

# Practical DNS protection

Marco Slot

Division of Mathematics and Computer Science  
Vrije Universiteit, The Netherlands  
marco@few.vu.nl

December 2007

## Abstract

Despite long-lasting attempts to improve the security of DNS fairly little has been achieved. Surveys show that the strongly advocated cryptographic DNS, DNSSEC, is not being adopted. In this paper we make a case for protection mechanisms on the resolving side of DNS. We revisit some existing attacks with recent statistics to show the inherent security problems of DNS. We argue that the next generation of DNS service providers can play an active role in protecting clients against attacks.

illusion. The adoption of the improved DNSSEC-bis version from 2005 is now at 0.000004% (with some margin of error). Clearly the market has spoken.

We must come to the conclusion that DNSSEC has failed. In this paper we make a case for hardening DNS by detecting and protecting against potential attacks rather than preventing attacks through cryptographic means. We revisit some attacks on DNS resolvers and name servers using recent statistics to show the inherent insecurity of DNS. We then look at how a new generation of DNS service providers can protect their clients better through techniques such as off-line intrusion detection.

## 1 Introduction

For over 13 years a lot of work has been done to cryptographically secure DNS in the form of DNSSEC. During that period, in which we saw the internet grow to billions of users, fairly little has been achieved. Meanwhile the protection of DNS against attacks has received little attention and nearly ever year critical vulnerabilities are found that put billions at risk of being misdirected to malicious addresses.

A recent DNS server survey by Infoblox [2] showed a staggering number. Almost 8 years after introduction 0.0018% of name servers in the COM and NET zones supported the first-generation DNSSEC. Although this could be a sign of apt system administrators as the first DNSSEC was heavily criticized and is now deprecated, the survey quickly destroys this

## 2 Birthday attack

A number of poor decisions were made during the design of the Domain Name System. Most pressing is the 16-bit transaction identifier which can be easily guessed through birthday attacks and pseudo-random number generator predictions. In theory a transaction would have an entropy of 28-30 bits with a random 16-bit transaction identifier and a random 16-bit port (with some ports reserved). In reality this is hardly ever the case. For example BIND9, which has a market share of 65%, uses a static port number and thus has only 16 bits of entropy.

The birthday attack [4] is still a larger problem to DNS servers than is often presumed. To do a birthday attack an attacker needs to trigger a large num-

ber of queries from the resolver. Each query will have one of  $2^{16}$  possible transaction identifiers and the attacker will send responses with one of  $2^{16}$  possible transaction identifiers. The probability of a match between reply and response is over 90% with 600 messages and a resolving name server will accept and cache the first match. The challenge is to beat the real name server which will always use the correct identifier. However 600 messages can be sent in an instant with today's bandwidth and if necessary we can slow down the real name server by flooding it with requests.

Preventive mechanisms for birthday attacks do exist. Resolvers need not have multiple outstanding request for the same name. However this attack can not be defeated through configuration alone if we become less picky about who our victims are. It is important to realize that the 600 requests sent by the resolvers are independent of each other and may be dispersed in time and location, as long as each of them gets the opportunity to match with 600 responses the birthday paradox will hold. We can make a large list of (the most used) recursive servers and trigger one query on each of them. We may not be able to push through 600 message before the real name server in all cases, but we can always add more servers to our list or try again later.

Although split-split architectures make it much harder to do a birthday attack because we either need to be on the inside of the network or trigger a burst of queries from the inside, there is still plenty of terrain to cover. Based on the results of their survey InfoBlox estimates there to be over 16 million open resolvers, plenty to benefit attackers.

This attack can not be thwarted by improving the random number generator, we have assumed it to be perfect. In reality weaknesses in pseudo-random number generation can be used to deceive recursive DNS servers with little effort. Recently an attack has been published that can predict the next transaction identifier for BIND 9 with reasonable accuracy after obtaining as little as 11 messages [3].

### 3 Exploiting dependencies

To prevent having a single point-of-failure in the DNS architecture names are advertised by multiple, redundant servers. For a long time it has been common practice to make agreements with other organizations to serve each other's names to get redundancy for free. Ramasubramanian et al. [6] have shown that this is not only very common, but also very dangerous due to the implicit transitive trust relationships. The attack they outlined is based on the dependencies of name servers on the names of their name servers. For example the nameserver for fbi.gov is dns.sprintip.com, while the name servers of sprintip.com are in the telemail.net and sprintlink.net domains. If one of these names gets compromised by means of a vulnerable name server the malicious information would eventually progress to the glue records of name servers higher in the hierarchy. This would fully compromise the name since queries for fbi.gov will be sent to a malicious name server. To make sure correct information of name servers which have not been compromised does not get through an attacker can slow those servers down through denial of service attacks during glue record updates.

In a survey done in 2004 by the authors found that at the time 17% of name servers had well-known vulnerabilities that could be used to compromise a name. Due to the dependencies this made 30% of the names they studied vulnerable. The names were of well-known websites listed in the Alexa top 500 or the Open Directory Project. Further they found that using only two Denial of Service attacks they could extend their reach to 84% of names. The attack was not put into practice and may not work in cases where glue records are configured manually, but given the abundance of vulnerable names it is safe to conclude that the Domain Name System can not be relied upon.

## 4 Next-generation DNS service providers

In the previous sections we have shown why DNS in its current state is inherently insecure. Software updates or new standards will not provide a cure any time soon. If we wish to protect clients from attacks on DNS our focus should no longer be on preventing, but on detecting and handling attacks.

A DNS resolution server is generally regarded a static component of the IT infrastructure and mostly a nuisance to system administrators. Recently this view has changed with the introduction of the commercial system OpenDNS [1]. This DNS resolving service provider maintains a world-wide network of DNS servers. OpenDNS actively maintains a far bigger cache than normal DNS resolvers and as a result it can serve queries faster and maintain better service even during Denial of Service attacks. Additional services are offered such as protection against known phishing-sites, typos, access statistics and shortcuts. OpenDNS is profitable by acting as a search engine which is displayed when the name of a website can not be resolved.

The model raises many privacy concerns since OpenDNS monitors and stores all internet access of its clients and could become subject of domestic law. However it does demonstrate that resolvers, be it the ISP or a third-party, can play an active role in protecting clients from any abuse of DNS.

Another example of a DNS resolver which actively tries to protect clients against attacks on DNS is CoDoNS [5]. CoDoNS is based on the Beehive peer-to-peer replication system which can achieve  $O(1)$  message complexity for queries following a power law distribution. CoDoNS is very resilient to (distributed) denial of service attacks since there is no central bottleneck and popular names will be heavily replicated (which does not directly protect against DoS, but does allow the system to maintain normal operation). Although CoDoNS provides internal security mechanisms it is still vulnerable to attacks on legacy DNS since that is where resource records are obtained. However name owners can also purchase a CoDoNS certificate to have fully secure name

resolution without having to provide a name server. CoDoNS does not match the administrative model of DNS very well which raises many questions regarding privacy, (forward) compatibility, cost of changes (can we add a million subdomains?), scalability and incentives, but it is a prime example of how DNS clients can be protected from the resolving side.

## 5 Off-line intrusion detection

In the future active providers like OpenDNS that maintain large caches and logs could go beyond static protection mechanisms. The knowledge that DNS is inherently insecure should trigger service providers to protect their clients better. Providers can use different techniques to detect and handle anomalies in DNS data they receive. They can do historical analysis on the changes and gain higher confidence by querying multiple times from different name servers.

If the intrusion detection mechanism does not find the changes sufficiently trustworthy the resolver can even check whether the IP addresses still show the same behaviour. For example it can check whether the same applications are running (Web server, FTP server, etc) and it can even look at individual web documents. Obviously many of these things will frequently change simultaneously on server migrations or website overhauls (rather than simple IP changes), but it is just one of the factors the intrusion detector can take into account when deciding whether to accept the changes. Clearly these systems need to keep their false positive rate at an absolute minimum as a failure of DNS can seriously harm the user experience of clients and profit of the service provider.

The verifications outlined above are too expensive to do in an active manner. If we have a cache miss we have little choice but to return the result immediately outside simple checks (blacklists). If the information is false this could potentially damage the client who did the request, but the benefit for the attacker of misdirecting a single client is small. What the system should take care of is that the incorrect information is not going to be in the cache used by everyone. Cache updates should therefore be done independent from requests. Not only does this give us the time to use

advanced intrusion detection techniques, it also gives us protection against birthday attacks since there is no way to trigger a cache update, for optimal protection cache updates should follow an unpredictable pattern (Poisson).

When an intrusion has been detected with some level of certainty several actions can be taken. If the resolver has detected an intrusion with very high certainty it can either act as if the name does not exist or redirect the name to a local server which gives an application specific error (e.g. webpage, e-mail, FTP message), this already happens for blacklisted phishing websites in OpenDNS. If not quite so certain it can display a warning and give the user an option to continue. In other cases a resolver can decide to return an older IP address or an IP address with a higher level of certainty to the client. It should of course take care to see whether that IP is still useful.

In short, resolvers can use many different techniques to provide their own clients which a much higher level of protection, rather than the passive role they play now.

## 6 Conclusions

DNSSEC has been long been seen as the absolute security mechanism for DNS. Today we can conclude that in terms of adoption the top-down approach of securing DNS has all but failed. In this paper we discussed some of the inherent security problems of DNS and the need for a more active role of the resolving servers in protecting clients from attacks on DNS.

Service providers have so far been too passive in protecting their users against attacks, but a new generation of DNS resolvers is starting to appear. OpenDNS provides users and organizations a whole new level of control for resolving domain names, but its protection against attacks is still limited to manual blacklisting. In the future active DNS providers could extend their protection by employing intrusion detection mechanisms for unusual changes of resource records. Another next generation DNS resolver, CoDoNS, offers strong protection against (distributed) denial of service attacks on the DNS systems and can offer name owners security guaran-

tees for a small fee (independent of legacy DNS). As CoDoNS is an open distributed system it could potentially expand to finally replace DNS with a more secure alternative. Although we do not believe that either OpenDNS or CoDoNS are the absolute answer to the DNS security issues, we do believe that future efforts to secure DNS should approach the problem from the bottom up.

## References

- [1] Opendns (<http://www.opendns.com/>).
- [2] InfoBlox Inc.  
"DNS Survey"  
(<http://dns.measurement-factory.com/surveys/200710.html>).  
*The Measurement Factory*, October 2007.
- [3] Amit Klein.  
"BIND 9 DNS Cache Poisoning"  
(<http://www.trusteer.com/docs/bind9dns.html>).  
*Trusteer*, July 2007.
- [4] Vagner Sacramento.  
"Vulnerability in the sending requests control of Bind versions 4 and 8 allows DNS spoofing".  
November 2002.
- [5] Venugopalan Ramasubramanian; Emin Gun Sirer.  
"The Design and Implementation of a Next Generation Name Service for the Internet".  
*In proceedings of SIGCOMM 2004*, August 2004.
- [6] Venugopalan Ramasubramanian; Emin Gun Sirer.  
"Perils of Transitive Trust in the Domain Name System".  
*In proceedings of the Internet Measurement Conference*, October 2005.