Global Information Assurance Certification Paper

Recent BIND Vulnerabilities With an Emphasis on the "tsig bug"

Alicia Squires
February 16, 2001

The Internet shapes much of our world today, and the Internet infrastructure itself is highly dependent upon Domain Name Servers. RFC 2845 defines a Domain Name Server (DNS) as "a replicated hierarchical distributed database system that provides information fundamental to Internet operations, such as name ⇔ address translation and mail handling information". So what does it mean when the most prevalent name server, Berkeley Internet Name Domain (BIND) server, is riddled with vulnerabilities? It implies that the very core of our electronic economy is on the brink of widespread disaster.

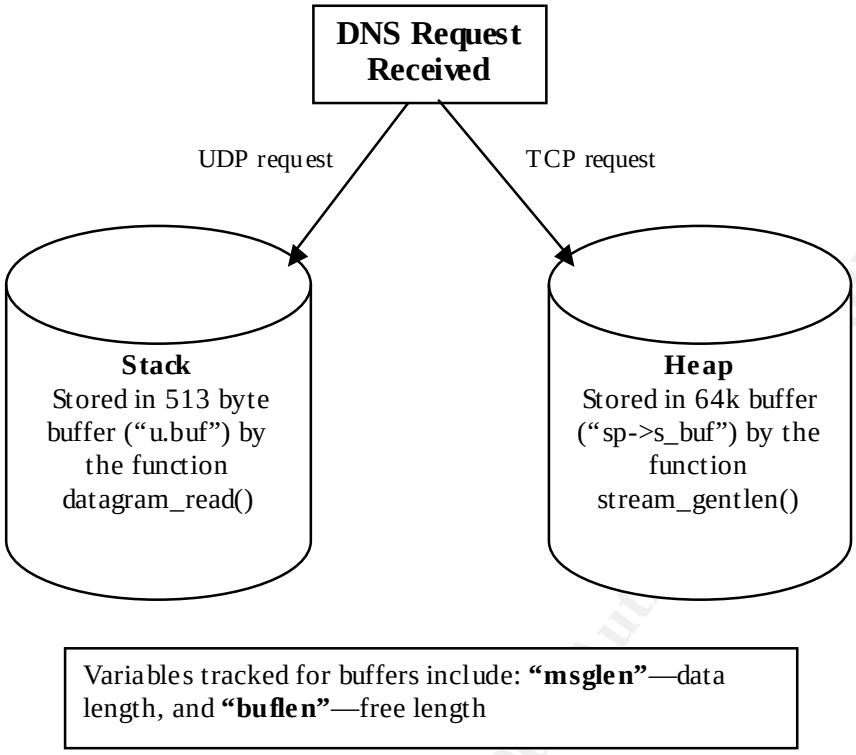
On January 29, 2001 Network Associates of California released a report that documented four recent vulnerabilities in BIND. Two of these vulnerabilities were buffer overflows, which can allow an attacker to either shut down the Domain Name Server (DNS) or gain root control of it. The DNS does this by accepting more data in a certain part of the program than it can handle. The extra data or code gets put into the memory that stores instructions, and eventually the code gets executed with the program. One of the buffer overflows is called the "tsig bug", which affects BIND version 8, and the other is the "complain bug" buffer overflow, which affects BIND version 4. The remaining two vulnerabilities documented by Network Associates include "infoleak", which affects both BIND 4 and 8, and the "complain bug" format string vulnerability, which affects only BIND 4. These vulnerabilities are the eleventh through the fourteenth vulnerabilities discovered with BIND since its design in the early 1980's. This majority of this paper will be dedicated to the "tsig bug", as it is currently the best documented.

The "tsig bug"

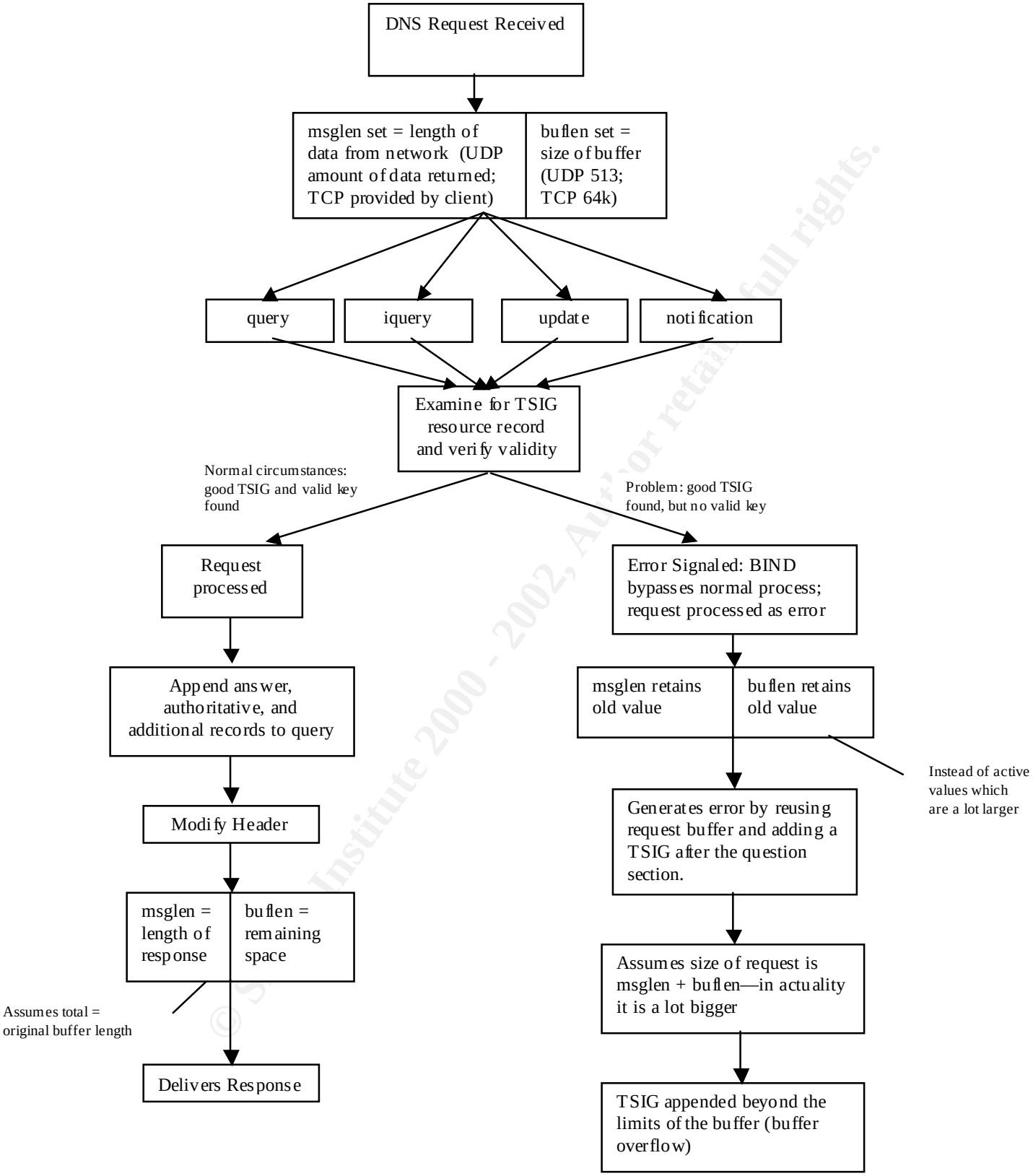
The "tsig bug" gets its name from the transaction signature (TSIG) that is used to authenticate communication between DNS servers in BIND 8.2.The TSIG bug is the most serious of the four recent BIND vulnerabilities. A TSIG is a higher-level DNS resource record. It is calculated for each DNS request or response, and then it is discarded. It is not reusable and should not be kept in the cache.

The TSIG is a complicated security mechanism. It has to be the last record in the additional section of the message. If there are multiple TSIG's present, or if it is in the wrong position, the package is dropped and an error code is sent back. TSIG's are also verified with two timer values, 'Time Signed' and 'Fudge', and a keyed message digest operation is performed to obtain a matching key. These mechanisms prevent attackers from grabbing random transaction signatures to reuse, although the logging of TSIG errors can create a possible Denial of Service condition and should be handled with care.

The TSIG bug affects domain name servers running BIND version 8.2 (any service pack), 8.2.1, 8.2.2 (packs 1-7), and all 8.2.3-betas. It is a very serious buffer overflow that can allow the attacker to gain remote access to and execute arbitrary code by invalidating the logic used by BIND to calculate the request buffer length. The TSIG vulnerability is illustrated in the following diagram and flow chart.



Variables tracked for buffers include: **"msglen"**—data length, and **"buflen"**—free length



When a DNS request is received it is sorted into the stack or the heap depending on transfer protocol. TCP requests are stored in a 64k buffer from the heap by the function *stream_gentlen()*. UDP requests are stored in a 513-byte buffer in the stack by the function *datagram_read()*. BIND handles requests by reading them in TCP or UDP, modifying them, and then creating an appropriate response by appending to the message the answer, authoritative, and additional records.

There are two variables that are maintained over the course of the request, including the "msglen," which refers to the length of data in the buffer, and the "buflen," which is equal to the remaining length in the buffer. When added together these two variables should always equal the size of the buffer.

For initial processing of the request msglen is set to the length of data received from the network. For UDP, this is the amount of data returned from a *recvfrom()* call, and for TCP this value is provided by the client. The buflen is set to the total buffer size: 64 K for TCP and 513 for UDP. It is then determined if the request is a query, iquery, update, or notification.

At this point, prior to processing the request, the TSIG comes into play. BIND searches the request for a valid TSIG resource record with the function *ns_find_tsig()*. It must also find a valid security key. If both of these items are present, the request is then processed normally. Without both of these items, an error is indicated and normal procedures are bypassed. Instead of processing the request, appending the necessary records, and modifying the header values—msglen and buflen—to their new values, the values of msglen and buflen are locked. Reusing the request buffer and adding a TSIG after the question section complete the error. This happens, because BIND's error-handling code initializes variables, msglen and buflen, in a different way than the normal process. This causes the later assumption of the buffer length based on these values to be incorrect.

The total value of the variables msglen and buflen is still assumed to be the total size of the buffer, when in reality the buffer size is much larger, so when the TSIG is appended at the end of the error buffer by the function *ns_sign()* it is beyond the limits of the buffer. This can allow the execution of arbitrary code by overwriting adjacent memory on the stack or heap.

For example, Company A has a DNS running BIND 8.2.2-Patch 5. An attacker finds the DNS along with this information by doing a simple portscan of a range of IP's and then a version.bind request of the server. The attacker then sends a TCP DNS request to this server that has been crafted to have a valid TSIG and an invalid security key. Company A's DNS receives the request, checks for the TSIG, finds it, and then signals an error when no valid security key is found. At this point the variables that track the buffer size are locked and will not reflect the actual size throughout the rest of the error processing. When the new TSIG is appended at the end of the request before returning it, it writes past the end of the buffer into the heap (stack if it is a UDP request) and overwrites memory. This can be exploited to allow code to be executed with the permissions of *named*, which is usually root.

Other Vulnerabilities

The next most serious of the recent BIND vulnerabilities is the "complain" buffer overflow that affects only BIND version 4, specifically 4.9.3 – 4.9.7. This vulnerability is the result of a stack overflow, which results from the way that the *sprintf()* function constructs error messages unsafely. For a domain name server to be vulnerable to this attack, it must be running BIND version 4 and be recursive. The attacker must also have control of or access to an authoritative DNS. It can result in a denial of service or execution of arbitrary code with the permissions of *named*.

As of this writing, no exploits for the "complain" buffer overflow have been released to the security community, although they almost certainly already exist in the Black Hat hacker community.

There is an additional vulnerability related to the "complain" buffer in BIND version 4. It is the NslookupComplain() Format String vulnerability. This vulnerability is the result of the way that BIND reports an error to the syslog when it is trying to determine IP addresses for name servers. As with the "complain" buffer overflow, no exploits for the "complain" format string vulnerability have been released to the security community yet, although they almost certainly already exist in the Black Hat hacker community.

The fourth of the BIND vulnerabilities that were recently discovered and released was the "Information" or "Infoleak" vulnerability. This vulnerability is different from previous BIND weaknesses, because it affects both BIND version 4 and 8. It allows an attacker to view the *named* process memory, which can give valuable information for future exploits. One of the exploits for the TSIG buffer overflow also exploits this vulnerability,

Exploits

There is an exploit for the TSIG buffer overflow that was released recently by Bugtraq. Interestingly a trojaned version of the script was released a few days earlier. Bugtraq released it on the message board without decompiling and analyzing it to see the true contents, and several people downloaded and tested it. As it turns out, hidden in hex shell code was a command to open multiple instances of itself and launch a Denial of Service attack against Network Associates, who originally discovered the BIND vulnerabilities.

The fixed exploit was posted on Packet Storm (http://packetstorm.securify.com/0102-exploits/bind8x.c), and it is a C program that can be compiles and used to exploit both the TSIG bug and the "Infoleak" vulnerability. The exploit forms several packets to be used for the buffer overflow, and interspersed in the packets are system calls that execute various commands. The first command is *socket()*, which creates an endpoint for the communication. Next *bind()* binds a name to the created socket allowing it to receive connections. This allows the attacker to telnet to the port with no authentication. *Listen()* listens for connections to the socket, operating in a continuous loop while waiting. *Accept()* accepts the connection to the socket. *Dup2()* duplicates file descriptors, making 3 copies of itself, and *execve()* executes a program. For this exploit the program is "/bin/sh," which gives a root shell to the attacker if *named* is running as root.

Recommendations

The high severity of these vulnerabilities is apparent, and vulnerable domain name server's *named* should be protected in one (or more) of the following ways. *Named* should be run as a normal user account, so that if it is compromised the attacker will not have root access and all of the privileges that come with it. *Named* should also be protected within a restricted filesystem with a "chroot" environment. This means that an environment must be designed where an account is confined to specific directories by making a *chroot()* call, which makes a root directory request actually point to a different file. In essence, it creates a smaller (chroot jail) filesystem within the larger one, and user accounts for the smaller filesystem cannot see, access, or execute commands against the larger filesystem. If *named* were given a user account within a restricted filesystem, an attacker would be locked within the jail, and would not be able to execute root level commands against the domain name server.

In addition to protecting *named*, it is highly advisable to upgrade any domain name servers running BIND 4.9.x or 8.2.x to BIND 4.9.8, 8.2.3, or 9.1. They are available from the Internet Software Consortium (ISC) at http://www.isc.org/.

References:

1. Vixie, P., Gudmundsson, O., Eastlake, D., and Wellington, B. "RFC 2845". May 2000. URL: http://www.landfield.com/rfcs/rfc2845.html. (2/5/2001).
2. Internet Software Consortium. "BIND vulnerabilities". URL: http://www.isc.org/products/BIND/bind-security.html. (1/29/01).
3. Network Associates, Inc. "Vulnerabilities in BIND 4 and 8". January 29, 2001. URL: http://www.pgp.com/research/covert/advisories/047.asp. (2/5/2001).
4. Carnegie Mellon Software Engineering Institute. "CERT Advisory CA-2001-02 Multiple Vulnerabilities in BIND". Revised February 02, 2001. URL: http://www.cert.org/advisories/CA-2001-02.html. (2/5/2001).
5. Poulsen, Kevin. "BIND holes mean big trouble". January 29, 2001. URL: http://www.securityfocus.com/templates/article.html?id=144. (1/29/2001).
6. SecurityFocus.com. "Bugtraq ID 2309; ISC BIND 4 nslookupComplain() Format String Vulnerability". January 29, 2001. URL: http://www.securityfocus.com/vdb/bottom.html?vid=2309. (2/12/2001).
7. SecurityFocus.com. "Bugtraq ID 2321; ISC BIND Internal Memory Disclosure Vulnerability". January 29, 2001. URL: http://www.securityfocus.com/vdb/bottom.html?vid=2321. (2/12/2001).
8. SecurityFocus.com. "Bugtraq ID 2302; ISC BIND 8 Transaction Signatures Buffer Overflow Vulnerability". January 29, 2001. URL: http://www.securityfocus.com/vdb/bottom.html?vid=2302. (2/12/2001).
9. SecurityFocus.com. "Bugtraq ID 2309; ISC BIND 4 nslookupComplain() Buffer Overflow Vulnerability". January 29, 2001. URL: http://www.securityfocus.com/vdb/bottom.html?vid=2307. (2/12/2001).
10. Packet Storm. "Remote Vulnerabilities in BIND versions 4 and 8". January 29, 2001. URL: http://packetstorm.securify.com/advisories/iss/iss.01-01-29.bind. (2/13/2001).

11. Hanley, Sinead. “DNS Overview with a discussion of DNS Spoofing”. November 6, 2000. URL: http://www.sans.org/infosecFAQ/DNS.htm. (2/5/2001).
12. Athanasiou, Ken. “DNS Remote Root Exploit – ADM Named 8.2/8.2.1 NXT Remote Overflow”. April 17, 2000. URL: http://www.sans.org/infoseqFAQ/malicious/DNS_exploit.htm. (2/5/2001).

Last Updated: July 25th, 2017

Upcoming Training

SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
Community SANS Omaha SEC401*	Omaha, NE	Aug 14, 2017 - Aug 19, 2017	Community SANS
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
Virginia Beach 2017 - SEC401: Security Essentials Bootcamp Style	Virginia Beach, VA	Aug 21, 2017 - Aug 26, 2017	vLive
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
Community SANS Pasadena SEC401 @ NASA	Pasadena, CA	Aug 23, 2017 - Aug 30, 2017	Community SANS
Mentor Session - SEC401	Minneapolis, MN	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
Mentor Session - SEC401	Edmonton, AB	Sep 06, 2017 - Oct 18, 2017	Mentor
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
Mentor Session - SEC401	Ventura, CA	Sep 11, 2017 - Oct 12, 2017	Mentor
Community SANS Albany SEC401	Albany, NY	Sep 11, 2017 - Sep 16, 2017	Community SANS
Community SANS Dallas SEC401	Dallas, TX	Sep 18, 2017 - Sep 23, 2017	Community SANS
Community SANS Columbia SEC401	Columbia, MD	Sep 18, 2017 - Sep 23, 2017	Community SANS
SANS Copenhagen 2017	Copenhagen, Denmark	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Boise SEC401	Boise, ID	Sep 25, 2017 - Sep 30, 2017	Community SANS
Baltimore Fall 2017 - SEC401: Security Essentials Bootcamp Style	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
Community SANS New York SEC401	New York, NY	Sep 25, 2017 - Sep 30, 2017	Community SANS
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Sacramento SEC401	Sacramento, CA	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Community SANS Charleston SEC401	Charleston, SC	Oct 02, 2017 - Oct 07, 2017	Community SANS
Mentor Session - SEC401	Arlington, VA	Oct 04, 2017 - Nov 15, 2017	Mentor
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event