The Rising Tide: DDoS by Defective Designs and Defaults

Richard Clayton

University of Cambridge, Computer Laboratory, William Gates Building,
15 JJ Thomson Avenue, Cambridge CB3 0FD, United Kingdom
richard.clayton@cl.cam.ac.uk

Abstract

We consider the phenomenon of distributed denial of service attacks that occur through design defects (and poorly chosen defaults) in legitimately operated, entirely secure systems. Particular reference is made to a recently discovered "attack" on stratum 1 Network Time Protocol servers by routers manufactured by D-Link for the consumer market, the latest example of incidents that stretch back for decades. Consideration is given to how these attacks might have been avoided, and why such failures continue to occur.

1 Introduction

Currently, when considering distributed denial-of-service (DDoS) attacks, the image is of a wicked individual commanding an army of "zombies" to emit specially crafted Internet Protocol (IP) packets so that an innocent victim is overwhelmed by the incoming traffic. The effect (a system that is unable to communicate effectively) can be just the same when there is a sudden rise in legitimate traffic, a "flash crowd" or "the Slashdot effect". Between these two extremes lies a further type of denial-of-service, where design defects in hardware or software create a substantial amount of traffic that is not malicious *per se*, but not necessarily legitimate either.

This paper is about these design error "attacks". There will typically be a slow increase in intensity; starting with small amounts of traffic from a handful of sources, but growing into a significant nuisance when the traffic is arriving from thousands or millions of endpoints. One might think of zombie-based DDoS attacks as directing a firehose onto the victim; likening slashdotting to being drowned in a flash flood; and the design errors creating an inexorably rising tide. In all three cases the result can be drowning.

In Section 2 we present a number of historical examples of DDoS attacks caused by design errors, then in Section 3 we describe a brand-new example in some detail. In Section 4 we divide the general problem of design errors into categories and discuss the type of mitigation and/or avoidance that could be appropriate. Finally in Section 5 we summarise and draw some fairly gloomy conclusions.

2 Historical Examples

2.1 `HOSTS.TXT`

The original ARPANET mapped hostnames to network addresses by means of a flat-file database (`HOSTS.TXT`) which was centrally updated and then distributed to every participating machine. The system was replaced (by about 1987) with the Domain Name System (DNS) that we still use today. A commonly cited justification for the need for the new system is the ability to delegate the responsibility for changes – it could take several days for administrative staff to add a new host to the central file.

More significantly, in the current context, these changes were batched together and an announcement made when a new file version was available. There would then be a flash-crowd as every ARPANET host attempted to fetch the new copy [15]. This effect was getting ever worse ($O(n^2)$) as the number of downloaders increased and the size of what they were downloading increased in proportion as well.

2.2 Root Nameserver Traffic

In January 2001 Brownlee *et al* studied [2] the traffic at the 'F' root nameserver, one of the machines that handles queries for data at the top-most level of the DNS. They found one host sent the same query 2 112 962 times in an hour (587 queries per second) and that 12%–18% of queries for A records were for IP addresses (viz: for values that were already resolved). Their overall conclusion was that a significant part of the system load could be directly attributed to traffic that did not follow the DNS protocol.

Although their analysis identified flaws in many different types of software (including some email viruses that made DNS queries) they drew attention to three specific issues from Microsoft – which their market share makes especially relevant.

Firstly, names from Microsoft's local networking protocols, such as WORKGROUP, were turning up in root nameserver queries as `.workgroup`. Secondly, Windows 2000 systems defaulted to generating "dynamic update requests" (reflecting reallocation of IP addresses by DHCP on local networks) and some of these updates were – entirely pointlessly – being sent to the root nameservers (in one case, a single machine sent 15 000 requests to 'F' in a day). Thirdly, when a DDoS attack on a router made Microsoft's nameservers unavailable (in defiance of Best Practice, they were both on the same subnet) requests for the top-level records for Microsoft domains reached 25% of the overall query load.

2.3 Netscape's Parallel Downloading

Until Netscape was released in October 1994, the standard way of fetching a web page was to download the file, parse it to locate further items – such as embedded images – and download those one at a time. Netscape ran faster by fetching the images in parallel, using a user-settable number of connections, defaulting to 4. Initially, this was extremely controversial, because it caused many more processes to be running on web servers, and this was considered to be tantamount to a DDoS attack (see, for example, [6] and related articles in the `comp.infosystems.www.misc` newsgroup in January 1995). In time, it became apparent that servers can be configured to cope, and this has become the standard way of fetching web pages (although note that in HTTP/1.1 multiple items can be fetched over a single connection).

Similarly, `qmail`, Dan Bernstein's Mail Transfer Agent (MTA) has attracted criticism (and robust defence) for its parallel delivery of email messages (which defaults to 20 parallel streams, but is often set higher) [4]. The community has dealt with this by ensuring that other MTAs have (configurable) limits on the number of incoming connections per host that they are prepared to accept.

2.4 Mojo Nation

Mojo Nation was an "emergent file store" that ran from July 2000 until February 2002 [17]. It was a distributed data haven that used digital cash ("mojo") to pay for storage services. Over 100 000 users downloaded the software, although there were only ever 100–600 persistently connected nodes and a maximum system size of about 10 000 nodes. The designers were surprised to find that sites were connected for short periods (less than an hour) and many sites never returned after connecting to the network once.

Unfortunately, the system had a single "original introduction" server which provided a list of existing nodes to new nodes that wished to join in. When Mojo Nation was mentioned on Slashdot [12] as an improvement on Napster and Gnutella, the server was overwhelmed by the large number of new users and failed to return responses to users for several days.

2.5 Network Time Protocol Servers

In 2000 the University of Delaware changed the name of their NTP server from `time.ultimeth.net` but three years later they found that DNS requests for it were increasing over time. A Windows program called 'NetTime' had hard-coded their system identity, along with a number of other well-known servers. In September 2003 they removed the domain from the DNS, but traffic was still arriving several weeks later, probably because of flawed DNS caching [8].

In October 2002 a web server at Trinity College Dublin was overwhelmed by HTTP traffic generated by 'Tardis', a shareware program for setting the time on Windows machines. A web-based time access system had been added for users whose firewalls blocked NTP traffic, and users were requesting synchronisation as often as once per minute. The resulting load, when it eventually became too much for the server to handle, was about 420 requests per second and rising. The software was not patched for some time, but reconfiguring the server to provide a bogus time led to a substantial fall in requests [13].

In May 2003 the University of Wisconsin – Madison found itself at the receiving end of continuous large-scale flood of inbound Simple Network Time Protocol (SNTP) packets [16]. The rate exceeded 280 000 packets per second. The source was determined to be consumer equipment (routers) manufactured by Netgear, of which there were about 700 000 in the field. These devices sent SNTP packets requesting the time and, if there was no answer within one second (!), asked again. Having received a satisfactory answer no further request would be made for one minute (or longer periods up to 24 hours, depending upon the product). Clearly, with this algorithm, once the connection to the SNTP server became congested, any systems that managed to receive answers would be contributing to the traffic again before the remaining systems had been lucky enough to get an answer.

In July 2003, two stratum 1 NTP servers run by

CSIRO in Australia were moved to "secret" IP addresses to avoid being accessed at a rate of 6 000 packets per second by approximately 85 000 routers manufactured by SMC Networks [18, 9]. The devices only accessed the clock a few times a day, but when the server operators decided this traffic was excessive and blocked it, the device firmware increased the request frequency to twice a minute. A contributing factor was that many of the routers were behind firewalls that discarded the NTP response, so these were already making requests at the faster rate.

2.6 Dynamic DNS

Many ISPs provide their customers with "dynamic IP addresses" that may be changed to a new value on the next connection, or at a later time. However, customers may wish to run servers, such as a web hosts, at invariant hostnames. Hence DNS entries must be updated to track the changing IP address. A number of companies provide a service whereby an HTTP transaction can trigger a DNS update, and it has become commonplace for cable modems and ADSL routers to be configurable to automatically perform an update whenever their IP address changes.

In October 2005 Dynamic Network Services Inc. (who operate dynamic DNS domains such as `dyndns.org`) announced that they would be failing to action HTTP requests containing the string `client/1.0`, which they believed to be sent by some models of D-Link device [7]. They had 1.4 million customers, but were receiving 21 million invalid updates per day, and estimated that approximately 10 000 devices were generating 25% of their traffic load.

3 D-Link & Stratum 1 NTP

In the late summer of 2005 Poul-Henning Kamp discovered that a very high proportion of the traffic to `gps.dix.dk` (the stratum 1 NTP server he operated in Denmark) consisted of obsolete NTP version 1 requests. All of the machines who should have been "chiming" against his system were using the current, NTP version 4, request packets, so the specious traffic was easy to quantify. He concluded that he was suffering a DDoS attack from zombie-infested computers and asked for help in tracking down the "botnet" he thought was responsible.

He collected the traffic in `tcpdump` format over a number of days and made it available for analysis to the present author. On a typical day, 1 Nov 2005, 3.19 million NTPv1 packets arrived (37 per second) from 276 256 unique IP addresses. Collating these IP addresses by AS (ISP) made it clear that if the source of the traffic, which uses UDP for transport, was spoofed, then the perpetrator had made an excellent job of mimicking the actual usage of IP addresses. It was far more likely that the source addresses were valid. After identifying a source IP address in the UK and contacting the user, it was possible – having eliminated a number of other possibilities – to identify that the traffic had been sent by a "DI-624" wireless router manufactured by D-Link.

An identical DI-624 was purchased, which arrived with v2.42 firmware dated 31 Mar 2004. In its default state (with no NTP server specified by the user), the device used a preset list of 63 NTP servers (including `gps.dix.dk`). Every 2.2 seconds the device made a DNS request to resolve a name selected from this list (in an unpredictable order, often resolving the same host several times in a row), and every 30 seconds it issued an NTPv1 request packet. Although initially the NTP requests were to the most recently resolved server, later requests could immediately precede the re-resolving of the relevant hostname. Hence the device was issuing many pointless DNS requests, ignoring the DNS conventions on the validity of cached results, and – by sending out two NTP requests a minute – ignoring the NTP conventions as well.

It will be noted that there is some discrepancy between the average arrival rate of NTPv1 packets at `gps.dix.dk` (averaging one per 2 hours per source) and the rate at which they are being sent (about once every 30 minutes to any particular NTP server). This can be partly explained by the use of dynamic IP addresses, but may well be because other D-Link products (several models were found to be using stratum 1 NTP servers) retry slightly less often.

Of the 63 NTP servers in the v2.42 firmware list, 52 remain (April 2006) in the `isc.org` canonical list of stratum 1 servers [10]. Of these, 24 are "Open Access" and do not require users to notify the owners of usage; 6 are "Open Access" but require notification; 5 are "Restricted Access" but don't require notification; 15 are "Restricted Access" and require notification; and the last 2 are "Closed Access". Although "Open Access" servers "may be used without restrictions by any client in any location", they are also documented to have particular "Service Areas", showing which particular countries or networks they are "intended to serve". More importantly, the standard "Rules of Engagement" (`http://ntp.isc.org/bin/view/Servers/RulesOfEngagement`) require clients that access stratum 1 servers to be in synchronisation subnets of two or more systems and to themselves be providing service to more than 100 other clients. Quite clearly, the D-Link DI-624 falls well outside all of these criteria.

The latest version of the firmware available for UK devices (in April 2006) is v2.59b3 (dated 30 Nov 2005). The server list is markedly changed from v2.42 (but identical to the list in v2.53 – 20 Apr 2005), with 31 entries removed (including `gps.dix.dk`) and 36 added to give a total of 68, all bar one of which is on the current canonical list. However, although 30 are now classed as "Open Access" the other systems still have restrictions and the two "Closed Access" systems remain. Also, although the server list has changed, the dynamic behaviour – synchronising the time every 30 seconds – is unaltered.

4 Addressing DDoS by Designs

We can identify several types of activity that give rise to the DDoS examples cited above:

Service Discovery is obtaining the identity of systems that can provide a service. For example, the ARPANET `HOSTS.TXT` file provided the identity of every machine on the network, and the Mojo Nation "original introducer" provided a list of active nodes.

Service Access is the use of a service, and the DDoS occurs from there being just too many users; specifically, in the various NTP examples, because access is being made by inappropriate systems.

Broken Systems are generating traffic which is just plain wrong and this causes DDoS effects, either first-hand, as in the systems identified by Brownlee *et al* sending hundreds of DNS queries per hour (or per second), or second-hand where the simultaneous failure of the `microsoft.com` DNS servers significantly increased the load on the root servers.

In addition, in a handful of the examples, such as the parallel usage of HTTP and SMTP streams, the "DDoS" has turned out, in the long term – with appropriate default behaviour – to be liveable-with.

4.1 Mitigation

Mitigation of DDoS caused by design issues is usually achieved through the replication and distribution of servers. For example, clients access DNS servers that cache results so as to avoid repetitive fetching from authoritative sources. This is combined (or should be) with the replication of those authoritative sources and the "root servers". In a like manner, Mojo Nation addressed its problems by replicating its servers.

For web content, companies like Akamai (`www.akamai.com`) now provide "content distribution" networks, specialising in serving popular content from servers that are hosted at ISPs all over the world. As servers fail or become overloaded they redirect requestors to other locations where the same content will be available – though they do this at the expense of reducing the timeout on DNS data to a few minutes or seconds, thereby adding load to that system.

The time servers also operate a distributed system but, crucially, it is based on convention rather than any on-the-wire protocol. The "stratum 1" servers are connected to atomic clocks or receive timing signals from GPS satellites. They are accessed by "stratum 2" servers that fetch the time from several stratum 1 machines, analyse the timestamps and round-trip duration, and thereby estimate the correct time. Similarly "stratum 3" servers access stratum 2 servers to get an only slightly less accurate time value. End-user machines, at the bottom of the pyramid, should be accessing stratum 3 servers.

The end-user devices have several ways of discovering local (stratum 2 or 3) NTP servers. They could use DHCP to obtain the IP addresses (the option code is 42 and was described in RFC2132 [1]); they could use the `pool.ntp.org` project – which maintains a list of almost 600 time servers and provides appropriately chosen random selections in response to DNS queries; or the user could be asked to manually configure a server name.

However, as was clear from Section 2 above, a number of manufacturers have chosen to obtain a list of stratum 1 time servers (these lists are the easiest to locate), have failed to read up on the usage restrictions, have placed these hostnames into their firmware, and have then shipped tens of thousands of units. They have also, on occasion, compounded their error by generating requests at unreasonable rates and failed to "back off" after a connection failure.

This has left the NTP servers with a complex mitigation problem. The requests consume processing power and bandwidth – whether answered or not – and this impacts the timing accuracy for legitimate users. Filtering the traffic at a network ingress point may be impractical (in the D-Link case, the undesirable traffic was easy to recognise since it was NTPv1, but other examples were more complex). The latest version of the time protocols [14] does contain a "kiss-of-death" response, used to tell clients to stop their requests, but implementations based on older documentation will naturally take no notice.

If the servers change their name then this affects their legitimate users, who cannot now locate the system. In principle, the name changing could be done by providing different DNS results to different requestors. But the distributed nature of the DNS could mean that a remote stratum 2 server (which should be given service) resolves the stratum 1 server name using a recursive DNS server that is also employed by hundreds of thousands of consumers.

A different solution is the use of an out-of-band authorisation system. The clients present credentials (or a compelling reason) to the authorisation system and it then provides a single-use server name which can be resolved to give the server address. There are obvious trade-offs between agility (moving the server around to prevent unauthorised access) and inconvenience to legitimate clients – that have to revisit the authorisation system. This system, albeit manually operated and with the IP address provided directly, is what is currently operated by the Australian National Measurement Institute to control access to the NTP servers formerly hosted by the CSIRO [18].

4.2 Education

It is reasonable to ask why there appears to be so much repetition in the examples cited above. The NTP servers seem be encountering problems on a regular basis, and the DNS traffic problems described by Brownlee *et al* are similar to those studied by Danzig *et al* almost a decade earlier [5].

The NTP incidents indicate that the high-level, conceptual, design of the system is poorly understood by the calibre of people who are currently implementing clients. This is worrying, because many aspects of the suite of Internet Protocols depend upon showing restraint, and the need for that restraint is not always immediately apparent. This may help to explain why, when flaws are suspected – as in the Netscape and `qmail` examples – the reaction from the developer "establishment" can be quite heated.

However, community-level design review remains relatively uncommon, and so the lack of comprehension that problems are possible, and the increasing ease of systems development by "just anybody", could well mean that we see ever more DDoS from design flaws. Many of the highest users of bandwidth (such as video and peer-to-peer file-sharing) avoid TCP and roll their own UDP-based schemes, paying limited attention to congestion issues. This is a key driver for the provision of protocols such as DCCP [11], which aim to provide an easier-to-use alternative to UDP that will – by design – prevent the clueless from DDoS-ing the Internet.

4.3 Economics

It used to be that by the time one had built a TCP/IP stack the ethos of restraint had been absorbed ('security through inculcation'), but if the next generation of stack builders try to obtain more performance for themselves and do not bother to back off in the face of congestion then we could see cascading network collapse. This, along with several of the historical examples, is essentially the "tragedy of the commons" and so it is attractive to consider an economic approach to fixing these problems. For example, after the SNTP problem, Netgear made a donation of $375,000 to the University of Wisconsin – Madison [3]. Clearly, this was not widely enough reported (or a sufficiently large sum) to register with the other manufacturers whose designs have caused similar DDoS problems.

Voluntary donations are never likely to be a long-term solution, and a financial penalty approach would need underpinning by the courts. Unfortunately, the legal system is not currently well suited to this type of problem. It is not especially easy to issue proceedings in other countries; individual damage may be small and so "class actions" would be required – posing difficulties in some jurisdictions; and it remains unlikely that local regulators (or consumer protection officials) will be interested in deeply technical matters, where apparently minor errors cause significant impact to other sites elsewhere in the world.

Another economic approach is to consider where the incentives are. When a design causes a DDoS, why should any of the millions of "attackers" fix their part of it? End-users can be frightened into removing zombies from their machines by suggesting that the security of their own data (and the likelihood of keeping their ISP account) is at risk. But no ISP is, realistically, going to disconnect customers for sending 120 NTPv1 packets an hour; and whether or not a wireless router knows the correct time is pretty much irrelevant to most people.[1] The experience of Trinity College, Dublin was that returning the wrong time can reduce traffic (though users may well have discarded the software rather than reconfiguring it), but it also showed that software, once shipped, can be just as hard to get updated as firmware.

5 Conclusions

We have shown that DDoS is not restricted to activity channelled through insecure zombies, but can also occur through requests that would be a tolerable nuisance in small numbers, but in the mass can have devastating results. These requests come from systems whose design just doesn't scale, or which contain bugs, or that encapsulate misapprehensions about what is acceptable behaviour. These problems have, to large extent, been addressed by replicating resources, but many still continue to pose significant challenges for the "attacked" machines.

[1] Which leads one to wonder why manufacturers bothered with NTP in the first place!

This type of DDoS is unlikely to be fixed by users (or ISPs on their behalf) "taking responsibility" for the traffic, since from their point of view they have done nothing terribly wrong. The "attacked" machines will get little mileage out of creating lists of "attackers" and demanding their disconnection.

What is most disturbing is the sense of déjà vu one gets from examining these events. The same types of error seem to be recurring, suggesting a failure to properly educate system developers to avoid designs that can DDoS. Looking at the problem from a security economics point of view is equally gloomy. The incentives are not suitably aligned to address the problem, and where manufacturers have paid for their mistakes this not yet been effective at preventing others from falling into the same trap – and there seems little short-term likelihood of a legal approach being effective at the network-scale necessary.

Finally, it is noteworthy that the systems that are being "attacked" are seldom managing to detect the undesirable traffic at an initial low volume (or even at an intermediate stage when its nature is bound to be apparent), but are only becoming aware of an issue when the problem has become acute. In 1992, Danzig *et al* recommended regular "policing" of server logs to detect anomalies and contact defective systems. This is still not common practice – but surely if one lives in a tidal area, keeping an eye on the water level is just commonsense.

Acknowledgments

The identification of the D-Link traffic was made possible with the assistance of Nick Webb and his family. Richard Clayton is currently working on the spamHINTS project, funded by Intel Research.

References

[1] S. Alexander, R. Droms: DHCP Options and BOOTP Vendor Extensions. RFC2132, IETF, 1997.

[2] N. Brownlee, kc claffy, E. Nemeth: DNS Measurements at a Root Server. IEEE Global Telecommunications Conference, 2001, (GLOBECOM '01), 25–29 Nov 2001, vol 3, pp. 1672–1676.

[3] Division of Information Technology, University of Wisconsin – Madison: Serendipity and hard work pay off. 21 Apr 2004. http://www.doit.wisc.edu/news/story.asp?filename=322

[4] Common Vulnerabilities and Exposure (CVE) List: Denial of service in Qmail by specifying a large number of recipients with the RCPT command. Candidate (under review) CVE-1999-0144. 7 Jun 1999.

[5] P. B. Danzig, K. Obrackza, A. Kumar: An Analysis of Wide-Area Name Server Traffic: A study of the Internet Domain Name System. ACM SIGCOMM Computer Communication Review, 22(4), Oct 1992, pp. 281–292.

[6] S. Desai: Re: FAQ: Netscape Communications and our products. comp.infosystems.www.misc, 14 Jan 1995. http://groups.google.com/group/comp.infosystems.www.misc/msg/e194375a44e90ea8

[7] Dynamic Network Services Inc.: User-Agent Block: client/1.0. 31 Oct 2005. http://www.dyndns.com/about/company/notify/archives/useragent_block_client10.html

[8] D. Gibson: Why we have abandoned the UltiMeth domain. Sep 2003. http://www.ultimeth.com/Abandon.html

[9] HPCwire: Network Devices Almost Take Down Atomic Clock. 11 Jul 2003. http://www.taborcommunications.com/hpcwire/hpcwireWWW/03/0711/105537.html

[10] Internet Systems Consortium Inc.: Stratum One Time Servers. http://ntp.isc.org/bin/view/Servers/StratumOneTimeServers

[11] E. Kohler, M. Handley, S. Floyd: Problem Statement for the Datagram Congestion Control Protocol (DCCP). RFC4336, IETF, Mar 2006.

[12] R. Malda: Forget Napster & Gnutella: Enter Mojo Nation. 9 Oct 2000. http://slashdot.org/article.pl?sid=00/10/09/1826243

[13] D. Malone: Unwanted HTTP: who has the time. Usenix ;login 31(2), Apr 2006.

[14] D. Mills: Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI. RFC4330, IETF, Jan 2006.

[15] National Research Council Committee on Internet Navigation and the Domain Name System: Signposts in Cyberspace. National Academies Press, Sep 2005, ISBN: 0-309-09640-5.

[16] D. Plonka: Flawed Routers Flood University of Wisconsin Internet Time Server. 21 Aug 2003. http://www.cs.wisc.edu/~plonka/netgear-sntp/

[17] B. Wilcox-O'Hearn: Experiences Deploying a Large-Scale Emergent Network. in: P. Druschel, F. Kaashoek, A. Rowstron (Eds.): Peer-to-peer Systems, First International Workshop, IPTPS 2002, Cambridge MA, USA, Mar 2002, pp. 104–110.

[18] M. Wouters: Re: Improved monlist scheme – a bit off topic. ntp:hackers mailing list, 15 Sep 2003. http://lists.ntp.isc.org/pipermail/hackers/2003-September/000326.html