Journal of Information Processing Vol.22 No.3 480–485 (July 2014)

[DOI: 10.2197/ipsjjip.22.480]

Regular Paper

Domain Registration Date Retrieval System for Improving Spam Mail Discrimination

Nariyoshi Yamai[1,†1,a)] Masayuki Matsuoka[2,†2] Kiyohiko Okayama[1] Keita Kawano[1]
Motonori Nakamura[3] Masato Minda[4]

Received: September 16, 2013, Accepted: February 14, 2014

Abstract: Recently, many spam mails associated with "One-click fraud," "Phishing," and so on have been sent to unspecified large number of e-mail users. According to some previous works, most spam mails contained some URLs whose domains were registered relatively recently, such that the age of the domain used in the URL in the messages would be a good criterion for spam mail discrimination. However, it is difficult to obtain the age or the registration date of a specific domain for each message by WHOIS service since most WHOIS services would block frequent queries. In this paper, we propose a domain registration date retrieval system, which updates zone files of some Top Level Domains (TLDs) every day, keeps track of the registration date for new domains, and works as a DNS server that replys with the registration date of the queried domain. According to the performance evaluation, the prototype system could update the registration date for all the domains of "com" TLD in two hours.

Keywords: e-mail, spam, domain, whois, DNS

1. Introduction

Today, with the increase in Internet users, e-mail is one of the most important communication tools that support social activities. However, e-mail is one of the most problematic applications in terms of security. Especially, spam mails sent to a large number of users for purpose of advertising have become a severe social problem. According to the announcement in August 2013 in the Symantec Intelligence Report, spam mails accounted for 65.2% of all the e-mail in terms of the number of messages [2]. These spam mails are harmful in terms of wasting communication and computer resources, wasting labor to discriminate spam mails, deleting legitimate e-mails as spam mails by mistake, damage to reputations of legitimate organizations due to spoofing sender's address by spammers and so on. In addition, most spam mails contain URLs that cause serious problems such as malware infection, phishing, and so on when accessing them.

Recently various measures are taken to these problems. A typical method among them is to discriminate the spam mails by introducing blacklists of domains frequently used in URLs in past spam mails. This approach seems effective in discriminating spam mails since most of them contain URLs in their messages. However, to pass through these blacklists, spammers often obtain new domains and throw them away in short time. Since this method frequently changes URLs in spam messages, it makes blacklists based on URLs less effective.

On the other hand, to use this method, spammers have to obtain new domains one after another. Thus the ages of these domains tend to short term. In fact, some previous works [3], [4], [5] showed that domains in URLs in spam mails or those of malicious websites were registered recently. Therefore, the age of a domain would be a good criterion for spam mail discrimination.

In previous works, WHOIS services were used to retrieve the information on domains. However, since most WHOIS services have access rate limit on frequent information retrieval, we cannot use them for realtime spam discrimination. Thus in this paper, we propose a system that gathers some zone information files provided by major Top Level Domain (TLD) registries and responds with the registration dates of queried domains.

In the rest of the paper, we describe the background on recent spam mails and anti-spam techniques in Section 2. Then, in Section 3, we explain the design of the proposed system to resolve the problem. We also explain the implementation and evaluation of the system in Section 4. Finally, we provide a summary of this paper and future work in Section 5.

Note that accuracy of spam mail discrimination with the proposed system is beyond the scope of this paper since many previous works discussed effectiveness of using the age of a domain as a filtering criterion for spam mails or malicious websites.

[1] Center for Information Technology and Management, Okayama University, Okayama 700–8530, Japan
[2] Graduate School of Natural Science and Technology, Okayama University, Okayama 700–8530, Japan
[3] Research and Development Center for Academic Networks, National Institute of Informatics, Chiyoda, Tokyo 101–8430, Japan
[4] Japan Registry Services Co., Ltd., Chiyoda, Tokyo 101–0065, Japan
[†1] Presently with Institute of Engineering, Tokyo University of Agriculture and Technology
[†2] Presently with Information Development Co., Ltd.
[a)] nyamai@cc.tuat.ac.jp

This paper is a revised version of Ref. [1].

2. Existing Anti-spam Techniques and Problems

There are so many anti-spam methods proposed so far. They can be categorized into two groups: stand-alone methods and cooperative methods. As for the former methods, tempfailing and throttling techniques are well known. As for the latter methods, DNS-based BlackLists (DNSBLs) based on sender's IP address or URL are widely used. In this section, we focus on some DNSBLs.

2.1 DNSBLs Based on Sender's IP Address

Spammers use various methods to send spam mails so that they conceal the controlling hosts and web servers that are used for the attack to prevent tracing. Thus they can send many spam mails without being blocked. As a conventional method, spammers often used external mail servers without sufficient setting (called open relay servers), through which spammers sent spam mails, along with spoofing the sender's address. Recently, spammers have used botnets which consist of many PCs infected by some viruses, instead of open relay servers.

As a countermeasure of these methods, DNSBLs based on sender's IP address are often used. DNSBLs register the IP addresses of open relay servers and PCs in botnets according to the reports from spam recipients. If these DNSBLs would keep their contents up-to-date, we could block many spam mails by consulting these DNSBLs. However, it is difficult to do so since PCs of botnets vary continually. In addition, some benign servers are often registered to some DNSBLs accidentally because they forward not only legitimate mails but also many spam mails to other servers. In this case, even legitimate mails from those servers are blocked.

2.2 DNSBLs Based on URL

As one of the cooperative methods, DNSBLs based on URL are also widely used. Since most spam mails contain some URLs for advertisement, phishing, and so on, these DNSBLs seem effective against most spam mails. Typical DNSBLs are SURBL (Spam URL Realtime BlackList) [6], URIBL (URI BlackList) [7], ivmURI (Invaluement Anti-Spam Blacklist) [8] and so on.

These DNSBLs register the domains of the URLs in spam mails. When a mail server consulting DNSBLs receives a message, it extracts the domain part of each URL in the message and queries the domain to one or more DNSBLs. If one of the DNSBLs replies to the server that the domain is registered in its blacklist, the mail server then marks the message as spam.

However, DNSBLs have become less effective for the following reason.

Regardless of filtering by DNSBLs, domains used for malicious activities would be inactivated by the domain name registries after a while. To continue malicious activities, spammers have to obtain a lot of new domains (throw-away domains) illegally, by abusing personal information for example, and use them in URLs one after another. Since these throw-away domains are so many but not used so much, it is difficult to keep the contents of the blacklists up-to-date. Even if these throw-away domains are registered promptly, they are used only in the short term. Consequently, the effectiveness of DNSBLs against these throw-away domains would be diminished.

2.3 Spam Discrimination Method Based on Domain Registration Date

According to the method mentioned above, spammers obtain a lot of new domains and throw them away in a short time. In other words, these throw-away domains are registered recently by registries. Thus, we can consider a spam discrimination method based on the registration date of the domain.

There are many previous works investigating the registration date of each domain in a blacklist. For example, the various information of each domain was previously placed in a blacklist called JWSDB [5]. According to the investigation report, they found that 90% of the domains in the blacklist were registered within a year. Therefore, we expect that the domain registration date would be a good criterion for spam discrimination.

We can obtain the registration date of a specific domain by accessing the WHOIS [9] service. This service provides various information on the queried domain such as the registrant, the registrar, the contact address, and so on, as well as the registration date. According to the Registrar Accreditation Agreement issued by ICANN [10], anyone can obtain these data free of charge.

However, it is difficult for a mail server to obtain the registration dates of domains for all received messages in real time due to the following reasons. One reason is the access rate limit of WHOIS service. Most WHOIS servers provide access rate control functions for massive accesses from a client as a countermeasure against the intended use of such data for marketing, address harvesting, and so on. If a mail server would access WHOIS servers whenever it would receive a message containing one or more URLs, the accesses to WHOIS servers would soon be blocked by the access rate limit. Another reason is the lack of the common format of domain data. Since the Registrar Accreditation Agreement regulates the data elements but not the format, registrars submit the domain data in their individual formats. Accordingly it is difficult for a mail server to analyze the registration data obtained from a WHOIS server and extract the registration date correctly.

3. Design of Domain Registration Date Retrieval System

As mentioned in the previous section, the domain registration date would be a good criterion for spam discrimination, but it is difficult for a mail server to retrieve from existing WHOIS servers. In this section, we propose a method for obtaining the domain registration date without consulting WHOIS services and describe our design of the domain registration date retrieval system.

3.1 Outline of the System

As mentioned above, it is difficult to use the WHOIS service for domain registration date retrieval. However, for some top level domains (TLDs) such as "com," "net," "org," and so on, we can obtain the lists of the existing domains, called "zone informa-

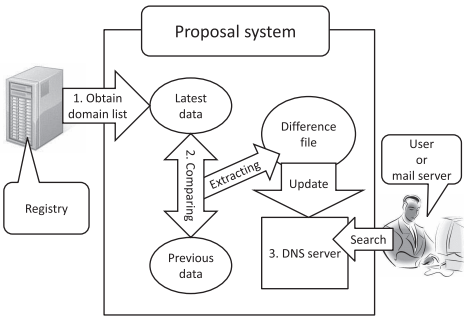

Fig. 1 The layout of the proposal system.

tion" or "zone file" from the corresponding registries. Since these lists are updated daily, we can find new domains by comparing the latest version with the previous version from the day before. Thus we can keep track of the registration date of each domain by treating the day of the first registration as the registration date. This method does not require many accesses to any existing services regardless of the number of target domains. Note that the registration date of a domain in this method is not always the same as that obtained by WHOIS service. For example, if a domain would be registered without name server (NS) resource records, its registration date would be found by WHOIS service, but not by the proposed system. However, we expect that the difference between registration dates is negligible for spam discrimination.

In addition to the function for obtaining the registration dates, the system must have a function to reply with the registration date of the queried domain like existing WHOIS services since the system would be accessed for each mail received. Since many spam filtering programs such as SpamAssassin [11] have a function to access DNS servers such as DNSBLs, we designed the system as a DNS server, which replies with the registration date instead of the IP address of the queried domain.

3.2 Design of the System

The layout of the proposed system is shown in **Fig. 1**. Each component of this figure has the following functions.

(1) Obtaining zone files
This function obtains the latest zone files from the registries for some TLDs everyday. Then, it generates a domain list containing only the domain names for each TLD for ease of processing.

(2) Comparing domain lists
This function compares today's domain list with yesterday's for each TLD. The new domains and deleted domains are stored in the difference file.

(3) DNS server
The DNS server updates its zone file according to the difference file. The new domains are registered in the zone file along with today's date and the deleted domains are excluded from the zone file. Then, it receives queries from clients and replies with the registration dates of queried domains.

4. Implementation of Prototype System and Performance Evaluation

In this section, we describe the implementation of the prototype system and performance evaluation of the system.

Table 1 Spam URL distribution based on Top Level Domain name.

TLD	Usage rate (%)
com	51.33
ru	13.14
info	9.34
net	6.88

```
$ORIGIN EXAMPLE.
$TTL 900
@ IN SOA a.gtld-servers.example. nstld.registry.example. (
                              1309579858 ;serial
                              1800 ;refresh every 30 min
                              900 ;retry every 15 min
                              604800 ;expire after a week
                              86400 ;minimum of 15 min
                              )

EXAMPLE0 NS NS1.EXAMPLE.COM.
EXAMPLE1 NS NS17.EXAMPLE.NET.
EXAMPLE1 NS NS18.EXAMPLE.ORG.
EXAMPLE2 NS NS14.EXAMPLE.EXAMPLE.
(The rest is omitted.)
```

Fig. 2 A sample zone file.

4.1 Implementation of Prototype System

To keep the domain registration dates up-to-date, we implemented the following processes.

4.1.1 Obtaining Zone Files

Some TLD registries provide Zone File Access Programs. We applied to Verisign [12] for these programs for "com" and "net," and to PIR [13] for "org." These zone files are updated as frequently as everyday, which means the registered date has the accuracy of one day. We obtained them since these TLDs were the top three TLDs used in spam mails when we started this research. According to the Symantec Intelligence Report of February 2013 [14], the usage rates of the top four TLDs used for spam mails are shown in **Table 1**. Since these rates fluctuate frequently, there is no "org" in this table, but it is also used for spam mail at a high rate. We did not obtain zone files for "ru" nor "br" since we could not find Zone File Access Programs for them when we started this research. We also did not obtain other major TLDs such as "info" and "biz" either. However, the system can deal with these domains if we can obtain their zone files.

We used FTP to download zone files for these domains. Since these files are too big to be downloaded without any errors, we used the "wget" command with "-c" option, which means "continue getting a partially-downloaded file." The format of the zone files are all the same and shown in **Fig. 2**.

Then we extracted domain names using the "awk"[*1] and "uniq"[*2] commands on a UNIX system. For example in Fig. 2, the system extracted the domain names EXAMPLE0, EXAMPLE1, and EXAMPLE2 through this process.

4.1.2 Comparing Domain Lists

Since the size of each domain list, especially of the "com" TLD, is huge, we introduce a database to store the domain names and their registration dates. We used MySQL for the database

[*1] A command for pattern scanning and processing language.
[*2] A command to omit repeated lines.

management system since the DNS server supports it, as mentioned below.

In an early implementation, we updated the database by inquiring each domain in today's domain list of the database. However, since the number of domains in the "com" TLD zone file is about 100 million, we could not finish the update within a day. Thus, we had to find an alternative method to update the database within a day. According to the analysis of zone files, we found that only the records of a small percent of domains were changed each day and the order of other records was not changed so much. Consequently, we used "diff" command of UNIX to efficiently print the difference between two similar files. The total process of updating the database is as follows.

(1) Today's domain list is compared with that of the previous day by the "diff" command. Accordingly, the difference in records, including new domains and deleted domains, are created.
(2) The database is updated according to the difference in records. That is, new domains are registered into the database along with today's date, and the deleted domains are removed from the database.

Note that, if a domain is marked as both "new domain" and "deleted domain", we can ignore this domain since it means the record of this domain has been moved.

4.1.3 Updating Zone Information

As mentioned above, the system works as a DNS server by replying with the registration date of the queried domain. We used the BIND [15] implementation as the server program.

We adopted BIND DLZ (Dynamic Loadable Zones) [16], which enables the use of an external database as a back end, to manage the zone file because of the following reason. In a conventional operation of BIND, we first update the zone file and then have the DNS server reload the zone file. However, reloading a huge zone file would take quite a long time on the order of several hours, during which the DNS server cannot respond to any queries. If we would use another conventional operation with dynamic DNS [17], updating a record would take a relatively short time. However, in this operation, all the records of the zone file is loaded in memory, even though most are not referred to. This means that the conventional operation consumes a lot of memory. DLZ can solve these problems. With DLZ, updating the database is equivalent to updating the zone file. Therefore the DNS server with DLZ works well even during the update of the database. In addition, the DNS server with DLZ consumes only a small amount of memory.

The total flow of the zone information update is shown in **Fig. 3**.

4.1.4 Retrieval of Domain Registration Date

We can retrieve the domain registration date from the system by querying the TXT record of the target domain. To distinguish queries for the domain registration date from conventional queries, the system provides the service on the "zone" domain [*3]. For example, to retrieve the registration date of the domain "example.net", the client queries the TXT record of the domain "example.net.zone" to the system. **Figure 4** shows an example output of the "dig" command.

[*3] Since "zone" is currently registered as one of the TLDs, we will use another domain such as "local" instead.

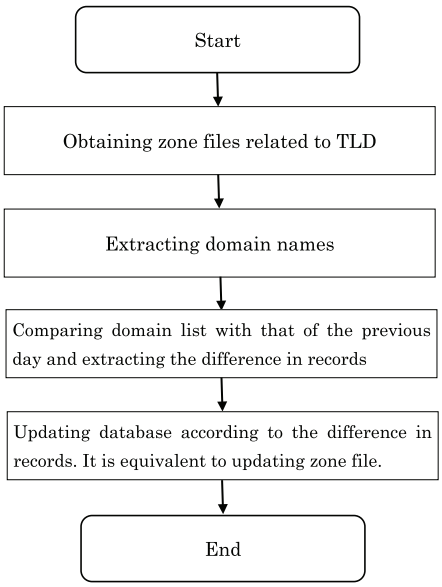

Fig. 3 The processing flow of the zone information update.

4.2 Performance Evaluation

4.2.1 Evaluation Environment and Testing Zone Data

The prototype system works on the environment shown in **Table 2**. Note that the system works on a virtual machine.

We used the zone files of "com," "net," and "org" TLDs for the performance evaluation. The number of domains and their NS records for these TLDs are shown in **Table 3**. This table also includes the numbers of added and deleted domains per day. Since these numbers fluctuate daily these numbers have been rounded.

4.2.2 Data Processing Time

Table 4 shows the processing time of each step and the total processing time. According to this table, the download time, denoted as "wget&gunzip," is the dominant part of the total processing time. Although it is difficult to improve the downloading time due to network and server condistions, the total processing time for updating the database is short enough for daily update.

In the early version of the system where we did not use the "diff" command nor the BIND DLZ function, it took as long as several days. However, by virtue of this command and function, we have achieved practical performance.

4.2.3 DNS Performance

We also evaluated the performance of the system as a DNS server by "queryperf," which is distributed as a contributed tool for BIND 9. The result of the performance evaluation is shown in **Table 5**.

According to the result of "queryperf," the performance of the prototype system was 37.4 queries per second. This value is relatively small because of the overhead of the backend MySQL and BIND DLZ. However, it does not seem a problem in practice since the applicable range of the proposed system is within the organization under the zone file access agreement. Even if the performance of the system is too low, it is easy to improve the performance by introducing the server replication technique, as

```
$ dig @150.46.XX.YY example.net.zone txt

; <<>> DiG 9.6.-ESV-R3 <<>> @150.46.XX.YY example.net.zone txt
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 29759
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;example.net.zone.                IN        TXT

;; ANSWER SECTION:
example.net.zone.       86400     IN        TXT      "20110623"

;; Query time: 261 msec
;; SERVER: 150.46.XX.YY#53(150.46.XX.YY)
;; WHEN: Wed Jan 23 18:55:47 2013
;; MSG SIZE  rcvd: 53
```

Fig. 4 An example of domain registration date search using the proposed system.

Table 2 The environment of the prototype system.

Host computer

OS	VMware ESXi5
CPU	Intel(R) Xeon(R) E5620
RAM	18 GB
HDD	1 TB

Virtual machine

OS	FreeBSD/amd64 8.2-RELEASE
Assigned CPU	2.40 GHz, 8 core
Assigned RAM	18 GB
Assigned HDD	512 GB

Table 3 Statistics of the domains of each TLD.

TLD	# doms. (NSes)		# added/day	# deleted/day
com	112 M	(260 M)	110 k	72 k
net	15 M	(36 M)	11 k	11 k
org	10 M	(24 M)	7 k	6 k

Table 4 Data Processing Time.

TLD	wget&gunzip [*4]	awk&uniq	diff	SQL	Total
com	133 min	4 min	3 min	40 min	180 min
net	18 min	1 min	1 min	5 min	25 min
org	5 min	1 min	1 min	2 min	9 min

Table 5 Result of DNS Performance Evaluation.

Run time	600 seconds
Queries sent	22,432 queries
Ave. RTT	0.54 seconds
Max. RTT	0.87 seconds
Min. RTT	0.018 seconds
Query Per Second	37.4 qps

usual in traditional DNS operation.

5. Conclusion

In this paper, we proposed a domain registration date retrieval system, which keeps track of the domains and their registration dates for some TLDs without accessing WHOIS services, and responds with the registration date for the queried domain. We also implemented a prototype system and evaluated the performance of the system.

With this system, we can use the age of the domain as a criterion of spam discrimination even if spammers frequently change the domains of the URLs in spam mails.

As future work, we would like to use the system for spam discrimination in the practical environment. In addition, we would like to examine the correlation between the age of the domain and whether the domain is used in spam mails.

Moreover, after we confirm the effectiveness of the system, we would like to ask many TLD registries to provide domain registration date retrieval systems similar to our prototype system as a public service.

Acknowledgments This research was partially supported by Japan Society for the Promotion of Science (JSPS), Grant-in-Aid for Scientific Research (C) No. 23500122 in 2011–2013.

[*4] depending on network and server conditions.

References

[1] Matsuoka, M., Yamai, N., Okayama, K., Kawano, K., Nakamura, M. and Minda, M.: Domain Registration Date Retrieval System of URLs in E-mail Messages for Improving Spam Discrimination, *Proc. 2013 IEEE 37th International Conference on Computer Software and Applications* (*COMPSAC 2013*) *Workshops*, pp.587–592 (July 22–26, 2013).

[2] Symantec Corporation: Symantec Intelligence Report: August 2013 (online), available from ⟨http://www.symantec.com/content/en/us/enterprise/other_resources/b-intelligence_report_08-2013.en-us.pdf⟩ (accessed 2013-09-16).

[3] Zhang, Y., Hong, J. and Cranor, L.: CANTINA: A Content-Based Approach to Detecting Phishing Web Sites, *Proc. 16th International World Wide Web Conference* (*WWW'07*), pp.639–648 (2007).

[4] McGrath, D.K. and Gupta, M.: Behind Phishing: An Examination of Phisher Modi Operandi, *Proc. 1st USENIX Workshop on Large-scale Exploits and Emergent Threats* (*LEET'08*) (2008) (online), available from ⟨https://www.usenix.org/legacy/events/leet08/tech/full_papers/mcgrath/mcgrath.pdf⟩ (accessed 2013-09-16).

[5] Felegyhazi, M., Kreibich, C. and Paxson, V.: On the potential of proactive domain blacklisting, *Proc. 3rd USENIX Workshop on Large-scale Exploits and Emergent Threats* (*LEET'10*) (2010) (online), available from ⟨https://www.usenix.org/legacy/event/leet10/tech/full_papers/Felegyhazi.pdf⟩ (accessed 2013-09-16).

[6] SURBL: SURBL (online), available from ⟨http://www.surbl.org/⟩ (accessed 2013-09-16).

[7] uribl: URIBL.COM (online), available from ⟨http://www.uribl.com/⟩ (accessed 2013-09-16).

[8] PowerView Systems: ivmURI (online), available from ⟨http://dnsbl.invaluement.com/ivmuri/⟩ (accessed 2013-09-16).

[9] Daigle, L.: WHOIS Protocol Specification, RFC 3912, IETF (2004).

[10] ICANN: Registrar Accreditation Agreement (17 May 2001) (online), available from ⟨http://www.icann.org/en/registrars/ra-agreement-17may01.htm⟩ (accessed 2013-09-16).

[11] SpamAssassin: Welcome to SpamAssassin (online), available from ⟨http://spamassassin.apache.org/⟩ (accessed 2013-12-14).
[12] Verisign: Verisign (online), available from ⟨http://www.verisigninc.com/⟩ (accessed 2013-09-16).
[13] Public Internet Registry: PIR (online), available from ⟨http://www.pir.org⟩ (accessed 2013-09-16).
[14] Symantec Corporation: Symantec Intelligence Report: February 2013 (online), available from ⟨http://www.symantec.com/content/en/us/enterprise/other_resources/b-intelligence_report_02-2013.en-us.pdf⟩ (accessed 2013-09-16).
[15] Internet Systems Consortium: BIND 9 Administrator Reference Manual (online), available from ⟨http://ftp.isc.org/isc/bind9/cur/9.8/doc/arm/Bv9ARM.pdf⟩ (accessed 2013-09-16).
[16] SOURCEFORGE.NET: Bind DLZ (online), available from ⟨http://bind-dlz.sourceforge.net/⟩ (accessed 2013-09-16).
[17] Vixie, P. (Ed.), Thomson, S., Rekhter, Y. and Bound, J.: Dynamic Updates in the Domain Name System (DNS UPDATE), RFC 2136, IETF (1997).

Nariyoshi Yamai received his B.E. and M.E. degrees in electronic engineering and his Ph.D. degree in information and computer science from Osaka University, Osaka, Japan, in 1984, 1986 and 1993, respectively. In April 1988, he joined the Department of Information Engineering, Nara National College of Technology, as a Research Associate. From April 1990 to March 1994, he was an Assistant Professor in the same department. In April 1994, he joined the Education Center for Information Processing, Osaka University, as a Research Associate. In April 1995, he joined the Computation Center, Osaka University, as an Assistant Professor. From November 1997 to March 2006, he joined the Computer Center, Okayama University, as an Associate Professor. From April 2006 to March 2014, he was a Professor in Information Technology Center (at present, Center for Information Technology and Management), Okayama University. Since April 2014, he has been a Professor in Institute of Engineering, Tokyo University of Agriculture and Technology. His research interests include distributed systems, network architecture and Internet. He is a member of IEICE and IEEE.

Masayuki Matsuoka received his B.E. and M.E degrees in engineering from Okayama University, Japan, in 2012 and 2014, respectively. He is currently with Information Development Co., Ltd. His research interests include E-mail system, anti-spam technology, and Internet architecture.

Kiyohiko Okayama received his B.S., M.S. and Ph.D. degrees in information and computer sciences from Osaka University, Japan, in 1990, 1992 and 2001, respectively. After he has worked in the Department of Information System at Osaka University and in the Graduate School of Information Science at Nara Institute of Science and Technology as a research associate, he joined the Department of Communication Network Engineering at Okayama University in 2000. From 2005 to 2011, he joined the Information Technology Center at Okayama University. Since 2011, he has been an Associate Professor in Center for Information Technology and Management at Okayama University. His research interests include network design and network security. He is a member of IEICE.

Keita Kawano received his B.E., M.E., and Ph.D. degrees from Osaka University, Osaka, Japan, in 2000, 2002, and 2004, respectively. From October 2004 to March 2010, he was an Assistant Professor of the Information Technology Center, Okayama University, Okayama, Japan. From April 2010 to March 2011, he was an Assistant Professor of the Center for Information Technology and Management, Okayama University. Since April 2011, he has been an Associate Professor of the same center. His research interests include mobile communication networks and distributed systems. He is a member of IEEE and IEICE.

Motonori Nakamura graduated from Kyoto University, Japan, where he received his B.E., M.E. and Ph.D. degrees in engineering in 1989, 1991 and 1996, respectively. From 1994, he was an Assistant Professor at Ritsumeikan University. From 1995, he was an Associate Professor at Kyoto University. Currently he is a professor at National Institute of Informatics, Japan (NII). His research interests are message transport network systems, network communications, next generation Internet and Identity & Access Management. He is a member of IEEE, ISOC, IEICE and JSSST.

Masato Minda graduated from University of Electro-Communications. After he worked in a software house, ISP, and a securities firm, he is engaged in employment of DNS or a system by Japan Registry Services Co., Ltd. (JPRS) from 2003.