

October 2016

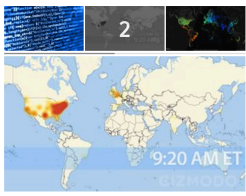
Geoff Huston

DNS DDOS

The recent attacks on the DNS infrastructure operated by DYN in October 2016 have generated a lot of comment in recent days. Indeed, it's not often that the DNS itself has been prominent in the mainstream of news commentary, and in some ways this DNS DDOS prominence is for all the wrong reasons! I'd like to speculate a bit on what this attack means for the DNS and what we could do to mitigate the recurrence of such attacks.

GIZMODO SPLOID PALEOFUTURE 109 FIELD GUIDE FACTUALLY

You may also like



This Is Why Half the Internet Shut Down Today

William Turton
Friday 9:36am · Filed to: DNS




Image: Getty

Twitter, Spotify and Reddit, and a huge swath of other websites were down or screwed up this morning. This was happening as hackers unleashed a large distributed denial of service (DDoS) attack on the servers of Dyn, a major DNS host. It's probably safe to assume that the two situations are

Report of the incident from Gizmodo.com (<http://gizmodo.com/this-is-probably-why-half-the-internet-shut-down-today-1788062835>)

I should note at the outset that when writing this soon after the event is a situation when there is not a lot of authoritative information about the attack, so we'll need to make a few guesses as to what was going on with this attack.

What we think is a likely guess:

- Firstly, it was a DNS attack. Perhaps this guess is more of an assumption than something we think is a likely guess. It is evident that the authoritative name servers of certain domain names were the target of this attack, and it is certainly possible that it could've been a ping attack or any other form of IP packets that attempt to saturate the network resources close to the locations of the authoritative name servers. But let's proceed here with the assumption that the attack was one that attempted to overwhelm a collection of DNS name servers with a set of otherwise quite conventional DNS queries. What would've been different about this activity that would make it an attack was the volume of such queries, and the span of end points generating such queries.

- The attack was based on sending “normal” DNS queries. This was evidently not a cause of using a crafted query that exercised some supposed vulnerability in the DNS infrastructure, but simply a case of sending a large volume of otherwise quite conventional queries.
- The attack was directed at the authoritative name servers for a set of target domain names
- The previous assumption would indicate that the queries were like *random-label.target-name*.
- The attack was reported to use cameras and other bits and pieces of internet-connected craft and co-opted them to launch DNS queries.

What we don't know:

- We don't know if all the co-opted devices were discoverable on the public IPv4 space or were located behind NATs – but it probably doesn't matter. The reported number of such devices (some “10s of millions”) (<https://dyn.com/blog/dyn-statement-on-10212016-ddos-attack/>) would tend to suggest that the pool of devices lie predominately behind NATs. However, I have to observe that this is a very large number and it does present some plausibility issues with the report.
- We don't know if the script included source address spoofing or not. We think not, as the NAT assumption above would tend to say that this is an attack where the source address cannot be readily spoofed.
- We don't know if attack used normal DNS infrastructure and made DNS queries via recursive name servers, or whether they were scripted to directly query particular authoritative name servers. Getting a device to pull down a particular URL is all part of a bot command and control channel, so at the very least you need to have the device perform a URL fetch upon command. In this case that basic ability is enough for such an attack. Explicitly scripting the device to perform a query directly to an authoritative name server requires further access to the particular tool or library call that allows this crafting of a DNS query, and this may not be an option on all these these co-opted devices.

A statement released by DYN (<https://dyn.com/blog/dyn-statement-on-10212016-ddos-attack/>) says "We observed 10s of millions of discrete IP addresses associated with the Mirai botnet that were part of the attack" If the attack used the conventional recursive resolver infrastructure, then such direct observation from the authoritative name server would not be possible, as all they would see is a large query volume from the usual set of recursive resolvers that handle all other DNS query traffic. So the claim from DYN that they directly observed some 10s of millions of discrete IP addresses would tend to suggest that the co-opted devices were communicating directly with the authoritative name servers and passing their queries directly to them, rather than via the recursive resolver infrastructure.

- We don't know what DNS query type was being used in this attack (A, AAA, ANY, or any other query type). We can assume that the simplest attack is a brute force attack with minimal device-scripting, so the co-opted device might well just send an A query type.
- We don't know if the queries specified EDNS(0) DNSSEC OK or any form of DNS options or padding to increase the size or complexity of the query or its answer. If the aim is to exhaust the resources of the authoritative name server then query options might well have been used, but again this requires additional capabilities in the attack script being run on the compromised devices.

What we can guess:

- We guess that this was a relatively simple script or set of scripts running on a large volume of co-opted devices without necessarily requiring a sophisticated root kit or other specialised penetration code running on the compromised device. It's likely to be a simple exploit.
- We guess that the code used in the attack had a command and control interface that allowed the co-opted device to start sending, at a minimum, very basic DNS queries with random terminal labels at a certain time.
- We guess that there were a lot of compromised devices enlisted in the attack (as distinct from a small number of devices performing source address spoofing) so that the per-device query rate was not necessarily excessive.

What happened (in DNS terms)

Because of the use of a unique label component as the terminal name in the query, the query is passed to the authoritative name servers for resolution. This will occur even if the local environment performs DNS interception and redirection and diverts externally address DNS queries into the local DNS resolver infrastructure.

The authoritative name servers for the attacked domains were evidently overwhelmed by the query volume, and started to drop queries.

When recursive name servers attempted to refresh their cache entries on the attacked names, evidently no authoritative name servers were able to answer these queries, so the recursive resolvers dropped the name from their local cache once their local cache expiry timer expired.

As the recursive resolvers lost their local cache entry, the attacked names started to disappear off the net.

Mitigations for this form of DNS DDOS Attack

There are a number of potential responses that could work to mitigate this form of attack. They are not equally effective, and indeed not all are even viable today. But they illustrate the range of potential responses to this form of attack on the DNS.

1 - More Foo

A common response to most forms of DDOS attack has been to build the wall higher.

In the DNS case, the victim domain name can add more name servers to the list of authoritative name servers, implying that an attacker needs to increase their attack volume if they want to overwhelm the entire authoritative name server set.

Another response is to add more capacity to the existing authoritative name servers, by adding more authoritative name servers into an anycast constellation, or by adding the number of name server engines and using a front end load balancer. Another option may be simply adding more memory and processing capability to the existing machinery.

The overall intent of all of these measures is to increase the query response capability to exceed the query volume presented during the attack. As long as this can be achieved the authoritative name servers will not be pushed into being unresponsive, and the attack would fail.

2 - Longer TTLs

The recursive name servers that handle user queries will hold in their local cache the valid responses from authoritative name servers for the presumably small set of defined names in the attacked name space for a specified period of time. As this timer draws down the recursive resolver will attempt to refresh the cached data. If the expiration timer expires without a successful cache refresh the local resolver will purge its cache of this entry.

A possible measure is not to address the attack per se, but to note that the that attack must last for at least the cache time to live (TTL) period in order to ensure that the recursive resolvers' cache expires in all resolvers.

Of course the problem with this is that the expiration timers running in the recursive resolvers will all be running with different epoch settings, so that in the period when the authoritative name servers are unavailable the set of recursive name servers carrying cached records will gradually expire, causing the name space to gradually wink out across the recursive resolver set, and the pool of affected users will grow as the attack continues.

Longer cache expiration times, if uniformly observed by recursive resolvers, will reduce the rate of decline in visibility, but not eliminate it. However, the observed behaviour that many recursive resolvers appear to behave as if they overwrite the authoritative server's timer values for the zone with local values, so setting a longer cache lifetime in the zone file may not have the desired effect on all recursive resolvers.

3 - Filter Queries

If the attack uses a random name part that has a fixed pattern, then it is possible to filter out these queries at the authoritative name server and drop them before the server's resources are consumed in generating a DNS response.

Of course this is a temporary measure in so far as the next attack will probably vary the random name part in other ways, but as a first response in attack traffic discard its often a useful measure.

4 - IP address filters

There is one observation that is potentially helpful in this space - the pool of IP addresses that query authoritative name servers falls into two distinct pools based on a simple classification of whether or not the address matches the set of known visible recursive resolvers.

There are some 10,000 IP addresses that correspond to the collection of visible recursive resolvers that query authoritative name servers that appear to serve some 95% of the entire user base of the Internet. It is feasible to load this set of addresses into a FIB cache of a front end router and perform a wire speed classification of queries into queries from known recursive resolvers and anomalous queries from individual devices using FIB lookup. The former could be placed into a normal query queue, while the other could be placed into a lower priority processing queue that might well fail under load. Under normal query loads this measure would be all but invisible to all users, while under the stress of an attack that directly contacted the authoritative name server the known recursive resolvers would still receive service while all other queries would experience some query drop rate.

This measure is effective if the attack is based on scripting the co-opted devices to directly query the authoritative servers, and the circuits and switches that provide access to the authoritative name server have sufficient capacity to carry all the queries, including the attack traffic, to the classifying router or routers.

If the attack uses the conventional recursive resolver infrastructure, then this measure is largely ineffectual as it's the known recursive resolvers that are presenting the attack queries. But this is the case, then another form of mitigation that uses the recursive resolvers to absorb the attack may be feasible.

5 - Aggressive NSEC-based caching

This is not a realistic option today, but it is a means of improving the resilience of the DNS in the case where such random name attacks occur through the recursive resolver infrastructure (the opposite of the case of the IP address filtering option).

When a name does not exist, as is the case for a random name attack, a signed zone response for the query is not only the NXDOMAIN code, but an NSEC record that indicates the span of the zone file than covers the query label. A recursive resolver could cache this record and use it to respond to any query that falls within the same span of names without further reference to the authoritative name server (see <https://tools.ietf.org/html/draft-ietf-dnsop-nsec-aggressiveuse-05> for the details of this approach).

```
$ dig +dnssec www.not-defined-here @a.root-servers.net

; <<>> DiG 9.10.4-P3 <<>> +dnssec www.not-defined-here @a.root-servers.net
;; global options: +cmd
;; Got answer:
;; -->HEADER<<-- opcode: QUERY, status: NXDOMAIN, id: 33673
;; flags: qr aa rd; QUERY: 1, ANSWER: 0, AUTHORITY: 6, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 1232
;; QUESTION SECTION:
;www.not-defined-here.          IN      A

;; AUTHORITY SECTION:
norton.          86400  IN      NSEC   now. NS DS RRSIG NSEC
norton.          86400  IN      RRSIG  NSEC 8 1 86400 20161108050000 20161026040000 39291 . [sig]
.                86400  IN      NSEC   aaa. NS SOA RRSIG NSEC DNSKEY
.                86400  IN      RRSIG  NSEC 8 0 86400 20161108050000 20161026040000 39291 . [sig]
.                86400  IN      SOA    a.root-servers.net. nstld.verisign-grs.com. 2016102600 1800 900 604800 86400
.                86400  IN      RRSIG  SOA 8 0 86400 20161108050000 20161026040000 39291 . [sig]

;; Query time: 182 msec
;; SERVER: 2001:503:ba3e::2:30#53(2001:503:ba3e::2:30)
;; WHEN: Wed Oct 26 05:12:47 UTC 2016
;; MSG SIZE rcvd: 1039
```

In this example DNS query and response, the response indicates that all names between “norton” and “now” are not defined in the root zone, and precisely the same authority section of the response could be used by the recursive resolver for any query name between “norton” and “now” for as long as this negative response is held in the recursive resolver’s cache, without any further querying of the authoritative name server.

What this measure allows is that in the case that a DNS random name attack is launched using the recursive resolver infrastructure on a DNSSEC-signed zone that is using NSEC or NSEC3 negative records that span the entire zone, then the recursive resolver infrastructure can be used to cache these responses from the authoritative resolver and use them to respond to subsequent attack queries where the random label falls into the span defined in the response. As the attack continues the recursive resolver will learn to respond with authoritative NXDOMAIN responses to the complete range of random values found in the query, thereby using the recursive resolver infrastructure to absorb the attack close to the attacking devices.

There are three problems about this approach which make this particular mitigation response frustratingly unhelpful: Very few recursive resolvers perform DNSSEC Validation, even fewer recursive resolvers have implemented the approach described as “aggressive use of NSEC”, and, thirdly, disturbingly few domains are DNSSEC signed in the first place. Perhaps this rather disturbing attack incident might motivate some further action in this space.

Some Closing Observations

This is never going to go away. The sheer volume of consumer devices being marketed today is a fertile breeding ground for not just low cost, but extraordinary low quality, devices that use corruptible and often already corrupted software.

Their volume of distribution within a rapacious consumer market that is largely ignorant of technical quality and robustness and is incredibly price sensitive ensures that poor quality cheap and essentially unsafe devices will continue to populate the edges of the Internet. For as long as this is happening, it will continue to be possible to orchestrate thousands if not millions of these devices into large-scale attacks that are capable of overwhelming most of our defences. It's hard to see how just building higher walls of defence will work as a long term strategy.

We can be smarter about this, and we can use both DNS and DNSSEC, and our knowledge of the way the DNS actually works to build a more robust DNS infrastructure that could be more capable of deflecting these forms of attack. Whether we have the collective motivation to take these steps and actually build a more robust and resilient DNS resolution infrastructure remains to be seen.

Of course this will not stop the attacks. Like the lion chasing its prey, the first objective of the potential victim is to run away faster than at least one of the others! In this case, the objective is to push the potential set of exploitable vulnerabilities away from the DNS. While is still work to do in other components of the Internet's infrastructure, at the very least we can take steps to ensure that the DNS is no longer the easy target of such attack.

That is assuming, of course, that we truly have a strong desire to try to fix this!

Postscript

As I have attempted to point out in the introduction, this article has been written based on the assumption that this attack used DNS queries. From the public information provided so far there is no basis to believe this assumption over and above the assumption that this was an instance of any of the more "traditional" forms of exhaustion attacks, namely pings, TCP SYN flooding, GRE packets and other ways to clog up the wire and the server(s), as distinct from specifically clogging up the DNS function of the servers. Much of this article is speculative in nature, looking at potential mitigation measures if we had a DNS query attack.

If this was a brute force exhaustion flooding attack then its perhaps a little harder to speculate upon mitigation measures, as its often the case that if you cannot prevent the packets at the source, then you need to find some readily identifiable "signature" of the attack stream that will permit you to pull out the attack traffic before it reaches the critically stressed resource.

We may, or may not get to know more about this particular attack – but one thing we can expect with much confidence: there will be more attacks. The Internet of Stupid Things is not going to get any smarter any time soon!

Author

Geoff Huston B.Sc., M.Sc., is the Chief Scientist at APNIC, the Regional Internet Registry serving the Asia Pacific region. He has been closely involved with the development of the Internet for many years, particularly within Australia, where he was responsible for building the Internet within the Australian academic and research sector in the early 1990's. He is author of a number of Internet-related books, and was a member of the Internet Architecture Board from 1999 until 2005, and served on the Board of Trustees of the Internet Society from 1992 until 2001 and chaired a number of IETF Working Groups. He has worked as an Internet researcher, as an ISP systems architect and a network operator at various times.

www.potaroo.net

Disclaimer

The above views do not necessarily represent the views or positions of the Asia Pacific Network Information Centre.