Global Information Assurance Certification Paper

Defense in Depth for DNS

Cheng C. Teoh

GSEC
Version 1.4b
Option 1

Abstract

The Domain Name System (DNS) is one of the key components of the Internet and most IP networks, for that matter. Despite its importance, not many people have even heard of DNS, much less know what it is and how to keep it secure. For the sake of the uninitiated, DNS is analogous to a telephone directory for the Internet. DNS translates the server names, that humans are more likely to remember, to IP addresses, which computers use to navigate through the Internet. For instance, it is easy for most people to remember that Example Inc.'s web server is at www.example.com. However, in order for your browser to locate the server, it has to ask a DNS server for the IP address of www.example.com. DNS is the application that saves you the nightmare of remembering all those IP addresses and instead use friendly names like www.example.com.

This paper will focus on security for the most widely used DNS server on the Internet, namely the Berkeley Internet Name Domain (BIND). Some of the perimeter defense techniques may be applicable to other implementations of DNS. In this paper we will be taking a layered approach to DNS security. First, we will investigate configurations on the DNS server itself to secure it. Next, will be a section on perimeter defenses to protect your DNS server, and lastly, is a section on backup considerations for DNS. With the application of proper precautions to your DNS servers, you can mitigate many of the risks and vulnerabilities associated with the service. As always, a prudent DNS administrator should also keep up with the latest news about bugs or vulnerabilities to stay ahead of the malicious hacker.

Secure DNS Configuration

In this section we will investigate the steps involved in configuring BIND to prevent most of the basic vulnerabilities. These guidelines will form the first layer of defense for your DNS servers.

Patches And Latest Builds

It is important to stay up to date with the latest releases of BIND where possible. Like any other piece of software, name server software evolves with each new release. Virtually all older name servers have widely-known vulnerabilities that can be exploited.[1] By running the latest release of BIND, you minimize the possibility of being susceptible to an exploit. BIND is currently maintained by the Internet Software Consortium (ISC) and so their website always has the current version of BIND. To download the code for the latest version, follow this link to the ISC's website: http://www.isc.org/products/BIND/

As of the writing of this document, the current releases are BIND Version 9.2.1 (released May 1, 2002), BIND Version 8.3.4 (released November 16, 2002) and BIND 4.9.11 (officially deprecated). The ISC website publishes the release notes for each version of BIND so that users can be informed of the new features and fixes. The release notes also include any configuration file changes that may be needed in order to take advantage of new features.

Split Internal And External DNS

It is good practice to run separate internal and external DNS servers. Typically, an organization's internal network is populated with hosts that hold sensitive data or systems. By having separate DNS servers for the internal network, you can keep the IP addresses and names of sensitive internal hosts invisible from the Internet.

Another advantage of this approach is that the external DNS server(s) can be dedicated "advertising only" servers that respond to queries about Internet-accessible services for the organization. This means the DNS administrator will have finer-grained control over how to configure BIND on each server. Recursion can be disabled on external DNS servers to prevent some exploits that will be discussed in the next section. Recursion will also be explained in that section.

"BIND 9 introduced a new feature called "views" that makes delivering different versions of a zone, and even different name server configurations, easy."[2] Using views, a single DNS server can return different responses to the same DNS query, depending on the source address of the query. Here's an example of the *named.conf* file of a BIND 9 server:

```
acl "internal" {
     192.168.1.0/24;
     192.168.2.0/24;
};
view "internal" {
     match-clients { "internal"; };
     recursion yes;
};
view "external" {
     match-clients { any; };
     recursion no;
};
```

In this example the BIND 9 server will respond to recursive queries coming from IP addresses defined in the Access Control List (192.168.1.0/24 and 192.168.2.0/24). Recursive queries will not be answered for clients outside of the subnets defined in the ACL. In effect, "views" allows you to easily run two DNS servers in one simple configuration.

Disable Recursion

In the last section we looked at restricting recursive queries. Recursion is the process in which a name server will "walk" the DNS tree to return the IP address of a host to the client that sent the query. To respond to a recursive query for www.example.com, the DNS server will first go to one of the root servers. The root server will refer the recursive DNS server to the .com name servers. Next, the recursive DNS server will ask the .com server about www.example.com and

will receive another referral to the example.com name server(s). Finally the example.com name server will answer with the IP address of www.example.com.

There are two main reasons why recursion should be disabled on external DNS servers:

1. Recursive queries take a longer time to resolve. Therefore, the DNS server's resources are tied up for a longer period of time. This can be a basis of DoS attacks that cripple a DNS server by a hacker who scripts a large number of recursive queries against the DNS server.
2. A malicious hacker can use recursive queries to perform DNS spoofing and poison the cache on the target DNS server. This may cause sensitive email to be misdirected to the attacker's server, or web browsers to be redirected to the wrong web sites. The SANS Reading Room has an excellent paper by Doug Sax regarding this attack.[3]

Single-Purpose DNS Server

Another good practice is to run DNS servers that are dedicated to a single purpose. By limiting the number of services running on your DNS server, you are also limiting the possible ways a hacker may attack it. Disable daemons that are not required on your DNS server, especially daemons that have well-known vulnerabilities. Make your DNS server a *dedicated* DNS server.

Most organizations have guidelines in place to harden any server connected to the network. Follow these guidelines if available. Otherwise, there is a wealth of information in printed and electronic form that will help you to harden your DNS server. Here are some online resources available:

- Lance Spitzner's Security Whitepapers
  http://www.spitzner.net/
- The Process of Hardening Linux by Chris Koutras
  http://www.sans.org/rr/linux/hardening.php
- Securing HP-UX 11 by Larry Harker
  http://www.sans.org/rr/unix/HP-UX11.php
- CERT's UNIX Configuration Guidelines
  http://www.cert.org/tech_tips/unix_configuration_guidelines.html

Diverse Locations For DNS Servers

"Microsoft's web service outage past January 2001, that lasted for over 24 hours, is a serious reminder of the potential consequences of incorrectly configured DNS (Domain Name System). The outage, caused both by the network failure and the subsequent hacker attack, could have been prevented if their servers had not been located on the same subnet."[4]

This Denial of Service (DoS) attack was possible because all of Microsoft's DNS servers were located in the same physical location and on a network behind a single router. The initial network failure revealed a single point of failure. The subsequent attacker was able to saturate the network that the DNS servers were located on and therefore cause an outage to many of Microsoft's Internet services.

One way to mitigate this problem is to diversify the network paths to your DNS servers. A good design locates DNS servers on different network segments so that failure or a DoS attack on any one segment would not render your DNS servers completely inaccessible. To take the concept another step further, consider hosting secondary DNS servers at a third party service provider.

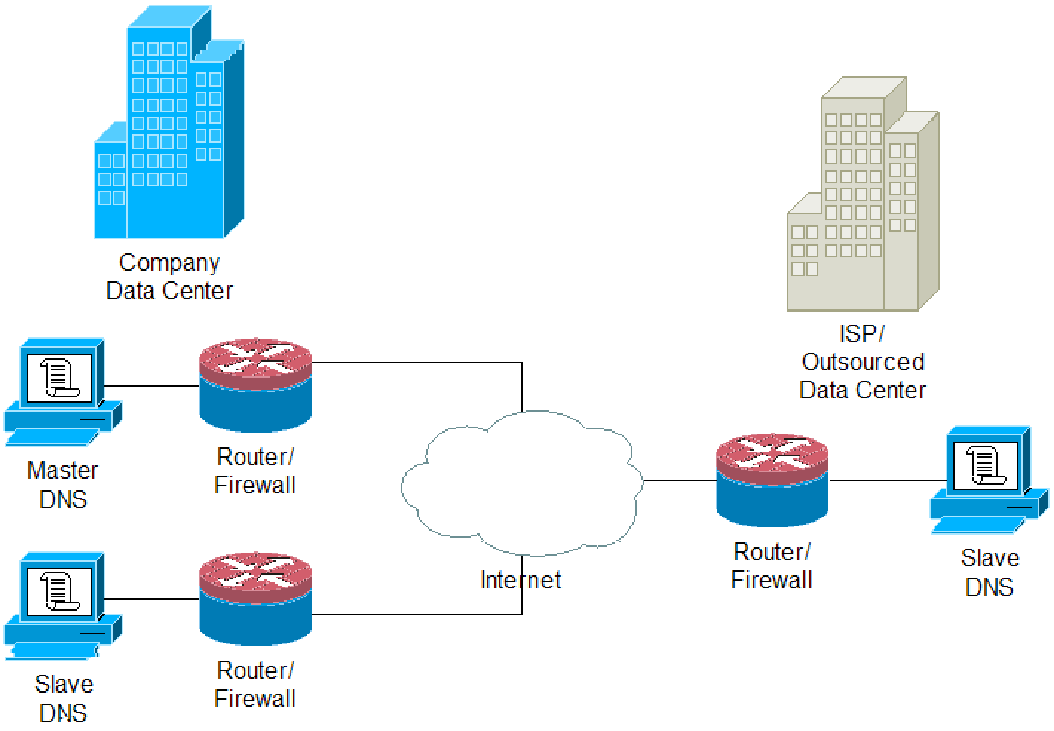

Figure 1: Diverse locations for DNS servers

Restrict Zone Transfers

Name servers use zone transfers to download data about an entire zone or domain name. This is a necessary feature that is implemented so that a slave or secondary name server may easily download data about a zone from its master or primary name server. By default, BIND allows zone transfers from any host on the Internet. This is a problem for two reasons:

1. DNS servers are vulnerable to Denial of Service attacks. The attacker can create a script that will repeatedly perform zone transfers from the target server, causing server resources and bandwidth to be tied up with the

transfers. This becomes an even bigger problem if the DNS server is authoritative for large zones.

2. By performing a zone transfer, an attacker can quickly and easily obtain a "map" of the target network. The zone data holds the name and IP address of every host on the network. The attacker can then use this information to quickly identify critical servers such as mail, name, FTP and web servers to be targets.

All current versions of BIND can be configured to only allow zone transfers to an explicit list of hosts. This list should include all slave name servers and any trusted host, (e.g., the DNS administrator's workstation). The configurations for restricting zone transfers are as follows:

With BIND 4.9.x, use the *xfrnets* directive:

```
xfrnets 192.168.1.10&255.255.255.255
```

This is a global setting for all zones hosted on this BIND 4.9.x name server.

With BIND 8.x.x, use the *allow-transfer* substatement:

```
options {
     allow-transfer { 192.168.1.10; };
};
```

This applies the zone transfer restriction to all zones hosted on the server. BIND 8.x.x allows for more flexible controls and will support the same substatement at the zone level:

```
zone "example.com" {
     type master;
     file "db.example.com";
     allow-transfer { 192.168.1.10; };
};
```[5]

Authenticate Zone Transfers

As mentioned in a previous section, slave DNS servers rely on zone transfers from the master DNS server in order to get its zone data. This is an extremely convenient feature for the DNS administrator because data only has to be entered once on the master DNS. All slave servers will then update their zone files based on the master. However, this brings up a problem with ensuring data integrity during the transfer. How does the slave server verify the authenticity of the zone data?

BIND 8.2 and newer revisions implement a feature for securing DNS messages called *transaction signatures* (TSIG). TSIG can be used to authenticate and verify zone data through the use of cryptography based on a pre-shared key. An identical pre-shared key has to be configured on both the master and slave servers for this to work. Here is an example:

Master Name Server:

```
key example-key. {
     algorithm hmac-md5;
     secret "eIEio";
};
server 192.168.1.10 {
     keys { example-key.; };
};
zone "example.com" {
     type master;
     file "db.example.com";
     allow-transfer { 192.168.1.10; };
};
```

Slave Name Server:

```
key example-key. {
     algorithm hmac-md5;
     secret "eIEio";
};
server 192.168.1.9 {
     keys { example-key.; };
};
zone "example.com" {
     type slave;
     file "db.example.com";
};
```

Restrict Dynamic Updates

BIND 8 and BIND 9 support the Dynamic Update standard described in RFC 2136. This allows authorized agents to update zone data by sending special update messages to add or delete resource records.[6] DNS updates are necessary in some environments. For example, Microsoft Active Directory servers require dynamic updates to DNS in order to add their SRV records. SRV records are special DNS entries that help clients to locate a particular server or service without knowing the specific server name. For example in an Active Directory environment, a Domain Controller will dynamically update DNS with its SRV records within the _msdcs.example.com subdomain. This enables a client to easily locate the Domain Controller to log into.

Support for dynamic updates is a double-edged sword because it also opens up a vulnerability on improperly configured DNS servers. Fortunately, there is a feature to limit exposure by restricting updates to known IP addresses or subnets.

The key to successful implementation of dynamic updates is to restrict updates only from trusted IP addresses or subnets. For instance, if all your Active

Directory servers are located on their own subnet, set up BIND to only accept updates from that subnet. Here is an example of a configuration that only allows subnet 192.168.10.0/24 to perform dynamic updates:

```
zone "windows.example.com" {
     type master;
     file "db.windows.example.com";
     allow-update { 192.168.10.0/24; };
};
```

Note that by default BIND 8 disables dynamic updates. However, some DNS administrators use the "catch all" approach to opening up dynamic updates to zones by using the directive `allow-update { any; }`. This is a risky practice that opens up the zone to malicious erroneous updates.

Hide the BIND Version

By default BIND will respond to queries regarding its version. This information is stored as a TXT record under the pseudo domain name *version.bind* in the CHAOSNET class.[7] While this is a useful feature for a DNS administrator to discover the version of BIND while troubleshooting, any other DNS client really has no business knowing the BIND version you are running. On the other hand, a hacker could find out the server version and use a known exploit against it. The following configuration in the options block of *named.conf* will change the response for a query for the version of BIND.

```
options {
     directory "/var/named";
     version "Unknown";
};
```

As a result of this configuration, BIND will respond "Unknown" to a version query.

Restrict Access

External DNS servers should only answer queries from clients with Internet-routable public IP addresses. Because of this, external DNS servers should never respond to queries from IP addresses in the Private Address Space as defined in RFC 1918. Based on this RFC, the Internet Assigned Numbers Authority (IANA) has reserved the following three blocks of IP address space for private internets:

10.0.0.0	-	10.255.255.255	(10/8 prefix)
172.16.0.0	-	172.31.255.255	(172.16/12 prefix)
192.168.0.0	-	192.168.255.255	(192.168/16 prefix)[8]

Since these addresses are not routable on the Internet, DNS answers will just be dropped by Internet routers. There is a good possibility that these queries come malicious sources that intend to mount a DoS attack on your name server. The following configuration will cause BIND to just ignore or "blackhole" the query.

```
acl "bogus-nets" {
     10.0.0.0/8
     172.16.0.0/12
     192.168.0.0/16;
};
options {
     blackhole { "bogus-nets"; };
};
```

Perimeter Security for DNS

In this section we will investigate some measures that can be taken on perimeter devices to protect our DNS server.

Firewall Protection

The IANA has assigned TCP and UDP ports 53 to DNS queries.[9] If you are running a single-purpose DNS server, then firewall rules should be written to only allow traffic to port 53 bound for the DNS server.

Since TCP connections are only established for zone transfers, the firewall should only allow TCP port 53 from authorized slave name servers. This list should match the list of servers in the *allow-transfer* directive for the DNS server. By doing this, we are adding another layer of security that will restrict zone transfers only to authorized servers.

Network Protection

Edge Internet routers can be configured to add another level of protection to your DNS servers. As discussed before, there is absolutely no reason that an external DNS server should receive queries from IANA Private Reserved source IP addresses as defined in RFC 1918. For that matter, private addresses should never traverse an Internet router. Therefore, it is good practice to set up access lists on Internet routers to block all traffic from RFC 1918 address space.

Backup Considerations

Even with all the precautions described in the previous sections, your DNS server may not be 100% hackproof. It is imperative to have backup systems ready to restore your DNS servers into service should the unthinkable happen. The following are some ways to plan for recovering from an attack in the shortest amount of time possible.

File Backup

At a minimum, backup all DNS server configuration files. This prepares you for incidents where a hacker may have broken through your defenses and corrupted the data on your DNS server. Once you have ensured that the hacker is locked out, you can quickly restore DNS service if you have all your data backed up in a safe location. Not having a good backup means that you will have to recreate all

the configuration files. This is not an easy task, especially if you have large or complex configurations for BIND.

DNS Management Software

Today there are several software products available that can be used to manage your DNS configurations. The main purpose of these systems is to take the drudgery out of manually configuring BIND. An added benefit of using DNS Management Software is that there is always a current backup of your configuration data on the management server. As soon as you have restored a hacked DNS server, you can quickly and easily generate the configuration files needed for BIND to function properly.

Hardware Spares

It is a good idea to have hardware spares of your DNS server(s). If a hacker has taken advantage of a newly-discovered exploit to break into your DNS server, you can isolate the problem efficiently by using a spare server. You can fix the vulnerability on the spare offline, and when ready it can be swapped into service.

It is also a fact that no piece of hardware is infallible. There will come a time when your DNS server hardware physically fails. While this may not specifically be a security issue, it is a factor that will affect the availability of your name server. Having a spare piece of hardware ready to be deployed is a great insurance policy against hardware failures.

Conclusion

DNS is a critical service for the proper functioning of the Internet. It is imperative that all precautions be taken in order to mitigate security risks of your DNS servers. The steps described in this paper should point the savvy DNS administrator in the right direction to protecting this critical service.

As with any other piece of technology, things are constantly changing. Always keep up to date with the latest developments and bug reports. By doing so, you can stay one step ahead of the hacker. The following are some good sources of information:

- ISC BIND Vulnerabilities
  http://www.isc.org/products/BIND/bind-security.html
- SANS Institute Reading Room – DNS Issues
  http://www.sans.org/rr/DNS/
- Men & Mice DNS Surveys
  http://www.menandmice.com/6000/6000_domain_health.html
- SecurityFocus (select Vendor: ISC)
  http://online.securityfocus.com/cgi-bin/sfonline/vulns.pl

References

[1] Householder, Allen; King, Brian. “Securing an Internet Name Server.” URL: http://www.cert.org/archive/pdf/dns.pdf (August 2002)

[2] Liu, Cricket. “Views in BIND 9.” URL: http://sysadmin.oreilly.com/news/views_0501.html

[3] Sax, Doug. “DNS Spoofing (Malicious Cache Poisoning)” URL: http://www.sans.org/rr/firewall/DNS_spoof.php November 12, 2000.

[4] Men & Mice. “Men & Mice ‘Single Point of Failure’ Research” URL: http://www.menandmice.com/6000/6300_single_point_failure.html

[5] Liu, Cricket. “Securing an Internet Name Server.” URL: http://www.acmebw.com/resources/papers/securing.pdf

[6] Albitz, Paul; Liu, Cricket. DNS and BIND 4th Edition. Sebastopol: O’Reilly & Associates, Inc., April 2001. 39

[7] Liu, Cricket. “Cooking with DNS & BIND.” URL: http://www.oreillynet.com/pub/a/network/excerpt/dnsbindcook_ch07/index.html

[8] Rekhter, Yakov; Moskowitz, Robert; Karrenberg, Daniel; de Groot, Geert Jan; Lear, Eliot. “RFC 1918: Address Allocation for Private Internets.” URL: http://www.ietf.org/rfc/rfc1918.txt?number=1918

[9] Internet Assigned Numbers Authority. “Port Numbers” URL: http://www.iana.org/assignments/port-numbers