The soft chewy center of the Internet

DNS ATTACKS

Are your systems secure against DNS attacks? We'll show you why they matter and help you determine whether you are vulnerable. **BY KURT SEIFRIED**

Like most of the original protocols on which the Internet is based, the original design decisions that led to their popularity and success are now coming back to haunt us with security problems.

Keep in mind that when the Internet was originally created, it was a relatively small, well-connected community. Security was not particularly high on the list of concerns – just getting it to work and do useful things was amazing enough.

Why DNS Attacks Matter

As I'm sure you know, DNS provides one of the fundamental infrastructure services on the Internet – specifically, the translation of human-readable names such as *www.linux-magazine.com* to an IP address such as 80.237.227.148.

This service is important because it allows a static name to be registered, but the underlying service(s) can be at arbitrary locations and can be created or moved easily. For example, I outsourced my email for *seifried.org* to Google's Gmail.

DNS Dependent

Thus, you rely on DNS almost every time you use another protocol or service, including email, the web, instant messaging clients, VOIP, etc. If attackers could initiate hostile actions, such as redirecting *www.your-bank.com* to their server, they would be able to execute any number of attacks, such as spoofed web sites, reading your incoming and outgoing email, and so forth.

Why DNS Is Fast (and Insecure)

One of the best decisions was to make DNS an extremely lightweight and fast protocol. The majority of requests and replies use the UDP protocol, which is stateless and similar to sending an SMS text message. (Larger replies might result in a TCP-based session.)

So, you are limited in how much data you can send, and you won't know whether the remote end receives it or replies; you're just left waiting for a reply.

A UDP packet is about as simple as it gets – you have the basic address information (source and destination IP address and ports) and packet information (packet type, length, checksum, data).

UDP has no significant security mechanism to ensure that the packet came from the machine it claims to be from or that it is part of a legitimate transaction, which is good for speed.

If you fire off a query, you just hope a reply gets sent back, allowing DNS servers to handle high volumes of requests. In fact, in 2007, it was on the order of 4 billion requests per day for root-level servers.

Spoofing a UDP Packet

When spoofing a UDP packet, you need to know the IP addresses and ports in use, which is trivial with a DNS query because the IP addresses are known (the server making the query and the server answering it) and, because it is a DNS request, the destination port is always 53. This leaves only the source port to determine, and because many operating systems simply use a static port for outgoing connections or ports incremented by one for each outgoing request, it's relatively easy for an attacker to guess.

Transaction ID

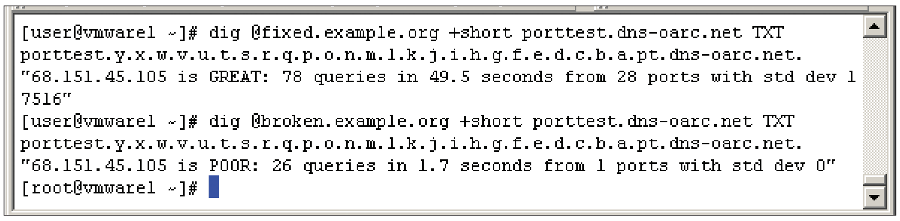
In an attempt to address the packet-spoofing issue within the DNS protocol, a transaction ID was added. A simple 16-bit number – with 65,536 possibilities – that is sent in the request and that must be copied into the answering packet theoretically prevents an attacker from blindly spoofing the replies because it must now guess the transaction ID as well.

Unfortunately, creating really good random values is surprisingly tricky, and some implementations of Bind simply use transaction IDs that increment by 1 for each request, making them completely predictable. Now you're back to the place where an attacker can easily spoof a packet and insert hostile data into a DNS server.

How the Attack Works

So how do attackers exploit this issue? The first thing they do is find a vulnerable server and a domain that they want to control (e.g., *www.your-bank.com*). Then they find a machine that is allowed to use the vulnerable server for DNS lookups.

Large ISPs – such as mine, which has two DNS servers for the city – are likely targets because compromising them gives the attackers access to thousands of clients, so compromising a single machine to launch the attack does not present a significant hurdle.



```
[user@vmware1 ~]# dig @fixed.example.org +short porttest.dns-oarc.net TXT
porttest.y.x.w.v.u.t.s.r.q.p.o.n.m.l.k.j.i.h.g.f.e.d.c.b.a.pt.dns-oarc.net.
"68.151.45.105 is GREAT: 78 queries in 49.5 seconds from 28 ports with std dev 1
7516"
[user@vmware1 ~]# dig @broken.example.org +short porttest.dns-oarc.net TXT
porttest.y.x.w.v.u.t.s.r.q.p.o.n.m.l.k.j.i.h.g.f.e.d.c.b.a.pt.dns-oarc.net.
"68.151.45.105 is POOR: 26 queries in 1.7 seconds from 1 ports with std dev 0"
[root@vmware1 ~]#
```

Figure 1: DNS test screenshot.

High-Volume Service

Alternatively, the attacker can use JavaScript to create a web page that triggers this attack, then the attacker can trigger a DNS lookup for *www.your-bank.com* and try to spoof packets with hostile data to the ISP's DNS server.

One more reason that this attack is so likely to succeed is that DNS is a high-volume service, with few sites logging incoming requests and answers, so detection of an attack is extremely unlikely. Attackers can simply hammer away at the server, making thousands of requests and spoofing replies until they succeed.

Are You Vulnerable?

Web-based and command-line tests check for this vulnerability. They generally trigger a number of DNS lookups that are examined, checking the port numbers and transaction IDs for randomness, and you can see the results quickly. Two web-based tests are available online [1][2].

Additionally, the DNS-OARC center offers a command line--based check that can be accessed with a tool such as dig or nslookup:

```
$ dig @ip.or.hostname ↩
+short porttest.dns-oarc.net TXT
```

To fix your vulnerability, you must update your DNS server; almost every vendor released an update in July. After you have updated your DNS server, and assuming you are using Bind, be sure that it is configured properly.

To do so, check your *named.conf* file and make sure you do not have something such as

```
query-source port 53;
query-source-v6 port 53;
```

in it, but instead, something like:

```
query-source port *;
query-source-v6 port *;
```

After you have updated, you should use one of the web-based or command-line tests to ensure it is working as expected.

Conclusions

DNS attacks illustrate both the limitations of some of the protocols in use on the Internet and the robustness inherent in the system, and it is unlikely this kind of attack will ever go away.

Even with the publicity surrounding this issue, a significant portion – upwards of 50 percent, according to some reports – of DNS servers still have not been fixed. Like spam, this kind of attack is something you will have to learn to live with. ■

INFO

[1] DoxPara: *http://www.doxpara.com/*

[2] DNS-OARC: *http://www.dns-oarc.net/*

THE AUTHOR

Kurt Seifried is an Information Security Consultant specializing in Linux and networks since 1996. He is married and has four cats but no fish (because the cats are more hungry than afraid of water). He often wonders how it is that technology works on a large scale but often fails on a small scale.