Journal of Information Processing Vol.27 525–535 (Sep. 2019)

[DOI: 10.2197/ipsjjip.27.525]

Recommended Paper

Detecting Dynamic IP Addresses and Cloud Blocks Using the Sequential Characteristics of PTR Records

Tomofumi Nakamori[1,†1,a)] Daiki Chiba[2,b)] Mitsuaki Akiyama[2] Shigeki Goto[1]

Received: November 26, 2018, Accepted: June 11, 2019

Abstract: Malware-infected hosts are used to conduct many types of cyberattacks. Most of such malware-infected hosts are end-user devices such as PCs, mobile devices and Internet of Things (IoT) devices. In Internet protocol (IP), the IP addresses of most end users are dynamic IP addresses that are allocated by Internet service providers (ISPs). Some countermeasures against cyberattacks use IP addresses as unique indicators of infected hosts. However, due to certain configurations and policies of the particular ISP, the same dynamic IP address is not always reallocated to the same host. Therefore, the accurate detection of dynamic IP address blocks is necessary to take appropriate countermeasures against cyberattacks. Furthermore, attacks from hosts on a cloud block have been observed. A cloud block is defined as an IP address block used in cloud services. Cloud service administrators can take countermeasures against these attacks, such as restricting suspicious traffic and disabling the accounts of suspicious users. Thus, to implement such appropriate countermeasures, the detection of cloud blocks is also important. Using conventional methods, dynamic IP address blocks can be detected by matching a PTR record of the target IP address with predefined keywords that indicate dynamic allocation. However, these keywords do not always match the PTR records of dynamic IP addresses. On the contrary, they can also falsely match non-dynamic IP addresses. In this study, we propose a new method for detecting dynamic IP address blocks more accurately and with a greater coverage rate than conventional methods. Our method focuses on a unique feature of dynamic IP addresses, namely that the PTR records of dynamic IP address blocks are sequentially configured by network administrators. In cloud block detection, our method uses a unique feature of cloud blocks, namely that the users of a cloud service can manually configure the PTR records of the hosts on the cloud blocks. The performance of our method was validated through evaluation using real and manually labeled data. We found that many hosts with the dynamic IP addresses detected by our method send malicious traffic through validation using real traffic data collected in a large-scale darknet.

Keywords: dynamic IP addresses, PTR record, cloud

1. Introduction

Various types of cyberattacks, such as sending spam emails, conducting denial of service (DoS) attacks, and stealing personal information, are sent from hosts infected with malware. Most of such malware-infected hosts are end-user devices such as PCs mobile devices and Internet of things (IoT) devices. In today's Internet, the IP addresses of most end users are dynamic IP addresses allocated by Internet service providers (ISPs). Some countermeasures against cyber attacks use IP addresses as unique indicators of infected hosts [2]. For example, the source IP addresses associated with previously detected malicious activity are added to blacklists [3], [4], [5]. However, the same dynamic IP address is not always reallocated to the same host due to the configuration and policies of the particular ISP. Hence, it is not always correct to consider a dynamic IP address as a permanent malicious host and put it on a blacklist. As another example, source IP addresses of malware-infected hosts [6], [7] and IoT devices [8], [9] in ISPs are used for user notifications. In such notifications, a malicious IP address is first detected and then the corresponding user for sending a notification is determined. The notification should be accurate and should not be sent to the wrong user. If the IP addresss is static, the user is easily determined. However, if the IP address is dynamic, the detemination of user is more complex, that is, the user should be determined based on the timestamp and communication logs. Therefore, the accurate detection of dynamic IP address blocks is helpful when implementing appropriate IP address-based countermeasures against cyberattacks. For example, if a dynamic IP address used by a malicious host is detected, the IP address should not necessarily be placed on a blacklist and notification lists since that IP address may be reallocated to a non-malicious host, which results in a false positive.

Conventional methods detect dynamic IP address blocks by matching a Pointer (PTR) record of the target IP address with predefined keywords that indicate dynamic allocation [10], [11], [12], [13]. A PTR record is a reverse DNS record defined in DNS configuration files, and it shows the name of the IP address.

[1] Waseda University, Shinjuku, Tokyo 169–8555, Japan

[2] NTT Secure Platform Laboratories, Musashino, Tokyo 180–8585, Japan

[†1] Presently with NTT Communications Corporation

[a)] nakamori.gt@gmail.com

[b)] daiki.chiba@ieee.org

The preliminary version of this paper was published at Computer Security Symposium 2018 (CSS2018), Oct 2018. The paper was recommended to be submitted to Journal of Information Processing (JIP) by the program chair of CSS2018.

This paper is an extended version of a paper published in APAN Research Workshop 2018 [1].

Such conventional methods detect dynamic IP address blocks using keywords such as `dhcp` and `dynamic`. However, the predefined keywords do not always match the PTR records of dynamic IP addresses. On the contrary, the keywords sometimes falsely match the PTR records of non-dynamic IP addresses. To resolve this problem, we propose a new method to detect dynamic IP address blocks more accurately and with a greater coverage rate than conventional methods. The coverage rate is defined as the ratio of the IP addresses that could be judged as dynamic or static to all input IP addresses. Our method focuses on a unique feature of dynamic IP addresses, namely that the PTR records of dynamic IP address blocks are sequentially configured by network administrators. Our method can accurately detect dynamic IP address blocks that do not match with the predefined keywords. In other words, our method increases the detection coverage of dynamic IP address blocks. Moreover, cyberattacks from hosts on a cloud block have been observed. We define a cloud block as an IP address block used in a cloud service. Cloud service administrators can take countermeasures against these attacks, such as restricting suspicious traffic and disabling the accounts of suspicious users. Thus, to take such appropriate countermeasures, detecting cloud blocks is also important. In cloud detection, our method uses a unique feature of cloud blocks, namely that the users of a cloud service can manually configure the PTR records of the hosts on the cloud blocks. Real and manually labeled data was used to evaluate our method and results indicated that compared with the conventional methods, our method can detect dynamic IP address blocks more accurately and with a greater coverage rate. In addition to a particularly high accuracy rate, the coverage rate of our method is considerably higher than those of conventional methods. Results of our evaluation further indicated that our method can detect cloud blocks accurately. To validate our method, real traffic data collected in a large-scale darknet was used, and results revealed that about 80% of attacks used dynamic IP addresses. The main contributions of our study are summarized as follows.

- A new method is proposed to detect dynamic IP address blocks using the sequential characteristics of the PTR records of IP addresses.
- The characteristics of dynamic/static IP addresses were verified using large and real IP address data.
- Our evaluation using real and manually labeled data indicated that compared to conventional methods, our method can detect dynamic IP address blocks more accurately and with a greater coverage rate.
- Results of our study further revealed that our method can detect cloud blocks based on the information of the IP address blocks which are created by our method.

The remainder of this paper is organized in the following manner. First, we define the types of IP addresses in Section 2. Then, in Section 3, we explain our method that detects dynamic IP address blocks based on the sequential characteristics of PTR records. Our evaluation results are presented in Section 4. In Section 5, we discuss the accuracy associated with all IP addresses. Section 6 discusses related work. Finally, Section 7 summarizes the conclusions of our study.

2. Background

2.1 Method of Assigning IP Addresses

2.1.1 Static IP Address Blocks

Generally, a static IP address is used for servers, such as web servers, DNS servers, and FTP servers, which provide services. Many PTR records corresponding to these IP addresses are named by the types of services. For example, `www` is used for web servers, `ns` and `dns` are used for DNS servers, and `ftp` is used for FTP servers. Network devices, such as routers and gateways, are also named similarly. For example, `router`, `rt`, and `rtr` are used for routers and `gateway` and `gw` are used for gateways. These PTR records are set manually for each server. Network administrators often use numbers that are meaningful to humans or good numbers for the IP address or PTR record. Typically, only a small number of static IP addresses among all IP addresses allocated to an organization are used.

2.1.2 Dynamic IP Address Blocks

Currently, the Internet is used by a massive number of people, most of whom are end users. Individuals rarely use the same Internet environment continuously for a long time. In this case, assigning static IP addresses is not efficient because the IP address remains unused or occupied for a long time. Thus, using dynamic IP addresses, which involves dynamic allocation of IP addresses when users need them, is one way to solve the problem. For example, an ISP allocates a dynamic IP address to a user when they start a point-to-point over ethernet (PPPoE) session. The same dynamic IP address is not always reallocated to the same host depending on the settings and policies of the particular ISP and the timing of reallocation. Thus, an end user corresponding to a dynamic IP address is not always the same user or device. Preferably such dynamic IP address blocks should be detected so that appropriate countermeasures can be taken against cyberattacks.

2.2 Advantages of Dynamic IP Address Block Detection

Most IP addresses utilized in cyberattacks are considered to be dynamic IP addresses. This is because attackers mainly use a botnet, which comprises malware-infected hosts (bots) to prevent the attackers from being identified. These bots infect the hosts of end users with malware by exploiting the vulnerabilities of software and device hardware. When the host is infected with malware, it begins connecting to command & control (C&C) servers, and joins an existing botnet. A botnet is used for conducting various types of cyberattacks, such as Denial of Service (DoS) attacks and spamming. For example, the Mirai botnet affected embedded and IoT devices with common and weak usernames or passwords or both and then used these devices for massive distributed DoS attacks [14], [15]. When detecting bots and taking countermeasures against botnets, source IP addresses are widely used as unique indicators. However, if the IP address is dynamic, the same host does not always use the same IP address, making it difficult to take appropriate countermeasures. Therefore, detecting dynamic IP address blocks is helpful in taking accurate and appropriate countermeasures against cyberattacks.

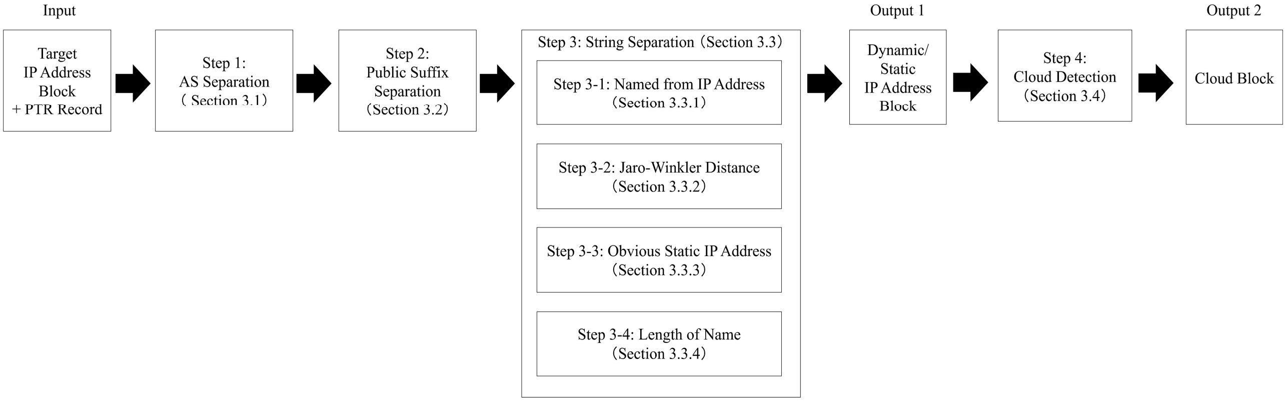

Input
Target IP Address Block + PTR Record
Step 1: AS Separation（Section 3.1）
Step 2: Public Suffix Separation（Section 3.2）
Step 3: String Separation（Section 3.3）
Step 3-1: Named from IP Address（Section 3.3.1）
Step 3-2: Jaro-Winkler Distance（Section 3.3.2）
Step 3-3: Obvious Static IP Address（Section 3.3.3）
Step 3-4: Length of Name（Section 3.3.4）
Output 1
Dynamic/ Static IP Address Block
Step 4: Cloud Detection（Section 3.4）
Output 2
Cloud Block

Fig. 1 Overview of our method.

2.3 Cloud Block

Most end-user mobile and IoT devices that are targeted by malware do not explicitly configure the PTR records of their IP addresses. Moreover, most of the Dynamic IP addresses allocated to these hosts are usually managed by the particular ISP. In this case, end-users cannot manually configure the PTR records on their devices. On the contrary, in the cloud blocks, users can configure their own PTR records to use the hosts in the cloud services in many ways. In fact, if a user wants to use the host as a mail server or web server, they must configure the PTR record. Taking Amazon Web Service (AWS) as an example, the users can configure the PTR record [16]. When the malicious traffic is sent from hosts on the cloud block, cloud service administrators can take countermeasures against these attacks such as restricting suspicious traffic and disabling the accounts of suspicious users. Thus, detecting cloud blocks provides important information that makes it possible to take appropriate countermeasures against cyberattacks.

3. Our Method

This section describes our proposed method for detecting dynamic IP address blocks using the results of reverse DNS (rDNS) lookups of IPv4 addresses. Our target IP addresses were all IP addresses included in the rDNS results. The difference between our method and conventional methods is that detection is not only performed individually for the rDNS result of one IP address but also by referring to the rDNS results of the preceding and succeeding IP addresses. Our method considers sequential characteristics when a network administrator manages IP addresses. When allocating a substantial number of dynamic IP addresses, a continuous number is often automatically assigned to the PTR records, and these PTR records often display a high degree of similarity. Therefore, in our method, an IP address block that has PTR records with a high degree of similarity is detected as a dynamic IP address block. Using conventional methods, numerous unidentifiable IP addresses cannot be detected using keyword matching and regular expressions. However, our method makes it possible to detect whether such addresses are dynamic or static. **Figure 1** provides an overview of our method. Our method includes four steps. Note that the four steps and their procedures are carefully designed to reflect the fact of how IPv4 addresses are managed by different organizations in today's Internet. First, a pair of target IP address blocks and their corresponding PTR records are used as input. Next, the delimiter position of sequential address blocks called segments is recognized and the segments are separated. In this step, two PTR records are compared, namely the PTR record of an IP address and that of the adjacent address in order. After that, we output the information of the detected dynamic or static IP address blocks. Finally, based on this information, the information of the detected cloud blocks becomes the final output. Note that we skip IP addresses that have no corresponding PTR records. We will explain each step comprehensively in the following sections. **Figure 2** shows an example of our method and the output. The horizontal line with the step number shows the delimiter positions recognized using our method. *Output 1* is obtained when Step 3 is completed; *Output 1* shows whether the IP address block is dynamic or static. *Output 2* is obtained when Step 4 is completed; *Output 2* indicates whether the IP address block belongs to a cloud block. Our method needs some predefined parameters in Step 3-3 and Section 3.3.3. When designing each step, we used a preliminary dataset which is a subset of the datasets shown later in Section 4. The preliminary dataset is composed of randomly sampled 10 IP address blocks which include up to 80 IP addresses in total (only 0.002% of total IP addresses in the datasets). We confirmed that the sampled IP addresses were not biased in terms of their corresponding countries, network operators, and service types. In order to provide ample generalizing capability with our method, we carefully monitored the parameter and their corresponding results to determine the optimal parameters.

3.1 Step 1: AS Separation

From the information of the target IP address block (e.g., `192.0.2.0/24`), we investigate the AS number to which each IP address belongs. We referred to the MaxMind database [17] to examine information of the AS number. If the target IP address block contains IP addresses belonging to different AS numbers, the segment is divided before and after the IP address because the management policy and method of the IP address vary significantly between different AS numbers. In Fig. 2, `192.0.2.128` and `192.0.2.129` are separated because the AS numbers are different. The IP address block from `192.0.2.0` to `192.0.2.128`

	Setting Value					Proposed Method	
	IP Address	Network	ASN	Public Suffix	PTR Record	Output 1	Output 2
	192.0.2.0	192.0.2.0/25	AS64496	example.net	host1.example.net	Dynamic	Not Cloud
	192.0.2.1	192.0.2.0/25	AS64496	example.net	host2.example.net	Dynamic	Not Cloud
	…	…	…	…	…	…	…
	192.0.2.20	192.0.2.0/25	AS64496	example.net	host20.example.net	Dynamic	Not Cloud
Step 2							
	192.0.2.21	192.0.2.0/25	AS64496	example.com	test21.example.com	Dynamic	Cloud
	192.0.2.22	192.0.2.0/25	AS64496	example.com	test22.example.com	Dynamic	Cloud
	192.0.2.23	192.0.2.0/25	AS64496	example.com	test23.example.com	Dynamic	Cloud
	192.0.2.24	192.0.2.0/25	AS64496	example.com	test24.example.com	Dynamic	Cloud
	192.0.2.25	192.0.2.0/25	AS64496	example.com	test25.example.com	Dynamic	Cloud
Step 3							
	192.0.2.26	192.0.2.0/25	AS64496	example.com	www1.example.com	Static	Cloud
	192.0.2.27	192.0.2.0/25	AS64496	example.com	mail.example.com	Static	Cloud
	192.0.2.28	192.0.2.0/25	AS64496	example.com	ftp.example.com	Static	Cloud
Step 3							
	192.0.2.29	192.0.2.0/25	AS64496	example.com	test29.example.com	Dynamic	Cloud
	…	…	…	…	…	…	Cloud
	192.0.2.128	192.0.2.128/25	AS64496	example.com	test128.example.com	Dynamic	Cloud
Step 1							
	192.0.2.129	192.0.2.128/25	AS64497	example.org	test129.example.org	Dynamic	Not Cloud
	…	…	…	…	…	…	…

Fig. 2 Example of the output of our method.

is a large segment that can be separated in this step.

3.2 Step 2: Public Suffix Separation

If the segment obtained in Step 1 contains IP addresses with different public suffixes, the segment is divided before and after the IP address. We used the Public Suffix List (PSL) [18] provided by the Mozilla Foundation to extract the Public Suffix. In Fig. 2, the IP address segment is separated between `192.0.2.20` and `192.0.2.21` because the Public Suffixes of the PTR records are different. The IP address block from `192.0.2.0` to `192.0.2.20` and that from `192.0.2.21` to `192.0.2.128` are segments that can be separated in this step.

3.3 Step 3: String Separation

In this step, we focus on a string in a PTR record of an IP address, which belongs to the segment obtained in Step 2. We exclude the public suffix (e.g., `example.net`) from the PTR record (e.g., `host1.example.net`) and consider only the substring (e.g., `host1`) in this step. If there are five or more IP addresses with similar substrings, the IP address block is detected as a dynamic IP address block. Otherwise, it is detected as a static block. We confirmed that there are many cases in which approximately four static IP addresses line up with the PTR records: `www1`, `www2`, `www3`, and `www4`. We empirically set the threshold at five so that they are not detected. For example, in the case shown in Fig. 2, this step separates the segment from `192.0.2.21` to `192.0.2.128` until Step 2. PTR records having string similarity are from `192.0.2.21` to `192.0.2.25` and from `192.0.2.29` to `192.0.2.128`. Thus, these segments are detected as dynamic IP address blocks. The segment from `192.0.2.26` to `192.0.2.28` does not exhibit string similarity; hence, it is detected as a static IP address block. Our method considers four string similarity criteria and corresponding sub steps, namely Step 3-1 (named from IP address), Step 3-2 (Jaro–Winkler distance), Step 3-3 (obvious static IP address), and Step 3-4 (length of the name), in order. In Step 3-1, if the PTR records contain IP addresses, they are considered sequential and Step 3 is complete. If the PTR records do not contain IP addresses, our method proceeds to Step 3-2. In Step 3-2, if two PTR records have a high degree of similarity, they are considered sequential and our method proceeds to Step 3-3. If two PTR records do not have a high degree of similarity, Step 3 is complete at this point. If two PTR records complete Step 3-3, these are not considered static IP addresses and we proceed to Step 3-4. If this is not the case, Step 3 is complete here. If two PTR records complete Step 3-4, it is judged that they are sequential. If two PTR records do not complete Step 3-4, they are not sequential. We explain each criterion or sub-step in detail in the following sections.

3.3.1 Step 3-1: Named from IP Address

In network management, IP addresses are often set as part of the PTR records. We determine whether a PTR record has an IP address-based value to detect sequential records in these types of PTR records. We extract the numerical value of the IP addresses that are separated by “`-`”, “`_`”, “`.`” with a regular expression. We also extract the numerical value of the IP addresses that are configured in reverse order with the same process. If the PTR records contain IP address-based values, our method characterizes these records as having string similarity. Considering the possibility of erroneous detection if only a numerical value is included, this operation is available when the extracted value exactly matches the target IP address.

3.3.2 Step 3-2: Jaro–Winkler Distance

When setting a dynamic IP address in a particular block, the network administrator often names the PTR records as “prefix + number” (e.g., `host1.example.net` and `host2.example.net`) in a systematic and automatic manner. Our method focuses on this sequential characteristic. The Levenshtein distance is a way of determining similarity between strings. This measure repeatedly inserts, replaces, and deletes characters in one character string until it reaches another character string and determines the distance based on the shortest time for this operation. However, it is difficult to determine an appropriate threshold. For a prefix + number PTR record, the number differs between sequential PTR records. When the digit in the number changes, the Levenshtein distance is the largest. By increasing the threshold value, detection is possible even if more digits are set. However, dissimilar PTR records are sometimes judged similar. For example, when the threshold is 8, `test9999999.example.net` and `test10000000.example.net` are sequential; thus, they can be detected. However, `invalid.example.net` and `localhost.example.net` are also judged similar in this threshold setting (the threshold is 8). On the other hand, if the threshold is too low, this method can erroneously detect when the PTR record is a short name (e.g., `ns1.example.net` and `www.example.net`).

Thus, to solve this problem, the Jaro–Winkler distance [19], [20], [21], [22], [23] is used instead of the Levenshtein distance, and the python package [24] was used to calculate the Jaro-Winkler distance in our paper. The Jaro–Winkler distance is a measure of string similarity that considers the prefix of a character string. It is defined as 1.0 when two strings are identical and 0 when two strings do not match at all. This method is based on the Jaro distance [20], [21], [22], which is defined as follows:

$$\Phi_J = W_1 \cdot c/d + W_2 \cdot c/r + W_\tau \cdot (c - \tau)/c \qquad (1)$$

where W_1 is the weight applied to the character of the first character string,
W_2 is the weight applied to the character of the second character string,
W_τ is the weight applied to the transpositions,
d is the length of the first character string,
r is the length of the second character string,
τ is the number of replaced characters, and
c is the number of common characters in a pair of strings.
If $c = 0$, then $\Phi_J = 0$.

c is the number of matched characters when the positions of two characters are within $(max(d, r)/2) - 1$. In practice, W_1, W_2, and $W\tau$ are set to $1/3$. Based on this definition, the Jaro–Winkler distance is defined as follows:

$$\Phi_{JW} = \Phi_J + p \cdot (0.08) \cdot (1 - \Phi_J) \qquad (2)$$

where p is the number of common characters in the prefix of a pair of strings and is no more than 4. We empirically set the weight applied to p at 0.08. The Jaro–Winkler distance is generally used to detect misspelled words or typographical errors. It is also used in predictive searching to suggest candidates for prediction with only a part of a character string when searching on the Web. The possibility of misjudging dissimilar strings as similar is low when using this method because it adds weight to the prefix which reflects sequential characteristics of the strings, and an appropriate classification can be obtained even if the PTR record has a short name. Using this method, we were able to classify similar PTR records using a threshold based on actual data. In this case, the threshold was set to 0.91, and if the Jaro–Winkler distance exceeded 0.91, the strings were classified as similar. **Figures 3** and **4** show examples of similar and dissimilar strings, respectively. In Fig. 3, `test` of the “First String” and “Second String” are the prefix and the Jaro–Winkler distance is 0.91; thus, our method judges that the strings are similar. In Fig. 4, since the strings do not have a common prefix, the Jaro–Winkler distance is 0. Thus, our method judges that the strings are not similar.

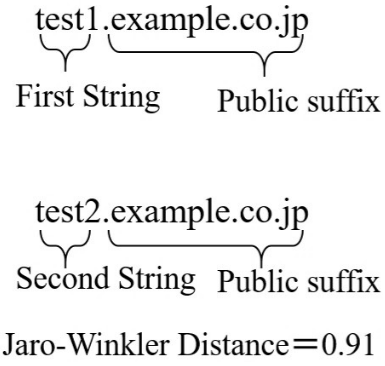


Fig. 3 Example of similar strings.

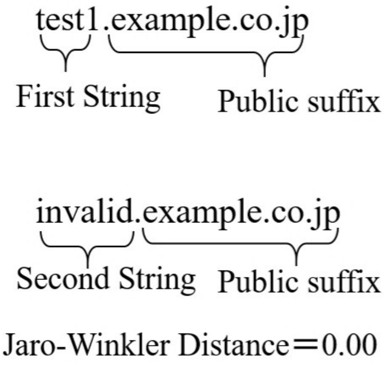


Fig. 4 Example of dissimilar strings.

3.3.3 Step 3-3: Obvious Static IP Address

Even if the above process is performed, it is still possible that a certain number of static IP addresses could be misjudged. If the IP addresses that are detected as dynamic IP addresses contain a keyword that indicates the IP address is a static IP address, it is classified as static. In this case, the static keyword was set with reference to the keyword used in conventional methods [10], [11], [12], [13]. The specific keyword list is presented in Section 4.2.

3.3.4 Step 3-4: Length of the Name

In an IP address block, PTR records with different characteristics from the preceding and following PTR records may be set as static IP addresses in some cases. To ensure that such an IP address is not erroneously determined as dynamic, the length of the preceding PTR record is considered in this substep. If the length of the PTR record differs significantly from that of the following

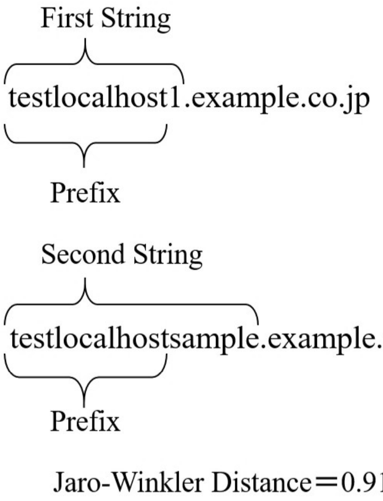

First String
testlocalhost1.example.co.jp
Prefix
Second String
testlocalhostsample.example.co.jp
Prefix
Jaro-Winkler Distance＝0.91

Fig. 5 Example of failure when comparing long strings.

PTR record, the records are treated as dissimilar. This case is applicable when the length of one string is greater than that of the other by three or more characters. When naming PTR records, numbers with up to three digits are often used. Therefore, the length difference between two names with numbers is within two characters, but the other long character strings differ by more than three characters. Therefore, we set the threshold as 3 characters.

An example of an erroneously judged character string when this process was *not* performed is shown in **Fig. 5**. In Fig. 5, the prefix of both strings is `testlocalhost`, but one of the following strings has a numerical value and the other contains alphabet characters. In other words, the strings are not sequential. However, the Jaro–Winkler distance is 0.91, implying that both strings are identical. This is an erroneous judgment.

3.4 Step 4: Cloud Detection

There are many ways to use the hosts on the cloud block, and users must set a PTR record when the host is used as a mail server or Web server. Therefore, a PTR record that indicates the host is used as static appears in the sequence of the PTR record in a cloud block, thus the PTR records are not sequential. Using this feature and the information of the IP address blocks obtained when Step 3 is complete, our method can detect cloud blocks. Specifically, when the IP address blocks consist of dynamic, static, dynamic then they fulfill this feature. Moreover if the dynamic IP address blocks belong to the same network range then they have a common prefix in the PTR records. In the above case that there are static IP address blocks between dynamic IP address blocks, our method judges that they are used in a cloud block. In Fig. 2, the segment from `192.0.2.21` to `192.0.2.25` and from `192.0.2.29` to `192.0.2.128` have the same prefix “`test`”. The segment from `192.0.2.26` to `192.0.2.28`, which is a static IP address block, is between these two dynamic IP address blocks. Thus, our method judges that the segments from `192.0.2.21` to `192.0.2.128` are used in a cloud block.

4. Evaluation

In this section the validation of our method is explained, detection accuracy is compared, and the cloud detection is evaluated using a real IPv4 address and the corresponding PTR record. Moreover, we validate whether the hosts allocated to dynamic IP addresses that are detected by our method are benign or malicious using a benign dataset and real traffic data collected from a large-scale darknet.

4.1 Dataset

4.1.1 PTR Records of IPv4 Address

To evaluate our method, the IPv4 address and the corresponding PTR record were collected in two ways. First, we used publicly available rDNS results that were published as part of Project Sonar [25]. This is a dataset containing the responses of reverse DNS lookups corresponding to the IPv4 address, and it does not include the results of the IP address defined in the blacklists of the project or private IP addresses. Note that there are no corresponding results for these data, and if an error occurred at the time of scanning, a PTR record did not originally exist or the IP address was unused. We used Project Sonar’s rDNS results for October 18, 2017 in Sections 4.2, 4.3, 4.5. We used the same dataset for August 29, 2018 in Section 4.4. Next, we collected additional rDNS results to supplement the Project Sonar dataset. We sent direct reverse DNS lookup queries from our experimental environment. For these reverse DNS lookup queries, we sent queries at a relatively low rate so that an excessive load was not exerted on the target network. We further restricted our lookups to the data that were not included in the Project Sonar dataset.

4.1.2 Labeled Dataset

To verify the detection results of the dynamic and static IP addresses using our method, we calculated the detection accuracy using a labeled dataset. Unfortunately, there is no ground truth regarding static/dynamic IP addresses. Thus, we need to label the data manually for evaluating our method. Basically, we carefully labeled the data in two ways. One way is relying on the keywords used in conventional methods [10], [11], [12], [13]. We used the keywords to determine reliable static/dynamic IP addresses first and then we expand the target and label similar IP addresses around the reliable IP addresses. The other way is manual labeling by researchers and engineers engaged in network/security services. They used various information regarding the corresponding IP addresses such as PTR records, offered services, and scanned results to determine whether the IP address is static/dynamic. We selected multiple and various regions of IP address blocks in the dataset. **Tables 1**, **2** summarize the details of these datasets. Datasets 1 and 3 include only dynamic IP addresses whose PTR records were set sequentially. Datasets 2, 4, 5, and 6 included both dynamic and static IP addresses. Dataset 7 includes only IP addresses of a cloud block. Dataset 8 includes only IP addresses which were not used in a cloud block.

4.1.3 Benign and Malicious Datasets

To validate whether the hosts allocated to dynamic IP addresses detected by our method are benign or malicious, two datasets were used.

As a benign dataset, the results of forward DNS lookups of domain names in the Alexa Top 1 million sites [26] were used. Highly visited Web sites work on the Web servers and static IP

Table 1 Labeled dataset.

Dataset	Range	Country	ISP	#Total Dynamic IP Addresses	#Total Static IP Addresses
Dataset 1	67.160.0.1/24	USA	Comcast	510	0
Dataset 2	100.17.0.0/22	USA	Verison	1,011	5
Dataset 3	104.0.0.0/22, 104.0.4.0/24	USA	AT&T	1,278	0
Dataset 4	106.72.0.0/15, 106.128.0.1 - 106.185.45.254	Japan	KDDI	3,596,918	1,856
Dataset 5	110.79.255.6 - 110.79.255.254, 110.158.0.1 - 110.158.2.20, 110.158.64.0/18, 110.158.128.0/17	Japan	OCN	44,668	39
Dataset 6	143.0.84.0/22	Brazil	Sim Telecom	992	9

Table 2 Labeled dataset for cloud detection.

Dataset	Range	Country	ISP	#Cloud IP Address	#Not Cloud IP Address
Dataset 7	175.41.129.0/24	USA	Amazon	256	0
Dataset 8	69.104.56.0/24	USA	AT&T	0	256

addresses are allocated to these servers, and they could be benign sites. In this paper, we used this dataset for June 12, 2018.

As a malicious dataset, we used darknet traffic data of NICTER Dataset 2018 which was provided by NICT [27]. The darknet is unused IP address space, so the traffic originally should not be observed in the darknet. In fact, a significant amount of malicious traffic was observed. For example, there were scanning packets, reply packets of DDoS, and preparation for reflection attacks [28]. As described in Section 2, these attacks were often sent by bots. In this paper, we investigated the source IP addresses in this traffic data from September 18 to October 1, 2018.

Considering the features of these two datasets, we hypothesized that the ratio of static IP addresses detected by our method is high for the Alexa dataset and the ratio of dynamic IP addresses detected by our method is high for the NICTER dataset.

4.2 Comparison of the Accuracy of Detecting Dynamic IP Address

We evaluated the detection accuracy of our method using a labeled dataset and investigated whether the dynamic and static IP address blocks were correctly judged by the proposed method. For comparison, using the same dataset, we also verified the detection accuracy of conventional methods that employ keyword matching [10], [11], [12], [13] and regular expressions [29].

The conventional methods detected the dynamic and static IP addresses using the corresponding predefined keywords in the PTR record [10], [11], [12], [13]. We first prepared both the static and dynamic keywords based on the above conventional methods. The keywords used herein were as follows.

- Static Keywords: `server, srv, svr, mx, mail, smtp, www, ns, ftp, router, rtr, rt, gateway, gw, dns, sw, and test`
- Dynamic Keywords: `dialup, modem, cable, hsb, dyn, dynamic, wireless, pool, access, dhcp, dialup, ppp, and adsl`

The conventional method using regular expressions was S25R [29]. S25R is a combination of regular expressions or rules for detecting end-users who sent spam emails. S25R investigated whether a PTR record of the target IP address matched the rules. Although S25R has general rules and a blacklist and whitelist, the blacklist and whitelist may detect mail servers in some cases. Thus, for our evaluation, we only used the general rules. The rules of regular expressions are as follows.

Table 3 Relationships among terms.

		Actual Value	
		Dynamic	Static
Detection	Dynamic	True Positive (TP)	False Positive (FP)
Outcome	Static	False Negative (FN)	True Negative (TN)

- `^[^.]*[0-9][^0-9.]+[0-9].*\.`
- `^[^.]\*[0-9]\{5\}`
- `^([^.]+\.)?[0-9][^.]*\.[^.]+\..+\.[a-z]`
- `^[^.]*[0-9]\.[^.]*[0-9]-[0-9]`
- `^[^.]*[0-9]\.[^.]*[0-9]\.[^.]+\..+\.`
- `^(dhcp|dialup|ppp|[achrsvx]?dsl)[^.]*[0-9]|`

Here, we define the metrics used in our evaluation. The aim of our method is to detect dynamic IP addresses. Thus, we treated the dynamic IP addresses as *positive* and the static IP addresses as *negative*. **Table 3** summarizes the relationships among the terms used in our evaluation. True positive (TP) indicates that the dynamic IP address was correctly detected as a dynamic IP address. False negative (FN) indicates that the dynamic IP address was incorrectly determined as a static IP address. False positive (FP) indicates that the static IP address was incorrectly determined as a dynamic IP address. True negative (TN) indicates that the static IP address was correctly detected as a static IP address. TP rate (TPR) is the rate at which the dynamic IP address was correctly determined as a dynamic IP address; TPR is defined as $TP/(TP + FN)$. FN rate (FNR) is the rate at which the dynamic IP address was incorrectly determined as a static IP address; FNR is defined as $FN/(TP + FN)$. FP rate (FPR) is the rate at which the static IP address was incorrectly determined as a dynamic IP address; FPR is defined as $FP/(TN + FP)$. TN rate (TNR) is the rate at which the static IP address was correctly determined as a static IP address; TNR is defined as $TN/(TN + FP)$. We also defined the coverage rate, which is the proportion of the IP addresses that could be judged as dynamic or static in the entire input IP address block.

Table 4 summarizes the evaluation results comparing the de-

Table 4 Comparison of detection accuracy.

Dataset 1

	#TPs	#FNs	#Total Detected Dynamic IPs	#FPs	#TNs	#Total Detected Static IPs	TPR	FNR	FPR	TNR	Cover Rate
Our Method	510	0	510	0	0	0	1.000	0.000	0.000	0.000	1.000
Keyword Matching	0	0	0	0	0	0	0.000	0.000	0.000	0.000	0.000
S25R	510	0	510	0	0	0	1.000	0.000	0.000	0.000	1.000

Dataset 2

	#TPs	#FNs	#Total Detected Dynamic IPs	#FPs	#TNs	#Total Detected Static IPs	TPR	FNR	FPR	TNR	Cover Rate
Our Method	1,011	0	1,011	0	5	5	1.000	0.000	0.000	1.000	1.000
Keyword Matching	759	0	759	0	0	0	1.000	0.000	0.000	0.000	0.747
S25R	1,011	0	1,011	4	0	4	1.000	0.000	1.000	0.000	0.999

Dataset 3

	#TPs	#FNs	#Total Detected Dynamic IPs	#FPs	#TNs	#Total Detected Static IPs	TPR	FNR	FPR	TNR	Cover Rate
Our Method	1,278	0	1,278	0	0	0	1.000	0.000	0.000	0.000	1.000
Keyword Matching	0	0	0	0	0	0	0.000	0.000	0.000	0.000	0.000
S25R	1,278	0	1,278	0	0	0	1.000	0.000	0.000	0.000	1.000

Dataset 4

	#TPs	#FNs	#Total Detected Dynamic IPs	#FPs	#TNs	#Total Detected Static IPs	TPR	FNR	FPR	TNR	Cover Rate
Our Method	3,594,396	2,522	3,596,918	327	1,529	1,856	0.999	0.001	0.176	0.824	1.000
Keyword Matching	1,041,779	640	1,042,419	0	405	405	0.999	0.001	0.000	1.000	0.290
S25R	3,596,841	0	3,596,841	489	0	489	1.000	0.000	1.000	0.000	0.999

Dataset 5

	#TPs	#FNs	#Total Detected Dynamic IPs	#FPs	#TNs	#Total Detected Static IPs	TPR	FNR	FPR	TNR	Cover Rate
Our Method	44,668	0	44,668	0	39	39	1.000	0.000	0.000	1.000	1.000
Keyword Matching	0	0	0	2	29	31	0.000	0.000	0.065	0.936	0.001
S25R	44,668	0	44,668	2	0	2	1.000	0.000	1.000	0.000	0.999

Dataset 6

	#TPs	#FNs	#Total Detected Dynamic IPs	#FPs	#TNs	#Total Detected Static IPs	TPR	FNR	FPR	TNR	Cover Rate
Our Method	991	1	992	0	9	9	0.999	0.001	0.000	1.000	1.000
Keyword Matching	0	0	0	0	7	7	0.000	0.000	0.000	1.000	0.007
S25R	0	0	0	0	0	0	0.000	0.000	0.000	0.000	0.000

tection accuracy of all the labeled dataset. Since S25R cannot detect a static IP address, TP and FN were always 0 for this method. The TPRs of our method were extremely high, ranging from 0.999 to 1.000 for every dataset. The TPRs of the conventional keyword-matching method were also high. The coverage rate was the highest: 0.747 for Dataset 2, 0.290 for Dataset 4, and the other was virtually 0. This indicated that keyword matching could not detect many dynamic and static IP addresses since the keywords were predefined. The TPRs of the S25R method were also high, but the coverage rate of S25R for Dataset 6 was 0. This indicated that S25R also could not detect IP addresses in a certain area. Moreover, for Datasets 2, 4, and 5, both the TPR and FPR of S25R were 1.000, and thus, S25R did not work well for these datasets. Our method did not make an erroneous detection in any dataset, except Dataset 4, which was a very large dataset. For Dataset 4, the number of FPs in our method was smaller than that of S25R. Furthermore, our method detected both dynamic and static IP addresses.

Although the coverage rate of our method was 1 (highest), dynamic IP address blocks were detected with an accuracy equal to or better than those of the conventional methods.

Table 5 Result of cloud detection.

	Dataset 7	Dataset 8
#Cloud IP Address	256	0
#Not Cloud IP Address	0	256

4.3 Cloud Detection

Using labeled datasets, we investigated whether the IP address blocks in a cloud block were correctly judged by our proposed method. **Table 5** shows that our method made correct judgement in Dataset 7 which was a cloud block. In Dataset 8, which was not a cloud block, our method judged the IP address blocks were not in a cloud block, and thus, our proposed method also judged correctly in Dataset 8.

4.4 Validation Using Benign and Malicious Dataset

Using the IP address block information in our method and the IP addresses in benign and malicious datasets, we investigated whether the IP addresses were dynamic IP addresses or static IP addresses. **Table 6** shows the results. *Cloud Dynamic* is an IP

Table 6 Validation using benign and malicious dataset.

Dataset	#Dynamic	#Cloud Dynamic	#Total Dynamic	#Static	#Cloud Static	#Total Static	#Total
Alexa	25,933 (8.597%)	86,542 (28.688%)	112,475 (37.285%)	142,985 (47.399%)	46,203 (15.316%)	189,188 (62.715%)	301,663
NICTER	315,786 (27.434%)	598,844 (52.025%)	914,634 (79.460%)	228,146 (19.820%)	8,288 (0.720%)	236,434 (20.540%)	1,151,068

Table 7 Density of IP address blocks.

	#IP Addresses	Length	Density
Dynamic	1,117,666,007	1,219,738,270	0.916
Static	100,581,528	1,246,101,282	0.081

address that is detected as a dynamic IP address used in a cloud block. *Cloud Static* is an IP address that is detected as a static IP address used in a cloud block. In the Alexa dataset, the ratio of total static IP addresses was 62.714%. By contrast, in the NICTER dataset, the ratio of total dynamic IP addresses was 79.460%. Thus, we found that the ratio of static IP addresses was high in the benign dataset and that of the dynamic IP addresses was high in the malicious dataset. Further, among the detected dynamic IP addresses in the malicious dataset, many of them were used in a cloud block. Thus, we confirmed cloud service hosts were being used as malicious hosts.

4.5 Validation of Our Method

Here, we validated the sequential characteristics of the PTR records used in our method by employing a large-scale dataset. Our method is based on the hypothesis that the dynamic IP addresses are set sequentially. If this is true, for a dynamic IP address block, it can be assumed that there are only a few missing IP addresses and that almost all of them are sequential. On the other hand, static IP addresses are set manually; therefore, it is assumed that the network administrator often chooses a number meaningful to humans or a good number for the IP address or PTR record. Therefore, we assumed that the static IP address is sometimes intentionally set to be skipped. To validate these assumptions, we investigated the density of the dynamic and static IP address blocks. The targets of investigation were all the dynamic and static IP address blocks detected by our method for all the IP addresses in the dataset described in Section 4.1.1. The density of an IP address block is defined as follows:

$$density = N/Length \tag{3}$$

where N is the number of IP addresses existing in the IP address block, and

$$\begin{aligned} Length = {} & (The\ Last\ IP\ address\ of\ the\ block) \\ & - (The\ First\ IP\ address\ of\ the\ block)\ +\ 1 \end{aligned}$$

For example, as shown in Fig. 2, consider an IP address block from `192.0.2.21` to `192.0.2.25`. The first IP address was `192.0.2.21` and the last IP address was `192.0.2.25`. Thus, the length was 5 (= `192.0.2.25` − `192.0.2.21` + 1). **Table 7** lists the results. The density of the dynamic IP address blocks was 0.916. Thus, we believe that our hypothesis was essentially correct. However, the density of the static IP address blocks was relatively low (0.081), indicating that static IP addresses were set to be skipped.

Because the static IP addresses were set to be skipped, we assumed that there was a deviation in the setting of static IP addresses. We further investigated whether the number of the fourth octet of an IP address was biased. The targets of investigation were all the static IP address blocks detected by our method from all the IP addresses in the dataset described in Section 4.1.1. **Figure 6** shows the distribution of the fourth octet and the number of static IP addresses. The vertical line in the figure is drawn every fifth octet. The results indicated that spikes occured when the fourth octet was a multiple of 10. **Table 8** lists the top 10 fourth octets of the static IP addresses. Lower numbers and the numbers in which the lower digit is 0 or 1 are prominent in this table. Many IP addresses whose fourth octets were 1 are often used for network gateways, and their PTR records contain `gw` in many cases.

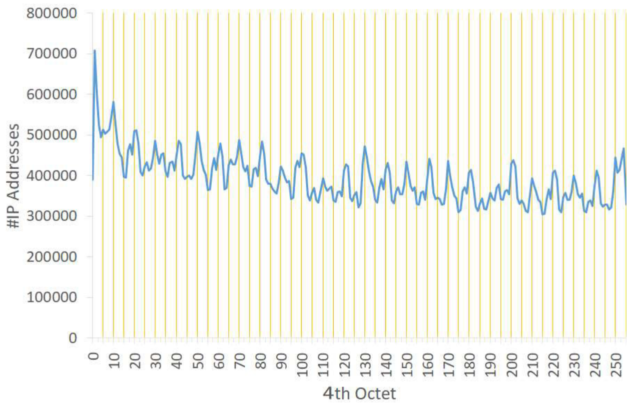


Fig. 6 Distribution of the fourth octet of IP addresses.

Table 8 Top 10 fourth octets of IP addresses.

Fourth Octet	#IP Addresses
1	708,797
2	603,773
10	582,178
9	546,392
3	525,353
11	524,611
8	513,059
5	512,647
21	512,271
20	510,025

5. Discussion

Conceptually, our method is based on the idea that network managers often name the PTR record using a "prefix + number" (e.g., host1.example.net and host2.example.net) approach. Therefore, if the PTR records are "number + common strings," (e.g., 1host.example.net and 2host.example.net) the Jaro–Winkler distance judges two strings as dissimilar in Step 3-2 of our method, which could lead to a low degree of accuracy. In other cases, if the PTR records are configured as "prefix + proper noun," (e.g., testjapan.net and testamerica.net) the hosts corresponding to these PTR records may not be allocated to dynamic IP addresses. However, the Jaro–Winkler distance between the strings will be high because the strings have a common prefix. This means that the strings are judged as similar, and this causes misjudgments by our method. In the regions which are outside our labeled dataset, we think that the accuracy of our method could be lower when PTR records such as those above are used. To improve the accuracy, it would be necessary to investigate which part of the strings has a sequential number. When the strings have a common prefix, it would also be necessary to verify the types of the strings after the prefix. For example, the strings must be checked to determine if they have a sequential number or a non-sequential number such as a proper noun.

As described in Section 4.1.3, traffic of DDoS, reflection attacks, and so on were observed in the darknet traffic data in the NICTER dataset. The source IP addresses of this traffic are spoofed. In our experiment, we did not exclude this traffic. When the darknet traffic data in the NICTER dataset from which the spoofed IP addresses were excluded was used in the experiment, the number of the static IP addresses detected in the darknet traffic data in the NICTER dataset would decrease and the tendency of many dynamic IP addresses in the malicious dataset becomes prominent.

The target of this study is PTR records of IPv4 addresses. PTR records of IPv6 addresses may not be registered since the number of IPv6 addresses is huge. In November 2018, a workaround to this problem was proposed in RFC 8501 [30]. Specifically, a DNS server dynamically creates a PTR record and replies it when the server receives a reverse DNS lookup query. When this workaround is applied, our proposed method can be applied to detect dynamic IP addresses.

6. Related Work

Many researchers have studied and analyzed dynamic IP addresses, their features, and the traffic originating from them. Xie et al. analyzed spam emails using a three-month Hotmail email server log [10]. They found that almost all servers sending spam emails were setup on dynamic IP addresses. They used predefined dynamic keywords (e.g., ppp and dhcp) and investigated whether the keyword matched a PTR record to detect dynamic IP address blocks.

Jin et al. detected dynamic IP address blocks based on active scanning [12]. They clustered real traffic data to analyze dynamic IP address blocks. Keyword matching was used to detect whether the IP address was dynamic or static.

Cai et al. performed block-level clustering to investigate the management of networks [13]. They researched the nature of the dynamic allocation of IP addresses and used predefined keywords to detect dynamic IP addresses. They reported that approximately 40% of /24 blocks were dynamically allocated and showed that the utilization rate differed based on the country.

Richter et al. analyzed a large CDN server log and investigated the activities of IPv4 addresses [11]. In one of their analyses, they surveyed the allocation of the static and dynamic IP addresses. The static and dynamic IP addresses were detected via keyword matching. In total, they detected /24 blocks, 456K dynamic IP address blocks, and 262K static IP address blocks.

Asami analyzed the PTR records of sending spam emails to propose a spam-filtering system [29]. He devised six rules for regular expressions. If one of the rules match the PTR records of email senders, the system rejects the email. The evaluation results revealed that this system can reject 99% of spam emails.

Padmanabhan et al. reported that dynamic IP addresses are often reallocated [2]. They showed that dynamic IP addresses are reallocated because of the policy or setting of the ISPs, disasters, and blackouts. They also observed that some IP addresses were reallocated to IP addresses with different BGP prefixes.

The aforementioned studies did not adopt concepts similar to that of our method wherein the sequential characteristics of PTR records were used to detect dynamic IP address blocks.

7. Conclusion

Herein, we propose a new method to detect dynamic IP address blocks using the sequential characteristics of PTR records corresponding to IPv4 addresses. The proposed method focuses on a unique feature of dynamic IP addresses: or namely that the PTR records of dynamic blocks are sequentially configured by network administrators.

While many conventional methods use predefined keywords to detect dynamic IP address blocks, our method does not rely on such keyword matching for individual IP addresses. Furthermore, our method uses the continuity of the PTR records of the preceding and succeeding IP addresses to detect dynamic IP address blocks. Our evaluation using real and manually labeled data revealed that compared to conventional methods, the proposed method can detect dynamic IP address blocks more accurately and with a greater coverage rate. The results of our study indicated that our method can detect cloud blocks by using the feature that users of cloud services can configure the PTR records at will and the information of IP address blocks of our method.

In our validation using benign and malicious datasets, we found that the IP addresses allocated to the malicious hosts were mostly dynamic IP addresses and many hosts on cloud services were being misused. Hence, the results from detecting dynamic IP addresses and cloud detection using the proposed method can aid us in taking appropriate countermeasures against cyberattacks associated with IP addresses.

Acknowledgments A part of this work was supported by JSPS Grant-in-Aid for Scientific Research B, Grant Number JP16H02832.

References

[1] Nakamori, T., Chiba, D., Akiyama, M. and Goto, S.: Detecting Dynamic IP Addresses Using the Sequential Characteristics of PTR Records, *Proc. Asia-Pacific Advanced Network* (*APAN*) (2018).

[2] Padmanabhan, R., Dhamdhere, A., Aben, E., Claffy, K. and Spring, N.: Reasons Dynamic Addresses Change, *Proc. ACM Internet Measurement Conference* (*IMC*), pp.183–198 (2016).

[3] Spamhaus: The Spamhaus Project Ltd., The Spamhaus Project, available from ⟨https://www.spamhaus.org/⟩.

[4] Spamhaus: The Spamhaus Project Ltd., Compoist Blocking List, available from ⟨http://www.abuseat.org/⟩.

[5] SORBS: Spam and Open Relay Blocking System (SORBS), available from ⟨http://www.sorbs.net/⟩.

[6] Çetin, O., Gañán, C., Altena, L., Tajalizadehkhoob, S. and van Eeten, M.: Let Me Out! Evaluating the Effectiveness of Quarantining Compromised Users in Walled Gardens, *Fourteenth Symposium on Usable Privacy and Security, SOUPS 2018*, pp.251–263 (2018) (online),

available from ⟨https://www.usenix.org/conference/soups2018/presentation/cetin⟩.

[7] Secretariat of Advanced Cyber Threats response InitiatiVE: ACTIVE An anti-malware support project through the public-private partnership, available from ⟨https://www.ict-isac.jp/active/en/⟩.

[8] Cetin, F.O., Ganán, C., Altena, L., Kasama, T., Inoue, D., Tamiya, K., Tie, Y., Yoshioka, K. and van Eeten, M.: Cleaning Up the Internet of Evil Things: Real-World Evidence on ISP and Consumer Efforts to Remove Mirai, *Network and Distributed Systems Security Symposium 2019* (2019).

[9] NOTICE SUPPORT CENTER: NOTICE Surveying vulnerable IoT devices and alerting users to the problem, available from ⟨https://notice.go.jp/en⟩.

[10] Xie, Y., Yu, F., Achan, K., Gillum, E., Goldszmidt, M. and Wobber, T.: How dynamic are IP addresses?, *Proc. ACM SIGCOMM 2007 Conf. Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp.301–312 (2007).

[11] Richter, P., Smaragdakis, G., Plonka, D. and Berger, A.W.: Beyond Counting: New Perspectives on the Active IPv4 Address Space, *Proc. ACM Internet Measurement Conference* (*IMC*), pp.135–149 (2016).

[12] Jin, Y., Sharafuddin, E. and Zhang, Z.: Identifying dynamic IP address blocks serendipitously through background scanning traffic, *Proc. ACM Conf. Emerging Network Experiment and Technology* (*CoNEXT*), p.4 (2007).

[13] Cai, X. and Heidemann, J.S.: Understanding block-level address usage in the visible internet, *Proc. ACM SIGCOMM 2010 Conf. Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp.99–110 (2010).

[14] Akamai: Attack Spotlight: An Internet of Things Botnet, available from ⟨https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/attack-spotlight-internet-of-things-botnet-threat-advisory.pdf⟩.

[15] Manos, A., Tim, A., Michael, B., et al.: Understanding the Mirai Botnet, *Proc. USENIX Security Symp.*, pp.1093–1110 (2017).

[16] Amazon: Route 53 Reverse DNS.

[17] MaxMind: GeoLite Legacy Downloadable Databases, available from ⟨https://dev.maxmind.com/geoip/legacy/geolite/⟩.

[18] Mozilla Foundation: Public Suffix List, available from ⟨https://publicsuffix.org/⟩.

[19] Harvard University: Jaro-Winkler Distance, available from ⟨https://scholar.harvard.edu/jfeigenbaum/software/jaro-winkler-distance⟩.

[20] Jaro, M.A.: Advances in Record-Linkage Methodology as Applied to Matching the 1985 Census of Tampa, Florida, *Journal of the American Statistical Association*, Vol.84, No.406, pp.414–420 (1989) (online), available from ⟨http://www.jstor.org/stable/2289924⟩.

[21] Jaro, M.A.: Probabilistic linkage of large public health data files, *Statistics in Medicine*, Vol.14, No.5–7, pp.491–498 (1995).

[22] Winkler, W.E.: String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage, *Proc. Section on Survey Research*, pp.354–359 (1990).

[23] Winkler, W.E.: Overview of Record Linkage and Current Research Directions, Statistical research division report series (2006).

[24] Python Software Foundation: pyjarowinkler PyPI, available from ⟨https://pypi.org/project/pyjarowinkler/⟩.

[25] Rapid7: Reverse DNS (RDNS), available from ⟨https://opendata.rapid7.com/sonar.rdns_v2/⟩.

[26] Amazon Web Services, I.: Alexa Top Sites - Up-to-date lists of the top sites on the web, available from ⟨https://aws.amazon.com/alexa-top-sites/⟩.

[27] Takata, Y., Terada, M., Matsuki, T., Kasama, T., Araki, S. and Hatada, M.: Datasets for Anti-Malware Research MWS Datasets 2018, IPSJ Technical Report 38 (2018). (in Japanese).

[28] Kasama, T.: Nicter Dataset 2018, available from ⟨http://www.iwsec.org/mws/2018/20180530/NICTER_Dataset_2018.pdf⟩.

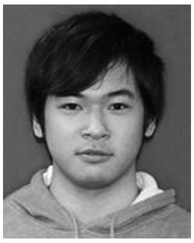
[29] Asami, H.: Study Report of an Anti-spam System with a 99% Block Rate, available from ⟨http://www.gabacho-net.jp/en/anti-spam/anti-spam-system.html⟩.

[30] Lee Howard: Reverse DNS in IPv6 for Internet Service Providers, RFC 8501 (2018).

Editor's Recommendation

The paper proposes a method to detect dynamically assigned IP address ranges and cloud service ranges. The method is particularly useful for analyzing hosts with dynamic IP addresses, such as bot-infected hosts. It is also notable that the method can be used even for the end users who are not capable of observing large network traffic. Careful evaluation with large and broad dataset ensures the effectiveness of the proposal. We believe that the paper provides very useful insight for analyzing cyber threats.

(The program chair of Computer Security Symposium 2018 (CSS2018) Katsunari Yoshioka)



Tomofumi Nakamori received his B.S. degree in Computer Science and Engineering from Waseda University in March, 2017. He is now a master student at the Department of Computer Science and Communications Engineering, Waseda University. His research interest covers Cyber Security. He is now with NTT Communications Corporation, Tokyo, Japan.

Daiki Chiba is currently a researcher at NTT Secure Platform Laboratories, Tokyo, Japan. He received his B.E., M.E., and Ph.D. degrees in computer science from Waseda University in 2011, 2013, and 2017. Since joining Nippon Telegraph and Telephone Corporation (NTT) in 2013, he has been engaged in research on cyber security through data analysis. He won the Research Award from the IEICE Technical Committee on Information and Communication System Security in 2016 and the Best Paper Award from the IEICE Communications Society in 2017. He is a member of IEEE and IEICE.

Mitsuaki Akiyama received his M.E. and Ph.D. degrees in Information Science from Nara Institute of Science and Technology, Japan in 2007 and 2013. Since joining Nippon Telegraph and Telephone Corporation NTT in 2007, he has been engaged in the research and development of network security, especially honeypot and malware analysis. He is now with the Cyber Security Project of NTT Secure Platform Laboratories.

Shigeki Goto is a professor emeritus, Waseda University, Japan. He received his B.S. and M.S. in Mathematics from the University of Tokyo. He also earned his Ph.D. in Information Engineering from the University of Tokyo. He has worked for NTT Laboratories from 1973 to 1996. He was a professor at Waseda University from 1996 to 2019. He is the president of JPNIC. He is a member of ACM and IEEE. He was inducted into the Internet Hall of Fame by Internet Society in 2017.