Hindawi
Security and Communication Networks
Volume 2019, Article ID 9078176, 15 pages
https://doi.org/10.1155/2019/9078176

Research Article

MSFA: Multiple System Fingerprint Attack Scheme for IoT Anonymous Communication

Tianbo Lu,[1,2] **Ting Meng**,[1,2] **Chao Li**,[3] **Guozhen Dong**,[1,2] **Huiyang Li**,[4] **Jiao Zhang**,[1] **and Xiaoyan Zhang**[1]

[1] *School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[2] *Key Laboratory of Trustworthy Distributed Computing and Service, Beijing University of Posts and Telecommunication, Ministry of Education, Beijing 100876, China*
[3] *Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, China*
[4] *Department of Computer Science and Engineering, The University of Texas at Arlington, Arlington, VA 76091, USA*

Correspondence should be addressed to Tianbo Lu; lutb@bupt.edu.cn and Chao Li; lichao@gzhu.edu.cn

Received 22 January 2019; Accepted 27 February 2019; Published 27 March 2019

Guest Editor: Fagen Li

For the past few years, Internet of Things (IoT) has developed rapidly and been extensively used. However, its transmission security and privacy protection are insufficient, which limits the development of IoT to a certain extent. As a technology of IoT information transmission, anonymous communication technology comes into being as an important means to ensure the security of healthcare data, which can better protect users' privacy in some ways. Nowadays, a variety of attack techniques for anonymous communication systems have been proposed by the academic community to track senders and receivers or discover communications between two users. Thus, the MSFA (Multiple System Fingerprint Attack) scheme for anonymous communication systems is presented in this paper where the MSFA scheme architecture, implementation in the Tor environment, and experimental data processing are described. Through a comparative analysis between two traces of visiting the same website based on the edit distance, it is shown that the longer the length of the site traffic data, the greater the edit distance of the site access traffic and the larger the range.

1. Introduction

A great number of research achievements have been gained about anonymous communication systems in the past 40 years. Ruei-Hau Hsu et al. [1] proposed network-covered and network-absent authenticated key exchange protocols for D2D communications to guarantee accountable group anonymity, end-to-end security to network operators. Anonymous communication technology is an effective technique for healthcare data privacy protection. Amin et al. [2] proposed the architecture of a patient monitoring system in WMSN (wireless medical sensor network) and designed an anonymous mutual authentication protocol suitable for mobile users to provide secure access and privacy for patient data. Mingshan Xie et al. [3] proposed the anonymization protection algorithm which is suitable for the data exchange in an incompletely open manner for the ego of data in the IoT. The anonymous dataset generated by the algorithm can effectively protect the sensitive information of IoT under the premise of ensuring the availability of the data. The EXCHANge protocol [4], a cryptoless over-the-air key establishment multiround protocol based on sender/receiver anonymity, was specifically conceived to secure IoT networks based on the IEEE 802.15.4 communication technology. Network malicious attackers or criminals rarely attack directly through their own computers. Before attacking the final target, they often land anonymous communication systems to hide their identities, such as Tor [5], JAP [6], Freenet [7], and I2P [8]. Tor and I2P have a large group of users and they have published software versions for mobile ad-hoc networks. The correspondence between input and output streams is hidden by Tor anonymous system in a variety of ways, while the attacker's goal is to identify the correspondence. Almuhtadi et al. [9] made a preliminary evaluation about Misty

clouds which is a privacy-preserving platform for online user anonymity in Social Internet of Things, indicating that the new algorithm was better than the existing Tor algorithm and could achieve the expected privacy goal within the expected performance cost. The communication relationship between sender and receiver in the anonymous communication network can be discovered by the network fingerprint attack. The academia has carried out extensive research on the scheme and application of fingerprint attack.

Network traffic can be disturbed; for example, packets may be cached on a relay node for a period of time or be cut, recombined, retransmitted, or even lost. In web fingerprint, passive traffic analysis attack techniques are used that only require an attacker to configure a network environment similar to a regulator and access the target site using the same encryption proxy technology. The actual address of the regulator's communication side is identified by analyzing the generated traffic characteristics. Fingerprinting [10] combines a number of input sources. Fingerprint attack technique identifies whether the sender communicates with a particular recipient by collecting the sender or receiving the feature information of both. The feature information can be network traffic characteristics, routing information characteristics, and node information characteristics. When the receiver communicates with the sender, the feature information between them will be collected by the fingerprint technology to form the fingerprint, which can determine a communication relationship between the sender and the receiver when they communicate again. In the existing fingerprint identification attacks [11–18], the researchers use the packet size distribution, the sum of the packet size, packet timing interval, etc. as the basic statistical features to characterize the web fingerprint feature set. Researchers have demonstrated the feasibility of the website fingerprint attack methods and conducted further research on web fingerprinting. Cai et al. [11] proposed a website fingerprint attack that could successfully attack the latest proposed defensive traffic analysis attack scheme HTTPOS [19].

Our contributions are as follows: based on the CAI fingerprint attack prototype [11], we propose the MSFA fingerprint attack scheme in three aspects, namely, MSFA scheme architecture and module design, the implementation of MSFA scheme in Tor anonymous communication network, and the capture of the original traffic information and data processing.

2. CAI Fingerprint Attack Scheme

In this section, the background of the CAI fingerprint attack scenario, the CAI fingerprint attack model, as well as the attack process and features will be outlined.

2.1. The Background of the CAI Fingerprint Attack Scheme. Cai et al. [11] proposed a website fingerprint attack that could successfully attack the HTTPOS [13] traffic analysis scheme. CAI fingerprint attack is based on a simple network behavior model which can correctly predict the pages accessed by users over half of the time for any defense model. At the same time, it can correctly identify whether the user accesses a specific site with the experimental success rate over 90%.

2.1.1. Web Page Tracking. Web pages contain multiple objects such as HTML files, images, and flash, and the browser sends a separate request to each object. With the way of a combination of multiple TCP links and pipelines, it is more quickly for browsers to load pages. The browser requests the page-related objects before loading the page. Note that the order of requests has inheritance stability, and an object can only be requested after the browser has received some referencing pages. Some requests may be delayed due to the CPU load and packet reordering so that the order of requests and responses may be different when the browser loads a page every time. Some requests may be omitted if there is a copy of the object in memory. The number of requests sent by the browser and the total number of packets returned to the server may vary with the change of the size of the dynamic web page and the objects it contains [11].

2.1.2. Damerau-Levenshtein Edit Distance. In information theory and computer science, Damerau-Levenshtein distance [20–22] indicates the distance between two strings. In short, it refers to two finite sequences of symbols that convert a string to another string with a minimum number of operations, where the operation is defined as an insertion, deletion, replacement, or swap of two adjacent characters. In Damerau's [20] study, not only are the four edits distinguished, but it also points out that they correspond to more than 80% of all spelling errors while Damerau only considers edits that could correct a misspelling. The difference between Damerau-Levenshtein and Classic Levenshtein is that Damerau-Levenshtein distance allows the exchange between characters while only insert, delete, and replace operations are permitted in Levenshtein distances. The Levenshtein distance is optimized to include the exchange of adjacent characters, resulting in the different measurement distances called the Damerau-Levenshtein distance [21].

2.1.3. LIBSVM. LIBSVM software is a library for integrated support vector machines that supports multicategory classification. Support vector machine (SVM) is a technology that effectively classifies data. The essence of LIBSVM is a library for support vector machines, and there is no need for users to understand the basic theory behind the support vector machines while just following the basic program to get the corresponding result. A classification task for LIBSVM usually involves two separate datasets, a training set and a testing set. A model can be generated by LIBSVM based on the training data, which predicts the target value of the test set. The data in the test set only provides the attribute values of the test data.

2.2. The Characteristics of CAI Fingerprint Attack. In the CAI fingerprint attack process (Figure 1), the client traffic

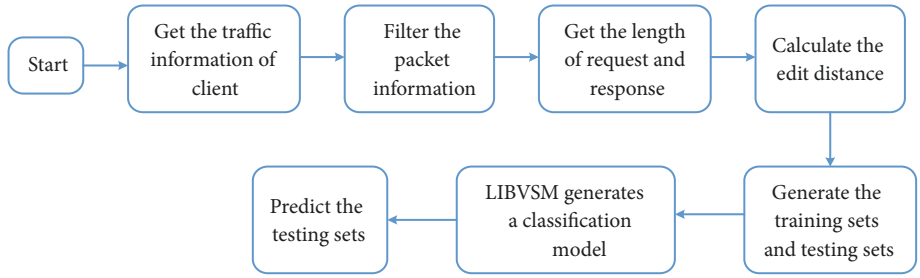

Figure 1: CAI fingerprint attack process.

information is captured through the client agent first and then the captured packet information is processed to obtain the packet length information. The edit distance is then calculated to generate the training and testing sets required for LIBSVM classification. The classification model is generated by LIBSVM for the prediction of the test set data to determine whether the user has visited the target website.

Firstly, Cai et al. proposed a new method to calculate the similarity of web access [11]. The order of the request and response packets shows the size and importance of the objects referenced in a page, so packet scheduling is very important for identifying web pages. CAI fingerprint attack transformed tracks into strings and compared the similarities between the two tracks using edit distances. Damerau-Levenshtein distance is a good metric that allows insertion, deletion, replacement, and interchange. Secondly, the CAI fingerprint attack scheme used LIBSVM to establish a classification model for the processed web packet data and predict the pages accessed by the user.

3. MSFA Attack Scheme

In this section, we have designed the MSFA fingerprint attack scheme.

3.1. MSFA Scheme Architecture. The architecture of the MSFA fingerprint attack scheme which includes the design goals of the MSFA scheme, the threat model, and the attack model is discussed.

3.1.1. MSFA Design Goal. The design idea of the MSFA attack scheme is based on CAI fingerprint attack prototype, and the concrete realization scheme is proposed for different anonymous systems. The following describes the MSFA attack scheme through three aspects:

(i) The database system of the MASF attack scheme. The captured network traffic information can be effectively saved by the database system, which designs a scientific and reasonable database for the site IP address, traffic information, site name, etc. The database system is provided with the advantages of simple structure and clear function.

(ii) The improved MSFA attack scheme is based on the CAI fingerprint attack scheme, and experiments are performed on different categories of websites for data collection. The MSFA attack scheme is implemented in the Tor anonymous communication system environment.

(iii) Capture the original flow of information. To obtain the fingerprint information of the website, it is necessary to capture the website original traffic information. A web page is composed of multiple objects, and a separate request is sent for each object by the browser. Multiple TCP links can be exploited by the browser to load the page faster. The browser will request the relevant objects of the page before loading the page that generates network traffic. The packets in the tracking can be roughly divided into two categories: request packets and response packets. When a user browses the encrypted proxy page, all the relevant documents on the page will be downloaded by the user browser and each of them requires a separate TCP link to return.

3.1.2. MSFA Scheme Threat Model. As shown in Figure 2, firstly, it needs to connect the I2P network through the local connection when Amy visits the website through the I2P network. Kad algorithm is exploited by the I2P network to obtain information on the network node and access the destination server through the nodes of the I2P network. Multiple links are used by the I2P to send and receive data, but if the links of sending data and receiving data are different, the number of nodes on the two links will be different. Ken accesses through the Tor network and it reaches the destination server to visit the site through the entrance node, intermediate node, and exit node three hops in the Tor network. The three-hop routing nodes which have been set up will not automatically change unless they are manually changed. Moreover, the I2P routing nodes have a valid period. Fingerprint attacker captures the traffic between the client and the anonymous communication system entry node, subsequently analyzes the traffic characteristics, and finally destroys the anonymous communication system to form an attack.

3.1.3. MSFA Attack Model. As shown in Figure 3, the attack model of MSFA scheme has been presented in this paper. When the user accesses the Internet through the ordinary browser or anonymous communication system, the original traffic information can be captured by Charles software or Wireshark [23] software.

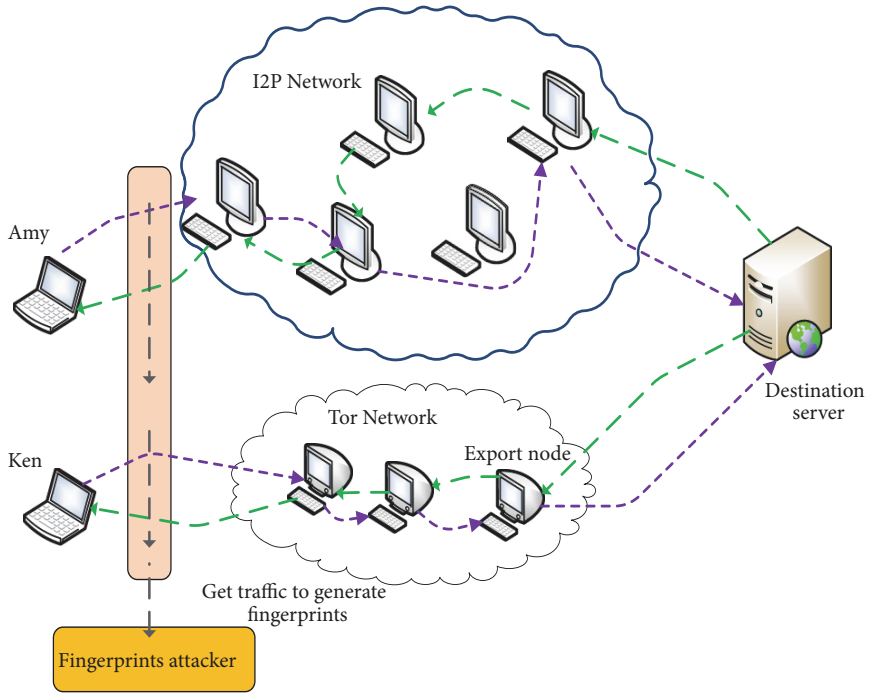

Figure 2: MSFA threat model.

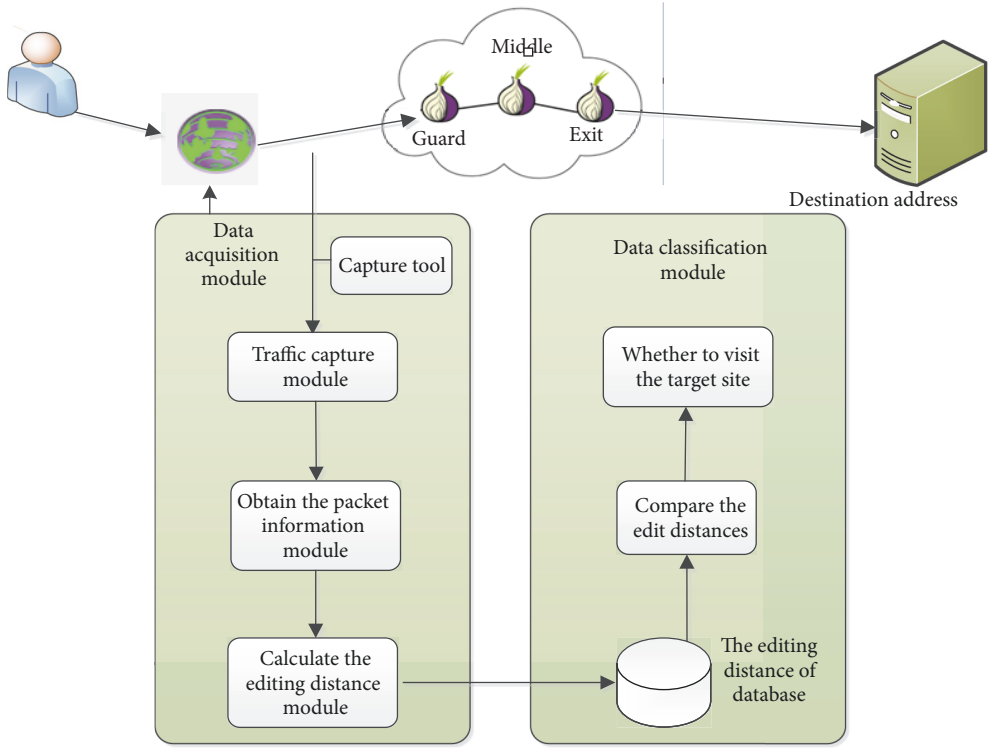

Figure 3: MASF attack model.

3.2. MSFA Scheme Module Design. We deeply study the module design of MSFA scheme (Figure 4), and mainly discuss the design of the traffic capture module, packet information acquisition module, and edit distance calculation module.

3.2.1. Traffic Capture Module Design. The experiment collects data from multiple target sites and organizes them according to the categories of target sites, giving each site a unique ID. In terms of data capture at the site, the experiment

Figure 4: Flow module design model.

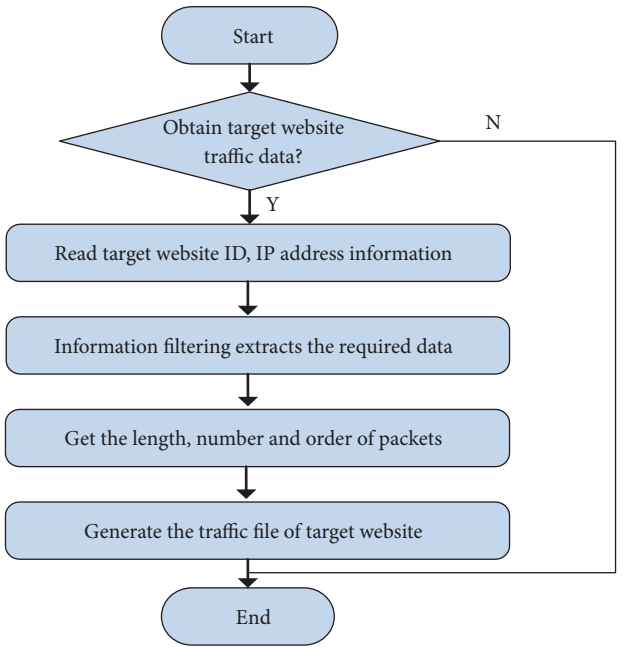

Figure 5: Get the packet information module design.

selects Wireshark to capture the traffic of TOR and I2P site.

3.2.2. Obtain the Packet Information Module Design. The module design for obtaining packet information is presented in Figure 5. In this experiment, we used Ubuntu system to capture the site traffic and Wireshark to capture data packets. The TCP/IP protocol defines the packets transmitted over the Internet, called IP Datagram. IP Datagram is independent of hardware and it consists of header and data. The first 20 bytes of the header are fixed length while the rear is an optional field whose length is variable. The source address and destination address of the header are both IP address.

The original data saved by Wireshark is complex while just the size and the number of Request and Response in the experiment need to be recorded to calculate the Damerau-Levenshtein distance. Therefore, the data needs to be processed. The experiment first groups the captured traffic data according to IP which must be the routing nodes of the Tor to obtain valuable packet information. In the experiment, after the traffic data is obtained, it only needs to request and respond to the packet size as it

```
(1) class Filter{
(2)   public:
        //Defines the storage server IP address; server_ips
(3)       unordered_set <string> server_ips;
        //Defines the storage client IP address: client_ips
(4)       unordered_set <string> client_ips;
        //Defines the maximum port that needs to be processed
(5)       int PROXYPORT_MIN;
        //Defines the minimum port that needs to be processed
(6)       int PROXYPORT_MAX;
(7)   public:
        //FilterConstructor
(8)       Filter(char* clientipfname, char* serveripfname, int portmin, int portmax);
        //Read the IP function
(9)       int read_ips(unordered_set<string>&set, char* fname);
        //Determines whether the file loads the function
(10)       bool is_onload(u_char* payload);
        //Determine whether the traffic function is listening
(11)       bool is_monitoredtraffic(char* src, unsigned int sport, char* dst, unsigned int dport);
        //Implement the transformation function for the data
(12)       RETparse_one(char* capfname, int proxy_port_min, intproxy_port_max, int remove_ack,
           char* monitoredoutname, char* localoutname, char* c2stau, char* s2ctau, char* timeseq);
(13)  };
```

Algorithm 1: Filter class code implementation.

```
(1) server_ips
(2) client_ips
(3) PROXYPORT_MIN
(4) PROXYPORT_MAX
(5) IF (Source address equals toclient_ips && Destination address equal toserver_ips
      && Source address port <= PROXYPORT_MAX
      && Source address port >= PROXYPORT_MIN
      && Destination address port <= PROXYPORT_MAX
      && Destination address port >= PROXYPORT_MIN)
(6) IF (Whether it is interrupted)
(7) IF (Destination address port <=PROXYPORT_MAX && Destination address port>=PROXYPORT_MIN)
(8)   Packet length* = -1
(9)   Output packet length
```

Algorithm 2: Algorithm for getting the packet information.

contains more complex data. Therefore, the original data needs to be processed. Filter class (Table 1) includes four attributes, namely, server_ips, client_ips, PROXYPORT_MIN, and PROXYPORT_MAX, and five methods, namely, Filter (), read_ips (), is_onload (), parse_one (), and is_monitoredtraffic (), and its partial details are shown in Algorithm 1. During the conversion of the website traffic file .cap, the lengths of the request and response packets are required in the experiment and saved in the file in turn. The request packet is identified as negative while the response packet is identified as positive. Algorithm 2 shows the algorithm for obtaining the packet information.

3.2.3. Calculate the Editing Distance Module Design. Get the length of the packet that the site accesses by obtaining the packet information module; then, it is necessary to calculate the editing distance between the two sites visit. In the experiment, the Damerau-Levenshtein distance, also known as the editing distance, is used which refers to the conversion of a string to another string in a minimum number of operations. The operation is defined as an insertion, deletion, or replacement of a character, or transposition of adjacent characters. In order to calculate the distance between the two sites, we use a matrix to store the distance and complete all edit distance calculations to output an edit distance matrix.

Table 1: Filter class functions and properties.

Filter	
+server_ips	Server-side IP
+client_ips	Client IP
+PROXYPORT_MIN	Minimum capture port
+PROXYPORT_	Maximum capture port
+filter()	Constructor
+read_ips()	Read the IP function
+is_onload()	Determines whether the file loads the function
+parse_one()	Conversion data function
+is_monitoredtraffic()	Determine the target traffic information function

Table 2: Calculate the Edit Distance Module algorithm.

Levenshtein	
+sizes	Store the length of file
+pool	Store the file pool of file
+str1	String 1
+str2	String 2
+websites	The number of storage sites
+trials	Store the number of visits per site
+distr	Store the distance
+get_distr	Get the distance function
+fetch_pool	Take the file function from the file pool
+is_size	To determine the length function
+read_size	Read the packet length function
+Parse_data	Conversion data function
+DLdis	Calculate the edit distance function

The algorithm for calculating the edit distance is shown in Table 2.

4. Implementation and Evaluation

4.1. Implementation of MSFA Scheme in the Tor Anonymous System Environment. The MSFA fingerprint attack scenario is tested under the Tor anonymous communication system.

4.1.1. Tor Anonymous System Installation and Configuration. In the experiment, we use the Linux system, download the corresponding Linux package on Tor official website, and then unpack the software package. Before running the Tor Browser, the global VPN proxy needs to be installed on the computer. Otherwise, the Tor Browser will not be able to establish a link and run normally.

4.1.2. Get the Packet Information. After the .pcap file is captured by the site, each .pcap file forms a .txt file that records the traffic information of the site.

4.2. MSFA Scheme Experimental Data Processing. The data processing of the MSFA scheme is discussed in detail, and the data of different kinds of websites are classified.

4.2.1. Calculate the Edit Distance. After fetching the packet length information required for the experiment, the next step is to further process the packet information and calculate the Damerau-Levenshtein edit distance. We still use the CAI fingerprint attack [CZJ2012] to standardize the edit distance to compensate for the changes of the packet tracking length. If d(t, t') means Damerau-Levenshtein edit distance, the fingerprint attack will normalize the edit distance as follows:

$$L\left(\mathrm{t}, t'\right) = \frac{d\left(t, t'\right)}{\min\left(|t|, \left|t'\right|\right)}. \tag{1}$$

$|t|$ represents the packet length in trace t, and the classifier normalizes the shortest value of two lengths. If the difference between t and t' is very large in length, then these two may come from different pages. In this case, dividing by $\min(|t|, |t'|)$ will result in a larger normalized distance, which is a feasible standardized distance. The implementation of calculation is shown in Algorithm 3.

4.2.2. Data Processing. According to the collection of the sites, we sort and select a few for processing. The specific process is as follows:

Select msn.com to do the experiment. The MSN was accessed through the Tor anonymous system at different times, with a total of 10 visits. The data for the site traffic was formed after the Wireshark captured the accessing traffic and the Filter class handled the file. The data is shown in Table 3.

After obtaining the traffic information for 10 visits to the msn.com website, we use Levenshtein_cantor_mpi to calculate the edit distance for this 10 traffic, as shown in Table 4.

When calculating the edit distance, the string that is accessed by the two traffic records of the site is compared. The smaller the edit distance is, the more similar the two records are.

By comparison, we find that the minimum distance of edit distance is 0.069 which is the fourth visit and the ninth visit msn.com site between the two edit distances. The maximum is 2.64 which is the editing distance between the first and fifth visiting. So we can initially determine that the edit distance of accessing MSN website ranges from 0 to 2.64. At the same time, the smallest distance to the other visiting average edit distance is selected as msn.com website fingerprint to store into the fingerprint database. By comparison, the minimum of average editing distance is between the second and other visiting msn.com, so the second visiting is put as a fingerprint of msn.com into the database.

Select Amazon to do the experiment. Amazon was accessed through the Tor anonymous system at different times, with a total of 10 visits. After Wireshark captured the accessing traffic and the Filter class handled the file, it formed the data for the site traffic. The data is shown in Table 5.

```
(1) double Levenshtein :: DLdis(int ms, int ns)
(2) {
(3)     double ret = 0;
(4)     int min;
        //Pretreatment
(5)     int m = ms;
(6)     int n = ns;
        //min takes the smaller between m and n
(7)      min = m < n ? m : n;
(8)      min = min = = 0 ? 1 : min;
(9)      int i, j;
(10)     double subcost, transcost;
        //Define operating costs to two
(11)     double idcost = 2;
        //Store the distance array
(12)     double** dis = new double*[m];
        //Initialize the array
(13)     for(i = 0;i < m;i++)
(14)         dis[i] = new double[n];
(15)     for(i = 0; i < m;i++)
(16)         for(j = 0; j < n; j++)
(17)             dis[i][j] = -1;
        //Calculate the operating costs of the first ramp line and the first vertical line
(18)     for(i = 0; i < m; i++)
(19)         dis[i][0] = i * idcost;
(20)     for(j = 0; j < n; j++)
(21)         dis[0][j] = j * idcost;
        //Calculate the operating costs of non-first rungs and first vertical lines.
(22)     for(i = 1; i < m; i++)
(23)     {
(24)         for(j = 1; j < n; j++)
(25)         {
                //If the two strings are equal, the operating cost is zero.
(26)     if(str1[i] = = str2[j])
(27)             subcost = transcost = 0;
(28)     else
(29)             {
                    //Otherwise the replacement cost is two.
(30)                 subcost = 2;
                    //The exchange cost is 0.1
(31)                transcost = 0.1;
(32)            }
                //The minimum cost is the edit distance, which is stored in the matrix.
(33)            dis[i][j] = minimum(dis[i-1][j] + idcost, dis[i][j-1] + idcost, dis[i-1][j-1] + subcost);
                //Two character exchanges
(34)     if(i >1 && j >1 && str1[i] = = str2[j-1] && str1[i-1] = = str2[j])
(35)             dis[i][j] = dis[i][j] < dis[i-2][j-2] + transcost ? dis[i][j]: dis[i-2][j-2] + transcost;
(36)     }
(37)  }
        //Free dis
(38)  for(i = 0; i < m; i++)
(39)      delete[] dis[i];
(40)      delete[] dis;
(41) }
```

Algorithm 3: Calculation of the edit distance implementation.

Table 3: The partial traffic of MSN site.

1	Upstream	-611	-68	-68	-68	-863	-1416	-68	-68	-611	-68	-68	-68	-863
	Downstream	68	611	611	68	611	1416	1416	1416	1416	1416	1416	1416	1416
2	Upstream	-68	-68	-1125	-68	-68	-68	-68	-68	-68	-68	-1416	-68	-863
	Downstream	68	611	611	68	611	68	611	68	611	1416	1348	68	611
3	Upstream	-611	-68	-68	-68	-252	-68	-68	-68	-1416	-1416	-68	-68	-68
	Downstream	611	68	68	68	611	1416	1416	1416	1416	1416	1416	1416	1416
4	Upstream	-611	-68	-68	-68	-805	-1416	-805	-1416	-68	-1125	-68	-68	-68
	Downstream	68	611	611	68	611	1416	1416	1416	1416	1416	1416	1416	1416
5	Upstream	-68	-68	-68	-611	-68	-1416	-1416	-68	-543	-68	-68	-1416	-1416
	Downstream	68	68	68	611	611	1416	1416	1416	1416	1416	1416	1416	1416
6	Upstream	-68	68	-68	-1154	-68	-68	-68	-68	-68	-68	-349	-68	-1416
	Downstream	68	611	611	68	611	1416	1416	1416	1416	1416	1416	1416	1416
7	Upstream	68	68	-68	1416	68	68	1416	68	68	68	68	68	68
	Downstream	68	611	611	611	68	611	68	611	611	68	611	1416	1416
8	Upstream	-68	-68	-68	-68	-68	-68	-1416	-68	-68	-68	-68	-68	-68
	Downstream	68	68	611	68	68	80	611	68	611	68	611	1416	805
9	Upstream	-68	-68	-1416	-68	-68	-68	-68	-68	-1406	-68	-68	-68	-1416
	Downstream	68	68	68	611	68	611	1416	1416	1416	1416	1416	1416	1416
10	Upstream	-68	-1416	-68	-68	-68	-1416	-1416	-68	-68	-68	-68	-1416	-1416
	Downstream	68	68	611	1416	291	68	1416	1416	1416	1416	1416	1416	1416

After obtaining the traffic information for 10 visits to the Amazon website, we use Levenshtein_cantor_mpi to calculate the edit distance for this 10 traffic, as shown in Table 6.

By comparison, we find that the minimum distance of edit distance is 0.034 which is the fourth visit and the fifth visit of amazon.com site between the two edit distances. The maximum is 11.81 which is the editing distance between the first and ninth visiting. So we can initially determine that the edit distance of accessing Amazon website ranges from 0 to 11.81. At the same time, the smallest distance to the other visiting average edit distance is selected as amazon.com website fingerprint to store into the fingerprint database. By comparison, the minimum of average editing distance is between the 9th and other visiting amazon.com, so the 9th visiting is put as a fingerprint of amazon.com into the database.

Select YouTube to do the experiment. YouTube was accessed through the Tor anonymous system at different times, with a total of 10 visits. After Wireshark captured the accessing traffic and the Filter class handled the file, it formed the data for the site traffic. The data is shown in Table 7.

The Levenshtein_cantor_mpi is used to calculate the edit distance for the 10 traffic which has been obtained by 10 visits to the youtube.com website, as shown in Table 8.

By comparison, we find that the minimum distance of edit distance is 0.27 which is the 8th visit and the 7th visit of youtube.com site between the two edit distances. The maximum is 8.44 which is the editing distance between the 10th and 9th visiting. So we can initially determine that the edit distance of accessing YouTube website ranges from 0 to 8.44. At the same time, the smallest distance to the other visiting average edit distance is selected as youtube.com website fingerprint to store into the fingerprint database. By comparison, the minimum of average editing distance is between the 8th and other visiting youtube.com, so the 8th visiting as a fingerprint of youtube.com is put into the database.

5. Conclusion

The real spreading of IoT services requires customized security and privacy levels to be guaranteed. Many IoT services and applications may expose sensitive and personal information which may be abused by attackers. As such, privacy protection must be considered and it is a core requirement in any IoT ecosystem. The MSFA attack scheme proposed in this paper is based on the edit distance to compare the similarity between the two visits. Firstly, the differences between the different types of website traffic can be observed from the above data. Take the Amazon which is the e-commerce website and YouTube which is the video website as examples. The traffic of Amazon website ranges from 0 to 11.81, while the traffic of YouTube video ranges from 0 to 8.44. Secondly, the length of the traffic data has an impact on the edit distance. In general, the longer the length of the site traffic data, the greater the edit distance of the site access traffic and the larger the range.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Table 4: The edit distance of MSN accessing traffic between each other.

	1	2	3	4	5	6	7	8	9	10
1	0	0.78	0.954065	0.083248	2.64735	0.869404	1.300171	1.215214	2.223183	0.948547
2	0.78	0	0.553862	1.591451	0.885303	1.047593	0.667157	0.787618	0.246367	1.110267
3	0.954065	0.553862	0	0.241667	0.545528	1.403252	1.972764	1.806911	1.641869	1.489634
4	0.083248	1.591451	0.241667	0	1.065391	0.869876	2.082786	2.279481	0.069204	0.867333
5	2.64735	0.885303	0.545528	1.065391	0	1.719807	1.169649	1.345401	0.0609	1.74
6	0.869404	1.047593	1.403252	0.869876	1.719807	0	0.64718	0.641953	0.150519	0.603576
7	1.300171	0.667157	1.972764	2.082786	1.169649	0.64718	0	0.544458	0.228374	0.772
8	1.215214	0.787618	1.806911	2.279481	1.345401	0.641953	0.544458	0	0.958478	0.7032
9	2.223183	0.246367	1.641869	0.069204	0.0609	0.150519	0.228374	0.958478	0	0.289273
10	0.948547	1.110267	1.489634	0.867333	1.74	0.603576	0.772	0.7032	0.289273	0
	1.102118	0.766962	1.060955	0.915044	1.117933	0.795316	0.938454	1.028271	0.586817	0.852383

Table 5: The partial traffic of Amazon site.

1	Upstream	-68	-68	-68	-68	-68	-68	-68	-68	-1416	-68	-68	-68	-68
	Downstream	68	68	611	68	611	68	611	68	611	1416	1416	1416	1416
2	Upstream	-68	-68	-68	-68	-68	-68	-68	-68	-1026	-68	-68	-68	-68
	Downstream	68	611	611	6868	611	1416	1416	999	1416	999	1416	805	68
3	Upstream	-611	-68	-611	-68	-611	-68	-1416	-1416	-1416	-68	-68	-611	-611
	Downstream	68	611	1416	1416	1416	1416	68	611	68	1416	291	1416	1416
4	Upstream	-68	-68	-68	-1416	-68	-68	-68	-68	-68	-68	-68	-68	-68
	Downstream	68	-611	611	1125	611	1125	611	1416	1416	1416	1416	68	1416
5	Upstream	-68	-863	-68	-68	-68	-68	-68	-68	-68	-1416	-611	-68	-68
	Downstream	1125	68	1125	1416	1416	1416	1416	1416	1416	1416	1416	1416	1416
6	Upstream	-68	-68	-68	-611	-68	-611	-68	-291	-1416	-68	-68	-572	-68
	Downstream	68	68	68	611	68	1416	1416	1416	1416	1416	1416	1416	1416
7	Upstream	-1416	-1416	-68	-68	-1416	-1416	-805	-1416	-1416	-80	-1416	-1416	-1416
	Downstream	68	68	68	611	1416	1416	1416	1416	1416	1416	1416	1416	1416
8	Upstream	-68	-68	-68	-68	-68	-68	-68	-68	-68	-68	-68	-68	-68
	Downstream	68	611	611	68	611	68	611	1416	1416	1416	1416	388	68
9	Upstream	-68	-68	-68	-68	-68	-68	-68	-68	-68	-68	-68	-68	-68
	Downstream	68	68	611	1416	1416	1416	1416	68	1416	1416	1416	514	1416
10	Upstream	-68	-68	-68	-80	-68	-68	-68	-68	-68	-68	-68	-68	-68
	Downstream	68	-611	611	68	611	1125	611	1416	1416	1416	1416	68	611

Table 6: The edit distance of Amazon accessing traffic between each other.

	1	2	3	4	5	6	7	8	9	10
1	0	0.359215	0.320107	4.281042	4.388232	8.531568	5.133218	11.81551	4.401792	0.00311
2	0.359215	0	0.223082	3.543489	3.637893	7.273014	4.296347	10.16845	3.64932	0.357429
3	0.320107	0.223082	0	3.69364	3.790635	7.539715	4.453549	10.16845	3.807355	0.319185
4	4.281042	3.543489	3.69364	0	0.034504	1.460591	0.431633	2.534893	0.267367	4.276196
5	4.388232	3.637893	3.790635	0.034504	0	1.405703	0.397243	2.462701	0.272188	4.383303
6	8.531568	7.273014	7.539715	1.460591	1.405703	0	1.063288	0.842246	1.407332	8.519348
7	5.133218	4.296347	4.453549	0.431633	0.397243	1.063288	0	2.008422	0.395934	5.124948
8	11.81551	10.16845	10.16845	2.534893	2.462701	0.842246	2.008422	0	2.452807	11.79957
9	4.401792	3.64932	3.807355	0.267367	0.272188	1.407332	0.395934	2.452807	0	4.395612
10	0.00311	0.357429	0.319185	4.276196	4.383303	8.519348	5.124948	11.79957	4.395612	0
	3.923379	3.350824	3.465262	2.052336	2.07724	3.804276	2.330453	5.458984	2.104971	3.91786

Table 7: The partial traffic of YouTube site.

1	Upstream	-1109	-1109	-595	-595	-1109	-595	-1109	-595	-1109	-1109	-595	-1400	-1332
	Downstream	595	595	595	595	595	1400	1400	1400	692	1400	304	1400	1400
2	Upstream	595	-595	-595	-1138	-595	-1109	-595	-1400	-1400	-527	-1138	-1400	-595
	Downstream	595	1400	1400	983	1109	595	595	1109	1400	1400	1400	275	595
3	Upstream	-1109	-595	-595	-595	-595	-1109	-1109	-595	-595	-595	-595	-1109	-1138
	Downstream	595	1400	1400	1400	721	595	595	595	595	595	595	1400	1400
4	Upstream	-595	-595	-595	-1109	-595	-595	-595	-595	-595	-595	-595	-1109	-1138
	Downstream	595	595	595	595	1400	1400	1400	692	595	595	595	1109	595
5	Upstream	-1109	-1109	-595	-1400	-847	-195	-1138	-1109	-1109	-1400	-304	-1400	-595
	Downstream	595	1400	1400	1400	1400	1400	459	1400	1400	1400	178	595	595
6	Upstream	-595	-595	-595	-1109	-1109	-595	-595	-1109	-595	-1109	-595	-595	-595
	Downstream	595	595	595	595	595	595	692	595	595	1109	595	595	595
7	Upstream	-595	-595	-595	-1109	-595	-1106	-595	-1109	-595	-595	-595	-1400	-333
	Downstream	595	595	595	595	1400	1400	1400	692	595	595	1109	595	595
8	Upstream	-595	-595	-595	-595	-1109	-595	-1109	-595	595	-595	-595	-1109	-1109
	Downstream	595	595	595	595	1400	1400	1400	692	595	1400	304	595	595
9	Upstream	-595	-595	-595	-1109	-595	-1109	-595	-595	-595	-595	-595	-595	-1109
	Downstream	595	595	595	595	1400	1400	1400	692	595	595	1109	595	1109
10	Upstream	-1109	-1109	-595	-1138	-595	-1400	-1399	-1400	-1390	-1400	-1400	-1388	-1400
	Downstream	595	1400	1400	1400	1400	1400	1400	1400	1400	595	1400	1400	595

Table 8: The edit distance of YouTube accessing traffic between each other.

	1	2	3	4	5	6	7	8	9	10
1	0	0.301	1.371	0.713	4.953	2.456	0.632	0.659	0.895	5.584
2	0.301	0	1.43	0.695	5.069	2.528	0.683	0.705	0.871	5.725
3	1.371	1.43	0	2.307	2.389	0.879	0.754	0.7	2.61	2.801
4	0.713	0.695	2.307	0	6.953	3.706	1.333	1.381	0.145	7.747
5	4.953	5.069	2.389	6.953	0	1.4224	3.677	3.526	7.581	0.574
6	2.456	2.528	0.879	3.706	1.424	0	1.644	1.572	4.107	1.748
7	0.632	0.683	0.754	1.333	3.677	1.644	0	0.27	1.562	4.194
8	0.659	0.705	0.7	1.381	3.526	1.572	0.27	0	1.614	4.041
9	0.895	0.871	2.61	0.145	7.581	4.107	1.562	1.614	0	8.44
10	5.584	5.725	2.801	7.747	0.574	1.748	4.194	4.041	8.44	0
	1.756	1.801	1.542	2.498	3.615	2.006	1.475	1.447	2.782	4.085

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the following grants: the National Natural Science Foundation of China under Grant no. 61170273; the China Scholarship Council under Grant no. [2013]3050. We thank the anonymous reviewers for their valuable comments and suggestions.

References

[1] R.-H. Hsu, J. Lee, T. Q. S. Quek, and J.-C. Chen, "GRAAD: group anonymous and accountable D2D communication in mobile networks," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 2, pp. 449–464, 2018.

[2] R. Amin, S. Islam, G. Biswas, M. Khan, and N. Kumar, "A robust and anonymous patient monitoring system using wireless medical sensor networks," *Future Generation Computer Systems*, vol. 80, pp. 483–494, 2015.

[3] M. Xie, M. Huang, Y. Bai, and Z. Hu, "The anonymization protection algorithm based on fuzzy clustering for the ego of data in the internet of things," *Journal of Electrical and Computer Engineering*, vol. 2017, Article ID 2970673, 10 pages, 2017.

[4] S. Sciancalepore, G. Oligeri, G. Piro, G. Boggia, and R. Di Pietro, "EXCHANge: Securing IoT via channel anonymity," *Computer Communications*, vol. 134, pp. 14–29, 2019.

[5] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *Proceedings of the 23rd USENIX Security Symposium*, USENIX Association, San Diego, CA, USA, 2014.

[6] O. Berthold, H. Federrath, and S. Köpsell, "Web MIXes: a system for anonymous and unobservable internet access," in *Proceedings of the International Workshop on Designing Privacy Enhancing Technologies: Design Issues in Anonymity and Unobservability*, vol. 2009 of *Lecture Notes in Computer Science*, pp. 115–129, Springer, 2000.

[7] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet: A Distributed Anonymous Information Storage and Retrieval System," in *Proceedings of International Workshop on Designing Privacy Enhancing Technologies: Design Issues in Anonymity and Unobservability*, vol. 2009 of *Lecture Notes in Computer Science*, pp. 46–66, Springer, 2000.

[8] B. Zantout and R. Haraty, "I2P data communication system," in *Proceedings of the 10th International Conference on Networks*, pp. 401–409, 2011.

[9] J. Al-Muhtadi, M. Qiang, K. Saleem, M. AlMusallam, and J. J. Rodrigues, "Misty clouds—A layered cloud platform for online user anonymity in Social Internet of Things," *Future Generation Computer Systems*, vol. 92, pp. 812–820, 2019.

[10] A. Das, N. Borisov, and E. Chou, "Every Move You Make: Exploring Practical Issues in Smartphone Motion Sensor Fingerprinting and Countermeasures," *Proceedings on Privacy Enhancing Technologies*, vol. 2018, no. 1, pp. 88–108, 2018.

[11] X. Cai, X. C. Zhang, B. Joshi, and R. Johnson, "Touching from a distance: Website fingerprinting attacks and defenses," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security, CCS 2012*, pp. 605–616, October 2012.

[12] T. Wang and I. Goldberg, "Improved website fingerprinting on Tor," in *Proceedings of the the 12th ACM workshop*, pp. 201–212, Berlin, Germany, November 2013.

[13] M. Juarez, S. Afroz, G. Acar, C. Diaz, and R. Greenstadt, "A critical evaluation of website fingerprinting attacks," in *Proceedings of the 21st ACM Conference on Computer and Communications Security, CCS 2014*, pp. 263–274, November 2014.

[14] R. Nithyanand, X. Cai, and R. Johnson, "Glove: a bespoke website fingerprinting defense," in *Proceedings of the 13th Workshop on Privacy in the Electronic Society, WPES 2014, in Conjunction with the ACM Conference on Computer and Communications Security, ACM CCS 2014*, pp. 131–134, 2014.

[15] A. Kwon, M. AlSabah, D. Lazar et al., "Circuit fingerprinting attacks: passive deanonymization of Tor hidden services," in *Proceedings of the 24th USENIX Security Symposium*, pp. 287–301, 2015.

[16] T. Wang and I. Goldberg, "On realistically attacking Tor with website fingerprinting," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 4, pp. 21–36, 2016.

[17] R. Overdorf, M. Juarez, G. Acar, R. Greenstadt, and C. Diaz, "How unique is your .onion? An analysis of the fingerprintability of tor onion services," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2021–2036, USA, November 2017.

[18] G. Cherubin, J. Hayes, and M. Juarez, "Website Fingerprinting Defenses at the Application Layer," *Proceedings on Privacy Enhancing Technologies*, vol. 2017, no. 2, pp. 186–203, 2017.

[19] X. Luo, P. Zhou, E. W. W. Chan et al., "HTTPOS: Sealing information leaks with browser-side obfuscation of encrypted flows," in *Proceedings of the 2011 Network and Distributed System Security Symposium*, San Diego, CA, USA, 2017.

[20] E. Brill and C. R. Moore, "An improved error model for noisy channel spelling correction," in *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics (ACL '00)*, pp. 286–293, October 2000.

[21] F. J. Damerau, "A technique for computer detection and correction of spelling errors," *Communications of the ACM*, vol. 7, no. 3, pp. 171–176, 1964.

[22] M. Li, M. Zhu, Y. Zhang, and M. Zhou, "Exploring distributional similarity based models for query spelling correction," in *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, COLING/ACL 2006*, pp. 1025–1032, Australia, July 2006.

[23] Wireshark, "Wireshark network protocol analyzer," 2017, http://www.wireshark.org/.