Report # I733-004R-2010
Date: 02/14/2011

BIND 9 DNS Security

Enterprise Applications Division
of the
Systems and Network Analysis Center (SNAC)

Information Assurance Directorate

Author(s)
I733

National Security Agency
Attn: I733
9800 Savage Rd. Suite 6704
Ft. Meade, MD 20755-6704

(410) 854-6191 commercial
(410) 854-6510 facsimile

DERIVED FROM: NSA/CSS 1-52
DATED: 08 January 2007
DECLASSIFY ON: 20320108

BIND 9 DNS Security

Prepared by:

_____
I733

Released by:

_____
Chief, I733

Distribution:
1 – I7 Library
2 - Vital Records
3 – DJP2 STINFO Library No. S-252,608
4 – I73 Technical Reports web page
(https://snac.cyber.iaevals.nsa/Technical_Reports/)

BACKGROUND

BIND is an open source Domain Name Server (DNS) software package from the Internet Systems Consortium (ISC) commonly used to resolve host names to IP addresses and vice versa. As a key element in the Internet's infrastructure, DNS servers have often been targets of attack by hackers, spammers, and phishers. By taking a few simple steps, you can help protect your networks and help protect the Internet as well.

Attacks on DNS servers fall into several categories …

- BIND software bugs – a number of vulnerabilities have been found and fixed in the BIND software over the years. The total rewrite with BIND 9, along with the code scanning funded by the Department of Homeland Security makes it a fairly reliable package, but no software is perfect.
- System intrusion attacks – a DNS server is only as secure as the operating system it runs on. This risk include attacks on other network services that are running on the same machine. Once the underlying server has been compromised, the DNS server can be subverted and used for various malicious purposes.
- Denial of service attacks – since so many functions depend on having a responsive DNS service, anything that markedly affects its performance is going to have a major impact on your infrastructure.
- Network attacks – these attacks are based on the DNS protocol itself. The highest profile of these is Cache Poisoning, where a DNS server is convinced to cache an incorrect IP address, resulting in users of the server being sent to the wrong site. DNS zone transfers, dynamic updates, and notifies are also subject to attack. Access to unrestricted DNS information can be used to leverage other attacks.

This paper is intended to contain the minimum information needed to help a BIND 9 administrator minimize the security risks to their DNS server. It is assumed they have some familiarity with DNS and computers. For a more full coverage of the topics involved, the reader is directed to the book "DNS and Bind, Fifth Edition" and the NIST online guide "Secure Domain Named System (DNS) Deployment" listed in the references.

There are, of course, a number of issues outside the scope of this paper. They include securing the computer system that the BIND server is running on. **By far, the most important thing is to not run the BIND server on the same machine as a web server, mail server, or other commonly attacked network services.**

Firewall Issues

These days a BIND server will usually sit behind some sort of firewall. This raises some issues that need to be mentioned prior to discussing how to secure BIND 9.

- UDP vs TCP - without a doubt the number one rookie mistake with DNS and firewalls is not understanding that DNS is not a UDP only service. Most query requests will be sent

using UDP by default, but if an error occurs or if the UDP packet size is exceeded, BIND will shift to TCP. When someone only enables a UDP port on their firewall this typically causes randomly appearing problems that are very hard to diagnose .

- Random Query Port – this is a side effect of trying to harden against DNS query response spoofing. Basically, a DNS query response is verified by using a 16 bit field. This was adequate in the early low-threat days of the Internet, but is now fairly easy prey for attackers. In the past, BIND and other DNS implementations used port 53 for queries (responses always return on the same port) which made setting up the firewall easy. Now, a random query port is assigned each time by BIND, which adds a small amount of power to the 16 bit field. The problem is that this now requires the entire UDP and TCP range be opened to the BIND server at the firewall. Clearly this results in a security tradeoff . The alternative here is to use some sort of active firewall technology like CISCO Context Based Access Control (CBAC) which opens a hole in the firewall only long enough for a query response to return.

- Large packet lengths - between DNSSEC and IPV6, DNS packets are getting quite sizable. Besides the UDP packet length issue mentioned above, long DNS packets can cause a problem for some firewalls, especially more advanced ones, which needs to be considered. A related problem is that the increased delay in processing queries may exceed the default timeouts in some firewalls. For example, in a CISCO CBAC-based firewall, you may need to increase "ip inspect dns-timeout".

BIND 9 Security – Network Attacks

The key here is to restrict the various network operations to just the people you really need to talk to, or block them entirely if not needed. Even the sites you allow to access your DNS server should only be able to see the portion of the data they really need. This section assumes some basic knowledge of configuring BIND. If you need some background first, consult one of the references or the online BIND 9 Administrators Reference Manual at ISC (http://www.isc.org/software/bin/documentation).

1. Restrict Queries - A surprising number of DNS servers are configured to allow ANYONE to use them for name look-ups. Aside from performance and DOS issues, open DNS servers are targets for attackers looking to set up shop. To keep your DNS server from being an open recursive name server, options can be set in the /etc/named.conf file. Basically, you want to restrict two things:

- outsiders making recursive queries on your DNS server

- outsiders making queries against look-ups currently cached by BIND

There are two separate paths for applying these measures.

In BIND versions before 9.4, the "allow-query" option is used. For example, in /etc/named.conf

```
options {
     allow-query { 192.168.100/24 };
     allow-recursion { 192.168.100/24 };
}

zone "foo.com" {
     type master;
     allow-query {any};
     file "foo.com";
};
```

will prevent recursion and access to cache by outsiders while still leaving the zone data visible.

Beginning with BIND 9.4, a new option "allow-query-cache" has been added. The defaults for the relevant commands are now.

```
allow-query-cache {localhost;localnets;};
allow-recursion {localhost;localnets;};
allow-query {localhost;localnets;};
```

where localnets is defined as networks physically attached to the box running BIND. In many cases, you would specify an access control list defining your local networks that you want to allow recursive queries for, i.e.

```
acl "mynets" {
     127/8; 192.168.100/24; 192.168.120/24;
};

allow-recursion {mynets;};
```

Essentially, the new "allow-query-cache" lets you control access to already cached data with a global option, rather than having to put an extra "allow-query" in every zone. The above example would become

```
options {
     allow-query-cache { 192.168.100/24 };
     allow-recursion { 192.168.100/24 };
}

zone "foo.com" {
     type master;
     file "foo.com";
};
```

In fact, by default, starting with BIND 9.4, if you only set one of the three options, the other two

automatically take the same values.

2. Restrict Zone Transfers

You want to minimize the number of sites that can transfer the entire contents of your DNS zones. Zone transfers should be restricted to only the servers that need to maintain a copy of your zones. The "allow-transfer" option controls this. In BIND 9, "allow-transfer" works as a global option, i.e.

```
options {
     allow-transfer  { 192.168.100.56; 192.168.100.132;  };
}
```

or as a zone option

```
zone "foo.com" {
     type master;
     file "foo.com";
     allow-transfer  {none};
};
```

with the zone invocation overruling the global "allow-transfer", if there is one.

3. Restrict Dynamic Updates and Notifies

The best way to restrict dynamic updates is to not use them at all. They open up a lot of ways for attackers to compromise your DNS server. If you must use dynamic updating, then restrict it as much as possible with allow-update and allow-update-forwarding. You should also seriously be thinking about using some sort of cryptographic protection, like transaction signatures (TSIG).

```
zone "foo.com" {
     type master;
     file "foo.com";
     allow-update {192.168.100/24};
};
```

The allow-update-forwarding command is used on the BIND server that is been granted permission to send dynamic updates to your BIND server.

```
zone "foo.com" {
     type slave;
     file "fum.foo.com";
     allow-update-forwarding {192.168.133/24};
};
```

As for notifies, they are by default only accepted from your master DNS server. If you want to permit notifies from other DNS servers, then explicitly list them with allow-notify, i.e.

```
options {
     allow-notify  {192.168.100.12};
};
```

4. Views

Views provide separate virtual DNS servers in BIND 9 that can be used to isolate different zones. Most commonly, it is used to divide internal zones from external zones, on a firewall for example. The relevant commands are "view", "match-clients", "match destinations" and “match-recursive-only". Basically, the match statements define who can access a given virtual zone.

Note - there are some ways to make mistakes when using views. For example, the order that views are listed in /etc/named.conf matters. Also, some commands, like "acl" cannot be defined inside views (but can be defined outside a view and then used inside it).

```
acl "mynets" {
     127/8; 192.168.100/24; 192.168.120/24;
};

view "localnets" {
```

```
     match-clients { "mynets";);
     /* Place internal zones here */
};

view "outside" {
     match-clients { any; };
     / * Place external zones here */
};
```

The "match-destinations" variant is used with multi-IP DNS servers and the "match-recursion-only" is used to select recursive queries

BIND 9 Security – General

Keeping your DNS server secure is an ongoing process. If you install the latest version of BIND, update it when security issues arise (as they are guarenteed to do), and log the BIND audit data, you will be well on your way to prevent compromise of your server.

1. Version - Run the latest stable version of BIND available at ISC (http://isc.org). If you are operating a BIND 4 or BIND 8 server the time to upgrade is NOW! BIND 9 is a complete rewrite which is a major improvement over the old deprecated BIND 8 server both in security and functionality. Ideally, at least BIND 9.3 should be used, as that includes full support for DNSSEC.

2. Stay Current - Keep up to date on breaking security issues relating to BIND via the ISC security site http://www.isc.org/sw/bind-security.php and the Google bind newsgroup http://groups.google.com/comp.protocols.dns.bind as well as security announcements at CERT (http://cert.org)

3. Syslog

Event data from BIND 9 should be sent to a syslog server on a secure log server. This is handled by the "logging" command in /etc/named.conf. There are two parameters, the syslog facility and the debug level or severity. Use the syslog facility to keep DNS logs separate from other logging streams. Debug is a good severity level. If that produces too much audit data, then you can drop down to "info".

```
logging {
        channel default_log {
                syslog local3;
                severity debug;
                };
        category default { default_log; };
 };
```

Unfortunately, the are a lot of misconfigured DNS servers in the world. Many of them are

"lame." These happen when a DNS server delegates a sub-domain to another DNS server and that server is not authoritative for the sub-domain. Alas, these are logged by default. If this is a bother, then you can turn off the logging of lame servers by adding

```
category lame-servers {null;}
```

to the logging section.

Note that beside the default logging categories, there are a slew of other categories. For example, you can log all queries processed by BIND, or all DNSSEC transactions. This mechanism can also be used to seperate the different type of log events into different files.

BIND 9 Security – Limiting the damage

Even if you do everything perfectly, BIND 9 is still a network service and hence has a chance of being compromised. There are a few steps you can take to limit the spread of the damage if it does happen.

1. Run BIND 9 as non-root user

First, you create a "named" user and "named" group. The choice of "named" is arbitrary but traditional. Make sure the “named” user can read the zone files (usually in /var/named). Then change your start-up script to invoke BIND 9 as

```
named -u named -g named
```

Note that if you are using DNS dynamic updates, the zone files will have to be writeable by the named user as well. Also, if you are syslogging to a file, that will have to be writeable by named.

2. Chroot BIND

BIND 9 can be chrooted into it's own little world. This can be slightly tricky, but it's not hard if you take it a step at a time. It's certainly much easier than it was in BIND 8. The main issues are

- making sure you put everything into the chroot area that BIND needs to live
- if your version of syslog does not support the "-a" argument then you can only syslog to a file inside the chrooted environment.People have developed tricks to work around the syslog issue over the years, but they are somewhat system specific

So, the process is

Make a directory structure for chrooting

```
mkdir /chroot
mkdir /chroot/dev
mkdir /chroot/etc
mkdir -p /chroot/usr/bin
mkdir -p /chroot/var/named
mkdir -p /chroot/var/run
```

Copy the BIND configuration file to the chroot

```
cp /etc/named.conf /chroot/etc/named.conf
```

Check out the device numbers of /dev/null on your system and create one

```
ls -l /dev/null
mknod /chroot/dev/null 13 2
```

Copy your zone and other data files from /var/named to the chroot /var/named directory

Start BIND with the "-t" option to point to the chroot area

```
named -t /chroot
```

There are reports that it is sometimes necessary to also move the system's zoneinfo file into the chroot area to avoid a GMT time-stamp on syslog entries. If you are maintaining all your other syslog entries in GMT then this is not an issue.

NOTE - A couple more things may need to be done here, like creating /dev/random, on some operating systems.

BIND 9 Security – Cryptographic solutions

There is no question that adding cryptography to the DNS structure adds complication and life cycle costs. These should not be underestimated. Fundamentally, though, it is the only way to really solve some of the basic security problems of DNS. Of the two solutions, TSIG and DNSSEC, the latter is much more powerful, but it requires a supporting infrastructure of signed root servers and a coordinated process for distributing and maintaing keys. TSIG is more useful when that infrastructure does not exist and you have control of all the BIND servers in an enclave.

1. TSIG

Before DNSSEC, a simple symmetric key system was designed into DNS (RFC 2845) to provide security in a pairwise manner between two servers for zone transfers. In certain enclave situations it can still be useful.

Start by generating a public/private pair of HMAC-MD5 keys using the dnssec-keygen program that comes with BIND. While key lengths of 128 and 256 are often used for TSIG, on principle you might as well use the maximum of 512.

```
dnssec-keygen -a HMAC-MD5 -b 512 -n HOST foo.bindkey
```

This generates 2 files

Kfoo.bindkey.+157+34764.private
Kfoo.bindkey.+157+34764.key

Add the information from those files to /etc/named.conf on both servers.

```
key tsigkey {
     algorithm "HMAC-MD5";
     secret "<insert the secret key from the private file>";
};
```

The standard way to do this is to create a file readable by root only in /var/named, "keyfile", for example, and then include that in named.conf by reference, i.e.

```
include "/var/named/keyfile";
```

Then on each server, add an entry telling BIND to use that key with the other server, i.e.

```
server 192.168.12.4 {
     keys {
          "tsigkey";
     };
};
```

on host 192.168.12.5 and

```
server 192.168.12.5 {
     keys {
          "tsigkey";
     };
};
```

on host 192.168.12.4.

To restrict zone transfers you change your zone entry to

```
zone "foo.com" {
     type master;
```

```
     file "foo.com";
     allow-transfer  {key tsigkey};
};
```

on the master server.

TSIG can also be used to secure dynamic updates. The BIND bind update-policy construct is used in complicated cases like granting partial access to a zone, but most bind sites will do fine with just allow-update, i.e.

```
zone "foo.com" {
     type master;
     file "foo.com";
     allow-update {
          key tsigkey;
     };
};
```

2. DNSSEC

The topic of implementing a DNSSEC infrastructure is very complicated and out of the scope of this paper. For background, you can read the NLnet Labs DNSSEC HOWTO at http://www.nlnetlabs.nl/dnssec_howto/dnssec_howto.pdf but we will just sketch out the DNSSEC elements that impact the BIND server.

First, some versions of BIND do not have DNSSEC enabled by default, so it's best to add that to your options. Releases of BIND 9 beginning with version 9.3 support DNSSEC.

```
options {
     dnssec-enable yes;
     dnssec-validation yes;
};
```

Then, a private key Zone Signing Key (ZSK) and Key Signing Key(KSK) are generated with dnssec-keygen for each zone, i.e.

```
     dnssec-keygen -a RSASHA1 -b 2048 -e -n ZONE foo.com

     dnssec-keygen -f KSK -a RSASHA1 -b 2048 -n ZONE foo.com
```

This will generate the keys
Kfoo.com.+005+43161
Kfoo.com.+005+24022

The algorithm and key length will be determined by the overall DNSSEC infrastructure plan. Another possible alternative is HMAC-SHA256 at 256 bits.

The two keys thus generated are included into the zone file in /var/named with

```
$include Kfoo.com.+005+43161.key ; ZSK
$include Kfoo.com.+005+24022.key ; KSK
```

Finally, the actual zones are signed with dnssec-signzone.

```
dnssec-signzone -o foo.com -k Kfoo.com.+005+24022.key \
     -N 1 db.foo.com Kfoo.com.+005+43161.key
```

This outputs into db.foo.com.signed and so you need to change the pointer in /etc/named.conf

```
zone foo.com {
     file “db.com.signed”;
};
```

After this, either restart BIND or run the commands:

```
rndc reconfig
rndc flush
```

Note that the zone must be resigned at least once a month.

BIND 9 contains an alternative mechanism for doing DNSSEC called DNSSEC Look-aside Validation (DLV) which is not discussed here. It is of limited importance now that the Internet's root zone signing is in progress.

References

Cricket Liu and Paul Albitz, DNS and Bind: Fifth Edition, O'Reilly, May 2006, ISBN 0596100574

Ron Aitchison, Pro DNS and BIND, Apress, 2005, ISBN 1590594940

Jeremy C. Reed, BIND 9 DNS Administration Reference Book, Reed Media Services, 2007, ISBN 0979034213

Ramaswamy Chandramouli and Scott Rose, Secure Domain Name System (DNS) Deployment, NIST Special publication SP800-81, May 2006
http://csrc.nist.gov/publications/nistpubs/800-81/SP800-81.pdf

Ramaswamy Chandramouli and Scott Rose, Open Issues in Secure DNS Deployment, IEEE Security and Privacy, vol. 7, no. 5, pp. 29-35, September/October 2009

Eric Osterweil and Lixia Zhang, Interadministrative Challenges in Managing DNSKEYs, IEEE Security and Privacy, vol. 7, no. 5, pp. 44-51, September/October 2009

Relevant DNS RFCs

RFC 1034 - P. Mockapetris, Domain Names - Concepts and Facilities,
IETF RFC 1034, November 1987
http://www.ietf.org/rfc/rfc1034.txt

RFC 1035 - P. Mockapetris, Domain Names - Implementation and Specification,
IETF RFC 1035, November 1987
http://www.ietf.org/rfc/rfc1035.txt

RFC 2845 - P Vixie, et. al. Secret Key Translation Authentication for DNS (TSIG),
IETF RFC 2845, May 2000
http://www.ietf.org/rfc/rfc2845.txt

RFC 3833 - D. Atkins and R. Austein, Threat Analysis of the Domain Name System (DNS)
IETF RFC 3833, August 2004
http://www.ietf.org/rfc/rfc3833.txt

RFC 4033 - R Arends et al., DNS Security Introduction and Requirements
IETF RFC 4033, March 2005
http://www.ietf.org/rfc/rfc4033.txt

RFC 4034 - R Arends et al., Records for DNS Security Extensions
IETF RFC 4034, March 2005
http://www.ietf.org/rfc/rfc4034.txt

RFC 4035 - R Arends et al., Protocol Modifications for the DNS Security Extensions
IETF RFC 4035, March 2005
http://www.ietf.org/rfc/rfc4035.txt