THE ULTIMATE GUIDE TO
Preventing DNS-based DDoS attacks

DNS IS CLEARLY A CRITICAL COMPONENT OF COMPUTER NETWORKING, BUT THERE ARE TIMES WHEN THESE TOOLS CAN BE USED FOR MALFEASANCE. HERE’S HOW TO REDUCE THOSE INCIDENTS.
The ultimate guide to preventing DNS-based DDoS attacks

DNS is clearly a critical component of computer networking, but there are times when these tools can be used for malfeasance. Here’s how to reduce those incidents.

BY CRICKET LIU
infrastructure is remarkably simple: The attackers send queries to name servers across the Internet, and those name servers return responses. Instead of sending the queries from their own IP addresses, though, the attackers spoof the address of their target -- which could be a Web server, a router, another name server, or just about any node on the Internet.

Spoofing DNS queries is particularly easy because they are usually carried over UDP (the connectionless User Datagram Protocol). Sending a DNS query from an arbitrary IP address is about as simple and has roughly the same effect as writing someone else's return address on a postcard.

Spoofing queries isn’t enough to incapacitate a target, though. If the responses to those queries were no larger than the queries themselves, an attacker would do just as well to flood the target with spoofed queries. No, to maximize the damage to the target, each query should return a very large response. It turns out that’s very easy to instigate.

Since the advent of EDNS0, a set of extensions to DNS introduced back in 1999, UDP-based DNS messages have been able to carry lots of data. A response can be as large as 4,096 bytes. Most queries, on the other hand, are fewer than 100 bytes in length.

Once upon a time, it was relatively difficult to find a response that large in the Internet's namespace. But now that organizations have begun deploying DNSSEC, the DNS Security Extensions, it’s much easier. DNSSEC stores cryptographic keys and digital signatures in records in the namespace. These are positively enormous.

You can see an example of a response from the isc.org zone that contains DNSSEC records on my blog. The size of the response is 4,077 bytes, compared with a query of just 44 bytes.

Now, picture attackers from around the Internet sending that spoofed query from your Web server’s IP address to the isc.org name servers. For each 44-byte query, your Web server receives a 4,077-byte response, for an amplification factor of almost 93 times.

Let’s do a quick calculation to figure out how bad this could get. Say each attacker has a relatively modest 1Mbps connection to the Internet. He can send about 2,840 44-byte queries across that link per second. This query stream would result in almost 93Mbps worth of replies reaching your Web server. Every 11 attackers represent 1Gbps.

Where would antisocial attackers find 10 friends to help them carry out an attack? Actually, they don’t need any. They’ll use a botnet of thousands of computers.

The ultimate effect is devastating. In their quarterly global DDoS Attack Report, Prolexic (a DDoS-mitigation company) reported a recent DNS-based attack against a customer that topped 167Gbps. Prolexic further reported that average DDoS attack bandwidth was up 718 percent to 48Gbps in a single quarter.

But wait! Couldn’t the isc.org name servers be modified to recognize that they’re being queried over and over for the same data, from the same IP address? Couldn’t they squelch the attack?

They certainly can. But the isc.org name servers aren’t the only ones an attacker can use to amplify his traffic. Sure, there are other authoritative name servers the attacker could use, but even worse are open recursive name servers.

An open recursive name server is simply a name server that will process recursive queries from any IP address. I can send it that query for isc.org data and it will reply to me, and you can do the same.
There shouldn’t be many open recursive name servers on the Internet. A recursive name server’s function is to look up data in the Internet’s namespace on behalf of DNS clients, like the ones on your laptop or smartphone. The network administrators who set up recursive name servers (such as your IT department) usually intend them for use by a particular community (for example, you and your fellow employees). Unless they’re running services such as OpenDNS or Google Public DNS, they don’t mean to have them used by the citizens of Moldova. So public-spirited, security-minded, and most especially competent administrators configure access controls on their recursive name servers to limit their use to authorized systems.

Given that, how big a problem could open recursive name servers be? Pretty big. The Open Resolver Project has collected a list of 33 million open recursive name servers. Hackers can fire spoofed queries at as many of these as they like to spew isc.org data at your Web server, name server, or border router until it chokes.

That’s how DNS-based DDoS attacks work. Thankfully, we have a few ways to combat them.

**How to weather the storm**

The first order of business is instrumenting your DNS infrastructure, so you’ll know when you’re under attack. Too many organizations have no idea what their query load is, so they’d never know if they were being attacked in the first place.

Determining your query load can be as simple as using BIND’s built-in statistics support. The BIND name server will dump data to its statistics file when you run `rndc stats`, for example, or at a configurable statistics interval. You can examine the statistics for query rate, socket errors, and other indications of an attack. Don’t worry if you’re not sure what an attack will look like yet -- part of the goal of monitoring DNS is to establish a baseline, so you can identify what’s abnormal.

Next, take a look at your Internet-facing infrastructure. Don’t limit yourself to your external authoritative name servers; examine your switch and router infrastructure, your firewalls, and your connections to the Internet. Identify any single points of failure. Determine whether you can easily (and cost-effectively) eliminate them.

If possible, consider broad geographical distribution of your external authoritative name servers. This helps avoid single points of failure, of course, but it also helps when you’re not under attack. A recursive name server resolving a domain name in one of your zones will try to query the authoritative name server closest to it, so geographical distribution will tend to provide better performance to your customers and correspondents. If your customers are clustered in certain geographies, try to place an authoritative name server near them to provide the quickest responses.

Perhaps the most basic way to combat DoS attacks is to overprovision your infrastructure. The good news is that overprovisioning your name servers isn’t necessarily expensive; a capable name server can handle tens or even hundreds of thousands of queries per second.
Not sure what your name servers’ capacity is? You might use query tools such as dnsperf to test your name servers’ performance -- preferably using a test platform similar to your production name servers in a lab rather than the production servers themselves.

Deciding how much to overprovision your name servers is subjective: What is your online presence worth? Are there other components of your Internet-facing infrastructure that will fail before the name servers? Obviously, it’s foolhardy to spend money to build first-class DNS infrastructure behind a border router or firewall that will fail well before your name servers even break a sweat.

If money is no object, it might be helpful to know that state-of-the-art DDoS attacks against DNS infrastructure can exceed 100Gbps. Using Anycast can also help resist a DDoS attack. Anycast is a technique that allows multiple servers to share a single IP address, and it works particularly well with DNS. In fact, the Internet’s root name servers have used Anycast for years to provide root zone data throughout the globe while still allowing the list of roots to fit into a single UDP-based DNS message.

To deploy Anycast, the hosts supporting your name servers will need to run a dynamic routing protocol, like OSPF or BGP. The routing process will advertise to its neighbor routers a route to a new, virtual IP address on which your name server listens. The routing process also needs to be smart enough to stop advertising that route if the local name server stops responding. You can glue your routing daemon to the health of your name server using code of your own construction -- or you can buy a product that takes care of that for you. Infoblox’s NIOS, not coincidentally, includes Anycast support.

How does Anycast defend against DDoS attacks? Well, say you have six external name servers in two Anycast groups (that is, three sharing one Anycast IP address and three sharing another). Each group contains one member in the United States, one in Europe, and one in Asia. A host mounting a DDoS attack against you can only send traffic to -- and hence only attack -- one member of either group from any point on the Internet at a time. Unless attackers can source enough traffic from North America, Europe, and Asia simultaneously to swamp your infrastructure, they won’t succeed.

Finally, there’s a way you can take advantage of wide geographical distribution and Anycast at the same time, without significant capital outlay: Use a cloud-based DNS provider. Companies such as Dyn and Neustar run Anycast name servers of their own in data centers around the world. You pay them to host your zones and answer queries for your data. And you can continue to maintain direct control over your zone data by asking a provider to configure its name servers as secondaries for your zones, loading the data from a master name server you designate and manage in-house. Just be sure you run the master hidden (that is, with no NS record pointing to it), or you run the risk that an attacker will target it as a single point of failure.

One word of caution when using cloud-based DNS providers: Most bill you at least partly based on the number of queries their name servers receive for data in your zones. In a DDoS attack, those queries might increase dramatically (completely outside of your control and not at all to your benefit), so make sure they have a provision for dealing with DDoS attacks without passing the cost of the traffic on to you.
DDoS attacks

How to avoid becoming an accomplice in DDoS attacks

Now you know how to configure your DNS infrastructure to resist a DDoS attack. It’s nearly as important, though, to ensure that you’re not complicit in a DDoS attack against someone else.

Remember the description of how DNS servers can amplify traffic? Attackers can use both open recursive name servers and authoritative name servers as amplifiers, sending spoofed queries that cause the name servers to send responses more than 100 times as large as the query to arbitrary targets on the Internet. Now, of course you don’t want to be the target of such an attack, but you don’t want to be an accomplice, either. The attack uses your name servers’ resources as well as your bandwidth. If the target takes measures to block traffic from your name server to its network, then after the attack ends, the target may not be able to resolve domain names in your zones.

If you run an open recursive name server, the solution is simple: Don’t. There are very few organizations that have any justification for running a name server open to recursive queries. Google Public DNS and OpenDNS are two that come to mind, but if you’re reading this, I’m guessing you’re probably not them. The rest of us should apply access controls to our recursive name servers to make sure only authorized queriers use them. That probably means limiting DNS queries to IP addresses on our internal networks, which is easy to do on any name server implementation worth its salt. (The Microsoft DNS Server doesn’t support IP address-based access controls on queries. Read what you want to into that.)

But what if you run an authoritative name server? Obviously, you can’t limit the IP addresses from which you’ll accept queries -- or not very much, anyway (you might deny queries from obviously bogus IP addresses, such as RFC 1918 addresses). But you can limit responses.

Two longtime Internet “white hats,” Paul Vixie and Vernon Schryver, realized DDoS attacks that use authoritative name servers for amplification exhibit certain query patterns. In particular, attackers send name servers the same query from the same spoofed IP address (or address block) over and over, seeking maximum amplification. No well-behaved recursive name server would do that. It would have cached the response and not asked again until the time to live of the records in the response elapsed.

Vixie and Schryver came up with a clever mechanism, called Response Rate Limiting (RRL), which allows an authoritative name server to track how often it has sent the same response to the same querier. If that rate exceeds some configurable threshold, the name server will stop sending that response to the querier for a set period. If the querier stops peppering the authoritative name server with the same question, the authoritative name server will stop squelching that response. The upshot is that the authoritative name server will never send any response to a querier at a rate higher than the threshold, which makes it useless in a DDoS attack.

RRL was incorporated into BIND name servers in version 9.9.4, and a few other name server implementations now support it, including NSD and Knot. As folks upgrade their name servers to newer versions or new implementations that support RRL, this should gradually make it more difficult for attackers to use DNS infrastructure as amplifiers.

I hope this discussion has helped you understand how DNS infrastructure is both targeted and exploited in DDoS attacks, and how you can best resist DDoS attacks and ensure your name servers don’t, unknown to you, participate in one.

Cricket Liu has co-authored all five editions of O’Reilly’s “DNS and BIND” book, which is generally regarded as the definitive guide on all things relating to the Domain Name System. Cricket is currently chief infrastructure officer at Infoblox.