侵入検知による強固な DNS の設計

力武 健次 ‡⋆　竹森 敬祐 ‡　三宅 優 ‡　中尾 康二 ‡
野川 裕記 ⋆　下條 真司 ⋆

‡ 株式会社 KDDI 研究所 ネットワークセキュリティグループ
〒356-8502 埼玉県上福岡市大原 2-1-15
⋆ 大阪大学 サイバーメディアセンター
〒567-0047 大阪府茨木市美穂ヶ丘 5-1

DNS(ドメイン名システム) サーバとリゾルバ (クライアント) は本質的に分散 DoS(サービス拒否攻撃) やスタックスマッシングなどの攻撃を受けやすい。これは DNS の問合せ用プロトコルが UDP を土台としており、これらの攻撃に対して弱いからである。本稿では、まずサーバとリゾルバ間の通信の確実性に注目した DNS 設計上の問題を考察する。そしてサービス中断のリスクを減らす設計手法として、T/TCP(トランザクショナル TCP) および DNS プロトコルのための IDS(侵入検知システム) を使う方法を提案する。

キーワード: インターネット、セキュリティ、ドメイン名システム、侵入検知

Design of Robust DNS by Intrusion Detection

Kenji Rikitake‡⋆, Keisuke Takemori‡, Yutaka Miyake‡, Koji Nakao‡,
Hiroki Nogawa⋆ and Shinji Shimojo⋆

‡ Network Security Laboratory, KDDI R&D Laboratories, Inc.
2-1-15 Ohara, Kamifukuoka City, Saitama 356-8502 JAPAN
⋆ Cybermedia Center, Osaka University
5-1, Mihogaoka, Ibaraki City, Osaka 567-0047 JAPAN
{kenji,takemori,miyake,nakao}@kddilabs.jp, {nogawa,shimojo}@cmc.osaka-u.ac.jp

DNS (Domain Name System) servers and resolvers (clients) are inherently prone to the simple forms of attack, such as DDoS (Distributed Denial of Service) and stack smashing, since the basic query protocol is based on UDP, which has no protection against these types of attacks. In this paper, we analyze the DNS design issues regarding the communication robustness between the servers and resolvers, and propose a design method to reduce the risk of the service disruption by introducing T/TCP (Transactional TCP) and the IDS (Intrusion Detection System) for the DNS protocol.

Keywords: Internet, Security, Domain Name System, Intrusion Detection

1 Introduction

Internet protocols and servers have been under continuous and persistent attacks as real-world activities become dependent on the global computer network. All well-known protocols are the targets of the intruders to exploit, since compromising and taking control of a host can lead to a financial and social gain, if not technical.

One of the most important component of the Internet Protocol Suite is DNS (Domain Name System) [13] [14]. In this paper, We summarize some of the outstanding security issues of DNS and propose a design to enforce the robustness of communication between the DNS servers and resolvers (clients).

We first describe the characteristics of ongoing major attacks of Internet in Section 2. We then describe the outstanding security issues on DNS in Section 3, and analyze the implementation issues in Section 4. And in Section 5, we propose an implementation model for making DNS more robust.

2 DoS Attacks on Internet

One of the most popular example of the attacks on Internet is *DoS (Denial of Service)*. A typical form of the DoS attack is sending incomprehensive packets to paralyze a server and to hamper the performance of processing requests from the clients. DoS is an old and well-known method to cause damages to an information system by reducing the availability of the system resources [9].

The global and ubiquitous availability of Internet, however, allows a even sophisticated method called *DDos (Distributed DoS)*. By DDoS, globally-dispersed intruders can act together while keeping themselves undetected using multiple computers simultaneously to attack a single server. Some intruders even forge packets by using valid third-party source addresses for the attacks so that the attack victim cannot easily distinguish the real and spoofed packets [10].

Even the simplest form of the brute-force DDoS, such as sending incomprehensive HTTP (HyperText Transfer Protocol) [20] requests simultaneously to a Web server, has successfully damaged many business activities ongoing over Internet.

A more sophisticated form of popular attacks on Internet is forcing a server to accept a malformed packet. The packet contains a piece of information compomising the server program by attacking the security hole of allowing access to the stack area of the program. This type of attack is called *stack smashing*, also known as the *buffer-overflow attack*.

The degree of damage caused by the stack smashing is much serious than a brute-force DoS, because the intruder can take over the attacked computer and make it a platform for launching another attack. Popular software packages have been major targets of the vulnerability discovery for stack smashing, as the more programs running on a computer gets complex, the more the vulnerablility increases against the intrusion attempts.

3 DNS and The Vulnerabilities

DNS servers and resolvers are one of the most important set of service-providing entities among the TCP/IP Network, since *all* Internet hosts are expected to support two-way lookups between the domain names and the IP addresses.

Every Internet-connected entity who is assigned a domain name must maintain the respective DNS server publicly accessible. Even in the private internal network, being able to resolve inter-organizational domain names to IP addresses is mandatory for a practical wide-area distributed operation. Mission-critical applications, such as electronic mail and Web, all profoundly depend on the availability of DNS, to ensure the integrity of identifiers such as the URLs (Uniform Resource Locators) and the mail addresses.

The daily DNS operation becomes more and more vulnerable as Internet becomes larger and more complex. Some of the current issues which should be solved to reduce the vulnerability of DNS are listed as follows:

- DNS uses UDP (User Datagram Protocol) [11] for the primary query protocol between the resolvers and the servers. When using UDP, the size of the response data block to the query *must not exceed 512 bytes* to avoid fragmentation of the single IP datagram, according to the Section 3.3.2 of RFC1122 [15], which defines the Host Requiments.
- DNS Cache and Zone-data servers must be

able to directly exchange the DNS UDP packets (on Port 53) to provide the services. This indicates a host that provides DNS service is always prone to the brute-force and other DoS-type attacks to the UDP port 53.

- The 512-byte limitation of DNS response packet size also leads to the restriction of limiting the number of DNS Root Servers to 13 [*1]. This inherently restricts the number of domain names which the entire Internet can support, since unresolvable queries from all the Internet hosts are eventually directed to the Root Servers.
- Windows2000 and Windows XP [*2] Operating Systems use DNS Dynamic Update [19] and the security enhancement for the Dynamic Update protocol [21] to support the plug-and-play configuration of the IP address and the domain name information. While this functionality is getting more popularity among inter-organizational networks, deploying this protocol to the public Internet still requires further feasibility and scalability analysis, since allowing the update of DNS database may allow intruders to do so as well.
- BIND [1], the de-facto standard of DNS server and resolver, have many serious security holes related to the buffer-overflow attacks. For example, A set of bugs on the resolver [4] forced major OS distributions such as FreeBSD to upgrade [6] , since so many programs depend on the resolver library. Another set of bugs exposes a vulnerability of BIND server into shutting down itself by a malformed external packet [3].

The above list does not contain the vulnerabilities caused by the configuration error of the DNS administrators. Nevertheless, redesigning DNS for gaining robustness against the DDoS and other attacks by solving the configuration-independent vulnerability issues is a practical candidate of an academic research, and the research results will be immediately applicable to a production-level network system, including the public Internet.

[*1] Only up to 13 NS Resource Records and the corresponding A RRs will fit into a DNS message of maximum 512 bytes.

[*2] *Windows2000* and *Windows XP* are registered trademarks of Microsoft, Inc.

4 Implementation Issues of Robust DNS

Making DNS more robust against various forms of attacks requires wide-range and detailed research. In this section we list up some critical issues to solve for building a robust DNS system.

4.1 Using T/TCP for DNS Queries

DNS is a distributed database system, so exchanges between the resolvers and servers can be considered as a set of transactions. One of the major security issue is that DNS is largely dependent on UDP for the transaction.

DNS resolvers are assumed first to use UDP, and then TCP (Transmission Control Protocol) [12], because most of the responses from DNS servers are considered to fit into the 512-byte size limit of a UDP packet. While this design philosophy of DNS works well on Internet, it also opens a vulnerability for the intruders, by opening both UDP and TCP ports for listening.

A DNS server must accept the UDP-based requests. This allows DDoS traffic sent to the UDP port, which cannot easily be filtered out or rejected by a packet-filtering device. DNS is an inter-organizational protocol, so the site administrator must leave the communication channel open to the public to provide the service on Internet.

On the other hand, network administrators can limit the communication range of other UDP-based protocols such as NTP (Network Time Protocol) [18] and SYSLOG [22], because they are intented to serve within a limited number of hosts. Packets of those protocols can be effectively filtered out by the boundary routers.

If DNS could get rid of UDP, protecting DNS against DDoS would be far easier than now, since DNS Cache and the Zone-data servers do not have to accept arbitrary packets. T/TCP (Transactional TCP) [17] is a good candidate to substitute UDP, because it is designed specifically for query-and-response transactions.

4.2 Defending DNS by IDS

Another approach of defending DNS from attacks is to monitor the packets and alert the possibility of intrusion by the IDS. Since the DNS queries consists on a single UDP packet, exten-

sively monitoring the packet by a specifically-designed IDS can be an effective solution, especially to prevent compromising by known form of attacks.

Note that the TCP port for DNS (Port 53) must also be monitored by the IDS as well. When using T/TCP, the protocol-specific characteristics should also be equally considered.

The IDS can also be used for monitoring the protocol integrity, by validating the protocol state from the header and the content information. This is also useful to discover possible attacks by violating the protocol or sending malformed packets.

The DNS Dynamic Update protocol demands a major security change to the DNS software, because the server must handle the Zone database in read-write mode from the previous read-only mode. Monitoring the DNS packets between a server handles the Dynamic Update and the resolvers is even more effective, since security holes of the program handles the updates may result in more serious damages than the previous vulnerabilities.

4.3 Optimizing DNS Software Design

DNS software packages have a lot of room to optimize, to prevent the emergence of security holes and to gain the robustness against DoS attacks. For example, the all-in-one design philosophy of BIND contributes to the discovery of the security holes. An alternative implementation djbdns [5] has successful results on eliminating the security holes, by choosing the modularized design which splits the DNS cache (`dnscache`) and the zone-data server (`tinydns`).

The `tinydns` program also introduces single database file approach for the whole zones which it serves, while BIND chooses multiple-file approach. This structural difference affects the amount of the response time between the two servers.

5 Our Proposal of Robust DNS

Regarding the issues described in the previous Sections 3 and 4, we propose the following implementation methods for a robust DNS design, and discuss the problems needed to be solved.

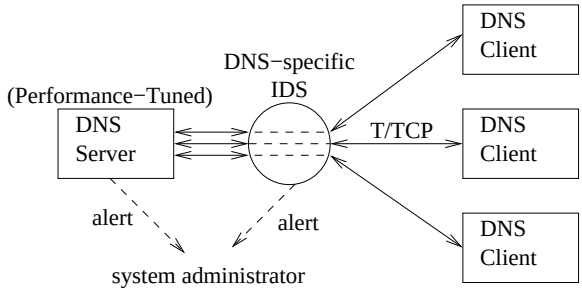

Fig. 1 Block Diagram of The Proposed Robust DNS

Figure 1 shows the block diagram of the proposed DNS components. In this set of components we use three implementation techniques to enforce the robustness of the DNS.

5.1 Using T/TCP for Transactions

The advantage of using T/TCP is to eliminate UDP and to optimize the usage of TCP on DNS transactions. This will ease the burden of writing a firewall [2] ruleset, since controlling TCP traffics is much easier than controlling UDP traffics.

T/TCP has been implemented to various BSD UNIX distributions and the details have already been published as a book [24]. Linux version of T/TCP has been implemented and have a successful result on HTTP [8].

One of the authors Rikitake has implemented a T/TCP version of Identification Protocol [16] client as a part of the WN Web Server [25], by modifying the existing non-transactional TCP code. This suggests that implementing a T/TCP code for existing DNS resolvers is not a difficult task to perform.

We will experiment replacing UDP-based query-and-response code of current DNS implementations by a newly-developed T/TCP-based code. We also need to perform a profound robustness analysis of T/TCP, since several research results indicate that T/TCP have certain weaknesses to DoS attacks, such as the TCP SYN-flooding [10] [8].

5.2 Making DNS-specific IDS

The advantage of making a DNS-specific IDS is to reduce the complexity of the implementation so that the IDS programmer can concentrate on detecting DNS-specific anomalies.

While the generic network-wide IDS approach [7] has been effective to help network administration issues and enhance security of the

overall systems, making a protocol-specific IDS is a practical method to find out protocol-specific attacks, as well as well-known attacks such as DDoS and stack smashing.

We will experiment making a DNS-specific IDS by modifying a popular implementation such as Snort [23], and by adding a new functionality to the host operating system kernel in which the DNS server and the IDS are running.

5.3 DNS Performance Tuning

The advantage of tuning DNS performance is to maximize the processing ability of DNS servers and resolvers, and to find out the potential bottlenecks for the larger-scale DNS deployments such as that on Internet. A faster DNS will give higher tolerance to the brute-force attacks, such as DDoS.

Current DNS servers and resolvers have caused some disruptions to the worldwide Internet operations, due to some major bugs in the implementations. Precise performance analysis may contribute to solve the current problems of DNS operation and the software design.

6 Summary and Further Works

In this paper we have summarized the current security issues of DNS. We have also proposed a design for a robust DNS with the justification of the design details.

The implementation and detailed evaluation are currently ongoing as of September 2002.

Acknowledgements

Our thanks go to Mr. Tohru Asami, the president of KDDI R&D Laboratories, Inc. for supporing our research activities.

References

[1] Internet Software Consortium: BIND, `http://www.isc.org/bind/`

[2] E. D. Zwicky, S. Cooper, and D. B. Chapman: *Building Internet Firewalls (2nd Edition)*, O'Reilly & Associates, ISBN 1-56592-871-7 (2000).

[3] CERT/CC: *Denial-of-Service Vulnerability in ISC BIND 9*, CERT Advisory CA-2002-15 (last revised: August 8, 2002). `http://www.cert.org/advisories/CA-2002-19.html`

[4] CERT/CC: *Buffer Overflows in Multiple DNS Resolver Libraries*, CERT Advisory CA-2002-19 (last revised: August 28, 2002). `http://www.cert.org/advisories/CA-2002-19.html`

[5] D. J. Bernstein: *djbdns*. `http://cr.yp.to/djbdns.html`

[6] The FreeBSD Project: *Buffer Overflow in Resolver*, FreeBSD Security Advisory FreeBSD-SA-02:28.resolv, June 26, 2002, `ftp://ftp.FreeBSD.org/pub/FreeBSD/CERT/advisories/FreeBSD-SA-02:28.resolv.asc`

[7] E. Amoroso: *Intrusion Detection*, Intrusion.Net Books, ISBN 0-9666700-7-8 (1999).

[8] M. Stacey, J. Nelson, and I. Griffin: *T/TCP: TCP for Transactions*, Linux Gazette Issue #47, Linux Journal (1999), `http://www.linuxgazette.com/issue47/stacey.html`

[9] S. Garfinkel, G. Spafford: *Chapter 25: Denial of Service Attacks and Solutions, Practical UNIX & Internet Security (2nd Edition)*, O'Reilly & Associates, ISBN 1-56592-148-8, pp. 759–778 (1996)

[10] V. Paxson: *An analysis of using reflectors for distributed denial-of-service attacks*, Computer Communication Review, Vol. 31, No. 3, pp. 38–47 (2001).

[11] J. Postel: *User Datagram Protocol*, RFC768 (also STD6) (1980).

[12] J. Postel: *Transmission Control Protocol*, RFC793 (also STD7) (1981).

[13] P. V. Mockapetris: *Domain names – concepts and facilities*, RFC1034 (also STD13) (1987).

[14] P. V. Mockapetris: *Domain names – implementation and specification*, RFC1035 (also STD13) (1987).

[15] R. Braden (Editor): *Requirements for Internet Hosts – Communication Layers*, RFC1122 (1989).

[16] M. St. Johns: *Identification Protocol*, RFC1413 (1993).

[17] R. Braden: *T/TCP – TCP Extensions for Transactions Functional Specification*, RFC1644 (1994).

[18] D. Mills: *Simple Network Time Protocol*

(SNTP) Version 4 for IPv4, IPv6 and OSI, RFC2030 (1996).

[19] P. Vixie (Editor), S. Thomson, Y. Rekhter, J. Bound: *Dynamic Updates in the Domain Name System (DNS UPDATE)*, RFC2136 (1997).

[20] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee: *Hypertext Transfer Protocol – HTTP/1.1*, RFC2616 (1999).

[21] B. Wellington: *Secure Domain Name System (DNS) Dynamic Update*, RFC3007 (2000).

[22] C. Lonvick: *The BSD syslog Protocol*, RFC3164 (2001).

[23] B. Caswell, M. Roesch et al., *Snort*, `http://www.snort.org/`

[24] W. Richard Stevens: *Part 1: TCP for Transactions*, *TCP/IP Illustrated, Volume 3*, Addison-Wesley, ISBN 0-201-63495-3 (1996), pp. 3–158

[25] J. Franks: The WN Web Server, `http://hopf.math.nwu.edu/`