Alleviating the impact of DNS DDoS attacks

LI Wei-min
School of Information and Communication Engineering Technology
Beijing University of Posts and Telecommunications
Beijing, China
lwm_bupt@163.com

CHEN Lu-ying
School of Information and Communication Engineering Technology
Beijing University of Posts and Telecommunications
Beijing, China
cly@kuanguang.com.cn

LEI Zhen-ming
School of Information and Communication Engineering Technology
Beijing University of Posts and Telecommunications
Beijing, China
lzm@kuanguang.com.cn

***Abstract*—The Domain Name System (DNS) is a critical fundamental service of the Internet that provides mapping between domain names and IP addresses. In the past few years, distributed denial of service (DDoS) attacks aimed at core DNS servers have caused huge losses. In this paper, we present a simple, practical scheme that can significantly reduce the extent of the DNS DDoS attacks. Firstly, we support that DNS servers should not clean-up TTL-expired domain-name records in the cache when they detected that relevant DNS servers are unavailable. Secondly, according to the data of 7-day DNS trace collected from three different DNS servers on the Internet, it shows that the DNS can still work well during DDoS attacks with a simple modification of the caching behavior.**

Keywords-DNS; Denial of Service; caching behavior; keep-alive;

I. INTRODUCTION

The Domain Name System (DNS) [1-3] provides domain-name service for the Internet. Many Internet applications are designed on the base of the DNS, such as SMTP, SIP, POP3, IMAP, SSH. The availability of DNS is closely related to the availability of other Internet applications. Robustness of the DNS is critical for the Internet.

In recent few years, a number of Distributed Denial of Service attacks have taken aim at the DNS[4-7]. As these attacks above have resulted in varying degree of damage to the domain-name resolution in the targeted zones, the DNS DDoS attacks have posed great threat to the Internet security. It is obvious that we need to improve the availability of DNS.

A lot of research has been done in order to enhance the DNS. Due to the hierarchical structure of DNS, the number of servers in the top-level domain is much smaller than the number of servers in the bottom-level domain. So, some suggestion increate the redundancy of the DNS root-servers and the top-level domain servers with the method of IP Anycast[8].This solution requires large number of name-servers, which may not be afforded by some small domains and would ask for additional cost ,which is not feasible .Another approach[9] focus on encrypted communications among DNS servers and DNS clients, which is not quite easy to deploy for the communication mechanism in the DNS should be changed to this solution. But it has good performance in preventing hackers or attackers from modifying or injecting DNS messages. And some efforts have been made to completely redesign the existing DNS[10].

Our approach is both easily implemented and feasible. It can be deployed either on one DNS server or in large DNS zones, without any protocol changes while achieving strong resistance against the DNS DDoS attacks. The rest of the paper is structured as follows: In section 2, we explain basic DNS concept and present our scheme. Section 3 evaluates of our solution with a set of real DNS traces. And in section 4 we discuss more about some details. Section 5 concludes this paper.

II. SOLUTION PROPOSE

A. How DNS Server Works Today

The DNS servers try to respond to a DNS query by searching the DNS data in the cache. A cache is a set of domain-name records, each of which is associated with a time-to-live (TTL) value. A domain-name record is evicted from the cache if its TTL expires. A DNS server responds to a DNS query with the following steps:

Search the cache for a matching record. If it's found, the DNS server responds to the query with it.

If a matching record is not found in the cache, the DNS server searches for the closest zone that encloses the query and caches the information. Then, starting from the closest enclosing zone, the DNS servers traverse down the DNS zone hierarchy by querying subsequent sub-zones. This process is not stopped until one of the followings happens, and a query is response:

(a) The zone responsible for the domain-name is reached and answers the query.

(b) An error is responded from a server which implies that the traversal cannot go on. The DNS defines border servers of a traversal to control the scope of a iterative query.

(c) If a server fails to get response from any relevant zones in the process of traversal, it return a "Server Failure" response that implies the traversal fails due to that any relevant zones fail to answer.

This work is supported by key project in the National Science & Technology Pillar Program (2008BAH37B04).

978-0-7695-4011-5/10 $26.00 © 2010 IEEE
DOI 10.1109/NSWCTC.2010.63

B. DNS DDoS Attacks

A Distributed Denial of Service (DDoS) attacks against the DNS disrupt the domain-name resolution of a DNS server, which includes any domain-names belonging to the zone and sub-zones. And also, any DNS servers that forward the server under DDoS attacks would fail to answer some of the DNS queries, which are not cached, and cannot resolve any domain-name if all the records are evicted due to TTL-expiration. In this way, the unavailability of a DNS server would lead to the unavailability of other DNS servers related to it.

C. Propose Scheme

We support that changing the caching behavior of a DNS server would significantly improve the availability of a DNS server, especially when the relevant servers are unavailable due to DDoS attacks. And the modification is feasible in the existing DNS. We argue that a DNS server should not evict the TTL-expired records when it detects the relevant DNS servers are unavailable. That means if a query is sent by the DNS server and could not receive a response from a relevant DNS server within a timeout period, the DNS server would not evict TTL-expired matching record which can answer the query. When the relevant DNS server recovers, it would be able to reply to the hello packet that we send and we can detect the recovery of the relevant DNS server, then we evicted all the TTL-expired domain-names in the cache, this would keep the records up to date. With this simple modification, the DNS servers related with a DNS server harmed by the DDoS attacks are capable to provide part of the domain-name resolving service normally, which means they can resolve part of the queries given to.

D. Details about Keep-alive

We propose a method for DNS servers to keep alive with each other based on the DNS framework. Define a hello packet that queries a domain-name which does not exist, such as the domain-name *a.* DNS server *A*, that needs to keep alive with a relevant server *B*, sends a hello packet defined before to the relevant server with the time interval. If server *B* is available, it would reply a NXDOMAIN response to server *A* which indicate that the domain-name queried does not exist, and if server *A* receives the response, it ensures the contact with server *B*. In this process, there are two key values: hello-interval Δt, which means server *A* would send a hello-packet every Δt; hold-time *T-hold*, which indicates that server *A* should receive the reply before *T-hold* expires, otherwise, server *A* determines that server *B* fails to respond to the hello-packet. And if server *B* fails to answer the hello-packet three times continuously, server *A* would conclude that server *B* is unavailable now and start to process the caching behavior we explain before. Thus, server *A* is able to detect the failure of the relevant server *B* in at least $(2*\Delta t + T\text{-}hold)$, which means server *A* needs the failures of three continuous hello-packets. The values of hello-interval Δt and hold-time *T-hold* are determined on many conditions, one of which is how fast the administrator needs a DNS server to detect the unavailability of a relevant server, and another of which is the DNS-query answering capability of the relevant server, for we don't want hello-packets to burden a relevant DNS server too much.

Through this method, we can apply keep-alive mechanism on the existing DNS framework independently and unilaterally, which means we don't need the other side of the keep-alive peer to support this mechanism. And also, there are many other keep-alive methods working on the Internet, which is bilateral, such as OSPF hello protocol [11], BGP keep-alive messages based on TCP [12]and Bidirectional Forwarding Detection (BFD) [13]. However, the above methods require mutually interact, which are not immediately deployable in the existing DNS.

III. EVALUEATION

In order to evaluate the effectiveness of our modification in the DNS caching behavior, we collected DNS traffic on three DNS servers of an ISP (Internet Service Provider) in China. These traces were collected for a period of 10 days-from 27th Nov, 2009 to 6th Dec, 2009. The details of the traces are given by Table 1.

TABLE I. DETAILS OF TRACES

Trace	Organization	Clients	Requests	Names
Trace1	ISP	57148	64247870	753324
Trace2	ISP	57489	76899400	860525
Trace3	ISP	240437	46467908	1393831

In Fig.1, we show the trend of growth in the number of new domain-names which are not queried before. The growth of new domain-names is important to our scheme, because if new domain-names emerge rapidly, it would be unreasonable to response to queries according to historical records.

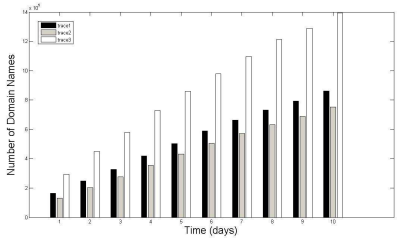

Figure 1. Details of Traces

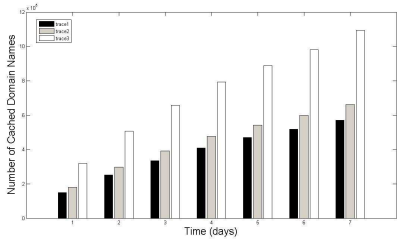

Figure 2. Domain-names Cached

In order to simulate the DNS DDoS attacks, we assume that relevant DNS servers of these three DNS servers were under attack on the same day and all queries cannot be processed by the relevant DNS servers. Domain-names queried before the attack are cached for a period of TTL, and due to our scheme, any TTL-expired record in the cache would not be evicted until the DNS servers detect that the relevant servers are available again, and these three DNS servers can still resolve domain-names based on the historical records cached. Then, we support that the relevant DNS servers need a certain time R to recover from the attack and be available. Such a simulator is governed by two key parameters:

- *Time-to-Live*: Time-to-Live of a record determines how many domain-names can be cached when a DDoS attack happens. In our simulation, we vary TTL from 1 day to 7 days. In Fig.2, we measure the accurate items cached for a range of TTL.
- *Recovery Time R*: This evaluates our scheme in a certain period of time, which is also important in a DNS DDoS attack. Our scheme needs to work as long as the relevant DNS servers resume to work. We simulate these three DNS servers can get contact with other DNS servers in 1, 2 and 3 days, and the result is presented in the following figures.

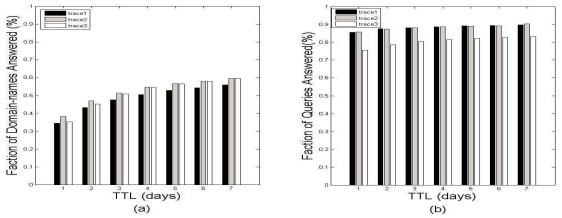

Figure 3. For (a) Domain-names and for (b) Queries Answered in the 1st day of the attack

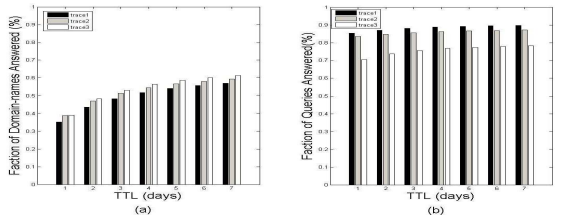

Figure 4. For (a) Domain-names and for (b) Queries Answered in the 2nd day of the attack

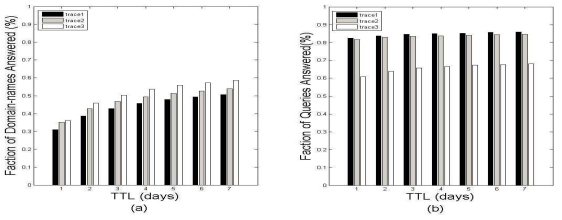

Figure 5. For (a) Domain-names and for (b) Queries Answered in the 3rd day of the attack

According to our analyze, with the scheme we propose, the three DNS servers are still capable to answer about 45% of all the domain-names queried and about 85% of all the queries in the 1st day of the DDoS attack and it does not deteriorate fast in the next 2 days. Take Trace 1 for example, when TTL=1, in the 1st day of the attack, the DNS server can successfully answer about 85.61% of the total queries, while in the 3rd day, it's about 82.44%, which lowers for only 3.17%. So it is with the rate of domain-names answered, which drops for only 3.55%. However, the DNS server can only answer 34.47% of all the domain-names queried, while it is 85.61% referring to the fraction of all the queries answered. Fortunately, what we are concerned about is to satisfy as many DNS requests as possible in a harsh situation such as the DNS DDoS attacks, so it's pleasing to get such a high rate of the total queries that can be answered according to historical records, disregarding the low one of the total domain-names answered. We notice that the fraction of queries answered is quite higher than that of the domain-names, from which we can conclude that there are much more queries for the domain-names cached than the ones not in the cache. Further, we can determine that in the DNS, most of the queries inquire about domain-names that were recently asked, which proves that the caching behavior is quite reasonable and effective.

Also, we can see that the longer domain-names is reserved, the more domain-names and queries can be answered during the attack. As it shows in Fig.2, the DNS server of Trace 1 needs to cache 571304 domain-name records when TTL=7, while only 149668 when TTL=1. We cache 421636 more or 2.81 times more records to increase the fraction of all queries answered from 85.61% to 89.82% in the 1st day of the DDoS attack, which is not obviously effective. Thus, we can conclude that lots of the domain-names cached are not frequently queried, while some of the others very popular.

IV. DISCUSSION

In this section, we consider more about some points that relate to our scheme and should be paid more attention to.

1) *Define Failed Query:* In our scheme, we cached not only domain-names that are resolved successfully, such as the IP address, the CNAME of the domain-name, but also records that indicate failure, such as NXDOMAIN, Server Failure and Refused. And we define any query that can be answered according to the records in the cache, which include success records and failure records, as a successful query. Firstly, we hope to minimize the impact of the DNS DDoS attacks on the DNS servers, so we prefer that the DNS servers under attack can response consistently with before. For that reason, we cache every response to accomplish that the DNS servers can response normally, in the scope of both success and failure. Secondly, some of the ISPs provide a NXDOMAIN redirect service, replacing NXDOMAIN message with other accessible URLs (Uniform Resource Locator), such as some advertisement or URLs which match a number of key words in the domain-name queried. So, it's not accurate to

judge a domain-name non-exist just based on the response the DNS server reply.

2) *Optimal Time-to-Live:* As we can conclude according to Fig.3, Fig.4, and Fig.5, a longer TTL determines a better fraction of queries and domain-names that can be answered, but it's not significantly effective. Additionally, as Fig.2 shows, the number of domain-names cached increases stably with the growth of TTL. So, we must weigh both the memory of the DNS server and the successful response rate. Further, if the DNS records have been changed, the records cached would be obsolete, so it's inappropriate to reserve DNS records for a long TTL. However, as it proves in Ref.[14], the mappings between domain-names and IP addresses tend to be quite stable.

V. CONCLUSION

Considering to alleviate the impact of DNS DDoS attacks, we propose our solution in this paper and demonstrate the effectiveness of reserving TTL-expired records when a DNS server detects the unavailability of the relevant servers with the keep-alive scheme we propose. Also, we analyze the results with varying TTLs and Recovery Times. Our scheme can be immediately deployed in the existing DNS framework and improve the resilience of the DNS against DDoS attacks substantially.

REFERENCES

[1] M. Lottor, Domain administrators operations guide, RFC1033, 1987.

[2] P.V. Mockapetris, Domain names concepts and facilities, RFC1034, 1987.

[3] P.V. Mockapetris, Domain names implementation and specification. RFC1035, 1987.

[4] "Root Server DDoS Attack, RIPE Mail Archive", https://www.ripe.net/ripe/maillists/archives/eof-list/2002/msg00009.html, 2002.

[5] UltraDNS DoS Attack. http://www.theregister.co.uk/2002/12/14/ddos_attack_really_really_tested/,2002.

[6] 'Zombie' PCs attack Akamai. http://news.cnet.com/2100-1038_3-5236403.html , 2004.

[7] DNSPod DNS DDoS Attack. http://support.dnspod.com/index.php?_m=news&_a=viewnews&newsid=7, 2009

[8] T. Hardie, Distributing Authoritative Name Servers via Shared Unicast Addresses, RFC3258, 2002

[9] Giuseppe Ateniese , Stefan Mangard . A New Approach to DNS Security , ACM CCS'01, 2001

[10] V. Ramasubramanian and E. G. Sirer, "The Design and Implementation of a Next Generation Name Service for the Internet," in Proc of ACM SIGCOMM, 2004.

[11] J. Moy, OSPF Version 2, RFC2328, 1998

[12] Y. Rekhter, T. Li, A Border Gateway Protocol 4, RFC1771, 1995

[13] Bidirectional Forwarding Detection, http://www.ietf.org/dyn/wg/charter/bfd-charter.html, 2008

[14] Lindsey Poole and Vivek S. Pai, ConfiDNS: Leveraging Scale and History to Improve DNS Security, Proceedings of WORLDS, 2006