The Domain Name System (DNS): Security challenges and improvements

Richard John Matthew Agar

Technical Report
RHUL–MA–2010–03
31st March 2010

Department of Mathematics
Royal Holloway, University of London
Egham, Surrey TW20 0EX, England

http://www.rhul.ac.uk/mathematics/techreports

The Domain Name System (DNS): Security challenges and improvements

Name: Agar, Richard John Matthew
Student Number: 100622219

Supervisor: Kenneth Paterson

Submitted as part of the requirements for the award of the MSc in Information Security at Royal Holloway, University of London.

I declare that this assignment is all my own work and that I have acknowledged all quotations from the published or unpublished works of other people. I declare that I have also read the statements on plagiarism in section 1 of the regulations governing examination and assessment offences and in accordance with it I submit this project report as my own work.

Signature:

Date: 4th September 2009

ACKNOWLEDGMENTS

I would like to thank Professor Kenneth Paterson for his support, enthusiasm and encouragement with the research for this project and preparation of the report. I would also like to express my appreciation to the remainder of the academic staff at the Information Security Group at Royal Holloway who have lectured over the past year, all of you have helped indirectly by providing valuable knowledge and understanding that has helped me with this project and report. Finally I'd like to thank Elizabeth for her help, understanding and humour during this process.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS AND ACRONYMS

A	DNS IPv4 address resource record.
AAAA	DNS IPv6 address resource record.
AD	DNSSEC Authentic Data, resource records that have been validated by a DNSSEC resolver.
BCP	Business Continuity Planning.
BGP	Border Gateway Protocol, a routing protocol.
BIND	Berkley Internet Name Domain, an open source DNS server.
CA	Certification Authority.
ccTLD	country code Top Level Domain.
CD	Checking Disabled, the CD bit indicates that the resolver is capable of validating DNSSEC resource records.
CNAME	DNS Canonical Name resource record.
DLV	DNSSEC Lookaside Validation.
DDOS	Distributed DOS.
Dig	Domain Information Groper.
Djbdns	Daniel Bernstein's DNS server.
DNSCurve	Link level protection for DNS.
DNSKEY	DNSSEC public key record.
DNSSEC	Domain Name System Security Extensions.
DO	DNSSEC OK.
DOS	Denial Of Service.
DS	DNSSEC Delegation Signer resource record.
ECC	Elliptic Curve Cryptography.
EDNS0	Extension Mechanisms for DNS.
FQDN	Fully Qualified Domain Name.
HINFO	DNS Host Info resource record.
ICANN	Internet Corporation for Assigned Names and Numbers.
ISC	Internet Systems Consortium.
KSK	DNSSEC Key Signing Key.
MX	DNS Mail Exchanger resource record.
NIST	National Institute of Standards and Technology (US).
NS	DNS Nameserver.
NSEC	DNSSEC Next Secure resource record.
NSEC3	DNSSEC Next Secure 3 resource record.
NTIA	National Telecommunications and Information Administration (US).

OpenDNS	A DNS service provider.
OPT	Resource record used by EDNS0 to provide additional fields.
PTR	Pointer to another part of the domain name space (used for mapping of IP addresses to names - reverse DNS).
RFC	Request For Comments, Internet Engineering Task Force document.
RR	DNS Resource Records.
RRSet	DNSSEC resource record set, group of resource records of same type and host.
RRSIG	DNSSEC signature of RRSet.
SOA	DNS Start of Authority.
SRI	Stanford Research Institute.
TLD	Top Level Domain.
TSIG	Transaction Signature.
TTL	Time To Live.
TXT	DNS Text resource record.
UltraDNS	Organisation managing some of the uk authoritative nameservers.
WANem	Wide Area Network Emulation software tool.
WPAD	Web Proxy Auto Discovery.
ZSK	DNSSEC Zone Signing Key.

Throughout the main body of the document DNS names are shown in italics.

1. EXECUTIVE SUMMARY

An analogy that is often used for the Domain Name System (DNS) is that it is the phonebook for the Internet. The DNS provides the mapping between the names that we use to identify applications, websites and e-mail recipients etc and the numerical addresses that are used by the components in networks. If an attacker can poison the DNS (i.e. make it return invalid information) then the user may unknowingly connect to the attacker's service, rather than the correct one. The user may then be exposed to confidentiality, integrity and availability issues.

In July 2008, security researcher Dan Kaminsky disclosed a significant issue in DNS that allowed an attacker to be able to poison the DNS with information of the attacker's choosing. Whilst this had always been possible, it was believed there was a narrow window of opportunity to attack, and that during that narrow window the possibility of a successful attack was very low. Dan Kaminsky showed that this was not the case; this report includes an analysis that shows an attack of 259 seconds duration has a 75% chance of success against vulnerable servers.

Weaknesses exist in client and server applications and operating systems, their configuration, procedures, people and the DNS protocol that allow a range of different factors that may cause confidentiality, integrity and availability issues to users and applications that rely on the DNS. This report provides an overview of related vulnerabilities and attacks, two of which are investigated in more detail; cache poisoning and amplification attacks (a type of denial of service attack).

DNS poisoning attacks can easily be conducted against servers not patched against the Kaminsky vulnerability. A tactical solution has been provided that makes these attacks harder, but still possible. A strategic solution is needed that provides a cryptographic response to cache poisoning. This report looks at two possible solutions to cache poisoning attacks: DNSSEC and DNSCurve, although neither provides the perfect solution.

The DNS is vulnerable to use in amplification attacks. The DNS can be abused to generate multi-gigabit attacks that can be used against any target to prevent legitimate use of resources at the target. Although DNSSEC provides protection against DNS poisoning attacks it does make amplification attacks easier.

2. INTRODUCTION

Computers communicate over a network using numerical addresses to identify endpoints and route data. Whilst computers find this 'easy', humans do not; we find it easier to remember names, such as *rhul.ac.uk*. In the early days of the ARPAnet (the network that eventually became the Internet) there was a small number of computers communicating. ARPAnet used a single file (hosts.txt) to map names to addresses, the Stanford Research Institute (SRI) managed the file; if any changes were required by users of the ARPAnet then these changes were submitted to SRI who updated the file once or twice a week. Any computer communicating on the network needed an up-to-date copy of the file, requiring an FTP download from SRI [LIU06].

As the number of hosts on the ARPAnet increased, the manual administration of the hosts.txt file became less manageable. More and more hosts were downloading the hosts.txt file causing resource issues at SRI. Additionally there was no authority responsible for assigning names, meaning that anyone could use any name, which caused name collisions that would disrupt traffic for both users of that name. Timely distribution of the new hosts.txt file became more difficult as the number of hosts grew, leading to inconsistencies in the files being used [LIU06].

Clearly a better system was needed at some point, and the DNS was developed to provide a more manageable solution to mapping names to addresses; however the hosts.txt file is still present in Operating Systems (OSs) today to allow local configuration of name to IP mapping.

The DNS is a distributed, scalable database that enables the Internet to operate as it does today. Websites, e-mail, instant messaging, distributed applications, e-commerce and many more applications depend on the DNS to be able to communicate. In our daily lives many of us have become dependent on these applications, and are therefore dependent on the DNS. It is not an overstatement to say that if the DNS were to fail, then the Internet would cease to operate as it does today.

3. DNS OVERVIEW

3.1. DNS architecture

This section is intended to provide a brief introduction to the DNS and appropriate background for the discussion later in this project report. The reader may wish to refer to other texts that cover the DNS in more detail, some of which are listed in the bibliography.

This section describes the architecture of the DNS; section 3.2 puts that together in the context of a DNS query and response walkthrough.

3.1.1. DNS domains

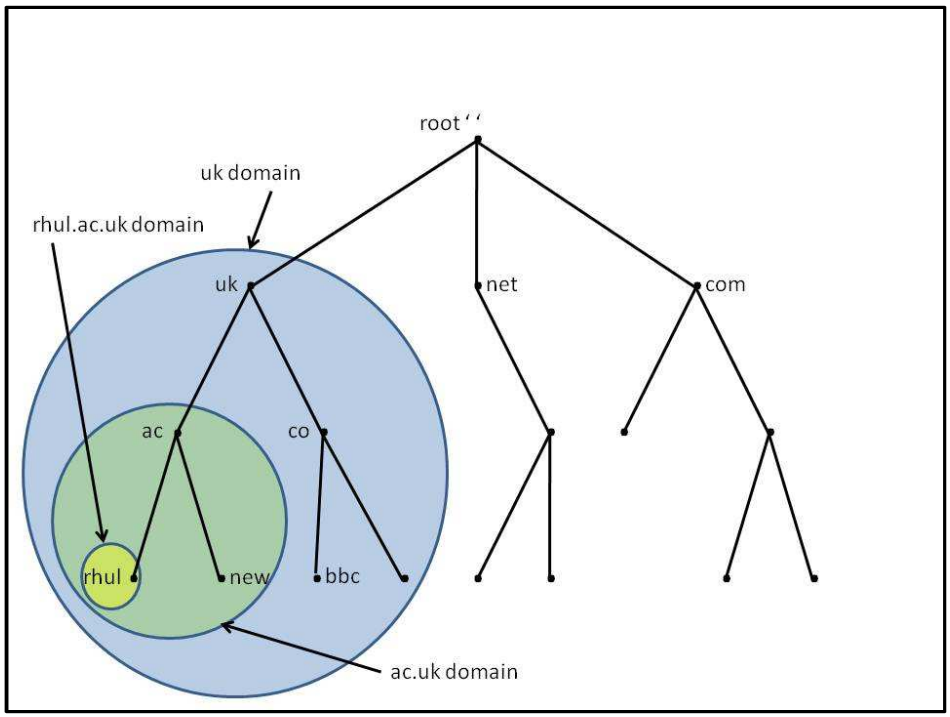

Figure 1 – DNS domains

The DNS is organised as an inverted tree (see figure 1). The root is at the top and has a null label; additional labels are at every point in the hierarchy (labels are used to name points on the tree). A domain is a point on the tree and is named by concatenating the different labels (separated by a '.') towards the root, for example *rhul.ac.uk*. Every **Fully Qualified Domain Name** (FQDN) is the complete name comprising all labels and is unique in the DNS.

Figure 1 shows an example of the domain and sub domain structure of the DNS. The **Top Level Domains** (TLDs) are classified as either generic TLDs (gTLDs) such as *com* and *net*, or country code TLDs (ccTLDs) such as *uk* or *fr*. Below the top level domains we have second level domains such as *ac.uk* or *co.uk*; these second level domains are **sub (or child) domains** of their parent domains. A **parent domain** contains all of the sub domains beneath it in the hierarchy, as illustrated in figure 1, with the *uk* domain (shown in blue) containing the *ac.uk* domain (shown in green) containing the *rhul.ac.uk* domain (shown in yellow).

Not all domains contain sub domains, although all domains contain **Resource Records** (RRs); these RRs contain the information in the DNS and are discussed in section 3.1.3.

3.1.2. DNS zones

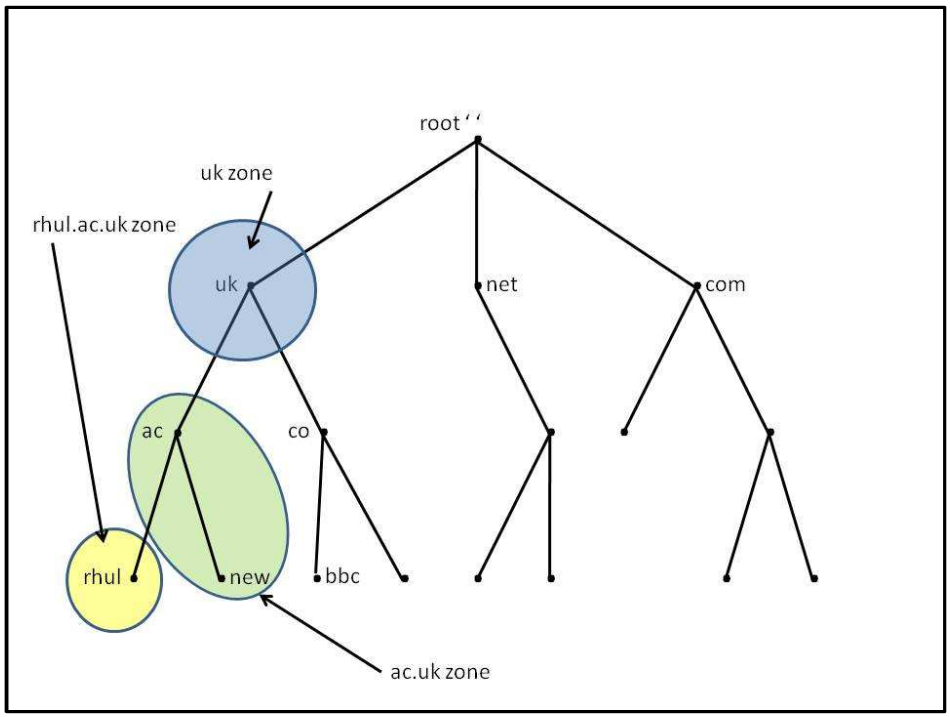

Figure 2 – DNS zones

In addition to domains, the DNS uses the concept of **zones**. Whilst a domain defines a position in the hierarchy, a zone defines an administrative boundary and contains the RRs in the DNS. RRs are stored in **zone files** (the files that store the RRs in a zone, see Appendix A for an example) on an **authoritative nameserver** (**nameservers** are programs that process queries and provide a response, an authoritative nameserver is a server that serves the authoritative RRs from a zone file

configured by the zone administrator) and are used to answer queries and allow navigation within the DNS. The left hand side of figure 2 illustrates three zones; *rhul.ac.uk* (shown in yellow), *ac.uk* (shown in green) and *uk* (shown in blue).

Figure 2 shows that the *ac.uk* zone contains the *ac.uk* domain and the *new.ac.uk* domain, this means that the *ac.uk* zone holds administrative responsibility for both domains.

Figure 2 shows the *rhul.ac.uk* domain in a separate zone to the *ac.uk* domain; this is possible as *ac.uk* has delegated administrative responsibility to *rhul.ac.uk* (referred to as **delegation**). Once delegated the *ac.uk* zone will no longer contain RRs for the *rhul.ac.uk* domain, it will instead contain the RRs of the authoritative nameservers for *rhul.ac.uk*. The authoritative nameservers for the *rhul.ac.uk* zone will then need to contain the RRs for the *rhul.ac.uk* domain.

Delegation enables local control in the DNS allowing the administrators of a zone to manage their own RRs. It may seem natural for a 1 to 1 domain to zone relationship, however this may not be desirable. A new or small organisation may not have the required resources to administer its own zone, or an organisation requiring multiple domains may not wish to delegate administration.

3.1.3. Resource Records

A zone contains the RRs for the domain that it is authoritative for. RFC 1035 [MOC87b] defines the following resource record types:

A	Address, the A record is an IP v4 host address.
CNAME	Canonical name of an alias, used to reduce administration. Where there is more than one hostname for a host the CNAME RR is used to point to the definitive hostname. If IP address changes are needed they are only needed to the single A record corresponding to the CNAME.
HINFO	Host Info, identifies the CPU and OS used by a host. As this may be used by an attacker conducting reconnaissance it should be used with care!
MX	Mail Exchanger, used for delivering e-mail.
NS	Nameserver, Authoritative name servers.
PTR	Pointer to another part of the domain name space (used for reverse DNS - mapping of IP addresses to names).

SOA Start of Authority, the SOA record is mandatory for a zone and includes:
The master nameserver for the zone.
The zone administrators e-mail address.
Time To Live (TTL) value (validity period in seconds).
Serial number – incremented when the zone data has been changed.

TXT Text, used for arbitrary data.

RFC 1886 [THO95] defined changes to the DNS for IPv6 and included:

AAAA Host address or addresses (IP v6).

The TTL value can be included in all RRs, but is mandatory in the SOA RR. If missing in an RR the value in the SOA is used. The TTL value is used to inform **caching nameservers** (also known as **caches**, are nameservers that serve a copy of the RRs from memory, see 3.1.4) how long RRs can be cached before they should be discarded. The TTL is a 32 bit number, with the value indicating the number of seconds to cache (giving a range from 0 [meaning don't cache] to over 136 years) [MOC87a].

3.1.4. Resolvers and Nameservers

Resolvers are programs that are used to query for RRs. OSs, applications and nameservers contain resolver code. In the case of a host machine a resolver is often referred to as a **stub resolver**.

A nameserver can be an authoritative nameserver, or a caching nameserver. A caching nameserver is a server that accepts queries from resolvers, obtains the response from other nameservers (typically authoritative nameservers), then caches and serves those responses to the resolvers (typically stub resolvers). In this document we will use the term **DNS server** to refer to a nameserver that is not authoritative.

DNS queries can be **iterative queries** or **recursive queries**. An iterative query expects one of three responses from the nameserver; the RRs containing the answer to the query, the RRs of the authoritative DNS servers for the next domain down the DNS tree (allowing the resolver to query that server) or an error. A recursive query expects one of two responses from the nameserver; the

RRs containing the answer to the query, or an error. **Open recursive servers** are servers on the Internet that will respond to recursive queries from any host.

We will use both iterative and recursive nameservers in the DNS query walkthrough in 3.2.

3.1.5. Availability and Performance

Availability and performance are achieved by replication (the use of multiple nameservers) and caching.

Stub resolvers allow multiple DNS servers to be configured for availability and performance. The resolver regularly monitors the configured servers and determines the best server to use.

DNS servers are provided with the IP addresses of the **DNS root servers**; the root servers contain the **root zone file**, this contains the RRs of the authoritative nameservers of the TLDs. There are 13 IP addresses for the root servers (13 is the maximum number of RRs that will fit in a RFC 1035 [MOC87b] compliant DNS message); however there are many more root servers than this. **Anycast** is used to replicate root servers across the Internet. With Anycast, a single IP address is associated with multiple hosts (BGP is used to advertise the different routes, and route traffic to the best destination). Anycast increases the number of servers per IP address, for example the Internet Systems Consortium (ISC) operates the F root server (the root servers are named A through to M) in 46 separate locations across the globe using Anycast [ISC09a]. Some TLD and recursive servers also use Anycast, for example the UltraDNS managed *uk* authoritative servers and the OpenDNS servers (a DNS service provider).

RFC 1034 [MOC87a] requires that any zone be available from at least 2 nameservers (1 master and 1 or more slaves); this provides both resiliency and load sharing. Replication of RRs is achieved through the use of **zone transfers** to replicate the zone files from the master nameserver (where the zone file is created and updated) to the slave nameserver(s) for a given zone.

Resolvers and servers improve performance by caching RRs they have learned for the period stated in the TTL. If a server receives a subsequent query for information in the cache within the TTL then it is served directly, rather than triggering a request to another server. The administrator of a zone needs to find the correct balance between a low TTL (changes in the zone are replicated

more quickly, although there is more traffic to the authoritative nameserver) and a high TTL (changes in the zone are replicated more slowly, with less traffic to the authoritative nameserver).

3.2. DNS query and response

3.2.1. DNS protocol

RFC 1035 [MOC87b] allows the use of either UDP or TCP for DNS transport, it recommends the use of UDP for queries and TCP for zone transfers. This seems reasonable as a UDP based DNS query requires only two datagrams – one for the query and one for the response, whilst a TCP based DNS query requires seven datagrams, five to set up and tear down the session and two for the query and response. Zone transfers however tend to be large and infrequent, and benefit from the reliability of TCP.

RFC 1035 [MOC87b] requires that the DNS message header includes a **transaction ID** and defines it as:

> *'A 16 bit identifier assigned by the program that generates any kind of query. This identifier is copied to the corresponding reply and can be used by the requester to match up replies to outstanding queries'.*

In addition to providing answers to queries, DNS provides two ways to allow responding servers to include extra information; these are authoritative nameservers and **additional records**. The intent is that this extra information will assist the querying server and is used for supplying the names of authoritative nameservers and corresponding **glue records**. Glue records are records that provide the IP addresses of the nameservers in a query response (this is explained in 3.2.5).

3.2.2. DNS request walkthrough

This section is a walkthrough of processing of a DNS query from a stub resolver, through its configured DNS server, and the iterative queries the DNS server sends to enable it to respond to the original query from the stub resolver. The queries and responses are shown in figure 3.

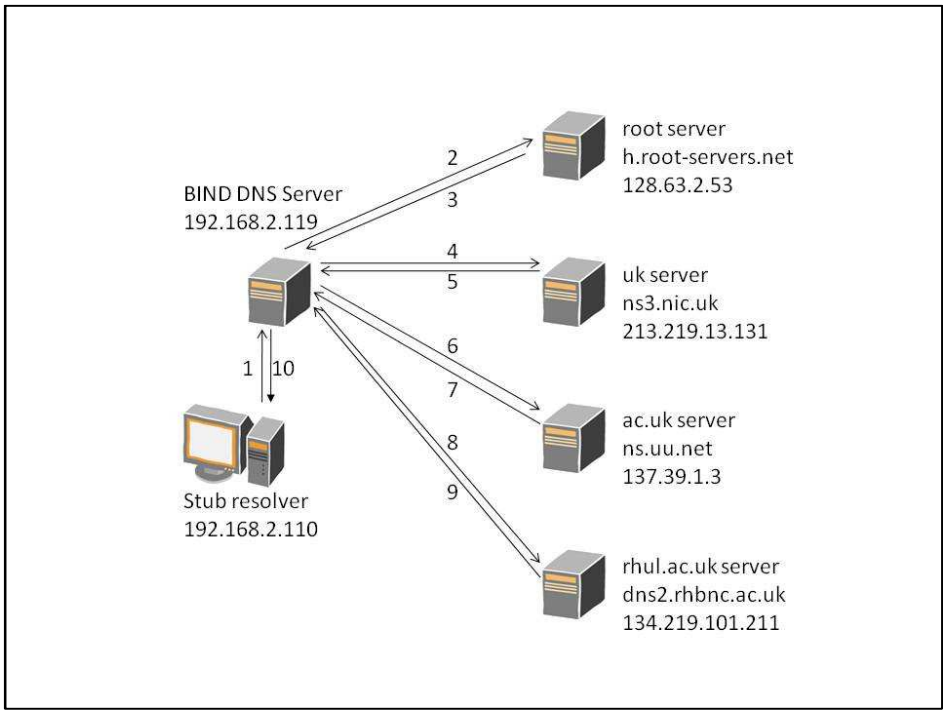

Figure 3 – DNS walkthrough

In this example the ICMP PING command was used on the stub resolver to trigger a query for *www.rhul.ac.uk*. The process is as follows:

- An ICMP PING command for *www.rhul.ac.uk* is issued at the command prompt and calls the resolving function of the OS.
- The stub resolver checks in its local hosts.txt file to see if there is an entry for *www.rhul.ac.uk*. If there is, then this will be used, the resolution process ends and the IP address is returned to the calling application.
- The stub resolver looks in the local cache (if supported) to see if there is an entry there for *www.rhul.ac.uk*. If there is then this will be used, the resolution process ends and the IP address is returned to the calling application.
- The stub resolver sends a recursive DNS query for the A record for *www.rhul.ac.uk* to its DNS server (step 1 in figure 3) and waits for the response.
- The DNS server will check its cache to see if it has the A record for *www.rhul.ac.uk*. If it does and it is within TTL, it will return that record to the stub resolver and the resolution process ends; if not, it will start sending iterative queries to obtain the answer.
- The DNS server will look in its cache for authoritative servers as near to *www.rhul.ac.uk* as possible and send a query to one of those servers. In this example the cache of the DNS

server is empty, so it selects a root server (*h.root-servers.net* in this case) and sends an iterative query to it (step 2 in figure 3).

- The root servers are authoritative for all top level domains, so respond to a query for the A record for *www.rhul.ac.uk* with the records for the authoritative nameservers for *uk* (step 3 in figure 3). This includes the names of the nameservers and glue records. The DNS server caches these records for the duration of the TTL (2 days in this case) so they can be used for future queries and/or responses.
- The DNS server selects one of the *uk* authoritative servers (*ns3.nic.uk* in this case) and sends an iterative query to it (step 4 in figure 3).
- The *uk* authoritative server responds to the query for the A record for *www.rhul.ac.uk* with the records for the authoritative nameservers for *ac.uk* (step 5 in figure 3). This does not include glue records, so the DNS server puts the query for *www.rhul.ac.uk* on hold whilst it resolves the authoritative nameserver records (this is not included in this walkthrough for brevity). The DNS server caches these records for the duration of the TTL (2 days in this case) so they can be used for future queries and/or responses.
- The DNS server selects one of the *ac.uk* authoritative servers (*ns.uu.net* in this case) and sends an iterative query to it (step 6 in figure 3).
- The *ac.uk* authoritative server responds to the query for the A record for *www.rhul.ac.uk* with the records for the authoritative nameservers for *rhul.ac.uk* (step 7 in figure 3). This includes the name of the nameservers and glue records. The DNS server caches these records for the duration of the TTL (1 day in this case) so they can be used for future queries and responses.
- The DNS server selects one of the *rhul.ac.uk* authoritative servers (*dns2.rhbnc.ac.uk* in this case) and sends an iterative query to it (step 8 in figure 3).
- The *rhul.ac.uk* authoritative server responds to the query for the A record for *www.rhul.ac.uk* with the CNAME record for *www.rhul.ac.uk* (the RR contains *nts22-fe.rhul.ac.uk*) and the A record for *nts22-fe.rhul.ac.uk* (step 9 in figure 3). The DNS server caches these records for the duration of the TTL (1 day in this case) so they can be used for future responses.
- The DNS server can now respond to the original query from the stub resolver for the A record for *www.rhul.ac.uk* with the CNAME record for *www.rhul.ac.uk* '*nts22-fe.rhul.ac.uk*' and the A record for *nts22-fe.rhul.ac.uk* (step 10 in figure 3).

This seems like a long process, and (in this case) it is. The **dig** (dig, or Domain Information Groper is a command line tool for obtaining DNS information) output shows a query time of 734 ms when using a server with an empty cache, however remember that the DNS server has learned

of many authoritative nameservers during this process and has cached their records in addition to caching the A and CNAME records for *www.rhul.ac.uk*. The next time (within the TTL) any client of the DNS server sends a query for the A record for www.rhul.ac.uk the DNS server will be able to respond directly (the dig output shows a query time of 1 ms). Any client of the DNS server sending a query for another host at *rhul.ac.uk* will only have to wait whilst the DNS server queries one of the *rhul.ac.uk* authoritative servers (as is the case for the other authoritative nameservers that the DNS server has learned).

3.2.3. Structure of a DNS query and response

We will now look at the structure of a DNS query from the stub resolver (dig 9.4.2 on Fedora 7) and a DNS response from the DNS server (**Berkley Internet Name Domain (BIND**) 9.4.0-6 on Fedora 7), again for *www.rhul.ac.uk*. As the DNS server has cached records we will look at a single query and response. The query was captured using the Wireshark capture tool and is shown in figure 4.

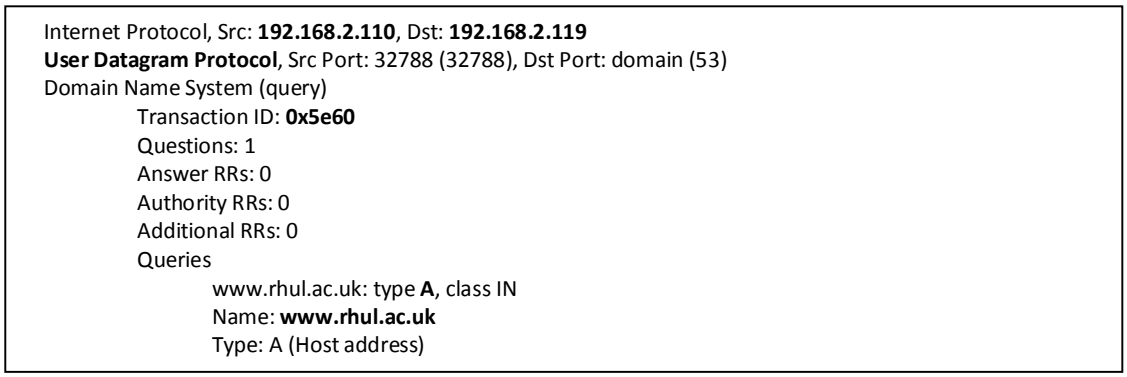

Figure 4 – DNS query

We can see from this capture that a query was sent from the stub resolver at 192.168.2.110 to the DNS server at 192.168.2.119 in a UDP packet requesting the A record for *www.rhul.ac.uk*. It included a transaction ID of 0x5e60.

The response was captured using the Wireshark capture tool and is shown in figure 5.

Internet Protocol, Src: **192.168.2.119**, Dst: **192.168.2.110**
User Datagram Protocol, Src Port: domain (53), Dst Port: 32788 (32788)
Domain Name System (response)
Transaction ID: **0x5e60**
Questions: 1
Answer RRs: 2
Authority RRs: 3
Additional RRs: 3
Queries
www.rhul.ac.uk: type A, class IN
Name: www.rhul.ac.uk
Type: A (Host address)
Answers
www.rhul.ac.uk: type CNAME, class IN, cname nts22-fe.rhul.ac.uk
Name: www.rhul.ac.uk
Type: CNAME (Canonical name for an alias)
Time to live: 23 hours, 56 minutes, 3 seconds
Primary name: nts22-fe.rhul.ac.uk
nts22-fe.rhul.ac.uk: type A, class IN, addr 134.219.202.248
Name: nts22-fe.rhul.ac.uk
Type: A (Host address)
Time to live: 23 hours, 56 minutes, 3 seconds
Addr: 134.219.202.248
Authoritative nameservers
rhul.ac.uk: type NS, class IN, ns dns2.rhul.ac.uk
Name: rhul.ac.uk
Type: NS (Authoritative name server)
Time to live: 23 hours, 56 minutes, 3 seconds
Name server: dns2.rhul.ac.uk
rhul.ac.uk: type NS, class IN, ns dns3.rhul.ac.uk
Name: rhul.ac.uk
Type: NS (Authoritative name server)
Time to live: 23 hours, 56 minutes, 3 seconds
Name server: dns3.rhul.ac.uk
rhul.ac.uk: type NS, class IN, ns auth1.dns.rhul.ac.uk
Name: rhul.ac.uk
Type: NS (Authoritative name server)
Time to live: 23 hours, 56 minutes, 3 seconds
Name server: auth1.dns.rhul.ac.uk
Additional records
dns3.rhul.ac.uk: type A, class IN, addr 134.219.101.212
Name: dns3.rhul.ac.uk
Type: A (Host address)
Time to live: 23 hours, 56 minutes, 3 seconds
Addr: 134.219.101.212
auth1.dns.rhul.ac.uk: type A, class IN, addr 134.219.204.48
Name: auth1.dns.rhul.ac.uk
Type: A (Host address)
Time to live: 23 hours, 56 minutes, 3 seconds
Addr: 134.219.204.48
dns2.rhul.ac.uk: type A, class IN, addr 134.219.101.211
Name: dns2.rhul.ac.uk
Type: A (Host address)
Time to live: 23 hours, 56 minutes, 3 seconds
Addr: 134.219.101.211

Figure 5 – DNS response

We can see from this capture that a response was sent from the DNS server at 192.168.2.119 to the stub resolver at 192.168.2.110 in a UDP packet supplying the answers. The response also includes a transaction ID of 0x5e60 and authoritative nameservers RRs with corresponding additional (glue) RRs. All the supplied RRs have a TTL.

3.2.4. Validating a query response

When a response is received it needs to be validated; the following checks are performed in order [MOC87b]:

- The IP address and UDP ports match the query.
- The transaction ID matches the query.
- The query section in the query matches the query section in the response.
- Any additional RRs are in the same domain as the original query (this is referred to as being '**in-bailiwick'**).

The first response that passes these checks is deemed valid and accepted and any RRs are added to the cache. In this example all are correct, so the response is valid.

3.2.5. Why glue records are important

Glue records are required if trying to contact authoritative nameservers to resolve a name when the authoritative nameservers are in the same domain as the target name. For example if a DNS server needs to resolve the hostname *www.example.com* and the *example.com* nameservers are *ns1.example.com* and *ns2.example.com*.

Without glue records the following will happen:

- The DNS server will ask the root servers for *www.example.com*.
- The root servers will tell the DNS server to ask the *com* servers.
- The DNS server will ask the *com* servers for *www.example.com*.
- The *com* servers will tell the DNS server to ask one of the *example.com* authoritative nameservers (*ns1.example.com* or *ns2.example.com*).
- The DNS server will ask the *com* servers for (say) *ns2.example.com*.

- The *com* servers will tell the DNS server to ask one of the *example.com* authoritative nameservers (*ns1.example.com* or *ns2.example.com*).

The DNS Server is not able to resolve *www.example.com* as it has no way to contact the *example.com* authoritative nameservers.

With glue records the following will happen:

- The DNS server will ask the root servers for *www.example.com*.
- The root servers will tell the DNS server to ask the *com* servers.
- The DNS server will ask the *com* servers for *www.example.com*.
- The *com* servers will tell the DNS server to ask one of the *example.com* authoritative nameservers (*ns1.example.com* or *ns2.example.com*) and supply the glue records.
- The DNS server will ask (say) *ns2.example.com* for *www.example.com*.
- The *ns2.example.com* authoritative nameserver will reply with the requested RRs.

The DNS Server now has the record it needs.

An alternative approach would be to host the *example.com* authoritative nameservers on a different domain, or even for changes to the protocol that would allow the NS records to be populated with IP addresses, rather than names.

4. OVERVIEW OF DNS SECURITY ISSUES

This section is intended to provide a brief overview of DNS security issues, sections 5 and 7 look into more detail at two specific DNS security issues: **cache poisoning** (introducing false information into a DNS cache) and **amplification attacks** (a type of Denial Of Service (DOS) attack using the DNS).

4.1. Services required from the DNS

DNS does not provide the following services:

- Data Origin Authentication.
- Data Integrity.
- **Authenticated Denial of Existence** (the ability to prove that a host, or domain does not exist).
- Confidentiality.

These services were not considered design goals of DNS [MOC87a]; this is understandable as at the time the Internet was in its infancy and was only used by a relatively small group of trusted users. The threats that we are aware of today simply did not exist at that time. The absence of these services enables a number of attacks, either against the DNS, or using DNS to assist the attack.

The lack of data origin authentication and the use of UDP enable an attacker to create spoofed DNS query responses. Without data integrity an attacker can modify DNS query responses and the changes will not be detected. The lack of authenticated denial of existence enables an attacker to spoof, or modify DNS query responses to incorrectly indicate that a host does not exist.

If a resolver accepts any of these responses, then the DNS will become poisoned and any relying party may use an incorrect IP address to connect to a service, or may not be able to connect to the service.

As privacy becomes more important and the use of encryption for bulk traffic on the internet becomes more commonplace DNS traffic could be monitored to identify web use, confidentiality would restrict this monitoring to the DNS service provider.

4.2. DNS Traffic redirection

DNS **redirection** attacks rely on the attacker being able to poison the DNS with false information that an application then relies upon. If the attack goes unnoticed, the false DNS information will remain for the duration of the TTL, or until the cache is flushed. Types of redirection attack include the following:

- E-mail redirection.
- Nameserver redirection.
- WWW redirection.

Following these three redirection attacks we will look at mitigation options in 4.2.4.

4.2.1. E-mail redirection

With e-mail redirection, the victim (or victim's SMTP server) receives a poisoned MX RR and connects to an IP address of the attacker's choosing. The attacker's server may take a copy of the e-mail and then forward the original e-mail (possibly after making modifications to the contents or adding malware) onto the intended destination. As the e-mail is delivered to the destination intended by the sender; this attack may go un-noticed for a long period of time. The vast majority of e-mail is neither encrypted nor integrity protected, e-mail is regularly used for sending confidential and/or sensitive information, meaning this attack could have a significant impact on the sender and/or the receiver.

4.2.2. Nameserver redirection

With nameserver redirection, a DNS server receives a poisoned NS RR. If this attack is successful, then all subsequent queries for any host in that domain (not cached and within TTL) will be sent to nameservers of the attacker's choosing. If the attacker controls those nameservers he can choose how to respond to queries for any host in the entire DNS domain. For example, he may wish to redirect e-mail only between two organisations. To enable this he could respond with a false MX RR to queries from the IP addresses of a chosen organisation, whilst all other queries

could receive the correct MX RR. This would benefit the attacker as he would only receive e-mail from the target organisation, and the attack would be less likely to be discovered.

4.2.3. WWW redirection

With WWW redirection, the victim receives a poisoned A RR and connects to an IP address of the attacker's choosing. This IP address will be hosting a spoofed website, and could be part of a Pharming attack to steal users' credentials, or other confidential information. Alternatively, it could make it appear to visitors of the spoofed site that the real site had been hacked and defaced (as was the case for RSA Security in 2000).

Pharming attacks are used as part of identity theft to fool users into divulging credentials. For example, the victim of a DNS poisoning attack may be a user of online banking. If he connects to what he believes to be his bank's website and does not check for the presence of an SSL encrypted session, and then enters his credentials, the attacker can steal those credentials. Even users of one-time authentication could be victims if the attacker's spoofed server acts as a proxy during the authentication phase; once the user has authenticated, the attacker has access to the victim's account(s).

4.2.4. Mitigating DNS traffic redirection

As these redirection attacks rely on the ability to poison a DNS RR, any protection that mitigates poisoning will mitigate these attacks. These include:

- Disable Open Recursive DNS Servers.
- DNSCurve.
- DNSSEC.
- IP spoofing prevention.
- Secure client.
- Secure server.
- Training and awareness.

Disable Open Recursive DNS Servers

Open Recursive DNS servers are used in both cache poisoning and DOS attacks. Configuring these servers to only accept queries from known clients will reduce the likelihood of them being used in these attacks.

DNSCurve

DNSCurve provides link layer confidentiality and integrity protection for DNS messages. The resolver receiving a DNSCurve response can validate that the message has come from the peer server and has not been modified.

Domain Name System Security Extensions (DNSSEC)

DNSSEC provides a way to verify the origin and integrity of a RR using digital signatures on RRs. The resolver receiving a signed response can validate the signature back to a trusted public key. Currently DNSSEC is only deployed in a very small number of domains (.org, .se and .gov TLDs, but not all sub domains). For DNSSEC to be a success this needs to be expanded, both in the domains that sign their RRs, and in resolvers that are able to validate DNSSEC signed RRs.

IP spoofing prevention

Spoofed IP addresses would be significantly reduced if network providers validated that source IP addresses were correct for traffic originating on their network. This would reduce spoofed DNS messages by dropping datagrams with spoofed source addresses at the network ingress. An approach to this is discussed in RFC 2827 [FER00] and updated in RFC 3704 [BAK04]. For spoofing prevention to be successful this needs to be widely implemented.

Secure client

Hosts infected with malware are often used to launch various attacks, including Distributed DOS (DDOS) attacks. Users should follow best practices for OSs and applications with regard to configuration and security updates; additionally any host connecting to the Internet should be appropriately protected (see technical controls in 4.3.3). Many users run with admin rights (the default in Windows XP); this exposes them to additional risk. Accordingly users' machines should run with the minimum required rights to perform their function. If followed, these measures will reduce the likelihood of a host being compromised and being used as part of an attack.

Secure server

Operators of (DNS) servers should ensure the security of their platforms. Following best practices for patching and hardening operating systems, DNS and other applications will reduce the likelihood of a server being compromised and being used in an attack.

Training and awareness

Training and awareness is an important factor for management, system administrators and users. Management need to understand the risks faced by the organisation so they can provide the resources required to ensure an acceptable level of risk. Administrators need to receive training to enable them to consider the security implications of the way they deploy and configure systems so they can apply appropriate safeguards; they also need to keep up-to-date with relevant developments. Users need to be aware of the policies, procedures and guidelines appropriate to their function; they also need to be aware of the way their actions can contribute to the security of their systems, and those of other users, and the effect that this can have on an organisation. The US National Institute of Standards and Technology (NIST) special publication 800-50 [WIL03] describes an approach to training and awareness programmes.

4.3. Client and Server Issues

4.3.1. Client Issues

Malware

Malware can modify the hosts.txt file or locally configured DNS servers. This change in DNS settings causes the client's host to use DNS servers of the attacker's choosing. Some examples are:

- The Trojan.Qhosts virus attacks Windows systems and makes changes to the Windows Registry to alter the DNS servers used for resolution, or to add entries to the hosts.txt file [HOL03].
- Variants of the DNS Changer Trojan have also been used to make changes to the DNS settings on both Apple Mac and Windows from either statically configured DNS settings, or those derived from DHCP [ZDR07] [ZEL06].
- Trojan.Flush.M installs a DHCP server on the compromised host that sends fake DHCP offers to other clients on the local network. The DHCP server offers include a valid IP address and gateway on the local network and DNS servers chosen by the attacker [CHI08].

In all of these attacks, the client will be able to use the network as normal; however, DNS queries will be sent to the attacker's choice of DNS server and may provide false responses.

Vulnerable Application Software

Vulnerable client software can enable attacks against a client's home or small business router [STA06] [OLI08]. These attacks identify the user's router and attempt to connect with default passwords. If successful then changes are made to the router's configuration, such as adding entries to the local hosts.txt file, changing the DNS servers it uses for resolving client queries, enabling remote management or to 'upgrade' the router to code modified by the attacker. This enables the attacker to perform Pharming attacks on all users of the local network (even those that are not vulnerable to the initial attack) and to remotely control the router.

Operating System

Vulnerabilities also exist in the stub resolver code included in OSs.

Various versions of Linux OSs (including Debian 2.2, RedHat 6.2, 7.0, 7.1, 7.2, 7.3 and SuSE 7.0, 7.1, 7.2, 7.3, 8.0) using resolver code derived from BIND have been vulnerable to buffer overflow attacks. These attacks could allow the attacker to execute code at the privilege level of the program making the query [MAN02].

Microsoft OSs (client and server) stub resolver code has also been vulnerable to remote code execution that could allow a remote, unauthenticated attacker to gain complete control of a system [MIC06] [PRU06]. Similarly Apple Mac OS X has been vulnerable to attacks allowing an attacker to execute arbitrary code with root privileges [GIO07].

4.3.2. Client attacks enabled by DNS

This section includes examples of some attacks that are enabled through the use of DNS:

- Application updates.
- Cache snooping.
- DNS rebinding attacks.
- Fast fluxing.
- Web Proxy Auto Discovery (WPAD).

Application updates

Many applications and OSs use DNS names to identify the IP addresses of their update servers. An attacker may wish to introduce false DNS information to prevent an application or OS from

updating itself, or with the intent of providing a false update if the connection or software binary is not authenticated.

If a host is unable to update its security applications and OS patches it will become more vulnerable to other attacks over time. An attacker could use this as part of a blended attack using exploits that the victim would not be vulnerable to if updates had been available.

McAfee Virex 7.0 for Apple MacOS X does not authenticate the connection to its update server, neither does it authenticate the binary. This is discussed a paper by Bellissimo, Burgess and Fu [BEL06], they were able to employ a DNS spoofing attack to use the Virex update manager to run arbitrary code with root privileges.

Cache snooping

Attackers can use this technique to identify if a DNS server holds specific RRs in its cache. For example if a DNS server does not hold the resource record for *windowsupdate.com* just after Microsoft have released patches then this may be an indication to an attacker that vulnerable hosts exist. Cache snooping relies on an organisation allowing its DNS server to be queried by the attacker, unless the attacker is an internal user the DNS server would need to be an open recursive server. An attacker conducting cache snooping simply needs to look at the time difference between 2 queries for the same RR, if the initial query takes significantly longer than subsequent queries this is an indication that the RR was not in the cache.

DNS rebinding

DNS rebinding attacks attempt to bypass a browser's 'same origin' policy to allow an attacker to bypass perimeter security by using the browser as a proxy to launch scans and attacks [JAC09].

The attacker tricks the victim into browsing to a website that he controls (there are various ways to achieve this such as targeted e-mail, social engineering etc); this website hosts some malicious mobile code. When the site's hostname is first resolved it will resolve to the public IP address of the website (the DNS record for that site will have a TTL value of 0 thus preventing caching). The user's browser will download malicious mobile code.

Mobile code is usually isolated in a browser, the intent being it can only communicate with its source server, not any other resources, this is the ‘same origin’ policy and is controlled by hostname.

The attack code now aims to connect to internal resources; however it uses the previously resolved hostname. As the query response was not cached the hostname is again resolved, this time the attacker’s nameserver responds with an IP address chosen by the attacker (this will need to be addresses valid on the internal network of the victim). The attack code can then attempt to connect to services on the target network. As the hostname has not changed the ‘same origin’ policy allows this connection.

Multiple consecutive DNS requests can be sent to allow the attack code to systematically scan the private network. The attacker’s nameserver will alternate between its public IP and the next target IP address, allowing the malicious code to connect back to the attacker for more instructions, and then to other IP’s on the target network.

Fast fluxing

In attempt to reduce the effectiveness of some network security solutions Fast fluxing is used to frequently change the IP address of dubious websites (such as those used in rebinding attacks). Many compromised hosts are used to proxy connections between the requesting client and the attacker’s server. A subset of these compromised hosts is registered with dynamic DNS and a low TTL. At regular intervals the registered IP addresses are changed to a different subset complicating attempts to block access to these sites based on IP addresses alone [HOL08].

Web Proxy Auto Discovery (WPAD)

WPAD is a method used by browsers in a managed endpoint environment to automatically discover proxy servers. Browsers request a DNS lookup of the name *wpad* and download a file that contains configuration parameters of the proxy servers; the browser then uses these proxy servers [LIU09].

If the configuration of the host’s DNS network adapter allows search suffix to be modified (for example allowing the search suffix to be modified or obtained by DHCP – see the Malware section in 4.3.1) then the stub resolver on the host will append the suffix to queries for non qualified domain names, such as wpad.

For example if the attacker adds the search suffix of *example.com* the stub resolver will append *example.com* to *wpad* and send a query for *wpad.example.com*. If the attacker controls the nameservers for *example.com* he can resolve *wpad.example.com* and cause the browser to use his rogue proxy. The rogue proxy will now see all traffic and can redirect traffic for any site chosen by the attacker.

4.3.3. Mitigating client issues

The above client issues are mitigated by the following:

- Disable Open Recursive DNS Servers (see 4.2.4)
- Secure client
- Secure infrastructure
- Secure server
- Secure updates
- Technical controls
- Training and awareness (see 4.2.4)

Secure client

In addition to the mitigations in 4.2.4, the client can be better secured by regular reviews of the installed OS and applications to ensure that patches are applied where appropriate. Additionally the WPAD issue can be mitigated by ensuring that the network adapters don't accept a DHCP issued search suffix. Some browsers use a technique called **DNS pinning** to reduce the effectiveness of DNS rebinding attacks by caching the server IP for a fixed duration, even if the RR TTL is set to 0.

Secure infrastructure

Ensuring that infrastructure, such as home/small routers are appropriately configured, can prevent their DNS configurations from being changed. Users should change passwords from the default, and should disable remote administration. Vendors should aim to provide products to be both useable and secure 'out of the box'.

Secure server

In addition to the mitigations in 4.2.4, the DNS server can be configured to mitigate DNS rebinding attacks by preventing DNS responses from the Internet specifying internal IP addresses.

Web servers need to be regularly patched and configured securely to prevent an attacker using them for attacks such as DNS rebinding.

Secure updates

Where host software requires an update, this should be from a source that can be validated as trusted. To achieve this either the connection must be trusted, for example by using SSL with a valid certificate, or the update code can be signed to provide data origin authentication and detect modification. It should be remembered that these options do not guarantee that it is itself bug free or completely secure, such is the nature of software.

Technical controls

Technical controls provide a combination of prevention and detection of client issues. URL filtering solutions such as web proxies can alert users and administrators about web sites likely to contain malware, or be sources of attacks. Web proxies can also be configured to prevent users for accessing those sites. Some Web proxies can run mobile code in a sandbox to assess if they are safe enough to run on a client host. Content filtering solutions (such as antivirus scanning and intrusion detection/prevention) can be used to detect and/or prevent hosts from connecting to sources of malware, or to identify those that are already infected so remedial actions can be performed.

Some client applications can control the network access available to a host; for example applications exist that allow a host to detect if it is not on its usual network. If so it can apply a more restricted security policy, such as only allowing a VPN connection back to a corporate network and forcing all network traffic to use that connection.

These mitigation options are well suited to a managed environment. There are, however, an increasing number of controls available to the home user that provide additional, and often free, protection such as McAfee Site Advisor and BlueCoat K9 Web Protection.

4.3.4. Server issues

DNS servers at every level of the DNS hierarchy are subject to both availability and integrity issues [CRO04]. Availability includes issues such as software reliability, configuration errors, network problems and failure of the organisation responsible for the server. Integrity includes issues such as modifying, spoofing and poisoning.

Software reliability

A vulnerability was discovered in July 2009 that allows a remote unauthenticated attacker to crash a BIND server (BIND 9.4 before 9.4.3-P3, 9.5 before 9.5.1-P3, and 9.6 before 9.6.1-P1) by sending a specially crafted dynamic update message to a zone master server. Even servers not configured for dynamic updates are vulnerable. The exploit is triggered when a dynamic update message containing a record of type "ANY" is received and where at least one RR set for the fully qualified domain name exists on the server [DOR09] [ISC09b].

Microsoft DNS servers (Windows Server 2000, 2003 and 2008) have suffered from bugs that aided spoofing attacks through predictable transaction IDs making poisoning attacks easier [GIO09] [MIC09].

Configuration errors

In October 2008, the Measurement Factory estimated that there are 4,300,000 open recursive DNS servers on the Internet [TMF08]. These servers may be at increased risk of spoofing and poisoning attacks and can be used in DOS attacks. DOS attacks are discussed further in sections 4.5 and 7.

BIND is sensitive to changes in its configuration and zone files; a simple typing error in the options statement can cause the server to fail to start, or a typing error in the zone statement may cause the zone to fail to load [LIU06].

Zone files contain all the RRs for a zone and could therefore be useful information for an attacker conducting reconnaissance. A zone file may contain information useful in an attack, for example some organisations use 'helpful' names to identify the specific application running a service on a host. If an attacker is able to obtain a copy of the zone file he may be able to use this information to identify vulnerabilities and launch an attack.

Network problems

As DNS relies on 'the network', network problems and attacks may have detrimental effect on the availability the DNS service.

Nameserver organisation failure

Many organisations host nameservers on behalf of other organisations; this includes the root, TLD and authoritative nameservers. A failure of any of these organisations will affect the availability of the DNS, the higher up the hierarchy the more significant.

Spoofing

The lack of origin authentication enables an attacker to create a spoofed DNS message to provide false DNS information, or to use in DOS attacks. If an attacker is able to predict that a resolver will be issuing a query, he can poison DNS RRs of his choosing, assuming he can prepare a response that the requesting host will accept.

At first inspection the attacker has a 1 in 65,536 chance of guessing the transaction ID correctly (assuming the transaction ID is generated by a good random number generator); the attacker must also guess the responding authoritative nameserver (at least 2 per domain). Whilst these odds don't seem good, this is per guess, and there are no limits on the number of spoofed responses the attacker can send. The attacker also has a head start; he can start sending the spoofed responses to arrive as soon as the server has sent the query, giving him a window of opportunity equal to the round trip time between the requesting host and the responding server.

Modifying

Of course, if the attacker has physical access to the network, he can be inline (perhaps using an ARP poisoning attack) and doesn't need to guess as he can read the transaction ID directly from the query.

Poisoning

Assuming the attacker has been able to spoof or modify a DNS response, he can poison the cache of the server by modifying or adding RRs whilst specifying a long TTL to extend the period of the attack. Early versions of DNS server software trusted all information included in the additional records section. Unsurprisingly attackers used this section to poison any DNS record.

This was fixed in subsequent implementations by requiring that additional records be in-bailiwick.

4.3.5. Mitigating server issues

The above server issues are mitigated by the following:

- Availability.
- Disable Open Recursive DNS Servers (see 4.2.4).
- DNSSEC (see 4.2.4).
- IP spoofing prevention (see 4.2.4).
- Redirection mitigation (see 4.2.4).
- Secure infrastructure.
- Secure server configuration.
- Secure updates.
- Technical controls (see 4.3.3).
- Training and awareness (see 4.2.4).

Availability

Availability of critical components should be considered when designing systems; this is true for both the DNS and the network, as both rely on each other. Multiple DNS servers should be available, ideally on discrete network infrastructure; this should be applied to DNS Servers for internal resolution, and similarly for external resolution. For critical Internet-based services, an organisation may wish to use more than one service provider to host their zone file for external resolution. This would provide both hardware and organisational resiliency.

Secure infrastructure

In a managed environment, policies, procedures and guidelines should be followed to ensure that adequate protection is provided and where changes are required they are considered for approval and are documented. This will reduce the likelihood of server configuration errors.

Secure server configuration

In addition to 4.2.4 and 4.3.3 DNS servers allow access lists to be configured defining the servers that can request zone transfers [LIU06].

Secure updates

Vendors and open source developers should strive to consider security in the complete lifecycle of their products. Considering security requirements at all stages of the lifecycle is not only more effective at reducing security issues, it is also quicker and cheaper than fixing bugs once code has been developed.

4.4. DNS registration issues

4.4.1. Domain hijacking

Domain hijacking is wrongfully taking control of a domain from the **registrant** (the rightful holder). It is made possible due to shortcomings in the domain name registration and management processes, such as weak registrant authentication by registrars (the organisations completing registration). For example, attacks have been enabled as e-mail was considered an acceptable form of authentication for authorising domain transfers. Social engineering can play a part in an attacker hijacking a domain.

Domain hijacking can cause significant issues to the losing registrant [ICA05]. Attackers may wish to hijack a domain for the following reasons:

- Brand damage, leading to loss of business.
- Denial of Service.
- Espionage.
- Fraud - acting as a man-in-the-middle to steal credentials for financial sites.
- Theft for resale / extortion.

4.4.2. Typo-squatting

Typo-squatting (registering similar domains - *www.gooogle.com* for *www.google.com* for example) relies on users incorrectly typing domain names, or clicking on links in e-mails that appear to be correct. These domains will be run by the attacker and could be used for phishing.

The use of **internationalised domain names** (names that contain non ASCII characters) has made it easier for attackers to make a link (or even an SSL certificate if enrolment is automatic) appear to be valid for a site. For example Gabrilovich and Gontmakher were able to register the homographed domain name [GAB09]:

www.mi%d1%81r%d0%bes%d0%beft.com

When decoded by applications that support internationalised domain names (such as Firefox version 3.5.2) this is displayed as:

www.microsoft.com.

4.4.3. Mitigating DNS registration issues

The above DNS registration issues are mitigated by the following:

- Improved registrant authentication.
- Review of registrations.
- Training and awareness.

Improved registrant authentication

Registrant authentication should be improved to provide assurance that when changes are requested for a domain, the identity of the registrant has been validated. We have seen that e-mail can be intercepted using DNS poisoning; clearly, using e-mail for authentication does not provide appropriate assurance of identity.

Reviews of registrations

Where domains names are being registered that are similar to existing domain names, additional checks should be conducted to identify why a certain domain name is required, and if the name is clearly to be used for illegal activity, registration should be denied.

Training and awareness

Training and awareness programmes should be run with registrar staff to ensure that they are aware of and comply with relevant policies and procedures. This should include identifying social engineering attempts.

4.5. DNS Denial of Service

4.5.1. DNS servers

DNS can be part of a denial of service attack as one of the following:

- Amplification attack.
- DNS used to enable a denial of service.

Amplification attack

The use of UDP in DNS means that the attacker can use a spoofed source address for DNS queries. Servers receiving these queries (which are likely to be open recursive DNS servers) will process them and reply to the spoofed address. As DNS queries are small, the source of the query uses a correspondingly small amount of bandwidth; however, if the query is for a large record, then the size of the response sent to the victim can be much larger. This kind of attack can saturate the Internet access bandwidth of the victim and create a DOS for all services using the same access.

Amplification attacks are covered in more detail in section 7.

DNS used to enable a denial of service

If an attacker is able to somehow introduce false DNS information to an application that relies on DNS, then the attacker can cause a denial of service by preventing the application from connecting to the required service.

4.5.2. Attacks against the Root and TLD servers

A significant DOS attack against the root servers was launched on 22 October 2002; it was targeted against all 13 of the root server IP addresses simultaneously. The root servers continued to operate during the attack, although not all queries reached the servers due to network congestion caused by the increased traffic.

An amplification attack was launched against every server of one of the TLD operators on 5 February 2006 and 7-8 February 2006, fully saturating two thirds of the operators access circuits and partially saturating the remaining circuits [LIU06]. Analysis of the attack shows that the nameservers were significantly affected with 4 of the 6 not responding to more than 50% of queries during the attack on 5 February and 4 of the 6 not responding to more than 90% of queries during the attack on 7-8 February.

Analysis of an attack against the E, F and G root servers on 15 February 2006, shows that during the attack the E root server was not responding to more than 50% of queries, the G root server was not responding to more than 90% of queries, however the F root server was only slightly impacted.

A second significant denial of service attack against the root servers was launched on 6 February 2007, this time targeted against 6 of the root server IP addresses simultaneously. Again the root servers continued to operate during the attack; although the G and L root servers were badly affected, of those attacked the G and L servers were the only ones not to have deployed Anycast.

4.5.3.Mitigating DNS Denial of Service

The above DNS Denial of Service issues are mitigated by the following:

- Anycast.
- Disable Open Recursive DNS Servers (see 4.2.4).
- DNSCurve (see 4.2.4).
- DNSSEC (see 4.2.4).
- IP spoofing prevention (see 4.2.4).
- Secure client (see 4.2.4).
- Secure server (see 4.2.4).
- Training and awareness (see 4.2.4).

Anycast

The 2002 attack against the root servers accelerated improvements in replication of the servers using Anycast [VIX02]. The Internet Corporation for Assigned Names and Numbers (ICANN) Security and Stability Advisory Committee report responding to the 2006 attacks and the ICANN factsheet on the root server attack on 6 February 2007 both concluded that Anycast is a good defence to these attacks [ICA06] [ICA07].

5. CACHE POISONING

5.1. Difficulty of cache poisoning

If an attacker wishes to poison a DNS RR, he can trigger a query for that RR on the target DNS server, generate a spoofed RR that will pass the resolver's checks and get it to the target server before the real response gets there. This has been known for years, although was not considered a significant problem due to caching (the DNS server will not issue another query for that RR for the duration of the TTL, meaning there is only one attempt possible per TTL period).

The likelihood of an attack succeeding can be calculated by the formula in Hubert and van Mook's paper 'Measures to prevent DNS spoofing' [HUB06]:

$$P_{cs} = 1 - \left(1 - \frac{D.R.W}{N.P.I}\right) \wedge \left(\frac{T}{TTL}\right)$$

Where:

P_cs:	Probability of success
I:	Number of distinct transaction IDs available (maximum 65536)
P:	Number of source ports used by DNS server (maximum around 64000, but often 1)
N:	Number of authoritative nameservers for a domain (minimum of 2)
R:	Number of packets sent per second by the attacker
W:	Window of opportunity, in seconds, bounded by the response time of the authoritative servers (often 0.1s)
D:	Average number of identical outstanding questions of a resolver (typically 1)
T:	Spoofing time period
TTL:	TTL of RR being poisoned
^:	Raise to the power of

Figure 6 – Pre-Kaminsky spoofing probability formula

To measure how difficult this is we will calculate two examples, the first with a TTL of 1 hour, the second with a TTL of 1 day. We will use the following metrics in the calculations:

- I: 65536.
- P: 1.
- N: 2.
- R: 7000.
- W: 0.1.
- D: 1.

The calculated probability of a spoofing attack succeeding with a TTL of 1 hour is:

- Average time for 10% chance of success: 0.8 days
- Average time for 25% chance of success: 2.2 days
- Average time for 50% chance of success: 5.4 days
- Average time for 75% chance of success: 10.8 days
- Average time for 90% chance of success: 17.9 days

The calculated probability of a spoofing attack succeeding with a TTL of 1 day is:

- Average time for 10% chance of success: 19.6 days
- Average time for 25% chance of success: 53.8 days
- Average time for 50% chance of success: 129.4 days
- Average time for 75% chance of success: 258.7 days
- Average time for 90% chance of success: 429.1 days

A DNS query response size of 70 bytes and 7000 packets per second gives an aggregate bandwidth of approximately 4Mbps. That's a very large amount of traffic to go un-noticed for a long period of time and the reason why poisoning attacks were considered impractical (unless of course the attacker is inline).

5.2. The Kaminsky vulnerability

The vulnerability that Dan Kaminsky disclosed in 2008 made it possible to attempt a poisoning attack without waiting for the TTL to expire [DOU08]. There are a number of aspects to this attack:

- The attacker chooses when to start.
- The attacker can send as many responses as he likes.
- Additional RRs that are in-bailiwick are accepted.
- The attacker can start again immediately if not successful.

The attacker chooses when to start

Once the target DNS server has sent a query to the authoritative server there's a race between the attacker and the authoritative server to get a valid response back to the DNS server. When this race starts is decided by the attacker, so he can start to send a spoofed response (from each of the authoritative nameservers) as soon as the query has left the DNS server. The attacker's spoofed response will get to the DNS server first; although the transaction ID may not be correct, he has a 1 in 65,536 chance of guessing correctly, meaning that it's extremely likely it will be rejected.

The attacker can send as many responses as he likes

The attacker can send as many responses as he likes, as long as they arrive before the valid response from the authoritative server. If the attacker can send 100 spoofed responses to the DNS server (from each of the authoritative nameservers) with randomly chosen transaction IDs before the valid response from the authoritative server, then he improves his chances. Again the transaction ID may not be correct, but he has improved his chances of guessing correctly to 1 in 656, meaning that it's very likely it will be rejected.

Additional RRs that are in-bailiwick are accepted

If a response is received with authoritative nameservers and glue records that are in-bailiwick they are accepted and added to the cache.

The attacker can start again immediately if not successful

This is the new aspect of the attack. Dan Kaminsky realised that if he could trigger queries for RR that did not exist (say *00000001.example.com*) then they could not possibly be in the cache of the DNS server. These RRs wouldn't be in the authoritative nameserver either. That, however, is not important; what is important is that there was a query for a known RR sent from the DNS server to an authoritative nameserver. The attacker could try and spoof a response to that query by sending as many responses as he can with random transaction IDs before the authoritative nameserver replies. If the attacker is very lucky he guesses the transaction ID correctly; if not he can try for another RR that does not exist (say *00000002.example.com*). Sooner or later the attacker will get a match and his response will be added to the cache.

The Kaminsky vulnerability enables the attacker to poison the cache without concern for the TTL; of course it does mean that he's poisoning it with invalid RRs that will never be requested.

The reason for doing this is that if the spoofed responses contain RRs for the authoritative nameservers for *example.com* and corresponding glue RRs, (both with a very long TTL and under the attacker's control) this extra information is added to the cache, overwriting the previous entries. As the valid RRs in the nameserver expire, new requests that should be sent to the authoritative nameservers for *example.com* are instead sent to the attacker's nameservers. He can now choose how to respond to all requests for the entire domain.

To measure how much easier this makes poisoning attacks, we re-calculate the two previous examples. This time the formula is revised to remove the TTL and is now:

$$P_{cs} = 1 - \left(1 - \frac{D.R.W}{N.P.I}\right)^{(T)}$$

Figure 7 – Post-Kaminsky spoofing probability formula

The probability of a spoofing attack succeeding with any TTL is (using the same metrics as in 5.1):

- Average time for 10% chance of success: 20 seconds
- Average time for 25% chance of success: 54 seconds
- Average time for 50% chance of success: 130 seconds
- Average time for 75% chance of success: 259 seconds
- Average time for 90% chance of success: 430 seconds

This time to have a greater than 50% chance of success an attack would need to be sustained for just 130 seconds, compared to over 5.4 days or 129.4 days prior to the Kaminsky attack. This very significant change makes a poisoning attack easy.

5.3. Mitigating the Kaminsky vulnerability

A DNS summit was hosted by Microsoft to investigate options to mitigate the vulnerability. The solution was needed quickly and needed to be easily to implement and backwards compatible with deployed DNS software. These requirements prevented solutions that required changes to the DNS protocol.

5.3.1. Short term solutions

Source port randomisation

A decision was made to implement source port randomisation for DNS queries. This is something that had been a feature of djbdns since 1999, and PowerDNS since 2006. Query source port randomisation makes it more difficult for the attacker to spoof DNS responses by randomising the source UDP port used to send queries from the DNS server. With query source port randomisation the attacker must guess both the transaction ID and the source port of the query.

After applying the patch from Microsoft Windows 2000 and 2003 servers allocate 2500 ports. 2500 is the default setting and was chosen to prevent the allocation conflicting with ports used by other services, it is however configurable for up-to 10000 ports [MIC08]. BIND servers can use the 1024-65535 range [ISC08].

Using Windows with a 2500 port range makes the attack 2500 times more difficult. The probability of a spoofing attack succeeding with any TTL to (using the same metrics as in 5.1) can now be calculated:

- Average time for 10% chance of success: 0.6 days
- Average time for 25% chance of success: 1.6 days
- Average time for 50% chance of success: 3.8 days
- Average time for 75% chance of success: 7.5 days

- Average time for 90% chance of success: 12.5 days

Using BIND with the maximum 64511 port range makes the attack 64511 times more difficult. The probability of a spoofing attack succeeding with any TTL to (using the same metrics as in 5.1):

- Average time for 10% chance of success: 14.8 days
- Average time for 25% chance of success: 40.1 days
- Average time for 50% chance of success: 96.8 days
- Average time for 75% chance of success: 193.5 days
- Average time for 90% chance of success: 321.1 days

Source case randomisation

RFC 1034 [MOC87a] requires case preservation from DNS queries to DNS query responses:

> *"domain name comparisons for all present domain functions are done in a case-insensitive manner...when you receive a domain name or label, you should preserve its case."*

This means that a query for www.example.com will retrieve the same records as WWW.EXAMPLE.COM, or WwW.eXaMpLe.CoM and that the response should maintain the same case as the query.

A method for using source case randomisation using 0x20-bit encoding has been proposed using the 0x20 bit in an ASCII letter (there is a 0x20 shift in ASCII between upper and lower case – A is encoded as 0x41, a is encoded as 0x61, K is encoded as 0x4b, k is encoded as 0x6b...) [DAG08]. The paper proposed that a query should be processed as follows:

- The name in a query is converted to canonical format, say all lowercase.
- The query is encrypted
- The ciphertext bits are used to determine the case randomisation used, 0 indicating upper case, 1 indicating lower case.
- The name in the query is encoded and processed as normal.
- When the response is received the process is repeated to compare the expected case with the case in the response. If they match the response is accepted, if not it is rejected.

If resolvers were configured to randomise the case of the query this would make the attacker's job more difficult; he would now have to guess the transaction ID, source port and the case of each of the letters in the name being queried. Short domain names would see a modest improvement, with protection increasing as the length of the domain name increases. The smallest domain name the writer can find is P1.fr, even this domain name would benefit from 3 extra bits, making the attack 8 times more difficult.

Source case randomisation was not chosen as a solution at the summit, although an individual or organisation could add this to any resolver to further reduce the possibility of a successful attack.

5.3.2. Long term solutions

Whilst adding randomisation to the query does provide a workable solution to the Kaminsky vulnerability, it does not remove the vulnerability, it just makes it more difficult to exploit. A fully patched BIND server has been successfully attacked [GOO08]. The attack took 10 hours on a directly connected gigabit Ethernet showing that the attack is far more difficult, although still possible. An attacker with access to a corporate or ISP network where gigabit Ethernet is common, may still be able to exploit the vulnerability.

History tells us that attacks only get easier over time, so a longer-term, more secure solution is required for DNS. Sections 6.1 and 6.2 cover potential solutions to the vulnerability, using cryptography to provide greater assurance.

6. MITIGATING CACHE POISONING

6.1. DNSSEC

DNSSEC aims to provide the following services [ARE05a]:

- Data Origin Authentication.
- Data Integrity.
- Authenticated Denial of Existence.

It does this using public key cryptography to provide digital signatures on RRs; the validity of these signatures can be traced back by a **validating security aware resolver** (in this context security aware indicates an understanding of DNSSEC) to a trust anchor. The aim is that DNSSEC will have a single anchor at the DNS root that all resolvers can trust and use to validate the digital signatures and RRs (see also 6.1.9 on DLV).

DNSSEC requires changes to the DNS protocol in the form of new RRs and new headers. The new RRs are:

- **DNS Public Key (DNSKEY).**
- **Resource Record Signature (RRSIG).**
- **Delegation Signer (DS).**
- **Next Secure (NSEC).**
- **Next Secure 3 (NSEC3).**

The new headers are:

- **Authentic Data (AD).**
- **Checking Disabled (CD).**

DNSSEC may generate larger messages than specified in the original DNS specification (RFC 1035 [MOC87b] specifies a maximum size of 512 bytes), so support for **EDNS0** [VIX99] is required (see 6.1.3).

6.1.1. New RRs in DNSSEC

DNSKEY

The DNSKEY RR contains one of the public keys of a zone and includes the following fields [ARE05b]:

- Flags to indicate **verification keys**, e.g. 256 for **zone verification key (see 6.1.8)**.
- Protocol field, always set to 3.
- Algorithm field, e.g. 5 for RSA with SHA-1.
- The public key, e.g. AQPSKmynfzW4... (key truncated).

These are shown below in an example from RFC 4034 [ARE05b]:

example.com. 86400 IN DNSKEY ***256 3 5 AQPSKmynfzW4****...* (key truncated)

RRSIG

The RRSIG RR contains the digital signature. RRs of the same type are grouped together in an **RRSet**, and the set is signed. For example, all of the NS records for a domain will be grouped together to form a RRSet for NS RRs and the signature is created on that set. This is more efficient than signing individual RRs as records of a particular type are served together.

The RRSIG RR contains the following [ARE05b]:

- **Type** covered, e.g. A records.
- **Algorithm** used to generate the signature, e.g. 5 is RSA with SHA-1.
- **Labels**, number of labels in the owner name, e.g. 3.
- **Original TTL**, needed to re-create original input when verifying the signature, e.g. 86400.
- **Signature expiration**, e.g. 20030322173103 for 17:31:03 on March 22nd 2003.
- **Signature inception**, e.g. 20030220173103 for 17:31:03 on February 20th 2003.
- **Key tag**, used as a pointer where more than 1 public key exists.
- **Signer's name**, e.g. *example.com*.
- **Signature** on the RRSIG fields, e.g. oJB1W6WNG... (key value truncated).

These are shown below in an example from RFC 4034 [ARE05b]:

> *host.example.com. 86400 IN RRSIG A **5 3 86400 20030322173103 20030220173103 2642 example.com oJB1W6WNG**...* (key value truncated)

DS

The DS RR is a hash of the DNSKEY record; it is stored in the parent zone (e.g. *com* for *example.com*) and is used when validating the DNSKEY RR.

The DS RR contains the following fields [ARE05b]:

- Key tag, used as a pointer where more than 1 public key exists, e.g. 60485.
- Algorithm used to generate the signature, e.g. 5 is RSA with SHA-1.
- **Digest Type** used to calculate the digest value, e.g. 1 is SHA-1.
- **Digest** value of the concatenation of the fully qualified owner name and the DNSKEY data, e.g. 2BB183AF5F22588179A53B0A.

These are shown below in an example from RFC 4034 [ARE05b]:

> *dskey.example.com. 86400 IN DS **60485 5 1 2BB183AF5F22588179A53B0A***

NSEC

The NSEC RR is used to provide authenticated denial of existence; it contains the following fields [ARE05b]:

- **Next Domain Name** when listed in canonical order, e.g. *host.example.com.*
- **Type Bit Maps** identify the RR types of the NSEC RR's owner, e.g. A MX RRSIG NSEC.

These are shown below in an example from RFC 4034 [ARE05b], for a response served to prove that *beta.example.com* does not exist:

> *alfa.example.com. 86400 IN NSEC **host.example.com. A MX RRSIG NSEC***

If a resolver sends a query for the A record for *beta.example.com* and subsequently receives, and verifies, the above, then the resolver has assurance that *beta.example.com* does not exist. The above NSEC RR shows that there are no RRs between *alfa.example.com* and *host.example.com* when RRs are arranged in canonical order.

Additionally if a resolver sends a query for the TXT record for *alfa.example.com* and subsequently receives, and verifies, the above then the resolver has assurance that the TXT record for *alfa.example.com* does not exist as the NSEC RR type bit maps state that *alfa.example.com* has only A, MX, RRSIG and NSEC RRs.

NSEC allows anyone to fully enumerate a zone by looking up the NSEC record for any host (even those that don't exist), and using the next domain name field to identify the next host in the zone and so on until all of the hosts have been identified. When hosts are arranged in canonical order, the last RR loops back to the first. NSEC was not well received for this reason. NSEC3 has been proposed as an alternative that prevents zone enumeration.

NSEC3

The NSEC3 RR has been proposed in RFC 5155 [LAU08] to provide authenticated denial of existence in DNSSEC whilst providing measures against zone enumeration. It achieves this by replacing the RRs with hashes of the RRs. NSEC3 does provide an improvement on NSEC; however, it does not protect against offline attacks against the hashed values of the RRs. Whilst this is more difficult and time consuming than NSEC, it does not prevent the possibility of (at least) partial enumeration.

6.1.2. New flags in DNSSEC

AD flag

The AD header can only be set by a validating security aware nameserver when communicating with a security aware resolver. The AD bit must only be set if the nameserver has been able to validate all of the DNSSEC RRs in the answer and authority sections of the response.

The AD header should only be used on a secure channel between the nameserver and the resolver to prevent unauthorised modification of the flag setting.

CD flag

The CD header can only be set by a validating security aware resolver when communicating with a security aware nameserver. The CD bit indicates that the resolver is capable of validating DNSSEC RRs, the presence of the flag tells the nameserver that it does not need to validate the RRs on behalf of the resolver.

6.1.3. EDNS0 support

EDNS0 support is required in DNSSEC due to larger message sizes, it is defined in RFC 2671 [VIX99]. The need for EDNS0 support is best demonstrated by an example; a DNS query for the NS records for the org domain is 29 bytes, the response is 167 bytes. To obtain the digital signatures that DNSSEC requires to validate these records requires an additional query for the RRSIG RRs. The query size is 40 bytes, the response is 983 bytes.

EDNS0 provides enhancements to the DNS protocol to allow the addition of more fields using the **OPT RR** (the OPT RR is a pseudo RR; it is only used for signalling and does not contain any DNS data). This allows a sender to indicate their maximum UDP payload in a DNS message.

One of the flags in the OPT RR is used for the **DO bit (DNS OK)** and tells nameservers that the resolver can accept DNSSEC RRs; if this flag is missing then security aware nameservers will not send DNSSEC RRs, thus increasing efficiency.

6.1.4. DNSSEC algorithms

DNSSEC supports the following algorithms, from RFC 4034 [ARE05b]:

- RSA/MD5.
- Diffie Hellman.
- DSA/SHA-1.
- Elliptic curve.
- RSA/SHA-1.

RFC 5155 [LAU08] adds the following:

- DSA-NSEC3-SHA-1.
- RSASHA-1-NSEC3-SHA-1.

6.1.5. DNSSEC digests

DNSSEC supports the following digests for DS RRs, from RFC 4034 [ARE05b]:

- SHA-1.

RFC 5155 [LAU08] adds the following:

- SHA-256.

6.1.6. Signing with DNSSEC

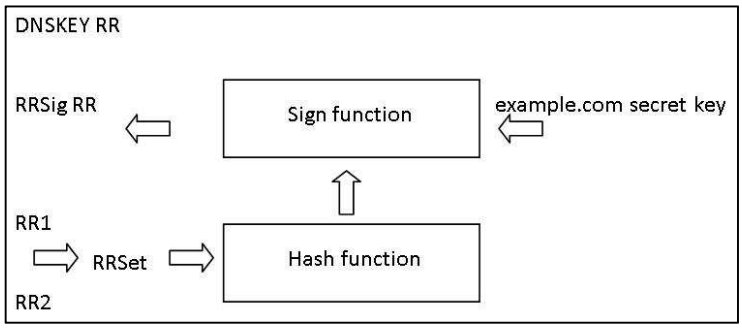

Figure 8 – Signing RRs with DNSSEC

The signing procedure for RRs of *example.com* is shown in figure 8. When signing, RRs are grouped together to form an RRSet. The RRSet is signed by applying a hash function, then taking the output of that hash function and generating a digital signature using the *example.com* secret key and a signing function. The resultant digital signature is then placed in a RRSIG RR ready to be served. The *example.com* public key is stored in the DNSKEY RR to be available for verification of the RRSIG by relying parties.

The RRs are signed in advance, for example once a month, taking into account the inception and expiration dates of the signatures, and the period it will take DNS caches to update to the latest RRSIG RRs.

6.1.7. DNSSEC Chain of trust

When verifying DNSSEC signed RRs, validating security aware resolvers require a way of verifying the authenticity of the *example.com* public key. The signing procedure for the public key of *example.com* is shown in figure 9.

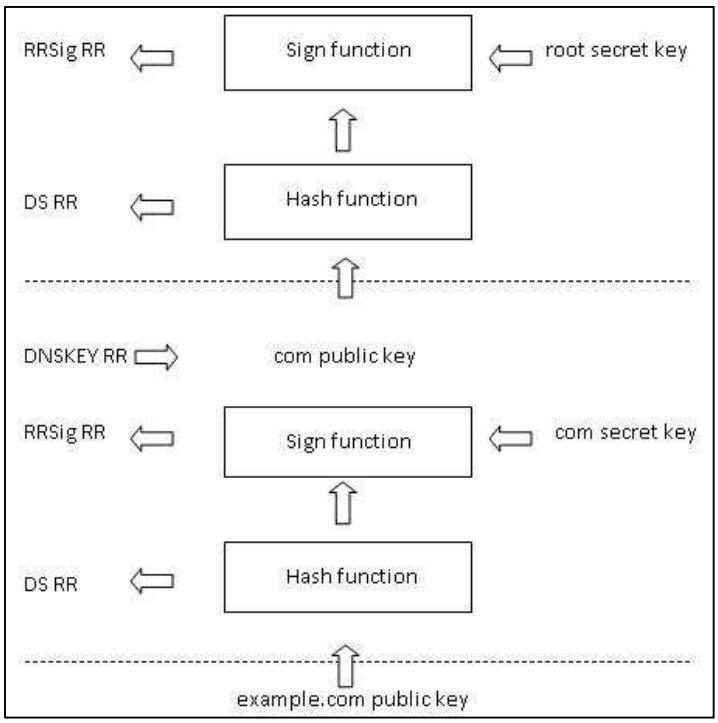

Figure 9 – Signing keys with DNSSEC

The DNSSEC administrator at *example.com* provides the DNSSEC administrator at *com* with the public key of *example.com* (and appropriate proof of identity). The *com* administrator then generates a hash of the *example.com* public key, stores this in a DS RR, takes the hash and generates a digital signature using the *com* domain secret key and a signing function. The resultant digital signature is then placed in an RRSIG RR. Both the DS and the RRSIG are stored in the *com* zone.

The validating security aware resolver now requires a mechanism of verifying the authenticity of the *com* domain public key. The above procedure is repeated enabling the root to generate a DS RR and RRSIG RR of the *com* public key. The validating security aware resolver must obtain a trusted copy of the root public key (the trust anchor for DNSSEC) by some trusted means.

When validating RRs the security aware resolver first validates the RRSIG on the DS in the parent zone. If this validates then the security aware resolver obtains the corresponding DNSKEY RR from the child zone, it takes a hash of the DNSKEY RR and compares that to the hash obtained from the DS RR in the parent zone. If the two hash values match, then the DNSKEY RR is validated.

6.1.8. DNSSEC keys

DNSSEC can optionally use two key pairs, **Zone Signing Keys** (ZSK) and **Key Signing Keys** (KSK). When two key pairs are used the ZSK secret keys are used to sign the zone RRs, whilst KSKs are used to sign the ZSK public key. The more a cryptographic key is used the greater the chance of successful cryptanalysis, the separation of ZSKs and KSKs removes the need to involve the parent domain when new ZSKs are generated, allowing more frequent ZSK changes, reducing the chance of successful cryptanalysis.

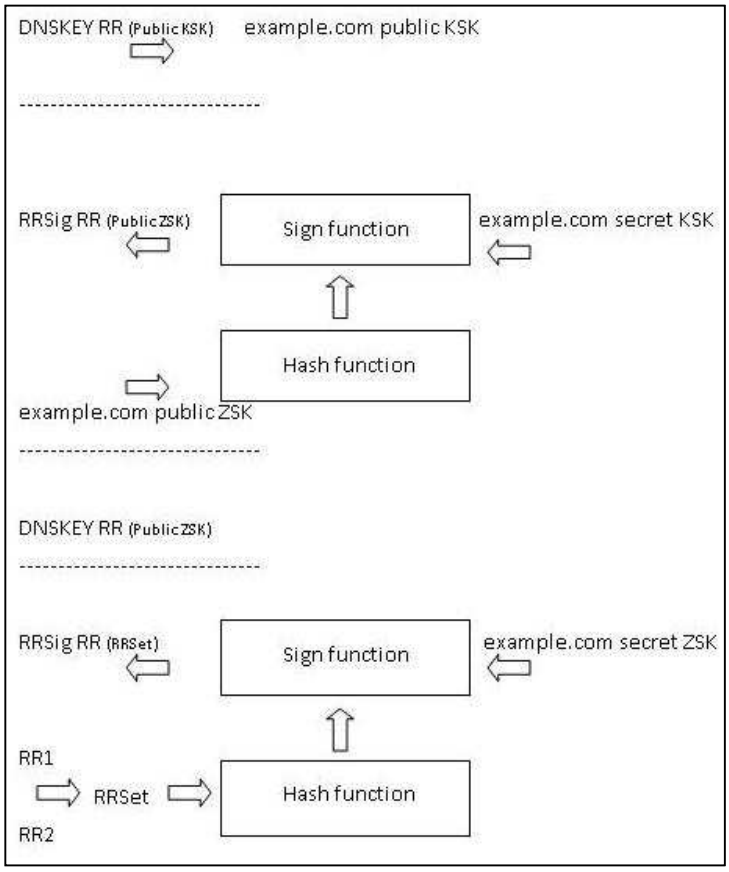

Figure 10 – ZSKs and KSKs in DNSSEC

The use of ZSKs and KSKs is shown in figure 10. An RRSet is signed using the *example.com* secret ZSK and is added to the RRSIG RR (on the RRSet). The public ZSK is published in a DNSKEY RR (on the Public ZSK), however for a relying party to trust this key it must be signed, therefore the public ZSK is signed using the secret KSK, the resultant signature is added to the RRSIG RR (on the Public ZSK). We are now in the position where the Public KSK needs to be trusted; this can be validated through the chain of trust to the trust anchor, as covered in 6.1.7.

6.1.9. DNSSEC Issues

The following DNSSEC problems are listed in the RFC 4033 [ARE05a]:

- Confidentiality is not provided, by design choice.
- DNSSEC resolvers may be vulnerable to denial of service attacks if an attacker interferes with signed RRs. This will consume resources when validating signatures.
- DNSSEC suffers from the ability of an attacker to enumerate a zone using NSEC. NSEC3 was developed to address this issue, although superior to NSEC, NSEC3 suffers from offline attacks not preventing zone enumeration.
- Protection is not provided for a non-security aware resolver from a malicious DNS server serving false responses. If this protection is required then the resolver would need to be a validating security aware resolver.
- Protection is not provided for queries and responses between a non-security aware resolver and the DNS server. DNSSEC provides the AD and CD bits to signal DNSSEC status, however does not protect them allowing an attacker to modify them. An external mechanism such as IPSec or **TSIG** (TSIG uses a keyed hash to provide data origin authentication and data integrity for DNS messages [VIX00]) is required to protect the channel.
- The RFC acknowledges that DNSSEC is complicated and is more likely to suffer from coding, implementation and configuration issues.

DNSSEC is also subject to the following issues:

- RRSIG expiration, if for some reason RRSIGs are allowed to expire then RRs will not be able to be validated. Policies, procedures and technologies are needed to ensure that RRs are re-signed taking into account the inception and expiration dates of the existing signatures, and the period it will take DNS caches to update to the latest RRSIG RRs.

- Migration is not catered for, DNSSEC deployment will take time with some zones signed, and others not. For this interim period it is likely that unsigned RRs will need to be trusted.
- DNSSEC does not provide revocation and is therefore vulnerable to replay attacks where RRs have been changed before the end of the validity of the digital signature.

Currently neither the root zone nor the *com* zone are signed so in reality this model can't be followed; if the relying party wanted to verify any *example.com* RRs today he would need a trusted copy of the *example.com* public key. If there are many zones that require validation this adds an overhead to obtain and import these keys, **DNSSEC Lookaside Validation (DLV)** has been proposed in RFC 5074 [WEI07]. DLV provides a mechanism for a server to automatically obtain a copy of the public key of a zone.

On June 3 2009 the US National Telecommunications and Information Administration (NTIA – part of the US Department of Commerce) and NIST announced that they were working with ICANN and VeriSign to an approach to signing the root zone in 2009 [FOR09]. VeriSign (who operate the A and J root servers and administer the *com* and *net* domains) have announced that the *net* zone will be signed by the end of 2010 and the *com* zone in early 2011 [LAR09].

6.2. DNSCurve

DNSCurve is a proposal from Daniel Bernstein (author of the djbdns DNS server) to address some of the issues in the DNS. At the time of writing DNSCurve cache software is not available, furthermore there is limited information available about DNSCurve; the material in this section is taken from the DNSCurve website [BER09a].

DNSCurve aims to provide the following services:

- Confidentiality.
- Integrity.
- Availability.

It does this using Elliptic-Curve Cryptography (ECC) to provide link level encryption between resolvers and servers; the integrity of DNS messages is validated by a resolver to identify if a

message has been modified. DNSCurve claims to enhance availability by dropping DNS responses that fail the integrity check, meaning the responses it accepts are the ones sent by the server (preventing attackers denying service by poisoning the DNS with incorrect RRs).

DNSCurve does not require any changes to the DNS protocol; the above services are provided between two DNSCurve aware devices whilst maintaining interoperability with DNS. Larger messages than allowed in the original DNS specification may be generated, so support for EDNS0 [VIX99] is required.

6.2.1. DNSCurve algorithms

DNSCurve uses ECC with curve 25519. ECC is a highly efficient algorithm and provides a high level of security whilst offering good performance [VAN03].

6.2.2. DNSCurve keys

DNSCurve uses public key cryptography with two key pairs; both the resolver and the server have a secret key each and a public key each. The server's public key is part of the DNS NS RR for the server; for example, an NS RR with the DNSCurve public key and glue would be:

> *nytimes.com.*
> *IN NS uz5xgm1kx1zj8xsh51zp315k0rw7dcsgyxqh2sl7g8tjg25ltcvhyw.nytimes.com.*
> *uz5xgm1kx1zj8xsh51zp315k0rw7dcsgyxqh2sl7g8tjg25ltcvhyw.nytimes.com. IN A*
> *199.239.137.201*

As with DNSSEC a method of trusting public keys is required. DNSCurve proposes that root servers support DNSCurve and the DNSCurve public keys of the root servers are statically configured on resolvers. This would enable a DNS resolver to use DNSCurve to protect communications with the root servers to securely obtain the NS RRs for the authoritative TLD servers (which would include the DNSCurve public keys), and so on as per the standard DNS behaviour. DNSCurve private keys are generated by the holder and can be changed when required.

6.2.3. DNSCurve communication

The process for sending a DNSCurve query is shown in figure 11.

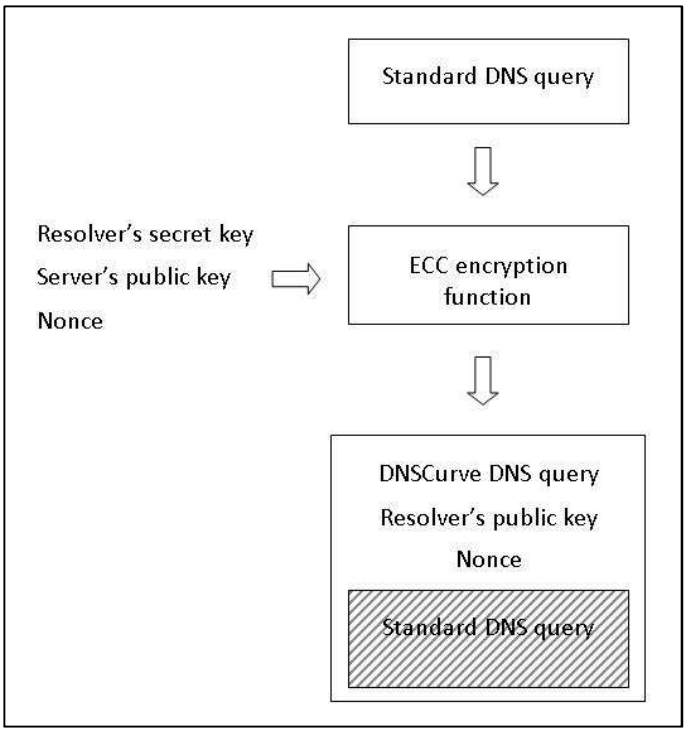

Figure 11 – DNSCurve outbound processing

When a DNSCurve resolver wishes to send a query, it looks at the server's hostname to identify if it has a DNSCurve key. The DNSCurve public key must be 54 bytes long; the first 3 bytes must be 'uz5', with the remaining 51 bytes being a-z or 0-9. If there is a matching key then it is used, if not a standard DNS query is sent.

The resolver protects the standard DNS query by generating a DNSCurve query – this contains the resolver's public key, a resolver generated nonce and a cryptographic box. The cryptographic box is encrypted and contains the standard DNS query; it is generated using the standard DNS query, the resolver's secret key, the server's public key and a resolver generated nonce.

On receiving the DNSCurve query the server opens the cryptographic box using the resolver's public key, the server's secret key and the resolver generated nonce. If a forgery is detected the server drops the forgery, otherwise the decrypted query is processed as a normal DNS query.

The server protects the standard DNS response by generating a DNSCurve response; this contains a server generated nonce and a ‘cryptographic box’. The cryptographic box contains the standard DNS response and is generated using the standard DNS response, the server’s secret key, the resolver’s public key and a concatenation of the server and resolver nonces.

On receiving the DNSCurve query the resolver opens the cryptographic box using the resolver’s secret key, the server’s public key and the concatenation of the server and resolver nonces. If a forgery is detected the resolver drops the forgery and waits for the valid response.

6.2.4. DNSCurve Issues

DNSCurve is also not without problems:

- DNSCurve does not provide authenticated denial of existence.
- DNSCurve does not provide data origin authentication.
- DNSCurve only provides data integrity and confidentiality at the link level. All servers must be trusted when using DNSCurve.
- DNSCurve resolvers may be vulnerable to denial of service attacks by interfering with messages to consume resources for all DNSCurve messaged received.
- Migration is not catered for, DNSCurve deployment will take time with some servers supporting DNSCurve, and others not. For the interim period all servers will need to be trusted.
- Protection is not provided for queries and responses between the DNS server and a non-DNSCurve aware resolver. An external mechanism such as IPSec or TSIG is required for this to protect this channel.
- Protection is not provided to a resolver from a malicious server serving false responses.

6.3. Analysis of DNSSEC and DNSCurve

Both DNSSEC and DNSCurve achieve what they set out to (with the exception of NSEC and NSEC3 for DNSSEC). Both solutions provide a cryptographic response to cache poisoning attacks, even where attackers are inline.

6.3.1. Trust model

The DNSSEC trust model is based on verifiable proof that the digital certificate corresponding to a RR was signed by the holder of a key. Any holder of a trust anchor public key gains data origin authentication and data integrity assurance (assuming that the secret ZSK has not been compromised). Non-validating stub resolvers must trust their DNS servers to validate the digital certificates corresponding to RRs and to serve them without modification.

DNSCurve does not provide the same level of assurance; its trust model provides assurance that data is not modified in transit. Protection is not provided against any modification of data whilst stored, or over non DNSCurve links. DNSCurve resolvers must trust their DNS servers to serve RRs without modification.

6.3.2. Confidentiality

DNSCurve provides confidentiality by design; however DNSSEC specifically excludes it in RFC 4033 [ARE05a]. Does DNS need confidentiality? Today much web content is not encrypted; whilst this is the case, there may not be a requirement for confidentiality in DNS messages. Moving forward as privacy issues increase, the use of encryption to provide confidentiality is likely to increase. At that stage DNS could be used to gain information about web usage. Encryption of DNS queries and responses, and the use of encryption to the endpoint would prevent DNS data from being used to monitor web usage. This would require trust in DNS service providers, and would not address traffic monitoring by IP address.

When providing authenticated denial of existence using NSEC and NSEC3, DNSSEC leaks zone information. DNSCurve does not suffer from this problem as it does not attempt to provide authenticated denial of existence. If DNSCurve were used for the entire resolution path then the non-existent domains and hosts would be correct, however DNSCurve provides no assurance of this to the stub resolver.

6.3.3. Integrity

DNSCurve provides hop by hop data integrity; it requires that the resolver trusts the servers involved in resolution. In comparison, DNSSEC provides end to end data integrity through digital signatures. DNSSEC does not provide revocation and is therefore vulnerable to replay attacks where RRs have been changed before the end of the validity of the digital signature. In this case anyone providing a service that relies on DNSSEC runs the risk of replay attacks of ‘old’, but still

‘in date’ certificates. To avoid this risk service providers must run services at both IP addresses until the ‘old’ certificates have expired.

6.3.4. Availability

DNSSEC provides protection against poisoning attacks for signed zones. DNSCurve provides protection against poisoning of traffic in transit, meaning that an attacker must compromise a DNS Server to poison a RR. Reducing poisoning attacks will (to some extent) have a positive effect on services that use the DNS. As DNSSEC signatures have specified validity periods, signatures must be regularly replaced. If they are not replaced in time then the remaining records will fail DNSSEC validation, causing an availability issue.

The certificates in DNSSEC require large RRs; as deployment of DNS increases this will lead to increased bandwidth and storage requirements, additionally these large RRs can be used in denial of service attacks (see sections 4.5 and 7). DNSCurve also requires an increase in bandwidth and storage due to the requirement to transport the keys, nonces and larger server NS records to accommodate public keys.

DNSSEC RRSIG RRs are generated ahead of time, so real time signing is not required; although real time validation is. Even though validation in DNSSEC is much faster than signing, a busy validating security aware resolver may require hardware upgrades. An attacker may attempt to consume resources at a validating security aware resolver by interfering with messages to consume resources. DNSCurve uses ECC to reduce the overhead associated with encryption, however DNSCurve may still be vulnerable to an attacker interfering with messages to consume resources at the DNSCurve host, this still needs to be factored for busy servers and resolvers.

6.3.5. Summary

At the time of writing it is 12 years since the first DNSSEC RFC was published (RFC 2065 [EAS97] in January 1997), however it is still suffering from significant issues. It has recently received much attention and seems to be gaining traction, probably due to the Kaminsky vulnerability. The increased attention may well help DNSSEC to address these issues over time.

DNSCurve does not have the same goals as DNSSEC; it does not provide data origin authentication or end-to-end integrity. It provides an answer to cache poisoning attacks and

confidentiality. DNSCurve benefits as it provides a solution that is less complicated, however provides less assurance.

Both DNSSEC and DNSCurve provide a response to cache poisoning attacks, however both need to address their issues if they are to be more widely deployed.

7. DNS AMPLIFICATION ATTACKS

7.1. DNS Amplification Attacks

DNS amplification attacks are attacks where a small query is sent to an open recursive DNS server with a spoofed source IP address. The server then responds to the specified source address (the address of the victim) with a much larger response. The amplification factor is the difference between the size of the query and the size of the response. The victim of a DNS amplification attack does not need to be DNS; as the attack generates a large amount of bandwidth, any service at the victim's location, or reached from, or via that location can be affected.

The DNSCurve website shows an example of this credited to Bert Hubert where a query of 31 bytes generates a response of 3974 bytes, giving an amplification factor of 128 [BER09b]. To be able to generate responses of greater than 512 bytes, support for EDNS0 is required at the open recursive DNS server. There are two ways to generate such responses: send queries for genuine large RRs, or look for an authoritative DNS server that is vulnerable to an exploit that will enable addition of a large TXT record.

The result is that a small amount of query traffic generates a significantly large amount of response traffic sent to the target. The amount of traffic can be increased if an attacker can use a botnet and send queries to a number of open recursive servers, creating a distributed amplification attack. In this kind of attack the bandwidth generated against the target can be very large; attacks of up to 7Gbps have been recorded [ICA06].

It's not just the target that is the victim of amplification attacks; resources at bots and at the open recursive servers are also consumed and other users of the server may see DNS response times increased during such an attack.

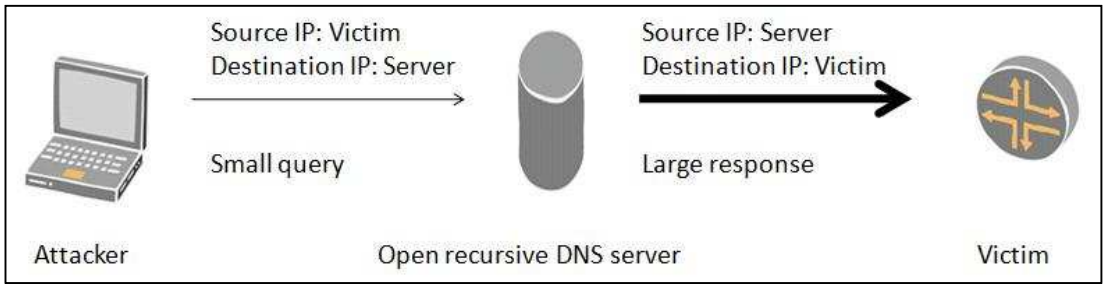

Figure 12 – Basic amplification attack

A basic amplification attack is shown in figure 12; the attacker generates DNS queries that will request a large response. These requests are sent to the open recursive DNS server, the server will obtain the RRs through recursion, and reply with the response to the spoofed IP address in the request, and this is the IP address of the victim of the attack. The traffic the attacker generates is amplified by the server and will consume a large amount of resources (bandwidth and CPU cycles) at the victim.

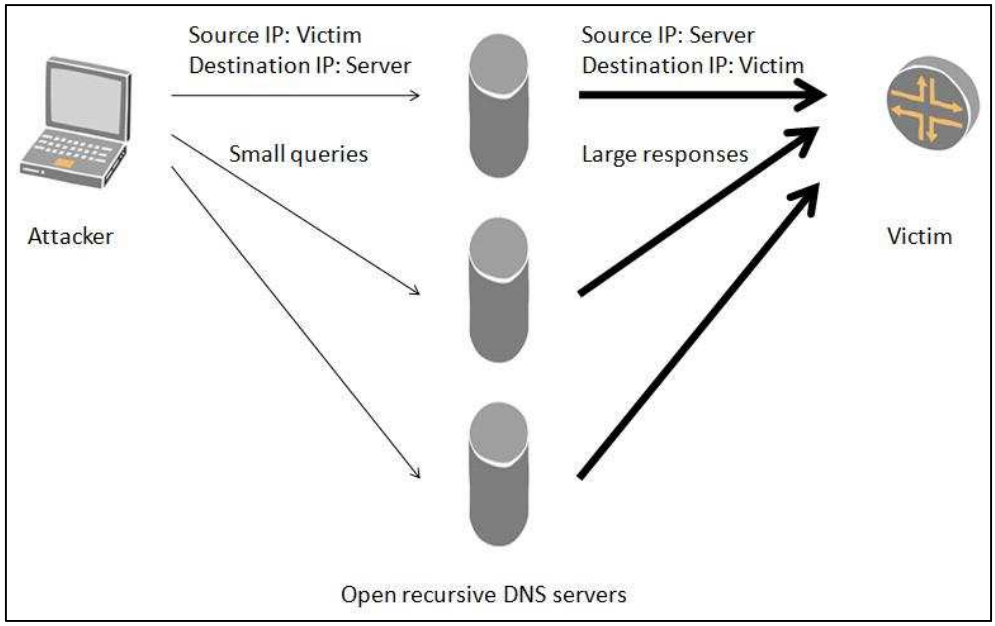

Figure 13 – Enhanced amplification attack

Figure 13 shows an enhancement where the attacker sends queries to more than one open recursive server. Not all open recursive DNS servers will respond with a large record (compare the response to the same query from different DNS Servers in Appendix C and Appendix D); some servers may combat this attack by limiting bandwidth, or may suffer from a lack of

bandwidth due to network conditions. Spreading the attack over multiple DNS open recursive DNS servers may increase the amount of traffic that reaches the victim.

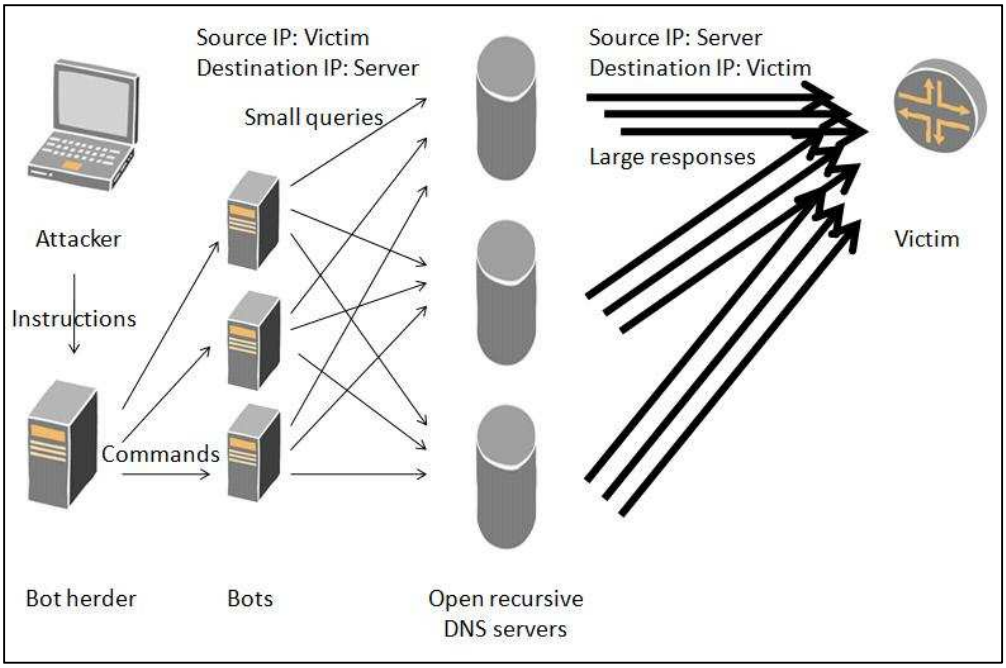

Figure 14 – Amplification attack using a botnet

Figure 14 shows an enhancement where the attacker uses the services of a botnet and combines that with more than one open recursive server. Rather than the attacker generating the traffic, the traffic is generated by bots. The attacker sends instructions regarding the attack to the bot herder, the bot herder then commands many bots to generate DNS queries, giving a second amplification effect. The maximum possible bandwidth generated during the attack is a factor of the number of bots generating queries, the number of queries generated and the size of the large responses.

7.2. DNSSEC and DNS Amplification Attacks

The example quoted in the second paragraph of section 7.1 is for a response containing DNSSEC records; the vast majority of the 3974 byte response is DNSSEC keys and signatures (the complete response is shown in Appendix B). Without DNSSEC the query generates a response of 270 bytes, still an amplification of the query, but to a much smaller extent. The signatures in DNSSEC generate large records that make amplification attacks more effective and do not require the addition of large TXT RRs to exploited servers.

7.3. Mitigating DNS Amplification Attacks

DNS amplification attacks can be mitigated by the following:

- Anycast (see 3.1.5 and 4.5.3).
- Disable Open Recursive DNS Servers (see 4.2.4).
- IP spoofing prevention (see 4.2.4).
- Secure server (see 4.2.4).
- Not serving overly large responses.
- Not using short TTL.
- Rate limit by source IP addresses.

Not serving overly large responses

Although not preventing the attack, an open recursive DNS server may protect itself from being used in an amplification attack by not serving overly large responses. For example the OpenDNS servers do not serve DNSSEC RRs (Appendix C shows DNSSEC requests to the OpenDNS servers for the org domain, the response size is 210 bytes, DNSSEC RRs were not included in the response, Appendix D shows DNSSEC requests to the Plusnet servers for the org domain, the response size is 549 bytes, and does include DNSSEC RRs). Whilst this may work today, with DNSSEC not widely used, this will not be a long-term solution if and when DNSSEC becomes more widely used. Additionally this behaviour will cause issues for applications that need the large RRs.

Not using short TTL

Although not preventing the attack, an authoritative DNS server that hosts large records may reduce the impact of an amplification attack by serving large responses with a long TTL. A short TTL would cause the authoritative servers to be re-queried for these records more regularly (the TTL in the example in Appendix B is 172800, or 48 hours).

Rate limit by source IP addresses

Applying rate limiting of DNS traffic to source IP addresses will reduce the maximum bandwidth that any open recursive server can get to the victim; however, for this to be effective it would need to be implemented before the traffic reaches the victim, so the victim's ISP would need to be able to apply such a policy.

8. PRACTICAL WORK

This practical work provides an example of how an attacker could execute a cache poisoning attack against a target domain, by exploiting the vulnerability that Dan Kaminsky disclosed. It shows how the cache poisoning could be extended to allow an attacker to obtain and use an SSL certificate for a spoofed copy of a website. This demonstrates an application that requires DNS RRs to be correct and procedural weaknesses in Certification Authority (CA) validation.

The practical work was split into two sections:

- In 8.1 we show how an attacker could execute a cache poisoning attack against the authoritative NS records for a domain, allowing him to choose how to respond to all requests.
- In 8.2 the attack is extended using poisoned A records that are used to direct traffic to a spoofed web server. This web server uses a certificate to authenticate the site (and suggests how an attacker might obtain a certificate trusted by browsers), giving users false confidence that the site is valid.

8.1. Practical example of a cache poisoning attack against NS records for an entire domain

8.1.1. Aim of the attack

The aim of this attack is to poison the authoritative nameserver RRs for a target domain on a vulnerable DNS server.

8.1.2. Lab set-up

To conduct the cache poisoning attack, a self contained lab was set up as shown in figure 15.

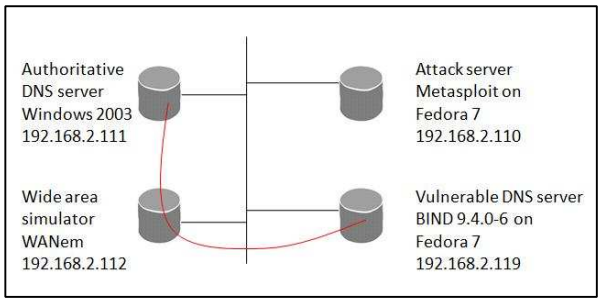

Figure 15 – Cache poisoning attack lab

The components of the lab were:

- An authoritative DNS server for the domain *richagar.co.uk* running on a Windows 2003 server.
- A caching DNS server using a static source port for queries (port 1053), making it vulnerable to cache poisoning attacks. The DNS server was BIND 9.4.0-6 running on Fedora 7.
- A wide area simulator to introduce latency between the vulnerable DNS server and the authoritative DNS server, WANem. WANem is an open source tool that provides the ability to simulate latency, packet loss, packet corruption, disconnections, packet re-ordering, Jitter, etc [WAN09].
- The attack server, running the Metasploit framework version 3.2 on Fedora 7. The Metasploit framework includes a number of security testing tools, including an exploit tool for the Kaminsky vulnerability.
- Ethernet LAN.

8.1.3. Before the attack

To provide latency between the caching server and the authoritative server, host routes were added to both servers causing them to route traffic via the wide area simulator (the route of the traffic is shown on figure 15 by the red line). The wide area simulator was configured to add 100ms of latency in each direction. This was validated using ICMP PING (Appendix E). The dig command was used to show that the authoritative DNS server for *richagar.co.uk* was correct; the complete dig output is shown in Appendix F.

8.1.4. Configuring the attack server

The Metasploit framework bailiwicked_domain attack tool was used; it was configured with the following options:

- DOMAIN was set to '*richagar.co.uk*'; this tells Metasploit the domain to hijack.
- NEWDNS was set to '*ns.poisoned.com*'; this tells Metasploit the name of the replacement nameserver.

- RECONS was set to ‘192.168.2.111’; this tells Metasploit the nameserver it should use for reconnaissance.
- RHOST was set to ‘192.168.2.119’; this tells Metasploit the address of the target DNS server.
- SRCADDR was set to ‘real’; this tells Metasploit the source address it should use for sending queries to the target DNS server. Real and random are accepted. In this practical work, real was used to make traffic investigation easier, in a real-world attack this would be set to random to protect the attacks identity.
- SRCPORT was set to ‘1053’; this tells Metasploit the target DNS server's source query port. If set to 0 the tool can detect this automatically.
- TTL was set to ‘604800’ (7 days); this tells Metasploit the TTL for the malicious entry.
- XIDS was set to ‘0’; this tells Metasploit the number of transaction IDs to try for each query. If set to 0 the tool detects this automatically.

8.1.5. Running the attack

On issuing the ‘run’ command, the exploit tool sent queries for non-existent hostnames at *richagar.co.uk*, (such as *ujxG0HtMUK.richagar.co.uk*; a packet capture of one of these queries is shown in Appendix G). The query for non-existent hostnames forced the vulnerable server to query the authoritative nameserver. The latency introduced by the wide area simulator gave a window of opportunity for the exploit tool to send spoofed responses to the vulnerable DNS server. An example of the spoofed responses is shown in Appendix H; each spoofed response included:

- Spoofed source IP address of the authoritative nameserver.
- UDP source port of 1053.
- Guessed transaction ID.
- Answers to the queries.
- Authoritative nameserver records of *ns.poisoned.com* for *richagar.co.uk*.
- Glue record of 221.143.12.218 for *ns.poisoned.com*.

This process was repeated until a match of the transaction ID was found. At this point the spoofed DNS response was accepted by the vulnerable server; it returned a response to the exploit tool,

which then stopped the attack. At that point the exploit tool displayed a confirmation; this is shown in Appendix I.

8.1.6. After the attack

The NS RR for *richagar.co.uk* at the target DNS server was poisoned. The answer section in a dig response showed:

- Authoritative nameserver records of *ns.poisoned.com* for *richagar.co.uk*.
- Glue record of 221.143.12.218 for *ns.poisoned.com*.

All subsequent queries for hostnames (for example MX and A RRs) in the *richagar.co.uk* domain that are not within TTL in the cache on the vulnerable server would then be sent to the attacker's nameserver. The full dig output for *richagar.co.uk* after the attack is shown in Appendix J.

8.1.7. Analysis

This practical work was successful in achieving its aims and shows that it is 'easy' to poison NS records on a DNS server that does not use source port randomisation. Any DNS servers not using randomised source ports should be patched to reduce the risk of such an attack.

8.2. Extending the cache poisoning attack to obtain and use SSL certificates

8.2.1. Aim of the attack

The aim of this section is to demonstrate applications that rely on DNS data being correct. It does this by:

- Showing that it is possible to obtain an SSL certificate with e-mail as the only authentication (which can be redirected, see 4.2.1). Thus showing that obtaining SSL certificates requires DNS data to be correct.
- Creating a spoofed copy of a website, using an SSL certificate obtained using e-mail as the only authentication and redirecting traffic to the spoofed web server. Thus showing that visitors of SSL websites require DNS data to be correct.

8.2.2. Lab set-up

To conduct the attack a client host and two web servers were added to the lab, as shown in figure 16.

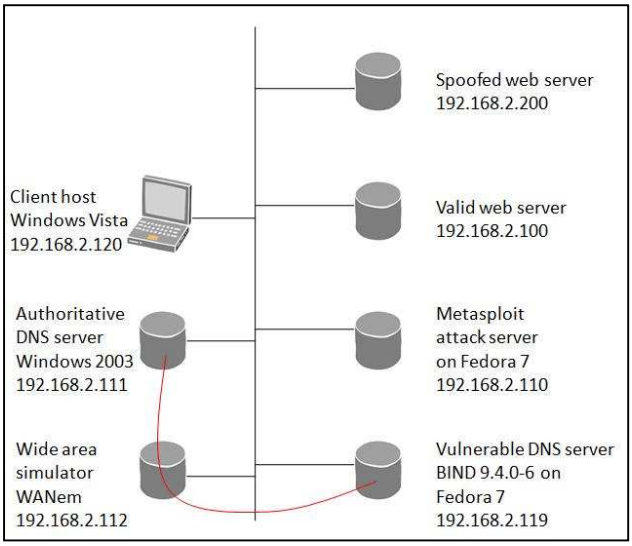

Figure 16 – Cache poisoning attack lab with SSL

8.2.3. Before the attack

The hostname *www.richagar.co.uk* resolved to 192.168.2.100 (see Appendix K for dig output), an SSL connection to port 443 was authenticated by the correct and valid certificate. It was possible to navigate to *www.richagar.co.uk* using SSL without being prompted with any certificate warnings (figure 17).

Figure 17 – Web page from valid web server with SSL

The certificate details were inspected and indicated that the certificate '*ensures the identity of the remote computer*', as shown in figure 18.

Figure 18 – Certificate details from valid web server with SSL

8.2.4. Preparing for the attack

A spoofed copy of the valid web server was created in the lab on a server with IP address 192.168.2.200.

A certificate request was generated on the spoofed web server and was used to request a free certificate for *www.richagar.co.uk* from the *www.instantssl.com* website. The process for authentication was fully automated; all that was required was the ability to receive an e-mail at the domain the certificate was being requested for. An e-mail was sent from the CA, including a validation code and a link; clicking on the link and entering the validation code was deemed sufficient authentication by the CA and the certificate was automatically issued by e-mail. The certificate was installed on the spoofed web server.

No attack was launched here; the owner of the domain requested a certificate. However, if an attacker had been able to poison the DNS server used by the CA, then the attack shown in 8.1 could have been used to enable the attacker to poison MX records, and obtain a certificate for a domain he does not own.

The Metasploit framework bailiwicked_host attack tool was used; it was configured as in 8.1.4, however the host attack tool replaces DOMAIN with HOSTNAME and NEWDNS with NEWADDR. The new options were configured as:

- HOSTNAME was set to '*www.richagar.co.uk*'; this tells Metasploit the host to hijack.
- NEWADDR was set to '192.168.2.200'; this tells Metasploit the IP address to use.

8.2.5. Running the attack

On issuing the 'run' command, the attack started. The attack ran as for the domain version of the attack, however rather than poisoning NS records the A records were poisoned.

8.2.6. After the attack

The A record for *www.richagar.co.uk* at the target DNS server was poisoned. The answer section in a dig response showed:

- The A record for *www.richagar.co.uk* was poisoned to 192.168.2.200.
- The TTL value had been increased to 602800.

All subsequent queries for *www.richagar.co.uk* to the vulnerable server contained the spoofed server's IP address, rather than the correct IP address. This was true for the duration of the extended TTL of 7 days; the previous TTL had been 1 hour. The full dig output for *www.richagar.co.uk* after the attack is shown in Appendix L.

An SSL connection to port 443 was authenticated by certificate (the one obtained by e-mail). It was possible to navigate to *www.richagar.co.uk* using SSL without being prompted with any certificate warnings, as shown in figure 19.

Figure 19 – Web page from spoofed web server with SSL

Identity assurance was gained by inspecting the certificate details to see if it provided assurance of the identity, this is shown in figure 20 (the certificate is included in Appendix M).

Figure 20 – Certificate details from spoofed web server with SSL

8.2.7. Analysis

This practical work was successful in achieving its aims and shows applications rely on DNS data to be correct. The aim of this project report is to look at DNS issues, rather than those of CAs. However, as it is now possible to obtain a free certificate (meaning no authentication through (say) a credit card account) and only e-mail as an authentication mechanism, the CA application is valid. Clearly using e-mail as the only authentication mechanism in granting a certificate is not satisfactory. Even a security savvy visitor to *www.richagar.co.uk* would find it difficult to tell the difference between the two certificates as both are trusted by browsers (both IE and Firefox were tested).

These examples show that users of any applications that rely on DNS need additional mechanisms to validate the IP addresses supplied by the DNS. If this is to be DNSSEC, the writer hopes that the authentication techniques used for validating the zone administrator when requesting signing of his KSK are stronger that those of the CA issuing this certificate, and those used by some registrars.

9. CONCLUSION

At the outset the intended objectives were:

1. **Provide a DNS overview, including the origins of the DNS, what the DNS is and how it works.**

   This objective was achieved in sections 2 and 3. Section 2 discusses the early days of the ARPAnet and the hosts.txt system it used originally, the issues of the hosts.txt system and why the DNS was (and is) needed. Section 3 looks at the components of the DNS, how records are stored, and then provides a walkthrough of a DNS query and response to show how the components of the DNS work together.

2. **Investigate DNS vulnerabilities, including a practical example of a DNS exploit and the security services required for the DNS.**

   This objective was achieved in sections 4, 5, 6, 7 and 8. Section 4 investigates DNS issues affecting clients, servers, registrars and also looks at denial of service. Section 5 looks into cache poisoning, specifically the Kaminsky vulnerability, in more detail. Section 7 looks at DNS amplification attacks. A lab was built to demonstrate the viability of the Kaminsky attack, and to extend that by looking at applications that rely on DNS data being correct; this is covered in section 8. Section 4 includes the services required by DNS, and when discussing DNS issues suggestions for mitigation are made. Sections 6 and 7.3 provide suggested mitigation methods for cache poisoning and amplification attacks respectively.

3. **Discuss DNS security improvements, including how DNS has been, and can be, improved.**

   This objective was achieved in sections 4, 5, 6 and 7. When discussing DNS issues in section 4 suggestions for mitigation are made. Section 5 explains the immediate response to the Kaminsky vulnerability and suggests a way of improving resistance to cache poisoning attacks. Section 7 makes suggestions for mitigating amplification attacks. DNSSEC is covered

in section 6, as is DNSCurve; both are possible solutions that could be used to provide a cryptographic solution to cache poisoning attacks.

This project report has detailed an investigation into the DNS, why DNS was (and is) needed, how it works, and has looked at the security challenges that the DNS has faced and mechanisms to address those challenges. This project report has aimed to consolidate some of the preceding work in this area to provide a background of DNS security issues for the reader before moving on to look at cache poisoning and amplification attacks in more detail.

Weaknesses in clients, servers, configuration, people and the DNS protocol allow a range of different attacks causing confidentiality, integrity and availability issues to applications that rely on the DNS.

Cache poisoning attacks can easily be conducted against servers not patched against the Kaminsky vulnerability; whilst the industry response has provided a tactical solution, cache poisoning attacks are still possible. A strategic solution is needed that provides a cryptographic response to cache poisoning. DNSSEC provides this and seems to be gaining traction; it does, however, suffer from issues that may slow its adoption. DNSCurve provides a simpler solution, does not suffer from zone walking issues, and has a lower impact on amplification attacks. DNSCurve does, however, provide less assurance than DNSSEC.

The DNS is vulnerable to DOS attacks; amplification attacks can generate multi-gigabit attacks that can be used against any target, including the DNS. The large records used in DNSSEC make amplification attacks easier.

The authentication used by at least one CA is inadequate. Authentication by email may be vulnerable to a DNS poisoning attacks, and may allow an attacker to falsely obtain a public key certificate that is trusted by browsers by default.

Further work is required to investigate how improvements can be made to DNS to mitigate cache poisoning attacks, whilst addressing the concerns of amplification and other DOS attacks. The writer would split this into 4 areas:

- Investigate changes to DNSSEC to address the issues it has today.
- Investigate changes to DNSCurve to provide greater assurance.
- Investigate alternatives to DNSSEC and DNSCurve to secure the DNS.
- Investigate a complete replacement for DNS that does not need to maintain backwards compatibility.

10. BIBLIOGRAPHY

References

[ARE05a]
R Arends, R Austein, M Larson, D Massey, S Rose
RFC 4033 DNS security introduction and requirements
http://www.ietf.org/rfc/rfc4033.txt
March 2005

[ARE05b]
R Arends, R Austein, M Larson, D Massey, S Rose
RFC 4034 Resource records for the DNS security extensions
http://www.ietf.org/rfc/rfc4034.txt
March 2005

[BAK04]
F Baker, P Savola
RFC 3704, Ingress filtering for multihomed networks
http://www.ietf.org/rfc/rfc3704.txt
March 2004

[BEL06]
A Bellissimo, J Burgess, K Fu
Secure software updates: disappointments and new challenges
http://www.cs.umass.edu/~kevinfu/papers/secureupdates-hotsec06.pdf
Last accessed: 2 September, 2009

[BER09a]
D Bernstein
DNSCurve website
http://dnscurve.org/
22 June, 2009

[BER09b]
D Bernstein
DNSCurve website
http://dnscurve.org/amplification.html
30 June, 2009

[CHI08]
R Chiodi, E Florio
Trojan.Flush.M
December 3, 2008
http://www.symantec.com/security_response/writeup.jsp?docid=2008-120318-5914-99

[CRO04]
S Crocker
Presentation to INET 2004 conference, Internet infrastructure security and stability
http://www.isoc.org/isoc/conferences/inet/04/presentations.shtml
May 2004

[DAG08]
D Dagon, M Antonakakis, P Vixie, T Jinmei, W Lee
15th ACM conference on computer and communications security, increased DNS forgery resistance through 0x20-bit encoding
October 2008

[DOR09]
W Dormann, C Dougherty
CERT vulnerability note VU#725188, ISC BIND 9 vulnerable to denial of service via dynamic update request
https://www.kb.cert.org/vuls/id/725188
July 28, 2009

[DOU08]
C Dougherty
CERT vulnerability note VU#800113, multiple DNS implementations vulnerable to cache poisoning
http://www.kb.cert.org/vuls/id/800113
July 8, 2008

[EAS97]
D Eastlake, C Kaufman
RFC 2065 Domain name system security extensions
http://www.ietf.org/rfc/rfc2065.txt
January 1997

[FER00]
P Ferguson, D Senie
RFC 2827, Network ingress filtering: defeating denial of service attacks which employ IP source address spoofing
http://www.ietf.org/rfc/rfc2827.txt
May 2000

[FOR09]
B Forbes, C Boutin, NIST website
Commerce department to work with ICANN and VeriSign to enhance the security and stability of the Internet's domain name and addressing system
http://www.nist.gov/public_affairs/releases/dnssec_060309.html
June 3, 2009

[GAB09]
E Gabrilovich, A Gontmakher
Technical report, the homograph attack
http://www.cs.technion.ac.il/~gabr/papers/homograph_full.pdf
Last accessed: 2 September, 2009

[GIO07]
R Giobbi
CERT vulnerability note VU#221876, Apple Mac OS X mDNSResponder buffer overflow vulnerability
May 25, 2007
https://www.kb.cert.org/vuls/id/221876

[GIO09]
R Giobbi
CERT vulnerability note VU#319331, Microsoft Windows DNS server response validation vulnerability
http://www.kb.cert.org/vuls/id/319331
March 10, 2009

[GOO08]
D Goodin, The Register
Patched DNS servers still vulnerable to cache poisoning
http://www.theregister.co.uk/2008/08/11/cache_poisoning_threat_remains/
August 11, 2008

[HOL03]
J Holmblad
The evolving threats to the availability and security of the domain name service
SANS GIAC/GSEC Practical
Part of the Information Security Reading Room
October 5, 2003

[HOL08]
T Holz, C Gorecki, F Freiling, K Rieck
Measuring and detecting fast-flux service networks
http://pi1.informatik.uni-mannheim.de/filepool/research/publications/fast-flux-ndss08.pdf
2008
Last accessed: 2 September, 2009

[HUB06]
A Hubert, R van Mook
Internet draft, measures to prevent DNS spoofing draft-hubert-dns-anti-spoofing-00.txt
http://tools.ietf.org/html/draft-hubert-dns-anti-spoofing-00
August 14, 2006

[ICA05]
ICANN security and stability advisory committee report
Domain name hijacking - incidents, threats, risks, and remedial actions
12 July, 2005

[ICA06]
ICANN security and stability advisory committee report
DNS distributed denial of service, (DDoS) attacks
March 2006

[ICA07]
ICANN factsheet
Root server attack on 6 February 2007
1 March, 2007

[ISC08]
ISC mailing list
Kaminsky vulnerability mailing list FAQ
https://lists.isc.org/pipermail/bind-users/2008-July/071835.html
July 31, 2008

[ISC09a]
Internet Systems Consortium
Web page on the "F" root domain server
https://www.isc.org/community/f-root
Last accessed: 2 September, 2009

[ISC09b]
Internet Systems Consortium
BIND dynamic update DoS
https://www.isc.org/node/474
July 28, 2009

[JAC09]
C Jackson, A Barth, A Bortz, W Shao, D Boneh
Protecting browsers from DNS rebinding attacks
ACM transactions on the Web, Vol. 3, No. 1, Article 2,
Publication date: January 2009

[LAR09]
Matt Larson, VeriSign
Presentation to Internet Society panel, Stockholm, Sweden: VeriSign's DNSSEC plans for .com, .net and the root
July 28, 2009

[LAU08]
B Laurie, G Sisson, R Arends, D Blacka
RFC 5155 DNS security (DNSSEC) hashed authenticated denial of existence
http://www.ietf.org/rfc/rfc5155.txt
March 2008

[LIU06]
C Liu, P Albitz
DNS and BIND, 5th edition
O'Reilly Media, Sebastapol
May 2006

[LIU09]
C Liu, Infoblox
A closer look at threats to the domain name system
Vendor webinar presentation
June 2009

[LOT87]
M Lottor
RFC 1033 Domain administrators operations guide
http://www.ietf.org/rfc/rfc1033.txt
November 1987

[MAN02]
A Manion
CERT vulnerability note VU#542971, Multiple vendors' domain name system (DNS) stub resolvers vulnerable to buffer overflow via network name and address lookups
August 1, 2002
https://www.kb.cert.org/vuls/id/542971

[MIC06]
Microsoft
Microsoft security bulletin MS06-041, Vulnerabilities in DNS resolution could allow remote code execution
September 13, 2006
http://www.microsoft.com/technet/security/bulletin/ms06-041.mspx

[MIC08]
Microsoft
Microsoft security bulletin MS08-037: Vulnerabilities in DNS could allow spoofing
http://support.microsoft.com/kb/953230
July 8, 2008

[MIC09]
Microsoft
Microsoft security bulletin MS09-008, Vulnerabilities in DNS and WINS server could allow spoofing
http://www.microsoft.com/technet/security/Bulletin/MS09-008.mspx
March 10, 2009

[MOC87a]
P Mockapetris
RFC 1034, Domain names – concepts and facilities
http://www.ietf.org/rfc/rfc1034.txt
November 1987

[MOC87b]
P Mockapetris
RFC 1035 Domain names - implementation and specification
http://www.ietf.org/rfc/rfc1035.txt
November 1987

[OLI08]
P Oliveria
Targeted attack in Mexico, part 2: yet another drive-by pharming
March 5, 2008
http://blog.trendmicro.com/targeted-attack-in-mexico-part-2-yet-another-drive-by-pharming/

[PRU06]
J Pruszynski
CERT vulnerability note VU#794580, Microsoft DNS client buffer overflow
August 8, 2006
http://www.kb.cert.org/vuls/id/794580

[STA06]
S Stamm, Z Ramzan, M Jakobsson
Drive-by pharming
Technical report
December 13, 2006

[THO95]
S Thomson, C Huitema
RFC 1886 DNS extensions to support IP version 6
http://www.ietf.org/rfc/rfc1886.txt
December 1995

[TMF08]
The Measurement Factory
DNS survey: October 2008
http://dns.measurement-factory.com/surveys/200810.html
October 2009

[VAN03]
S Vanstone, Certicom
Next generation security for wireless: elliptic curve cryptography
http://www.compseconline.com/hottopics/hottopic20_8/Next.pdf
2003

[VIX02]
P Vixie, G Sneeringer, M Schleifer
Events of 21-Oct-2002
November 24, 2002
http://d.root-servers.org/october21.txt

[VIX99]
P Vixie
RFC 2671, Extension mechanisms for DNS (EDNS0)
http://www.ietf.org/rfc/rfc2671.txt
August 1999

[VIX00]
P Vixie, O Gudmundsson, D Eastlake, B Wellington
RFC 2845 Secret Key Transaction Authentication for DNS (TSIG)
http://www.ietf.org/rfc/rfc2845.txt
May 2000

[WAN09]
WANem website
http://wanem.sourceforge.net/
Last accessed: 2 September, 2009

[WEI07]
S Weiler
RFC 5074 DNSSEC Lookaside Validation (DLV)
http://www.ietf.org/rfc/rfc5074.txt
November 2007

[WIL03]
M Wilson and J Hash
NIST special publication 800-50, building an information technology security awareness and training program
October 2003

[ZDR07]
B Zdrnja
DNS changer trojan for Mac (!) in the wild
November 1, 2007
http://isc.sans.org/diary.html?storyid=3595

[ZEL06]
L Zeltser
An overview of the FreeVideo Player trojan
November 19, 2006
http://isc.sans.org/diary.html?storyid=1872

11. APPENDICES

Appendix A – Example zone file from RFC 1033 [LOT87]

```
SRI.COM.      IN    SOA     KL.SRI.COM. DLE.STRIPE.SRI.COM.
                    870407  ;serial
                    1800    ;refresh every 30 minutes
                    600     ;retry every 10 minutes
                    604800  ;expire after a week
                    86400   ;default of an hour

SRI.COM.            NS      KL.SRI.COM.
                    NS      STRIPE.SRI.COM.
                    MX      10      KL.SRI.COM.

;SRI.COM hosts

KL                  A       10.1.0.2
                    A       128.18.10.6
                    MX      10      KL.SRI.COM.

STRIPE              A       10.4.0.2

STRIPE              A       128.18.10.4
                    MX      10      STRIPE.SRI.COM.

NIC                 CNAME   SRI-NIC.ARPA.

Blackjack           A       128.18.2.1
                    HINFO   VAX-11/780      UNIX
                    WKS     128.18.2.1      TCP TELNET FTP

CSL                 A       192.12.33.2
                    HINFO   FOONLY-F4       TOPS20
                    WKS     192.12.33.2     TCP TELNET FTP SMTP FINGER
                    MX      10      CSL.SRI.COM.
```

Appendix B – Example of response suitable for amplification attack

```
; <<>> DiG 9.5.1-P2 <<>> +bufsize=4096 +dnssec any se @a.ns.se;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 9525
;; flags: qr aa rd; QUERY: 1, ANSWER: 25, AUTHORITY: 0, ADDITIONAL: 20
;; WARNING: recursion requested but not available
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;se.                    IN      ANY
;; ANSWER SECTION:
se.             172800 IN      SOA             catcher-in-the-rye.nic.se.
registry-default.nic.se. 2009082808 1800 1800 2419200 7200

se.             172800 IN      RRSIG SOA       5 1 172800 20
                090903052319 20 090828161809 33786 se.
        oxWtF4+VqCXq9tdj5l6hDRIkeV/uTub/p+hwlx6yRFu+kgJKw5bBijy1YCUgfajLZRZA
SuC9X6NP7easu4//KKzADMhfIULO0FG9nKAikdn0QXs3DufyYFPSHJdwu9y5+NS6vJXqC4k
kFnFBJES9a/9UMPwDMSqfF4uWrObl Cfc=
se.             172800 IN      NS              c.ns.se.
se.             172800 IN      NS              d.ns.se.
se.             172800 IN      NS              e.ns.se.
se.             172800 IN      NS              f.ns.se.
se.             172800 IN      NS              g.ns.se.
se.             172800 IN      NS              h.ns.se.
se.             172800 IN      NS              i.ns.se.
se.             172800 IN      NS              j.ns.se.
se.             172800 IN      NS              a.ns.se.
se.             172800 IN      NS              b.ns.se.

se.             172800 IN      RRSIG NS        5 1 172800
                20090902183531 20090828161809 33786 se.
        vq49yTd6u5bVx7t99ES9peFQ+OYjBWXOr3rx9lCM60cdEez2C2OjtQurPlyXzCPbA5C
QRJU7pbILGg3ZGxv76eOqt3TnDH9KGZZnn26+tLP1gVEwvrARaUzgJhaYFFk6AhpJ06XK+r
yV1TOMwFu/NOhUgRy7t+g5Q0HDIq8H /PQ=

se.             172800 IN      TXT
"SE zone update: 2009-08-28 17:09:38 +0000 (EPOCH 1251479378)"

se.             172800 IN      RRSIG TXT       5 1 172800
                20090902171109 20090828161809 33786 se.
        QyGgH1wsmeoVoSv4HquNReo8b3tKttxPnEo4Efng8oqsL1CZCy9uhz+68b1ga7QqD08
xSN/pji/9wWbiYpM9Cl9LVHdTRjXQ0xKhCbffbkZObwaBNb62GvN7DNR2IfbaU/snC06jiIZ0
QwSGfG/7IqeOFZf1od3EUZb5Gz4N peE=

se.             7200   IN      NSEC  0-0.se.
NS      SOA     TXT    RRSIG NSEC    DNSKEY
```

```
se.             7200    IN      RRSIG   NSEC    5       1       7200    20090903102340
                200     90828041809     33786   se.
        4Ipuw0Al82k9E5wF9kbkIMfuYSHU3WCx0mEo7ooWgUDu/G2v8Bm9znFgABj3kM2
a1BZ8ydYgp00T9RqQjFCMDXkTreDjaJZjv8eyj+pTBs0N+5JRFZNORGn6NHPJLh39Qx5TLa
YDrKRGRZkVheiQyWXuiHTDY7FZtHUTo2KSkvU=

se.             3600    IN      DNSKEY  257     3       5
        AwEAAeeGE5unuosN3c8tBcj1/q4TQEwzfNY0GK6kxMVZ1wcTkypSExLCBPMS0w
WkrA1n7t5hcM86VD94L8oEd9jnHdjxreguOZYEBWkckajU0tBWwEPMoEwepknpB14la1wy3
xR95PMt9zWceiqaYOLEujFAqe6F3tQ14lP6FdFL9wyCflV06K1ww+gQxYRDo6h+Wejguvpeg
33KRzFtlwvbF3AapH2GXCi4Ok2+PO2ckzfKoikIe9ZOXfrCbG9ml2iQrRNSM4q3zGhuly4NrF/
t9s9jakbWzd4PM1Q551XIEphRGyqcbA2JTU3/mcUVKfgrH7nxaPz5DoUB7TKYyQgsTlc=

se.             3600    IN      DNSKEY  256     3       5
        AwEAAZzN62+7wyMKUs06PMCv2FV/mo7phe+7447lPcvJvOF2uZ0nou87Yp1waA2F
PR84VWmK5qduytrNGpmBl1HM2FsDbdllA+TdgJvGETXHkXHpxu5FkUtlTzdLmiKZ4Q/0Jm
HlftHGGQ5EBhbDCZbbtORXesvJvxnRDR6WCngcV5

se.             3600    IN      DNSKEY  256     3       5
        AwEAAc9B2MqxtXW/bQUYBV5zHf1z8e8MRSVWleDlQ3XZaOMlki4g4qdXhJ2rHN
69ZzmvMj2DjTvLd8lMTN54NiHntofBIg0ZJ9pi0pcDemeeVQ7Bh7T1OYzaM2rnId7xfSaoXBd
X+6tMhkB/3CXM/cgqgar33emkhzdPUDdS8FYNBGuL

se.             3600    IN      DNSKEY  256     3       5
        AwEAAeplhwBO0eadnSP6xMiFQyQ8PpVjHeutOzHSn5a47h48aPwnTUFpJzAWbHEl
XtweFU9g5gMtatNhHdofXz7ndsEoFArNS2M4DapSPaklrZGBSIdSMAUrhKsDPsmwIBqLHQ
gYVeAx4De4aEobnpTcVX/221w1+zpckwghF79trWZD

se.             3600    IN      DNSKEY  257     3       5
        AwEAAdKc1sGsbv5jjeJ141IxNSTdR+nbtFn+JKQpvFZETaY5iMutoyWHa+jCp0TBB
AzB2trGHzdi7E55FFzbeG0r+G6SJbJ4DXYSpiiELPiu0i+jPp3C3kNwiqpPpQHWaYDS9MTQ
Mu/QZHR/sFPbUnsK30fuQbKKkKgnADms0aXalYUuCgDyVMjdxRLz5yzLoaSO9m5ii5cI0d
QNCjexvj9M4ec6woi6+N8v1pOmQAQ9at5Fd8A6tAxZI8tdlEUnXYgNwb8eVZEWsgXtBhoyA
ru7Tzw+F6ToYq6hmKhfsT+fIhFXsYso7L4nYUqTnM4VOZgNhcTv+qVQkHfOOeJKUkNB8Q
c=
se.             3600    IN      RRSIG   DNSKEY  5       1 3600 20090901192500 2
                009082714181033786  se.
        3wIMg+6rjKkDnEu90KkYJDr7KXemFrUc3IxQTC8RzpjHw70REVNv8+mBDfwrVR1
EadWAw/KT6dkUS4Te6jKwIuYpEYdO1g6ZctfpYtZpmP4H9KOW7C6mxDK5HEGty0LD4O
VdlfUANc0xSzctWG5ZdP2FGu1AJqZsPIVNwnkisOg=

se.             3600    IN      RRSIG   DNSKEY  5       1 3600 20090914000000 2
                009080408053687 79  se.
        gGtYyS7qZ89UgHHxzyfs754dtdpbipQV+SfwLxWDsBWNz8qZ2RHO82vYaqEBVcA3
xo1onmRdDrijFvy28//HmcC3Jb7ZkMh5UAxo+GFHMZnSo0aJUhHGNsy2Ul2YjqRUKJoOml
MkX/GhcrOUCZFkgmZPEalrlk74nnYSdB5M4KbdCGJM2Wm82X5gO7ahe+ozqfGOu+pxM2
wMOgdthFSOYkA1QVjKLkgrfc8/xo04WWCHudz/ldmng4myVzE4Zv4TJYuqOk9HXjKmdtW
IsIQlrCWP2SU2lFvElGJ/DvOMPj2z1QtPpbjNb8+q8LpahhJlljVSjG14mnqxKob0 LLC0QQ==

se.             3600    IN      RRSIG   DNSKEY  5       1 3600 20090914000000 2
                009080408053649678  se.
```

oiRILCvgKW7Eqhf25cjXY6OChTW5/F2zfNrXHn56/dLYLDaXkc0NqrkBGhT3iyEws7
SBHu59GmJR2OhK1LzdyH5ApSw6TpOaARVSl4XbJD8GeyENoBqiZKaY705T+PQH/d8uU9
u3sT1N3CYsOGcWIML0r4M9WGLEVri2n45ffGu9JPP9gnxRrRw38Gb8R2/ucbIN6rJOBFZU4
D9FqtQOev/7NkO+Pk57MUalqoM5//lnQl6VKJiGSWuv2ineCk8H4dspWp2kRdH/SPUwXmRR
IrBkLM2dars9dbqh3YCQ3s9uFuAwvsBXAcozVrkuv5rg2WtOBAXTzsxGUbdz uw++MA==

;; ADDITIONAL SECTION:

a.ns.se.	172800	IN	A				192.36.144.107
a.ns.se.	172800	IN	AAAA				2a01:3f0:0:301::53
b.ns.se.	172800	IN	A				192.36.133.107
c.ns.se.	172800	IN	A				192.36.135.107
d.ns.se.	172800	IN	A				81.228.8.16
e.ns.se.	172800	IN	A				81.228.10.57
f.ns.se.	172800	IN	A				192.71.53.53
g.ns.se.	172800	IN	A				130.239.5.114
g.ns.se.	172800	IN	AAAA				2001:6b0:e:3::1
h.ns.se.	172800	IN	A				199.7.49.30
i.ns.se.	172800	IN	A				194.146.106.22
j.ns.se.	172800	IN	A				199.254.63.1
j.ns.se.	172800	IN	AAAA				2001:500:2c::1

a.ns.se. 172800 IN RRSIG A 5 3 172800 20090903235813
20090828161809 33786 se.
4jx+TQyFcqmIfNw01hReOeyobS4VrCAPmBUo2rVSJS3LR8R0jMiPZDEo2VvFLl0v2
pHC3ZLiNo6aIVChstXkkbH3TcDEhgOLvgNWyym6sOICZoeQGA0z3V2z5ahCLYxD0oRtnF
ZQSCiAR2VomtTAOSDV67pPyYGJn7I+A/X0MbY=

a.ns.se. 172800 IN RRSIG AAAA 5 3 172800 20090904075134
20090828161809 33786 se.
Ib6GpsMY8KVWEOWh6ZVDix/ApyJbB7LHD4eEtCT9+9FchlxJMP49turJLLqd3RWu
7PnwZP5s66Gd+m88A8wi5WtdId3FfbMIbNNw7UXG52BR/ozpZh/MA2mfgH6E/g4hffI8VPno
51ceKyR4shHCSxgxyYKw3MIAPvv5dpgIwPA=

b.ns.se. 172800 IN RRSIG A 5 3 172800 20090904041214
20090828161809 33786 se.
vLrTUk/Shf1C3kpYLEH6OK4i215bOLYA9LFGE3gzl0OsPW86giTl5DzxwpdX2puk3n
VaEDs340VqcpqT6U7obzG2yUM1vshDJzcr/Gzbqv0PTkOMQA3quxCvUKhneDleAbb0rg9GE
chnIVbVm/tKjyO0o8XioGYIW2KCL+bdgwE=

c.ns.se. 172800 IN RRSIG A 5 3 172800 20090903235628
20090828161809 33786 se.
OY/AVwmsa9mjj3xUXNXnFw3jhP9tmJrltiFulJDSE7B1oRanVBs9CHKJ/kJ6N4HP9Y
UEBzrkIaLLWB2V69y2DIjWMut2YloPoJ76h3vkuT6icwZZPdZaVjAtWheicfp1r0VXloF4KpW
PLa96gOWdLyME+FnBMyttJF/bYcjIY7I=

d.ns.se. 172800 IN RRSIG A 5 3 172800 20090902164457
20090828161809 33786 se.
I3ZSNRDE0Fst7jdV70M5egYjiICwQrGleFwmgKHh/c9IDJwKzizJfSG/RUUFTuQ4bVl
0b7zv3MlKmfks2pM2brH16cIZtFERAn2L0M1+HPsueWAvVeOkKRcIKdkXoLvJwUtBp7YqY
B/JNCFR3EwqAeVP8GTuaxqONkb/9RKQl9k=

```
e.ns.se. 172800 IN RRSIG A 5 3 172800 20090903230122
                20090828161809 33786 se.
        21YKABd9/PDMHO0kbi8Az4xF/Od7Ps1GrBNGPKjYJYVKmlwkiBBnwHhsHJxdTS4
foCa52L+2tUkH59omulVsKGVXLRnX7fxbI3PONV9PZcwz2FpYtDyKjY9YAf6rh58zb4gY7d
AEWUYijEzQA+cLQgekgZMzjOzgztVpq8s1Veo=

;; Query time: 76 msec
;; SERVER: 192.36.144.107#53(192.36.144.107)
;; WHEN: Fri Aug 28 18:52:50 2009
;; MSG SIZE  rcvd: 3974
```

Appendix C – Response to a DNSSEC query from the OpenDNS service

```
; <<>> DiG 9.5.1-P2 <<>> +bufsize=4096 +dnssec any org
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 28991
;; flags: qr rd ra; QUERY: 1, ANSWER: 7, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;org.                    IN      ANY
;; ANSWER SECTION:
org.                    172800  IN      NS      B0.ORG.AFILIAS-NST.org.
org.                    172800  IN      NS      D0.ORG.AFILIAS-NST.org.
org.                    172800  IN      NS      A0.ORG.AFILIAS-NST.INFO.
org.                    172800  IN      NS      B2.ORG.AFILIAS-NST.org.
org.                    172800  IN      NS      C0.ORG.AFILIAS-NST.INFO.
org.                    172800  IN      NS      A2.ORG.AFILIAS-NST.INFO.
org.                    590     IN      SOA     A0.ORG.AFILIAS-NST.INFO.    noc.AFILIAS-
NST.INFO. 2008795754 1800 900 604800 86400
;; Query time: 24 msec
;; SERVER: 208.67.222.222#53(208.67.222.222)
;; WHEN: Sat Aug 29 17:04:20 2009
;; MSG SIZE  rcvd: 210
```

Appendix D – Response to a DNSSEC query from the Plusnet DNS servers

```
; <<>> DiG 9.5.1-P2 <<>> +bufsize=4096 +dnssec any org @212.159.13.49
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 59859
;; flags: qr rd ra; QUERY: 1, ANSWER: 7, AUTHORITY: 6, ADDITIONAL: 7
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;org.                    IN      ANY
;; ANSWER SECTION:
org.            69423   IN      NS      b0.org.afilias-nst.org.
org.            69423   IN      NS      b2.org.afilias-nst.org.
org.            69423   IN      NS      c0.org.afilias-nst.info.
org.            69423   IN      NS      d0.org.afilias-nst.org.
org.            69423   IN      NS      a0.org.afilias-nst.info.
org.            69423   IN      NS      a2.org.afilias-nst.info.
org.             74228  IN      RRSIG   NS 7 1 86400 20090908154424 20090825144424 63488
org.                    wue3le4jS64Yjj975nX7/mNA9VlmI/aINPwXArQSK17Ec1gAY3qVStVJ
XY2X0Kj+EFEK7aJLlUp7zbQ3tK/RfS8eOppVYD0ce+VlXGBbwkbPDPxO
3ztHH22BdzK/EHhD2Dlqsb+N1DkHsVlm0XAUMoBqzWb+9PrLxeZ5BE9N RDY=
;; AUTHORITY SECTION:
org.            69423   IN      NS      a2.org.afilias-nst.info.
org.            69423   IN      NS      b0.org.afilias-nst.org.
org.            69423   IN      NS      b2.org.afilias-nst.org.
org.            69423   IN      NS      c0.org.afilias-nst.info.
org.            69423   IN      NS      d0.org.afilias-nst.org.
org.            69423   IN      NS      a0.org.afilias-nst.info.
;; ADDITIONAL SECTION:
a2.org.afilias-nst.info. 3824   IN      A       199.249.112.1
a2.org.afilias-nst.info. 3824   IN      AAAA    2001:500:40::1
b2.org.afilias-nst.org. 2001    IN      A       199.249.120.1
b2.org.afilias-nst.org. 28237   IN      AAAA    2001:500:48::1
c0.org.afilias-nst.info. 75341  IN      A       199.19.53.1
c0.org.afilias-nst.info. 1692   IN      AAAA    2001:500:b::1
;; Query time: 28 msec
;; SERVER: 212.159.13.49#53(212.159.13.49)
;; WHEN: Sat Aug 29 17:06:52 2009
;; MSG SIZE  rcvd: 549
```

Appendix E – Latency test for cache poisoning practical work

The following was executed on the caching server to validate that WANem was adding latency between the caching server and the authoritative server:

```
[root@dns etc]# ping 192.168.2.111 -c 10
PING 192.168.2.111 (192.168.2.111) 56(84) bytes of data.
64 bytes from 192.168.2.111: icmp_seq=1 ttl=127 time=205 ms
64 bytes from 192.168.2.111: icmp_seq=2 ttl=127 time=205 ms
64 bytes from 192.168.2.111: icmp_seq=3 ttl=127 time=206 ms
64 bytes from 192.168.2.111: icmp_seq=4 ttl=127 time=206 ms
64 bytes from 192.168.2.111: icmp_seq=5 ttl=127 time=203 ms
64 bytes from 192.168.2.111: icmp_seq=6 ttl=127 time=204 ms
64 bytes from 192.168.2.111: icmp_seq=7 ttl=127 time=205 ms
64 bytes from 192.168.2.111: icmp_seq=8 ttl=127 time=205 ms
64 bytes from 192.168.2.111: icmp_seq=9 ttl=127 time=205 ms
64 bytes from 192.168.2.111: icmp_seq=10 ttl=127 time=206 ms

--- 192.168.2.111 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 8995ms
rtt min/avg/max/mdev = 203.655/205.493/206.484/0.903 ms
[root@dns etc]#
```

Appendix F – dig output for richagar.co.uk prior to attack in section 8.1

```
[root@dns etc]# dig @192.168.2.119 richagar.co.uk NS
; <<>> DiG 9.4.0 <<>> @192.168.2.119 richagar.co.uk NS
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 27569
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; QUESTION SECTION:
;richagar.co.uk.                IN      NS
;; ANSWER SECTION:
richagar.co.uk.         2272    IN      NS      ns.richagar.co.uk.
;; ADDITIONAL SECTION:
ns.richagar.co.uk.      2272    IN      A       192.168.2.111
;; Query time: 10 msec
;; SERVER: 192.168.2.119#53(192.168.2.119)
;; WHEN: Sun Aug 30 17:46:58 2009
;; MSG SIZE  rcvd: 65
[root@dns etc]#
```

Appendix G – Metasploit generated query

Internet Protocol, Src: 192.168.2.110, Dst: 192.168.2.119
User Datagram Protocol, Src Port: 44293 (44293), Dst Port: domain (53)
Domain Name System (query)
Transaction ID: 0xd11f
    Questions: 1
        Answer RRs: 0
        Authority RRs: 0
        Additional RRs: 0
    Queries
        ujxG0HtMUK.richagar.co.uk: type A, class IN
        Name: ujxG0HtMUK.richagar.co.uk
        Type: A (Host address)

Appendix H – Metasploit generated spoofed response

Internet Protocol, Src: 192.168.2.111, Dst: 192.168.2.119
User Datagram Protocol, Src Port: domain (53), Dst Port: remote-as (1053)
Domain Name System (response)
Transaction ID: 0x87ab
Questions: 1
Answer RRs: 1
Authority RRs: 1
Additional RRs: 1
Queries
ujxG0HtMUK.richagar.co.uk: type A, class IN
    Name: **ujxG0HtMUK.richagar.co.uk**
    Type: A (Host address)
Answers
ujxG0HtMUK.richagar.co.uk: type A, class IN, addr 221.143.12.218
Name: ujxG0HtMUK.richagar.co.uk
    Type: A (Host address)
    Time to live: 7 days
    Addr: 221.143.12.218
Authoritative nameservers
        richagar.co.uk: type NS, class IN, ns ns.poisoned.com
            Name: richagar.co.uk
            Type: NS (Authoritative name server)
            Name server: ns.poisoned.com
    Additional records
        ns.poisoned.com: type A, class IN, addr 221.143.12.218
            Name: ns.poisoned.com
            Type: A (Host address)
            Time to live: 7 days
            Addr: 221.143.12.218

Appendix I – Metasploit output after the attack

msf auxiliary(bailiwicked_domain) > run
[1] Targeting nameserver 192.168.2.119 for injection of richagar.co.uk. nameservers as ns.poisoned.com
[2] Querying recon nameserver for richagar.co.uk.'s nameservers...
[3] Got an NS record: richagar.co.uk. 3600 IN NS ns.richagar.co.uk.
[4] Querying recon nameserver for address of ns.richagar.co.uk....
[5] Got an A record: ns.richagar.co.uk. 3600 IN A 192.168.2.111
[6] Checking Authoritativeness: Querying 192.168.2.111 for richagar.co.uk....
[7] ns.richagar.co.uk. is authoritative for richagar.co.uk., adding to list of nameservers to spoof as
[8] Calculating the number of spoofed replies to send per query...
[9] race calc: 100 queries | min/max/avg time: 0.2/0.31/0.23 | min/max/avg replies: 34/89/58
[10] Sending 87 spoofed replies from each nameserver (1) for each query
[11] Attempting to inject poison records for richagar.co.uk.'s nameservers into 192.168.2.119:1053...
[12] Poisoning successful after 250 queries and 21750 responses: richagar.co.uk. = = ns.poisoned.com
[13] Auxiliary module execution completed
msf auxiliary(bailiwicked_domain) >

Appendix J – dig output for richagar.co.uk after the attack in section 8.1

```
[root@dns etc]# dig @192.168.2.119 richagar.co.uk NS
; <<>> DiG 9.4.0 <<>> @192.168.2.119 richagar.co.uk NS
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 47770
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; QUESTION SECTION:
;richagar.co.uk.                IN      NS
;; ANSWER SECTION:
richagar.co.uk.         604451  IN      NS      ns.poisoned.com.
;; ADDITIONAL SECTION:
ns.poisoned.com.        604451  IN      A       221.143.12.218
;; Query time: 10 msec
;; SERVER: 192.168.2.119#53(192.168.2.119)
;; WHEN: Sun Aug 30 19:53:53 2009
;; MSG SIZE  rcvd: 77
[root@dns etc]#
```

Appendix K – dig output for www.richagar.co.uk prior to attack in section 8.2

```
[root@dns ~]# dig www.richagar.co.uk @192.168.2.119
; <<>> DiG 9.4.0 <<>> www.richagar.co.uk @192.168.2.119
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 7682
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
; QUESTION SECTION:
;www.richagar.co.uk.            IN      A
;; ANSWER SECTION:
www.richagar.co.uk.     3600    IN      A       192.168.2.100
;; Query time: 434 msec
;; SERVER: 192.168.2.119#53(192.168.2.119)
;; WHEN: Mon Aug 31 13:56:32 2009
;; MSG SIZE  rcvd: 52
[root@dns ~]#
```

Appendix L - dig output for richagar.co.uk after the attack in section 8.2

```
[root@dns ~]# dig www.richagar.co.uk @192.168.2.119
; <<>> DiG 9.4.0 <<>> www.richagar.co.uk @192.168.2.119
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 43251
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0
;; QUESTION SECTION:
;www.richagar.co.uk.          IN      A
;; ANSWER SECTION:
www.richagar.co.uk.     601129  IN      A       192.168.2.200
;; Query time: 11 msec
;; SERVER: 192.168.2.119#53(192.168.2.119)
;; WHEN: Mon Aug 31 15:04:55 2009
;; MSG SIZE  rcvd: 66
[root@dns ~]#
```

Appendix M – Certificate obtained with only e-mail authentication in PEM format.

-----BEGIN CERTIFICATE-----
MIIEiTCCA3GgAwIBAgIRAP+ahBj1+eRuyh/6jWsLJ2AwDQYJKoZIhvcNAQEFBQAwcjEL
MAkGA1UEBhMCR0IxGzAZBgNVBAgTEkdyZWF0ZXIgTWFuY2hlc3RlcjEQMA4GA1UEB
xMHU2FsZm9yZDEaMBgGA1UEChMRQ09NT0RPIENBIExpbWl0ZWQxGDAWBgNVBAM
TD0Vzc2VudGlhbFNTTCBDQTAeFw0wOTA4MjkwMDAwMDBaFw0wOTExMjcyMzU5NTl
aMFMxITAfBgNVBAsTGERvbWFpbiBDb250cm9sIFZhbGlkYXRlZDERMA8GA1UECxMIR
nJlZSBTU0wxGzAZBgNVBAMTEnd3dy5yaWNoYWdhci5jby51azCBnzANBgkqhkiG9w0BA
QEFAAOBjQAwgYkCgYEA3KdxQCCgOW7xwUjwvQQyylNUhLDfpXxwyF2FUifrMz3Opb
MmMkcaQtZINcCC+Ay2Ilrf/HR4xWQPVZeOLiRiz3wVd2ZQVwPuUzlTzPutUjjfOrbvE7Zucq
n4oQDAW1OuMjMUm0rBHbKsV3oycWRrpfrInS5fGGMKuIeUpLj6oo8CAwEAAaOCAbswg
gG3MB8GA1UdIwQYMBaAFNrL6q1bCF3M//wmVM5J5VXGOPT4MB0GA1UdDgQWBBTb
wO22UgcK5h/ZzqF4+PDefodPLjAOBgNVHQ8BAf8EBAMCBaAwDAYDVR0TAQH/BAIwA
DA0BgNVHSUELTArBggrBgEFBQcDAQYIKwYBBQUHAwIGCisGAQQBgjcKAwMGCWC
GSAGG+EIEATBFBgNVHSAEPjA8MDoGCysGAQQBsjEBAgIHMCswKQYIKwYBBQUHA
gEWHWh0dHBzOi8vc2VjdXJlLmNvbW9kby5jb20vQ1BTMDsGA1UdHwQ0MDIwMKAuoCy
GKmh0dHA6Ly9jcmwuY29tb2RvY2EuY29tL0Vzc2VudGlhbFNTTENBLmNybDBuBggrBgE
FBQcBAQRiMGAwOAYIKwYBBQUHMAKGLGh0dHA6Ly9jcnQuY29tb2RvY2EuY29tL0V
zc2VudGlhbFNTTENBXzIuY3J0MCQGCCsGAQUFBzABhhhodHRwOi8vb2NzcC5jb21vZG9j
YS5jb20wLQYDVR0RBCYwJIISd3d3LnJpY2hhZ2FyLmNvLnVrgg5yaWNoYWdhci5jby51az
ANBgkqhkiG9w0BAQUFAAOCAQEAAeH7Tj12dUcn50SUo1w6QLOCT1JCk1G7TxSoyQoE
xrO+G+tCHCuEvpFcv8FvGCVSqLVDs/qu1b16JkzH5wJ1GStXZyq/dYKhZcsIeauF9xDq6vlhO
nvXFYwaVJrgkvkWTgwmwSwFFs5jHQSe/YSYY3l9zlSd0+9ILEuG0Bmf8RpqKZQdqndxUJJ
S8s5Y+GGIJqMQeqPoT/9JG8Cutgxvu3CW+wfSLEg4rdSd0zH3VZ2lPrtuxWHJu2XsvwGSMG
p2lOaOvBn7Xum/uE9U6zEE1LZb2DkbkuuntP2313w01gR8WUEKrZG9KFcruKnvboD/HpCI1/
z5zx9e+qhtRx4HrA==
-----END CERTIFICATE-----

END OF DOCUMENT