近畿大学工学部研究報告　No.51，2017年，pp.47-51
Research Reports of the Faculty of Engineering,
Kindai University No.51 2017, pp.47-51

A Cache Management Strategy for Shortening DNSSEC Name Resolution Time

Shuta FUKUDA [†1] and Takayuki FUJINO [†2]

Abstract

To protect the DNS data from cache poisoning attack, the DNSSEC has been deployed on a global basis. Although the DNSSEC provides origin authentication and integrity protection to the DNS data, it requires longer name resolution time due to digital signature processing. In this article we propose the cost metric based cache management strategy for the DNS caching server. We explain the cost metric and show that our proposed strategy effectively reduces the name resolution time in the DNSSEC enabled environment.

Keywords: DNSSEC, name resolution time, cache management

1. INTRODUCTION

The Domain Name System[1,2] provides the name resolution service, which translates from a host name to corresponding IP address. Since most of the application software depend on the DNS, it can be said that the DNS is a critical part of the Internet. However, various threats including cache poisoning attack emerged and eroded the DNS trustworthiness. The summary of these threats is given in[3]. In addition, so-called Kaminsky attack[4], which is a derivative of the cache poisoning attack, revealed that the spoofed data could be effectively inserted into the DNS cache.

In order to cope with these threats the DNS community has developed cryptographically enhanced DNS protocol called the DNS Security Extensions (DNSSEC)[5-7]. Although the DNSSEC provides origin data authen-tication and data integrity to the DNS, it imposes the additional operational problems.

This paper focuses on the name resolution time problem. Since the DNSSEC is based on the public key crypto-graphy, the DNS clients or the DNS caching server needs to validate the signature corresponding to individual DNS resource record (RR). This processing clearly makes the name resolution time longer. In addition, inclusion of associated signatures increases the DNS message size. In some case this results in TCP fall back and it also negatively affects the name resolution time[8]. To mitigate these problems we propose cost metric based cache management strategy for the DNS caching servers. Currently the DNS already has the caching scheme, but it focuses on reducing redundant DNS interactions. If the cache quota is full the DNS caching server needs to select the cache data to be replaced. During this process the name resolution time from the client perspective is not considered. We introduce the cost metric which takes the name resolution time into account. The cache management strategy based on this cost metric favors the cache data which requires longer validation time or longer retrieval time. As a result, from the client perspective average name resolution time become shorter.

2. DNSSEC AND NAME RESOLUTION TIME PROBLEM

When a DNS zone enables DNSSEC, the zone typically uses two types of signing keys, Zone Signing Key (ZSK) and Key Signing Key (KSK). While the ZSK is used for signing individual DNS RR, KSK is used for signing the ZSK. Both signing keys are

[†1] Graduate School of Systems Engineering, Kindai University
[†2] Department of Electronic Engineering and Computer Science, Kindai University

published as DNSKEY RRs in the zone. The signatures associated with RRs are stored and published in RRSIG RRs. The zone delegation is authenticated by a DS record which stores the cryptographic digest of child zone's KSK.

When DNSSEC aware caching server tries to resolve certain name, it has to establish chain-of-trust from the DNS root zone to the target zone by using these DNSSEC RRs. In each zone the caching server has to verify and validate the ZSK, the KSK, and the signatures associated with various RRs. A series of the crypto-graphic processing increases the name resolution time.

Since the additional processing time depends on the key length, we explored the impact to the name resolution time in advance. We set up three level DNS hierarchy (DNS root, .local, sub.local domains) as illustrated in Fig.1. Each zone is configured as DNSSEC enabled. We measured the name resolution time in the DNS client while changing the key length in each zone. Note that the processing time imposed by DNSSEC may differ among individual implementations. Therefore we used several well-known implementations. Specifically BIND[9] and Unbound[10] are used as the DNS caching servers, and BIND, NSD[11], yadifa[12] and KnotDNS[13] are used as the DNS authoritative servers. The results are shown in Fig.2. X-axis represents key length of ZSK and KSK of each zone in bit. Y-axis represents name resolution time in milliseconds (ms). The prefix "C:" and "A:" indicate "Caching server" and "Authoritative server" respectively. The "OFF" is the case where the DNSSEC is disabled (which is equivalent to traditional DNS). Although there are slight differences among the implementations, name resolution time is proportional to the key length. While the average name resolution time is 1.5 [ms] in the DNSSEC disabled case, it increases to 8.1 [ms] in the case where both ZSK and KSK adopt 4096 bit key length.

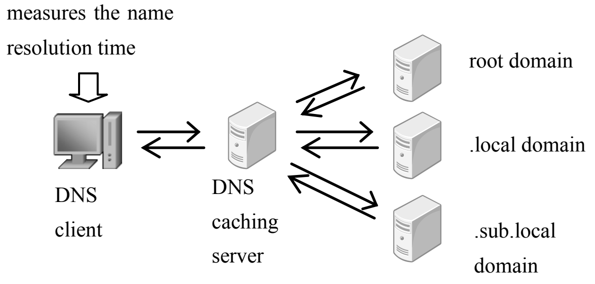

Fig. 1.　Measurement environment.

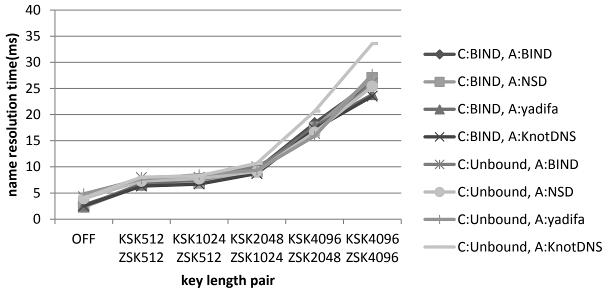

Fig. 2.　Signing key length and the name resolution time

In addition to the problem mentioned above, the DNSSEC increases the DNS message size because the signing key information and signatures must be included in the message to validate the DNSSEC RRs. Broek et al. reported that additional problem caused by message enlargement[8]. Since DNSSEC message size is often greater than path MTU, the message is fragmented to multiple IP fragments. Unfortunately some firewalls block the fragments to prevent some types of cyber-attacks. In such a case name resolution process falls back to TCP instead of normal UDP interactions. This fall back behavior clearly lengthens the name resolution time.

Due to these reasons the name resolution time will become longer when the DNSSEC is deployed on the global basis. However, the study on this problem is not conducted enough. Guillard explored the DNSSEC impact on the authoritative server[14]. The paper compared the throughput between the traditional DNS server and the DNSSEC server, then it clarified the DNSSEC degrades the server performance. Soejima et al. investigated the caching server performance in the same way[15]. However there is no work focuses on the name resolution time from the client perspective. Because DNS itself has native caching function, it is obvious that the cache can effectively mitigate the name resolution time problem. But most of the work focused on the hit ratio only[16,17]. They did not take additional DNSSEC processing time into account. In next section, we propose the cache management strategy which focuses on reducing the name resolution time in DNSSEC enabled configurations.

3. PROPOSED CACHE MANAGEMENT STRATEGY

In this section we introduce cost metric based on the name resolution time. The goal is to prefer the cache which requires longer name resolution time and is more likely to be referred frequently. We define the cost as:

$$\begin{aligned} \text{cost} = \{&\text{retrieval time} + \text{validation time}\} \\ &\times \text{the number of referrences} \end{aligned} \quad (1)$$

The “retrieval time” is the sum of the communication time between the caching server and individual authoritative servers which is required to get the desired RRs. If TCP fall back occurs, the value of this item will be larger. The "validation time" is the sum of the validation time of all DNSSEC RRs. Therefore this item is zero when the resolving name does not use the DNSSEC. As showed in Fig.2, the value of this item is proportional to the key length. The number of reference is equal to the number of cache hit of the data.

When the DNS caching server tries to replace the cache data using this cost metric, it will select the data which has the smallest cost value. For example, consider the caching server which has the cache data shown in Table 1. The caching server selects “example1.jp A” to be replaced if it follows FIFO strategy, or it selects “example3.jp A” based on LFU strategy. In contrast to them, the caching server which adopts our cost metric based strategy will choose “example2.jp A” because the cost metric of the data has the smallest value.

Table 1. The cache data and the cost metric.

Cache data	Retrieval time ms]	Validation time [ms]	# of ref.	The cost metric
example1.jp. A	300	5	10	3050
example2.jp. A	100	7	20	2140
example3.jp. A	500	3	5	2515

4. PERFORMANCE EVALUATION

4.1 Experimental Settings

To evaluate our caching strategy, we have conducted simulation experiment. At first, we extracted 10,356 unique domain names from anonymized query log of the caching server located in the faculty of engineering, Kindai University. In terms of individual domain name we measured the communication time between our caching server and corresponding authoritative server. During the simulation experiment we used this data as the communication time between the caching server and the DNS root server or each authoritative server. We assumed that the 30% of the 10,356 unique domain names are DNSSEC enabled. We also assumed that the three key length pair, KSK 4096 bit/ZSK 4096 bit, KSK 4096 bit/2048 bit and KSK 2048 bit/ZSK 1024 bit are distributed uniformly among DNSSEC enabled domains. We used the validation time of the DNSSEC RRs as shown in Table 2. There are the average validation time measured in the preliminary experiment depicted in Fig.1 and Fig.2. We assumed that these validation times are imposed whenever the caching server gets the DNSSEC RRs from the authoritative servers.

Table 2. The validation time of key pairs.

KSK length [bit]	ZSK length [bit]	Validation time [ms]
4096	4096	7
4096	2048	5
2048	1024	3

The simulator traces the query pattern of the query log. It picks up the host name and sends a query to the caching server. The caching server resolves the host name using preprocessed communication time data and validation time data. The caching server stores the name resolution result. If the quota of the cache is full, the caching server replaces the cache data based on cache management strategy.

We used FIFO, LFU and our proposed strategy for the cache management. We also varied the quota of the cache data from 3000 RRs to 10000 RRs. We measured the total name resolution time of 100000 queries.

4.2 Simulation Results

Fig.3 and Fig.4 show the comparison of the total name resolution time. The cache quota is different, Fig.3 is 3000 RRs and Fig.4 is 10000 RRs. Although we investigated other cache quotas, the results are omitted in the interest of space. In both figure X-axis represents the number of queries and Y-axis is corresponding cumu-lative name resolution time in seconds. Both figures clearly indicate that our proposed strategy using the cost metric can effectively reduce the total name resolution time. Specifically our proposed strategy can reduce the name resolution time from 41% (cache 3000 RRs) to 51% (cache 10000 RRs) in comparison with FIFO strategy. Our proposed strategy achieved the shortest name resolution time in all results including the case where the cache quota is 5000, 8000. Fig.5 depicts the relationship between the cache quota and total name resolution time. Every cache management strategy can utilize increased cache quota, but our proposed strategy utilizes the cache data most effectively.

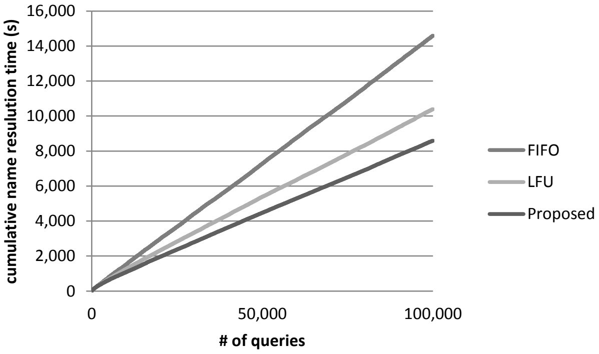

Fig. 3.　Comparison of total name resolution time (cache 3000 RRs)

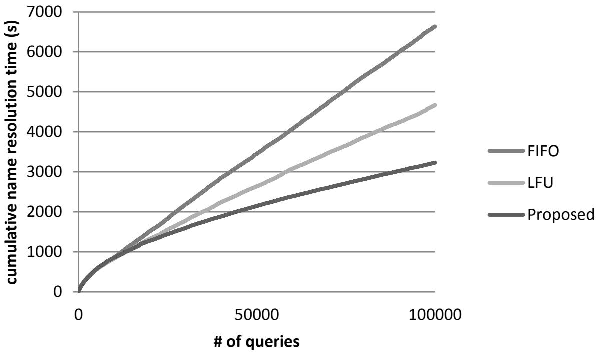

Fig. 4.　Comparison of total name resolution time (cache 10000 RRs)

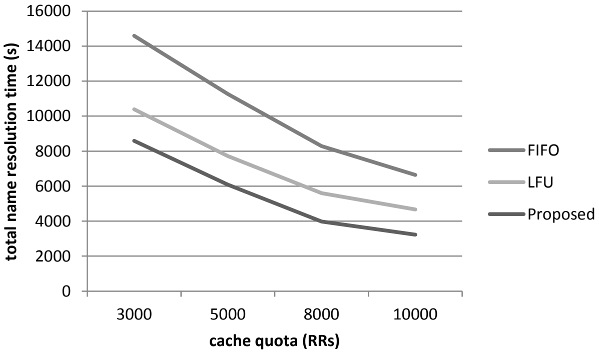

Fig. 5.　Cache quota and the name resolution time transition

5.　OVERHEAD CONSIDERATION

We demonstrated that our proposed strategy can effectively reduce the name resolution time. On the other hand, our proposed strategy requires additional processing and data structures. These factors might give bad influence to the name resolution time. Since the extent of the impact depends on the quality of the programming code, it is quite difficult to measure the impact strictly. Therefore we conducted an approximate estimation.

Consider the situation where the caching server needs to replace the cache data. If the caching server adopts the FIFO strategy, it simply discards the first cache data. In contrast, if the caching server follows our proposed strategy, it has to explore the cache quota to pick up the data which has the smallest cost metric. Therefore the additional processing time imposed by our proposed strategy can approximate the time required to scan the whole cache data.

We used the same configuration with Fig.1 but we did not use .sub.local authoritative server. We used BIND as the caching server implementation. At first the DNS client queries certain amount of the host names (we used k1.local, k2.local… k200000.local). This enables the caching server to store the cache data. Then the client sends the query for the host name which is guaranteed not in the cache. When the caching server receives the query, it checks whole cache data, after that it tries to resolve the host name. By comparing the case where the caching server does not have any cache data and the case the caching server has the cache data, we can measure the scan time of whole cache data. The results are shown in Table 3. It is obvious that measured scan times are quite small. The average name resolution time measured in the simulation mentioned in Section 4 is around 32 [ms] (our proposed strategy with 10000 cache quota). Considering these factors it can be said that the overhead imposed by our proposed strategy can be ignorable.case where the cache quota is 5000, 8000. Fig.5 depicts the relation

Table 3.　The scan time of whole cache data.

# of cache data	Name resolution time [ms]	Difference (scan time) [ms]
0	1.030	0
500000	1.037	0.007
1000000	1.051	0.021
1500000	1.054	0.024
2000000	1.059	0.029

6.　CONCLUSION

The DNSSEC makes name resolution time longer. To cope with the problem we introduce the cost metric which considers the name resolution time. The cache management strategy using this cost favors the cache data which requires longer validation time. We conducted the simulation experiment to estimate our proposed strategy. The results show that our proposed strategy can effectively reduce the name resolution time. In our future work we plan to implement our proposed strategy and evaluate the real world environment.

REFERENCES

1) P.Mockapetris, “DOMAIN NAMES - CONCEPTS AND FACILITIES”, RFC1034 (Nov. 1987).
2) P.Mockapetris, “DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION”, RFC1035 (Nov. 1987).
3) D. Atkins and R.Austein, “Threat Analysis of the Domain Name System (DNS)”, RFC3833 (Aug. 2004).
4) S. Friedl, "An illustrated guide to the Kaminsky DNS vulnerability.", Unixwiz.net Tech Tips (Aug.2008) (http://unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html).
5) R.Arends, R.Austein, M.Larson, D.Massey and S. Rose, "DNS Security Introduction and Requirements", RFC4033(Mar.2005).
6) R.Arends, R.Austein, M.Larson, D.Massey and S. Rose, "Resource Records for the DNS Security Extensions", RFC4034(Mar.2005).
7) R.Arends, R.Austein, M.Larson, D.Massey and S.Rose, "Protocol Modifications for the DNS Security Extensions", RFC4035(Mar.2005).
8) Gijs van den Broek, Roland van Rijswijk-Deij, Anna Sperotto, and Aiko Pras, "DNSSEC meets real world: dealing with unreachability caused by fragmentation", Communications Magazine, IEEE, Vol.52, Issue.4, pp.154-160(Apr.2014).
9) BIND, Internet Systems Consortium , (https://www.isc.org/downloads/bind/).
10) Unbound, NLnet Labs , (http://unbound.net/).
11) NSD, NLnet Labs, (http://www.nlnetlabs.nl/projects/nsd/).
12) yadifa, EURid , (http://www.yadifa.eu/).
13) Knot DNS, CZ.NIC, EURid , (https://www.knot-dns.cz/).
14) Alexis Guillard, "DNSSEC Operational Impact and Performance", Proceedings of the International Multi-Conference on Computing in the Global Information Technology (ICCGI’06)(Aug.2006).
15) Y.Soejima, T.Wakasugi, Y.Shimamura, M.Hirano and E.Oka, "Performance Analysis of DNS Caching Server using DNSSEC", IEICE Technical Report, IN2008-128, pp.37-42(Fec.2009) (in Japanese).
16) J.Jung, E.Sit, H.Balakrishnan, and R.Morris, "DNS Performance and the Effectiveness of Caching", Proceeding of the ACM SIGCOMM Internet Measurement Workshop, 2001.
17) C.E.Wills and H.Shang, "The Contribution of DNS Lookup costs to Web Object Retrieval", Technical Report WPI-CS-TR-00-12, Worcester Polytechnic Institute (WPI), 2000.