although DoS attacks are one of the most commonly used methods by hackers to attack target hosts, most mainframe computers are equipped with excellent DoS attack prevention and control programs. Nevertheless, most IoT devices do not have high computing power and are thus prone to DoS attacks. Therefore, this study examined the feasibility of using a lightweight, low-rate DoS attack prevention and control program in IoT devices with low computing power. The objective is to enable these devices to prevent and control DoS attacks.

Keywords: HTTP/2, Denial-of-service attacks, Low-rate denial-of-service attacks, Information security

1 Introduction

In recent years, the Internet of things (IoT) has been widely used in smart homes and in the field of industrial control. IoT embodies the concept of creating a network in which everything is connected. For users, the IoT provides a novel way of interacting with devices. The interaction process includes collecting relevant data; for example, electronic devices may be programmed according to user instructions to turn on automatically right before the user returning home and turn off automatically when the user leaves the house. The use of IoT in the field of industrial control is even more prevalent than that in smart homes. For instance, smart factories add numerous sensors to relevant equipment. When the equipment malfunctions, the networking devices send warning messages through wireless transmission to inform users of the abnormal situation, achieving early disaster prevention [3-4].

In general, devices connected to an IoT-based network contain a network component with data transmission capability. In addition, several sensors that have dissimilar goals or purposes are installed. These sensors are comparable to human senses and can be used to collect relevant data in surrounding environments. For example, in the field of debris flow detection, researchers can install soil moisture sensors in mountainous areas that are prone to debris flows; when the sensors detect soil moisture saturation and that critical values are being reached, warning messages are sent to terminal devices through wireless transmission, after which the terminal devices inform relevant personnel through their networking capability [1-2].

In addition to the aforementioned conventional IoT applications, Google introduced an IoT-related program called "Physical Web" in 2014 [15]. The main concept of the program was to modify the way in which the next IoT generation would be accessed, transforming the conventional IoT access method from Internet protocol-based (IP-based) to uniform resource locator-based (URL-based), creating a novel approach called the Web of Things (WoT). The WoT primarily comprises IoT and web-enabled technologies. All physical objects can be operated through a string of

[*] Corresponding Author: Rung-Shiang Cheng; E-mail: rscheng@mail.ksu.edu.tw

DOI: 10.3966/160792642019052003020

URLs and data can be accessed and used by RESTFul (representational state transfer web services). Users do not need to install additional applications to operate the physical objects; all they must know are the URLs corresponding to the physical objects [5].

Although IoT and WoT devices have matured, their power consumption needs to be considered during their initial design process, which has led to them exhibiting a performance that is far inferior to that of conventional computers (Figure 1). Thus, when these devices are exposed to denial-of-service (DoS) attacks, the conventional defense mechanisms employed by these devices are unable to effectively detect and block attacks. Accordingly, designing a lightweight protection program for IoT devices is crucial.

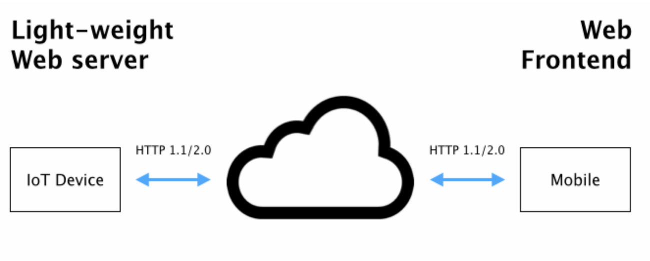


Figure 1. IOT and web frontend

Because webpage transmission exhibits cross-platform-like characteristics, it is used in numerous IoT applications. Similarly, the HTTP communication protocol is used as the IoT transmission protocol when transmitting data. At present, HTTP/1.1 is generally used as the HTTP communication protocol. However, this communication protocol was introduced in 1999 and has been in use for more than 16 years. Because network structures and related applications have changed dramatically during this period, comparisons between the older HTTP/1.1 and the upgraded HTTP/2.0 must be made.

Since the introduction of HTTP/2 in May 2015, related studies have mainly compared differences in the performance of the old and new communication protocols, whereas few studies have investigated the security of these protocols. Because Internet attacks are continuing to evolve, the present study explored information security in the HTTP/2 network protocol.

Studies on low-rate DoS attacks against HTTP/2 services have demonstrated that compared with the plaintext transmission method employed by HTTP/1.1, the binary transmission method used by HTTP/2 requires servers to complete more calculations to support related services. Thus, HTTP/2 is more prone to DoS attacks. The main goal of such attacks is to substantially consume server resources and prevent access by other users [1].

Because the aforementioned security risks against HTTP/2 are different from those created by previous DoS attacks, this study proposed a set of defense mechanisms and simulated real environments to verify whether the proposed defense approach can reduce the security threats on users.

2 Literature Analysis

2.1 From the IoT to the WoT

The conventional IoT involves the use of numerous sensors that transmit related data to a cloud platform through a network device. Users who need to control or access relevant data can do so by connecting to the cloud platform and accessing inquired data. The Physical Web program introduced by Google in 2014 specified that all sensors and devices have URLs, which are the basis of connection in the web environment; these URLs are connected to physical devices to allow users to quickly control and use the devices. Each of these devices discloses its communication method by using the RESTFul application programming interface defined by the device itself. This prevents users from needing to control the devices via a centralized machine. Therefore, users can develop related applications more quickly. The use and control of these devices are similar to those of hyperlinks.

HTTP and RESTFul are design concepts that allow users to easily assign URLs to physical objects. For example, assuming that a device has the control URL http://device1.wot.kuas.edu.tw/ and a light sensing module, then users can use [GET] http://demo.com/light to obtain relevant sensor data [15].

2.2 From the IoT to the WoT

HTTP is currently the most prevalent web-based protocol on the Internet; its primary purpose is to enable servers to respond to user requests [2].

2.2.1 HTTP/1.0

HTTP/0.9 is the original version of HTTP, wherein users submit basic requests and servers respond using simple semantics. HTTP/1.0 improved the HTTP protocol and allowed messages to be transmitted in the MIME format.

In the HTTP/1.0 operation process, users request resources from servers (during which transmission control protocols (TCPs) are established) and servers fulfill the request by returning the data demanded by the users, ending the connection process. The servers do not engage in follow-up tracking or submit further record requests. Users who request additional webpage resources must reconnect to the network and repeat the aforementioned process (Figure 2) [3].

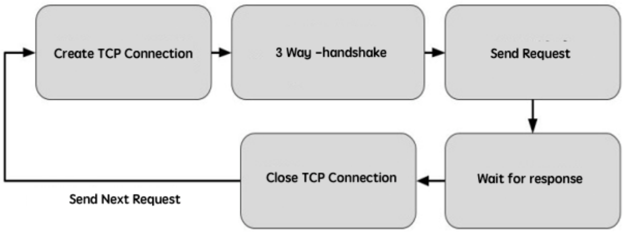


Figure 2. The HTTP/1.0 operation process

2.2.2 HTTP/1.1

HTTP/1.1 has become the most widely used Internet protocol on the Internet. The main objectives of the protocol are to establish strict guidelines and further improve on HTTP/1.0. The main functions of HTTP/1.1 are as follows:

a. Default HTTP Persistent Connection

For HTTP/1.0 to collect any resources from the server, an independent TCP connection must first be established. However, this increases the burden on the server and easily leads to network congestion. Therefore, HTTP/1.1 mandates the use of a default HTTP persistent connection and allows connection reuse, which reduces the burden on the server considerably and decreases the time required to establish a TCP connection (Figure 3).

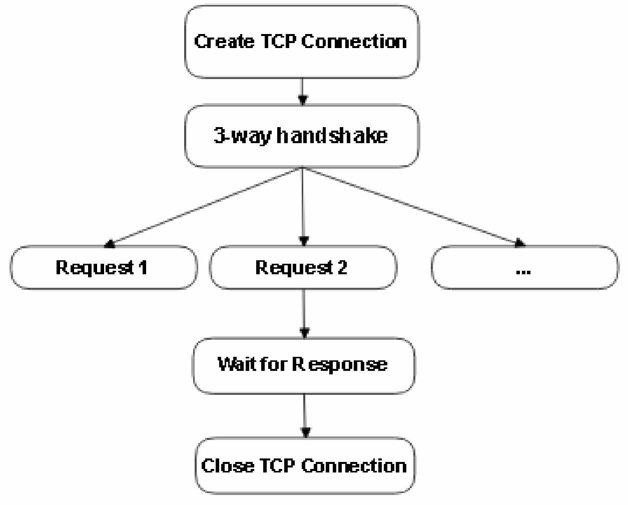


Figure 3. HTTP/1.1 operation process

b. Pipelining

HTTP/1.1 pipelining allows multiple requests to be submitted at once. However, the servers must respond to these requests according to the order in which the requests were submitted (Figure 4), which reduces response waiting time. However, this mechanism creates a head-of-line (HOL) blocking problem because requests that take servers substantial time to process cause delays to responses to subsequent requests. To solve this problem, 6-8 TCP connections are currently built per browser to process submitted requests.

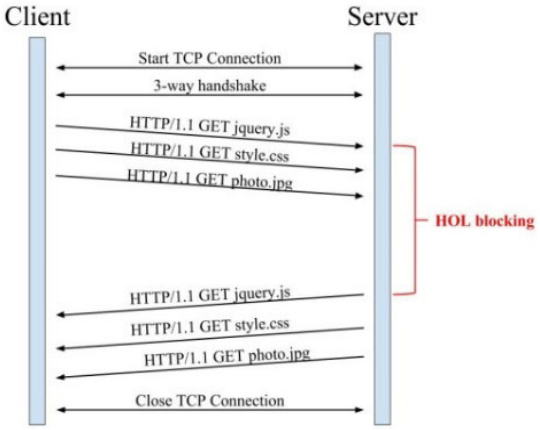


Figure 4. HTTP/1.1 pipelining

HTTP/1.1 also features other functions such as a buffer mechanism, domain name mechanism, error hints, and expansibility. Because of these features and its rigorous guidelines, HTTP/1.1 remains in use after 16 years [4].

2.2.3 HTTP/2

The HTTP working group used Speedy (SPDY) as a basis on which to successfully develop HTTP/2. In May 2015, they officially released two documents (RFC 7540 and RFC 7541) that presented the second major version of the HTTP protocol. This protocol improves on the potential problems in HTTP/1.1, decreases webpage loading times, increases webpage transmission speed, and lowers webpage processing time.

The HTTP/2 request process differs from that of HTTP/1.1. For instance, HTTP/1.1 establishes 6-8 TCP connections to speed up the inquiry time, whereas HTTP/2 establishes only one TCP connection so as to reduce the burden on servers. After a TCP connection is established, browsers can establish multiple noninterfering streams and use the smallest unit frame to allocate the request content, facilitating browser-server communications (Figure 5) [5-10].

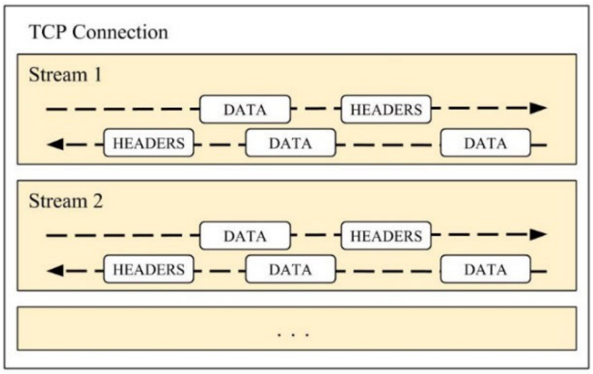
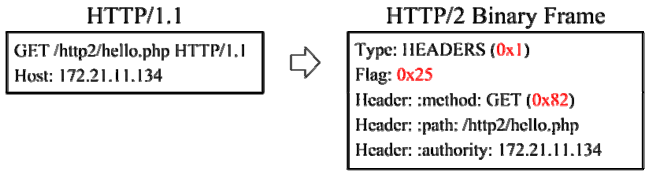


Figure 5. HTTP/2 request submission process

The main functions of HTTP/2 are as follows:

a. Binary Frame

Requests submitted by webpages were previously transmitted in plaintext, which created large network packets, hindering transmission speed. Thus, HTTP/2 changed conventional requests into binary frames and encoded and compressed plaintext in requests (Figure 6) to reduce the size of the network packets, enhancing transmission speed.



HTTP/1.1

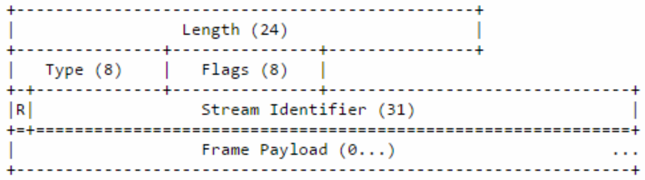
```
GET /http2/hello.php HTTP/1.1
Host: 172.21.11.134
```

HTTP/2 Binary Frame

```
Type: HEADERS (0x1)
Flag: 0x25
Header: :method: GET (0x82)
Header: :path: /http2/hello.php
Header: :authority: 172.21.11.134
```

Figure 6. Binary frame

RFC 7540 states that in each binary frame, the length is expressed as a 3-byte fields, type as a 1-byte fields, flag as a 1-byte fields, reserved word as a 1-bit field, and stream identifier as a 31-bit fields. The frame payload requests data according to where the length is placed (Figure 7). RFC 7540 also specifies the 10 different frame types to be used (Table 1), of which "headers" and "data" are the most common; these two functions correspond to the "header" and "body" functions of HTTP/1.1 [5-9].



```
+-----------------------------------------------+
|                 Length (24)                   |
+---------------+---------------+---------------+
|   Type (8)    |   Flags (8)   |
+-+-------------+---------------+-------------------------------+
|R|                 Stream Identifier (31)                      |
+=+=============================================================+
|                   Frame Payload (0...)                      ...
+---------------------------------------------------------------+
```

Figure 7. HTTP/2 frame layout

Table 1. HTTP/2 Frame

Binary Frame	Type	Description
DATA	0x0	Transmits the body of HTTP/1.1 requests and responses
HEADERS	0x1	Creates a stream in which the header contains header block fragments
PRIORITY	0x2	Prioritizes or reprioritizes resources
RST	0x3	Notifies that a stream is allowed to terminate immediately
STREAM	0x4	Sets the configuration data specifying how two endpoints are to communicate
SETTING	0x5	Represents the building of a stream and promises that the referenced resources will be provided
PUSH	0x6	Measures the minimum submission-response time and determines whether the TCP connection is still working
PROMISE	0x7	Stops the connection when the server finds a serious error with the request and when the idle time is excessively long
PING	0x8	Controls flow
GOAWAY	0x9	Connects a series of header block fragments

b. Multiplexing

Although HTTP/1.1 pipelining enables users to send multiple requests, it is prone to HOL blocking. By contrast, HTTP/2 rebuilds the pipeline, creates multiple streams according to the number of requests received, and adds a corresponding stream ID to every request submitted and response issued (Figure 8). These steps prevent requests that will take a long time to process from affecting when other requests are processed, solving the HOL blocking problem and confirming the effectiveness of the multiplexing function [8].

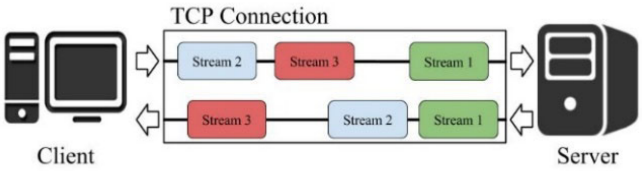


Figure 8. Multiplexing procedure

c. Stream Prioritization

RFC7540 explains that all streams can be dependent on other streams. After a server processes an "independent" stream (i.e., a stream that another stream is dependent on), resources are reallocated to the "dependent" stream (Figure 9). During the transmission process, users can thus request to download the most important content first to avoid data congestion [8].

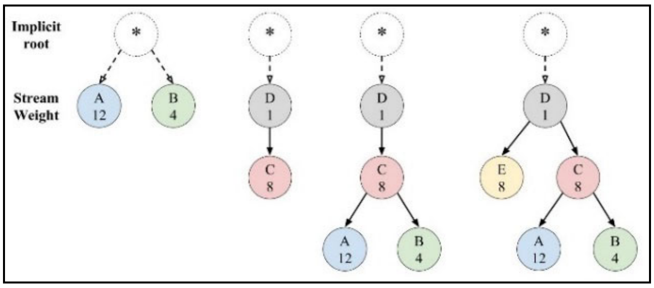


Figure 9. Stream prioritization diagram

d. Server Push

Conventionally, when webpage requests are submitted, browsers analyze the responses received from servers. When the browsers find that they require additional resources to display the webpages properly, they send further requests to the servers to obtain these resources. This leads to an increase in the time expended due to the transmission of network packets back and forth. HTTP/2 introduces the server push function, which allows servers to automatically "push" the additional resources needed by webpages to users, saving the time spent submitting requests and speeding up a webpage's display time (Figure 10) [8, 10].

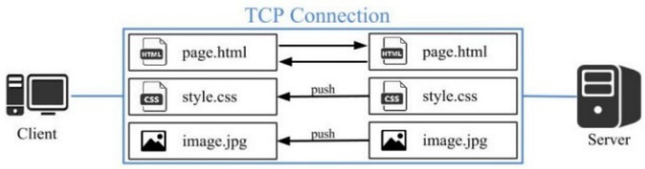


Figure 10. Server push process

e. Header Compression

When users submit multiple requests, the headers of the requests occasionally contain repeated information. Therefore, HTTP/2 compresses and stores repeated information (Figure 11); when repeated information is found in subsequent requests, different header information is sent, reducing the network packet size and enhancing the transmission speed [9].

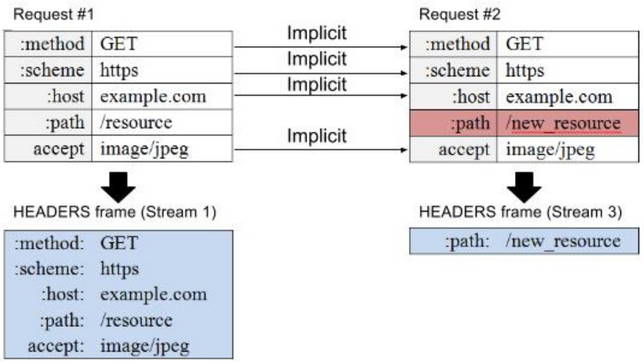


Figure 11. Header compression process

2.3 DoS Attacks

DoS attacks are cyber-attacks aimed at depleting the network or system resources of servers, causing the servers to temporarily suspend or terminate their services and preventing users from using the services [11].

Low-rate DoS attacks are a variation of DoS attacks; they attack by continuously sending a small number of network packets to attack server response times or buffer zones, causing depletion of server resources, resulting in service termination [12-14].

A study on low-rate DoS attacks on HTTP/2 services [1] confirmed that HTTP/2 security is at risk of low-rate DoS attacks. In such attacks, a virtual host using a type 1 ping and WINDOW_UPDATE frame defined by HTTP/2 attacks the virtual server. In the experiment of the aforementioned study, the degree of CPU depletion, size of the network packets received per second, and number of network packets received per second were used as a basis for assessing low-rate DoS attacks [15].

3 System Framework and Design

This study designed lightweight DoS-attack prevention and control programs for IoT devices that support WoT functions. Because RESTFul is the primary method for facilitating communication between devices, this study focused on designing a program that protects HTTP from low-rate DoS attacks. In addition, related power consumption requirements were considered to ensure that the program can be used for long periods of time under battery power.

HTTP/2 is the latest version of HTTP. Compared with HTTP/1.1, it has superior transmission capacity and lower power consumption. However, HTTP/2 is prone to low-rate DoS attacks. Thus, this study designed a defense mechanism in which the server firewall records the frames requested by users within a set time period (10 and 20 ms in this study) and identifies whether the frames are repeats and thereby pose a risk of a low-rate DoS attack. If the two criteria(Send packets Less than10ms or 20ms) are met, the firewall initiates a filtering process (Figure 12), which reduces the impact of the attacks on other users.

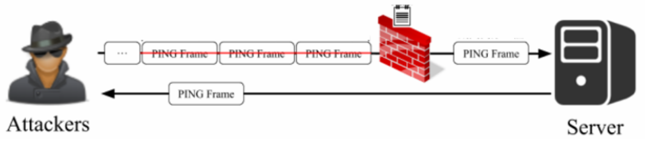


Figure 12. Defense procedure

This study used a Raspberry Pi (Figure 13) to perform a simulation analysis. In addition, the electric current detection method was employed to calculate whether there were substantial differences in the power consumption situation prior to and after the introduction of the prevention and control method. The results were used to verify whether the prevention and control program is feasible for physical web devices.

Figure 13. Raspberry Pi

Two experiments were performed. The objective of the first experiment was to confirm whether HTTP/2 is actually prone to real security threats and whether such threats can affect legitimate users. The present study divided the webpages viewed by legitimate users into five categories: “simple, static webpages,” “medium-performance webpages,” “high-performance webpages” “image download webpages,” and “new query-based database webpages.” Next, simulations were performed in which 0, 1, 5, 10, and 15 attackers sent ping frames to the HTTP/2 (Table 2) to investigate the effect of the number of attackers on legitimate users' usage experiences.

Table 2. Webpage and content

Web pages type	Web page content
Simple	A blank page contains some simple text (ex:Hello World)
Static Webpage	A webpage with more content contains several pictures
medium-performance	Webpages with time complexity $O(N^2)$
high-performance	Webpages with time complexity $O(N^3)$
image download	1Mb picture image file
new query-based database webpages	A simple page for query employee lists

The first experiment demonstrated that intensively sending ping network packets sent by attackers to the devices within a short period of time severely affected the devices' performances. Related parameters such as (Table 3) Accordingly, this study proposed the aforementioned network packet filtering method, which identifies whether users are located at the same IP address when they submit requests. If so, the method further examines whether their requests are repeats. If they are, the users are asked to refer to the previous responses issued to them to prevent them from sending a large number of requests within a short period of time, causing transmission delays or damaging the system.

Table 3. Experiment parameters

Webpages browsed	Number of attackers	Filter time
Simple, static webpages	0	Unfiltered
Medium-performance webpages	1	10 ms
High-performance webpages	5	20 ms
Image download webpages	10	-
New query-based database webpages	15	-

The second experiment implemented the defense mechanism introduced in this study and determined whether it can effectively reduce the risk of low-rate DoS attacks. Given that existing webpages do not normally fall into only one of the aforementioned five categories and that users regularly request a variety of resources such as images and data types, all five webpage types were used in the second experiment.

4 Performance Assessment

4.1 Average Time Required to Send and Receive Network Packets and the Final Network Packet Return Time

Instead of exploring the extent to which attackers deplete server resources, this experiment was conducted to determine the effect of attackers on legitimate users' webpage-browsing experience. In this experiment, users were divided into two groups: attackers and legitimate users. The attackers initiated their attacks by continuously sending PING frames, whereas the legitimate users browsed webpages of all five types. A TCP connection was established every time a user visited a webpage. Once a connection was established, ten header frames were sent, which were then received and responded to in order to establish a new TCP connection. To prevent unclosed TCP connections from affecting the experimental results, signals indicating a closed TCP connection were sent to servers prior to completing new TCP connections. Users were required to wait 1 s before browsing the next webpage. Each experiment was performed 30 times (Figure 14).

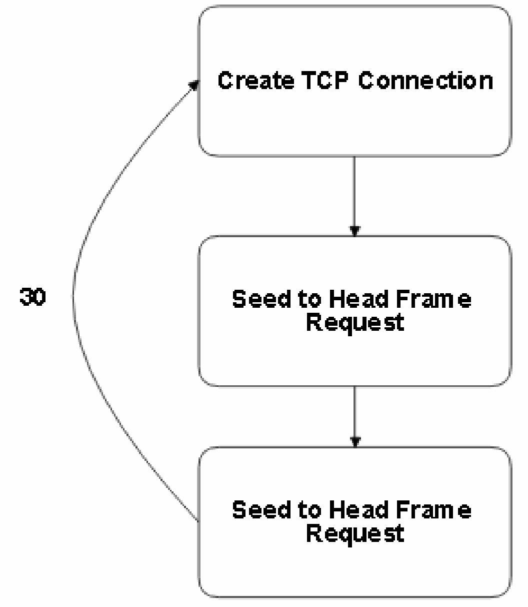


Figure 14. Experiment procedure

Follow-up assessments of the experiments were made using two time intervals. The first interval denoted the time required to receive returned data after a header frame was sent; this transmission process was repeated 10 times to determine the average time required, which was defined as the average network packet return time. The second interval denoted the time required to establish TCP connection and simultaneously send and receive 10 header frames (Figure 14); this value was defined as the final return time.

4.2 Experiment Results

4.2.1 Results of Experiment 1

According to the experiment procedure detailed in Section 3.1, this study conducted a experiment for browsing the five webpage types, each underwent 30 trials. The measured time intervals were ranked in ascending order, and the 10 middle values were averaged to plot Figure 15 and Figure 16. The results confirmed the effect of the number of attackers on legitimate users' usage experience; in particular, the effect was strongest for high-performance webpages.

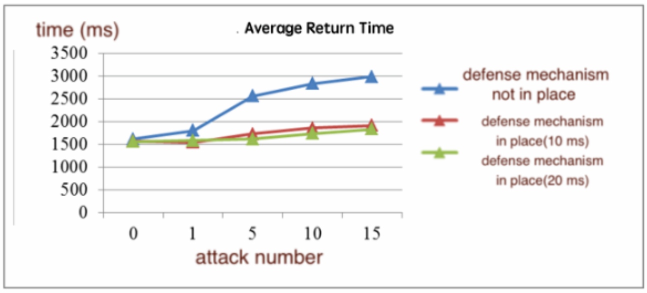


Figure 15. Average time required to send and return network packets, with the defense mechanism used

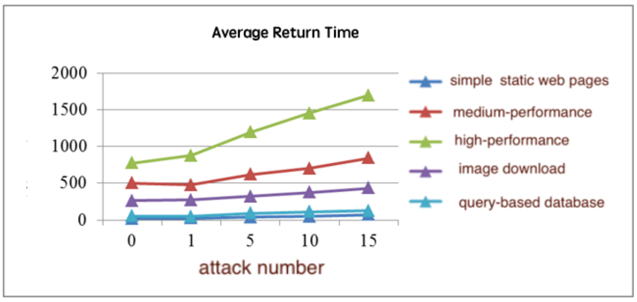


Figure 16. Average time required to send and return network packets for the five webpage types

4.2.2 Results for Experiment 2

Experiment 2 was performed to verify whether introducing the defense mechanism could effectively reduce the effect of attackers on legitimate users' usage experience. Similarly to Experiment 1, measurements from the experimental trials were listed in ascending order, and the 10 middle values were averaged to plot Figure 17 and Figure 18. The two graphs reveal that the defense mechanism effectively lowered the risk of a successful attack.

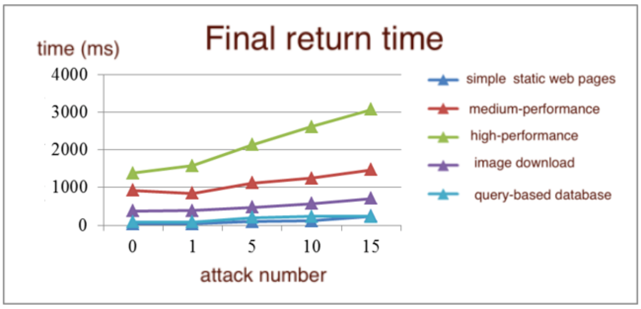


Figure 17. Final network packet return time, with the defense mechanism used

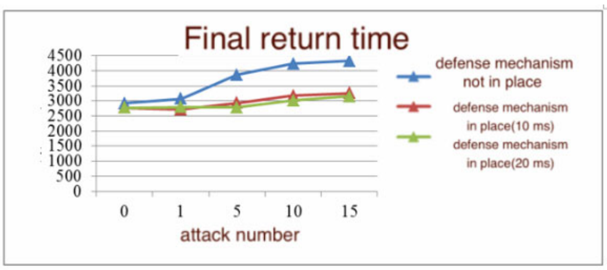


Figure 18. Final network packet return time for the five webpage types

4.2.3 CPU Usage Comparison and Power Consumption

Through the built-in monitor display of the operating system, and record the usage rate of the CPU before and after the protection and use usb Electric current Record power(Table 4), to ensure that our method will not cause a serious burden on the server. According to Figure 19 and Figure 20, we know that this protection mechanism is not It will put too much burden on the CPU and ensure that it can run normally on lightweight devices.

Table 4. Power consumption

Page Type	No protection mechanism	Use protection mechanisms
Simple	101 mA	102 mA
Static Web Page	101 mA	101 mA
Medium Performance	102 mA	103 mA
High Performance	110 mA	111 mA
Image Download	105 mA	104 mA
New query based	120 mA	120 mA

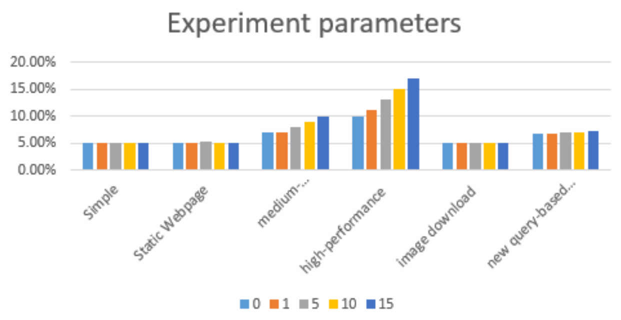


Figure 19. Experiment parameters

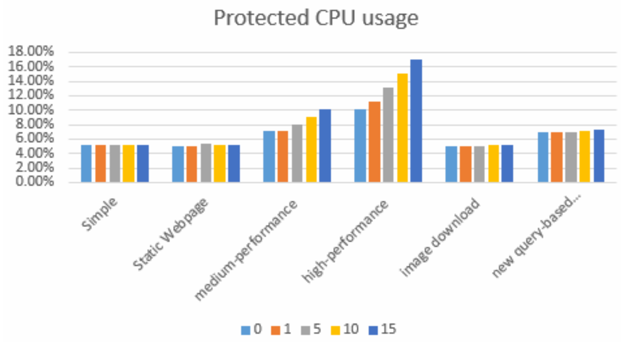


Figure 20. Protected CPU usage

5 Conclusion

Currently, the IoT is one of the most crucial information technologies in everyday life. However, despite the convenience of the IoT, it possesses security issues that must be quickly addressed.

Because HTTP is one of the major transmission methods used in IoT communication, protecting related devices from attack-led paralysis during the transmission process must be considered. HTTP/2 is the latest version of HTTP and is also the first upgrade of the 16-year-old protocol. HTTP/2 solves the potential HOL blocking problem of HTTP/1.1 and uses the binary transmission method to speed up transmission effectively. In addition, HTTP/2 has special functions such as multiplexing and stream prioritization. However, although HTTP/2 has numerous advantages, studies have revealed that it also has several problems, one of which is its security. Therefore, this study conducted a series of experiments to explore this issue. The first experiment confirmed the existence threats to HTTP/2 security, which have also been identified in previous studies. Thus, the experimental results of this study offered two major contributions. The first is the revelation that the higher the number of attackers, the longer the amount of time is required for legitimate users to load webpages and that the effect is strongest when loading high-performance webpages. The second major contribution is the proposed defense mechanism that was verified in the second experiment; this mechanism can effectively reduce the effect of attackers on the usage experience of legitimate users.

The experimental results also demonstrated that ping and WINDOW_UPDATE-type binary frames are susceptible to low-rate DoS attacks. However, WINDOW_UPDATE was not considered when conducting the experiments in this study. Therefore, this factor should be considered in future studies to make the defense mechanism introduced in this study more complete.

References

[1] E. Adi, Z. Baig, C. P. Lam, P. Hingston, Low-rate Denial-of-service Attacks against HTTP/2 Services, *5th International Conference on IT Convergence and Security (ICITCS)*, Kuala Lumpur, Malaysia, 2015, pp. 1-5.

[2] Wikipedia, *Hypertext Transfer Protocol*, https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol.

[3] D.-J. Deng, Y.-P. Lin, X. Yang, J. Zhu, Y.-B. Li, J. Luo, K.-C. Chen, IEEE 802.11ax: Highly Efficient WLANs for Intelligent Information Infrastructure, *IEEE Communications Magazine*, Vol. 55, No. 12, pp. 52-59, December, 2017.

[4] T. Berners-Lee, R. Fielding, H. Frystyk, *Hypertext Transfer Protocol--HTTP/1.0*, No. RFC 1945, May, 1996.

[5] D. J. Deng, C. H. Ke, H. H. Chen, Y. M. Huang, *Contention Window Optimization for IEEE 802.11 DCF Access Control, IEEE Transactions on Wireless Communications,* Vol. 7, No. 12, pp. 5129-5135, December, 2008.

[6] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, *Hypertext Transfer Protocol--HTTP/1.1*, No. RFC 2616, June, 1999.

[7] S. Chowdhury, V. Sapra, A. Hindle, *Is HTTP/2 More Energy Efficient than HTTP/1.1 for Mobile Users?*, PeerJ PrePrints, December, August, 2015.

[8] I. Grigorik, Making the Web Faster with HTTP 2.0, *Communications of the ACM*, Vol. 56, No. 12, pp. 42-49, December, 2013.

[9] M. Varvello, K. Schomp, D. Naylor, J. Blackburn, A. Finamore, K. Papagiannaki, To HTTP/2, or Not To HTTP/2, That Is The Question, *Networking and Internet Architecture*, July, 2015.

[10] M. Belshe, M. Thomson, R. Peon., *Hypertext transfer protocol version 2 (http/2)*, No. RFC 7540, May, 2015.

[11] R. Peon, H. Ruellan, *HPACK: Header Compression for HTTP/2. No. RFC 7541*. May, 2015.

[12] A. Kuzmanovic, E. W. Knightly, Low-rate TCP-targeted Denial of Service Attacks: The Shrew vs. the Mice and Elephants, *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications,* Karlsruhe, Germany, 2003, pp. 75-86.

[13] Y. Zhang, Z. M. Mao, J. Wang. *Low-Rate TCP-Targeted DoS Attack Disrupts Internet Routing*, NDSS, February, 2007.

[14] Y. Xiang, K Li, W Zhou. Low-rate DDoS Attacks Detection and Traceback by Using New Information Metrics, *IEEE Transactions on Information Forensics and Security 6.2*, pp. 426-437, January, 2011

[15] D. Namiot, M. Sneps-Sneppe, The Physical Web in Smart Cities, *2015 Advances in Wireless and Optical Communications (RTUWO)*, Riga, Latvia, 2015, pp. 46-49.

Biographies

Chi-Che Wu currently a Ph.D. student in the Department of Electrical Engineering of National Kaohsiung University of Sciences and Technology. His research interests include the mobile and cloud computing, computer network, algorithm.

Rung-Shiang Cheng received the M.S. degree in Computer Science and Information Engineering from National Cheng Kung University, and the Ph.D. degree in Electrical Engineering from National Cheng Kung University in 2001 and 2008, respectively. He was an assistant professor and an associate professor in 2008 and 2011, respectively. Then he became a professor in December 2017. He joined the Overseas Chinese University as a Professor and Chair in the Department of Information Technology in August 2018. He has published over 64 referred journal and conference papers in wireless and mobile communication protocols. His research interests include network simulation and performance analysis, wireless communications, and computer networks.

Chiung-Wen Hsu is an associate professor in the Department of Information Management at National Kaohsiung University of Science and Technology, Taiwan. She received her Ph.D. in management information systems from the Sun Yat-sen University in 2007. Her research interests relate to the domains of cognitive decision science, human computer interaction, electronic commerce, and technology adoption.

Li-Wei Wu is a Professor in the Department of International Business, Tunghai University. He received his Doctoral degree in Business Administration from the National Cheng Kung University. His main research areas include Services Marketing, Relationship Marketing and Internet Marketing. His research papers have been published in Journal of Management, Management Review, NTU Management Review, Sun Yat-Sen Management Review, International Journal of Commerce and Strategy, Tunghai Management Review, Asia Pacific Management Review, Journal of e-Business, Journal of Financial Services Marketing, Psychology & Marketing, Managing Service Quality, Journal of Services Marketing, Journal of Business and Industrial Marketing, Journal of Business Research, Management Decision, and others.