IMPROVING DNS SECURITY

A MEASUREMENT-BASED APPROACH

ROLAND VAN RIJSWIJK-DEIJ

www.example.com

Improving DNS Security A Measurement-Based Approach

Roland van Rijswijk-Deij

*Het verbeteren van de veiligheid van DNS
Een op metingen gebaseerde aanpak*

Roland van Rijswijk-Deij

Graduation committee:

Chairman/secretary: Prof. dr. P.M.G. Apers
Supervisor: Prof. dr. ir. A. Pras
Co-supervisor: Dr. A. Sperotto

Members:

Prof. dr. J.L. van den Berg, University of Twente, The Netherlands
Prof. dr. ir. L.J.M. Nieuwenhuis, University of Twente, The Netherlands
Prof. dr. K.C. Claffy, CAIDA, University of California, San Diego, USA
Prof. dr. J. Schönwälder, Jacobs University, Bremen, Germany
Dr. B.J. Overeinder, NLnet Labs, Amsterdam, The Netherlands

Funding sources:

SURFnet's GigaPort3 project for Next-Generation Networks
SURF's P8 Innovation Programme for a Secure and Trustworthy Network
EU FP7 Flamingo Network of Excellence – 318488
OpenINTEL project funded by SURFnet and SIDN
D3 - Distributed Denial-of-Service Defense – NWO project № 628.001.018

CTIT

CTIT Ph.D. thesis Series № 17-430
Centre for Telematics and Information Technology
P.O. Box 217, 7500 AE
Enschede, The Netherlands

SURFnet B.V.
Moreelsepark 48, 3511 EP
Utrecht, The Netherlands

ISBN 978-90-365-4329-3
ISSN 1381-3617 (CTIT Ph.D. thesis Series № 17-430)
DOI 10.3990/1.9789036543293
<https://doi.org/10.3990/1.9789036543293>

Type set with L^AT_EX. Printed by Gildeprint.
Cover design by Paul Eversdijk at Vrije Stijl Utrecht.

Copyright © 2017 Roland van Rijswijk-Deij
This work is licensed under a Creative Commons
Attribution-NonCommercial-ShareAlike 4.0 International License.
<https://creativecommons.org/licenses/by-nc-sa/4.0/>

IMPROVING DNS SECURITY: A MEASUREMENT-BASED APPROACH

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof. dr. T.T.M. Palstra,
volgens besluit van het College voor Promoties,
in het openbaar te verdedigen
op woensdag 28 juni 2017 om 14:45 uur

door

Roland Martijn van Rijswijk-Deij

geboren op 17 maart 1977
te Arnhem

Dit proefschrift is goedgekeurd door:

This thesis has been approved by:

Prof. dr. ir. A. Pras (promotor/*supervisor*)

Dr. A. Sperotto (co-promotor/*co-supervisor*)

Donderdag 23 maart 2017

Thursday March 23rd, 2017

Acknowledgements

Arguably, this is one of the most important parts of the thesis: where I thank all the people that have helped me through these four years. I start by thanking all of you whom I have forgotten to mention personally: it is entirely my fault, and my heartfelt apologies in advance!

Four years ago, I started working on a Ph.D. next to my job at SURFnet. I would like to start by thanking my employer SURFnet, and especially my manager at the time, Harold, for having confidence in me, and for letting me take on this challenge. Harold, I am overjoyed that you will be one of my paranymphs. I also take this opportunity to thank all of my colleagues at SURFnet for their genuine interest in my work. I would like to mention a few people in particular. Frans, Joost, Paul, Pieter, Remco, Rogier, Wim and Xander, we had a great time and a great team, MWS forever! Paul, thanks for all the hours of support with OpenStack. Xander, thank you for stimulating discussions about lies, damn lies, and, well, you know what I mean, and also for helping make sense of flow data. Remco, you are the best project manager one can wish for, always enthusiastic about my work and helpful and understanding in making the combination of my Ph.D. research and innovation projects work. Thanks also to Jac; you ‘inherited’ a team member frantically working on his Ph.D. and gave me the space I needed to complete this work.

Then I would like to thank all my friends and colleagues in the Digital Security group at the Radboud University in Nijmegen. I started my Ph.D. with you four years ago, only to change horses to Twente during the race. I enjoyed all the serious and not-so-serious conversations over coffee. Special thanks to Freek and Barbara for being my first roomies, and sharing tasty Nespresso. Also thanks to all the folks in the IRMA team: Bart, Fabian, Gergely, Jaap-Henk, Pim, Ronny and Wouter. IRMA was a fun project to work on! A final thank you to Erik Poll for being the first to take on the challenge of supervising me.

After one year of Ph.D., I suffered a bit of a crisis, finding the work rewarding, but very hard to combine with my ‘day job’. Fortunately, I managed to find a working solution by transferring from Nijmegen to Twente, where Aiko Pras and Anna Sperotto took me under their wing. Thank you both for supervising my work, and for taking me on board. I had a most wonderful time in Twente, at DACS, chatting over lunch and coffee, working together on research and having

fun during ‘DACS uitjes’. Thanks Jair, Luuk and Ricardo for collaborating on some nice papers. A very warm thank you to Mattijs, without you OpenINTEL would not have happened. I enjoyed spending time together in both Japan and California. And thank you for being my second paranymph and taking the time to travel all the way from sunny San Diego to (hopefully) sunny Enschede for my defense! Finally, Jeanette: you helped me and many of us navigate the bureaucracy of the university, and your laugh brightens all of our days.

A special note of thanks goes to the students who contributed to this thesis. Niels Monen did the first exploratory work on fragmentation under my supervision. Gijs, you were an extraordinary student. I enjoyed all our conversations, be it about DNSSEC and fragmentation, or about the state of the world. I have seldom seen someone so dedicated to his work, and have learned much from how you conduct yourself. If only everyone strived for the same standard of work and civilized behaviour. Dōmo arigatō gozaimasu! And finally Kaspar, who worked with me on Chapter 6. I hope you find a good group, in a nice place, to do your own Ph.D. They will be lucky to have your talent on board!

Next, I would like to express my gratitude to my committee, for taking the time to read through what has eventually become a small book. Special thanks to the members from abroad, Jürgen and kc, for making the journey to faraway Enschede to attend my defense.

Mom and dad, you are the source of all this. Your unconditional love and support has helped me throughout the years, and your decision to buy a home computer for your kid in the 1980s was visionary. Without that little seed, this thesis would never have existed. Ingrid and Kees: thank you for lovingly taking me in as your son in law. Petra, Andrew, Peter, Lieske and Martin: you are more important to me than I can express in words. And to all the rest of my family, you all contributed to my life and happiness, thank you. Last, and most importantly: Saskia and Simon, I dedicate this work to you, you are the bright shining stars of the future.

My final words are for Menno. Twenty years ago this year, we met in Enschede, at the same university I will now be defending my Ph.D. thesis at. You are the love of my life, you understand me in more ways than I can express and have always been there for me, in the good times, when we are travelling together, enjoying our shared passion for scuba diving deep under the surface of the ocean, and in the bad times, when work stress took its toll on me. You know like no other what it means to spend endless hours on perfecting papers and a thesis. Soon, we will both be ‘not that kind of doctor’. I dream that one day we will sit outside our house, somewhere in a sunny country, with a sign above the door saying ‘*Os dois doutores*’. With a glass of wine in our hands, we will reminisce about the relativity of life, and working hard, and perhaps we will hand our theses over to our niece and nephew, to inspire a new generation.

Abstract

The Domain Name System (DNS) plays a vital role in today's Internet. It translates human readable names (such as `www.example.com`) into machine readable information (such as `93.184.216.34`). The DNS was developed in the early 1980s, as the Internet started to expand. At that time, security was not a design concern. Consequently, it is no surprise that DNS turns out to be vulnerable. In the '90s, a type of attack called *cache poisoning* surfaced. This type of attack became more sophisticated over time, reaching a crescendo in 2008. In that year, a security researcher revealed a variant of the attack that could effectively take over an entire domain, potentially misdirecting thousands of Internet users to malicious sites, such as phishing sites.

The discovery of cache poisoning triggered the development of the DNS Security Extensions. DNSSEC addresses DNS's vulnerabilities by adding two key properties. First, *authenticity*, the guarantee that information in the DNS comes from a legitimate source. Second, *integrity*, the guarantee that DNS data has not been modified in transit. DNSSEC uses *digital signatures* to achieve these goals. DNSSEC improves both the security of the DNS, and the trust fabric of the Internet. Yet DNSSEC deployment is far from universal. Only an estimated 3% of all domains worldwide deploy DNSSEC. This number seems disappointing, raising the question if there are problems with DNSSEC.

Apart from the fact that the DNS itself needs to be secure, the data contained in the DNS is vital for the security and stability of the Internet. Take e-mail, for example. The DNS contains data used, e.g., to combat spam and e-mail forgery. As the DNS is such an integral part of Internet services, online criminals, by necessity, also have to register data in the DNS. E.g., to mount a phishing campaign, domains need to be set up in the DNS from which to send e-mail. The DNS also plays a role in the resilience of Internet services to attacks. For example, the DNS can be used for traffic redirection to services that protect against DDoS attacks. DNS data can also reveal single points of failure, such as the use of a single DNS provider for a domain. Real-world experience shows that such single points of failure break down under sustained attack.

Because of the vital role of data in the DNS in Internet security and stability, researchers and anti-abuse organisations have developed approaches to measure what is in the DNS. Existing approaches to large-scale measurement of the

DNS passively record DNS usage. This can unveil, e.g., malicious activity after it occurred. But what if a domain has not been used yet? Or what if a domain is not requested very frequently and thus rarely observed in passive traces? Actively measuring what is in the DNS will record data for domains, regardless of use or popularity. Additionally, active measurements can be timed to record DNS data at predictable intervals, rather than depending on passively observing DNS traffic as it happens. Such time series can assist in uncovering trends in Internet use. Yet no large-scale active DNS measurement systems exist.

Given the importance of DNS and DNSSEC for the security of the Internet, this thesis has two main contributions. First, this thesis studies if there are *technical problems with DNSSEC, that limit its deployment*, and if that is found to be the case, what these problems are and how they can be solved. Second, this thesis studies how we can *actively perform comprehensive measurements* of the global DNS in a *scalable* manner, to complement existing passive DNS measurement approaches.

Focusing on the first contribution, this thesis shows that two technical problems in DNSSEC form a barrier to deployment. The first is *fragmentation*. Due to the inclusion of digital signatures, DNSSEC messages are larger than DNS messages. Therefore, DNSSEC messages are frequently fragmented into multiple Internet packets during transmission. Unfortunately, fragmented messages are prone to being blocked by, e.g., firewalls. This thesis shows that up to 10% of Internet hosts are unable to receive fragmented DNSSEC messages. This leads to reachability problems for domains that deploy DNSSEC. The second problem is *amplification*. DNS can be abused for so-called amplification attacks. These can have devastating effects on the Internet. Amplification attacks were used for some of the largest denial-of-service attacks in recent history. The effectiveness of these attacks depends on message size. Thus, since DNSSEC makes DNS messages larger, it makes the amplification problem worse. This thesis finds that use of DNSSEC can increase amplification by $6\times$ to $12\times$, on average.

DNSSEC's larger message size is at the heart of both problems. Can we, therefore, solve these problems by somehow reducing DNSSEC message sizes? DNSSEC messages are large because of the inclusion of cryptographic keys and digital signatures. By default, DNSSEC uses a signature scheme called RSA. While RSA was the 'safe' choice during DNSSEC's development, it has one major drawback: its keys and signatures are large, and continue to grow in size as security requirements are tightened. Fortunately, newer signature algorithms are equally suited for use in DNSSEC. In particular, algorithms based on Elliptic Curve Cryptography (ECC), which have much smaller keys and signatures. This thesis proves that use of ECC can effectively and safely solve the problems of fragmentation and amplification. This was not self-evident, since a major drawback is that ECC signature validation is up to two orders of magnitude

slower than RSA. Use of ECC could thus potentially introduce performance problems in DNSSEC. In this thesis we analyse this potential problem and prove that ECC's slower signature validation will not impose an insurmountable burden on the global DNS infrastructure. This takes away the last hurdle to large-scale deployment of DNSSEC using ECC.

Focusing on the second contribution, this thesis introduces a large-scale long-term active measurement system for the DNS. The system currently performs daily measurements for 60% of the global domain namespace, including major domains such as `.com`, `.net` and `.org`. The thesis shows how we overcome the unique challenges that come with measuring the DNS at such scale. It proposes a measurement architecture able to measure even the largest top-level domain (`.com`) on a daily basis. Furthermore, and equally importantly, the thesis shows how we can measure at such a scale without adversely impacting the DNS by overloading it with queries. The measurements result in unique longitudinal datasets of the DNS, that can serve to improve the security of the Internet. This is illustrated with five case studies. The first two focus on e-mail security. The third demonstrates use of this data in a real-world security incident, a case of so-called 'CEO fraud'. The fourth case study is about the resilience of the Internet under large-scale DDoS attacks, in particular the large attack on DNS operator Dyn in 2016. The fifth and final case study measures adoption of the elliptic curve signature schemes discussed earlier, that can help solve the problems in DNSSEC.

Finally, as the measurement data that forms the basis for the results presented in this thesis has value for the research community, where possible we have released datasets as open data. Where data cannot be publicly disclosed, it is made accessible to fellow researchers under restrictions or in aggregate form. This provides a basis for future research that builds on the results of this thesis.

Samenvatting

Het *Domain Name System* (DNS) vervult een vitale functie voor het internet. DNS vertaalt leesbare namen (zoals `www.example.com`) in voor computers bruikbare informatie (zoals `93.184.216.34`). DNS werd ontwikkeld in de jaren tachtig, toen het internet begon te groeien. In die tijd was veiligheid geen belangrijk ontwerpcriterium, en het is dan ook niet verrassend dat DNS kwetsbaarheden kent. In de jaren negentig kwam voor het eerst een aanval op DNS genaamd “*cache poisoning*” aan het licht. Naarmate de tijd vorderde werd deze aanval steeds krachtiger, en bereikte een piek in 2008. In dat jaar ontdekte een beveiligingsexpert een variant van de aanval waarmee effectief een heel internetdomein overgenomen kan worden door aanvallers. Als gevolg daarvan werd het triviaal om duizenden internetgebruikers te misleiden en naar kwaadaardige websites te sturen, bijvoorbeeld zogenaamde “*phishing*” sites.

De ontdekking van *cache poisoning* was een directe aanleiding voor het ontwikkelen van *DNS Security Extensions*. DNSSEC lost de kwetsbaarheden in DNS op door twee kerneigenschappen te introduceren. Allereerst, authenticiteit, de garantie dat informatie in DNS uit een legitieme bron komt. Ten tweede integriteit, de garantie dat DNS data niet is veranderd tijdens het versturen over het netwerk. DNSSEC gebruikt hiervoor digitale handtekeningen. DNSSEC verbetert hiermee zowel de veiligheid van DNS als de veiligheid van het internet. De uitrol van DNSSEC is echter verre van universeel. Volgens schattingen heeft slechts 3% van alle domeinen wereldwijd DNSSEC uitgerold. Dat is teleurstellend en doet de vraag rijzen of er problemen zijn met DNSSEC.

Afgezien van het feit dat DNS zelf veilig moet zijn, speelt de data die in DNS wordt opgeslagen ook een belangrijke rol in de veiligheid en stabiliteit van het internet. Neem bijvoorbeeld e-mail; informatie in DNS kan worden gebruikt bij het bestrijden van spam en e-mailvervalsing. Omdat DNS zo belangrijk is voor het internet ontkomen criminelen er niet aan om ook gebruik te maken van DNS. Als criminelen bijvoorbeeld een phishing campagne willen uitvoeren moeten ze daarvoor domeinen registreren en deze configureren voor het versturen van e-mail. DNS speelt ook een belangrijke rol bij het versterken van de weerbaarheid van het internet tegen aanvallen. Zo wordt DNS bijvoorbeeld gebruikt om verkeer om te leiden in het geval van zogenaamde “*DDoS*” aanvallen. Daarnaast kan data uit het DNS ook worden gebruikt om kwetsbaarheden te analyseren,

zoals het gebruik van één enkele dienstverlener voor bepaalde functionaliteit. Ook op het internet gaat het principe van “één is géén” op. Als zo’n enkele dienstverlener wordt aangevallen, kan dat grote gevolgen hebben.

Omdat DNS zo’n belangrijke rol speelt voor de veiligheid en stabiliteit van het internet hebben onderzoekers en veiligheidsorganisaties manieren ontwikkeld om te meten welke data er in DNS zit. Bestaande aanpakken om grootschalig aan DNS te meten berusten op het passief observeren van DNS verkeer. Daarmee kunnen bijvoorbeeld kwaadaardige activiteiten worden blootgelegd op het moment dat ze plaatsvinden. Maar wat als een domein wel bestaat maar nog nooit gebruikt is? Of wat als een domein bijna nooit wordt opgevraagd, en daardoor nog nooit in verkeer is geobserveerd? Actief opvragen wat er in het DNS zit biedt dan uitkomst. Onafhankelijk of een domein ooit daadwerkelijk door gebruikers wordt opgevraagd kan met actieve metingen informatie worden verzameld over welke gegevens zich in DNS bevinden. Daarnaast kunnen actieve metingen gepland worden, zodat op vastgestelde momenten (bijvoorbeeld eens per dag) gegevens worden opgevraagd. Dit soort tijdgebonden gegevens kunnen trends onthullen op het internet. Ondanks de voordelen van het actief verzamelen van DNS gegevens bestaat hiervoor nog geen systematische aanpak.

Gegeven het belang van DNS en DNSSEC voor de veiligheid van het internet heeft dit proefschrift twee hoofdbijdragen. Allereerst wordt in dit proefschrift bestudeerd of er *technische problemen* zijn met DNSSEC die de *uitrol* ervan *hinderen*, en, als dat het geval blijkt te zijn, hoe deze problemen kunnen worden opgelost. Ten tweede wordt in dit proefschrift onderzocht hoe op een *schaalbare* manier *actieve metingen* van het *wereldwijde DNS* kunnen worden uitgevoerd, als aanvulling op bestaande passieve manieren om het DNS te meten.

Kijkend naar de eerste bijdrage toont dit proefschrift aan dat er twee technische problemen zijn, die de uitrol van DNSSEC hinderen. Het eerst probleem is *fragmentatie*. Omdat DNSSEC-berichten digitale handtekeningen bevatten zijn ze groter dan standaard DNS-berichten. Zoveel groter zelfs, dat DNSSEC-berichten regelmatig worden opgeknipt in fragmenten bij transport over het netwerk. Helaas worden gefragmenteerde berichten vaak geblokkeerd door firewalls. Dit proefschrift laat zien dat 10% van de systemen op het internet problemen heeft met het correct ontvangen van gefragmenteerde DNS-berichten. Dit leidt tot bereikbaarheidsproblemen voor domeinen die DNSSEC gebruiken. Het tweede probleem is *amplificatie*. DNS kan worden gebruikt voor zogenaamde amplificatieaanvallen. Dit type aanval kan een catastrofaal effect hebben op het internet. Een aantal grote DDoS aanvallen in de recente geschiedenis maakten gebruik van amplificatie. De effectiviteit van een amplificatieaanval hangt af van de berichtgrootte. Omdat DNSSEC-berichten groter zijn dan standaard DNS-berichten maakt dat het amplificatieprobleem groter. Dit proefschrift laat zien dat DNSSEC de amplificatie met een factor zes tot twaalf groter maakt.

De grotere berichten van DNSSEC zijn de belangrijkste oorzaak van beide problemen. De vraag is daarom: kunnen we DNSSEC-berichten kleiner maken en zo deze problemen oplossen? DNSSEC-berichten zijn zo groot omdat er digitale handtekeningen en cryptografische sleutels in zijn opgenomen. Standaard gebruikt DNSSEC een systeem voor digitale handtekeningen genaamd “RSA”. Hoewel RSA de veilige keuze was toen DNSSEC voor het eerst werd ontwikkeld, kent RSA één grote tekortkoming: de sleutels en handtekeningen zijn groot, en ze zullen in de toekomst, nu de veiligheidseisen steeds strenger worden, alleen nog maar groter worden. Gelukkig zijn er modernere systemen voor digitale handtekeningen, die ook geschikt zijn voor gebruik in DNSSEC. In het bijzonder algoritmen gebaseerd op *Elliptic Curve Cryptografie* (ECC). Deze algoritmen hebben veel kleinere sleutels en handtekeningen, met gelijke of zelfs betere cryptografische eigenschappen. Dit proefschrift toont aan dat het gebruik van ECC effectief en veilig zowel het fragmentatie- als het amplificatieprobleem in DNSSEC kan oplossen. Dat lag niet voor de hand, aangezien een belangrijke beperking van ECC is dat het controleren van ECC digitale handtekeningen tot twee ordegrottes trager kan zijn dan RSA. Het gebruik van ECC zou zodoende kunnen leiden tot verminderde prestaties van DNSSEC. Dit proefschrift analyseert daarom ook dit potentiële probleem en bewijst dat de langzamere controle van handtekeningen niet kan leiden tot grote problemen voor DNS en het internet. Daarmee wordt de weg vrijgemaakt voor een grootschalige uitrol van DNSSEC op basis van Elliptic Curve Cryptografie.

De tweede hoofdbijdrage van dit proefschrift is dat er een schaalbare groot-schalige meetomgeving voor actieve metingen aan het DNS wordt geïntroduceerd. Dit systeem verricht op dit moment dagelijkse metingen van 60% van het wereldwijde DNS, inclusief de allergrootste top-level domeinen zoals `.com`, `.net` en `.org`. Dit proefschrift laat zien hoe de uitdagingen die het meten op deze schaal met zich meebrengt worden aangepakt. Verder wordt aangetoond hoe zulke metingen op een verantwoorde wijze kunnen worden uitgevoerd zonder overlast te veroorzaken op het internet. Uit deze meetomgeving komen unieke langetermijnmetingen van DNS, die kunnen worden gebruikt om de veiligheid en stabiliteit van het internet te verbeteren. De bruikbaarheid van de verzamelde meetgegevens wordt aangetoond door middel van vijf concrete voorbeelden. De eerste twee voorbeelden gaan over de beveiliging van e-mailcommunicatie. Het derde voorbeeld gaat over een waargebeurd veiligheidsincident, een geval van zogenaamde “CEO fraude”. Het vierde voorbeeld laat zien hoe internetbedrijven reageren op een grootschalige aanval, in dit geval op het bedrijf “Dyn” in 2016. Het vijfde en laatste voorbeeld meet in hoeverre de elliptic curve algoritmen voor digitale handtekeningen, die eerder zijn besproken, daadwerkelijk worden ingezet in DNSSEC.

Tot slot: de meetgegevens die zijn gebruikt voor dit proefschrift en die worden verzameld door het grootschalige meetsysteem dat is geïntroduceerd in dit proefschrift kunnen zeer waardevol zijn voor andere onderzoekers. Daarom worden waar mogelijk alle gegevens beschikbaar gemaakt als *open data*. En als dat niet mogelijk is (bijvoorbeeld vanwege privacy), wordt alles in het werk gesteld om de gegevens onder strikte voorwaarden toegankelijk te maken, of wordt afgeleide informatie vrijgegeven. Hiermee wordt de basis gelegd voor toekomstig onderzoek dat voortbouwt op de resultaten uit dit proefschrift.

Contents

1	Introduction	1
1.1	The Domain Name System	1
1.2	Problems With DNS Security	3
1.3	DNS and Internet Security and Stability	4
1.4	Goals, Research Questions and Approach	5
1.5	Organisation and Key Contributions	7
2	Background on DNS and DNSSEC	15
2.1	Reading Guide	15
2.2	The Origins of the DNS	16
2.3	Core Concepts of the DNS	16
2.4	Cache Poisoning: a Weak Link in the DNS	30
2.5	The DNS Security Extensions	36
2.6	Building Internet Trust with DNSSEC	49
3	DNSSEC and IP Fragmentation	54
3.1	Introduction	54
3.2	Extent of the Problem	57
3.3	Solutions	60
3.4	Evaluation	64
3.5	Towards Operational Advice	66
3.6	Concluding Remarks	71
4	DNSSEC and Amplification Attacks	73
4.1	Introduction	73
4.2	Background on DNS(SEC) Amplification	74
4.3	Methodology	78
4.4	Datasets	82
4.5	Analysis	84
4.6	Countermeasures	92
4.7	Related Work	95
4.8	Concluding Remarks	96

5	The Case for Elliptic Curve Cryptography	98
5.1	Introduction	98
5.2	Problems with DNSSEC	100
5.3	Elliptic Curves in DNSSEC	104
5.4	The Effect of ECC on DNSSEC Problems	106
5.5	Potential Issues with ECC	110
5.6	Concluding Remarks	113
6	The Impact of ECC on DNSSEC Validation	114
6.1	Introduction	114
6.2	Related Work	115
6.3	Background	116
6.4	Approach	117
6.5	Model Validation	122
6.6	Predicted Impact	128
6.7	Discussion	135
6.8	Concluding Remarks	138
7	Large-Scale Active DNS Measurements	141
7.1	Introduction	142
7.2	Existing Approaches to DNS Measurements	143
7.3	Goals and Challenges	147
7.4	Design Choices	149
7.5	System Design and Implementation	152
7.6	Measurement Performance and Impact	155
7.7	Case Studies	161
7.8	Data Sharing	170
7.9	Concluding Remarks	171
8	Adoption of ECDSA in DNSSEC	172
8.1	Introduction	172
8.2	Methodology and Data	174
8.3	Results	175
8.4	Recommendations for Operators	183
8.5	Concluding Remarks	184
9	Conclusions	186
9.1	Main Conclusions	186
9.2	Revisiting the Research Questions	188
9.3	Directions for Future Research	195

Appendices	198
A DNS Server Implementations	199
B Open Dataset Management	200
B.1 Introduction	200
B.2 Open Access Publishing Practices	201
B.3 Accessibility of Non-Open Datasets	201
B.4 Per-Chapter Datasets	202
C Responsible Measurement Practices	204
C.1 Ethical Considerations	204
C.2 Passive Measurements	204
C.3 Active Measurements	206
Bibliography	208
About the Author	224

Introduction

This chapter introduces the background and motivation for this Ph.D. thesis. It discusses the research goals, formulates research questions and introduces the approach taken to answer these. The chapter concludes with an overview of the organisation of this thesis and lists the main contributions per chapter.

Over the past decades, the Internet has evolved from an academic experiment, spearheaded by universities, research institutes and research network operators, to an integral part of the fabric of society. It is near impossible to exhaustively list the impact the Internet has had on society. In our day-to-day business we rely on the Internet for communications through e-mail, we increasingly order products online, from electronics to clothing and groceries, and we even find our life partners online. Internet has become a focal point for the interaction between citizens and government, and some nation states have gone so far as to rely on the Internet for their democratic processes, such as referendums and elections [176]. In our own realm of academia, it is almost impossible to imagine the days before the advent of tools such as Google Scholar¹, which help us to rapidly find, access and assess publications in our field. And which scholar has not scoured Google Scholar profiles to peek at the *h*-indexes of fellow academics? Indeed, a self-referential search in said service finds over 126,000 scholarly articles for the search term ‘*Impact of Google Scholar on Academia*’.

1.1 The Domain Name System

If the introduction provided above is meant to illustrate one thing, it is the *trust* that we confer on the Internet on a day-to-day basis. And with this, we implicitly trust the key services that together make up the Internet. Many of these key services were designed in an era of the Internet when security and trust were not prime motivators for protocol and system design. Conversely, many core services have no inherent security properties, making them vulnerable to all sorts of attacks. The Domain Name System (DNS) is one of these

¹<https://scholar.google.com/>

Figure 1.1: Map of the online world, courtesy of Nominet

core services. The DNS plays a vital role in human interaction with the Internet. It translates human readable names into machine readable information. For example, it maps the human readable ‘`www.example.com`’ to the machine readable `93.184.216.34`. And DNS does more than that. It also plays a vital role in machine-to-machine interaction, for example by specifying hosts to which e-mail must be delivered. Figure 1.1 illustrates the importance of DNS for the online world². It shows a world map with country sizes scaled according to the number of domain names in country-code top-level domains (ccTLDs).

When the DNS was first introduced, in 1983, security was not a concern. This did not pose any insurmountable problems until security researcher Dan Kaminsky uncovered a major flaw in 2008, called *cache poisoning* [112]. In essence, the flaw Kaminsky uncovered makes it possible for an attacker to falsify information in the DNS practically at will. This has a direct impact on users. What if, e.g., they can no longer trust that the domain name for their bank is faithfully translated to the IP address of the bank’s server?

The uncovering of this flaw in the DNS was a catalyst that accelerated the development and deployment of the DNS Security Extensions (DNSSEC). What made this vulnerability so serious is that the DNS protocol misses two vital security properties in its design: *authenticity* and *integrity*. DNSSEC adds these properties to the protocol using digital signatures. These signatures guarantee the authenticity of the content in DNS messages, that is: they prove that data in these messages originates from the legitimate owner of a domain.

²Figure reproduced with permission from the copyright holder. Map of the online world created by Nominet in March 2016, data provided by Zooknic with numbers estimated on previously available data. See also <https://www.nominet.uk/mapping-the-online-world/>.

In addition to this, the signatures safeguard the integrity of DNS messages when they are transmitted. If the message is changed in transit, the signature will fail to validate, thus showing that the message has been tampered with.

The initial drive behind the deployment of DNSSEC was the unveiling of Kaminsky's cache poisoning attack in 2008. Once deployed, however, DNSSEC brings additional benefits. Because all data in a DNS zone that deploys DNSSEC is digitally signed, the DNS can now function as a trusted repository of information about hosts. This has many security applications. For instance in the realm of e-mail, as will be discussed in more detail in Chapter 2, DNSSEC can be used to significantly improve the security and privacy of e-mail transport between mail servers. These additional features are new drivers of DNSSEC deployment, and have led to governments adding DNSSEC, and standards that depend on it, to comply-or-explain lists for public tenders (e.g., [2, 3, 4]).

1.2 Problems With DNS Security

DNSSEC, however, is not without criticism or problems. DNSSEC suffers from two kinds of problems. Both kinds of problem originate from the fact that DNSSEC makes DNS messages much larger than 'classic' DNS messages. The first type of problem is of an operational nature. DNSSEC's larger messages make it susceptible to IP fragmentation, which can lead to problems reaching these domains, because fragmented messages are often blocked by firewalls. The second type of problem is a security issue. Due to its design, DNS can be abused to perform potent distributed denial-of-service attacks. These so-called *amplification attacks* rely on the fact that DNS is connectionless, making it vulnerable to IP address spoofing. Moreover, DNS responses are generally larger than DNS requests. Since DNSSEC generally makes DNS responses larger, deployment of DNSSEC may make it easier for attackers to perform this type of attack. So while the main goal of DNSSEC is to improve the security of the DNS, and thus the Internet, its deployment may actually increase existing threats to the security and stability of the Internet. These two problems form a barrier to deployment of DNSSEC. Large operators are unlikely to adopt DNSSEC if it means that their domain may suffer from reachability problems or may be abused to perform denial-of-service attacks. As Figure 1.2 illustrates, DNSSEC deployment in, for example, the largest three generic top-level domains is very low. The same is true if we look at the Alexa top 1 million popular websites. And while there are some exceptions, such as in the .nl country code top-level domain, estimates based on publicly available statistics³ put DNSSEC deployment in the global DNS namespace at around 3%.

³<https://www.internetsociety.org/deploy360/dnssec/statistics/>

Figure 1.2: DNSSEC deployment on January 1, 2017

1.3 DNS and Internet Security and Stability

It is not just the DNS itself that needs to be secure. Data in the DNS also plays an important role in the security and stability of most Internet services. For example, the Sender Policy Framework (SPF) [114] uses DNS records to combat e-mail forgery. And it is not just legitimate users that use the DNS. Because of the important role DNS plays in most Internet services, criminals cannot avoid leaving tracks of their activity in the DNS. Considering e-mail again, for example, criminal activities such as spam campaigns and phishing require an active presence in the DNS. Next, the DNS plays a role in protecting against, e.g., Distributed Denial-of-Service (DDoS) attacks. Many services that protect against DDoS attacks leverage the DNS to redirect traffic to scrubbing centres [109]. Finally, studying what is in the DNS can uncover weak spots in online services. Take, for example, the large-scale DDoS attack on Dyn, that took place in October 2016 [93]. Companies that relied solely on Dyn’s services were severely affected by this attack. Those that spread their risk, however, and also used other service providers next to Dyn, were much better off. Knowledge of what is in the DNS can uncover such single points of failure.

The fact that DNS plays such an important role in the security and stability of the Internet has led to the development of systems to study the state of the DNS. These systems rely on passive measurement of DNS traffic at many vantage points on the Internet. While there is abundant evidence that this type of DNS research yields relevant security results, there is one drawback to passive DNS measurements: data about domains is only available once a domain has actively been used, and this use has been observed at a vantage point at which data is collected. This means that domains that have been set up in the DNS but not yet actively been used, or domains that are not observed by passive DNS sensors, are missing from passive DNS datasets. An alternative would of course be to perform active DNS measurements. Yet to date, no active DNS measurement systems exist that are similar in scale to existing passive DNS deployments.

1.4 Goals, Research Questions and Approach

Based on problems with the security of the DNS itself (Section 1.2) we define the first research goal of this thesis as follows:

Goal 1: *to study technical problems in DNSSEC that limit its large-scale deployment, and to identify and validate solutions to these problems.*

Furthermore, in Section 1.3 we argued that the DNS plays a vital role in the security and stability of Internet services. And while large-scale passive approaches to measuring the DNS exist, we argued that these have limitations that could be overcome by having a large-scale active measurement approach. We therefore define the second research goal of this thesis as follows:

Goal 2: *to develop ways to perform large-scale long-term active measurements of the data contained in the DNS.*

In the following sections we break these two goals down into research questions and provide an overview of how we approach answering these questions.

1.4.1 Goal 1: Technical Problems in DNSSEC

Research Questions

In the research goal we stated that we want to focus on technical problems that hamper large-scale deployment of DNSSEC. Consequently, our first research question is about identifying such problems:

RQ 1: *which technical problems exist in DNSSEC, that are severe enough to limit large-scale deployment of the technology? And to what extent do these problems occur in practice?*

We address RQ 1 in Chapters 3 and 4 of this thesis.

Once we have identified the major technical problems and studied to what extent they occur, the next question is, of course, if we can find effective solutions to these problems. The second research question, therefore, is:

RQ 2: *if answering RQ 1 identified technical problems in DNSSEC, what are effective solutions to these problems?*

RQ 2 is addressed in Chapter 5.

Then, we want to ensure that the solutions we have identified do not introduce new problems in DNSSEC. This leads to the following research question:

RQ 3: *what is the impact of the solutions identified as responses to RQ 2 on the global DNS infrastructure?*

Chapter 6 addresses RQ 3.

Finally, we want to measure if the solutions that we have identified are adopted in practice. This leads us to the final research question for this goal:

***RQ 4:** to what extent are the solutions identified in RQ 2 adopted in practice?*

RQ 4 will be addressed in Chapter 8.

Approach

To address the research questions that will be answered in the first part of this thesis, we take a systematic measurement-based approach. The Internet and the DNS are extremely complex. It is therefore difficult to perform reliable simulations [37, 71]. For this reason, we take an empirical approach to the research in the first part of this thesis. We use large-scale passive and active measurements from different vantage points at major network operators to study problems in DNSSEC. We then use statistical analysis methods to characterise the results of these measurements and to identify key behaviours in the DNSSEC ecosystem. If necessary, we develop models of this behaviour, which we validate with independent measurement data. To enable reproducibility and future research, we release any data we collect as open access data. We discuss our approach to open access datasets in Appendix B.

1.4.2 Goal 2: Active DNS Measurements

Research Questions

The second goal of this thesis is to develop the means for large-scale long-term active DNS measurements. The first question, of course, is if it is possible at all to make such a measurement scale. We therefore define the first research question addressed in the second part of this thesis as:

***RQ 5:** is it technically possible to perform an active measurement of a large portion of the global DNS namespace at regular, daily intervals? Can such a measurement scale to encompass even the largest top-level domain (.com)?*

Equally importantly, if we are able to show that it is technically possible to perform such a measurement, is the question if such a measurement might impose an unacceptable burden on the global DNS infrastructure. The next research question therefore is:

***RQ 6:** what is the performance impact of a large-scale daily active measurement of significant portions of the global DNS namespace?*

Next, once we have shown that it is possible to collect this data and that it does not impose an unacceptable burden on the global DNS infrastructure, we have to be able to store and analyse this data. This leads to the following research question:

RQ 7: *how do we efficiently store and analyse large-scale DNS datasets that are collected over long periods of time?*

These three research questions – RQ 5, RQ 6 and RQ 7 – will be addressed in Chapter 7 of this thesis.

Finally, we want to demonstrate the applications of large-scale longitudinal actively collected DNS datasets. This leads us to the final research question:

RQ 8: *can we demonstrate uses of large-scale longitudinal active DNS datasets with real-world examples?*

This final research question, RQ 8, is addressed in four case studies included in Chapter 7 and in a larger, fifth case study in Chapter 8.

Approach

To address the research questions related to the second goal of this thesis, we take the following approach. First, we identify the properties and limitations of existing measurement approaches for large-scale measurements of the DNS. This includes the only existing large-scale approach, passive DNS (pDNS). We then outline the specific goals and challenges of building a large-scale active measurement system for the DNS. This takes on board lessons learned from the active and passive measurements performed to address the first goal of this thesis (addressing problems in DNSSEC). Based on the goals we set for the measurement system, we create a design and explain and justify our design decisions. Next, we use performance measurements of the system to gauge the impact on the global DNS infrastructure. Finally, we discuss our open approach to sharing the data collected by our large-scale active DNS measurement system with the academic community. Again, we also provide more information on our approach to open access data in Appendix B.

1.5 Organisation and Key Contributions

Figure 1.3 shows a schematic outline of the structure of this thesis. The figure shows how chapters relate to each other and suggests a reading order. For example, in order to understand Chapter 8, it is recommended that readers are familiar with Chapters 5 and 7. Below we provide a brief summary of each chapter, list its key contributions and provide references for the publications on which the chapter is based.

Figure 1.3: Schematic thesis outline

Chapter 2: Background on DNS and DNSSEC

In this chapter, we provide background information on the Domain Name System (DNS) and the DNS Security Extensions (DNSSEC). We start with a brief history of the DNS, followed by a discussion of the security vulnerabilities in the original DNS protocol that led to the development of DNSSEC. We discuss the developments that led to the current DNSSEC standard, and discuss the core principles of the protocol. Finally, the chapter provides an outlook on recent applications of DNSSEC. These build on top of the trust fabric that DNSSEC provides to improve the security of other Internet protocols, such as the SMTP protocol for transporting e-mail. This chapter is partly based on the following technical reports:

- R. van Rijswijk-Deij, R. van Rein, D. Yoshikawa and P. Brand. Technical Report: *Hardening the Internet - The Impact and Importance of DNSSEC*. Utrecht, The Netherlands: SURFnet, 2009 [197].
- R. van Rijswijk-Deij. Technical Report: *Deploying DNSSEC - Validation on Recursive Caching Name Servers*. Utrecht, The Netherlands: SURFnet, 2012 [184].

Part 1

The following four chapters together form the first part of the thesis, and deal with analysing and solving technical problems in DNSSEC.

Chapter 3: DNSSEC and IP Fragmentation

This chapter discusses the first of the two technical problems we identify in DNSSEC: reachability problems due to fragmentation. Because of the inclusion

of digital signatures, DNSSEC messages are typically larger than ‘classic’ DNS messages. In fact, messages are sometimes so much larger, that they no longer fit in a single packet and are fragmented at the IP layer. As one of the funding organisations of this thesis⁴ experienced, this can lead to reachability problems of DNSSEC-signed domains [181, 182]. In the chapter we study why, how and to what extent fragmentation can lead to reachability problems for DNSSEC-signed domains due to firewalls blocking IP fragments. The main contributions of this chapter are that we:

- Develop a methodology to detect DNS resolvers that experience problems with fragmented DNSSEC messages;
- Quantify the extent to which DNS resolvers on the Internet suffer from this problem;
- Suggest two ways to mitigate this problem on the side of the operator of a DNSSEC-signed domain;
- Test these two solutions in practice and show that it is possible to mitigate this problem effectively;
- Provide concrete guidance for operators based on real-world data to implement solutions to this problem.

The results of the work discussed in this chapter were presented at a number of conferences and meetings frequented by the DNS and network operator community⁵. Through these presentations, this work has influenced the revision of the EDNS0 specification [52] and led to improved fallback behaviour in common open source DNS resolver implementations⁶.

This chapter is based on the following peer-reviewed publication:

- G. van den Broek, R. van Rijswijk-Deij, A. Sperotto and A. Pras. *DNSSEC Meets Real World: Dealing with Unreachability Caused by Fragmentation*. IEEE Communications Magazine, Vol. 52 (April 2014), pp. 154-160 [180].

Chapter 4: DNSSEC and Amplification Attacks

This chapter studies the second major problem with DNSSEC: the potential for amplification attacks. Like all protocols that rely on UDP, DNS is susceptible

⁴SURFnet, the National Research and Education Network (NREN) in The Netherlands (<https://www.surf.nl/en/about-surf/subsidiaries/surfnet/>).

⁵Specifics are discussed in Chapter 3, Section 3.5.

⁶An overview of common implementations is provided in Appendix A.

to IP spoofing and can be abused for so-called amplification attacks. In such an attack, a DNS query – which is generally small in size – is sent to a server. The query has a spoofed source address (that of the victim). The DNS response – which is generally larger – is then sent to the spoofed address. If performed at scale, such a denial-of-service attack can result in a large traffic volume towards victims. Because DNSSEC makes DNS responses larger, its deployment increases the potential for DNS amplification attacks. This is in fact one of the most frequent criticisms of DNSSEC (e.g., [31]). Yet while DNSSEC deployment has seen steady progress since 2008, there was no ground truth as to what extent DNSSEC increased the potential for DNS amplification attacks. In this chapter, we establish this ground truth, by studying DNS response sizes for 2.5 million DNSSEC-signed domains⁷ and comparing the amplification factor with an equal-size random sample of domains that do not use DNSSEC. The contributions of this chapter are that we:

- Perform the first large-scale study of the effect of DNSSEC on DNS amplification attack potential;
- Show that the use of DNSSEC results in an average increase in amplification potential between $6\times$ and $12\times$ with the highest measured amplification factor around $179\times$;
- Show that this increase in amplification potential is mainly problematic for so-called **ANY** queries, and to a lesser extent for the DNSSEC-specific **DNSKEY** query type, but that the increase in amplification is not so dramatic for the more common DNS query types (such as **A**, **AAAA**, ...);
- Discuss mitigation strategies for this problem;
- Release the data collected for the study into the public domain to facilitate future research and reproducibility.

This chapter is based on the following peer-reviewed publication:

- R. van Rijswijk-Deij, A. Sperotto and A. Pras. *DNSSEC and Its Potential for DDoS Attacks - a Comprehensive Measurement Study*. Proceedings of ACM IMC 2014. Vancouver, BC, Canada: ACM Press [193].

The publication on which this chapter is based has received recognition in the form of the following awards:

- The 2014 ACM SIGCOMM IMC Community Contribution Award.
- The 2015 IRTF Applied Networking Research Prize.

⁷At the time of the study in 2014 this was around 70% of signed domains.

Chapter 5: The Case for Elliptic Curve Cryptography

The only mandatory digital signature algorithm in DNSSEC is based on the RSA cryptosystem. Up until 2015, DNSSEC deployments almost exclusively used the RSA cryptosystem for digital signatures. In Chapter 4 one of the suggested mitigation methods to dampen the amplification potential in DNSSEC is to switch to alternative signature algorithms based on Elliptic Curve Cryptography (ECC). This is because ECC signatures and keys can achieve a higher level of cryptographic security than RSA with much smaller keys and signatures. This makes ECC an attractive option for use in DNSSEC, not only to reduce the amplification potential, but also to reduce the probability of fragmentation (Chapter 3). In this chapter, we revisit the studies performed in Chapters 3 and 4 and study how the problems discussed in these chapters can be alleviated by switching DNSSEC to signature algorithms based on elliptic curve cryptography. The contributions of this chapter are that we:

- Quantify how a switch to ECC practically eliminates fragmentation problems in DNSSEC;
- Quantify the reduction in amplification potential if DNSSEC switches to ECC-based signature algorithms;
- Discuss potential barriers to deployment of ECC-based signature algorithms in DNSSEC.

This chapter is based on the following peer-reviewed publication:

- R. van Rijswijk-Deij, A. Sperotto and A. Pras. *Making the Case for Elliptic Curves in DNSSEC*. ACM SIGCOMM Computer Communication Review (CCR), Volume 45, Issue 5 (October 2015) [195].

Chapter 6: The Impact of ECC on DNSSEC Validation

In Chapter 5, we identified a major hurdle for the deployment of elliptic curve-based signature algorithms in DNSSEC. Signature validation of ECC-based signatures can be up to almost two orders of magnitude slower than the validation of the RSA signatures in common use in DNSSEC. Thus, a large-scale deployment of ECC-based signature algorithms in DNSSEC runs the risk of imposing a significant additional burden on DNS resolvers that validate DNSSEC signatures. Therefore, before we can recommend a switch to ECC, to benefit from its smaller keys and signatures, it is imperative that we ascertain that deploying ECC does not impose an undue burden on validating resolvers. In this chapter, we study what impact deployment of ECC would have on validating DNS resolvers. The contributions of this chapter are that we:

- Introduce a novel model for the behaviour of validating DNS resolvers;
- Validate this model using real-world data for two popular open source DNS implementations;
- Use the model to predict the impact of ECC deployment on validating DNS resolvers;
- Show that ECC signature validations can be handled on a single modern CPU core, even in a worst case scenario, in which DNSSEC deployment becomes universal.

This chapter is based on the following peer-reviewed publication:

- R. van Rijswijk-Deij, K. Hageman, A. Sperotto and A. Pras. *The Performance Impact of Elliptic Curve Cryptography on DNSSEC Validation*. IEEE/ACM Transactions on Networking, Volume 25, Issue 2 (April 2017) [188].

The publication on which this chapter is based has received recognition in the form of the following award:

- The 2017 IRTF Applied Networking Research Prize.

Part 2

The following two chapters form the second part of the thesis and discuss how we have designed and implemented a large-scale active measurement platform for the DNS. We illustrate the applications of the data collected by this platform using four smaller and one larger case study.

Chapter 7: Large-Scale Active DNS Measurements

This chapter introduces the goals, challenges and design of a large-scale long-term active measurement system for the DNS. We explain in detail how proactive longitudinal measurements of the DNS can aid in understanding and improving the security of the Internet and the adoption of new protocols over time. The main contributions of this chapter are that we:

- Position our active measurement approach as complementary to the only existing large-scale approach to DNS measurements, passive DNS (pDNS);
- Discuss the challenges of tracking the state of large portions of the DNS over time (including the largest top-level domain, .com);

- Design and implement a novel large-scale active measurement system that meets these challenges;
- Show that our measurement, while large, does not overload the global DNS infrastructure;
- Validate and illustrate the use of the measurement system using four case studies, two on cloud e-mail services and two on real-world security events.

This chapter is based on the following peer-reviewed publications:

- R. van Rijswijk-Deij, M. Jonker, A. Sperotto and A. Pras. *The Internet of Names: A DNS Big Dataset – Actively Measuring 50% of the Entire DNS Name Space, Every Day*. Proceedings of ACM SIGCOMM 2015. London, UK: ACM Press [190].
- R. van Rijswijk-Deij, M. Jonker, A. Sperotto and A. Pras. *A High-Performance, Scalable Infrastructure for Large-Scale Active DNS Measurements*. IEEE Journal on Selected Areas in Communications (JSAC), Volume 34, Issue 7 (May 2016) [191].

Chapter 8: Adoption of ECDSA in DNSSEC

In this chapter, we study the adoption of a particular elliptic curve algorithm in DNSSEC, the Elliptic Curve Digital Signature Algorithm (ECDSA). The use of ECDSA in DNSSEC was standardised by the IETF in 2012 [94], yet when the study that forms the basis for Chapter 5 was performed in 2015, adoption of this algorithm by DNS operators was virtually non-existent. The goal of the study discussed in this chapter is to see if this has changed. Especially after CloudFlare – a large operator of content delivery and DDoS protection services – announced in 2015 that they were going to start supporting DNSSEC and would be using ECDSA as signing algorithm, and after the studies in Chapter 5 [195] and Chapter 6 [188] were published and presented in a number of venues, including an ICANN meeting and a NANOG meeting. We perform this study using data collected by the system discussed in Chapter 7 over a period of almost two years. The contributions of this chapter are that we:

- Illustrate how the large-scale active DNS datasets collected by the system discussed in Chapter 7 can be used for the longitudinal study of the adoption of new protocols;
- Show a clear trend in adoption of ECDSA for DNSSEC, first for domains operated by CloudFlare, but later also by other DNS operators;

- Find evidence of hurdles to adoption of ECDSA in the dataset;
- Provide recommendations for domain name owners and operators on how to proceed with ECDSA and DNSSEC adoption.

This chapter is based on the following peer-reviewed publication:

- R. van Rijswijk-Deij, M. Jonker and A. Sperotto. *On the Adoption of the Elliptic Curve Digital Signature Algorithm (ECDSA) in DNSSEC*. Proceedings of the 12th International Conference on Network and Service Management (CNSM 2016). Montréal, Canada: IFIP [189].

Chapter 9: Conclusions

This final chapter of the thesis draws overall conclusions based on the research discussed in the other chapters. The chapter also discusses potential future research directions.

Background on DNS and DNSSEC

This chapter provides background information on the Domain Name System. It discusses the origins of the DNS, over 30 years ago, and introduces its core concepts. It also provides a detailed explanation of the so-called cache poisoning attack, the major vulnerability in the DNS. This attack triggered development of the DNS Security Extensions (DNSSEC), the history and working of which is discussed next. The chapter ends by discussing how DNSSEC, once deployed, can improve the security of other Internet protocols.

2.1 Reading Guide

This chapter provides detailed background information about the DNS and DNSSEC. Readers who wish to focus on the background information required to understand the other chapters of this thesis are recommended to read Section 2.3, which introduces the core concepts of the DNS and Section 2.5, which explains how DNSSEC works. The chapter also provides a broader context for the work in this thesis. Section 2.2 discusses the origins of the DNS. Section 2.4 explains in detail how cache poisoning attacks work. Finally, Section 2.6 provides information about recent developments where DNSSEC is used as basis to enhance trust in other Internet protocols.

2.2 The Origins of the DNS

Standards for naming hosts on the network are almost as old as the Internet (or rather its precursor, the ARPANET) itself. Initially, every site connected to the early network maintained a copy of a file called `HOSTS.TXT` that provided a mapping from names to network addresses [171]. The early pioneers realised that keeping separate copies of this file synchronised for a growing network was bad practice. In late 1973, when there were fewer than fifty hosts on the ARPANET [171], the first document in the RFC series relating to naming of hosts on the network was written [55]. This, and subsequent RFC documents specified how and where the list of hostname-to-address mappings was to be maintained. In the early 1980s it became ever clearer that maintaining such a central database scaled badly and was error prone. Discussion, in early 1982, on problems in relaying e-mail across the network led to the current concept of structured hierarchical domain names first appearing [157]. This was followed in late 1983 by the first set of specifications for the Domain Name System [138, 139] and transition plans to migrate from a centrally managed database of names to the DNS [158, 159, 160]. In 1987 the DNS specifications were updated, resulting in the basic protocol that is still in use today [140, 141].

2.3 Core Concepts of the DNS

This section provides a detailed discussion of the core concepts of the Domain Name System. Throughout this section and the remainder of this chapter we will use the DNS terminology as specified by the IETF DNS operations working group [97]. Furthermore, definitions of aspects of the DNS protocol originate from the original DNS specifications [140, 141], unless specified otherwise.

2.3.1 The Structure of the DNS

Domain Name Concepts and Terms

The central concept in the DNS is the *domain name*. A domain name is represented as a structured ASCII character string. In this representation domain names are built up from labels separated by dots. Figure 2.1 shows examples of domain names with associated concepts. The left-hand side of the figure introduces the following terms relating to domain names:

- **Label** – Domain names are made up of labels, where each label is limited to a maximum of 63 characters in length. Labels may contain the letters A–Z, a–z, the digits 0–9 and the hyphen (-). Labels are *case insensitive*,

Figure 2.1: Domain name concepts

that is: ‘`www`’ and ‘`WWW`’ are equivalent. In DNS messages labels are encoded using a single unsigned byte value that indicates the length of the label, followed by 8-bit ASCII characters for the label text.

- **Root Label** – The root label is the terminator of a domain name and is represented as an empty label. In a textual hostname, the presence of the root label is sometimes indicated by a single dot at the end of the name, but this dot is often omitted. In DNS messages the root label is represented as a single byte value set to `0x00`. This label indicates the top of the DNS hierarchy (which will be discussed below). Parsers of DNS messages must stop processing a domain name when they encounter the root label.
- **Hostname** – This term sometimes refers to the left-most label of a domain name (in which case it typically refers to the local name of a machine). In other cases, the term is used to refer to the whole domain name. Because of this ambiguity, we try to avoid use of this term in this thesis.

The right-hand side of the figure shows the following terms:

- **Fully Qualified Domain Name** – Sometimes abbreviated to FQDN, this term means the whole domain name, i.e., all labels that make up the name, including the root label. This term is often used interchangeably with the shorter ‘*domain name*’. In this thesis, when we use the term ‘domain name’ we generally refer to an FQDN.
- **{cc | g}TLD** – The acronym TLD is short for *Top-Level Domain*. TLDs are the domain names directly below the root in the DNS hierarchy (as will be discussed below). The terms *ccTLD* and *gTLD* are also frequently used. In the former, ‘cc’ refers to *Country Code*, as these TLDs are specific to geographic countries. In the latter, ‘g’ refers to *Generic*. Generic TLDs are, as the term implies, not specific to a country.

Figure 2.2: Example DNS hierarchy

- **Public Suffix** – As we will explain below, some TLDs divide the namespace under their control into separate branches. The combination of the branch label and the TLD label is often referred to as a public suffix. There is even a publicly available list of such suffixes¹.

The DNS Hierarchy

The DNS has a hierarchical organisation, shaped like a tree. Figure 2.2 illustrates this showing a part of the actual DNS tree. At the top of the tree is the root of the DNS. The root of the DNS is managed by the Internet Corporation for Assigned Names and Numbers (ICANN). They delegate responsibility for the maintenance of top-level domains, shown directly below the root, to so-called *registries*. Some registries divide the namespace under their control into separate branches (public suffixes), as is for instance shown in the figure for the .uk ccTLD, with, e.g., a .co.uk for commercial domains and, e.g., a .ac.uk for academic institutions.

The next level down in the tree are second-level domains. These are the domain names that generally belong to people or organisations. Below second-level domains we find third and further level domains. There is no formal convention for how this part of the namespace is organised, although there are common practices. As Figure 2.2 shows, for example, it is highly likely that there is a ‘www’ label to indicate a World Wide Web service.

¹<https://publicsuffix.org/>

The Domain Name Industry

Initially, the number of top-level domains in the DNS was very limited. In 1985, the first ccTLD, `.us`, was added to the DNS, soon followed by further ccTLDs. The names of ccTLDs are based on ISO-specified country codes² [105]. Initially, domain name registrations were handled centrally through IANA. When Internet growth really took off, in the 1990s, this no longer scaled. This led to the introduction of a tiered model, where TLDs have *registries*, that allow separate companies, called *registrars*, to sell domain names to interested parties. The owner or holder of a domain name is referred to as a *registrant*. For gTLDs, this model is mandatory; there is a common set of requirements for registrars of gTLDs, set out by ICANN, against which registrars have to be accredited [104]. For ccTLDs the registration policy is determined by the registry operator, and differs from ccTLD to ccTLD. We note that the registration and administration of domain names is often referred to as taking place through the *Registry-Registrar-Registrant* (or RRR for short) channel. This channel is separate from the DNS and uses its own protocols (e.g., the EPP protocol [98] for communication between registrars and registries).

With the advent of the World Wide Web in the 1990s, the demand for domain names exploded. Domain name registration became a lucrative business, and people started trading domain names, selling popular domain names for high prices. Inevitably, there were soon disputes leading to litigation about domain names. These disputes led to dispute resolution policies coming into being [133], and certain jurisdictions, such as the United States, adopting laws to regulate ‘cybersquatting’ (the illegitimate exploitation of domain names) [177]. The demand for domain names also led to new forms of abuse. There have been, for example, attempts to establish new TLDs and sell domain names in these. One approach to this, that was for instance attempted by the ‘New.net’ company, was to sell domain names that would only be visible if users installed web browser extensions (because these TLDs did not actually exist in the DNS) [40]. Another example of this practice was the AlterNIC company³. One of the founders of this company, Eugene Kashpureff, performed the first well-documented act of DNS cache poisoning, which we will discuss in more detail in Section 2.4. These practices of establishing alternate DNS hierarchies eventually led to the Internet Architecture Board (IAB) issuing a clarification on the need for a single DNS root for the Internet [107]. In the period between 2000 and 2012, ICANN introduced a limited number of additional gTLDs. In 2011, ICANN announced a new policy that effectively opened up applications for a potentially unlimited

²With a few exceptions: `.ac`, `.eu`, `.su` and `.uk`.

³<https://en.wikipedia.org/wiki/AlterNIC>

What?	Example value	Example DNS name
IPv4 address	93.184.216.34	34.216.184.93.in-addr.arpa.
IPv6 address	2001:620:0:9::1103	3.0.1.1.[...]0.2.6.0.1.0.0.2.ip6.arpa.*
Telephone number	+31-88-7873000	0.0.0.3.7.8.7.8.8.1.3.e164.arpa.

*Truncated to save space.

Example 2.1: Numerical DNS name examples

number of new gTLDs. Under this policy, well over 1000 new gTLDs have been added to the DNS since 2013.

Today, domain names are a multi-billion US dollar industry. The largest domain name registrar in the world alone, GoDaddy, reported an annual revenue in excess of USD\$ 1.8B in 2016⁴. There are very few verifiable sources of the total turnover in the industry, but to give an indication, business intelligence firms quote revenues of USD\$ 2B in the US alone in 2016⁵. One of the reasons given by ICANN for allowing additional gTLDs was to offer registrants new options to obtain the domain name of their choice. This has, however, met with criticism, with many brand owners claiming that they are effectively forced to perform defensive registrations of their brands. Recent research looked at developments in the .xxx gTLD that was added by ICANN in 2011 [84] and at the new gTLDs added as of 2013 [83]. Both studies find large numbers of defensive registrations and speculative registrations (where the researchers infer that the registrant has registered the name with the intention of selling it at a profit). Especially noteworthy is that in both studies defensive and speculative registrations far outnumber what the authors call primary registrations, that is: registrations of domains that host actual web content.

Reverse DNS and Other Numerical Names

Generally, the DNS is used to translate human readable names into machine readable information. The reverse, however, is also possible. By taking an IPv4 or IPv6 address and reversing its numerical representation, a domain name can be constructed. DNS queries for this name can then be used to, for instance, find the name associated with an IP address (see also Section 2.3.2 below). Example 2.1 shows example mappings between IPv4 and IPv6 addresses and their corresponding reverse DNS names. As the example shows, for IPv4 addresses the name is simply a reverse of the dot notation of the address. For IPv6, the reverse name consists of all 32 nibbles of the address; as the example shows, this can be quite cumbersome.

⁴Source: GoDaddy Annual Report 2016

⁵<https://www.ibisworld.com/industry/web-domain-name-sales.html>

Figure 2.3: DNS message format, header layout and resource record format

Another application of numerical names in DNS is the incorporation of telephone numbers in the DNS (for use in, e.g., Voice-over-IP (VoIP) or other communication applications). This can be done according to the ENUM specification [36]. ENUM specifies how international telephone numbers that follow the ITU’s E.164 standard [106] can be converted to DNS names. Example 2.1 also includes an example of a phone number in DNS ENUM notation.

2.3.2 Base DNS Protocol

Message Format

The DNS uses the same basic message format for all messages, with certain fields filled, depending on the message type. Figure 2.3 shows the DNS message format. The middle part of Figure 2.3 shows that a DNS messages consists of a header, followed by four sections. The format of the header is shown in Table 2.1. Each of the four sections is filled with *resource records*. The general format of resource records is discussed in Table 2.2. In a DNS query, only the question and sometimes the additional section (see Section 2.5.2) contain information. In a DNS response, all four sections may contain information. The content of each section depends on many factors, including the response status of a DNS request (the *RCODE*). In general each of the four sections has the following semantics (according to the original DNS specification [140]):

- **Question** – Contains the question in a DNS query (generally the name and type queried for).
- **Answer** – Contains the resource records that form the response to the question.

Field	Description
Query ID	Identifies the query and helps match queries and responses.
QR	This flag indicates whether the message is a query (0) or a response (1).
Opcode	This field identifies the DNS operation. The most common value is 0 for a query/response operation (other values are assigned in [66]).
AA	This flag indicates whether the DNS response is an Authoritative Answer.
TC	This flag indicates whether the message was truncated because it exceeded the maximum message size.
RD	This flag indicates whether Recursion is Desired (explained in Section 2.3.3).
RA	This flag indicates whether Recursion is Available.
Z	Set to zero and reserved for future use.
AD	This flag indicates whether the response contains Authenticated Data (see Section 2.5.4).
CD	If set, this flag means that DNSSEC Checking should be Disabled (see Section 2.5.4).
RCODE	The response status of the DNS request. Important values are NOERROR (0), SERVFAIL (2), NXDOMAIN (3) and REFUSED (5).
Question count	Indicates how many questions are in the question section. Currently, this field is always set to 1 in DNS queries and responses.
Answer count	Number of records in the Answer, Authority and Additional section.
Authority count	
Additional count	

Table 2.1: DNS header fields

- **Authority** – Contains resource records pointing to authorities (name servers) for the queried name.
- **Additional** – Contains additional resource records pertaining to aspects of a DNS message, for example resource records with additional information on the authorities listed in the authority section.

In DNS responses, the answer, authority and additional section are all optional. Typically, though, in a successful response to a query, the answer section will contain one or more resource records that answer the query. In a successful response, the authority and additional section are usually optional, that is: they may be left empty, for instance to save space in a DNS message.

Query/Response Protocol

DNS messages are normally transported using UDP, and the original DNS specification lists a maximum payload size for DNS messages of 512 bytes. The use of UDP means that most DNS exchanges are asynchronous and connectionless. In some cases, messages are exchanged over TCP. A typical DNS message exchange, in which a DNS client sends a query to a DNS server, and the server returns a response to the client, looks like this:

1. *Client sends query* – The client composes a query by filling the question section of a DNS message. In this section, the client indicates the name,

Field	Description
Name	The domain name this resource record pertains to. Note that names may be compressed, to save space in datagrams. DNS compression works by replacing a label in a DNS name by a pointer to another DNS name in the same datagram. Compression is explained in more detail in [141].
Query type	The query type is an integer that indicates the specific kind of resource record. Common record types are discussed further on in the section.
Query class	An integer that indicates the query class. Historically, the DNS distinguished multiple classes of networks. These have, however, become obsolete over the years, and in almost all cases the query class is set to 1 to indicate the Internet (class IN).
Time-To-Live	The TTL field is an integer that provides an indication how long (in seconds) a resource record may be cached. The use of this field is discussed in more detail in Section 2.3.3.
RDATA Length	This field indicates the total length of the resource record-specific data that follows.
RDATA	Variable length field with data that is specific to the resource record type.

Table 2.2: General DNS resource record layout

query type and query class in which they are interested. The client sets the **QR** flag to 0 to indicate that the message is a query. The client optionally sets the **RD** flag to 1 to indicate that the client would like the receiving server to perform DNS recursion on its behalf (see Section 2.3.3).

2. *Server sends response* – The server responds to the request in the question section of the query. It copies the question section into the response, and fills the other sections of the response depending on whether or not it is able to answer the request. The server then sends the response back to the client.
3. *(optional) Fallback to TCP* – If the server cannot make a full response fit in a single DNS message, it will set the **TC** flag in the response over UDP. This is an indication for the client to retry the query over TCP to get the full response.

Typically, DNS clients will initiate a request to a DNS server over UDP, but there is no hard requirement to do so. They may also directly initiate a request over TCP. In addition to this, clients may keep the TCP connection to a server open and issue multiple requests in a single session. Generally, UDP is still the preferred way to transport DNS messages. The main reason for this is performance; setting up a TCP connection requires more network round trips, and keeping TCP connections open for long periods of time unnecessarily consumes resources on both the client and the server. There are changes to the DNS on the way, though. A workgroup focusing on DNS privacy has recently

Type	Description
A	Maps a domain name to an IPv4 address.
AAAA	Maps a domain name to an IPv6 address.
CNAME	Specifies an alias for a name. If a CNAME exists for a name, incoming queries for that name are translated into queries for the name that the CNAME points to. For example: if there exists a CNAME that maps the name <code>foo</code> to the name <code>bar</code> , then a query for the A record for <code>foo</code> will effectively be treated as an A query for <code>bar</code> . CNAMEs may be chained, that is: a CNAME may point to another CNAME.
MX	Specifies Mail eXchange records for a name. These are the servers that handle incoming e-mail for a domain. Mail servers attempt to deliver e-mail sent to <code>user@example.com</code> to the servers specified in the MX records for <code>example.com</code> .
NS	Specifies the names of authoritative name servers for a domain name.
PTR	Pointer record from a domain name to another domain name. This record type is most commonly used for reverse DNS, to map e.g. IP addresses to domain names (see Section 2.3.1).
SOA	Start Of Authority record. The SOA record specifies metadata about a DNS zone, such as the serial number of the DNS zone. DNS zones are explained in more detail further along in the section.
TXT	Text record. TXT records may contain arbitrary text strings with a maximum length of 255 characters each. TXT records are, for example, used for the so-called Sender Policy Framework (SPF) [114], which is designed to combat e-mail forgery.

Table 2.3: Common DNS resource record types

standardised DNS over TLS [100]. In the standard, the authors suggest using another recent development, TCP Fast Open [44], to reduce the overhead of using TCP. A detailed discussion of DNS privacy and DNS over TLS is outside the scope of this thesis.

DNS Resource Record Types

Table 2.3 introduces the most commonly used DNS record types in alphabetical order. Only basic record types that are part of the original DNS specification [141] are listed, other record types, such as those used for DNSSEC, will be introduced later on when extensions to the protocol are discussed.

DNS Zones

Data for domains in the DNS is organised into so-called *zones*. DNS zones contain resource records under a certain name in the DNS hierarchy. Zones are represented using ASCII text in so-called *zone files*, the format of which is specified in the original DNS specification [141]⁶.

⁶The original DNS specification [141] refers to zone files as ‘master files’.

1	\$ORIGIN example.com.					
	...					
	domain name	TTL	class	type	value	
2	@	86400	IN	A	93.184.216.34	← RRset #1
3	@	86400	IN	AAAA	2606:2800::...:1946	← RRset #2
4	@	86400	IN	NS	a.iana-servers.net.	} ← RRset #3
5	@	86400	IN	NS	b.iana-servers.net.	
6	www	86400	IN	CNAME	example.com.	← Alias
7	sub	3600	IN	NS	ns1.example.org.	} ← Delegation
8	sub	3600	IN	NS	ns2.example.org.	
9	lorem.ipsum	86400	IN	A	127.0.0.1	← Results in empty non-terminal
10	*.dolor	300	IN	TXT	"Sed ut perspiciatis unde omnis..."	← Wildcard

Example 2.2: Example DNS zone snippet for `example.com`

Example 2.2 shows part of a DNS zone file for `example.com`⁷. At the top, on line 1, the `$ORIGIN` statement tells whatever software parses the zone file that all domain names in the file are relative to `example.com`. In other words: the label ‘`www`’ on line 6 should be interpreted as ‘`www.example.com`’.

Lines 2-5 show resource records for the so-called apex record of the zone. The apex record is signified using the `@`-sign in the zone file, and points to the origin of the zone (`example.com`). Lines 2-5 also show the concept of resource record sets, or *RRsets*. An *RRset* consists of all resource records of a certain class and type for a certain name (e.g., lines 4-5 are an *RRset* consisting of all *NS* records for `example.com`).

Line 6 shows how a *CNAME* can be used to create an alias, in this case from `www.example.com` to the apex records of `example.com`.

Lines 7-8 show a *delegation* of a subdomain called `sub.example.com` to be managed by the two authoritative name servers specified (see also Section 2.3.3 below). Any queries for names in that subdomain should be directed to these name servers. Also note that on lines 4-5 there are *NS* records for `example.com` itself. This is not a delegation, these are the authoritative information on what the name servers for `example.com` are. In general, the delegation in a parent zone and the *NS* records in a child zone should be the same, but in practice these frequently diverge [152]. This is mostly due to human error; administration of delegations is usually a very different process from editing of a DNS zone file. Especially delegations in TLDs are generally updated through the RRR channel, which, as we mentioned in Section 2.3.1, is completely separate from the DNS.

Line 9 shows a resource record consisting of two labels. This record illustrates that a DNS zone can contain multiple label levels under a delegation point. In this case, the zone thus not only contains records in `example.com` but also in

⁷Line numbers are included for convenience, and are not present in an actual zone file.

Figure 2.4: High-level architecture of the DNS

`ipsum.example.com`. Because the subdomain `ipsum` is not delegated to other name servers, and because there are no records in the `example` zone for `ipsum` itself, this has another effect: `ipsum.example.com` becomes a so-called *empty non-terminal*. This has consequences for DNSSEC denial-of-existence proofs (Section 2.5.3).

Line 10, finally, illustrates that the DNS also supports *wildcards*. A wildcard is always the leftmost label in a domain name, and matches any label or labels provided (i.e., it also matches `<label1>.<label2>.dolor`). DNS servers will only return a wildcard record if the queried record does not explicitly exist. Thus, if, e.g., a record `'nullam.dolor'` is added, and a query is received for this name, that record will be returned rather than the wildcard.

2.3.3 Domain Name Servers

The DNS generally has two server roles. The first role is that of the authoritative name server, the second role performs DNS resolution. In principle, these two roles can be combined on a single name server. From a security perspective, however, this is undesirable. Combining both roles on a single server can, for instance, more readily lead to servers being misconfigured as so-called ‘open resolvers’ (discussed in more detail in Chapter 4).

Figure 2.4 shows these two roles from an architectural perspective. Authoritative name servers are shown on the righthand side of the figure (III). The actors involved in DNS resolution are shown on the lefthand side of the figure (I+II). The two sections below explain these two roles in more detail.

Authoritative Name Servers

Authoritative name servers are, as their name implies, the authority for a domain. An authoritative name server can serve many domains. As mentioned in the previous section, authority for a domain is delegated to an authoritative name server in the parent zone of that domain, using **NS** records. As we will see in the next section, there is a delegation chain, from the root, which delegates to a TLD, which delegates to a second-level domain, etc.

If an authoritative name server responds to a query for a name for which it is authoritative, it indicates this in the response by setting the **AA** flag (Authoritative Answer, see Section 2.3.2). If a server is configured as only authoritative, and it receives a query for a name for which it is not authoritative and does not know of a delegation for the queried name to another authoritative name server, it will refuse the query by setting the **RCODE** in the response to **REFUSED**.

DNS Resolution

DNS resolution is the process of mapping a domain name to a value contained in the DNS. This process starts on the client (shown on the lefthand side of Figure 2.4). Say, for example, that a user wants to visit the URL **https://www.example.com/**. They type this URL into their web browser and press ‘Go’. The first thing the browser will do is to attempt to resolve the address for **www.example.com**. To do this, it most likely calls a function of the operating system. Most OSes have a built-in *stub resolver*. This is a very limited DNS client that can send queries to DNS servers and can process responses returned by these. More importantly, however, is that a stub resolver cannot perform a process called *recursion*. That is: it cannot traverse the authoritative name servers in the DNS hierarchy to find a response to a query. Instead, a stub resolver typically sends a query to a *recursive caching name server* (shown in the middle of Figure 2.4). Recursive caching name servers are often referred to as a ‘DNS resolver’, or simply a ‘resolver’. Whenever one of these two terms is used in this thesis, we are referring to a recursive caching name server.

As its name implies, a recursive caching name server performs a process called *DNS recursion* and it caches the results of this process. Figure 2.5 shows the DNS recursion for **www.example.com** to continue the example from above. The figure shows the following steps of the recursion process⁸:

1. **Query to an authoritative name server for the root** – the resolver will start by sending the query for the **A** record for **www.example.com** to one of the authoritative name servers for the root of the DNS. The root

⁸Note that the figure shows a full recursion, which will only take place if none of the intermediate results required by the process are cached on the DNS resolver.

Figure 2.5: Example of a DNS recursion for `www.example.com`

name servers are operated on a vast, globally distributed infrastructure. The IPv4 and IPv6 addresses of the thirteen root name servers are well-known and preconfigured in most DNS resolver software. When a resolver first starts up it will typically perform what is known as a *root priming query* [119]. This means that it sends a query to one of the known root name server addresses to request the set of authoritative name servers for the root (NS query). It uses this to prime its cache with up-to-date information on the authoritative name servers for the root. Since the root name servers are not authoritative for `example.com`, they cannot respond to the query. The root, however, has a delegation for `.com`, and the queried root name server will respond with a *referral* that includes the list of authoritative name servers for the `.com` TLD. In other words: the root responds with *'I do not know, ask .com'*.

2. **Resolve addresses for .com authoritative servers** – a resolver with an empty cache will not have the addresses for any of the `.com` authoritative servers, in which case it will need to perform a separate recursion process to resolve these addresses in order to be able to query one of these servers.
3. **Query to a .com authoritative name server** – the resolver will now send a query for the A record for `www.example.com` to one of the `.com` authoritative name servers. Again, these name servers are not authoritative for `example.com` and will not know the answer. Thus, they will also

<i>domain name</i>	<i>TTL</i>	<i>class</i>	<i>type</i>	<i>value</i>
google.com.	172800	IN	NS	ns1.google.com.
google.com.	172800	IN	NS	ns2.google.com.
google.com.	172800	IN	NS	ns3.google.com.
google.com.	172800	IN	NS	ns4.google.com.

Example 2.3: Delegation for `google.com` in the `.com` zone

respond with a referral, in this case to the delegation they have to one of the `example.com` authoritative name servers.

4. **Resolve addresses for `example.com` authoritative servers** – just like for `.com`, above, the resolver may not have the addresses for the `example.com` name servers in its cache, in which case it will perform a separate recursion process to resolve these addresses.
5. **Query to an `example.com` authoritative name server** – finally, the resolver sends an A record query for `www.example.com` to one of the `example.com` name servers. Since these are authoritative for the domain, they will return the requested response, the IPv4 address associated with `www.example.com`.
6. **Respond to client and cache** – once it knows the response to the query, the resolver returns the response it has learned to the client (in our example the stub resolver in the operating system) and it stores a copy of the response in its cache. This is also where the TTL comes into play, responses may not be cached for longer than the TTL specifies (but may, of course, be cached for a shorter period of time, for instance because a cache is full). Note that in case a resolver can answer a query from a client from its cache, it sets the TTL in the response to the remaining TTL, that is: the number of seconds that the record will remain in its cache. This ensures that cached records expire correctly if, for example, stub resolvers cache data, or when resolvers are chained such that one caching resolver forwards (most) queries to another upstream resolver.

To aid the recursion process, in case of a referral, so-called *glue records* may be added to a DNS zone. Glue records are non-authoritative address records for the name servers of a domain. In particular, glue records are used to prevent circular dependencies in the DNS. Take, for example, `google.com`. Example 2.3 shows the delegation for `google.com` in the `.com` zone. As the example shows, all four name servers are under `google.com` itself (this is sometimes referred to as *in bailiwick*). A resolver with an empty cache would be unable to resolve any name in `google.com` without knowing the addresses for any of these four

name servers. However, to be able to resolve those addresses it would need to know the address for a name server for `google.com`, etc., etc. To remedy this, the `.com` zone includes glue records with the `A` and `AAAA` records for the four Google name servers. If a `.com` authoritative name server returns a referral for `google.com`, it includes these glue records in the additional section of the DNS response.

2.4 Cache Poisoning: a Weak Link in the DNS

When the DNS was designed in the 1980s, security was not an important design consideration. At that time, the number of hosts on the Internet was still very small⁹. This meant that interpersonal trust between operators connected to the Internet made it unnecessary to protect the DNS against malicious activity. As the Internet grew, these implicit early assumptions ceased to hold. Given the importance of the DNS for the Internet, it was only a matter of time before the first attacks on the DNS showed up. In this section, we consider a particular form of attack called *cache poisoning*. As the name suggests, the aim of a cache poisoning attack is to inject falsified information into the cache of a recursive caching name server. This is a particularly powerful class of attack, as a recursive caching name server may have many clients depending on it. For example, residential Internet users typically use one or more DNS resolvers provided by their ISP. Injecting false information into the cache of such resolvers thus potentially affects all customers of the ISP. In the following subsections, we discuss variants of cache poisoning attacks, which ultimately led to the Internet community developing and deploying DNSSEC, which will be discussed in Section 2.5.

2.4.1 The Origins of Cache Poisoning

The first description of cache poisoning in the literature dates back to 1995 [25]. As the author, Bellovin, wrote in his paper, however, the issue had already been uncovered in 1990. Bellovin states in the paper that he held back from publishing the work for fear of it being exploited to break in to remote systems. In the epilogue to the paper, Bellovin also discusses why he decided to eventually publish the work in 1995. A second paper by Vixie [201], describing how the Berkeley Internet Name Daemon (BIND) addressed the vulnerabilities discussed in Bellovin's paper, was published at the same conference.

⁹The number of hosts on the Internet crossed 1.000 in 1984, 10.000 in 1987 and 100.000 in 1989 [221].

<i>Query:</i>				
;; QUESTION SECTION				
www.alternic.net.		IN	A	
<i>Response:</i>				
;; QUESTION SECTION				
www.alternic.net.		IN	A	
;; ANSWER SECTION				
www.alternic.net.	3600	IN	A	207.51.48.15
;; ADDITIONAL SECTION				
www.internic.net.	604800	IN	A	207.51.48.15

Example 2.4: Example of Kashpureff cache poisoning

Bellovin alludes to anecdotal evidence that cache poisoning had been abused by attackers before he even published his paper. The first well-documented instance of cache poisoning, however, was performed by a man named Eugene Kashpureff. As already mentioned in Section 2.3.1, Kashpureff was one of the founders of a company called AlterNIC. In the Internet gold rush of the 1990s, AlterNIC attempted to compete with the then legitimate administrators of the DNS, InterNIC¹⁰. AlterNIC hosted an alternative DNS hierarchy, in which they introduced new TLDs in which users could register domain names (at a lower price than InterNIC charged for domain names). The problem with such an alternative DNS root system, of course, is that by default, software talked to the legitimate root of the DNS, which had no knowledge of this alternative system. Over time, AlterNIC got into an escalating discussion about who controlled what on the Internet (the specifics of which are outside the scope of this thesis). This eventually led to a drastic action by Kashpureff, who decided to hijack InterNIC's website. He used a cache poisoning attack variant that relied on a vulnerability in BIND, the most widely used DNS implementation at that time. The vulnerability boils down to BIND caching DNS records from the additional section of a DNS response, regardless of whether these additional records have any relation to the query, answer or authority section of the response. This flaw can be exploited by a malicious authoritative name server, by including the record it wishes to inject in a victim's cache into responses it sends. It is important to note that Kashpureff did not invent this attack, as the outlines of the attack had already been published [25, 201]. Nevertheless, the attack is sometimes referred to as 'Kashpureff cache poisoning'.

¹⁰Operated by the Network Solutions corporation.

Example 2.4 shows an example of a Kashpureff-style cache poisoning attack¹¹. The example shows the query sent by the DNS resolver, whose cache is being poisoned, and the response sent by the malicious authoritative name server. In the example, the malicious name server includes a falsified entry for `www.internic.net`, which points to the same IPv4 address as AlterNIC's own website. Because of the vulnerability in BIND, this record will get cached. Of course, the attacker will set a high TTL on the falsified record (in this case of seven days), to ensure that it stays in the cache for a long time. All the attacker now has to do to poison a resolver's cache is to get it to send a query to the malicious authoritative name server it controls. The attacker does this by issuing a query to the resolver for, e.g., `www.alternic.net`.

To remedy this attack, DNS resolver implementations now perform *bailiwick checking*. This means that they check whether the domain names of records in the additional section are in the same domain as the question in the query. If they are not, the resolver discards these records. It is easy to see that this prevents a Kashpureff-style attack as shown in Example 2.4; the `A` record for `www.internic.net` included in the additional section is clearly not in the bailiwick of the domain in the question section (`alternic.net`).

Nevertheless this still leaves the DNS vulnerable to another form of cache poisoning. As discussed in Section 2.3.2, each DNS query is identified by a Query ID. Since DNS normally uses UDP to transport messages, it is susceptible to *IP address spoofing*¹². An off-path attacker who is capable of sending packets with a falsified source IP address can exploit this to attempt to poison a DNS resolver's cache. Schematically, this attack looks like this:

1. The attacker issues a query for the name and record they wish to poison to a victim resolver.
2. If and only if the resolver does not have the requested record in its cache, the resolver will issue a query to the authoritative name server for the domain the name is in. If the resolver has the requested record in its cache, the attack fails here.
3. At this point, an arms race starts. The attacker starts sending falsified responses to the DNS query that the resolver sends to the authoritative name server. In these falsified responses, the attacker spoofs the source address of the response to be that of the legitimate authoritative name server. Furthermore, as the attacker is off-path and does not know this value, the attacker guesses the Query ID used by the resolver in the query

¹¹This, and other examples use the notation for DNS queries and responses as used by the `dig` tool included in the BIND DNS software distribution.

¹²IP spoofing will be discussed in more detail in Chapter 4.

to the authoritative name server. The attacker sends many packets with different Query IDs.

4. If a falsified response arrives at the resolver and the Query ID matches the Query ID used by the resolver, the attack succeeds. The resolver will accept this response, because it cannot distinguish it from a legitimate response, and caches the result. If the legitimate response to the query from the real authoritative name server arrives before the attacker is able to send a response with a matching Query ID, the attack fails.

Intuitively, one would think that the attacker has only a small chance of success in step 3. The Query ID is a 16-bit field, so the probability of guessing the correct ID is small. This is, however, not the case, for two reasons. First, many DNS resolver implementations used fixed (older versions of Windows) or sequential, and thus predictable (older versions of BIND) query IDs. And even software that purported to use randomised IDs had vulnerabilities that made it easy to guess the query ID. Until 2007, the BIND DNS implementation had a flaw in its pseudo random number generator (PRNG) that made it trivial to predict query IDs [115]. Even assuming that a resolver uses a good PRNG and thus good random query IDs, the number of guesses required for an attacker is still much lower than one would intuitively expect, due to the Birthday Paradox.

$$n(p; d) \approx \sqrt{2d \cdot \ln \left(\frac{1}{1-p} \right)} \quad (2.1)$$

The Generalized Birthday Paradox, shown in Equation 2.1, states that given an input probability p and a set of d integers, the function $n(p; d)$ gives the number of attempts n required to have a probability p of randomly drawing the same number twice. Applied to the case of query IDs above, this means that in order to have a 50% probability of guessing the correct query ID, the attacker will only have to send approximately 300 falsified responses.

The search space, within which an attacker has to guess a correct value, can be further expanded by applying source port randomisation. Recall that DNS uses the UDP protocol. This protocol has the notion of ports, from which messages are sent and on which messages are received. Port numbers are 16-bit integers, and generally the use of ports 1 to 1024 is restricted. This leaves DNS implementers to send queries (and receive the accompanying responses) on one of $2^{16} \cdot (2^{16} - 2^{10})$ ports. If we apply the Generalized Birthday Paradox with probability $p = 0.5$, this means the attacker will have to send approximately 76,558 falsified responses to have a 50% probability of successfully performing this type of cache poisoning attack. While that may sound much harder, as we will see in Section 2.4.2, this is quite feasible.

<i>Query:</i>					
<hr/>					
;; QUESTION SECTION					
randomname.example.com.		IN	A		
<i>Response:</i>					
<hr/>					
;; QUESTION SECTION					
randomname.example.com.		IN	A		
;; ANSWER SECTION					
<i>Empty (no answers)</i>					
;; AUTHORITY SECTION					
randomname.example.com.	172800	IN	NS	www.example.com.	
;; ADDITIONAL SECTION					
www.example.com.	172800	IN	A	192.0.78.24	

Example 2.5: Example of Kaminsky cache poisoning

What we have not touched upon, yet, is that there is an entirely different reason why this form of the cache poisoning is not as serious as it may seem. Recall from step 2 above, that this form of the attack will only succeed if the record that the attacker is trying to poison has not already been cached by the resolver. Since resolvers cache records according to the TTL, this severely limits the window of opportunity within which an attack can be performed. Even with a very small TTL of, say, 60 seconds, this means the attacker can only attempt a new attack once every minute.

Unfortunately, as we will see in the next section, there is a more sophisticated variant of this attack that does not require the attacker to wait until the TTL of a record expires.

2.4.2 The Kaminsky Attack

In 2008 security researcher Dan Kaminsky discovered a much more serious variant of cache poisoning attack [112]. Unlike the attack variant discussed at the end of the previous section, Kaminsky’s variant of the attack does not require the attacker to wait until the TTL of a record expires. Instead, Kaminsky found a way to circumvent this waiting period by querying for non-existent names in the domain the attacker wants to poison.

Example 2.5 shows how Kaminsky’s attack works. In his variant, the attacker sends a query for a random name in the domain in which they want to poison a record. Since the query name is random, the resolver will not have the response in its cache and is guaranteed to send a query to the authoritative name server of the domain. The attacker then starts sending falsified

responses to this query. However, instead of falsifying the answer directly in the response, as was the case in the last variant described in the previous section, the attacker sends a *falsified referral*. In other words: the attacker does what normal authoritative name servers do during the DNS resolution process¹³, and says: ‘*I do not know the address for `randomname.example.com`, but you can ask `www.example.com`*’. The attacker very helpfully also includes a *falsified glue record* for `www.example.com`, which promptly gets cached by the resolver. The attacker can perform this attack at will, instead of having to wait until the TTL expires, and if they guess the query ID and source port used by the resolver correctly, can poison the cache of the resolver.

Patching Against Kaminsky

Unlike Kashpureff, who exploited a known DNS vulnerability for his own purposes, Kaminsky practiced responsible disclosure and worked with all major implementors of DNS software to implement appropriate patches. Many DNS implementations did not properly implement query ID randomisation or source port randomisation at the time Kaminsky discovered the vulnerability. As we have seen at the end of Section 2.4.1, due to the Birthday Paradox this makes it very easy to perform a successful attack. The ensuing mass update of DNS software did not escape the media’s notice. The effort was extensively covered in tech media (e.g., [1, 144]) and in the main stream media (e.g., [33]). When there was reasonable certainty that the majority of systems had been patched, Kaminsky finally unveiled the details of the vulnerability he had uncovered at the Black Hat security conference [112].

Unfortunately, applying patches to DNS resolver software to introduce query ID and source port randomisation are only a stopgap measure. Soon after the vulnerability was unveiled, discussion ensued whether these patches really prevented successful attacks. Research soon after showed that indeed it did not, it just extended the time required to perform the attack from several seconds to several days [70]. While this might seem like an improvement, an attacker probably has no hard deadline, and can patiently wait for their attack to succeed.

The debate in the Internet community soon focused on deployment of DNS-SEC, the development of which, as we will discuss in Section 2.5, was at least partially driven by the need to deal with cache poisoning. There were, however, also efforts to improve the DNS in other ways, to mitigate this form of attack. A notable example is the DNS ‘0x20’ approach [51]. Recall that DNS is case-insensitive, and that the question section from a query is copied into the response sent back by a server. The ‘0x20’ approach leverages this by encoding a bit pattern in the query name in the question section using capital letters and

¹³See also steps 1 and 3 in Section 2.3.3.

lower case letters¹⁴. This potentially adds a few extra bits of entropy to that of the random query ID and random source port, making it more difficult for the attacker to spoof a response that is accepted by the resolver. According to the authors, over 99.7% of DNS implementations at the time of their study supported this approach. The idea never gained much traction, however, as consensus was that this was just another stopgap measure, and that the real solution would be to deploy DNSSEC.

Beyond Kaminsky

More recently, another cache poisoning attack was discovered, which relies on IP fragmentation [91]. This attack leverages extensions to the DNS that allow for DNS responses larger than 512 bytes (see Section 2.5.2 below). If an attacker can force DNS responses to be fragmented, they can insert falsified records in the DNS message by spoofing one of the follow-up fragments after the first fragment. This completely circumvents query ID and source port checking by the resolver, as the query ID and port number will only be present in the first fragmented of a response that has been fragmented at the IP level. Any follow-up fragments just contain data with the remainder of the message. All the attacker has to do to succeed is ensure that the IP packet identifier (IPID) in the spoofed fragment is correct. This is not hard as, as the authors of [91] point out, many IP stack implementations use sequential IPID values. And even if this value is randomised, it is only a 16-bit value in IPv4, which means it is trivial to get a high attack success rate due to the Birthday Paradox. Ironically, deployment of DNSSEC may actually make this particular cache poisoning variant more likely to be successful. Deployment of DNSSEC results in large message sizes that make fragmentation more likely. And even if resolvers do not perform DNSSEC validation, they often still ask for DNSSEC data, and may get fragmented responses because of this. The problem of DNSSEC and fragmentation is covered in much more detail in Chapter 3.

2.5 The DNS Security Extensions

The previous section has shown that cache poisoning is a serious vulnerability in the DNS. Furthermore, it showed that it is hard, if not impossible, to completely protect against this vulnerability within the confines of the original DNS protocol specification. This led the Internet Engineering Task Force (IETF) to start considering securing the DNS. From 1997 onward, the DNS Security Extensions (DNSSEC) were developed to address the challenge of securing the

¹⁴Hence the name ‘DNS 0x20’, as the difference between upper case and lower case letters is 32 (0x20 hexadecimal) in the ASCII encoding.

DNS against, for example, cache poisoning attacks. In this section, we will discuss how DNSSEC came into being, what changes in the DNS it builds on and how it works.

2.5.1 A Brief History of DNSSEC

Development of DNSSEC started in 1994, when work got started on the first RFC in an IETF workgroup tasked with developing the DNS Security Extensions. The first RFC resulting from this effort was published in 1997 [64]. The basic premise of DNSSEC – the use of digital signatures, based on public key cryptography – was already introduced in this specification. In DNSSEC, records in a DNS zone are digitally signed. If properly executed, digital signatures cannot be forged, and can only be created by the entity holding a secret cryptographic key. These signatures can then be validated by the recipient of a DNS message using the public key that accompanies the secret signing key. If the signature is valid, this proves the *authenticity* of the data in the message, as only the owner of the secret key could have created the signature. In addition to this, a valid signature also proves that the message was not modified in transit (the *integrity* of a message), as any change to the content of the message would invalidate the signature.

The original specification did not see any significant uptake by DNS implementors and operators. There was, however, a lot of feedback on the specification from potential adopters, which led to a revision that was published in 1999 [65]. This version of the specification was implemented in BIND, but again, it saw very little uptake. Furthermore, experiments by potential adopters showed that there were still problems with the protocol, especially with the management of signing keys. This, again, led to a working group in the IETF revisiting the DNSSEC specifications. This group was tasked with revising the protocol to deal with the problems that were reported. The DNSSEC-bis working group started work in 2001. One of the main improvements introduced by this working group was the Delegation Signer (DS) record type, which, as we will see in Sections 2.5.3 and 2.5.4 below, plays an important role in establishing trust between parent and child domains. The work of the DNSSEC-bis working group eventually led to a set of three specifications [15, 16, 17] published in 2005 that still form the foundation of the DNSSEC protocol as it is in use today.

2.5.2 Prerequisites for DNSSEC: Larger DNS Messages

DNSSEC adds extra data to DNS messages in the form of digital signatures. As discussed in Section 2.3.2, the original DNS specification limits DNS message sizes to a maximum UDP payload of 512 bytes. This is problematic for

Figure 2.6: EDNS0 OPT resource record specification

DNSSEC. Take, for example, a DNSSEC message that includes one signature and a public key. If the key used is a 2048-bit RSA key, then the representation of the signature and key alone would already require more than 512 bytes. For this reason, DNSSEC relies on the so-called Extension Mechanisms for DNS, colloquially known as 'EDNS0' [52].

The EDNS0 specification adds a way for DNS clients and servers to signal to each other that they have additional capabilities. This is done by including a pseudo resource record in the additional section of DNS queries and responses, the so-called OPT resource record. Figure 2.6 shows a detailed specification of the OPT record. Two fields in this record (marked in red) are important for the work in this thesis. First of all, the requestor's UDP payload size (encoded in the query class field). This value indicates to the recipient of an EDNS0 message what the maximum message size is that the sender is capable of processing. Second, the DNSSEC OK (DO) flag. If this flag is set in an EDNS0 message, it indicates the requestor wishes to receive DNSSEC data if available.

Just like DNSSEC, EDNS0 was defined to be backward compatible. Implementations that do not support the OPT resource record will simply ignore it, or return an error message. This should be interpreted by the sender of the EDNS0 message as a signal that the recipient does not support EDNS0.

2.5.3 DNSSEC Signing

DNSSEC signing is performed by the owner of a domain, and is done at the zone level. DNSSEC was designed to allow both offline and online signing. That is: the zone may be signed separately, before it is loaded and served from an authoritative name server, but signing may also be performed just-in-time as the zone is being served by an authoritative name server. A discussion of the

	<i>domain name</i>	<i>TTL</i>	<i>class</i>	<i>type</i>	<i>value</i>	
1	example.com.	86400	IN	A	93.184.216.34	← RRset #1
2	example.com.	86400	IN	RRSIG	A 8 2 86400 ...	← signature for RRset #1
3	example.com.	86400	IN	AAAA	2606:2800::...:1946	← RRset #2
4	example.com.	86400	IN	RRSIG	AAAA 8 2 86400 ...	← signature for RRset #2
5	example.com.	86400	IN	NS	a.iana-servers.net.	} ← RRset #3
6	example.com.	86400	IN	NS	b.iana-servers.net.	
7	example.com.	86400	IN	RRSIG	NS 8 2 86400 ...	← signature for RRset #3
8	example.com.	3600	IN	DNSKEY	257 3 8 AwEAAaCU...	} ← Set of public keys
9	example.com.	3600	IN	DNSKEY	256 3 8 AwEAAZ0a...	
10	example.com.	3600	IN	RRSIG	DNSKEY 8 2 3600 ...	← signature over DNSKEY set

Example 2.6: Example DNSSEC-signed zone snippet for `example.com`⁷

considerations for choosing between these two options is outside the scope of this thesis. The remainder of this subsection explains the signing process, how the DNSSEC chain of trust works and how DNSSEC deals with non-existent names.

Zone Signing

To sign a zone, the signer first needs to generate one or more cryptographic key pairs. The secret key of the key pair is used to create the actual signatures, the public key is included in the DNS zone. The public key will be used by recipients of DNSSEC-signed responses to validate the signatures included in the response. Keys are included in DNSSEC-signed zones using the `DNSKEY` record type. DNSSEC signatures always cover resource record sets (RRsets). The concept of RRsets was explained in Section 2.3.2 and is shown in Example 2.2. For each RRset a signature is included in the DNSSEC-signed zone.

Example 2.6 shows part of a signed zone for `example.com`. Lines 1, 3 and 5–6 show three RRsets, and lines 2, 4 and 7 show the signatures over each of these RRsets. Lines 8–9 show the `DNSKEY` records with the public keys with which the signatures can be validated. Line 10, finally, shows that all RRsets are signed, including the `DNSKEY` set.

Figure 2.7 shows a detailed view on the fields of the `RRSIG` resource record type. An actual signature record for `ietf.org` was used for illustration purposes. The following fields are shown, as specified in [16] (from left to right, top to bottom):

- **Type covered** – indicates the type of the RRset covered by this signature, in this case the RRset with `A` records for `ietf.org`.

Figure 2.7: RRSIG fields

- **Algorithm** – the signature algorithm used (in this case RSA PKCS#1 signatures [142] that use the SHA1 hash algorithm¹⁵).
- **Label count** – the number of DNS labels in the name covered by this signature. This field is used by validators to distinguish between signatures that cover a single name and those that cover a wildcard.
- **Original TTL** – the original TTL of the RRset covered by the signature. As discussed in step 6 of the DNS resolution process (Section 2.3.3), resolvers typically return the remaining TTL for cached records. Validators that use this data as input for validation must know the original TTL in order to be able to validate the signature [17].
- **Signature expiration** – the time at which the signature expires, and should be considered invalid. This time is represented in text as the concatenation of the year, month, day, hour, minute and second and on the wire as a UNIX epoch timestamp¹⁶. The signature in the example is thus valid until September 21, 2017 at 19:32:56h UTC¹⁷.
- **Signature inception** – the time from which the signature is valid. The signature in the example is thus valid from September 21, 2016 at 18:33:46h UTC.
- **Key tag** – this 16-bit value is meant as a hint to validators which key should be used to validate the signature. Note that key tags are not guaranteed to be unique, so should only serve as a way to select candidate keys for validation.

¹⁵ An overview of current identifiers for DNSSEC can be found at <https://www.iana.org/assignments/dns-sec-alg-numbers/dns-sec-alg-numbers.xhtml>

¹⁶ A 32-bit unsigned integer representing the number of seconds elapsed since January 1, 1970, 00:00:00h UTC.

¹⁷ And thus until after the date this thesis is defended.

- **Signer's name** – the owner name of the DNSKEY resource record that validators are supposed to use to validate signatures. Generally, this is the zone name of the zone the RRset covered by the signature is in.
- **Signature data** – the actual signature data, the format of which is specific to the indicated signing algorithm.

Signatures are computed as follows (where ‘|’ denotes concatenation, $\mathcal{S}$ is a digital signature function, $\mathcal{C}$ is a function that converts names to canonical format and $\mathcal{O}$ is a function that canonically orders resource records in an RRset):

```

signature =  $\mathcal{S}(\text{RDATA}_{\text{RRSIG}} \mid \text{RRsetData})$ 
RDATARRSIG = typeCovered | algorithm | labelCount |
               originalTTL | expiration | inception |
               keyTag |  $\mathcal{C}(\text{signerName})$ 
RRsetData =  $\mathcal{O}(\text{RR}[1] \mid \text{RR}[2] \mid \dots)$ 
RR[i] =  $\mathcal{C}(\text{ownerName}) \mid \text{type} \mid \text{class} \mid$ 
          originalTTL | RDATALength | RDATA

```

The canonical representation of a domain name consists of the full representation of the name including the root label. Representing names canonically is needed because DNS messages may contain compressed names (see Table 2.2). Equally, the resource records that make up an RRset need to be put in canonical order before the signature is generated. Otherwise, RRsets containing multiple RRs can have different signatures depending on the order of the RRs in the set.

Chain of Trust

Before a validator can check the validity of signatures, it first needs to establish that it can trust the public keys it will use to perform the validation. It would be inconvenient if validators need to establish the trustworthiness of the public keys in every signed zone. Therefore, DNSSEC is designed to establish a *chain of trust* between parent and child zones. Normally, this chain of trust extends all the way from the root of the DNS down to an individually signed record. The chain of trust is established by creating a reference in the parent to a public key in the signed zone. This is done using the DS record type. The parent zone contains one or more DS records, that each reference a public key in the child zone that is a so-called *secure entry point*. This secure entry point is a key that ultimately is the root of trust for all signatures in a zone.

Figure 2.8: DNSKEY and DS record fields

Figure 2.8 shows a detailed view of the fields of DNSKEY and DS records respectively. The example is based on real records for the `ietf.org` domain. For DNSKEY records the following fields are shown:

- **Flags** – 16-bit field that specifies properties of the key. The most important flags are encoded in bit 7 and bit 15. Bit 7 is always set on keys used to sign resource records. If bit 15 is set, this indicates that the key is considered a secure entry point. Note that it is not mandatory to set this bit, it is only intended as a hint to validators. In the example, bits 7 and 15 are set.
- **Protocol** – indicates the DNSSEC protocol used. This is always set to 3 for the current DNSSEC protocol [16].
- **Algorithm** – indicates the signing algorithm. Uses the same identifiers as the RRSIG record¹⁵.
- **Public key data** – encodes the public key. The encoding is specific to the cryptographic algorithm that is used.

DS records have the following fields:

- **Key tag** – the key tag provides a hint about which DNSKEY record this DS refers to. The key tag is computed in the same way as the key tag in RRSIG records.
- **Algorithm** – the algorithm of the DNSKEY this DS record refers to¹⁵.

Figure 2.9: DNSSEC chain of trust for `example.com`

- **Hash algorithm** – the hash algorithm used to create a hash of the public key data in the DNSKEY record this DS refers to¹⁵.
- **Hash of public key data** – the actual hash of the public key of the DNSKEY this DS record refers to.

Figure 2.9 shows how this all fits together into a trust chain. The figure shows the trust chain for the `example.com` domain. Assuming that a validator wishes to validate the signature over a resource record for `www.example.com`, they can traverse the trust chain from that signature all the way up to the signing keys for the root of the DNS. What the figure also shows is the most common configuration for signed zones, which is to have two keys. The first key is called the *Key Signing Key* (KSK). This key is the secure entry point for the zone and is only used to generate a signature over the DNSKEY RRset. The second key is called the *Zone Signing Key* (ZSK). This key is used to generate the actual signatures over the RRsets in the zone. The rationale behind this key management model is discussed below when we discuss the operation of signed zones. Note also that the trust chain always ends at a *trust anchor*. In almost all cases, the trust anchor is the KSK for the root zone of the DNS. Validating resolvers typically have the root trust anchor as one of their configuration parameters, as they need this anchor to be able to validate all other signatures. Finally we note that DNSSEC effectively turns the DNS into a public key infrastructure (PKI). The DNSSEC PKI shows similarities to the X.509 PKI infrastructure used for TLS on the web. X.509 certificates are typically issued by a higher authority, and browsers normally do not trust individual certificates, but rather have a repository of trusted root authorities that issue certificates. The biggest difference between the X.509-based PKI used on the web and DNSSEC is, of course, that the DNS only has a single root authority.

	<i>domain name</i>	<i>TTL</i>	<i>class</i>	<i>type</i>	<i>value</i>	
1	a.example.com.	86400	IN	A	10.0.0.1	← RRset #1
2	a.example.com.	3600	IN	NSEC	c.example.com. A	
3	c.example.com.	86400	IN	A	10.0.0.2	
4	c.example.com.	86400	IN	AAAA	fe80::2	} ← RRset #2
5	c.example.com.	3600	IN	NSEC	k.example.com. A AAAA	
6	k.example.com.	86400	IN	TXT	"some remark"	← RRset #3
7	k.example.com.	3600	IN	NSEC	a.example.com. TXT	

Example 2.7: Example of NSEC records in `example.com`⁷

Authenticated Denial of Existence

DNSSEC guarantees the authenticity of the records in a zone, and with that the presence of these records. But what if a record does not exist? In the DNS, name servers return a non-existent domain (NXDOMAIN) response if a record does not exist, but how can we trust that this is the case? DNSSEC also addresses this problem through a mechanism called *authenticated denial of existence*. Next to signing records in a zone, DNSSEC also generates cryptographically signed proofs of non-existence. These allow validators to verify that the name and record type in a query, for which they have received an NXDOMAIN response, do indeed not exist.

The original DNSSEC specification [16] defines the Next Secure (NSEC) record for this purpose. When signing a zone, the signer generates a chain of NSEC records that prove the existence of gaps in a zone, in which no names exist.

Example 2.7 shows a toy NSEC chain in the `example.com` domain¹⁸. The zone in the example is sorted in canonical order. For each of the RRsets in the zone, on lines 1, 3–4 and 6 the corresponding NSEC records are shown. On line 2, the NSEC record has two strings in its value field. The first string is the *next secure* name, in this case `c.example.com`. What the signer states with that is that no records for names between ‘a’ and ‘c’ exist in the `example.com` zone. The second string lists the types for which records exist that have ‘a’ as owner, and in this case only an A record exists (shown on line 1). By showing what records exist, the signer implicitly asserts that no records of other types exist with owner name ‘a’. The NSEC record on line 5 shows that no names exist between ‘c’ and ‘k’ and that only an A and a AAAA record exist for ‘c’. Finally the NSEC chain is completed by the record on line 7, which completes the circle by stating that no records exist between ‘k’ and ‘a’. Since ‘a’ comes before ‘k’ in canonical order, this wraps the chain.

¹⁸Note that for clarity, signatures are not shown.

Soon after the release of the DNSSEC specifications, there was criticism on the NSEC authenticated denial of existence approach. What makes NSEC problematic is that it allows so-called *zone walking*. By sending targeted queries for a signed zone that uses NSEC authenticated denial of existence, a querying party can establish which names exist in a zone, since the NSEC record conveniently lists the next secure (and thus existing) name. This means it is trivial to enumerate all records in a signed zone that uses NSEC. Some consider it undesirable that external parties can learn what names exist in a zone, as this may make it easier to perform targeted attacks.

To mitigate the problem of zone walking, an alternative mechanism was introduced, called NSEC3 [128]. In NSEC3, the next secure name is represented as a hash of an owner name, rather than the name itself. Canonical ordering of NSEC3 records is then done by ordering hash values. This means that trivial zone walking no longer uncovers names in the zone, but rather hashes of names in the zone. Since hashes are fairly easy to reverse with dictionary attacks, NSEC3 has the option of adding salt (a random value) into the hash function prior to hashing and to hash multiple times, to increase the computational burden for an attacker attempting to reverse the hash. There has been a lot of debate about whether or not this is effective. Recent research seems to suggest that this may not be the case; by harnessing the computational power of GPUs NSEC3 can be reversed quite successfully [78, 204]. The doubt whether NSEC3 actually protects zones against leaking names has led to researchers proposing another authenticated denial of existence mechanism that they have called NSEC5 [76, 77, 151]. The researchers that proposed NSEC5 have created a draft specification for discussion in the IETF. Given that NSEC5 is in the early stages of development and not yet under formal consideration for standardisation, we do not discuss it in detail in this thesis.

DNS Operation of a Signed Zone

As discussed earlier, DNSSEC signatures have a limited validity. This means that signatures need to be refreshed at regular intervals. In addition to this, best current practice suggests replacing signing keys at regular intervals [121, 143]. This has consequences for DNS operators that deploy DNSSEC signing. Where classic DNS only requires an occasional intervention to change a DNS zone, DNSSEC requires regular maintenance. Fortunately, there is mature DNSSEC software that can automate most of these processes.

First we look at re-signing DNSSEC zones. Most DNSSEC-capable software implementations can automate this process for operators. Typically, and as the best current practice [121] suggests, this means replacing signatures some time before they expire. This provides operators some respite if, for example,

Inception	Expiration	Validity period Hours (Days)	Time Before Old Signature Expires	Remarks
2016-09-23 20:55:17	2016-10-23 21:37:52	±721h (±30d)	n/a	
2016-10-16 08:37:52	2016-11-15 08:51:33	±720h (±30d)	181h	Regular re-signing
2016-10-21 20:13:26	2016-11-20 21:13:26	721h (±30d)	±589h	Dyn attack [93, 136]
2016-11-13 08:13:26	2016-12-13 08:31:46	±720h (±30d)	181h	Regular re-signing
2016-11-23 02:06:59	2016-12-23 03:06:59	721h (±30d)	±486h	A RRset changed
2016-12-15 14:06:59	2017-01-14 15:05:45	±721h (±30d)	181h	Regular re-signing
2017-01-07 02:05:45	2017-02-06 02:19:50	±720h (±30d)	181h	Regular re-signing
2017-01-29 13:19:50	2017-02-28 14:17:36	±721h (±30d)	181h	Regular re-signing

Table 2.4: Signature validity for signatures on the A RRset for `paypal.com`

re-signing fails for some reason (e.g., a software crash). Take, for instance, the example shown in Table 2.4, which shows signature validity for the signatures on the A RRset for `paypal.com` over the period October 2016 – January 2017. This example shows four regular automated re-signing events, which all occur exactly 181 hours before the old signature would expire. The example also shows two deviations from the pattern, which indicate that some change in the zone triggered re-signing. The first instance is most likely due to PayPal reacting to the denial of service attack on their DNS provider, Dyn [93, 136]. This signature refresh occurs on the day of the attack, and the timestamp of the signature is after the attack. The set of name servers for `paypal.com` changed around the same time, which is highly likely to be a response to the attack¹⁹. The second deviation from the regular pattern is due to a change in the A RRset, which, of course, required a fresh signature. One other thing to note is the validity period of signatures. As the table shows, this validity period is not always exactly the same (although always around 30 days). This is most likely because the signer software introduces ‘jitter’ (see also [121]). The function of this jitter is to spread out future re-signing of records in a zone. If all signatures would expire at exactly the same time, this can cause peaks in working load on the signer, the use of jitter prevents this.

Second, we look at the regular replacement of keys. This practice is called *key rollover* and the best current practice suggests that this should be done on a regular basis. Again, just like for signature refreshes, modern DNSSEC software supports automated key rollover. Figure 2.10 shows how this type of automation results in very regular key rollover patterns. The figure shows all key rollovers for the `nobelprize.org` domain between March 2015 and January 2017. KSKs are shown in blue, on top, and ZSKs are shown in red underneath. As the figure illustrates, the life cycles of keys overlap. New keys are introduced before they are used to create signatures, because it takes at least the TTL of

¹⁹Before the attacks, the NS set exclusively contained Dyn-operated name servers, after the attack two name servers operated by another provider were added to the set, most likely as a way to spread risk.

Figure 2.10: Key rollover for `nobelpize.org` (March 2015 – January 2017)

the DNSKEY RRset before all caches will see a newly introduced key. If signatures with a key are published before the key has propagated to all caches, this can result in validation failures. Similarly, keys are removed some time after the last signatures generated with the key have disappeared from a zone. Again, this is to allow the TTL on RRsets with signatures made with the key to be removed to expire. A very important thing to note is that replacing a secure entry point into a zone generally requires interaction with the parent zone, because the DS will also need to be replaced. If the parent zone is a TLD, then this interaction typically takes place through the Registry-Registrar-Registrant channel (see Section 2.3.1). This is one of the key reasons for having a split key model with a KSK and a ZSK. The KSK can be cryptographically stronger, and longer lived (requiring fewer interactions with the parent zone), and the ZSK can be cryptographically weaker, but replaced more often, independent of the parent zone. There are a number of possible variations for key rollovers, these are discussed in the IETF’s best current practice [121]. Furthermore, Yang et al. discuss DNSSEC as an example of an Internet-scale PKI [217]. In their paper, they also reflect on DNSSEC key management and key rollover models. Finally, we note that key rollovers introduce complexity into DNSSEC. We will further discuss this complexity in Chapter 5.

2.5.4 DNSSEC Validation

The final aspect of DNSSEC is the validation of signatures. This task is typically performed by DNS resolvers, which are then referred to as *validating resolvers*. A validating resolver needs a trust anchor to be able to validate signatures. As discussed in the previous section, this trust anchor is normally the KSK for the root zone of the DNS.

Figure 2.11 shows a DNS recursion for `www.example.com` together with the signature validations typically performed by a validating resolver. Each point where a validation takes place is marked by a red and gold ribbon. The pattern

Figure 2.11: DNS recursion with validation for `www.example.com`

of validations at each level is similar, and conceptually consists of the following three steps:

1. **Fetch the DNSKEY** – the resolver needs to know the public keys to use to validate signatures in the zone.
2. **Check DNSKEY against DS and validate RRSIG** – to verify that the DNSKEY that was returned is the one in which the parent zone expressed trust, the resolver first checks if the DNSKEY set contains at least one key that matches a DS record in the parent zone. If that is the case, it validates the RRSIG record(s) in the response to the DNSKEY query to ensure that there is also a signature with a matching key over the DNSKEY RRset.
3. **Validate RRSIG in response to actual query** – finally, the resolver will validate the RRSIG record(s) in the response to the actual query using key(s) from the DNSKEY RRset.

There is one exception to this pattern. The root, of course, does not have a parent zone. Rather, whether the root zone returns the correct DNSKEY is checked against the trust anchor configured on the resolver.

Validating resolvers will normally cache the result of signature validations, to conserve resources. Signature validations are cryptographic operations and can use a substantial amount of CPU resources. Thus, if a signature verifies correctly, the resolver will mark this in its cache. Another consequence of signing is that it impacts how resolvers evaluate the TTL of records. Because signatures

have a limited validity, the signature may actually expire before the TTL of a record expires. Resolvers must therefore compute what time comes first: the expiry of the TTL or the expiry of the signature, and remove records from their cache accordingly.

As discussed at the end of the previous section, DNSSEC current best practices suggest to replace signing keys regularly. This has not happened in full for the root zone of the DNS. The root was signed in 2010. Since then, the ZSK has been replaced at three-month intervals. The KSK of the root, however, has not been replaced. This is scheduled to happen for the first time in the autumn of 2017. This has consequences for validating resolvers, as these will need to switch to a new trust anchor when this happens. Fortunately, the IETF has specified a protocol for tracking trust anchors as these are replaced [172]. All modern implementations of validating resolvers support this protocol, but only time will tell if all operational validating resolvers correctly pick up the new root trust anchor. This will be a major test of DNSSEC's operational resilience.

2.6 Building Internet Trust with DNSSEC

We have discussed how DNSSEC works, and that the goal of the protocol is to improve trust in the DNS by introducing authenticity and integrity for DNS responses. Once DNSSEC is deployed, however, this opens up new possibilities. In this final section of the chapter we will discuss how the community relies more and more on DNSSEC as a framework that can be used to enhance trust in other Internet protocols. This is done through DNS-based Authentication of Named Entities, or DANE for short.

2.6.1 DANE: a Brief Introduction

The DNS already fulfills a multitude of functions in resolving information about hosts for various Internet protocols and services. As we discussed earlier, DNSSEC effectively turns the DNS into a public key infrastructure (PKI). The idea with DANE is to leverage this PKI to use the DNS to enhance trust in Internet services. Many Internet protocols can use Transport Layer Security (TLS) [56] to protect communication by encrypting the transport channel. TLS servers authenticate themselves using X.509 certificates. These certificates may be issued by a trusted PKIX Certificate Authority (CA)²⁰, but in many cases these are also self-issued or otherwise untrusted certificates. The IETF published a case study for using the DNS to provide an alternative channel for proving the authenticity of the certificates used by a TLS server [24]. The concept was further

²⁰These are the CAs that, for example, come pre-configured in web browsers.

developed into a standard for encoding information about certificates in a new DNS record type called **TLSA** [61, 80, 96]. By including a **TLSA** record in a DNS zone, the operator of a service can provide information about the legitimate certificate(s) that a TLS server on a certain host and port may present to peers. **TLSA** records are intended to be used in a DNSSEC-signed zone, the DNSSEC signature on the **TLSA** record(s) thus proves that the record was inserted by the trusted operator of the DNS zone. There are two dimensions to the use of **TLSA** records. First, the scope of the trust expressed by a **TLSA** record. This scope may either cover issuers of valid certificates for a TLS server (i.e., CAs that are authorized to issue certificates that are used by the server), or the scope may specify a single specific certificate that may be used by the server. The second dimension is whether the **TLSA** record covers issuers or certificates that also have a valid PKIX trust path, or whether the record covers self-issued certificates. In the first case, **TLSA** provides additional security by offering a second independent channel to verify the validity of a certificate. In the second case, the **TLSA** record (or rather the DNSSEC signature on it) provides the sole PKI trust path to check the certificate's validity.

DANE has many applications, a number of examples are listed below:

- *Enhancing trust in PKIX certificates used to authenticate HTTPS web servers* – this is one of the first applications described for DANE. It would require validation of **TLSA** records on end hosts, by web browsers. This, however, is something that is generally considered complex and potentially error prone. For this reason, browser vendors have expressed little interest in implementing support²¹.
- *Providing a trusted entry point to find OpenPGP keys* – OpenPGP is an open standard [38] for encrypting data (such as e-mail and other documents). There is a standard [215] that leverages DANE as a means to find trusted OpenPGP keys.
- *Improving e-mail transport security* – the Simple Mail Transfer Protocol (SMTP) [116] normally transmits e-mail messages in clear text. This, of course, renders e-mail susceptible to surveillance. Mail transport can be secured using TLS, but there are a number of issues with this approach. As this is one of the most promising applications of DANE, we will discuss it in more detail in the next section.

DANE deployment is still in its infancy, as a study of its deployment shows [223]. Currently, the most promising application appears to be securing e-mail

²¹The ccTLD registry for .cz, CZ.NIC, has developed browser plugins for **TLSA** validation, see <https://www.dnssec-validator.cz/>

transport, which we discuss in more detail in the next section. There will, of course be other applications of DANE, but an exhaustive discussion of these is outside of the scope of this thesis.

2.6.2 Improving E-mail Security with DANE

As mentioned above, e-mail transport using the SMTP protocol can be protected using TLS, more specifically, the SMTP protocol has an extension called ‘STARTTLS’ [95]. When using this extension, peers connecting to an SMTP server may initiate secure communication by calling the STARTTLS command. This will start a TLS negotiation and all subsequent communication will proceed over the TLS-secured channel. It is highly desirable to secure e-mail communication, but SMTP over TLS suffers from a number of problems. Arguably, the two most important problems are that:

- *STARTTLS is vulnerable to downgrade attacks* – an on-path adversary able to relay the SMTP connection through a point controlled by them can pretend that the receiving end of the connection does not support STARTTLS.
- *Certificate validation is problematic* – while the TLS connection is being established, the server authenticates itself using an X.509 certificate. Unfortunately, SMTP servers often use self-issued certificates, and even if PKIX certificates are used validation is still problematic. If the connecting party does not know the issuer of a certificate, there is no way to signal this to an administrator. It then becomes a matter of policy what should be done if an untrusted certificate is encountered; should the connection be closed and e-mail not delivered? Or should the connection proceed and should the validation failure be logged? This may not scale very well on servers handling large numbers of e-mails, and in any case, if the validation failure was indeed a sign of some sort of attack, simply logging it, but proceeding with the mail transfer means that the damage is already done.

The problems with SMTP and STARTTLS led the IETF to publish a specification on how to secure e-mail transport using DANE [62]. By publishing TLSA records for SMTP servers, operators can make two things explicit. First, the presence of a TLSA record clearly signals support for STARTTLS, thus preventing downgrade attacks. Second, the TLSA record unambiguously specifies what is or are valid certificate(s) used by the SMTP server.

DANE for securing SMTP with STARTTLS is already seeing uptake. Support for DANE validation has been implemented in a number of SMTP server

implementations²². Concerns over surveillance of their citizens and state institutions have led the German Federal Government to issue specifications on securing e-mail communication with STARTTLS and DANE [3]. A number of German e-mail service providers have adopted these standards for their service. In a similar vein, the Dutch National Government has put STARTTLS and DANE on the comply-or-explain list [4]. This list is a guideline for vendors on what standards they need to support when putting in an offer on a public tender issued by the government.

²²At the time of writing, these were Exim, Postfix and Halon.

Introduction to Part 1

In the first part of this thesis we focus on addressing the first research goal: studying technical problems in DNSSEC that limit large-scale deployment. We start by studying the two technical problems in DNSSEC, namely fragmentation (in Chapter 3) and amplification (in Chapter 4). Then, we study how the use of alternative signature algorithms for DNSSEC, based on elliptic curve cryptography (ECC), can solve these two problems (Chapter 5). We end this part of the thesis with a study of the one major drawback of ECC, slow signature validation and how this impacts the global DNS infrastructure (Chapter 6).

DNSSEC and IP Fragmentation

In this chapter, we study the first of our two problems: fragmentation. Our goal is to answer the following questions: to what extent do DNS resolvers on the Internet experience problems with fragmented responses? And are there measures that an operator of a DNSSEC-signed domain can take to reduce fragmentation, to prevent problems for resolvers that cannot receive fragmented responses? The study discussed in this chapter was performed in 2012/2013 and was published as a research paper [180]. In addition to this, the results of the study and follow-up results were presented in a number of forums frequented by operators and developers of DNS software (e.g., RIPE meetings, the USENIX LISA conference and ICANN DNSSEC workshops [183, 185, 186, 187]). We discuss these follow-up results and the operational recommendations stemming from them in the last part of this chapter.

3.1 Introduction

In the previous chapter, we explained that DNSSEC relies on an extension mechanism to the DNS, called EDNS0 (Section 2.5.2). The fact that DNSSEC responses include digital signatures means that DNSSEC responses will frequently not fit in a classic DNS datagram, with a maximum payload size of

Figure 3.1: Problems with a firewall blocking fragmented DNSSEC responses

512 bytes. A 2010 study, for example, by Cowperthwaite and Somayaji [50] found an eleven-fold increase in response size for DNSSEC when compared to classic DNS.

In some cases, DNSSEC responses are so large that they are fragmented into multiple IP fragments. Fragmentation is part of the core protocol specifications of both IPv4 [155] and IPv6 [53]. Fragmentation occurs when the message size exceeds the Maximum Transmission Unit (MTU) size of the underlying medium over which the IP traffic is transported. The MTU for the most common medium, Ethernet, is 1500 bytes [99]. Theoretically, the minimum MTU for IPv4 is 68 bytes, although a reasonable minimum value in practice is 576 bytes [155]. The IPv6 specification lists a minimum MTU of 1280 bytes.

For DNSSEC, fragmentation poses a problem¹. Firewalls are frequently configured to block fragments, to protect against certain types of fragment-based attacks [166]. As a consequence, DNSSEC responses that are fragmented may not arrive at their destination. Firewalls that block fragments introduce problems as shown in Figure 3.1. The resolver sends a query (1) to an authoritative name server. The name server sends back a large response that exceeds the MTU and is fragmented. The fragments arrive at a fragment blocking firewall, where the first fragment is allowed to pass, but remaining fragments (2) are blocked. The IP stack on the resolver is now unable to reassemble the fragments to get the original response. After a timeout expires (the duration of which may differ per operating system), the IP stack gives up on reassembling the response. To signal to the sender of the response that this problem occurred, the IP stack on the resolver will send an ICMP Fragment Reassembly Time Exceeded (FRTE) message (3) [34, 49, 156]. If this ICMP FRTE message is not blocked by a firewall, the authoritative name server will receive this ICMP FRTE message. The end effect of fragment blocking firewalls is that complete zones, even the DNS root [11], become unreachable for resolvers behind such firewalls and con-

¹Note that fragmentation also causes problems for other UDP-based protocols such as SIP [90] or RADIUS [214].

sequently to every client using these resolvers [149]. In the remainder of this chapter we will refer to such resolvers as *problematic resolvers*.

The obvious approach is to solve this problem at the problematic resolver's side. Educational materials, such as [184], explain this issue in detail. Besides configuring firewalls correctly, a solution could be that problematic resolvers switch to TCP, instead of UDP, in order to avoid fragmentation. TCP, however, is expensive compared to UDP in terms of response time and system resource consumption. Another solution would be to detect problems with fragmentation at the problematic resolver side, for example by inspecting the maximum MTU to a given authoritative name server, and alter future requests to such authoritative name servers to avoid fragmentation [162, 208].

The issue with relying on operators of problematic resolvers to deal with this issue, however, is two-fold. First, it gives very little control to domain owners whose DNSSEC-signed zones may become unreachable for end users behind problematic resolvers. This may hinder wide-scale deployment of DNSSEC, as parallels with IPv6 show, where large players on the Internet, such as, e.g., Google [47], were reluctant to deploy IPv6 because it potentially meant that a small percentage of their users would experience connectivity issues.

Second, resolver operators may be unaware that they suffer from this problem. The majority of DNS resolver implementations request DNSSEC data regardless of whether or not that data is validated and have default settings that inevitably lead to fragmented responses to some of their queries. The two most important open source DNS implementations, BIND² and Unbound³, both send EDNS0 queries by default, and have the `DO`⁴ flag set. Furthermore, their default behaviour is to indicate that the maximum UDP response size they are capable of receiving is 4096 bytes. Thus, even if a resolver that runs these software packages is not configured to validate DNSSEC responses, unless default settings have been changed, they will ask for DNSSEC data and could be sent fragmented responses by authoritative name servers. Similarly, Microsoft DNS, included in Windows Server, has EDNS0 enabled with the `DO` flag set since Windows Server 2008R2, and a default maximum UDP response size set to 4000 bytes.

In this chapter, we focus on answering the following questions:

- To what extent do DNS resolvers experience problems receiving fragmented responses in the real world?
- How can fragmentation of DNS responses be avoided *on the side of the authoritative name server*?

²Since version 9.5.0, released in 2008.

³Since the first release.

⁴DNSSEC OK, see also Section 2.5.2.

Figure 3.2: CDF of DNS response sizes in traces

- How do the solutions we discuss for avoiding fragmentation on authoritative name servers compare in a real operational setting?
- Can we provide guidance to operators of authoritative name servers and DNS software implementors on dealing with fragmentation?

3.2 Extent of the Problem

This section presents real-world observations of problems with fragmented responses. The observations come from network traces recorded on an authoritative name server at SURFnet. At the time of the measurements, in 2012, this server served some 4000 zones, including 300 DNSSEC-signed zones. Over the measurement period it received 500 queries per second on average.

The traces we recorded contain about 8.5 million DNS(SEC) messages and were collected in early 2012 over a period of 6 hours. In these traces we identified 231,391 unique resolvers (based on IP addresses). Approximately 75% of all queries used EDNS0, indicating that the querying resolver is capable of handling responses with EDNS0. The average response size (UDP and IP headers not included) is 840 bytes. Note that this is higher than the limit for traditional DNS messages (512 bytes). The cumulative distribution of the response size is shown in Figure 3.2. About 36% of all responses would be fragmented at the IPv6 minimum MTU⁵. Moreover, our analysis of the traces shows that 57% of all resolvers received a fragmented response at some time during the measurement.

⁵A DNS message size of 1232 bytes requires 1280 bytes to transport at the Ethernet level.

Next, we analysed the recorded traces to search for evidence of problematic resolvers. This led us to identify five behavioural patterns indicative of problematic resolvers. We discuss these patterns in more detail below:

CASE 1 The simplest pattern to detect is receiving ICMP Fragment Reassembly Time Exceeded (FRTE) messages at the authoritative name server, as discussed in Section 3.1. These ICMP messages unequivocally confirm that the resolver experiences problems with fragmented responses. 1.3% of the resolvers seen in the traces exhibit this behaviour. Note that, as mentioned above, the firewall affecting a problematic resolver may also block ICMP messages sent back to an authoritative name server for security reasons, as discussed in [166]. Thus, the figure of 1.3% gives a lower bound on the fraction of resolvers that experience problems with fragmented responses.

CASE 2 Some DNS software will, when not receiving a response to a query that uses EDNS0, retry a query using traditional DNS (most likely under the assumption that the authoritative name server being queried does not support EDNS0). A resolver cannot distinguish not receiving a response because of blocked fragments, from not receiving an answer because EDNS0 is not supported. We therefore expect resolvers may use this strategy if confronted with not receiving a response because of blocked fragments. In the traces, we detected this behaviour in 2.4% of all resolvers.

CASE 3 When using EDNS0, resolvers advertise the maximum response size they support in all queries (see Section 2.5.2). These advertisements are used by authoritative name servers to ensure that responses do not exceed this size limit. Consequently, if the advertised maximum response size exceeds the MTU of the link between the resolver and the authoritative name server, this will lead to fragmentation. Our traces show that 69% of all queries contain a maximum response size advertisement set to the default⁶ value of 4096 bytes. Considering that the MTU is mostly ≤ 1500 bytes, advertising a maximum response size of 4096 bytes likely results in fragmented responses.

Some resolver software dynamically adapts the advertised maximum response size if it fails to receive responses to queries. Thus, if we detect this behaviour, this is indicative of a problematic resolver. Our traces show 3.5% of all resolvers applying this strategy.

⁶That is: the default for the most popular name server software.

Interestingly, our traces also show that 1.8% of all queries used EDNS0 with a maximum response size advertisement of just 512 bytes and that 2% of all queries had a maximum response size advertisement between 1280 and 1472 bytes. This range is likely chosen by resolver operators such that most responses do not get fragmented, since these values are below the MTU of Ethernet⁷.

CASE 4 If an answer does not fit in the maximum response size advertised by a resolver, the authoritative name server will send back a *truncated response*, indicating to the resolver that the response is incomplete. In order to avoid response size restrictions for UDP, DNS then allows the resolver to retry the query using TCP. Some 0.1% of all resolvers, however, use TCP after a UDP response that is not truncated. We have strong suspicions that these are problematic resolvers attempting to avoid fragmentation issues.

CASE 5 Finally, there are resolvers (9.7% of all resolvers) for which we only detect series of retries that always result in responses larger than 512 bytes. Since we only see retries, it becomes difficult to determine which of these resolvers are actually problematic resolvers. We will discuss this problem in more detail in Section 3.3.2.

Note that only the first pattern is a definite indicator that the resolver has problems with fragments. Because this pattern may be affected by the fire-wall that causes the problems for the resolver we have also looked at the four other patterns discussed, which are heuristic indicators for problematic resolvers. Table 3.1 summarises the patterns. Per pattern, the table lists a description and the occurrence of the behaviour as a percentage of the total number of resolvers. Note that these behavioural patterns, and their occurrence, are not mutually exclusive. That is: a resolver may show more than one of these behavioural patterns, and is counted once for each behavioural pattern it exhibits

Weaver et al. [208] state that up to 9% of all Internet hosts may have problems receiving fragmented UDP messages. Our measurements confirm this; traces show that most resolvers receive fragmented responses. We identified five different resolver behaviours that can indicate problems receiving fragmented responses. As much as 10.5% of all resolvers showed one or more of these behaviours. To ensure that the results we found are not biased because of particular characteristics of the authoritative name server on which we collected traces,

⁷78% of all paths between any 2 nodes on the Internet have the Ethernet MTU (1500 bytes) and 96% are ≤ 1500 bytes [208]. The maximum size for DNS responses (without IP and UDP headers) that avoids fragmentation, given an MTU of 1500 bytes, is 1472 bytes for IPv4 and 1452 bytes for IPv6.

Resolver Behavioural Pattern	Occurrence
CASE 1: Sends ICMP Fragment Reassembly Time Exceeded	1.3%
CASE 2: Fallback to ‘classic’ DNS	2.4%
CASE 3: Reduce advertised maximum response size in retries	3.5%
CASE 4: TCP fallback w/o truncated UDP response	<0.1%
CASE 5: Retries for large responses (>512 bytes)	9.7%

Table 3.1: Behavioural patterns of problematic resolvers

we verified our findings using data collected independently. We analysed traces collected on an authoritative name server at the University of Pennsylvania, over a period of 4 hours, also in early 2012. In these traces, we found the same distribution of behavioural patterns indicative of problematic resolvers.

3.3 Solutions

This section presents two potential solutions to reduce or avoid fragmentation on the side of the authoritative name server. We will evaluate the effectiveness of these solutions in Section 3.4.

3.3.1 Avoiding Fragmentation for All Responses

The first, straightforward solution to the problem is to attempt to avoid most response fragmentation in general. This involves changes to authoritative name servers, as shown in Figure 3.3.

As discussed in Section 3.2 (CASE 3), resolvers advertise a maximum response size in EDNS0 queries. Problematic resolvers advertise maximum response sizes that are too high (as they cannot receive fragmented responses). Generally speaking, the response size on an authoritative name server is only limited by the size advertised by querying resolvers and not by configuration settings on the authoritative name server itself.

The solution we propose is to restrict the maximum response size in the configuration of the authoritative name server, such that (most) response fragmentation is avoided. When responding to queries the authoritative name server will then use the minimum of the configured response size and that advertised in the query.

The response size limit for the authoritative name server should be as high as possible, while avoiding most fragmentation. If the limit is chosen too low, this may result in undesirably high levels of truncated responses, resulting in retries over TCP. Considering that the MTU is usually 1500 bytes⁷, the limit should

Figure 3.3: Authoritative name server with limited response size

be below that value. We propose to use a limit of 1232 bytes, based on the minimum MTU for IPv6 of 1280 bytes [53]. This ensures that any response sent over IPv6 by the authoritative name server is not fragmented. The minimum IPv4 MTU is 68 bytes [155], which is too small to be used here, but since 1232 bytes is well below the expected MTU of 1500 bytes we expect this limit to avoid fragmentation for IPv4 as well. Note that other limits are possible and that 1232 bytes may be too small for some zones [161]. Most DNS server implementations allow a limit on the maximum response size to be set with a single configuration parameter⁸.

The response size limit will help most problematic resolvers, even if just one of the authoritative name servers per zone returns responses of limited size. This is because resolvers will try to query all authoritative name servers for a zone in case they do not receive a response [219].

The solution presented in this section avoids most response fragmentation thus helping problematic resolvers. Although this solution is very simple, care must be taken that a proper size limit is chosen.

3.3.2 Selectively Avoiding Fragmentation

The second solution is to avoid response fragmentation by limiting the response size for problematic resolvers only. Figure 3.4 shows the setup of this solution. This solution is based on DNSRM (DNS Router/Modifier), a tool we developed

⁸For an overview of common DNS server implementations, see Appendix A.

Figure 3.4: Authoritative name server with a host-proxy (DNSRM) and an external sensor

specifically for this purpose. DNSRM operates as a host-proxy on an authoritative name server and acts on information supplied by a separate sensor tool that detects problematic resolvers.

Modifying Queries Using DNSRM

The purpose of DNSRM is to allow an authoritative name server to differentiate in response size, depending on the querying resolver. DNSRM achieves this goal by acting as a host-proxy between the local authoritative name server process (e.g., BIND) and the outside world. It is transparent for resolvers in the outside world. DNSRM also forwards a copy of all DNS traffic to a sensor, which is tasked with detecting problematic resolvers and their maximum response size. DNSRM receives updates to a list of IP addresses of detected problematic resolvers from the sensor.

Finally, DNSRM modifies queries from problematic resolvers detected by the sensor, before it passes them on to the local authoritative name server process. DNSRM overwrites the advertised maximum response size in a query. It does so when a query from a problematic resolver has a maximum response size advertisement larger than the maximum response size for that problematic resolver detected by the sensor. DNSRM does not modify responses.

Detecting Problematic Resolvers Using a Sensor

Our sensor detects problematic resolvers based on DNS traffic forwarded by DNSRM. The sensor analyses the traffic per resolver, based on the five problematic resolver behaviours discussed in Section 3.2. The sensor uses a system of thresholds when detecting a problematic resolver, in order to avoid false positives. For instance, network issues causing packet drops may result in resolver behaviour that could incorrectly suggest a problematic resolver. The actual values of the thresholds depend on the network characteristics of the authoritative name server running DNSRM⁹. The sensor works as follows for each problematic resolver behaviour pattern:

- CASE 1 The sensor detects ICMP Fragment Reassembly Time Exceeded messages and marks resolvers sending these messages as problematic.
- CASE 2 A resolver that first attempts a query using EDNS0 and then retries using classic DNS is marked as problematic by the sensor.
- CASE 3 A resolver that reduces its advertised maximum response size in retries, will also be marked as a problematic resolver.
- CASE 4 The sensor detects the use of TCP in retries. As mentioned in Section 3.2 this is normally only done when a truncated UDP response precedes the fallback to TCP [35]. Resolvers that use TCP without a preceding truncated UDP response are marked as problematic.
- CASE 5 There are a number of resolvers that do not show any of the behaviours described above, but that do send sequences of similar queries that appear to be retries. Since we only see these retries, it is difficult to determine if these resolvers are problematic resolvers. Some will be *non-caching resolvers*, that do not store a response and therefore frequently send the same query. Non-caching resolvers are not necessarily problematic resolvers. If the number of retries from a resolver after responses ≤ 512 bytes exceeds a pre-set threshold we consider them to be non-caching. Here, we assume that fragmentation issues do not affect responses ≤ 512 bytes. If we see retries from a resolver that is not considered a non-caching resolver, then it will be marked as problematic if the number of retries after responses > 512 bytes exceeds a pre-set threshold.

The sensor not only detects problematic resolvers but also the maximum response size it supports. This is done by marking the lowest response size

⁹An expected packet loss of 4% was used, based on Wang et al. [207].

for which problematic resolver behaviour still occurs. The maximum response size for a problematic resolver is reduced stepwise, starting at the MTU of the medium (e.g. Ethernet⁷) being used by the authoritative name server running DNSRM. The process continues for as long as problematic resolver behaviour is detected¹⁰ and will help a problematic resolver quicker with a larger step size.

The solution presented in this section avoids response fragmentation, but only for problematic resolvers detected by a sensor. This sensor detects a maximum response size per problematic resolver that avoids response fragmentation. The size of a response from an authoritative name server is reduced by overwriting the advertised maximum response size in queries using DNSRM. In essence, this solution implements response size reduction at the side of the authoritative name server rather than as a fallback strategy in a DNS resolver implementation as suggested by Rikitake et al. [162].

3.4 Evaluation

In this section, we evaluate the two solutions that were presented in Section 3.3.1 and 3.3.2. First, the characteristics of the solutions are compared. Next, the results of real-world testing are discussed.

3.4.1 Comparing Characteristics

The first solution avoids response fragmentation by limiting the size of all responses sent by an authoritative name server by altering a single parameter on the server. The advantage of this approach is that it is trivial to implement. The biggest disadvantage, however, is that it is a blanket approach that affects all resolvers. As a consequence, well-behaved resolvers ($\pm 90\%$ of all resolvers) may suffer performance penalties because they do not receive optimal answers to their queries, or are forced to fall back to TCP to obtain some responses. Additionally, problematic resolvers are not given an incentive to alter their configuration. Thus one could claim that this approach rewards bad behaviour. Finally, this solution does not address all issues of problematic resolvers (e.g. blocking of DNS messages ≥ 512 bytes).

The second solution only avoids response fragmentation for problematic resolvers. Its main advantage is that it is adaptive; it only limits the response size for resolvers that are suspected to be problematic. Also, it limits the response size dynamically. Thus, unlike the first solution, it also helps problematic resolvers that suffer additional constraints on the path between themselves and the authoritative name server. A final advantage of this solution is that it only

¹⁰It will never go below 512 bytes since that is the lower limit for EDNS0.

assists problematic resolvers for a limited amount of time, giving administrators of these hosts a bigger incentive to improve their behaviour. What makes this solution less attractive, especially in a production environment, is its complexity. It requires two additional components, a host-proxy directly in the path to the authoritative name server and a sensor application that is CPU intensive. If the second solution is deployed, it is imperative that the system(s) hosting the DNSRM and the sensor application be properly sized and engineered such that the second solution's deployment is not a bottleneck, i.e., that the effective throughput of the name server is not decreased.

3.4.2 Real-World Testing

Both solutions were tested for 6.5 hours during office hours on the same authoritative name server of SURFnet that was used to measure the extent of the problem (Section 3.2). Network traces were recorded during the tests for later analysis.

Avoiding Fragmentation for All Responses

To test the first solution we reduced the response size from a typical 4096 bytes to 1232 bytes. Traces show that no responses were fragmented at the authoritative name server. Consequently, we saw no ICMP FRTE messages, which indicates that this solution effectively helped the hosts that were previously sending these error messages. We saw a slight increase in the number of truncated UDP responses, but the increase was not statistically significant. It is relevant, nevertheless, to consider this metric in any future research since truncated UDP responses lead to undesirable and expensive TCP fallbacks.

Selectively Avoiding Fragmentation

The second solution was tested with DNSRM on the same authoritative name server and a sensor hosted on a separate machine. Traces show that fragmentation was down about 50% compared to normal operations; note that fragmented responses still occur for resolvers that are not marked as problematic resolvers.

The number of ICMP FRTE messages was reduced to 18% of normal. This number is not zero, as the sensor first needs to detect a problematic resolver. Only after detection will the resolver be helped and will these ICMP messages disappear.

The number of truncated UDP responses almost doubled in this experiment. This suggests that the detected maximum response size for some problematic resolvers may have been too small. Analysis of the detected sizes confirms this. Approximately 18% of all problematic resolvers were assigned a maximum

response size of 512 bytes, likely resulting in truncated UDP responses. Most of these problematic resolvers may actually be non-caching resolvers, that have no problem receiving fragmented responses. Our mechanism for detecting non-caching resolvers may therefore need to be improved.

Our evaluation suggests that the first solution we proposed is preferred in production environments due to its simplicity and effectiveness. If just one of the authoritative name servers for a zone limits its response size to a value that avoids most response fragmentation, then problematic resolvers may already be able to receive responses. Depending on the zone, a proper response size limit needs to be chosen, in order to avoid an unacceptable increase in truncated UDP responses and TCP fallbacks [161]. We believe that our second solution is not without merit, however. Implementing it yielded a useful categorisation of the problems encountered by resolvers in the form of five problem cases. Our tests also showed that despite its complexity the solution performs well in a production environment.

3.5 Towards Operational Advice

The research presented in this chapter was not only published in an academic paper [180], but was also presented in forums frequented by DNS operators and DNS software implementors. Specifically, the work was presented at RIPE meetings [183, 186], in the ICANN DNSSEC workshop [185] and at the USENIX LISA Conference [187]. At each of these meetings there were discussions with operators and with DNS software implementors. These discussions led us to perform additional measurements to gauge the impact of fragmentation and to study to what extent the first solution we presented, configuring authoritative name servers to limit the response size, is effective. In this section we present the results of these additional measurements. Furthermore, we discuss a third potential solution that is a compromise between no changes on the side of the authoritative name server and hard response size reduction. Finally, we discuss operational recommendations resulting from this research.

3.5.1 Additional Measurements on Fragmentation

In Section 3.3.1 we proposed to solve fragmentation by limiting the maximum response size on the authoritative name server with a configuration setting. Furthermore, in Section 3.4, we concluded that this is the simplest solution and that implementing it on a single authoritative name server for a zone will likely solve the problem for most problematic resolvers. In order to quantify if this is indeed the case, and if we can thus recommend this solution to operators, we performed a set of measurements.

Figure 3.5: Measurement setup for measuring the effectiveness of solution 1

We set up a measurement environment (shown in Figure 3.5) to measure the effectiveness of this solution. The left-hand side of the figure shows the authoritative name servers for `surfnet.nl`, a DNSSEC-signed domain operated by SURFnet. In the middle is a firewall, configured to block fragments. The right-hand side of the figure shows an instrumented resolver. We installed three different DNS resolver implementations on this resolver and measured the round-trip times (RTT) of DNS queries sent to the resolver. Using this setup, we performed four experiments discussed below. These experiments were performed under the following conditions:

- each experiment started with a resolver with an empty cache;
- each experiment was repeated 500 times, to establish a good baseline of average behaviour;
- at the start of each experiment we performed a root priming query¹¹;
- the same query was used in each experiment, we verified that the response to this query from the authoritative name servers was always fragmented.

¹¹This ensures that the resolver has up-to-date information on the addresses of the DNS root name servers [119]. If no root priming is performed, a resolver with an empty cache will do this when it receives the first query. Since this could skew the results of our experiment, we include an explicit root priming query as part of our experimental protocol.

Experiment	BIND 9.8.1				Unbound 1.4.18				MS Windows Server 2012			
	Min.	Mean	×*	Max.	Min.	Mean	×*	Max.	Min.	Mean	×*	Max.
#1: normal ops	75ms	323ms		830ms	40ms	193ms		505ms	187ms	303ms		359ms
#2: w. firewall	2,516ms	3,413ms	(10.6×)	4,403ms	421ms	568ms	(2.9×)	904ms	11,747ms	14,882ms	(49.1×)	20,747ms
#3: 1 auth. rlimit	89ms	1,068ms	(3.3×)	2,154ms	47ms	444ms	(2.3×)	855ms	199ms	5,660ms	(18.7×)	19,577ms
#4: 2 auth. rlimit	90ms	544ms	(1.7×)	1,321ms	43ms	307ms	(1.6×)	852ms	187ms	1,896ms	(6.3×)	4,742ms

*Number of times the mean RTT is slower than without a fragment-blocking firewall.

Table 3.2: Measurement results (95% interval)

The four experiments we performed with this measurement setup and under these conditions then were that we:

1. measured query RTT with the fragment-blocking firewall disabled;
2. measured query RTT with the fragment-blocking firewall enabled, and no modifications on any authoritative name servers;
3. measured query RTT with the fragment-blocking firewall enabled, and one authoritative name server configured to limit the maximum response size;
4. measured query RTT with the fragment-blocking firewall enabled, and two authoritative name servers configured to limit the maximum response size.

We performed each experiment for three different DNS resolver implementations, specifically BIND version 9.8.1, Unbound version 1.4.18 and Microsoft Windows Server 2012 DNS. Table 3.2 shows the results of each experiment for the different implementations we tested. It shows the 95% interval for results and shows the minimum, mean and maximum query RTT in milliseconds. The table also shows the slowdown caused by the resolver failing to receive fragmented responses compared to the mean RTT without fragment blocking.

The results clearly show that failing to receive fragmented responses seriously impacts DNS resolution times. Especially Microsoft DNS is particularly sensitive to this problem. If it takes seconds, or even tens of seconds to resolve DNS names, this can have disastrous consequences for, e.g., web page load times. Users incurring such significant page load penalties will likely abandon their visit of the problematic page. The results also show a significant improvement when changing settings on one or more authoritative name servers for a domain, to prevent fragmentation. If just 40% (two out of five) of the authoritative name servers for our test domain restrict response sizes to avoid fragmentation, then this reduces the slowdown to reasonable levels. Nevertheless, any slowdown of DNS resolution should be considered problematic, as, for example, DNS resolution constitutes a significant proportion of web page load times (up to 13% according to [206]).

Max. UDP size	% of responses truncated for		
	A query for 'www'	AAAA query for 'www'	DNSKEY query
4096 bytes	0.0%	0.0%	0.0%
1472 bytes	1.8%	1.8%	8.1%
1232 bytes	2.9%	3.5%	40.0%

Table 3.3: Truncated responses when restricting the maximum response size

Does this then mean that we would recommend restricting response sizes on all authoritative name servers for DNSSEC-signed domains? Not necessarily; as we discussed before, restricting response sizes may lead to an increase in truncated responses and consequently TCP fallback. To gauge the impact of restricting response sizes, we measured the number of truncated responses for 995 domains (from .com, .edu, .mil, .net and .nl) listed in the SecSpider DNSSEC measurement system [147] in mid-2012. Table 3.3 shows the results. The table lists the percentage of domains for which we receive a truncated response for the specified queries. The first column shows the maximum response size in bytes, set in the query. The remaining columns show the percentage of truncated responses per query.

As Table 3.3 illustrates, restricting response sizes may lead to a significant increase in truncated responses. Especially the DNSKEY query type appears to be particularly problematic. We find that 40% of tested domains return a truncated response to the DNSKEY query if the maximum UDP response size specified in the query is set to 1232 bytes (this aligns with results found by others, e.g., Rikita et al. [162]). As we will discuss in much more detail in Chapter 5, this makes it important to search for ways to reduce not only fragmentation, but also DNSSEC response sizes.

3.5.2 Minimal Responses: a Compromise

There is one other way in which response sizes can be reduced. Recall from Chapter 2 that DNS responses consist of four sections: the question, answer, authority and additional section. Most authoritative DNS server implementations will include records in the authority and additional sections, even if including such records is optional. For example, consider the response from one of the authoritative name servers for `example.com` to an A query for `www.example.com`, shown in Example 3.1. The information in the authority and additional sections of this response is optional. That is: the resolver does not need this information to answer the query, and the response would still be valid if only the data in the answer section is present.

This means that there is an opportunity to reduce response sizes by leaving out (part of) the optional information in the authority and additional sections.

```

;; QUESTION SECTION
www.example.com.                IN      A

;; ANSWER SECTION
www.example.com.                86400   IN      A      93.184.216.34
www.example.com.                86400   IN      RRSIG  A 8 3 86400 ...

;; AUTHORITY SECTION
example.com.                    86400   IN      NS      a.iana-servers.net.
example.com.                    86400   IN      NS      b.iana-servers.net.
example.com.                    86400   IN      RRSIG  NS 8 3 86400 ...

;; AUTHORITY SECTION
a.iana-servers.net.            1800    IN      A      199.43.135.53
b.iana-servers.net.            1800    IN      A      199.43.133.53
a.iana-servers.net.            1800    IN      AAAA   2001:500:8f::53
b.iana-servers.net.            1800    IN      AAAA   2001:500:8d::53

```

Example 3.1: Response to A query for `www.example.com`

If all optional records were removed from Example 3.1, this would reduce the size of the response by more than half. Leaving out optional information in the authority and additional sections is commonly referred to as sending ‘minimal responses’. Most authoritative DNS server implementations have configuration options to enable some form of minimal responses, although the interpretation of what constitutes a minimal response differs¹². Now, we note that the reason that removing the optional records from Example 3.1 reduces the size of the response by more than half, is mainly because it removes a signature from the response. As discussed in Chapter 2, signatures are large, and the main reason DNSSEC responses are much larger than ordinary DNS responses.

In many cases, if the authority and additional sections in responses for a DNSSEC-signed zone are filled, these will contain one or more signatures. What this means is that configuring an authoritative name server to send minimal responses can also help reduce the response size, to prevent fragmentation. To verify if this can be an alternative to strictly reducing the response size on authoritative name servers, we reconfigured an authoritative name server at SURFnet and measured the effect. Figure 3.6 shows the result. The average response size drops from 840 bytes to less than 200 bytes, a reduction by a factor of more than four. Furthermore, the number of fragmented responses drops steeply, to around 0.5% of DNSSEC-signed responses.

¹²For example, BIND will completely remove all optional records in a response if configured for minimal responses. NSD, on the other hand, will leave out optional records until the response fits in a certain pre-defined size (see <https://www.nlnetlabs.nl/projects/nsd/>).

Figure 3.6: Average response size for `ns3.surfnet.nl`, late 2012 - early 2013

3.5.3 Operational Recommendations

Based on the results presented in this section, we formulated operational advice that was communicated in the aforementioned forums frequented by operators and DNS software implementors [183, 185, 187]. Summarised, this advice is to enable minimal responses, if the authoritative name server implementation used to operate a DNSSEC-signed zone supports this. If the number of fragmented responses is still high after that change, restrict the maximum UDP response size on the authoritative name server to below the MTU. Setting it to 1472 bytes is a good compromise, if the server also has IPv6 connectivity 1232 bytes should also be considered, at the risk of an increase in TCP fallback, especially for DNSSEC-specific queries such as the `DNSKEY` query (as shown in Table 3.3). In all cases, operators of DNSSEC-signed zones must ensure that their authoritative name servers accept queries over TCP.

3.6 Concluding Remarks

The introduction of DNSSEC creates new problems, such as Internet zones becoming unreachable due to poorly configured firewalls that block fragmented UDP packets. Although these problems can easily be solved by changing the firewall rules or by changing the behaviour of DNS resolver software, our research shows that DNS zone operators cannot rely on these solutions always being implemented in practice. To avoid becoming unreachable for clients behind problematic resolvers, DNS zone operators have to take measures to prevent DNS responses getting fragmented and blocked by firewalls.

We measured the extent of this problem using traces with real-world DNS data from authoritative name servers at a major NREN and at large university in the US. Analysis of these traces shows that 58% of all resolvers received a fragmented response at some point in time. To identify which resolvers exper-

ience problems, we identified five ways problematic resolvers can behave. We found that 10.5% of all resolvers showed problematic behaviour.

After measuring the extent of the problem, we presented two solutions. The first solution is to limit the size of all responses by an authoritative name server. This can usually be accomplished with simple configuration changes. The second solution is more sophisticated and requires special software that alters DNS queries, but only for detected problematic resolvers. These altered queries result in smaller responses and avoid response fragmentation for problematic resolvers.

These two solutions were then tested in a real-world setup on authoritative name servers operated by a major NREN. We concluded that the benefits of the second solution, where only detected problematic resolvers are assisted, do not outweigh the simplicity and results of the first solution.

After presenting this work to operators and DNS software implementors and discussing the results with them, we performed additional measurements to refine the effectiveness of the first solution. This led to solid operational recommendations. Furthermore, discussions after the presentation at RIPE 65 [183] influenced the revised specification for EDNS0 [52]. Our research also influenced DNS resolver implementors to improve fallback behaviour for EDNS0 queries in the case of connectivity problems.

Nevertheless, even if the operational recommendations we provide are followed, fragmentation remains a problem. We will revisit this problem in Chapter 5 when we study an alternative path to reduce fragmentation: reducing DNSSEC message sizes through the use of different cryptographic signing algorithms.

Acknowledgments

We would like to express our gratitude to Shumon Huque who helped provide data from the University of Pennsylvania to validate our research.

DNSSEC and Amplification Attacks

A common criticism of DNSSEC is that it can make DNS amplification attacks much worse. Since the effectiveness of this type of attack relies on the response message size, it is easy to see, based on the previous chapters, how DNSSEC can indeed increase the potential for such attacks. Yet despite the fact that DNSSEC started getting deployed at scale after the Kaminsky attack was published in 2008, there was surprisingly little data on DNSSEC's impact on amplification. In this chapter we present a large-scale measurement study that establishes ground truth around this question. We study the potential for amplification in 70% of DNSSEC-signed domains in operation in 2014. The study presented in this chapter was published earlier as [193].

4.1 Introduction

Since they were first seen at scale at the turn of the century [73], Distributed Denial of Service (DDoS) attacks have become one of the biggest threats to the Internet's security and stability. The scale of DDoS attacks keeps growing. What was then termed 'the biggest DDoS ever' of 300 Gbit/s in early 2013 [129] has already been surpassed again and again. In 2016, the first attack passing the 1 Tbit/s mark was reported [79].

These large volumetric attacks usually rely on the same basic principles. First, they use *spoofing*. Spoofing allows the attacker to falsify the source IP in a request to a network service, resulting in the response to this request being sent to the falsified source IP (this is known as *reflection*). Second, attackers leverage *amplification*, the principle that some network protocols (such as DNS and NTP) return a large answer to a relatively small request. Amplification is of particular interest to an attacker since a small investment in attack traffic results in large attack volumes.

A commonly used DDoS attack is *DNS amplification* (as e.g. Arbor Networks' annual security report shows [14]). As the name suggests, this attack relies on bandwidth amplification using the DNS protocol, where amplification is defined as $\frac{\text{response size}}{\text{query size}}$. Typical DNS requests are in the order of magnitude of 20-60 bytes in size. The classic DNS protocol [141] limits responses to at most 512 bytes; assuming a request size of 40 bytes this already yields an amplification factor of $\frac{512}{40} \approx 12.8\times$. And as we will see further on, DNS messages that use EDNS0 easily result in amplification factors of $100\times$ or more.

As we already discussed in Chapter 2 and Chapter 3, DNSSEC responses are typically larger than ordinary DNS messages. Many experts consider this a major drawback of DNSSEC (e.g. Cowperthwaite & Somayaji [50]) and noted opponents of DNSSEC cite this as one of the reasons why they feel DNSSEC should not be used (e.g. Bernstein [31]).

Although the DDoS potential keeps coming up time and again in discussions about DNSSEC, very little ground truth existed about the actual DDoS potential of the millions of DNSSEC-signed domains that are in operation. Using data from six major top-level domains, we performed a comprehensive measurement study covering 70% of all DNSSEC-signed domains in operation in mid-2014. We compare the DDoS potential of DNSSEC-signed domains to a representative sample of domains without DNSSEC. Using this data, we address the main question covered in this chapter: *How bad is DNSSEC really?*

4.2 Background on DNS(SEC) Amplification

4.2.1 DNS Amplification: a Brief Primer

Attacks Using Open Resolvers

Figure 4.1 shows how DNS amplification attacks work. Attacks are initiated from a swarm of machines (left-hand side of the figure) under the control of the attacker. The attacker uses this swarm to send large numbers of DNS queries in which the sender IP address is spoofed to be the victim's IP address (bottom middle of the figure). Queries are sent (1) to so-called *open DNS resolvers*.

Figure 4.1: DNS amplification attack

These are misconfigured DNS resolvers that do not restrict which clients may send them queries. In turn, the open resolvers will – if the query result is not cached – contact authoritative DNS servers (2) to resolve the query. Finally, the open resolvers send the responses (3) to the victim. In general, queries (1) are small whereas responses (3) are large, hence achieving amplification.

Unfortunately, open resolvers are plentiful on the Internet. Kühner et al. [125] report observing between 23 and 25.5 million open resolvers during weekly Internet-wide scans performed between November 2013 and February 2014. This makes this attack easy to carry out and thus attractive for attackers.

EDNS0 and Amplification

Typically, the goal of an attacker is to achieve a high amplification factor. This gives them the best return on investment where they only need to generate a small amount of traffic for a large attack. As discussed in Section 2.5.2, EDNS0 makes DNS responses larger than the original 512 bytes possible. Figure 4.2a shows whether clients to one of SURFnet’s authoritative name servers use EDNS0. As the figure shows, about two thirds of clients use EDNS0; earlier research (e.g. [124]) shows that this figure is similar for other name servers across the Internet. The prevalent configuration for EDNS0 is to set the maximum response size to approximately 4 Kbytes. Figure 4.2b shows the maximum response size reported by clients of the same name server as in Figure 4.2a, with around 90% advertising a size over 4000 bytes. Again, this matches earlier results reported by Kreibich et al. [124].

Figure 4.2: EDNS0 measurements on `ns1.surfnet.nl`

Query size	Response size	Amplification	Attacker	Victim
40 bytes	512 bytes	12.8×	100 Mbit/s	1.28 Gbit/s
40 bytes	1472 bytes	36.8×	100 Mbit/s	3.68 Gbit/s
40 bytes	4096 bytes	102.4×	100 Mbit/s	10.24 Gbit/s

Table 4.1: Theoretical effect of DNS amplification

In theory, given a query size of 40 bytes, an amplification of $\frac{4096}{40} \approx 102.4\times$ is achievable. Table 4.1 gives some examples of the theoretical attack volumes that can be achieved if the attacker generates 100 Mbit/s in queries for various amplification factors. As the table shows, when using EDNS0’s expanded maximum response size, significant attack results can be achieved with a relatively small investment in bandwidth on the side of the attacker.

Crafted domains

Also of interest is how attackers select the DNS query to use in an attack. One current practice is that attackers craft special domains for which certain DNS queries are guaranteed to return large responses. Vaughn & Evron [199], for instance, analyse a case where an attacker crafted a domain with a large `TXT` record that guaranteed a large response to certain queries. A particularly interesting query type for attackers to abuse is the `ANY` query; when faced with such a query, name servers are supposed to return all record types for the name specified in the query. In practice, a combination of both a crafted domain and the use of `ANY` queries is often observed. An investigation of so-called ‘booters’ that provide ‘DDoS-for-hire’ services [164], observed a number of DNS amplification attacks, most of which used EDNS0, `ANY` queries and crafted domains.

4.2.2 Benefits DNSSEC offers to the attacker

There are a number of ‘hindrances’ for attackers given the attack methodology described in the previous subsections:

- bandwidth on the open resolvers that are abused for the attack may be scarce (for instance because the open resolver is on a DSL CPE¹), necessitating the use of large numbers of amplifiers;
- creating crafted domains makes the attacker vulnerable to prosecution since domain registrations can – sometimes – be traced back to the attacker;
- crafted domains may be taken down when discovered;
- filtering of responses for crafted domains can be a possible mitigation strategy for victims.

DNSSEC-signed domains offer an attractive alternative to attackers since they help avoid the hindrances described above. Given the high number of signed domains, it gets more interesting to directly abuse authoritative name servers for DNSSEC-signed domains, instead of open resolvers. It is more likely that authoritative name servers have sufficient bandwidth available than, e.g., open resolvers on DSL CPE devices. Also, instead of crafting a dedicated domain, attackers can now choose from a wide range of DNSSEC-signed domains for which the answers to, e.g., ANY queries are most likely quite large due to the inclusion of DNSSEC key material and signatures. This makes attackers less vulnerable to discovery, prosecution and take-down of the crafted domains. Finally, DNS responses for legitimate DNSSEC-signed domains are much harder to filter since filtering is likely to also impact legitimate queries and responses.

These benefits to attackers underline the necessity of establishing ground truth about the actual DDoS potential of the already large and growing number of DNSSEC-signed domains. With this data in hand, it becomes possible to develop targeted strategies to address this potentially dangerous side effect of deploying DNSSEC.

There have already been attacks that abuse DNSSEC-signed domains for amplification. In mid-2012 many organisations across the Internet saw their DNSSEC-signed domains being abused directly on authoritative name servers for amplification attacks. Documentation about these attacks is, unfortunately, not available as they were mainly discussed in private by large operators, on mailing lists and at meetings such as RIPE and the IETF. The author of this

¹Customer Premise Equipment (CPE) used on a residential Internet connection; colloquially known as a ‘home router’.

thesis, however, was involved in many of these discussions. An indirect discussion of these attacks was written up by Cisco in their security blog².

We note that the DNS and DNSSEC community (consisting of network engineers, DNS software developers and organisations concerned with online security) has already started to take steps to mitigate the dangers of DNSSEC-based amplification attacks by introducing Response Rate Limiting (RRL), which is discussed in Section 4.6.

4.3 Methodology

4.3.1 High-level goals

Our high-level goal, as stated in Section 4.1, is to gauge the DDoS potential of DNSSEC-signed domains in comparison to unsigned domains. In order to achieve this goal, we perform a large scale measurement of all DNSSEC-signed domains in six top-level domains (TLDs) that have large numbers of signed domains. We compare these against a representative sample of unsigned domains in the same TLDs.

4.3.2 Acceptable upper limit for amplification

An important question to ask – but one that is difficult to answer – is what constitutes an acceptable upper limit on the amplification possible for a particular domain on a particular server. The simple answer is that any amplification is bad, but such an answer is too simple, since the DNS protocol already inherently has some amplification effect.

As we outlined in the previous section, EDNS0 vastly increases the amplification potential in DNS. Since – other than combating open resolvers – no large scale action has been taken to reduce the amplification potential in pre-EDNS0 classic DNS, we take the maximum amplification achievable using classic DNS as an upper limit. We calculate³ this maximum achievable amplification by 1) assuming a query for the shortest name in a domain (e.g., ‘x.com’), which will consequently result in the smallest query and 2) a response that uses the maximum allowed size for regular DNS (512 bytes). For our example (x.com) this yields an amplification of $\frac{\text{response size}}{\text{query size}} = \frac{512}{23} \approx 22.3\times$.

²Case study 2 in <https://blogs.cisco.com/security/real-world-dns-abuse-finding-common-ground/>

³Note that this is not just a theoretical value; analysis of our measurements shows that each measured TLD has a non-negligible number of domains that meet this pattern (small query, maximum size response) when queried using classic DNS.

4.3.3 Query types

We selected a number of queries to perform for each domain. These queries are specified below, together with a rationale for why we selected this particular query type:

- **ANY** – as explained in Section 4.2.1 this query results in the largest possible response since it includes all resource records for the queried name.
- **MX** – returns the names of mail exchangers for the domain; domains usually have more than one mail exchanger hence the answer to this query may be relatively large.
- **NS** – returns all authoritative name servers for a domain; again, domains usually have more than one authoritative name server, thus the answer to this query may also be relatively large.
- **A** – returns the IPv4 address(es) for the queried name and is typically the most common DNS query performed for a domain.
- **AAAA** – returns the IPv6 address(es) for the queried name and is another common query type since most modern software will look for both the IPv4 and the IPv6 address of a host.
- **TXT** – returns textual information for the queried name; this can, for instance, be information about mail handling for the domain (used by spam filters). Additionally, as mentioned in Section 4.2.1 this query type is sometimes used by attackers in domains especially crafted for amplification attacks.

For the latter three query types (**A**, **AAAA** and **TXT**) we perform multiple queries. One for the so-called apex record (denoted by **@**), one for the **'www'** label under the domain and one for the **'mail'** label. We expect that at least one of these names exists in most domains.

For DNSSEC-signed domains we also measure the answer to two DNSSEC-specific query types:

- **DNSKEY** – returns the set of public keys required to validate signatures in a domain. The answer is usually large as there is often more than one key and individual key records are large as RSA keys of sizes 1024- and 2048-bits are commonly used (as suggested in [121] and because these are default values for many DNSSEC implementations).

- **NSEC(3)** – this is not a query type as such, but is the record type that is returned when the queried name does not exist (see Section 2.5.3); especially responses of the newer variant **NSEC3** are likely to be quite large.

Each individual query is performed once using classic DNS and once using EDNS0 with the EDNS0 maximum response size set to 32,768 bytes⁴. For DNSSEC-signed domains we also perform the query with EDNS0 and the DNSSEC OK (DO) flag set to `‘true’`, to get DNSSEC-signed responses.

4.3.4 Metrics

For each domain we measure, we record the set of metrics listed below. These metrics not only allow us to calculate the amplification factor of a DNS response, but also allow us to analyse specific factors that contribute to the response size, such as the presence or absence of optional sections in the DNS response.

- the query size (DNS UDP datagram size);
- the response size (DNS UDP datagram size, possibly reassembled from multiple UDP fragments);
- the amplification factor (defined as $\frac{\text{response size}}{\text{query size}}$);
- the EDNS0 maximum response size reported by the authoritative name server (puts an upper bound on the maximum amplification that can be achieved using this particular name server);
- whether or not the response was truncated (this indicates that the authoritative name server was unable or unwilling to return all requested data in the response), and indicates that the querying host should fall back to TCP to get the full response;
- the number of answers in the response;
- the number of authority records in the response (in most responses including this information is optional and indicates which name servers are authoritative for the queried name);
- the number of additional records in the response (these optional records, e.g., specify IP addresses for name servers in the authority section);
- the number of distinct resource record types in the response.

⁴We chose this value to also register results that exceed the commonly used maximum response size of 4KB; we decided not to use the maximum value (65,535 bytes) since we did not want to risk running into possible boundary conditions in DNS software implementations.

To ensure that we are not inadvertently including DNSSEC-signed domains in our control dataset of unsigned domains, and vice versa, we check for each domain whether it is DNSSEC-signed or not (by checking the presence of `DNSKEY` records). If the domain turns out to belong to another category than expected, we exclude this domain from the measurement. Equally, those domains for which we are unable to determine the set of authoritative name servers or for which none of the authoritative name servers responds to queries are excluded from the datasets. We note that, for reasons of efficiency, we attempt each query only once, since authoritative name servers are generally always online.

4.3.5 Measurement software

To perform the measurements we developed two applications. The first is a zone file parser, which, given the size of the DNS zones for TLDs, we needed specially tailored software for. The parser application is able to extract DNSSEC secure delegations, i.e., DNS delegations for which a DNSSEC chain of trust is established by including one or more `DS` records in the TLD (see Section 2.5.3). It is also able to extract regular domains from the zone. The parser stores the extracted domain names in a SQLite database. Optionally, the application can take a random sample of domains.

The second application is a scanner application. This application operates on the database created by the zone file parser and for each domain in the database will perform the queries outlined in Section 4.3.3. The application is massively parallel and launches several hundreds of threads to perform the scan. Scanning takes place in two phases. In phase one, the application determines the set of authoritative name servers for each domain in the database and the corresponding IPv4 and IPv6 addresses for these name servers. In phase two, the application sends the queries described in Section 4.3.3 to each individual IP address for each name server of each domain. After each set of queries to a single IP address, the domain is placed back at the end of the queue until no more IP addresses remain for the domain to be scanned in which case it is marked as completed. This design choice ensures that no excessive amounts of queries for a single domain are sent in bursts while keeping the amount of state the scanner needs to maintain manageable. We found that this strategy strikes an optimal balance between the load imposed on the scanning system as well as the scanned systems.

Both applications make extensive use of the LDNS library by NLnet Labs⁵, a popular software package that provides a comprehensive set of DNS-related functionality.

⁵<https://www.nlnetlabs.nl/projects/ldns/>

TLD	Data Obtained	#Domains	#DNSSEC
.com	Full zone	113.1M	326.5k (0.3%)
.net	Full zone	15.2M	69.5k (0.5%)
.org	Full zone	10.3M	37.6k (0.4%)
.nl	Selection	5.4M	1696.1k (31.2%)
.se	Full zone	1.4M	334.9k (24.8%)
.uk	Selection	10.6M	10.2k (0.1%)

Table 4.2: Source data (retrieved April 2014)

Note that the SQLite database schema was designed such that the data can easily be anonymised. This was done on purpose because the resulting datasets are shared as open data with the research community.

4.3.6 Ethical considerations

As mentioned in the previous section, we took particular care to ensure that our measurements do not impose an undue burden on authoritative name servers we scan. To this end we made deliberate design choices as discussed in Section 4.3.5. In addition to this, we monitored traffic from our experiment while it was running, to check if the design worked as foreseen. Additionally, we only performed legitimate queries that are expected to be part of day-to-day traffic to authoritative name servers.

With respect to the datasets for the TLDs we used, we obtained these through specific processes established by the TLD operator or under a specific contract with individual TLDs. In both cases, we took care to inform the TLDs about the purpose for which we were going to use the data and obtained their consent.

4.4 Datasets

4.4.1 Source Data

We obtained data covering 70% of all DNSSEC-signed domains. Data came from six different top-level domains. This data was retrieved in April 2014. Table 4.2 lists the TLDs from which we obtained data. The table lists the type of data obtained (either the full zone or a selection containing all secure delegations and a random sample of unsigned domains), the total number of domains in the TLD and the number of secure delegations (as an absolute value and a percentage).

TLD	#Domains	#Failed	#Skipped	#Queried	#Queries	#Auth NS
.com	326,504	7,416	471	318,576	54.6M	2,550
.net	69,552	2,672	55	66,814	11.0M	2,476
.org	37,621	555	19	37,024	6.7M	2,073
.nl	1,696,103	12,304	1,002	1,682,770	233.3M	1,316
.se	334,880	8,696	100	326,067	43.3M	3,681
.uk	10,225	314	10	9,894	1.6M	570

Table 4.3: Overview of DNSSEC datasets

TLD	#Domains	#Failed	#Skipped	#Queried	#Queries	#Auth NS
.com	498,502	55,909	2,231	436,593	37.6M	72,168
.net	99,564	13,904	355	84,882	7.4M	26,396
.org	100,000	11,031	277	88,372	7.5M	27,761
.nl	1,000,000	69,092	6,812	921,441	69.3M	31,108
.se	499,999	37,361	149,560	311,871	21.5M	23,756
.uk	26,131	3,883	92	21,858	1.6M	7,091

Table 4.4: Overview of non-DNSSEC datasets

4.4.2 Collected Data

We ran two scans for each TLD in the source dataset. The first scan covered all DNSSEC-signed domains in the TLD. The second scan examined a representative uniformly random sample of unsigned domains with a size in the same order of magnitude as the number of DNSSEC-signed domains in the TLD.

For each scan type Table 4.3 and Table 4.4 show the total number of domains for which queries were attempted, the number of domains for which we failed to obtain the list of authoritative name servers, the number of domains that were skipped (because they were DNSSEC-signed whereas they were expected not to be or vice versa), the actual number of domains that were successfully queried, the total number of queries included in the dataset and the number of distinct authoritative name servers observed during the scan. We note that there may be a slight difference between the total number of domains for which queries were attempted (column 2) and the number of failed, skipped and successfully queried domains added up (columns 3, 4 and 5). The reason for this is that for a small number of domains, although we were able to determine the set of authoritative name servers, none of these responded to queries. Since for both the regular as well as the DNSSEC datasets this difference is very small (0.52% and 0.03% on average over all TLDs respectively) it is not shown in the table.

4.4.3 General Observations

If we look at the result datasets in Table 4.3 and Table 4.4 three observations stand out.

1. The fraction of domains for which the set of authoritative name servers could not be determined is significantly larger for unsigned domains (on average 10.9% for unsigned domains versus 2.3% for DNSSEC-signed domains). This seems to imply that DNSSEC-signed domains are generally better configured and are more likely to have functioning delegations.
2. The ratio of distinct authoritative name servers versus the number of domains in the dataset is much higher for unsigned domains (by more than a factor of ten). This is probably due to the fact that a number of large DNS operators enabled DNSSEC for all domains they manage stimulated by TLDs such as `.nl`, `.se` and `.org` offering financial incentives for deploying DNSSEC.
3. A very high number of domains in the `.se` TLD were skipped in the non-DNSSEC dataset because they turned out to be DNSSEC-signed after all; a random sampling of the skipped domains shows that this is very likely due to one large DNS operator having enabled DNSSEC without creating secure delegations in the parent TLD.

4.5 Analysis

In this section, we provide a detailed analysis of our measurements and examine the differences between regular and DNSSEC-signed domains. Rather than looking at the exact response size in bytes for different queries, we use the amplification factor as main metric. We do this because it allows for an unbiased comparison between queries for different domains independent of the length of the domain name. To illustrate this with an example, the shortest scanned domain name in `.com` is one character long; the EDNS0 query size for this domain is 34 bytes, the response size for an `ANY` query with DNSSEC enabled is 3549 bytes. This gives an amplification factor of $104.4\times$, which is very high as we will see later. The EDNS0 query size for the longest name (63 characters) scanned in `.com` is 96 bytes and the response size is 3805 bytes. That only yields an amplification factor of $39.6\times$ (around average as we will see later). If we had compared these two domains on the basis of the response size then the second domain would have been classified as ‘worse’ than the first domain, whereas in fact it is not. We also note that the amplification factor more accurately reflects the return on investment for attackers. A certain amount of traffic n sent by

an attacker results in an attack that is $n \times$ amplification in size. Note that we calculate the amplification factor using the UDP datagram size for both query and response rather than using the full packet size. This prevents noise from varying packet sizes caused by differences in the underlying network technology.

Another general remark concerns the selection criteria for the graphs we plot. In most of our analyses, we only include responses we consider valid, defined as those responses that include at least one answer in the answer section and that have a response code (RCODE) that indicates success (NOERROR). We do this to exclude as many authenticated denial-of-existence answers as possible from the datasets used to plot the graphs, since these may skew the data in the graphs (as especially the **A** and **AAAA** queries we perform may result in authenticated denial-of-existence as the name we query for may not exist). We cover authenticated denial-of-existence itself in a separate graph, which of course does include responses with no answers and with response codes that indicate that the name does not exist.

In our analysis, we look at three areas of interest. First we examine the main vehicle for DNS amplification attacks, **ANY** queries. Next, we look at query types for which we expect large answers. We finish by examining the bread and butter of the DNS, address queries.

4.5.1 ANY Amplification Increase

Since **ANY** queries are the most interesting for attackers, we start our analysis by looking at these in more detail. To compare the amplification factor achievable

Figure 4.3: Amplification of **ANY** queries for **.net**

Figure 4.4: Amplification of ANY queries (all TLDs)

with an ANY query for regular domains and for DNSSEC-signed domains we plot this factor for both types of domains in a single graph. To do this, we distribute the data across bins of size 0.1; plotting the actual data results in a graph as shown in Figure 4.3, which shows the ANY amplification factor for the .net TLD. As shown, graphing the original data (thin red and blue lines) leads to a jagged graph. This is due to the fact that the amplification factor is derived from discrete values (query and response size) resulting in a bias for certain bins. To mitigate this, we represent the data using a Bézier curve that follows the average fill of the bins (thicker cyan and green lines).

The graph in Figure 4.4, plotted as described above, compares the ANY amplification factor for all TLDs for which we performed measurements. Regular domains are represented by filled grey curves whereas DNSSEC-signed domains are represented using coloured lines. This approach to plotting is repeated in all other graphs that compare amplification of regular versus DNSSEC-signed domains. The vertical black line (indicated with an arrow) represents the theoretical maximum achievable amplification factor using regular DNS, which we set as an acceptable upper limit to amplification in Section 4.3.2.

Looking at Figure 4.4 we see that, as expected, the amplification factor for DNSSEC-signed domains is significantly higher than for unsigned domains. On average, the amplification for unsigned domains is around $5.9\times$ whereas for DNSSEC-signed domains the average lies around $47.2\times$ (about eight times higher). We also note that – apart from outliers not visible in the graph – the amplification factor for ANY queries for DNSSEC-signed domains always exceeds the acceptable upper limit we set.

Figure 4.5: Examining the two amplification peaks for DNSSEC-signed domains

Next, the distribution of observed amplification factors is much more spread out for DNSSEC-signed domains. This is as expected; a DNSSEC-signed answer includes a separate signature (**RRSIG** record) for each resource record set. Thus, the more different resource record types in an answer, the more signatures. Signature records are large (over 150 bytes for a 1024-bit signature) and contribute significantly to the amplification factor.

Finally we note that the distribution of amplification factors for DNSSEC-signed domains shows two peaks for each TLD, one around $40\times$ and a lower peak around $55\times$. We found two effects that play a role in this.

The first effect occurs for the majority of the TLDs we measured (**.com**, **.net**, **.org**, **.nl** and **.uk**). For these TLDs, the two peaks can be explained by looking at the number of **DNSKEY** records in the zone. The first peak coincides with having two **DNSKEY** records in the zone, the second peak coincides with having three or more **DNSKEY** records in the zone. Figure 4.5a shows this correlation for the **.nl** TLD (the blue line in Figure 4.5a corresponds to the data for **.nl** as graphed in Figure 4.4 in orange). Note that we do not have direct data for the number of **DNSKEY** records per scanned domain. Rather, we derive this from the number of answers in the **DNSKEY** query for a domain. If this number is four or below (red line in Figure 4.5a), then there are two or fewer **DNSKEY**s present (four answers means two **DNSKEY** records and two signatures in the answer, one signature with each key). If this number is higher than four (blue line in Figure 4.5a) then there are three or more **DNSKEY**s present. The reason for having different numbers of **DNSKEY** records in a domain is simple: different software implementations that use different key rollover strategies (a detailed

TLD	With Amplification		Maximum Amplification	
	Over 100×	(%)	With DNSSEC	w/o DNSSEC
.com	144	(0.05%)	119.2×	75.0×
.net	168	(0.25%)	178.6×	51.0×
.org	139	(0.38%)	143.1×	33.6×
.nl	145	(0.01%)	131.0×	80.9×
.se	211	(0.06%)	120.0×	63.7×
.uk	26	(0.26%)	148.6×	26.7×

Table 4.5: Outliers per TLD

discussion of which is outside of the scope of this thesis, for more information on DNSSEC key rollovers see e.g. [121]).

The second effect occurs for one TLD (.se) only. Interestingly, in this case the two peaks show no relation to the number of DNSKEY records. Rather, the peaks are related to whether or not the additional and authority sections in a DNS answer are filled. These sections are optional in most DNS responses (see Section 2.3.2). Most DNS software implementations have a configuration option that allows administrators to disable filling of these optional sections of a DNS answer. Disabling these optional responses can have a dramatic effect on the response size, as illustrated by Figure 3.6 in Section 3.5.2. Especially for DNSSEC-signed domains, the difference in size can be dramatic since signatures will be included in these optional sections. In Figure 4.5b we graphed the amplification for DNS responses that did not include the optional authority and additional sections (red line), the amplification for responses that did include optional sections (blue line) and the original data for ANY responses in .se (blue filled curve) corresponding to the brown line in Figure 4.4.

Outliers

Figure 4.4 shows the distribution of amplification factors for the majority of domains measured. There are, however, outliers that have much higher amplification factors. Table 4.5 gives an overview of the outliers. Per TLD, it shows the number of domains with an amplification factor higher than 100× as well as the absolute maximum amplification factor of any domain measured.

The table shows some staggering outliers, especially keeping in mind that an amplification factor of 100× means an attacker can mount an attack of 1 Gbit/s by sending only 10 Mbit/s. For comparison, the rightmost column of Table 4.5 shows the maximum amplification factor for the unsigned domains we measured. These are clearly a lot lower, with none exceeding 80.9×

Figure 4.6: Amplification factor for DNSSEC-specific query types

4.5.2 Other Large Queries (DNSKEY, NSEC3, MX, NS, TXT)

First, we analyse two DNSSEC-specific query types, the **DNSKEY** query and authenticated denial-of-existence. As explained in Section 4.3.4, the answer to a **DNSKEY** query for a domain is likely to be large. Figure 4.6a shows the amplification factor for **DNSKEY** queries for all TLDs we measured. On average 37.8% of **DNSKEY** queries yield an amplification factor that exceeds the acceptable upper limit we set in Section 4.3.2. We observe that most TLDs have two peaks in the graph; this is again caused by varying numbers of **DNSKEY** records per domain as described in the analysis of **ANY** queries in Section 4.5.1.

The other query type likely to result in large responses when DNSSEC is used, is a query that has no answer. Just like in regular DNS, a special response is returned when the name and/or record type queried for does not exist. The difference is that in DNSSEC a proof is included that the name and/or record type does not exist. This is called authenticated denial-of-existence and relies on two special record types, **NSEC** and **NSEC3**. These record types are two protocol variants that achieve the same goal. The difference between the two is that **NSEC** uses names in proofs whereas **NSEC3** uses hash values in proofs (for a detailed discussion of authenticated denial-of-existence, see Section 2.5.3). Depending on the protocol and the type of proof required, a number of **NSEC** or **NSEC3** records are included in an authenticated denial-of-existence proof, each accompanied by a signature. Figure 4.6b shows the amplification factor for **NSEC(3)** responses for all TLDs measured. As the graph shows, except for some outliers, the majority of responses fall within the acceptable upper limit we set. The graph contains multiple peaks for each TLD. Analysis of the data shows that the peaks are related to the number of **NSEC(3)** records in a response.

Figure 4.7: Amplification of non-DNSSEC query types

In Section 4.3.4 on metrics we said that we also expected the MX and NS query types to yield potentially large answers. Our measurements show, however, that this is not the case. Figure 4.7a and Figure 4.7b respectively show a comparison between the amplification factors we measured for regular domains versus DNSSEC-signed domains for MX and NS queries. As both graphs illustrate, the increase in amplification is low (between two and three times) and – outliers excepted – remains well below the acceptable maximum upper limit we set in Section 4.3.2. This is in line with other regular queries that will be discussed in the next section.

Similarly, we discussed that the TXT query type, that is, for instance, commonly used to convey information about spam filtering, may also result in larger responses. Additionally, it is sometimes also abused by attackers in specially crafted domains created to perform DDoS attacks (see Section 4.2.1). Figure 4.8 shows the comparison in amplification for TXT queries. As the graph shows, the majority of responses is below the acceptable upper limit. We note that for the .nl domain there is a small peak in responses just beyond the acceptable upper limit. Closer examination shows that this peak is caused by authenticated denial-of-existence answers. Recall from the introduction to this section that we only include responses from the dataset in a plot if these responses have one or more actual answers in them. It turns out that the responses that make up this peak are CNAME responses. In DNS, a CNAME functions like an alias that refers to another name⁶. The DNS server will attempt to expand a CNAME answer to include the actual record that the CNAME refers to. If this is not possible (because no record of the queried type exists) then it will return only the CNAME

⁶For more information on CNAMEs, see also §2.3.2, Table 2.3.

Figure 4.8: Amplification of TXT queries

answer together with an authenticated denial-of-existence (NSEC(3)) proof in the authority section of the response. We examined the dataset used to plot Figure 4.8 and compared the average number of answers in the authority section for responses both below as well as above the acceptable upper limit. For responses below the limit there are about 2.98 answers in the authority section, whereas for responses above the upper limit this number is 7.75. Examination of a random sample of domains combined with this information provides a clear indication that this peak is caused by authenticated denial-of-existence answers related to CNAME expansion. We repeated this analysis for the other TLDs and saw a similar pattern albeit with peaks at slightly lower amplification factors (between $15\times$ and $20\times$).

4.5.3 Regular queries (A, AAAA)

We end with queries for the IPv4 (A) and IPv6 (AAAA) addresses of a name, which are the bread and butter of DNS. We start with the first, A queries. Figure 4.9a shows the comparison in amplification factor between regular and DNSSEC-signed domains. As the graph shows, the vast majority of responses to A queries is well below the acceptable upper limit defined in Section 4.3.2. Generally speaking, the increase in amplification for DNSSEC-signed domains is between two and four times. We examined the higher amplification factors, above $12.5\times$ present in the graph, to find out why these responses are larger. We believe that this is caused by a configuration difference in the name servers. Responses with an amplification factor above $12.5\times$ include significantly more

Figure 4.9: Amplification of address queries

answers in the authority and additional section (3.90 and 3.46 for responses with amplification over $12.5\times$, 0.16 and 0.02 for responses with amplification under $12.5\times$). This indicates that answers with an amplification over $12.5\times$ come from name servers that fill the optional authority and additional sections, whereas answers with a lower amplification come from name servers configured to give minimal responses (see also Section 4.5.1).

Figure 4.9b shows the situation for AAAA queries. The graph shows strong similarities with the graph for TXT records, with the majority of the responses falling within the acceptable upper limit and with a small peak for the .nl domain just beyond the upper limit around an amplification factor of $26\times$. Repeating the analysis we applied to TXT and A records shows that the cause of this peak is the same as for TXT records, namely authenticated denial-of-existence answers for CNAME expansions. Again, this was also the case for the other TLDs which have a similar peak around the somewhat lower amplification factor of $18\times$.

4.6 Countermeasures

In Section 4.5 we showed that certain queries for DNSSEC-signed domains yield amplification factors that allow attackers to attain high attack volumes with a relatively small investment in bandwidth on their side. In this section we discuss a number of mitigation strategies to address this problem.

4.6.1 Ingress Filtering

The root cause of any type of amplification attack is attackers having the ability to spoof source IP addresses in connectionless protocols (e.g., UDP). This problem has been known for a long time, and the obvious solution is to stop source address spoofing. This solution is known as ingress filtering. It is described in two IETF best-current-practice documents, BCP 38 [69], and BCP 84 [20] (a follow-up document that specifies ingress filtering for multi-homed networks). The basic premise behind ingress filtering is that network operators only allow traffic to enter their network if the source IP address is a legitimate address within their network. Although this solution is highly effective, it only works if deployed Internet-wide. Unfortunately, that is far from the case. CAIDA runs a project⁷ that measures deployment of BCP 38. Their measurements show that a significant number of networks measured still allow spoofing. As long as this is the case, this mitigation strategy is ineffective.

4.6.2 Response Rate Limiting

As mentioned briefly in Section 4.2.2, Response Rate Limiting (RRL) [202] was introduced in 2012 after a slew of DNS amplification attacks that abused DNSSEC-signed domains. The idea behind RRL is that authoritative name servers rate limit outgoing responses, when responses to the same query are sent repeatedly to the same IP block. At first glance, RRL can significantly dampen the impact of DNS amplification attacks. A closer examination, however, shows a number of problems:

- RRL only works for authoritative name servers; since most attacks do not directly use authoritative name servers but instead abuse open DNS resolvers its mitigating effect is limited. Especially if an attack uses queries that are cached by an open resolver, RRL on the authoritative name server for the domain abused for the attack will have little or no effect.
- As Section 5 of the memorandum on RRL [202] mentions, attackers can craft an attack that circumvents RRL by using a spread of queries in the attack rather than one single query.
- RRL can affect legitimate queries. Resolvers that do not (properly) cache results will suffer from RRL, which results in service degradation for clients behind these resolvers. Additionally, RRL can be turned into a denial-of-service weapon. By flooding authoritative name servers that use RRL

⁷<https://www.caida.org/projects/spoofers/>

with queries in which the source address is spoofed to be that of a legitimate resolver (e.g., the resolver of a large ISP) this legitimate resolver can experience service degradation in resolving specific queries on the authoritative name server under attack.

Thus, although RRL can certainly play a role in mitigation, it is not the definitive solution to name servers being abused for DNS amplification attacks.

4.6.3 EDNS0 Cookies

Before EDNS0 was introduced the potential for amplification attacks using DNS was limited by the hard upper response size limit of 512 bytes. As we have shown in Section 4.2.1, EDNS0 is now in wide-spread use across the Internet. Because it allows for much larger responses, DNS amplification attacks have gained much more potential. As mentioned before, the root cause that allows for these attacks is source address spoofing. A potential solution to this is some form of source address authentication, that allows the recipient of a packet to establish that the source address has not been spoofed. An effective way to implement this is by using cookies, as discussed in [67]. In short, the idea is that a name server does not send large responses to a client using EDNS0 unless the client proves its authenticity using an authentication cookie established during an initial interaction between client and server. We believe that this can be a particularly effective solution to the DNS amplification attack problem, but only if this solution is widely deployed.

4.6.4 Response Size Limiting

Another mitigation strategy is to limit the size of DNS responses such that no answer exceeds the acceptable upper limit for amplification set in Section 4.3.2. The results of our analysis in Section 4.5 show that doing this would not affect most regular queries, but would probably only impact ANY queries. A beneficial side effect of this mitigation strategy is that it also prevents IP fragmentation of DNS responses, which, as we showed in Chapter 3, is a serious stability problem affecting DNSSEC-signed domains. If applied, DNS responses over UDP are restricted to a certain maximum size. Answers exceeding that size will be truncated, forcing clients to retry the query over TCP. This effectively stops attackers from using responses larger than the chosen maximum size as TCP is connection-oriented and thus immune to source address spoofing.

4.6.5 Restricting or Blocking ANY Queries

The biggest amplification problem – if we do not consider crafted domains – is ANY queries. This begs the question whether this query type should be severely restricted or even blocked. Restricting ANY queries can be done using response size limiting (Section 4.6.4) and is automatically done by RRL (Section 4.6.2). A more extreme measure would be to block ANY queries altogether. This can only be done, however, if no legitimate use cases depend on ANY queries. We know, however, that certain software, such as `qmail`⁸, uses ANY queries. Blocking ANY queries might thus affect some legitimate users.

Assessing the full impact of blocking ANY queries is outside the scope of this thesis but definitely warrants further investigation. Since early 2015, an Internet draft document has been under discussion in the IETF DNS Operations Working Group [6]. We note that – just like RRL – blocking ANY queries is not a complete solution as other query types, such as DNSKEY queries, can still result in a significant amount of amplification.

4.7 Related Work

Geva et al. [74] discuss DDoS attacks in a general sense. They performed simulations that illustrate the damage that attacks can cause and describe a spectrum of different DDoS attack mechanisms and various mitigation strategies against these attacks. They conclude that many defense mechanisms are problematic to deploy and that they may struggle to protect against ever increasing threat levels. A similar, but more practical and elaborate overview of DDoS attacks and defense mechanisms is given by Zargar et al. [222].

On the measurement side, Casalicchio et al. [41] describe a highly detailed reference architecture for measuring the stability and security of the DNS. They argue that the DNS is the most important infrastructure underpinning the Internet and that there is a great need to assess the health of the DNS on a continuous basis.

Rossow [163] performed measurements for fourteen protocols that can be abused for amplification attacks. Where we focus on DNSSEC, Rossow covers amplification in its breadth. For DNS, Rossow considers amplification through open resolvers as well as direct abuse of authoritative name servers. In the latter category he examines ANY queries for DNSSEC-signed domains. We corresponded with him and his dataset covers around 25,000 domains distributed over more than seventy TLDs. Rossow finds ANY amplification factors in the same range as we do, although the outliers in our dataset exceed his findings and the

⁸<https://fanf.livejournal.com/122220.html>

distribution of amplification factors differs. We believe that this difference in distribution can be explained by the difference in sample size and composition; we use a much larger sample in which groups of domains hosted by a single operator may dominate. If such an operator has configured their servers such that they are less amenable to amplification then this will be reflected in the distribution of amplification factors in our results.

Highly relevant is work by Anagnostopoulos et al. [13]. They also perform extensive measurements to assess the potential impact of DNS amplification attacks. Where their work differs from our approach is that they focus on open resolvers. They measure numbers of open resolvers for three European countries and assess their capabilities (i.e., what amplification factor can be achieved by abusing them). This makes their work complementary to our study.

4.8 Concluding Remarks

The goal of this chapter is to answer the question how bad DNSSEC really makes DNS amplification. The simple conclusion from our measurements is that it is quite bad. If we only consider *ANY* queries, then DNSSEC-signed domains yield high amplification factors, averaging between $40\times$ and $55\times$. This exceeds the average amplification of regular DNS by a factor of between six and twelve times. Looking deeper, however, the picture is more nuanced. For many common DNS queries, using DNSSEC results in larger responses but the amplification factor mostly stays within the acceptable upper limit (based on the maximum amplification of classic DNS) set in Section 4.3.2. Nevertheless, an attacker needs only one or a few domains with large amplification factors, and by carefully choosing a signed domain, attackers can achieve significant amplification using, e.g., *DNSKEY* queries. It is clear then that this needs to be addressed, both by mitigating the risk of carrying out a successful amplification attack and by improving the DNSSEC protocol. One possible improvement is to use cryptographic signature schemes with more favourable key and signature sizes that reduce the size of DNSSEC responses. Elliptic curve digital signature schemes have such properties. We will discuss this in much more detail in the next chapter (Chapter 5). We also bring back to mind that attacks with large amplification factors were already feasible without DNSSEC. Attackers already use crafted domains and can continue to do so regardless of whether DNSSEC is implemented. DNSSEC does, however, give attackers more options, such as directly abusing authoritative name servers instead of open resolvers and foregoing the use of crafted domains.

There are a number of countermeasures that can be deployed to mitigate the effect of DNS amplification attacks. The first two we discussed (BCP 38 and

RRL) are already in active (albeit not universal) use. The other three, however, require further work. We consider EDNS0 cookies to be a particularly promising strategy for combating DNS amplification attacks but note that deployment of this technology is in its early stages. And, much like BCP 38, the technology is only really effective if it is widely deployed. We already discussed the benefits of response size limiting at the end of Chapter 3. As this mitigation strategy can easily be deployed, we strongly recommend DNS operators who have not deployed configurations for their authoritative name servers to provide minimal responses, to do so. Blocking *ANY* queries, finally, also warrants further work as it can be a good stopgap measure against the worst amplification attacks; in this light, it is important that the draft [6] circulating in the IETF DNSOP working group keeps progressing.

4.8.1 Open Access Data

The data collected for the research in this chapter has been released as open access data. The data can be downloaded from <https://traces.simpleweb.org/>. Furthermore, a technical report with a detailed discussion of the datasets is also available [194]. More information on our approach to open access data can be found in Appendix B.

Acknowledgements

Our thanks go to Siôn Lloyd (Nominet, `.uk`), Marco Davids (SIDN, `.nl`) and Patrik Wallström (IIS, `.se`) for helping out with DNS data for their TLDs. Additionally, we thank VeriSign for sharing data on `.com` and `.net` and the Public Interest Registry (PIR) for sharing data on `.org`.

The Case for Elliptic Curve Cryptography

In Chapters 3 and 4 we studied the major technical problems with DNSSEC. What became clear in these studies is that the common denominator in both problems is DNSSEC’s message size. DNSSEC’s larger responses lead to fragmentation, causing reachability problems, and make DNSSEC-signed domains an attractive vector to abuse in DNS amplification attacks. At the end of Chapter 4, we already mentioned that it may be possible to reduce DNSSEC’s message overhead by using different signature schemes, based on elliptic curve cryptography (ECC). In this chapter we quantify the effectiveness of this approach. Using real-world measurements, we show the effect switching to ECC signature schemes has on the problems studied in Chapters 3 and 4. Furthermore, we ask ourselves: are there potential downsides to switching to ECC for DNSSEC? This chapter is based on a study published earlier as [195].

5.1 Introduction

While DNSSEC can improve the security of the Internet, uptake is still lacklustre. Less than 3% of domains worldwide deploy DNSSEC¹ and at best

¹<https://www.internetsociety.org/deploy360/dnssec/statistics/>

15% of clients are protected by DNSSEC validation². We argue that part of the reason for this is the technical problems we discussed in Chapter 3 and 4. The first, fragmentation, leads to unreachability, which, as we argued in Chapter 3, is an important reason for, e.g., large players such as Google not to deploy a technology such as DNSSEC [47]. The second, amplification, has often been cited by opponents as a reason for not deploying DNSSEC. Recent reports on DDoS attacks suggest that is becoming common practice to abuse DNSSEC-signed domains for amplification attacks [10, 48, 113, 123]. Organisations will be reluctant to deploy DNSSEC if this means that their domain may be quoted as having been abused to perform large attacks.

In this chapter, we argue that the root cause of these problems is the choice of RSA as default signature algorithm for DNSSEC. RSA keys and signatures are large, compared to traditional DNS messages. As we mentioned, though, there are alternatives, based on elliptic curve cryptography (ECC). ECC keys and signatures are much smaller, while their cryptographic strength is excellent. This is attractive for DNSSEC as it reduces response sizes, addressing the problems with fragmentation and amplification. One particular ECC-based scheme, ECDSA, was already standardised for use in DNSSEC in 2012, but as of 2015 is still rarely used in practice. Given the potential benefits, we argue that this should change. Therefore, in this chapter we set out to build a case for a switchover to ECDSA and other elliptic curve signature schemes.

5.1.1 Related Work

The overhead of DNSSEC on the DNS was first studied by Ager et al. [8]. Their work transforms DNS traces without DNSSEC data into traces that do contain DNSSEC, based on a fixed set of rules discussed in the paper. Using these transformed traces, they then simulate the overhead DNSSEC imposes on DNS traffic. Already in this study, ECC is mentioned as a potential alternative to RSA, albeit in not much detail and with key and signature sizes that would not be considered secure today. This chapter adds to their work with an analysis that uses real DNSSEC traffic (rather than simulated traffic). Additionally, we have used state-of-the-art ECC algorithms in our analysis that current recommendations suggest are secure for the coming thirty years at least [21, 170].

Yang et al. [217] perform the first systematic analysis of DNSSEC as an Internet-scale deployment of public key cryptography. They examine cryptographic aspects as well as the complexities of incremental deployment, partial trust chains and key management. What they do not touch on, though, are problems with fragmentation and amplification that we argue are a direct result of choices related to cryptography.

²<https://stats.labs.apnic.net/dnssec/XA>

Herzberg and Shulman [92], like us, discuss the problem of cryptographic algorithm choices in DNSSEC. They propose a protocol for DNS clients and servers to negotiate an optimal cipher suite, similar to the protocol used to negotiate cipher suites for Transport Layer Security (TLS). Their goal is to reduce the amount of cryptographic material that needs to be exchanged, in order to reduce DNS message sizes. While this reduces fragmentation and amplification, it does not reduce the complexity of DNSSEC. Rather, it further complicates the DNSSEC protocol. We choose a different path. Instead of introducing additional complexity, we build a case for a complete switch to elliptic curve cryptography in DNSSEC.

5.2 Problems with DNSSEC

In this section we briefly revisit the two problems with DNSSEC discussed in Chapter 3 and Chapter 4. In addition to this, we discuss an additional issue with DNSSEC: key management.

5.2.1 Revisiting Fragmentation

As we discussed in the thesis so far, DNSSEC responses are larger than regular DNS responses. The reason for this is that they include digital signatures. In Chapter 3, we discussed that this sometimes leads to packet fragmentation. Moreover, some DNSSEC-specific query types are particularly at risk from this. The `DNSKEY` query type – crucial for DNSSEC – can have large responses as it includes all public keys used in signing the zone.

In Chapter 3, we showed that fragmentation is a major problem. Up to 10% of hosts may be unable to handle fragmented responses. Domains with fragmented DNS responses are effectively unreachable for these hosts. Fortunately, as we discussed in Section 3.5.2, it is possible to reduce response sizes for DNSSEC-signed domains by configuring name servers to return so-called minimal responses. Based on this research, SURFnet enabled ‘minimal responses’. We demonstrated in Figure 3.6 that this reduces the average response size by 80%. As we also mentioned, however, configuring a server to return minimal responses does not set a hard limit on response size, and fragmentation may still occur.

To quantify fragmentation under minimal responses, we examined traffic captured from August 2013 to April 2015 at one of SURFnet’s authoritative name servers. On average, out of 12 million DNSSEC responses per day, 0.5% still suffer fragmentation, for approximately 15,000 distinct query names per day. Fragmentation occurs for all common query types and response statuses

(success, referral and non-existence). This underlines that fragmentation remains a problem when deploying DNSSEC, even if authoritative name servers are configured to return minimal responses.

5.2.2 Revisiting Amplification Attacks

We showed in Chapter 4 that DNSSEC sharply increases the potential for DNS amplification attacks. On average, deploying DNSSEC for a domain increases its amplification potential by between six and twelve times. Increasingly, there is evidence that attackers specifically look for DNSSEC-signed domains to abuse in amplification attacks. Krämer et al., for instance, describe an analysis of a bot that specifically [123] asks for DNSSEC information to abuse in amplification attacks. Furthermore, Internet security companies report evidence of the increased use of DNSSEC for amplification attacks. In 2016, Akamai issued a separate security bulletin on the use of DNSSEC-signed domains for amplification [10] and later that year ComputerWorld quotes an analysis by Akamai on a large DDoS attack that specifically used DNSSEC-signed domains [48]. Similarly, security company Kaspersky also reports an increased use of DNSSEC-signed domains for amplification, suggesting that domains in the .gov TLD are abused particularly often as DNSSEC-signing for .gov domains is mandatory [113]. Public honeypot data seems to confirm this, with many .gov domains listed in the top 100 attack domains³.

5.2.3 Key Management

In Chapter 2, Section 2.5.3, we explained that the most commonly used key management scheme for DNSSEC uses two types of keys, a Key Signing Key (KSK) and a Zone Signing Key (ZSK). As we also showed in Figure 2.9, in a typical DNSSEC trust chain, the KSK is used in the secure delegation from the parent zone to the child zone, and the ZSK is used to sign the actual records in the DNS zone. Measurements we performed in mid-2015 show that 99.99% of DNSSEC-signed .com, .net and .org domains use this key model.

As Yang et al. [217] discuss, best practices suggest that keys need to be replaced at regular intervals. The length of the interval depends on, among other things, the cryptographic strength of the key. Key rollovers in DNSSEC require interaction with the parent zone to update the secure delegation (DS). This interaction is often manual and thus should not occur too often. To limit the frequency, keys need to be strong. But there is another requirement: signatures should not be too large to limit the overhead DNSSEC imposes on DNS

³http://www.nothink.org/honeypot_dns.php

messages. This creates a conflict: a low rollover frequency means stronger, larger keys, but that means larger signatures. The KSK/ZSK split balances the two requirements. It allows for a large, long-lived KSK whose signatures only occur in the `DNSKEY` set, versus a small, short-lived ZSK with smaller signatures in the rest of the zone.

There are two issues with the KSK/ZSK key management model, however. First, this split key model means that by definition, the `DNSKEY` set for a zone always has at least two keys in it, meaning that responses to a `DNSKEY` query are almost guaranteed to be large. A further complication is that during a key rollover, the number of keys in the `DNSKEY` set will likely be even higher. A key rollover commonly requires having two keys of that type in the zone during the rollover. If a KSK and ZSK rollover coincide, four keys may be present concurrently. On top of that, some DNSSEC policies require standby keys, further increasing the number of keys in the `DNSKEY` set. Consequently, a `DNSKEY` response is at risk from fragmentation – particularly during rollovers – and is an attractive target for abuse in amplification attacks.

The second issue with the split key model is that it is complex. It requires special software to manage key rollovers, and thus introduces additional potential for failure. Furthermore, the scheme is hard to understand for system administrators and makes troubleshooting in case of problems harder.

An alternative to the split KSK/ZSK model is to use a single key. This much simpler scheme is often referred to as a Combined Signing Key (CSK) because the single key combines the roles of KSK and ZSK. This scheme is currently rarely used, however, and we explain why in the next subsection.

5.2.4 RSA: the Root Cause?

A big contributor to the three problems discussed above is use of the RSA public key cryptosystem in DNSSEC. According to the DNSSEC specification [16], support for RSA is mandatory. In fact, it is the only mandatory algorithm. Virtually all DNSSEC deployments use RSA. Our measurements, e.g., show that in mid-2015 99.99% of signed domains in `.com`, `.net` and `.org` used RSA.

There are two problems with RSA in the context of DNSSEC. First, DNSSEC responses contain one or more signatures. Each 1024-bit RSA signature is 128 bytes in size, each 2048-bit signature 256 bytes. That is significant in relation to a typical DNS message. As we mentioned earlier, including two 2048-bit RSA signatures in a DNS message means it will already exceed the 512-byte maximum message size of classic DNS. Representing an RSA public key in a `DNSKEY` record requires even more space (around four additional bytes). Second, increasing the cryptographic strength of RSA has historically meant doubling the key size (from 512 to 1024 and on to 2048 bits). While there is no crypto-

TLD	KSK: 2048 ZSK: 1024	KSK: 1024 ZSK: 1024	KSK: 2048 ZSK: 2048	KSK: 1280 ZSK: 1280	KSK: 4096 ZSK: 2048	KSK: 4096 ZSK: 4096	Other	!Power of 2 KSK	ZSK
.com	59.9%	37.9%	0.9%	0.3%	0.3%	0.2%	0.5%	0.3%	0.4%
.net	54.3%	42.3%	1.3%	0.4%	0.5%	0.3%	0.9%	0.5%	0.5%
.org	55.4%	41.3%	1.1%	0.3%	0.6%	0.3%	1.0%	0.4%	0.5%

TLD	KSK: 2048 ZSK: 1024	KSK: 1536 ZSK: 1280	CSK: 2048	KSK: 2048 ZSK: 2048	CSK: 1024	KSK: 4096 ZSK: 2048	Other	!Power of 2 KSK	ZSK
.nl	96.2%	2.3%	0.9%	0.2%	0.2%	0.1%	0.1%	2.3%	2.3%

Table 5.1: Key size distribution for .com, .net, .org and .nl, February 2017

graphic need for this (in theory, increasing the key size by a single bit doubles the burden of cryptanalysis), this has become common practice. As RSA is in widespread use and may have been implemented under the assumption that key sizes are always a power of two, changing this practice may be infeasible.

The complex KSK/ZSK split discussed in the previous section comes into play here as well. KSKs are often 2048 bits to be sufficiently strong. But to keep signature sizes manageable, the ZSK is often 1024 bits. This also illustrates why having a CSK is unattractive; either it is 2048 bits, imposing significant signature overhead on all DNS messages, or it is smaller and needs to be rolled frequently with (manual) parent interaction.

Finally, another problem is looming on the horizon: 1024-bit keys are increasingly considered weak. The National Institute of Standards and Technology (NIST), that issues guidelines for the US federal government, disallows use of 1024-bit keys beyond 2014 [21]. But that NIST made an exception for DNSSEC until October 2015 ‘*due to message size constraints [fragmentation]*’ is telling ([23], §8.1.3).

To give an impression of the current use of RSA keys, Table 5.1 gives an overview of the most commonly used key sizes in four TLDs⁴. As the table shows, the most commonly used configuration for the .com, .net and .org TLDs is to either use a 2048-bit KSK with a 1024-bit ZSK, or to use two 1024-bit RSA keys. Additionally, the table shows that the fraction of domains that use RSA keys whose size is not a power of two is 0.5% or less for these three TLDs. The table also shows the distribution of key sizes for .nl. This ccTLD has the largest number of DNSSEC-signed domains of all TLDs (in absolute numbers). As the table shows, the distribution is quite different from that for .com, .net and .org. One particular configuration, a 2048-bit KSK with a 1024-bit ZSK is used by the vast majority of domains. Interestingly, a non-trivial fraction of .nl domains use a CSK. Finally, in .nl there are also significantly more domains that use RSA keys of a size that is not a power of two.

⁴The data for this table comes from the OpenINTEL measurement system that will be discussed in detail in Chapter 7.

5.3 Elliptic Curves in DNSSEC

We argue above that RSA contributes to the DNSSEC problems we described. Fortunately, good alternatives to RSA exist. Elliptic Curve Cryptography (ECC) offers similar functionality (public key cryptography and digital signatures) with smaller keys and signatures. One particular ECC signature scheme, ECDSA, is standardised for use in DNSSEC, but until 2015 was rarely used in practice (our measurements show less than 0.01% of domains in `.com`, `.net` and `.org` using it by mid-2015). This section builds a case for replacing RSA by ECC by quantifying the benefits for DNSSEC, based on real-world measurements.

5.3.1 ECC: a Brief Introduction

Elliptic Curve Cryptography is a form of public key cryptography. It is based on algebraic groups defined by mathematical functions called ‘elliptic curves’ and on operations with points on these curves. While a detailed discussion of ECC is out of scope, we summarise why ECC is an attractive alternative to RSA (for an in-depth introduction to elliptic curve cryptography, see [87]).

Like all public key cryptosystems, ECC is based on the intractability of a number-theoretic problem. In ECC this is the Elliptic Curve Discrete Logarithm Problem (ECDLP). ECDLP is a variant of the classic Discrete Logarithm Problem (DLP), which applies in multiplicative groups like $\mathbb{Z}_N^*$. Given a group G of order N with generator $g \in G$, classic DLP states that given $h = g^x, h \in G$, it is hard to find x , the *discrete logarithm* of h . ECC groups are additive rather than multiplicative. The main operation in ECC is *point multiplication*, implemented as repeated addition. The ECDLP is defined as follows: given two points P and Q on a curve, that satisfy the equation $Q = dP$ for some scalar value d , it is hard to find d , the *elliptic curve discrete logarithm* of Q . What makes ECC attractive is that the most efficient algorithms for breaking ECDLP are of $\mathcal{O}(\sqrt{n})$ where n is the order of the ECC group.

In practice, this means that to achieve 128-bit security, the ECC group order needs to be roughly squared that value, i.e., 256-bits. In comparison, to achieve 128-bit security with RSA requires a group size of 3072-bits [21], twelve times more. As we will show in the following subsections, this smaller group size (and subsequent smaller key size) makes ECC very attractive for use in DNSSEC.

5.3.2 Elliptic Curve Signature Algorithms

In this subsection, we discuss two classes of ECC signature algorithm. First, we discuss the Elliptic Curve Digital Signature Algorithm (ECDSA). This algorithm has been standardised by the IETF for use in DNSSEC [94]. Second, we

discuss the the Edwards-curve Digital Signature Algorithm (EdDSA), a more modern ECC-based signature scheme [110, 173].

ECDSA

The Elliptic Curve Digital Signature Algorithm (ECDSA) is a US Federal Standard published by NIST [146]. ECDSA works in conjunction with a hash function that has an output block size equal to the curve group size. For example, with the P-256 curve (one of the curves recommended in the standard), the SHA256 hash algorithm is used. Signatures are represented using two integers – commonly represented by the tuple (r, s) – with each integer of the same bit size as the curve group. Thus, on a 256-bit curve, signatures are 512 bits in size.

ECDSA has been standardised for use in DNSSEC [94] with two NIST curves, P-256 and P-384. The RFC specifies that public keys are to be stored using the full representation of the curve point (x, y) , meaning that 512 bits are required to store the key. This is suboptimal; using a technique called point compression, key storage can be reduced to storing just x with one extra bit to determine the sign of y (i.e., 257 bits for a public key on P-256). Because of fears over patent infringement, however, the authors of the RFC did not include this optimisation (more on this in Section 5.5).

EdDSA

The Edwards-curve Digital Signature Algorithm (EdDSA) [28] is an alternative to ECDSA based on so-called twisted Edwards curves, a class of elliptic curves introduced by Bernstein, Birkner, Joye, Lange and Peters [27]. The designers claim that EdDSA significantly outperforms ECDSA, both for signing and for validation, while maintaining the same level of security. Measurements of the eBACS benchmarking project confirm this claim [68]. EdDSA is not only attractive from a performance point of view, but its public keys are also smaller. EdDSA public keys are always stored as a 256-bit value, half the size of a canonically-represented ECDSA public key. EdDSA has been standardised for use in IETF protocols [110] and very recently also for use in DNSSEC [173]. Because of its relative novelty, EdDSA is not yet commonly implemented in DNSSEC software.

Implementation Choice	ecdsa384	ecdsa256	ecdsa384csk	ecdsa256csk	eddsasplit	eddsacsk
(a) ECDSA vs. EdDSA	ECDSA	ECDSA	ECDSA	ECDSA	EdDSA	EdDSA
(b) Curve	P-384	P-256	P-384	P-256	Ed25519	Ed25519
(c) KSK/ZSK vs. CSK	KSK/ZSK	KSK/ZSK	CSK	CSK	KSK/ZSK	CSK
	<i>Most Conservative</i>		←————→		<i>Most Beneficial</i>	

Table 5.2: Deployment scenarios for ECC in DNSSEC

5.4 The Effect of ECC on DNSSEC Problems

5.4.1 Revisiting DNSSEC problems

We revisit the DNSSEC problems from Section 5.2 and analyse to what extent switching from RSA to ECC mitigates these problems. In the analysis the following ECC implementation choices are covered:

- (a) ECDSA versus EdDSA.
- (b) The specific curve used to create signatures – the choice of curve determines the size of `DNSKEY` and `RRSIG` records, and also determines the level of security.
- (c) The ‘traditional’ KSK/ZSK split versus a single Combined Signing Key (CSK) – the relevance of this is that the use of the KSK/ZSK split implies that the `DNSKEY` set always contains at least two keys, whereas if the CSK signing scheme is used, the `DNSKEY` set will contain a single key (if no key rollover is in progress).

Table 5.2 provides a set of scenarios covering these choices. The rows of the table show the implementation choices, the columns provide convenient short names for the scenarios. The scenarios are sorted from most conservative (in terms of existing standards and practices, and with respect to security and proven cryptography) to most beneficial in terms of tackling the problems we are attempting to solve (but requiring implementation changes or standardisation and relying on more novel cryptographic algorithms). In the following subsections, we will test to what extent these scenarios solve the problems we discussed using real-world measurements.

5.4.2 Fragmentation

To show the impact of the scenarios in Table 5.2 on fragmentation, we performed two measurements. First, we re-issued queries that resulted in fragmentation in our measurement from Section 5.2.1. We examined if answers to these queries would be fragmented under each of the scenarios and find that even the most

Figure 5.1: CDF for DNSKEY response sizes

conservative scenario (**ecdsa384**) vastly reduces the occurrence of fragmentation. Only 0.3% of previously fragmented responses would still be fragmented under this scenario. Under the most beneficial scenario (**eddsacsk**), less than 0.003% of responses would still be fragmented. To all intents and purposes this is a negligible number.

The second measurement examined the effect of our scenarios on DNSSEC-specific query types that earlier research showed suffer from fragmentation (see, e.g., Section 3.5.1). In particular, the response to a **DNSKEY** query may suffer from fragmentation. We examined data collected in mid-2015 with **DNSKEY** responses for the 0.5 million **.com**, **.net** and **.org** domains that are DNSSEC-signed. For each of these responses, we calculated the response sizes under each one of our scenarios. Figure 5.1 shows the top 10% of a CDF plot of the results. The figure shows that 6.5% of the original **DNSKEY** responses exceed the IPv6 minimum MTU and that 0.6% exceed the MTU of Ethernet. It also shows that even switching to the most conservative scenario (**ecdsa384**) effectively stops fragmentation. But even more remarkable is that two CSK scenarios (**ecdsa256csk**, **eddsacsk**) are so effective that the majority of **DNSKEY** responses would fit in a classic DNS datagram of 512 bytes. We briefly examined the long tail that exceeds the classic DNS limit for these two scenarios and found that simple configuration changes – e.g., enabling ‘minimal responses’ (Section 3.5.2) – can make all answers fit in a classic DNS datagram under these two scenarios.

Figure 5.2: Effect of scenarios on ANY-amplification

5.4.3 Amplification Attacks

To determine the impact of the scenarios in Table 5.2 on amplification, we repeated the amplification measurements we performed for Chapter 4. We limited our measurement to the DNSSEC-signed domains in `.com`, `.net` and `.org` in mid-2015 (approximately 0.5 million domains). Also, we did not examine all query types, but only examined queries that showed high amplification before (ANY and DNSKEY), as well as regular queries (A and AAAA). First, we examine the effect of our scenarios on amplification. Figure 5.2 shows the amplification for the worst amplifier, the ANY query. The figure shows that the amplification that can be achieved with current domains has not changed compared to the measurements from Chapter 4, that were performed a year earlier. Next, the figure illustrates that switching to a conservative ECC scenario (`ecdsa256`) already causes a 40% decrease in amplification. Finally, the CSK scenarios dampen amplification even more (up to 55%).

It can be argued that the ANY query type can be deprecated, thus removing its dangerous amplification potential. Indeed, as discussed in Chapter 4, there is an active draft RFC that specifies how to provide minimal-sized responses to ANY queries [6]. So why spend time decreasing amplification? As we showed in Chapter 4, DNSSEC-specific query types, also have significant amplification potential. The DNSKEY query is integral to DNSSEC. Figure 5.3a shows that DNSKEY queries for a significant proportion (32.3%) of domains exceed the ac-

Figure 5.3: Effects of scenarios on amplification and response sizes

ceptable upper limit we defined⁵ in Chapter 4. But the figure also shows that even the most conservative ECC scenario (**ecdsa384**) dampens amplification for the **DNSKEY** query to such an extent that it falls within our acceptable upper limit. If we then look at the CSK scenarios, just as demonstrated in the previous section on fragmentation, these significantly reduce the amplification potential, such that abuse becomes unattractive because of the low amplification.

Finally, we examined the effect on regular address queries (**A** and **AAAA**). As noted in Chapter 4, the amplification that can be achieved with these query types falls well within the acceptable upper limit. Unsurprisingly, applying one of the ECC scenarios further improves this situation. What is of particular interest – as Figure 5.3b shows – is that if we combine the more favourable ECC scenarios (with a 256-bit group and 512-bit signatures) with response minimisation as discussed in Section 3.5.2, the size of **A** responses is reduced such that they easily fit in ‘classic’ DNS messages of 512 bytes or less.

5.4.4 Key Management

Section 5.2.3 showed that DNSSEC key management is complex using the best current practice (the KSK/ZSK model with frequent rollovers). Furthermore, the choice to have two keys is also an important cause of the large size of **DNSKEY** responses. And while an alternative scheme with a single key exists (the CSK scheme), we argued that the cryptographic limitations of RSA make this scheme unattractive. Table 5.1 confirmed this, showing that the vast majority of DNSSEC-signed domains that use RSA for signing use the KSK/ZSK model.

⁵Recall from Section 4.3.2 that this limit is defined as the maximum amplification that can be achieved with ‘classic’ DNS, i.e. where the maximum DNS message size is limited to 512 bytes.

As we discussed in Section 5.3.1, ECC has a much more favourable cryptographic strength to key size ratio. Both NIST [21] and the European ECRYPT-II project [170] claim that ECC implementations using 256-bit or larger group sizes (with 128-bit security) are sufficiently strong for use beyond the 2030s. Based on this, there is no cryptographic reason to perform key rollovers of ECDSA or EdDSA keys with a high frequency. With that in mind, there is no longer a need for a ZSK, that is replaced often, and a KSK that is replaced less frequently. Instead, a combined key (CSK) can be used, which may optionally be replaced at the same frequency as a KSK in the KSK/ZSK scheme (see [121]).

As we have shown while discussing the impact of our scenarios on fragmentation and amplification, an additional benefit of using the CSK scheme is that it further reduces DNSSEC response sizes. This does not mean that there are no longer valid use cases for the KSK/ZSK split. In high security environments this split is useful as it allows for a KSK that is protected by being kept offline. Arguably, however, in the vast majority of DNSSEC deployments such security measures are unnecessary (and most likely not taken). We therefore strongly advocate that use of the much simpler CSK scheme with ECC-based signatures should be preferred, given the benefits we have shown.

5.5 Potential Issues with ECC

We have made a strong case for switching to an ECC signature scheme for DNSSEC. There are, however, also drawbacks to ECC, which are discussed in this section.

5.5.1 Validation Speed

Validation speed is the most serious challenge when using elliptic curve cryptography for DNSSEC. As the specification for the use of ECDSA in DNSSEC [94] mentions, validation of ECDSA signatures is slower than validation of RSA signatures. Table 5.3 compares RSA to ECC signature schemes in terms of validation speed (number of CPU cycles), based on benchmarks from the eBATS project [68]. The table lists the number of times validation under the ECC scheme is slower than the RSA scheme. Values are based on the arithmetic mean over the median number of CPU cycles for four recent CPU types⁶. Table 5.3 clearly shows ECC schemes are outperformed by RSA. This is especially true for the most common RSA key size (1024-bit, the ZSK size in over 95% of DNSSEC-signed .com, .net, .org and .nl domains) compared to ECDSA P-256.

⁶The h9ivy, hydra8, titan0 and hydra9 systems in the eBATS project.

	RSA 1024	RSA 2048	RSA 3072
ECDSA P-256	26.3× (6.6×)	13.4× (3.4×)	7.6×
ECDSA P-384	79.1×	40.4×	22.8×
EdDSA Ed25519	5.6×	2.9×	1.6×

Table 5.3: Comparing signature validation speed

Since DNSSEC validation is performed frequently by validating DNS resolvers this is potentially a serious issue. While switching to ECC solves issues that are mostly faced by operators of authoritative name servers, its introduction will impose an additional burden on DNS resolver systems. It is as yet unclear if existing DNS resolvers can deal with the increased CPU load of ECC signature validation, especially if deployment of DNSSEC accelerates. We believe this question urgently needs answering, before ECC-based signature schemes are deployed on a large scale in DNSSEC, and study the impact of ECC on DNSSEC validation in the next chapter, Chapter 6. We note that recent advances in the popular OpenSSL library speed up ECDSA on certain common CPU architectures by a factor of $8\times$, and RSA by about $2\times$ [81]. This positively affects the speed difference. We extrapolated the effects of these advances and included them in Table 5.3 in parentheses for ECDSA P-256.

5.5.2 Trust in NIST Curves

It is impossible to talk about ECC without addressing the elephant in the room. Since the Snowden revelations, experts have cast doubt on the trustworthiness of the NIST curves. Two of these – P-256 and P-384 – are standardised for use in DNSSEC. For example, well-known security expert Bruce Schneier suggests in a Guardian article to *‘prefer conventional discrete-log-based systems over elliptic-curve systems; the latter have constants that the NSA influences when they can’* [167]. While it appears likely that a NIST standard for random number generation was influenced⁷, there is no proof that the NIST curves were deliberately weakened. In fact noted experts on elliptic curve cryptography Neil Koblitz and Alfred Menezes provide convincing arguments for why this is (in their words) *‘highly implausible’* [117].

There are, however, other reasons for looking at alternatives to the NIST curves. Both Koblitz and Menezes [117] and Bernstein and Lange [29, 30] raise other concerns about the NIST curves. Their main reasoning is that the use of NIST curves may lead to security problems because of implementation errors

⁷The – by now – infamous Dual_EC_DRBG random number generator. A good write-up of the problems with this random number generator can be found here: https://en.wikipedia.org/wiki/Dual_EC_DRBG.

in the cryptographic operations on these curves. Bernstein and Lange suggest exploring alternatives such as their EdDSA signature scheme Ed25519 [28]. This is one of the reasons why we included Ed25519 in our scenarios (of course, the fact that EdDSA on curve Ed25519 outperforms ECDSA, as Table 5.3 shows, is also attractive).

5.5.3 Intellectual Property Worries

Another long-standing issue with ECC are the intellectual property rights (IPR) surrounding it. CertiCom, a Canadian company⁸, holds patents relating to ECC and its implementation. Fear of litigation over these patents has hampered the adoption of elliptic curve cryptography. The IETF has taken steps around this IPR issue by describing non-patented forms of ECC [135]. This, though, sometimes leads to suboptimal implementation choices. As outlined in Section 5.3.2, the public key for ECDSA can be stored using so-called ‘point compression’, which requires less storage space. This optimisation did not make it into the standard due to IPR concerns. In the interest of fairness, it should be noted that CertiCom has stated [39] it will grant prospective implementors of the standard for ECDSA in DNSSEC [94] a conditional royalty free license to any patents it owns that may be applicable to the standard.

EdDSA, on the other hand, does not seem to suffer from IPR issues. In fact, Bernstein claims⁹ that the base curve for EdDSA does not infringe any patents related to ECC. Bernstein also provides arguments for why many other potential IPR claims about ECC have prior art that pre-dates patenting of the techniques in question.

5.5.4 Support for ECC in DNSSEC Software

The final issue is support for ECC in DNSSEC software. In principle, all major DNSSEC software distributions¹⁰ support ECDSA, both for signing and validation. Practical measurements by Huston and Michaelson, however, show that not all online resolvers that currently perform DNSSEC validation recognise ECDSA signatures, probably because their software needs upgrading [102]. Fortunately, they also observe that the fraction of resolvers that support ECDSA validation is growing (currently at over 80%). We note that for obvious reasons, EdDSA support in DNSSEC software is non-existent. Now that EdDSA has been standardised for use in DNSSEC [173], however, this will gradually

⁸Which is now part of BlackBerry, <https://www.certicom.com/>.

⁹<https://cr.yp.to/ecdh/patents.html>

¹⁰For example, all the implementations listed in Appendix A support ECDSA both for signing and for validation.

change. Finally, not unimportantly, the systems used by TLD registries and domain name registrars need to support registration of ECDSA keys for secure delegations. Most major TLD registries already support this, it is unclear to what extent domain name registrars offer support though.

5.6 Concluding Remarks

As we showed earlier in this thesis (Chapter 3 and Chapter 4), DNSSEC suffers from technical problems (fragmentation and amplification). We argued that the root cause of these problems is the use of RSA as default signature scheme in DNSSEC. In this chapter we build a strong case for using alternative digital signature schemes, based on elliptic curve cryptography. Using real-world measurements, this chapter has shown how the use of ECC can effectively prevent fragmentation of DNSSEC responses as well as significantly reduce the amplification attack potential in DNSSEC. An additional benefit of ECC is that its cryptographic strength means that DNSSEC key management can be simplified significantly.

While the case for using ECC in DNSSEC is very strong, one important question remains: will the additional computational burden of validating ECC signatures be bearable for operational DNS resolvers? We study this key question in the next chapter, Chapter 6.

The Impact of ECC on DNSSEC Validation

The previous chapter built a strong business case for replacing RSA with an elliptic curve signature scheme in DNSSEC. Yet one open question remained: how will the additional workload required to validate ECC signatures impact validating DNS resolvers? In this chapter we perform an in-depth study. We build a model that can reliably predict the required number of signature validations a DNS resolver needs to perform. We then validate this model using real-world data. Finally, we use the model to predict the expected workload on validating DNS resolvers if DNSSEC would completely switch to an ECC signature scheme and if DNSSEC deployment becomes universal. The study presented in this chapter was published earlier as [188].

6.1 Introduction

So far, we have studied the two major technical problems in DNSSEC, fragmentation and amplification. In the previous chapter, we argued that the default signature scheme for DNSSEC, RSA, is at the root of these problems. We then showed that signature schemes based on elliptic curve cryptography (ECC) can effectively solve these two problems in DNSSEC. This is because ECC signatures are significantly smaller in size, leading to smaller DNS responses.

While switching DNSSEC to ECC-based signature algorithms is highly beneficial and solves serious issues in DNSSEC, it introduces a new problem: validation of ECC signatures can be almost two orders of magnitude slower than validation of the RSA signatures currently in widespread use in DNSSEC. This may have consequences for the global DNS infrastructure. Currently, using RSA, the most CPU intensive operation in DNSSEC is the signing process. This is performed at regular intervals by the DNS operators of signed domains. Validation of signatures is performed by recursive caching name servers (DNS resolvers). Thus, a switch from RSA to ECC-based signatures imposes a significant additional burden on DNS resolvers, effectively pushing the cost of cryptographic operations in DNSSEC to the edges of the network.

In this chapter, we address this new problem by answering the question: *What is the performance impact on DNSSEC validation of switching from RSA to ECC-based signature algorithms?* We break this down into the following subquestions:

- *What is an upper bound on the number of ECC signatures a resolver can validate on current hardware?*
- *How many signatures does a typical resolver validate at present?*
- *How would the number of signatures to validate increase for a growing global DNSSEC deployment?*
- *Based on these figures, can a resolver cope with the switch from RSA to ECC in current and future scenarios, where DNSSEC deployment becomes universal?*

6.2 Related Work

Numerous past studies have looked at performance aspects of the DNS. Jung et al. [111] study the performance of DNS resolution from a client perspective, based on trace analyses and simulations. In particular, they study the effect of the Time-To-Live (TTL) of DNS records on cache effectiveness. Gao et al. [72] have more recently revisited the DNS from a resolver perspective. Compared to Jung, they find that the TTL for address (A) records decreased significantly in the intervening ten years since Jung's study, but that the TTL of name server (NS) records remained stable. Wessels et al. [211] perform a measurement and simulation study with the purpose of studying how DNS resolver implementations impact high level DNS servers (i.e. at the root and TLD level). Koç et al. [118], finally, create and validate a model of the DNS. According to their paper,

Figure 6.1: High-level architecture of the DNS

the purpose of their model is to study ongoing and future changes of the DNS, such as the introduction of DNSSEC or the growing deployment of IPv6.

More closely related to the work in this chapter are past studies that have looked at the impact of DNSSEC on the Domain Name System. Wijngaards and Overeinder [213] were the first to study the impact of DNSSEC on DNS resolvers. In their work, they use simulations to quantify the computational overhead of validating digital signatures on DNS resolvers. Wijngaards and Overeinder's study is limited to signatures created using the RSA cryptosystem (which was the only viable option for DNSSEC at the time of their study). Where their work used a simulated DNS environment, the study discussed in this chapter uses a model that we feed with real world data from validating DNS resolvers. Migault et al. [137] carry out a number of performance tests related to DNSSEC, looking both at the impact on authoritative name servers and DNS resolvers. The focus of their work is to assess the impact of DNSSEC deployment for operators. Finally, Lian et al. [132] study if clients are protected by DNSSEC validation, and what problems clients can experience due to name resolution errors related to DNSSEC.

6.3 Background

As discussed in Chapter 2, the high-level architecture of the DNS can be divided into three parts, as shown in Figure 6.1. This chapter focuses on the impact of ECC signature validation on recursive caching name servers, highlighted in the middle of the figure and indicated with the Roman numeral II. Just like in other parts of this thesis, we use the colloquial terms DNS resolver and validating DNS resolver to refer to recursive caching name servers. For the remainder of this

chapter it is important to have an understanding of how the DNS resolution process works (Chapter 2, Section 2.3.3), and in particular how this process works in the case of DNSSEC (Chapter 2, Section 2.5.4).

6.4 Approach

This section discusses the approach taken to predict the impact of ECC validation on validating DNS resolvers. It starts by analysing what factors play a role in the number of signature validations a DNS resolver has to perform. Then, a model is introduced that describes the relationship between the tasks of a DNS resolver and the number of signature validations. The section ends with a validation of the model against real world data measured on four validating DNS resolvers for two popular open source DNS resolver implementations.

6.4.1 Validation by DNS Resolvers

To accurately model validating DNS resolvers, we first need to examine the factors that determine the number of signatures that a resolver needs to validate. Intuitively, one might assume that the prime determinant is the number of incoming queries received from clients. But in actual practice, this is not the case. A validating DNS resolver validates signatures in responses to queries it initiates. While there is a relationship between the number of incoming queries from clients and the number of outgoing queries sent by the DNS resolver, it is sufficient only to consider the number of outgoing queries initiated by the DNS resolver to estimate the number of signatures that need to be validated.

Given the number of outgoing queries, Q , sent by a validating DNS resolver, there are four factors that determine the number of signatures, S_v , that need to be validated. Figure 6.2 shows these factors, and they are described below:

1. Not all queries (Q) initiated by a DNS resolver result in a response (R) from an authoritative name server.
2. DNSSEC is not yet universally deployed. Currently, around 3% of second-level domains on the Internet have deployed DNSSEC. Thus, not all responses will contain signatures. We designate responses that contain signatures with R_s and responses without signatures with R_n .
3. The number of signatures in a response, which we designate S , varies from response to response because:
 - (a) Responses may contain RRs for multiple types. Consider, e.g., a CNAME response (an alias), that is returned to an A query; this re-

Figure 6.2: Relation between outgoing queries and signature validation

sponse may also contain the **A** record(s) that the **CNAME** alias expands to if the authoritative name server knows about these **A** records.

- (b) Next to the *answer* section, which contains the answers to a query, DNS responses may also have an *authority* section (for information about authoritative name servers for a domain) and *additional* section (for additional information, such as the addresses for name servers listed in the authority section). These two sections of a DNS response can also contain signatures.
- (c) DNSSEC has authenticated denial-of-existence to prove that a queried name and type do not exist. Such a proof may require multiple so-called **NSEC** or **NSEC3** records that are each accompanied by a signature.
- 4. Finally, not all signatures need to, or can be validated. Signature validations can be cached by a resolver, or validation may be impossible because no full chain of trust to that particular signature exists. Also, signatures in the authority and additional sections of a response are not always validated. We refer to signatures that need to be validated as S_v and signatures that do not need to be, or cannot be validated as S_n .

6.4.2 Model

The previous subsection discussed the factors that determine the number of signatures a DNS resolver needs to validate. The next step is to create a model of a validating DNS resolver that accurately predicts the number of signature validations (S_v) it needs to perform given a certain workload in terms of the

Figure 6.3: Schematic overview of the measurement setup

number of queries it sends to authoritative name servers (Q). Thus, we want to find a function f , such that:

$$f : Q \rightarrow S_v \quad (6.1)$$

The factors discussed in the previous subsection each play a role in defining f . We hypothesize that each factor can independently be described using a function, and that a combination of these four functions approximates f . In other words:

$$\begin{aligned} &\exists f_1, f_2, f_3, f_4 : \\ &\begin{array}{ll} f_1 : Q \rightarrow R & f_3 : R_s \rightarrow S \\ f_2 : R \rightarrow R_s & f_4 : S \rightarrow S_v \end{array} \\ &f \cong f_4 \circ f_3 \circ f_2 \circ f_1 \end{aligned} \quad (6.2)$$

To gain an intuition about $f_1 \dots f_4$, we examined empirical data collected on three validating DNS resolvers ($r_1 \dots r_3$) operated by SURFnet. We performed a live capture of traffic from clients to these DNS resolvers and replayed this traffic against an instrumented DNS resolver. A schematic overview of our measurement setup is shown in Figure 6.3. As the figure shows, traffic is captured live on the link between clients and the production DNS resolver. This traffic is instantly replayed to the instrumented DNS resolver. The number of queries from clients (Q_c) and responses to clients (R_c) as well as the distribution of DNS response codes is measured for both the production resolver and the instrumented resolver. These measurements are used to verify correct functioning

Figure 6.4: Scatter plots showing the relationships between measured variables

of the instrumented resolver, by checking if the measurements of Q_c and R_c correspond within a small error margin.

To characterise $f_1 \dots f_4$, the variables Q , R , R_s and S are measured on the network link between the instrumented resolver and upstream authoritative name servers. The variable S_v (the actual number of signatures verified) is measured through instrumented code in the DNS resolver software. Figure 6.4 shows four scatterplots that graph measurement data collected on the three resolvers r_1 , r_2 and r_3 over a one week period. The axes show the average parameter value per second over 2-minute time slots. From top-left to bottom-right, plot (a) shows the relation between Q and R . Plot (b) shows the relation between R and R_s . Plot (c) shows the values for R_s and S . Finally, plot (d) shows the data for S and S_v .

The plots suggest a linear relationship between each pair of variables. In other words: they suggest that each function is of the form $f_n = ax + b$. The plots also illustrate that this relationship is weakest between R and R_s (f_2). This can be explained by three intuitions based on the fact that only a fraction of domains worldwide are DNSSEC-signed:

1. Query name popularity among clients influences this relationship; if more popular names are DNSSEC-signed, then the fraction of responses that contain signatures (R_s) will be higher. We expect this to vary between resolvers that have different client populations, and thus different query name popularity distributions. The resolvers used to develop our model, $r_1 \dots r_3$ have different (albeit partially overlapping) client populations. As Figure 6.4b shows, they have differing values for R_s versus R .
2. DNSSEC-deployment across the Internet changes over time. This means that R_s will vary over time. Given current DNSSEC deployment trends, R_s will tend to grow over time. We will exploit this fact later when evaluating future DNSSEC deployment scenarios.
3. Query name popularity among clients varies over time; this can be explained in two ways. First, user behaviour varies during the day (with different interests at different times of day). Second, the distribution of client types varies during the day; automated systems tend to be active all day long, whereas human users tend to show diurnal behaviour (more activity during the day, less during the night). This can be seen in Figure 6.4b as a larger variability in R_s versus R than for the other measured relations.

Given that the plots suggest linear relationships between the variables, we define our model to be a set of parametrised linear functions $f_1 \dots f_4$ specified below:

$$\begin{aligned}
 f_1 : R &= \bar{r}Q + \beta_1 & f_3 : S &= \bar{s}R_s + \beta_3 \\
 f_2 : R_s &= \alpha_s R + \beta_2 & f_4 : S_v &= \alpha_v S + \beta_4
 \end{aligned}$$

with:

$$\begin{aligned}
 \bar{r} &- \text{the average number of responses per query} \\
 \alpha_s &- \text{the fraction of responses with signatures} \\
 \bar{s} &- \text{the average number of signatures per response} \\
 \alpha_v &- \text{the fraction of signatures that is validated}
 \end{aligned} \tag{6.3}$$

These functions can then be combined to give f :

$$\begin{aligned}
 f : S_v &= aQ + b \\
 a &= \alpha_v \bar{s} \alpha_s \bar{r} \\
 b &= \alpha_v (\bar{s}(\alpha_s \beta_1 + \beta_2) + \beta_3) + \beta_4
 \end{aligned} \tag{6.4}$$

Finally, to use the model, the four parameters $\bar{r}$, α_s , $\bar{s}$ and α_v need to be estimated. We do this by performing linear regression on the measurement data

obtained for each parameter. Two approaches for linear regression were considered. The first, Simple Linear Regression (SLR), fits a straight line through a set of points, such that the sum of the squared residuals (the distance between a point and the fitted line) is minimised. Although simple linear regression has the smallest overall error, it is susceptible to outliers. As Figure 6.4 shows, all four variables have some outliers. For this reason, we also considered a second approach, the Theil-Sen Estimator [169, 175], which is robust in the presence of outliers. Comparison of the fit for both approaches shows negligible differences. Therefore, we chose to use the simplest approach, simple linear regression, for the final model.

6.5 Model Validation

Before the model is used to analyse the impact of elliptic curve signature validation on DNS resolvers, the predictive qualities of the model need to be validated first. In order to do this, we evaluate four criteria:

- I. The model works for different DNS resolver implementations.*
- II. The model has stable properties over time; in particular, the values of $\bar{r}$, $\bar{s}$ and α_v remain relatively stable over longer periods of time and only α_s varies significantly as time progresses (as explained in the previous subsection).*
- III. The model works for different client populations (i.e., for different operational DNS resolvers).*
- IV. The model is a good predictor of empirically observed data.*

Only if all four criteria are met can the model be used to make meaningful predictions about the number of signature validations required in future scenarios (where DNSSEC deployment grows). Each criterion is evaluated separately in the paragraphs below. Live data from four production DNS resolvers was used for the evaluation. Table 6.1 characterises each resolver in terms of estimated client population size and the average, peak and minimum workload. Resolvers $r_1 \dots r_3$ (also used for the initial model development discussed in the previous subsection) are operated by SURFnet. These resolvers are open for use by around 200 organisations (universities, research institutes, ...) with some 1 million users connected to the SURFnet network. Resolver r_4 is operated by a medium-size university in The Netherlands. It serves the networks in the university buildings as well as the network in student dormitories on campus.

Resolver	Operator	#Clients	Workload (Queries/Second)		
			Average (24h)	Peak (1h)	Minimum (1h)
r_1	SURFnet	$\pm 125k$	2623 qps	6062 qps	625 qps
r_2	SURFnet	$\pm 58k$	781 qps	1441 qps	373 qps
r_3	SURFnet	$\pm 48k$	568 qps	888 qps	223 qps
r_4	University	$\pm 11k$	281 qps	520 qps	127 qps

Table 6.1: Resolver characteristics

6.5.1 Resolver Implementations

To test whether the model works for different DNS resolver implementations¹, we compared two popular open source packages. The first is Unbound², developed by NLnet Labs. Unbound is a resolver-only implementation, designed from the ground up to support DNSSEC validation, and is optimised for speed. The second is BIND³, the oldest and most popular⁴ open source DNS implementation. BIND implements both resolver and authoritative name server functionality in a single application. Based on the measurement setup shown in Figure 6.3, two instrumented resolvers were deployed, one running Unbound, the other running BIND. Both resolvers ran simultaneously for a day and were fed live client data from production resolver r_1 .

Figure 6.5 shows the measurement data and resulting parameter estimation based on simple linear regression. Three things stand out. First, as subfigures (a) and (c) show, parameters $\bar{r}$ and $\bar{s}$ are almost identical for the two resolver implementations. Given that both resolvers were sent the same query stream, this is as expected. Second, subfigure (b) shows a difference in the fraction of responses that contain signatures (α_s). This is due to implementation differences between Unbound and BIND. Third, as subfigure (d) shows, the most significant implementation difference between Unbound and BIND immediately becomes apparent when we perform the parameter estimation. BIND validates significantly more signatures given the same input queries (almost three times more). The main takeaway is that the model works for the two different resolver implementations. As we will show in more detail when discussing criterion IV, the parameter estimation through linear regression leads to a good fit for both implementations.

¹For more information on popular DNS implementations, see Appendix A.

²Unbound version 1.5.6 was used.

³BIND version 9.10.3 was used.

⁴Recent work suggests BIND has a 55% market share (<https://indico.dns-oarc.net/event/24/session/11/contribution/11/material/slides/0.pdf>).

Figure 6.5: Modelling two different open source resolver implementations

6.5.2 Stability over Time

As we wrote in the introduction to this section, we want to use the model to evaluate future DNSSEC-deployment scenarios. Predictions are only meaningful if the parameters of the model remain stable over time. In particular, $\bar{r}$, $\bar{s}$ and α_v should not change much over time. To evaluate if this is the case, we performed measurements for r_1 at three different times over a four month period.

Figure 6.6 shows the resulting scatter plots and parameter estimations through linear regression. Time t_1 is early October 2015, t_2 is early December 2015 and t_3 is late January 2016. In all three cases, data was captured over a 24 hour period on a working day. As the figure shows, the parameters we are particularly interested in vary little, thus the model is stable over time. The only noticeable fluctuations occur for $\bar{s}$ (c) and α_v (d) at t_2 . We note that this fluctuation is self-canceling, because if $\bar{s}$ rises while α_v decreases the net effect on a prediction for the total model is negligible. A likely explanation for this fluctuation is that at t_2 slightly more responses from authoritative name servers were observed that had signatures in the optional *authority* and *additional* sections of the DNS response. Because these signatures are less likely to be val-

Figure 6.6: Evaluating model parameters over time for r_1

idated, such a change would lead to $\bar{s}$ rising and α_v falling. Finally, note that α_s (b) differs significantly for $t_1 \dots t_3$. This was expected, as this parameter is a function of DNSSEC deployment over time as well as query name popularity. We will be varying α_s in Section 6.6 to simulate changes in global DNSSEC deployment.

6.5.3 Different Client Populations

To evaluate how well the model works for differing client populations, we performed parameter estimations based on measurements for all four resolvers $r_1 \dots r_4$ described in Table 6.1. Despite having different client populations of different sizes, as can already be seen in Figure 6.4, the parameter estimations for $r_1 \dots r_3$ lead to almost the same values for $\bar{r}$, $\bar{s}$ and α_v . The only variation is observed for α_s , which, as mentioned above, does not influence the predictive capabilities of the model.

While we see few differences between $r_1 \dots r_3$, there is a noticeable difference between these three resolvers and r_4 . Figure 6.7 shows a comparison between the

Figure 6.7: Modelling resolvers with different client populations (r_1, r_4)

parameter estimation for r_1 and r_4 . As the figure shows, only $\bar{r}$ is roughly the same, while the other two important parameters, $\bar{s}$ and α_v differ significantly. There are two explanations for this. First, the query name popularity for r_4 differs from that for r_1 ; just like the difference between times $\{t_1, t_3\}$ and t_2 , this most likely means that r_4 receives more responses with signatures in the *additional* and *authority* sections. Second, and more importantly, the client population and query load for r_4 are much smaller than for the other three resolvers. This leads to a much smaller distribution in observed values. This is reflected in the scatter plots for measurement results in Figure 6.7, which show that the blue scatter points for r_4 are bunched much more tightly together towards the bottom left of each of the four subplots. As we will show when evaluating criterion IV next, this leads to a less accurate parameter estimation. This then, is a shortcoming of the model: it will tend to be less accurate for DNS resolvers with a lower query load.

Resolver	Time	DNS software	R^2	Criteria
r_1	t_1	Unbound	0.981	II (Figure 6.6)
r_1	t_2	Unbound	0.966	I & II (Figures 6.5 & 6.6)
r_1	t_2	BIND	0.976	II (Figure 6.6)
r_1	t_3	Unbound	0.987	I (Figure 6.5)
r_1	t_5	Unbound	0.976	III (Figure 6.7)
r_4	t_4	Unbound	0.842	n/a
r_4	t_4	BIND	0.772	n/a
r_4	t_5	Unbound	0.801	III (Figure 6.7)

Table 6.2: R^2 for evaluation scenarios

6.5.4 Predictive Qualities

Finally, we evaluate if the model is a good predictor of observed data. We do this by performing a goodness of fit test that checks how well the prediction from the model fits empirically observed data. In particular, we compare the number of signature validations predicted by the model to the observed number of signature validations and then compute the *coefficient of determination* (Equation 6.5).

$$R^2 = 1 - \frac{\sum_i (y_i - f_i)^2}{\sum_i (y_i - \bar{y}_i)^2} \quad 0 \leq R^2 \leq 1 \quad (6.5)$$

The value of R^2 is a measure for the fraction of the variance in the observed data that can be explained by the model. In general a higher value for R^2 , closer to 1, indicates a better fit, and thus a better model. For each of the evaluations of the previous three criteria, we performed parameter estimation and input the resulting values into the model. Then, using the observed value for Q (the number of outgoing queries from the resolver), we used the model to predict how many signatures would need to be validated ($S_{v_{predicted}}$). We compared this to the number of signatures that were actually validated ($S_{v_{observed}}$) and computed R^2 . All of these evaluations were performed over 24 hour periods on working days.

Table 6.2 shows the resulting R^2 values. The table includes one additional scenario, in which the performance of the model for Unbound and BIND is compared for queries to r_4 at t_4 . The takeaway from the table is that the model is a good predictor in most cases, but as already observed during the evaluation of criterion III and reflected in the value of R^2 , its predictive capabilities are diminished for r_4 because of its smaller population size and lower query load.

Summarising, based on the evaluation of the four criteria we conclude that the model is a good predictor of the number of signature validations (S_v) that need to be performed given a certain number of outgoing queries (Q) from a

DNS resolver. We note, however, that the DNS resolver to which the model is applied must have a sufficiently large client population and a sufficiently high query load. Given that a large client population and high query load constitute a worst-case scenario in terms of the expected number of signature validations, this makes the model well-suited to analyse the impact of ECC signature validation on validating DNS resolvers.

6.6 Predicted Impact

Based on the model introduced in the previous section, this section studies current and future DNSSEC deployment scenarios in order to quantify the impact a DNS-wide switch to elliptic curve-based signature algorithms will have on the global DNS. The section starts by describing the scenarios to be evaluated. Next, baseline benchmarks for the performance of elliptic curve-based signature algorithms are established, which will be used together with the scenario predictions to quantify the impact of a switch to ECC. Finally, the scenarios introduced at the beginning of the section are evaluated.

6.6.1 Scenarios

Our goal is to quantify the impact a DNSSEC-wide switch to ECC-based algorithms will have on the global DNS, and in particular what the performance impact is on validating DNS resolvers. To do this, we will evaluate two scenarios for current and future DNSSEC-deployment, described below:

- I. Current DNSSEC deployment* – this scenario evaluates what the performance impact would be if all domains that currently deploy DNSSEC would switch to an ECC-based signature algorithm overnight.
- II. Popular-domains-first growth to 100% DNSSEC deployment* – this scenario evaluates the performance impact of a growing DNSSEC-deployment in which the most popular domains (in terms of outgoing queries from the resolver) are the first to deploy DNSSEC. Effectively, this is the worst-case scenario as it requires the most signature validations at the shortest possible notice.

When the scenarios are evaluated, the model will be used to measure (for scenario I) or predict (for scenario II) the number of signature validations required in that particular scenario. This number is then compared against a benchmark figure indicating the number of signature validations that can be performed on a single modern CPU core for specific elliptic curve digital signature schemes. Just as in the previous chapter, in which we study the benefits

of the use of ECC in DNSSEC (Chapter 5), we examine multiple signature schemes. We include the two signature schemes currently standardised for use in DNSSEC, ECDSA P-256 and ECDSA P-384 [94, 146]. Next, we include the Ed25519 signature scheme based on twisted Edwards curves [27, 28]. Finally, new in this chapter, we include a more recently introduced twisted Edwards curve-based scheme that is cryptographically stronger, Ed448 [85]. Both Ed25519 and Ed448 have only very recently been standardised for use in DNSSEC by the IETF [173].

6.6.2 ECC Benchmarks

In the previous chapter (Chapter 5) we relied on benchmarks from the eBACS project⁵ to compare RSA and elliptic curve implementations. For this chapter, we performed new benchmark tests. We did this because we explicitly wanted to incorporate recent performance improvements in ECC implementations for both ECDSA and EdDSA. Second, we wanted to standardise benchmarks to a single common CPU architecture, that is representative of modern server systems on which validating DNS resolvers are typically deployed.

The benchmarks were performed for five ECC implementations: three versions of OpenSSL and two independent high-performance implementations of Ed25519 and Ed448 respectively. OpenSSL versions were selected based on the following criteria: the first (0.9.8zh) we consider a ‘legacy’ implementation, the second (1.0.1f) is the mainstream implementation that, for instance, ships with current Ubuntu and Debian Linux distributions and the third implementation (1.0.2e) is the newest stable release branch that incorporates significant performance improvements for ECDSA P-256. Benchmark data was collected by performing 100 independent speed tests for each of the five implementations. A single speed test consists of a ten-second run with continuous calls to signature validation functions, from which the average number of validations per second is calculated. The benchmark tests were run on an Intel Xeon E5-2695 v3 CPU operating at 2.3GHz.

Table 6.3 shows the average results over 100 tests together with the standard deviation. The performance of ECDSA P-256 as well as 1024- and 2048-bit RSA improved significantly between OpenSSL versions. Interestingly, there was no performance improvement for ECDSA P-384. Table 6.4 provides a speed comparison between different implementations. Note that from a cryptographic point of view, comparing 1024-bit RSA to ECDSA P-256 is comparing apples to oranges. The cryptographic strength of ECDSA P-256 is roughly equivalent to 3072-bit RSA [21]. The reason we make this comparison is because RSA

⁵<https://bench.cr.yp.to/index.html>

Implementation	RSA				Signature algorithm and curve							
	1024-bit		2048-bit		ECDSA P-256		ECDSA P-384		Ed25519		Ed448	
	mean	σ	mean	σ	mean	σ	mean	σ	mean	σ	mean	σ
OpenSSL 0.9.8zh	74,221.3	508.2	22,632.1	248.4	2,694.8	29.0	1,285.2	13.7	-	-	-	-
OpenSSL 1.0.1f	95,909.5	721.1	28,948.7	235.9	3,684.8	26.7	1,236.2	12.6	-	-	-	-
OpenSSL 1.0.2e	112,516.0	903.5	35,078.8	507.4	9,786.6	75.7	1,288.9	16.3	-	-	-	-
ed25519-donna	-	-	-	-	-	-	-	-	14,162.4	212.2	-	-
ed448-goldilocks	-	-	-	-	-	-	-	-	-	-	4,816.9	48.3

Table 6.3: ECC benchmarks (signature validations per second, single core)

ECC algorithm	OpenSSL version [†]	Compared to [*]			
		RSA		ECDSA	
		1024	2048	P-256	P-384
ECDSA P-256	0.9.8zh	27.5	8.4	-	-
	1.0.1f	26.0	7.9	-	-
	1.0.2e	11.5	3.6	-	-
ECDSA P-384	0.9.8zh	57.7	17.6	-	-
	1.0.1f	77.6	23.4	-	-
	1.0.2e	87.3	27.2	-	-
Ed25519	(1.0.2e) [‡]	7.9	2.5	0.7	0.1
Ed448	(1.0.2e) [‡]	23.4	7.3	2.0	0.3

^{*}ECC algorithm is x times *slower*

[†]Comparison of ECC and RSA for this OpenSSL version

[‡]Independent implementations compared to OpenSSL 1.0.2e

Table 6.4: Comparison of RSA and ECC signature validation speed

1024-bit is the most common signature type in DNSSEC at present, while ECDSA P-256 is the most attractive candidate to replace the current RSA-based schemes (see Chapter 5).

6.6.3 Scenario Evaluation

Before we evaluate the two scenarios, we make explicit what assumptions we made during the evaluation. We assume that:

- A1 – we only consider signature validations when calculating CPU use (i.e., we do not consider CPU use for other resolver functions, as this is highly dependent on, the number of clients, how many queries these send, ...);
- A2 – the DNS resolver runs on a single CPU core (worst-case scenario);

A3 – there are no future advances in ECC implementation performance compared to the benchmarks in Section 6.6.2;

A4 – DNSSEC policies do not change significantly⁶.

In the following paragraphs we evaluate the two scenarios.

Current DNSSEC Deployment

To evaluate this scenario, we looked at the peak signature validation rate observed on resolver r_1 (the busiest resolver). The highest rates measured were observed in the measurement at t_2 . For the Unbound resolver implementation, validation peaked at 124 signatures per second, for BIND it peaked at 224 signatures per second. Looking at Table 6.3, this is far below the maximum signature validation rates that can be achieved with each of the benchmarked ECC signature schemes. In other words, if all of current DNSSEC deployments on the Internet were to switch to an ECC-based signature scheme overnight, this would not pose a problem for validating DNS resolvers, and would leave ample room for growth both in terms of DNSSEC deployment and an increase in query load on the resolver.

Popular-Domains-First Growth to 100% DNSSEC Deployment

Next, we evaluated the worst-case scenario, where the most popular domains (in terms of number of queries for that domain) enable DNSSEC first. For this evaluation we measured query name popularity for outgoing queries from a DNS resolver. The reason that the query name popularity on the outgoing side was chosen is that this represents the absolute worst-case scenario for the resolver for which the distribution is measured. On the outgoing side, popularity is not just determined by popularity of the name among the client population of the resolver, but is also determined by the Time-To-Live (TTL) of records for certain names. Moderate popularity on the client side combined with a low TTL for DNS records will lead to a high number of outgoing queries (to refresh the cache). For the evaluation of this scenario, we measured the query name popularity for outgoing queries from the busiest DNS resolver r_1 . Figure 6.8 shows the distribution of the query name popularity observed at t_4 ⁷. On the x-axis are domain names ranked in order of popularity (from highest- to lowest-ranked). The y-axis shows the number of queries for each domain name. Both

⁶As we note in Chapter 5, a switch to ECC-based signature schemes warrants simpler key management schemes with a single key per zone; this would result in fewer signature validations.

⁷The distribution is almost identical for other time periods and resolvers.

Figure 6.8: Query name popularity (outgoing queries from r_1 at t_4)

axes are plotted using a logarithmic scale. The shape of Figure 6.8 resembles a Zipf distribution, commonly seen for many phenomena on the Internet [7]. In essence, in a Zipf distribution few entities (in this case domain names) account for the majority of observations (in this case queries). Jung et al. [111] also observed that query name popularity follows a Zipf distribution.

Using the observed distribution, and assuming that DNSSEC is deployed according to popularity rank from highest- to lowest-ranked, we calculate which fraction of queries would contain signatures under this assumption. In other words: we vary model parameter α_s based on the observed distribution. To calculate α_s for $x\%$ of domains deploying DNSSEC, we take the first n domain names that constitute $x\%$ of the total number of observed domains (d_{obs}). Then, with q_i being the number of queries observed for domain i and q_{obs} the total number of queries observed, Equation 6.6 gives the value for α_s . Using the estimated model parameters for the busiest resolver, r_1 at time t_4 , we then predict the number of signature validations required (S_v) for Q outgoing queries.

$$n = \lceil x\% \cdot d_{obs} \rceil \quad \alpha_s = \sum_{i=0}^n \frac{q_i}{q_{obs}} \quad (6.6)$$

Figure 6.9 shows this prediction for r_1 when running the Unbound DNS implementation. The x-axis shows the number of outgoing queries (Q), the y-axis the required number of signature validations (S_v) and the z-axis the percentage of domains (ranked by popularity) that deploy DNSSEC. The figure compares the required number of validations to the worst-performing ECC signature scheme from Table 6.3, ECDSA P-384. The intercept of the red surface (the prediction for S_v) with the gray plane shows where validations would ac-

Figure 6.9: Predicted validation requirements for r_1 running Unbound, compared to peak ECDSA P-384 performance

count for 100% CPU saturation. The blue line indicates the maximum number of observed outgoing queries for r_1 over all measurements performed to date (1738 queries per second). As the figure clearly shows, even if 100% of domains on the Internet deploy DNSSEC using ECDSA P-384 (which is highly unlikely), the number of outgoing queries could almost double before signature validations would account for 100% of the CPU use on a single core.

The worst-case scenario indicates that for Unbound the margin for growth is generous, both in terms of DNSSEC deployment and in terms of the number of outgoing queries. The picture for BIND, however, looks markedly different. Figure 6.10 shows the same plot when using BIND. As the plot shows, in this worst-case scenario BIND will quickly suffer CPU saturation, even if only a small proportion of popular domains deploy DNSSEC. Since it is unlikely that ECDSA P-384 will become the dominant implementation, however, we have also plotted lines for Ed448 (as high-security alternative) and ECDSA P-256 (as short term most likely candidate for deployment). These lines make clear that even for BIND, which clearly performs worse than Unbound in terms of the number of signatures it needs to validate for a given query load, 100% DNSSEC deployment is unlikely to lead to CPU saturation due to signature validations.

Figure 6.10: Predicted validation requirements for r_1 running BIND, compared to peak ECDSA P-384 performance

6.6.4 Resulting Recommendations

Looking at the results of the scenario evaluations, it is clear that – from a performance point of view – even if DNSSEC deployment grows from the current 3% to 100%, and assuming worst-case conditions, the use of ECC-based signature schemes would not pose an insurmountable problem for validating DNS resolvers. This is a very positive result, as we have already shown in Chapter 5 that it is highly attractive to switch DNSSEC to ECC-based signature schemes.

Given these results, we strongly recommend that new DNSSEC deployments select ECC-based signature schemes and that existing implementors consider gradually switching to ECC-based signature schemes. For the short term, it is recommended to choose the ECDSA P-256 scheme. This offers excellent security properties combined with good performance in terms of signature validation speed. Indeed, a major early adopter of ECC-based DNSSEC signing (CloudFlare) has chosen to use ECDSA P-256 (see also Chapter 8). For the longer term we recommend considering Edwards curves-based signature schemes, in particular Ed25519 as future default algorithm and Ed448 for deployments with high security requirements.

6.7 Discussion

6.7.1 Representativeness of Results

In this work, we have used data obtained at four separate DNS resolvers from two different operators. As we have shown in Section 6.5, the simple linear model we have constructed works well in diverse environments. Nevertheless, one could argue that the set of resolvers we included in this study is far from representative of DNS resolvers worldwide. To mitigate this limitation, we have deliberately evaluated results against absolute worst-case scenarios, and have shown that even under these worst-case conditions, the workload imposed on a validating DNS resolver as a result of signature validations for ECC-based signature schemes is far from prohibitive. Additionally, since we illustrated that our model is a good predictor, one can further vary parameters to simulate even worse conditions. For instance, assuming all queries lead to a response ($\bar{r} = 1.0$), all responses contain signatures ($\alpha_s = 1.0$) and that all signatures are validated ($\alpha_v = 1.0$), the average number of signatures per response ($\bar{s}$) would need to reach an average of almost five signatures per response (we observed an average around 2.1) before validation of ECDSA P-256 signatures would saturate a single CPU core at an outgoing query rate of 2000 queries per second.

6.7.2 Denial-of-service Through CPU Starvation

In discussions with a large ISP, the issue of denial-of-service attacks on validating DNS resolvers through CPU starvation was raised as a potential barrier to adoption of ECC-based signature schemes. They argued that an attacker could craft queries to a validating resolver that would lead to large numbers of validations. As Table 6.3 shows, validation of ECC-based signatures is highly CPU intensive (much more so than RSA). Thus, forcing large numbers of validations could lead to CPU starvation. In particular, an attacker could send queries for random non-existent names in a DNSSEC-signed domain, which would lead to authenticated denial-of-existence answers. Every signature over a proof of non-existence would then need to be validated by the resolver.

To assess the impact of such an attack, we performed two attack experiments:

1. Using a domain signed with ECDSA P-256 with a regular NSEC3 chain for authenticated denial-of-existence (i.e., a domain with a pre-computed, fixed set of authenticated denial-of-existence records as specified in [128]). This is currently the most common configuration for DNSSEC-signed domains.

2. Using a domain signed with ECDSA P-256 that uses ‘NSEC3 White Lies’ [75]. In essence, for such a domain every authenticated denial-of-existence answer is minimally enclosing and thus almost certainly unique⁸.

The experimental attacks were performed against a test resolver with a single CPU core specifically set up for the experiment. The two experiments were performed against an instrumented version of Unbound as well as an instrumented version of BIND. The first set of experiments did not result in a denial-of-service of any significance, neither for Unbound nor for BIND. While there is an initial peak workload, where CPU use peaks at 100%, the impact of the attack quickly diminishes as NSEC3 records are cached. Since in a regular NSEC3 chain there is a limited number of records, the attack is in essence self-limiting. The second set of experiments, however, did result in denial-of-service both for Unbound as well as for BIND. In a ‘white lies’ or ‘black lies’ scenario, each authenticated denial-of-existence response is unique. Thus, caching does not help diminish the impact of the attack, requiring the resolver to expend CPU cycles validating the signatures in these responses. There was a notable difference in resilience against this type of attack. While Unbound’s performance degraded, it reliably kept on serving answers from its cache for non-attack query traffic. Queries that required recursion, however, became very slow. After ceasing the attack, Unbound returned to normal operation within seconds. BIND, on the other hand, showed a significant performance degradation, also for responses to non-attack queries that it could have served from its cache. The degradation was such that this attack type can be considered a very effective denial-of-service against BIND. Worse, however, was that BIND did not recover and return to normal service after the attack was stopped. We did not investigate in detail what caused this breakdown in BIND.

We note that this attack could be much worse if domains signed with slower ECC signature schemes are abused (e.g., ECDSA P-384). While there are currently no mitigation mechanisms incorporated into validating DNS resolver implementations, we expect that some form of rate limiting could be an effective countermeasure. Such a mechanism would need to keep track of clients or netblocks that require excessive numbers of signature validations and should rate limit queries from these clients or netblocks. It is likely that mechanisms currently implemented for Response Rate Limiting (RRL)⁹ by authoritative name servers can be re-used.

⁸A recent draft RFC [179] suggests a similar approach the authors call ‘Black Lies’ that uses the NSEC record type and always only requires a single authenticated denial-of-existence proof.

⁹<https://kb.isc.org/article/AA-01000/0/A-Quick-Introduction-to-Response-Rate-Limiting.html>

6.7.3 Remaining hurdles for ECC adoption

While we recommend that operators switch to ECC-based signature schemes, a number of hurdles that may stand in the way of deployment still remain. Whether or not these hurdles are an actual barrier depends on many factors. In general, however, these hurdles are rapidly being tackled by the Internet community.

Software Support

- *TLD registry and registrar secure delegation support* – ECC signature schemes require changes to registry and registrar systems to support the creation of secure delegations. Many registry operators and registrars perform some form of validation on secure delegations that are submitted by domain owners; these checks will need to be updated to support ECC schemes. We note that a number of large TLDs (including `.com`, `.net` and `.org`) already support both currently standardised ECC schemes, ECDSA P-256 and P-384.
- *Signer software support* – DNSSEC signing software needs to support ECC signature schemes. All mainstream implementations support ECDSA P-256 and P-384. Support for the newer algorithms that have only recently been standardised (Ed25519 and Ed448), however, is almost non-existent. Operators may need to upgrade to newer versions of DNSSEC signer software to gain ECC support.
- *Validating DNS resolver support* – on the other side of the DNS, resolver software also needs to support validation of ECC-based signatures. Again, all mainstream implementations support validation of ECDSA P-256 and P-384 signatures, but support of newer algorithms is lacking. As shown in this chapter, validation of ECC-based signatures does not require costly CPU upgrades.

Algorithm Rollover

One particular hurdle was raised by an operator in discussions during the research that led to this chapter: algorithm rollover. For DNSSEC-signed domains that use RSA, there is a gradual upgrade path in case advances in cryptanalysis require stronger keys. RSA keys can simply be increased in size during regular key rollovers. For ECC-based signature schemes, however, this is not possible. Each ECC signature scheme has its own algorithm identifier in DNSSEC, that fixes the curve, and thus the key size. This is because in ECC signature schemes, the hashing algorithm used in signature creation is fixed and linked directly to

the curve group size. Thus, for ECC signature schemes, if stronger keys are required this means an algorithm rollover will need to be performed. Algorithm rollovers (described in [121]) are considered more complex than key rollovers by operators. We note, however, that the likelihood of needing to perform an algorithm rollover because of serious advances in cryptanalysis that compromise ECC schemes such as ECDSA P-256 is small. Both European [170] as well as US [21] authorities currently recommend that 128-bit or higher cryptographic security is sufficient for the next 30 years at least. All ECC schemes discussed in this thesis offer 128-bit security or more.

ECC Deployment in Light of Recent NSA Recommendations

Finally, there has been conflicting advice from the NSA about the adoption of Suite B cryptographic algorithms¹⁰. In August 2015, the NSA recommended that implementors should no longer expend energy on a transition to Suite B algorithms, but should rather focus on implementing post-quantum cryptography (PQC)¹¹. This led to speculation about the motivations behind this message from the NSA as well as the security of elliptic curve cryptography. Noted ECC experts Koblitz and Menezes provide a detailed analysis of the announcement by the NSA [117]. They make a strong case, based on the collective experience of the academic cryptography community over decades, that it is unlikely that there have been significant advances in cryptanalysis against ECC. Furthermore, as we showed in the introduction, the use of ECC-based signature schemes in DNSSEC offers significant benefits, tackling two major current issues with DNSSEC. The main benefits relevant in this context are smaller signatures and keys. None of the current PQC schemes offer these benefits. On the contrary; currently proposed PQC signature schemes all have key and signature sizes ranging from thousands to millions of bits [154], making them unsuitable for an application such as DNSSEC. This makes the NSA recommendation to focus on PQC implementation, rather than Suite B algorithms, impractical for DNSSEC. In light of these considerations, and taking into account the compelling arguments made by experts about ECC security, we stand by our earlier recommendation to switch to the use of ECC algorithms for DNSSEC.

6.8 Concluding Remarks

In this chapter we have conclusively answered the question *can validating DNS resolvers handle the additional CPU load imposed by the validation of elliptic*

¹⁰NIST curves P-256 and P-384 are part of Suite B.

¹¹Algorithms resistant to a particular class of cryptanalysis that can be performed on a sufficiently powerful quantum computer.

curve-based signatures? We have shown that a set of linear relationships accurately models the behaviour of a validating DNS resolver. Using this model, we can reliably predict future developments in signature validation. By combining these results with benchmarks of various elliptic curve digital signature schemes, we have shown that the CPU requirements for signature validations do not exceed the capacity of a single modern CPU core, even if the most CPU-intensive ECC scheme is used.

We discussed several remaining hurdles that operators wishing to switch to ECC-based signature schemes may encounter, such as support for ECC keys by TLD registries and domain name registrars. It is likely, however, that these problems are transient; all are in the process of being resolved by the Internet community [218]. We also discussed one more serious concern, raised by an operator, which is the potential for denial-of-service attacks on validating DNS resolvers through CPU starvation. This threat requires the attention of implementors of validating DNS resolver software, who may be able to implement effective countermeasures by applying some form of rate limiting.

As we have shown in Chapter 5, the use of elliptic curve digital signature schemes in DNSSEC has significant advantages. The use of ECC-based signature schemes can tackle serious issues in current DNSSEC deployments: packet fragmentation (Chapter 3) and amplification attacks (Chapter 4). Given the findings presented in this chapter, we strongly recommend that DNS operators, who are considering to deploy DNSSEC, do so using ECC-based signature schemes. Additionally, existing operators should consider switching to ECC signature schemes as part of their regular upgrade cycle.

Introduction to Part 2

The first part of this thesis focused on improving the DNS itself, in particular the DNS Security Extensions (DNSSEC). In the second part of this thesis, we shift to a broader focus: measuring the data in the DNS to study the security and stability of the Internet. In particular, we introduce a novel large-scale active measurement system for the DNS. Through a number of case studies we show how such active measurements can be used to gauge the state and development of security improvements to the Internet, and how such active measurements can guide future improvements to Internet security.

Large-Scale Active DNS Measurements

The DNS plays a vital role in most Internet services. As discussed in Chapter 2, e.g., there are DNS records that encode which servers handle e-mail for a domain. And to protect e-mail against forgery, technologies such as the Sender Policy Framework (SPF) use the DNS to signal legitimate sources of e-mail for a domain. Consequently, measuring what is in the DNS provides a rich source of information about the evolution of the Internet, especially if such information is collected on a large scale and over long periods of time. Such data can be used for security purposes and, for example, to track the deployment of new protocols on the Internet. In this chapter we introduce a novel large-scale long term DNS measurement infrastructure that collects a unique dataset that currently covers approximately 60% of the global DNS namespace. We illustrate the use of the data collected by this system with four case studies. This chapter is based on two earlier publications [190, 191] and extends these with two additional case studies on e-mail fraud and the effects of a large DDoS attack on a DNS provider.

7.1 Introduction

Almost all networked services depend on the DNS to store information about the service. Often this information is about what IP address to contact, but also whether or not e-mail received from another host is legitimate or should be treated as spam. Thus, measuring the DNS provides a wealth of data about the Internet, ranging from operational practices, to the stability of the infrastructure, to security. Consider, for example, e-mail handling. In the DNS, the **MX** record type specifies which hosts handle e-mail for a domain. Thus, examining which **MX** records are present can tell us, for example, if e-mail handling for that domain is outsourced to a cloud provider such as Google, Microsoft or Yahoo. Another example is the monitoring of protocol adoption such as IPv6 and DNSSEC. The analysis of **AAAA** or **DNSKEY** resource records can provide insight into the adoption of, and operational practices for these protocols over time. Finally, DNS data can also play a vital role in security research, for instance for studying botnets, phishing, spam campaigns, malware and other forms of online fraud.

The DNS has been the focus of, or used in, past measurement studies. These studies, however, had a limited scope, in time, coverage of DNS records or number of domains measured. It remains highly challenging to measure the DNS in a comprehensive, large-scale, and long-term manner. Nonetheless, because this type of measurement can provide such valuable information about the evolution of the Internet, we challenged ourselves to do precisely this. Our research goal is to perform daily active measurements of all domains in the main top-level domains (TLDs) on the Internet (including **.com**, **.net** and **.org**) and to collect this data over long periods of time potentially spanning multiple years.

This chapter focuses on the challenges of achieving this goal by answering the following main question: ‘*How can one perform a daily active DNS measurement of a significant proportion of all domains on the Internet?*’. Throughout the chapter, we show that we can:

- scale measurements to cover the largest TLD (**.com** with 127M names);
- ensure that the traffic such a measurement generates does not adversely affect the global DNS infrastructure;
- and efficiently store and analyse the collected data.

Our measurements create a novel large-scale dataset of great value to the research community as well as in other contexts (e.g., for security and forensic purposes). We therefore make collected data available as open access and provide fellow academic researchers access to our data repositories. How we do this is discussed in more detail toward the end of the chapter and in Appendix B.

Finally, in order to validate our system in practice and to illustrate potential uses of the data it collects, we include four case studies in this chapter. Given the growing research interest in cloud services, the first two case studies focus on the use of cloud e-mail services. Based on almost two years of data collected by the measurement system between March 2015 and February 2017, we studied the following questions:

- Is Google the most popular cloud mail service provider, or are others, such as Microsoft or Yahoo, more popular? Which of these three providers sees the fastest growth?
- Do domains that use these cloud mail services use the Sender Policy Framework (SPF) [114] to combat e-mail forgery, especially since most providers support SPF?

The third case study considers an active case of so-called ‘CEO fraud’. This type of fraud is a highly targeted form of phishing, in which the attacker sends e-mails pretending to be the CEO of a company. The case study illustrates that the data collected by our system also has immediate operational security applications. The fourth case study, finally, investigates the impact of the large DDoS attack against the DNS hosting company Dyn in October 2016.

7.2 Existing Approaches to DNS Measurements

In this section we discuss existing approaches to DNS measurements. We pay particular attention to passive DNS, the only currently existing large-scale long term approach. We show the differences between our active approach and passive DNS. In the remainder of the section we discuss other related work on DNS measurements.

7.2.1 Passive DNS

Development of the only existing large-scale approach to DNS measurements, passive DNS (pDNS) [210], was primarily driven by security benefits. In most cases, pDNS is used to capture DNS traffic between a recursive caching name server (resolver) and the authoritative name servers it communicates with. This ensures that the privacy of users of the resolvers where data is captured is preserved. There are large-scale deployments of pDNS that capture data at many vantage points. Notable examples are Farsight Security’s DNSDB¹ and

¹<https://www.dnsdb.info/>

Figure 7.1: Passive DNS

the pDNS infrastructure operated by CERT.at². These large pDNS deployments are often used in operational security contexts. They are commonly operated by or for Computer Security Incident Response Teams (CSIRTs).

Figure 7.1 shows an abstract view of the DNS. Passive DNS systems typically collect data on the link between recursive caching name servers and authoritative name servers (light gray area). Data from pDNS setups can, e.g., be used to track names associated with IP addresses that exhibit malicious activity. Current passive DNS deployments, such as the aforementioned DNSDB and CERT.at systems, are comparable in scale to our active measurement approach. Where pDNS differs from our approach is that pDNS systems collect dynamic DNS data that is the result of queries by end clients (as Figure 7.1 shows). Thus, pDNS databases will only contain information on domain names that are actively queried by clients of resolvers behind which pDNS data is collected. Another consequence of this is that pDNS databases will have more frequent data points for popular domains, but may also have very few data points for domains that are not very popular (and no data for domains that are never or have never been queried by clients of the resolvers behind which data is collected). This makes pDNS less suitable for longitudinal studies, as there is no guarantee that data for domains is collected at predictable intervals. It also means pDNS will not have data for domains that have, e.g., been configured for a phishing campaign, but have not yet been used by attackers. We also note that the spread of TLDs covered by a large-scale pDNS setup will typically be very diverse, but that it is less likely that a pDNS database will cover a TLD in its entirety.

²The Austrian National CERT team, https://www.cert.at/index_en.html

In contrast, our active measurement covers DNS data for all domains (also domains that are unpopular or not in active use yet) in the TLDs we measure, and it has data for each of these domains for every day. In our active approach we tightly control which queries are performed, thus ensuring that our measurement covers all domains in a TLD. This automatically also includes domains that are not yet in active use, such as, e.g., the aforementioned domains that are set up for malicious use, but have not been used yet. And we not only control which queries are performed, but also when these queries are performed, guaranteeing that data for a domain is collected at predictable points in time (currently once every 24 hours).

7.2.2 Other Related Work

Measuring the DNS has a number of dimensions. In particular, we identify the following: the measurement goal, passive and active approaches, ‘one-shot’ versus longitudinal measurements and vantage points of the measurement. We note that these dimensions are not necessarily independent; for instance: in most cases passively collecting DNS data only makes sense if the measurement is distributed over many vantage points, while collecting data at authoritative name servers is probably limited to a few vantage points as it is difficult for researchers to gain access to such data sources. In the next subsections we describe these dimensions and discuss past and present research efforts in measuring the DNS in the context of these dimensions.

Goal of the Measurement

The DNS can be measured to study the behaviour of the DNS infrastructure itself (e.g., security, resilience, ...), or it can be measured because it provides information about operational practices on the Internet (for example the presence of AAAA records says something about IPv6 deployment). A notable example (on account of scale and running time) of studying the DNS itself is the Internet Domain Survey [108]. This automated survey publishes statistics on the number of IP addresses that have a name associated with it in reverse DNS and has been running since 1987. Another example is a study by Osterweil et al. [148] that examines the day-to-day performance of one of the authoritative name servers for the .com and .net TLDs. Pappas et al. [152] study the effect of configuration errors on the DNS; notably, they perform a number of one-shot active measurements that sample around 10% of the domains in the .com TLD.

Examples of studies examining the DNS to uncover underlying behaviour of, or on, the Internet can, e.g., be found in the security space. Bilge et al. [32] and Perdisci et al. [153] study malicious domain names and botnets, respectively.

Other Passive Measurement Approaches

In addition to large-scale passive DNS collection as discussed in Section 7.2.1, passive traffic measurements are also used to study operational aspects of the DNS. In this case pDNS is often deployed at specific vantage points, for example, [54, 216] study DNS traffic for the `.it` and `.nl` TLDs respectively. Finally, passively collected DNS data can be used to enhance other network measurements. E.g., Bermudez et al. [26] use DNS data to tag network flow data.

Existing Active Measurement Approaches

Active measurements, such as the system we introduce in this chapter, work by sending targeted queries to the DNS. There are fewer examples of active DNS measurements in the literature. Examples include work by Schomp et al. [168] who use active DNS measurements to investigate the client-side DNS infrastructure. They perform these measurements by randomly selecting IPv4 addresses and address blocks to find certain types of DNS servers. Their goal is to characterise the behaviour of the DNS servers themselves, not to collect DNS content. Work in other chapters of this thesis (Chapters 4 and 5) also used active DNS measurements to study aspects of the DNSSEC protocol. Zhu et al. [223] study the deployment of DNS-based Authentication of Named Entities (DANE) by actively sending DNS queries for all DNSSEC-signed domains in `.com` and `.net`.

When compared to existing work that uses active measurements, the approach taken in this chapter stands out in two ways. First, our approach is generic, that is: not specifically designed to study a single aspect of the DNS or the Internet. Second, the scale at which we measure is orders of magnitude larger than previous studies that use active DNS measurements.

Time

For certain research, it is sufficient to perform one or perhaps a few single shot DNS measurements. This is the case for the studies covered in Chapters 3, 4 and 5 of this thesis, and, for example in the studies described in [42, 54, 130]. All of these are based on ‘one-shot’ measurements. Other research, however, looks at developments over time and thus needs DNS data collected over a period of time. For instance, [32, 153] use pDNS data collected over longer periods. There are also examples of active measurements that cover longer periods, e.g. [108, 223]. The intervals at which data is collected varies. For pDNS, data points are scattered over time, as they depend on live queries that arrive at unpredictable times. For active measurements, this varies from twice per year [108] to daily in case of the approach taken in this chapter, and by [223].

Vantage points

The final dimension is whether just a single or multiple vantage points are used to perform the measurement. Whether or not multiple vantage points are necessary depends on the measurement. For instance, measuring location-sensitive DNS answers from content delivery networks [9] obviously requires multiple vantage points, whereas measuring how many domains use a certain DNSSEC configuration can be done from a single vantage point (Chapters 4 and 5) and [223]. Also the scale of the measurement has an impact on the choice of the location and number of vantage points. Osterweil et al. [149], for example, follow the operational status of DNSSEC deployment since its rollout by means of distributed measurement points. Given the ambition of the active DNS measurement system introduced in this chapter, measuring from many vantage points would almost certainly impose an unwanted burden on the global DNS. We therefore foresee distribution as a future expansion aimed at studying specific aspects of DNS behaviour.

7.3 Goals and Challenges

7.3.1 Goals

The goal of the active measurement infrastructure discussed in this chapter is to create a large-scale dataset covering the state of the DNS for a significant proportion of the global namespace. The dataset should record this state at regular intervals, in order to be able to create time series that track trends and developments on the Internet. To achieve this ultimate goal, we define the following sub-goals:

- G1* **Measure every domain in a top-level domain (TLD)** – this allows us to build a comprehensive picture of large parts of the DNS namespace.
- G2* **Be able to measure even the largest TLD (.com)** – if the system can measure .com (127M names) it can also measure other, smaller TLDs.
- G3* **Measure a fixed set of relevant resource records for each domain** – the DNS has different resource record types that serve specific purposes. In Table 7.1 we define the set of queries we want to perform. Queries have been chosen such that they cover the most common DNS uses with the minimum number of queries.
- G4* **Measure each domain once per day** – to be able to create reliable time series, each domain must be measured exactly once every 24 hours.

Resource Record	Description
SOA	The Start Of Authority record specifies key parameters for the DNS zone that reflect operational practices of the DNS operator.
A*	Specifies the IPv4 address for a name.*
AAAA*	Specifies the IPv6 address for a name.*
NS	Specifies the names of the authoritative name servers for a domain.
MX	Specifies the names of the hosts that handle e-mail for a domain.
TXT	Contains arbitrary text strings. This record type is used to convey – among other things – information required for spam filtering and is also often used to prove control over a domain to e.g. cloud and certificate authorities.
SPF	Specifies spam filtering information for a domain. Note that this record type was deprecated in 2014 [114], we query it to study decline of an obsolete record type over time.
DS	The Delegation Signer record references a DNSKEY using a cryptographic hash. It is part of the delegation in a parent zone, together with the NS records and establishes the chain of trust from parent to child DNS zones in DNSSEC.
DNSKEY	Specifies public keys for validating DNSSEC signatures in the DNS zone.
NSEC(3) [†]	Used in DNSSEC to provide authenticated denial-of-existence, i.e. to cryptographically prove that a queried name and record type do not exist. [†]

*Performed for the apex, ‘www’ and ‘mail’ labels, e.g. for `example.com`, `www.example.com` and `mail.example.com`.

[†]Only performed for DNSSEC-signed domains with one or more DNSKEY RRs.

Table 7.1: Query types to perform

- G5 Store at least one year’s worth of data** – to do meaningful research, we should be able to store at least one year of data, and preferably more.
- G6 Analyse data efficiently** – we expect to be collecting data for tens of millions of domains; this means that we must explicitly design for efficient analysis through modern technologies such as the Hadoop ecosystem [212].
- G7 Scalability** – the measurement should scale to both handle TLD growth and to measure additional TLDs. At minimum, the system should be able to measure the main generic TLDs (gTLDs) `.com`, `.net`, and `.org`. Together these contain approximately 50% of domain registrations in the global DNS.

7.3.2 Challenges

To meet the goals above, a number of challenges will have to be overcome:

- C1 Query volume** – as *G1* and *G4* state, we want to be able to measure all domains in the largest TLD (`.com` with 127M names) once every 24 hours. For each name, 14 queries are performed (*G3*). Next to direct queries, the system needs to send additional queries as part of normal DNS recursion

(e.g., to find the authoritative name servers for a domain). A conservative estimate is that this requires one additional query per domain. Thus, querying every domain in `.com` requires at least 1.9 billion queries per day.

- C2* **Query pacing** – a challenge related to *C1* is pacing of queries. It is important that the queries we send do not impose an excessive load on authoritative name servers. Especially traffic flows to the top-level servers that are authoritative for the TLDs that are measured need to be monitored, as queries for individual domains also lead to queries to these servers due to the hierarchical way the DNS is organised. Similarly, we have to monitor the traffic volume to large hosting providers, since these may provide authoritative DNS services for large numbers of domains.
- C3* **Storage** – taking the `.com` TLD as yardstick – and assuming that each of the 14 queries performed for each domain returns ± 150 bytes of data – more than 266GB of results need to be stored per day for `.com` alone. Considering *G5* and *G6* this is particularly challenging.
- C4* **Robustness** – the measurement must run continuously and not suffer from downtime due to maintenance or crashes.
- C5* **Ease of operation** – to meet most of the other challenges outlined above, we foresee a distributed system of machines that perform the measurement. Management and administration of such a distributed infrastructure has to be simple. Additionally, scaling the measurement to incorporate more TLDs should also be straightforward.

7.4 Design Choices

Before implementing such a large-scale measurement system, we carefully considered key choices to make in order to ensure that the system tackles all the challenges and meets all the goals discussed in Section 7.3. This section discusses the major design decisions made while creating the measurement system and motivates our choices by discussing the options we explored.

7.4.1 DNS Software

Given the goal of the system, the most important decision to be made concerned the software to use to perform the actual DNS queries that make up the measurement. Two options were considered:

A bare metal approach – in a bare metal approach the focus is on maximum measurement speed. An example of this approach is the ZMap network scanner

[63], which performs network scans by directly generating Ethernet frames. This approach bypasses all intermediate layers in the network stack, allowing scans at near line-speed. While a bare metal approach potentially is an attractive way to tackle the measurement speed challenge we face, there are disadvantages to taking such an approach. Most importantly, resolving DNS queries is a complex task, much more complex than, e.g., the simple port scans ZMap performs. Re-implementing DNS resolution in a bare metal fashion would require significant effort and runs a high risk of bugs that adversely affect the reliability of the measurement system (challenge C_4).

Using off-the-shelf DNS software – this option relies on maximum re-use of existing software. The measurement software would need to incorporate a simple DNS stub resolver that is capable of sending single queries, and the more complex task of DNS recursion is left to an off-the-shelf implementation. The advantage of such an approach is that it entails the smallest risk of falling into the pitfalls of the complex task of implementing DNS recursion. The disadvantage is, of course, that such an approach will be slower.

Taking the advantages and disadvantages of these two options into consideration, we chose to explore the second option – using standard DNS software as much as possible – in more detail. First, this approach requires the least complexity in terms of software development. This is important especially because it provides the best guarantees for the robustness of the system (challenge C_4), which is a key requirement for long-term data collection (goal G_5). Second, our intuition was that this option would perform sufficiently well to meet challenge C_1 . Next to that, the top-speed performance offered by the first option (bare metal) is not actually a requirement. Rather, to manage the impact of the measurement on the global DNS (challenge C_2) there must be a trade-off between speed and impact of the measurement. In order to confirm our intuition that this approach performs sufficiently well, a proof-of-concept was implemented, the goal of which was to measure the medium-sized `.org` top-level domain. Given the time taken to measure this TLD, we could extrapolate that this approach would make meeting challenge C_1 (measuring very large TLDs such as `.com`) feasible. Based on these considerations we chose to proceed with the implementation of the second approach. All that remained was to determine the impact of the measurement on the global DNS; this is discussed in Section 7.6.3.

7.4.2 Scalability of the Measurement

The second design decision focused on how to best scale the measurement system (goal G_7 and challenge C_5). The first option considered was to run the measurement software on a single system. Given measurements we performed for earlier studies, such as the one discussed in Chapter 4, we knew from ex-

perience that this would put high requirements on the system on which the measurement would run, mainly in terms of CPU utilisation. Choosing this option would therefore make it hard to scale the measurement in the future.

To ensure scalability, we thus chose a distributed approach with a central orchestration system and a cloud of worker nodes. Given the ready availability of cloud computing stacks, we focused on an implementation that is amenable to deployment on cloud platforms, and chose to implement worker nodes as a virtual machine image. While we initially envisage a deployment of the measurement system in a single location, this design choice means that we can scale out to commercial cloud platforms if we run out of local resources, and it also means that we can relocate parts of the measurement to other geographical regions, in case we need to perform measurements from multiple vantage points. The latter option may also be advantageous for measurements on, for example, country-code top-level domains (ccTLDs) with a strong geographic binding, where measuring from a local vantage point relative to the ccTLD can have performance benefits in terms of network latency.

7.4.3 Data Format and Analysis

The final design considerations concern storage and analysis of the measurement results (goals *G5* & *G6* and challenge *C3*). The de-facto toolchain for analysing big datasets – such as the one our measurement system collects – is the Hadoop ecosystem³. Thus, we designed the system such that the resulting measurement data is suited to processing in the Hadoop ecosystem.

For storage, we decided on a two-tiered approach. In the first step, results are to be stored in the Apache Avro file format⁴. Avro is a structured, self-describing data serialisation format with built-in support for compression, which is used by the system to reduce the storage size and thus meet *C3*. We use a simple flat schema that encodes a single DNS record as one row, with sparse storage. This means that only fields belonging to the particular DNS record type that is being stored are filled, other fields are assigned a `null` value. This approach was selected over nested structures because it is simple to map to most database paradigms. As a second step, to further improve analysis performance, measurement data will be converted to the Apache Parquet⁵ columnar storage format *in situ* on a Hadoop cluster on which data is analysed. Traditional row-oriented databases are optimised for access to all the data in a single row. Queries that aggregate data from many rows and that, for instance, accumulate counts based on filters on certain columns are typically inefficient on this type

³For an in-depth introduction to Hadoop, see [212].

⁴<https://avro.apache.org/>

⁵<https://parquet.apache.org/>

of database. A columnar storage system stores all data in a single column sequentially. This makes aggregation across a single or a few columns much more efficient. Additionally, because data in a single column uses the same data type and is typically made up of similar values, sequential columnar data can be compressed efficiently using, e.g., run-length and delta encoding techniques.

There are two reasons for this two-tiered approach. First, storing measurement results in the Avro format with the schema discussed above makes this data suitable for separate long term archival (see 7.5.3 below). The row-oriented nature of the Avro schema means that the data can easily be converted to future database paradigms. Second, the Avro files are structured such that they can also be analysed outside of a Hadoop cluster. All results relating to a single domain name are stored sequentially in an Avro file. Knowledge of this structure allows for development of efficient analysis tools without the help of the Hadoop ecosystem. While performance will be less than on a Hadoop cluster, this makes the data usable to researchers who do not have access to such resources.

To analyse the data collected by the measurement system, we use the Apache Impala⁶ engine. This allows us to perform batch-based analyses using SQL queries. The optimal batch size depends on the complexity of the query; in general, processing is done in batches per day or per calendar month. As an example, the analyses we performed for the first two case studies on cloud e-mail, discussed in Section 7.7, took under 2 hours each, processing over 1 trillion data points.

7.5 System Design and Implementation

Given the design considerations discussed in Section 7.4 and given the goals and challenges outlined in Section 7.3.2 we arrived at the design as depicted in Figure 7.2. The figure shows an overview of the entire system and identifies the three stages that the system is divided into with a grey rectangle. Each stage is described in detail below.

7.5.1 Stage I - Input Data Collection

Stage I collects input data, consisting of full DNS zones for the TLDs measured (Table 7.2). New zone data is collected two times per day. From this data, a delta is computed (domains added and removed). The domain names in a TLD are stored in a separate database per TLD. Each database has two tables, one for the set of active domains (i.e., the current state of the DNS zone), and one with all domains seen since the start of the measurement. The latter table reflects

⁶<https://impala.apache.org/>

Figure 7.2: High-level architecture of the measurement system

developments in the zone and stores timestamps for when a domain name was first seen, when it was last removed from the zone, and when it reappeared in the zone (the latter two are only present if applicable). This design decision means that the Stage I database can be used for stand-alone analysis of changes in the TLD zones. This makes some forms of analysis more efficient, which contributes to achieving goal *G6*.

7.5.2 Stage II - Measurement

The second stage has three functional components. The first is a cluster manager that takes care of dividing work across the second component, a cloud of worker nodes. The third component is a metadata server. It maintains up-to-date IP address to autonomous system (IP-to-AS) mappings as well as Geo IP data.

The cluster manager collects chunks of work from the database. A chunk consists of a set of domains that were last measured before midnight UTC. This ensures that each domain is queried exactly once per day (goals *G1*, *G4*). Chunks are added to a pool of work to be performed, and the domains in each chunk are marked as checked out in the database. As workers process chunks of work, the cluster manager takes care of administrative tasks, managing the pool of available work, and updating the database upon job completion by workers. The cluster manager also monitors measurement progress and will reassign a chunk to a new worker if its current worker takes too long. This prevents worker crashes from causing parts of the measurement to fail (challenge *C4*).

Worker nodes obtain chunks of work from the cluster manager, perform the queries specified in Table 7.1 for each domain in the chunk and collect the results. Workers store all resource records included in the answer section of the

DNS response, including all DNSSEC signatures, CNAME records and full CNAME expansions. Upon completion of a chunk, the worker reports back to the cluster manager and obtains new work. The worker also enriches the collected data based on available metadata (IP-to-AS and Geo IP) and submits the measurement results to the storage system in Stage III. Finally, a worker node will check in with the metadata server to obtain new metadata if available. Worker nodes are generic components; this helps meet goal *G7* as additional workers can be deployed easily to increase measurement throughput.

The cluster manager and worker software were custom-developed (in the C programming language) for this measurement system. The worker uses LDNS⁷ for all DNS-specific processing (issuing queries and parsing query results). To reach goal *G2* and meet challenge *C1*, workers run multiple query threads. This prevents workers from prolonged inactivity if queries time out (which may halt a querying thread for up to 30 seconds). Each worker node also runs a local DNS resolver for which we selected Unbound as software. Caching by this resolver helps reduce the query load on the global DNS (challenge *C2*). Caching of infrastructural information, such as the IP addresses of authoritative name servers, is particularly useful, as large numbers of domains run by a single operator tend to share the same authoritative servers. To ensure fresh data is collected each day, the resolver caches are configured to expire all data after a day. In addition to caching, another important function of the DNS resolver is distributing queries evenly over authoritative name servers, which is especially important to reduce the load on top-level domain servers. Unbound strikes a good balance between query round-trip time (RTT) and distribution of queries over multiple authoritative name servers by randomly selecting authoritative name servers with an RTT below 400ms [219]. As Section 7.6.3 will show, this results in a good distribution of queries over top-level domain servers.

Finally, as discussed in Section 7.4, Stage II of the measurement system is based on virtual machines. These currently run on top of a private cloud infrastructure based on OpenStack⁸. While we run a large number of worker nodes, as will be discussed in the next section (7.6), these consume minimal resources. In the current setup, for each worker only a single CPU core, 2GB of RAM and 5GB of disk are allocated.

7.5.3 Stage III - Storage and Analysis

Stage III takes care of two tasks. First, data is copied from the aggregation point, where workers deposit data, to long-term storage. This serves two purposes: safeguarding a backup copy of the data on reliable storage redundantly

⁷<https://www.nlnetlabs.nl/projects/ldns/>

⁸<https://www.openstack.org/>

TLD	Registry	Type	Country	#Domains	(% of DNS*)	Start Date†
.com	VeriSign	gTLD	-	126.6M	(37.8%)	2015-02-20
.net	VeriSign	gTLD	-	15.0M	(4.5%)	2015-02-20
.org	PIR	gTLD	-	10.5M	(3.1%)	2015-02-20
.info	Afilias	gTLD	-	5.2M	(1.6%)	2016-04-06
.mobi	Afilias	gTLD	-	0.6M	(0.2%)	2016-04-06
CZDS‡	<i>various</i>	gTLD	-	25.4M	(7.6%)	2016-04-01
.nl	SIDN	ccTLD	The Netherlands	5.7M	(1.7%)	2016-02-09
.se	IIS	ccTLD	Sweden	1.4M	(0.4%)	2016-06-07
.nu	IIS	ccTLD	Niue	0.3M	(0.1%)	2016-06-07
.ca	CIRA	ccTLD	Canada	2.5M	(0.7%)	2016-07-07
.fi	Ficora	ccTLD	Finland	0.4M	(0.1%)	2016-11-23
.at	NIC.at	ccTLD	Austria	1.3M	(0.4%)	2017-01-04
.dk	DK Hostmaster	ccTLD	Denmark	1.3M	(0.4%)	2017-02-06
Total				196.2M	(58.6%)	

*Relative to the latest estimate by VeriSign [200]

†Date of first measurement

‡New gTLD zones obtained from ICANN [103]

Table 7.2: Coverage as of February 2017

distributed over two physical locations, and retaining the unmodified source data as measured. Second, data is copied onto a dedicated Hadoop cluster, which we use for analysis. During the copying process, the data is converted to the Parquet format discussed in Section 7.4.3 to enable efficient analysis of the data later on.

7.6 Measurement Performance and Impact

This section discusses operational aspects of the measurement system. It covers what data we currently collect, provides performance metrics, and discusses the impact of the measurement on the global DNS infrastructure. Stage I of the measurement system became operational in July 2014, while Stages II & III have been operational since February 2015.

7.6.1 Coverage

Initially, we obtained access to the zone files of the .com, .net and .org generic top-level domains. Access to these zone files is regulated under contracts^{9,10}

⁹For .com and .net see https://www.verisigninc.com/en_US/channel-resources/domain-registry-products/zone-file/index.xhtml

¹⁰For .org see <https://pir.org/resources/file-zone-access/>

TLD	Stage I		#Workers	Stage II			
	Time			Time (batch)		Time (total)	
	Mean	σ		Mean	σ	Mean	σ
.com	4h 17m	1h 15m	80	54m	6m	17h 10m	2h 23m
.net	45m	31m	10	52m	8m	14h 29m	2h 15m
.org	19m	6m	10	37m	4m	7h 19m	57m

Table 7.3: Measurement performance March - December 2015

with the registry operators of these TLDs. Over time, we have expanded coverage to include additional generic TLDs and ccTLDs. Table 7.2 shows the coverage as of February 2017. An up-to-date overview of the coverage of the measurement can be found on the measurement project website¹¹.

7.6.2 Performance

Stage I retrieves each TLD zone twice a day, extracts the list of domain names from each TLD zone, and computes the delta relative to the previous version. It then updates the databases for each TLD. Table 7.3 shows the average running times for Stage I over 2015 as well as the standard deviation. The variability in running times for .com and .net is caused by intermittent throttling of the zone file download by registry operators. Stage I runs are scheduled to complete before the cluster manager starts checking out batches of work for Stage II. The two daily runs along this schedule guarantee that new domains are part of the measurement within 24 hours of appearing in a TLD.

Table 7.3 also shows the configuration and average measurement times for Stage II over the period March-December 2015. The table shows the number of workers per domain, the average measurement time per batch, and the total duration of a single day measurement. For the latter two values, the mean as well as the standard deviation are displayed. As shown, the average measurement time per batch varies significantly between TLDs. Closer examination reveals two reasons for this. For .com, the higher average duration is due to certain batches being dominated by domains registered from China. Network latency per query causes these batches to have significantly longer measurement times. The average round-trip time (RTT) per query for these batches is up to 7 times higher than the average RTT. For .net, the duration per batch is higher because the RTT for queries is about a third higher than for .org on average. There appears to be no discernible cause for this; it is most likely due to a difference in the infrastructure of the TLD.

¹¹OpenINTEL, <https://www.openintel.nl/>, under ‘Coverage’.

TLD	Results for December 31, 2015				Averages over Mar-Dec 2015					
	#Results	#Domains	Size	(Raw)	Res./Dom.		NXDOMAIN		No result	
					Mean	σ	Mean	σ	Mean	σ
.com	1419M	122.3M	28.8GB	(211.6GB)	12.35	0.04	0.85%	0.17%	5.82%	0.66%
.net	166M	15.5M	3.4GB	(24.3GB)	11.93	0.03	1.23%	0.19%	8.54%	1.23%
.org	125M	10.7M	2.5GB	(18.4GB)	12.55	0.04	1.61%	0.22%	7.15%	1.61%

Table 7.4: Analysis of Stage II measurement results March - December 2015

As Table 7.3 shows, the system manages to perform a full measurement well within a 24-hour window, meeting goals $G1$, $G2$ and $G4$. The total measurement time per day, however, varies substantially. This can be explained by two effects. First, stage I runs twice per day in order to ensure that new domains become part of a measurement within 24 hours. As a result of this, there are occasional measurements for a small number of batches at the end of the day. The total measurement time is computed as the time between the first measured domain on a day and the last. Thus, these late night batches skew the total measurement time. Second, all three TLDs included in the measurement have grown over the period for which the value was computed, leading to longer overall measurement times at the end of the period. Nevertheless, even the longest measurement (for .com) has ample room to run longer while still remaining within a 24-hour window.

To assess how future-proof the measurement is, we experimented with the number of workers per domain to bring down the measurement time. There is a strong relation between the number of workers and the average duration of the measurement, despite the fact that worker VMs share hardware and network infrastructure. We can thus meet goal $G7$ and cope with growth in the number of domains by adding additional workers. The average batch duration and overall measurement time are monitored continuously so additional workers can be provisioned on time to remain within the 24 hour window set in goal $G4$. CPU utilisation of the workers is also monitored and we aim for an average utilisation between 25% and 50%, to strike a balance between keeping room for brief bursts of high activity while not underutilising resources. In general, around a quarter of CPU use on the worker is due to our measurement application while the other three quarters are used by Unbound.

Table 7.4 shows a statistical analysis of the measurement results over the period March-December 2015. The left-hand side of the table shows the daily statistics for December 31, 2015. The first column shows the total number of results per TLD, followed by the number of domains for which data was successfully collected. The next two columns show the size of the collected data per TLD. Looking at the amount of data produced per day shows that the data

compression discussed in Section 7.4.3 works well, achieving a stable average compression rate of 1 : 7.4.

The right-hand side of the table shows three average metrics over the period March-December 2015. These metrics are an indication of the stability of the measurement. The first metric shown is the average number of results per domain for which at least one data point was successfully collected. Such a data point is defined as a DNS response with the status code set to `NOERROR`. As the table shows, the averages for the three TLDs that were analysed are close together. More importantly, the standard deviation for this metric is very low for all three TLDs, suggesting that there is little day-to-day variability in this metric. This is a strong indication that the measurement is very stable over time. Note that the average number of results per domain is lower than the number of queries that we perform (14 as shown in Table 7.1). This was expected, since not every queried type or name will exist in every domain (for example, relatively few domains in these three TLDs are DNSSEC-signed so there will be no response to DNSSEC-specific queries).

The second metric shown is the average percentage of domains for which an `NXDOMAIN` response was returned to the first query sent for the domain. This indicates that either the domain was removed from the TLD or that the authoritative name servers configured in the delegation for the domain no longer return data for the domain. There are slight differences between the TLDs, with a slightly higher percentage of domains for which an `NXDOMAIN` response was received for the `.net` and `.org` TLDs.

The third and final metric shown is the average percentage of domains for which no successful results were obtained. This means that the first query for these domains returned an error, or timed out. This figure gives an indication of the number of domains with configuration problems. A common problem are so-called lame delegations (domains for which none of the delegated name servers respond to queries) [152]. Again, we observe differences between the three TLDs, with `.net` scoring the worst in terms of misconfigured domains. Intra-day variability for this metric is also higher compared to that of the other metrics.

7.6.3 Impact on the DNS

As discussed in Section 7.3.2 (challenge *C2*), we have to ensure that the measurement does not impose an unacceptable burden on the global DNS infrastructure. There are two reasons for this. First, we consider it ethically unacceptable if the measurement were to put significant load on individual DNS servers. This might negatively impact DNS performance for ‘real’ users. Second, the contracts under which we gained access to the TLD zone files for `.com`, `.net` and `.org`

Figure 7.3: Measurement flows versus other DNS flows in the SURFnet network

all stipulate that it is not allowed to run ‘...*high volume, automated, electronic processes that send queries or data to the systems of [the] Registry Operator ... except as reasonably necessary...*’. While this clause pertains mostly to the registry service itself and is in spirit intended to stop aggressive scanning of registry data in order to claim domain names that have also been registered in other TLDs, we nevertheless also apply it to our measurement and strive to minimise the load on the DNS servers operated by the TLD registries.

An obvious way of limiting the load imposed by the measurement is to actively rate limit queries. We chose not to do this for two reasons. First, to support this form of throttling, modifications to the standard DNS resolver software we use would be required. Second, the query load is not distributed evenly over authoritative name servers because of the hierarchical nature of the DNS. Authoritative name servers for top-level domains typically receive many more queries because they have to be consulted to find the specific authoritative name servers for every domain name measured. Conversely, these servers higher up the DNS hierarchy are designed and configured to handle many more queries. Thus, we would have to apply a different rate limiting policy to these servers, making the measurement system much more complex. Instead, our approach is to analyse the impact of the measurement, to show that our system design makes rate limiting unnecessary.

Figure 7.4: CDF showing the distribution of flow rates to individual IPs

To gauge the impact of measurements, flow data was collected for the network from which the measurement operates. The infrastructure is hosted by SURFnet, which collects sampled flow data from its core routers with a sampling rate of 1 : 100. While sampling means some flows (especially very small ones) will be missed, it provides a good picture of the top talkers. We are interested in these, since these are the systems burdened the most by the measurement. To get an idea about the magnitude of the query volume that the measurement generates, we compared the query volume to that of the entire SURFnet network. Figure 7.3 shows this comparison; the traffic volume from the measurement system exceeds the regular DNS traffic from the SURFnet network. This network has over 1 million end users in some 200 institutes for higher education and research, so the query volume from the measurement system is quite significant.

To quantify how much traffic individual IP addresses receive, we examined outgoing flows for 24 hours ordered by average number of packets per second (pps)¹². Figure 7.4 shows the top 1% of a CDF for the flow rate in pps. What is immediately evident is that there are very few flows with a high pps rate. Second, no flow exceeds 400 pps. Only 35 IPs are true top talkers (more than 100 pps). Unsurprisingly, the top of the list consists exclusively of gTLD DNS servers for `.com` and `.net`. On average, each of these servers (there are 13)

¹²The flow rate was adjusted to correct for the 1 : 100 sampling.

receives ± 400 queries per second. A study from 2011 [148] reports that one particular gTLD DNS server receives over 900 million queries per day ($\pm 10,400$ per second). Under the conservative assumption that the query load did not increase since 2011, our measurement would add 3.8% to the query load of that server. More recent figures from VeriSign¹³ suggest that the actual figure is probably lower. We estimate that on average 200 million queries per day from our measurement reach the gTLD name servers. Given the more recent figures from VeriSign, this means that the measurement would account for between 0.1% and 0.2% of all queries to the gTLD servers. Also, in private communication, VeriSign has indicated that they see the measurement and that while it is a non-trivial amount of traffic, it is not problematic.

The next group of top talkers receives less than 200 queries per second. Closer examination shows these top talkers are either companies that practice domain parking¹⁴ or very large domain name hosters. While we have no data on the infrastructure of these companies, it is safe to assume that a query rate of less than 200 queries per second can easily be handled by a name server. One thing should be noted: the figures provided are averages over one measurement period, meaning there may be peaks during which more traffic is sent. While it is hard to quantify to what extent such peaks occur, they are most likely not extreme as that would have shown up in Figure 7.3. Nevertheless, to ensure that measurement spikes do not cause problems, we have active flow monitoring filters that send alerts if traffic to the top talkers exceeds pre-set thresholds. More information on our measurement practices can be found in Appendix C.

This analysis demonstrates that the measurement does not impose an excessive burden on the global DNS infrastructure (challenge *C2*). Nevertheless, the load is significant, which makes it undesirable that large numbers of researchers start running similar measurements. Therefore, we pay specific attention to data sharing in Section 7.8.

7.7 Case Studies

This section contains four case studies that serve to illustrate how the datasets collected by our active measurement system can be used. The first two case studies cover the questions regarding the use of cloud mail service providers introduced in Section 7.1. The third case study showcases how datasets collected by the measurement platform can be used for handling operational security incidents and to perform forensic analyses. The fourth and final case study investigates the effect of a large DDoS attack on a DNS operator.

¹³<https://www.verisign.com/assets/infographic-dnib-Q22016.pdf>

¹⁴https://en.wikipedia.org/wiki/Domain_parking

7.7.1 Case Study I - Use of Cloud E-mail Services

E-mail is one of the oldest services on the Internet. Up until the mid 2000s, mail was either hosted on premises or a service provided by the ISP. Increasingly, however, there is a trend to outsource e-mail to cloud service providers. In this context, we discern three classes of service provider. First, *hosting providers* offer domain registration, web hosting, (virtual) private servers and e-mail. These providers often provide basic e-mail services with few user mailboxes or the option to forward mail to an address set by the user. Second, *cloud providers* offer fully hosted office ICT services. Their service offering in the e-mail space is often rich, allowing customers to provision e-mail accounts for all their users and integrating every day office requirements such as calendaring and document sharing and editing. Third, *protection services* focus on protecting e-mail against malware, phishing, spam and other malicious activity. They process e-mail to filter unwanted content and forward sanitised results to another mail service. This first case study focuses on the second category, *cloud providers*. Based on data collected by our measurement platform over a period of almost two years, between March 2015 and February 2017, we study the use of such services in the .com top-level domain.

To identify which e-mail providers handle e-mail for the most domains in the .com TLD, we examined the *Mail eXchange* (MX) records in the DNS. The first step in the analysis identified the top MX records used by domains in .com by examining all data points for a single day (March 1, 2015) in the dataset. MX records were grouped by second-level domain (SLD) to filter out multiple records that point to hosts within the same service provider. For example, the SLD for Microsoft's Office 365 cloud service offering is *outlook.com*. We manually classified the results of this analysis to determine which service provider the MX records belong to and in which of the three classes of service provider they fall. Looking at cloud providers, we find that on March 1, 2015 the top three consists of what we would term the usual suspects: Google (serving 4.09M domains), Microsoft Office 365 (948k domains) and Yahoo (609k domains). Note that, while these are large numbers, cloud providers are not the dominant mail handler. The most common MX record by far (± 27 M) for domains in .com points to GoDaddy, a large domain name registrar and hosting provider.

In the introduction to this chapter we asked the question: '*which cloud e-mail provider sees the fastest growth?*'. Intuitively one might answer 'Google'. Surprisingly, however, that is not the case. In absolute numbers, the growth for Google and Microsoft is almost the same. In relative numbers, however, Microsoft grows much faster than Google, growing by almost 120% between March 2015 and the middle of February 2017. In absolute numbers, Microsoft went from 948k domains using their service to 2.04M, Google went from 4.09M

Figure 7.5: Relative growth in use of cloud e-mail providers for the .com TLD

to 5.24M, and Yahoo dropped from 609k to 456k. Figure 7.5 shows the growth relative to the starting point of the analysis (March 1, 2015) in the number of domains that use one of the three cloud mail providers. Again, Microsoft is by far the fastest grower. However, there is a twist to that figure. The blue line shown for Microsoft is an aggregate of domains that use Windows Live (formerly Hotmail) and Office 365. Microsoft has discontinued Windows Live as a brand for mail services and this is visible in the data. The dashed line with open dots shows the decline in use of Windows Live (hotmail.com). Looking purely at Office 365 (dashed line with closed dots), Microsoft's growth is even more noticeable. One explanation for Microsoft's fast growth can be that the large registrar and hoster GoDaddy (mentioned above) is an Office 365 reseller since 2014¹⁵. A staggering 70% of the growth in domains using Office 365 can be directly attributed to domains registered through GoDaddy. Also of interest is the slow decline of Yahoo. While we did not study this in detail, we note that Yahoo has regularly been in news headlines over the past years as struggling.

The measurement system is not only suited to one-shot analyses and time series, but can also be used to detect significant anomalies in the DNS namespace. To illustrate this, we discuss an example anomaly encountered while

¹⁵<http://www.computerworld.com/article/2487663/enterprise-applications/godaddy-touts-simplicity-over-price-as-it-launches-office-365-sales.html>

performing the analysis of MX records above. In the middle of May 2015 a sharp decline occurred for one of the top MX SLDs, from 2.51M domains advertising this record to 1.27M. While the provider the MX SLD belongs to is not a cloud mail provider¹⁶, we investigated the drop nevertheless, to ensure that this anomaly was not caused by problems with the measurement. Interestingly, it turns out that this MX SLD is associated with a service that appears to be targeted at companies specialised in domain parking. The goal of the service is to respond to e-mails sent to parked domains. The assumption behind this appears to be that users may erroneously send e-mail to parked domains; rather than returning a standard error message, the service will return a customised error containing advertisements. The sharp drop in May 2015 is caused by a mass change in MX records previously pointing to this service. We did not analyse the rationale for this change further, but leave this to future study.

7.7.2 Case Study II - Sender Policy Framework Practices

A common problem with e-mail is sender forgery. This is the case when a rogue or compromised mail server with no relation to a domain sends e-mails that appear to originate from this domain. To combat this, the Sender Policy Framework (SPF) was introduced [114]. SPF allows domain owners to specify which servers may send e-mails on their behalf, and as such helps combat forgery. Domain owners publish SPF information¹⁷ in the DNS by means of a TXT record (cf. Table 7.1). This case study evaluates the use of SPF by domains that use one of the three large cloud e-mail providers from the previous case study.

Like the first case study, data was analysed over almost two years, from March 2015 until the middle of February 2017. We searched for the presence of SPF information for domains that use either Google, Microsoft, or Yahoo to handle their e-mail. Figure 7.6 shows the result of this analysis. The lines in the figure represent the fraction of domains that use either Google or Microsoft and that publish SPF information in the DNS. Yahoo is not shown in the figure as less than 0.6% of domains that use Yahoo's mail service publish SPF information. Significant numbers of both users of Google's and of Microsoft's services publish SPF records. There is, however, a surprising difference between the two. As the figure shows, around 31.3% of domains that use Google publish SPF information, growing to 38.2% at the end of the period. For Microsoft these figures are significantly higher, growing from 92.3% to 95.1%. Both Microsoft and Google provide instructions on how to publish SPF information when using their service. We have not examined in detail why this difference in SPF deploy-

¹⁶For ethical reasons, we do not disclose the name of the company as it is not a large publicly traded company.

¹⁷http://www.openspf.org/SPF_Record_Syntax

Figure 7.6: SPF usage growth for cloud e-mail providers in the .com TLD

ment occurs. One possible explanation is that the majority of domains that use Microsoft’s Office 365 do so via resellers that set the appropriate SPF records automatically. For example, of domains using Office 365 through GoDaddy, 98.8% publish SPF information. This certainly merits further study as the use of SPF is an important tool in combating e-mail fraud.

7.7.3 Case Study III: CEO Fraud

The third case study showcases how the dataset collected by our measurement system can help with real-world security incidents and forensic analysis. On August 30, 2016 SURFnet’s Computer Security Incident Response Team (CSIRT) reported an incident of so-called ‘CEO fraud’ directed at SURFnet itself, and possibly also SURFnet’s constituency¹⁸. CEO fraud is a highly targeted form of phishing, and a form of Business E-mail Compromise (BEC) scam [134]. Typically, in this type of scam, attackers send e-mail to, e.g., the finance department of a company. The attacker pretends to be the CEO of the company (hence the name of the scam) and, for example, asks for urgent assistance in carrying out a transfer of funds. There appears to be a significant investment on the side of fraudsters performing this type of scam. They, for instance, find out the actual

¹⁸Institutes for higher education and research in The Netherlands.

name of the CEO, and use the native language of the company. In the case studied here, they also took the trouble to register a domain name that has a high degree of similarity to the actual company domain name.

Shortly after the first report by SURFnet's CSIRT, a list with more domain names that appeared to be associated with the same fraudsters was circulated. This list also contained domain names with high similarity to organisations in SURFnet's constituency. We wanted to find out if the dataset collected by the measurement system discussed in this chapter could assist in tracking down more fraudulent domains. We took the original domain name from the incident report¹⁹, `'surfnet-nl.net'` and studied what information was available in the dataset. When we did this, five observations stand out:

1. There was a limited set of record types available, specifically **MX**, **NS**, **TXT** and **SOA**. Interestingly, no address records (**A** or **AAAA**) were present. While this is hard to confirm, this appears to suggest that the domain was set up solely for handling e-mail.
2. The authoritative name servers for the domain were all in the `bdm.microsoftonline.com` domain, which is used by Microsoft Office 365 managed DNS.
3. The **MX** record for the domain also points to Office 365.
4. The first of two **TXT** records contains an SPF record for Office 365, suggesting that the fraudsters wanted to be sure that e-mail from their fraudulent domain would not be marked as spam.
5. The second **TXT** record contains an Office 365 specific token. The specific semantics of these tokens is not documented as far as we could determine, they have a general form that looks like the following example:

```
"mscid=tnFgnza [...] AQiaChwUSRUHY+Q=="
```

In other words, the string `'mscid'` followed by a Base64-encoded value.

The fifth observation is of particular interest. Further examination showed that this token is linked to the specific user, account or environment with which the domain is associated in Office 365. Using this token as a search criterion, we searched for other domains using the same token. This resulted in a much larger set of domains being found. We extended the search to cover days preceding the incident and days following the incident. An overview of the result is shown in Table 7.5. An example of one of the additional domains we found

¹⁹We have obtained approval from SURFnet's Security Officer to include this information in this case study.

TLD	#Domains sharing token with ‘surfnet-nl.net’												
	August						September						
	26	27	28	29	30	31	1	2	3	4	5	6	7
.com	36	36	77	77	199	259	306	334	334	352	352	394	404
.net	-	2	2	2	17	17	20	38	43	43	44	54	54
.org	-	15	15	15	18	18	23	23	26	26	28	28	28
Total	36	53	94	94	234	294	349	395	403	421	424	476	486

Table 7.5: Development of related CEO fraud domains August/September 2016

is ‘utwente-nl.net’, which is clearly meant to be similar to ‘utwente.nl’, the main domain of the University of Twente²⁰.

By systematically analysing the additional domain names extracted from the dataset, we find that:

- There are already domains that use the same token as ‘surfnet-nl.net’ before the incident is first reported by SURFnet’s CSIRT.
- The domain ‘surfnet-nl.net’ first appears on the day the incident is reported by SURFnet’s CSIRT (August 30).
- The number of domains set up for the fraud campaign grows steadily over a 13-day period. After September 7 no new domains appear, and those that exist on that day still exist months later (i.e., are not actively removed by either the fraudsters or anti-abuse organisations).
- We observe multiple ways in which the fraudsters attempt to make domains look similar to their legitimate equivalents. For example, by registering names like ‘surfnet-nl.net’, in which the legitimate name is taken and the dot between the second-level domain name and the TLD is replaced by a hyphen. Another example is the use of character substitutions where, e.g., letters are substituted by similar looking digits (‘1’ instead of ‘i’).
- All domains have been registered through the same registrar and have anonymised registrant data (so-called ‘WHOIS anonymisation’).

This case study illustrates how actively collected DNS data can be used for operational security purposes and forensics. The additional domains found were shared with the security community in The Netherlands to help warn against this particular fraud campaign. In addition to this, the fact that the dataset

²⁰We have obtained approval from the University of Twente’s Security Officer to include this information in this case study.

contains a rich set of DNS records enabled us to search for malicious domains in a novel way, leveraging specific content of a DNS record. This is complementary to, for example, existing work that leverages lexicographical aspects of domain names (e.g., [174]), another form of analysis that can uncover malicious names. While this case study illustrates the potential of the dataset, this is just the first of many potential applications. In Chapter 9 we will provide an outlook on future research in this direction.

7.7.4 Case Study IV: Impact of the Attack on Dyn

The final case study looks at the impact of the large DDoS attack on Dyn, a DNS hosting provider [93]. This attack took place on October 21, 2016 and was extensively covered in the media. The attack targeted Dyn’s authoritative name server infrastructure. The attack hit Dyn’s infrastructure on the East Coast of the US in particular. This resulted in a number of high-profile Internet companies, e.g., Twitter and PayPal becoming unreachable for several hours.

While the direct effects of the attack have been covered extensively in both the mainstream and tech media, there is a secondary story that got far less coverage. The attack was such a success because the Internet companies that were most affected exclusively used Dyn’s DNS platform. This meant that when that platform became unreachable, effectively companies that exclusively used the platform also became unreachable. With this in mind, we asked ourselves two questions. First: *does the use of Dyn’s services decline in the aftermath of the attack?*, and second: *do organisations affected by the attack switch from exclusive use of Dyn’s DNS service to non-exclusive use?*

To study these questions we analysed the dataset for the .com TLD. We counted the number of domains that exclusively have NS records that point to Dyn name servers (we label this category as *exclusive* Dyn users). We also counted the number of domains that have some NS records that point to Dyn name servers and some that point to name servers from other operators (we label this category as *non-exclusive* Dyn users).

We start by answering the first question. The day after the attack, the number of domains using Dyn (either exclusive or non-exclusive) drops by about 0.7%. The number of domains keeps decreasing until November 13, 2016 at which point the decrease in use has grown to -1.67% relative to the day before the attack. It takes until December 17, 2016 before the number of domains using Dyn has risen back to the level of the day before the attack. All things considered, this shows that there is an impact, but that the damage in terms of reduced customer confidence is limited in time.

The second question, however, shows a much more dramatic effect of the attack. Figure 7.7 shows the fraction of domains the use Dyn’s services ex-

Figure 7.7: Exclusive versus non-exclusive Dyn use in .com around the date of the attack (October 21, 2016)

clusively versus those that use both Dyn’s service and another DNS operator. There is a steep drop in exclusive use after the attack. In absolute numbers over 4000 domains change from exclusive use to non-exclusive use. We also examined the months before the attack, to see if such dramatic changes occur more frequently. In the nine months preceding the attack, the fraction of domains that showed non-exclusive use had steadily been declining. That means that the aftermath of the attack marks a notable trend change. This is also supported by another observation. We looked at the entire dataset to see if there were any other notable trend changes. We found only one, in December 2015, at which time almost 10,000 non-exclusive use domains were added. This, however, was not the result of a shift from exclusive to non-exclusive use, but rather a new customer (a large provider of adult web content) starting to use Dyn’s services non-exclusively for its domains. This means that there is strong evidence that the trend change after the attack is directly related to increased awareness of the vulnerability of using a single provider, and that it indicates a conscious change in behaviour by a large number of Dyn users.

Finally, we also examined the specific set of domains that changed from exclusive to non-exclusive use of Dyn. We retrieved registrant data from the WHOIS service. This shows a wide range of domain owners changing their

behaviour. The list includes a number of large Internet company names, media ventures and gaming platforms, but also industrial enterprises (such as a large manufacturer of earth-moving equipment).

On the one hand, using a single provider who takes extensive security measures, such as DDoS protection, redundant infrastructure, etc., has advantages. On the other hand, it goes against the resilience that the distributed nature of the Internet and the DNS offers. Organisations that exclusively use a single DNS provider, such as Dyn, are effectively putting all their eggs in one basket. If such a provider is brought down by an attack, or, for example, goes bankrupt, this can have far-reaching consequences for the Internet. This case study shows how the data collected by our active measurement infrastructure enables us to study changes in operational behaviour in detail, something that is not possible without reliable longitudinal data from the DNS.

7.8 Data Sharing

We realise that the data we collect is highly valuable for other researchers. Also, it is clear that while Section 7.6.3 illustrates that the impact our measurement has on the global DNS infrastructure is well within reasonable bounds, if lots of researchers were to set up similar infrastructures this would have a significant and possibly disruptive impact on the Internet. This means that we have an obligation to make our data accessible to other researchers.

We share our data in two ways. First, we make part of our data available as open access data. In particular measurement data based on public sources of domain names is released as open data through our website²¹. At the time of writing there are two classes of open access data available. There is a dataset that covers measurements for the Alexa top 1 million list of the most popular web domains; this dataset goes back to January 2016. The second class of open access data contains measurement results for TLDs that release their zone files as open data. At the time of writing this includes the `.se` and `.nu` ccTLDs.

Second, we provide restricted access to the non-public datasets to fellow academic researchers. We have two options for such access. We can extract relevant data for specific research and transfer this to the requester for further processing. This works well if, for example, the extracted data has already been aggregated to a level where we can share the data without violating the contracts under which we have obtained access to the source TLD zones. The other option is to provide access to our analysis infrastructure, and allowing guest researchers to perform their own queries on the datasets. In all cases we first determine what contracts are needed to provide access. In many cases, interested researchers

²¹<https://www.openintel.nl/>

will have to apply for the same TLD zone file access contracts under which we gain access to the source zone files (see Section 7.6.1). In some cases, additional measures may be required, such as a non-disclosure agreement.

7.9 Concluding Remarks

Measuring the DNS is a potent tool for studying the day-to-day evolution of the Internet. In this chapter we presented a unique active measurement infrastructure that collects DNS data for a significant proportion of the global DNS namespace. The chapter discussed the challenges we had to overcome to design and implement such a system. We have shown that our design scales to reliably measure even the largest top-level domain (.com at 127M names). Careful analysis of the traffic generated by the measurement system shows that while it generates a significant amount of traffic, the load on the global DNS infrastructure is at an acceptable level. Measurements started in February 2015, and collect daily data for all domains in .com, .net and .org and a number of other TLDs and ccTLDs (around 60% of all names on the Internet). Since the start of measurements, the system has collected over 1.4 trillion data points totalling over 32TB of compressed data.

As the case studies illustrate, reliable longitudinal DNS datasets enable new forms of research that can help improve the security and stability of the Internet. The two case studies on e-mail show that a significant number of domains now use cloud mail services offered by Google, Microsoft and Yahoo. While – as expected – Google serves the largest number of domains, surprisingly, the use of Microsoft’s Office 365 grows much faster. An investigation of the use of so-called SPF records for combating e-mail forgery also yielded interesting results. While both Google and Microsoft have detailed instructions on how to configure SPF when using their cloud services, use of SPF lags for Google users (at only 38.2%) compared to Microsoft users (over 95.1%).

The third case study showed that access to longitudinal DNS data can be a valuable tool when dealing with real-world security incidents. The data assisted in warning against an active case of CEO Fraud that targeted, among others, higher education and research in The Netherlands. The fourth case study, finally, showed that our data can also reveal remarkable changes in operator behaviour in the aftermath of large DDoS attacks.

Acknowledgements

We would like to thank the security officers of SURFnet and the University of Twente for permission to include the case study on CEO fraud in Section 7.7.3.

Adoption of ECDSA in DNSSEC

The first part of this thesis studied problems in DNSSEC, and identified a way forward to solve these problems: the deployment of elliptic curve signature schemes in DNSSEC. In the previous chapter, a large-scale active measurement platform for the DNS was introduced and its use was illustrated with four case studies. In this chapter, these two things are brought together, as we use results from the active DNS measurements performed by the system introduced in Chapter 7 to study adoption of the Elliptic Curve Digital Signature Algorithm in DNSSEC. The study presented in this chapter was published earlier as [189]; the study has been updated to include an additional four months of measurement results.

8.1 Introduction

In the first part of this thesis we studied the technical problems in DNSSEC. We found that fragmentation (Chapter 3) and amplification (Chapter 4) are both serious problems with DNSSEC. At the root of both problems is the increased message size that DNSSEC requires, because of the inclusion of digital signatures. In Chapter 5, we argued that the root cause of the problematic increase in message size is the choice of RSA as default signature algorithm for DNSSEC. If

current recommendations for RSA key sizes are followed [21], DNSSEC deployments should use 2048-bit RSA keys. This means that every signature requires over 256 bytes of space in a DNS message. Messages often include multiple signatures, quickly inflating their size. Fortunately, as we showed in Chapter 5, there is a way out: the use of elliptic curve signature schemes instead of RSA. Yet when the study discussed in Chapter 5 was performed, there was virtually no adoption of elliptic curve signature schemes for DNSSEC. In the `.com`, `.net` and `.org` TLDs, less than 0.01% used the ECDSA signature scheme, despite that fact that this algorithm had already been standardised for use in DNSSEC in 2012 [94].

There is increasing interest in the use of ECC for DNSSEC. At least one large operator, CloudFlare [46], has thrown its weight behind deployment of ECC in DNSSEC. Second, the IETF has very recently standardised additional ECC algorithms for DNSSEC [173]. Finally, there has been active evangelisation of the use of ECC in DNSSEC in recent conferences frequented by operators, such as ICANN meetings, the IETF and NANOG. This raises the question whether these efforts are paying off.

In this chapter, therefore, we study the adoption of elliptic curve signature schemes by operators. For this study, we use almost two years of data collected by the large-scale active DNS measurement system introduced in Chapter 7. Using this data we:

- provide the first detailed insight into operational adoption of a new cryptographic algorithm in DNSSEC;
- show how adoption of ECC in DNSSEC grows, based on almost two years of longitudinal data;
- show evidence of hurdles to deployment in our datasets;
- illustrate how adoption by a single operator can potentially be a game changer for the use of ECC in DNSSEC.

The study in this chapter brings together the two aspects of this thesis: we use the large-scale active DNS measurement system introduced in Chapter 7 to gauge how the improvements to the security of the DNS itself, that we studied in Chapters 3-6, are adopted in practice.

Figure 8.1: Partial versus full adoption of DNSSEC

8.2 Methodology and Data

8.2.1 Methodology

To study the adoption of ECC, we search for domains that use the currently standardised ECC algorithms, ECDSA P-256 and P-384 [94]. We do not yet consider the recently standardised Ed25519 and Ed448 signature schemes [173]. These are too recent to have seen any adoption and have yet to be implemented in popular DNS software¹.

Algorithm identifiers in three DNSSEC-specific resource record (RR) types signal the use of these algorithms:

- **RRSIG** – the record type for digital signatures.
- **DNSKEY** – the record type for the public keys that are required to validate the signatures in **RRSIG** records.
- **DS** – the *delegation signer* record that resides in the parent zone and signals a *secure delegation* from the parent to the signed zone.

Based on these records, we identify two levels of adoption (see Figure 8.1):

- **Full deployment** – there are signatures (**RRSIG** records), keys (**DNSKEY** records) and a secure delegation (**DS** record(s)) for an ECC algorithm.
- **Partial deployment** – there are signatures and possibly keys for one or more ECC algorithms, but no secure delegation (**DS** record(s)).

Dataset#	TLD	Start date	End date	#Domains*	#Signed*	(%*)
I	.com	Mar. 1, 2015	Feb. 14, 2017	126.6M	0.63M	(0.5%)
	.net			15.1M	0.10M	(0.7%)
	.org			10.5M	0.08M	(0.7%)
II	.nl	Feb. 9, 2016	Feb. 14, 2017	5.7M	2.59M	(45.5%)
III	.gov	February 14, 2017		1083	990	(91.4%)
IV	.at	February 14, 2017		1.3M	< 0.01M	(0.3%)
	.ca			2.5M	< 0.01M	(< 0.1%)
	.dk			1.3M	0.02M	(1.8%)
	.fi			0.4M	< 0.01M	(0.4%)
	.nu			0.3M	0.08M	(26.0%)
	.se			1.4M	0.07M	(48.6%)

*On February 14, 2017

Table 8.1: Datasets

8.2.2 Datasets

As mentioned previously, the data used for this study comes from the large-scale active DNS measurement platform introduced in Chapter 7. Table 8.1 shows which data we used from this platform. The first dataset was selected because it covers the longest time period, and includes the point in time when the statistics for the study in Chapter 5 were collected (which show virtually no adoption of ECDSA). The second dataset covers the .nl ccTLD. This ccTLD has the largest DNSSEC deployment worldwide, and enabled support for ECDSA secure delegations only recently (on March 1, 2016). The third dataset is a one-day snapshot of the .gov TLD reserved for US Government use. The snapshot is based on a publicly available list of .gov domains². Studying .gov is of interest as DNSSEC-signing is mandatory and there are specific guidelines that recommended a switch to ECDSA signing by 2015 [23]. The fourth and final dataset is a new addition compared to the publication on which this chapter is based [189]. This dataset reflects the state of six ccTLDs on the final day of the other datasets and will be used to determine current adoption levels of ECDSA in those six ccTLDs.

8.3 Results

8.3.1 Adoption of ECDSA in .com, .net and .org

First, we look at adoption of ECDSA in .com, .net and .org. These TLDs supported secure delegations for domains signed using ECDSA P-256 and P-384

¹For an overview of DNS software, see Appendix A.

²<https://usgv6-deploymon.antd.nist.gov/cgi-bin/generate-gov>

Figure 8.2: ECDSA P-256 adoption in .com, .net and .org

over the entire period covered by the dataset. From here on, the analysis exclusively focuses on ECDSA P-256, as adoption of ECDSA P-384 is negligible ($< 0.01\%$). The most likely explanation for the low adoption of ECDSA P-384 is that while it is cryptographically stronger, there is not much incentive to use it over ECDSA P-256. Recommendations in [23, 170] indicate that ECDSA P-256 is sufficiently strong for 30+ year use. And as shown in Chapter 5, the reduction of amplification is much more moderate for ECDSA P-384.

Figure 8.2 shows adoption of ECDSA P-256 from October 1, 2015 until the end of the dataset. Adoption of ECDSA P-256 is virtually non-existent until November 10, 2015, when CloudFlare introduces its DNSSEC service [46]. Until April 2016, the use of ECDSA P-256 for DNSSEC-signed domains is almost exclusively limited to domains that use CloudFlare’s DNSSEC service. From early April, however, this changes as the first domains using ECDSA P-256 not operated by CloudFlare start appearing. Almost all of these domains are operated by one company (a media venture). Then, from the second week of October 2016, the number of domains signed with ECDSA P-256 starts rising rapidly. This rise can almost entirely be attributed to a single hosting provider from Norway, Domainnameshop³. Finally, a third, more modest rise can be seen at the beginning of February 2017, which can also be attributed to a single hosting provider from Denmark, called Netsite [58].

A second observation, that stands out, is that a significant number of ECDSA deployments that use CloudFlare’s DNSSEC service are partial deployments. Section 8.2.1 defined a partial deployment as having signatures and keys, but

³<https://domainnameshop.com/?currency=EUR&lang=en>

no secure delegation. Effectively, this means that the domain is DNSSEC-signed, but that the signatures cannot be validated by DNS resolvers, as no chain of trust exists. As a result, even at the end of the dataset over 37.4% of domains signed using CloudFlare’s DNSSEC service cannot be validated because there is no complete chain of trust. While this is not an operational problem (DNS resolution for these domains will still function correctly), it does signal a potential barrier to adoption of ECDSA for DNSSEC. The cause of this problem is the way in which secure delegations are currently created in the DNS. While DNSSEC signing can be done independently by a DNS operator, the creation of a secure delegation has to go through the so-called RRR (Registrant, Registrar, Registry) channel, which is primarily used for domain-name registration. This has two implications. First, registrants (domain name owners) need to request a secure delegation through their domain name registrar when they, or their DNS operator, sign a domain. Second, registrars need to support the creation of secure delegations for domains signed using ECDSA P-256. If either one of these conditions is not fulfilled, then deployment can only be partial. DNS and TLD registry operators recognise the existence of this problem. There is an IETF standard for signalling information for the creation and maintenance of secure delegations directly through the DNS [126]. Furthermore, several parties, including CloudFlare and the ccTLD registry operator for .ca (CIRA), currently have an active Internet draft describing a means to bootstrap the creation of secure delegations [127]. What Figure 8.2 also shows is that this problem does not occur for the two larger deployments that occur in October 2016 and February 2017. The explanation for this is simple: both of these deployments are from full-service providers, that provide both domain name registration and managed DNS services. Such a provider can thus not only sign the zone, but can also directly create the secure delegation once signing has been enabled.

Problems in registering secure delegations are a barrier to deployment of ECDSA P-256 for DNSSEC signing. Yet these problems are not exclusive to ECDSA and also exist for other DNSSEC signing algorithms. Figure 8.3 shows the adoption state of two variants of RSA signing over the full dataset. The figure shows that while deployments for one particular RSA variant (RSA-SHA1-NSEC3, left-hand side of the figure) are almost entirely full deployments, for the other one (RSA-SHA256-NSEC3) partial deployments outnumber full deployments for all three TLDs. Further analysis of these partial deployments shows that they occur for at least fifteen DNS operators that sign more than a thousand domains. While a detailed analysis of the causes of these partial deployments is outside the scope of this chapter, an initial assessment shows the most likely cause again is either a lack of action by the registrant (to create the secure delegation), or a lack of support for secure delegations by the registrar.

Figure 8.3: Adoption of RSA Variants in `.com`, `.net` and `.org`

Figure 8.3 reveals another surprising result. The figure shows a noticeable drop in domains signed using RSA-SHA256-NSEC3 in `.com` and `.net` starting in October 2016. Recall from earlier that Figure 8.2 showed a significant rise in deployment of ECDSA, also in October 2016. Closer investigation shows that the operator responsible for this ECDSA deployment switched from signing with RSA-SHA256-NSEC3 to signing with ECDSA P-256. This is the first evidence of a large-scale algorithm rollover to ECDSA. This is significant, as algorithm rollovers are considered difficult. We revisit algorithm rollover in Section 8.4.

In the publication on which this chapter is based, we also mapped the distribution of signing algorithms used in `.com` [189]. We concluded at that time that use of ECDSA P-256 was still modest, at just 2.3% of signed domains. Since then, the situation has changed a lot, as Figure 8.4 shows. At 14.73% use, the use of ECDSA has almost overtaken the second-most-popular RSA variant, RSA-SHA256-NSEC3. And as we will see in the next subsection, where we zoom in on CloudFlare, ECDSA can potentially become the dominant signing algorithm in `.com`, `.net` and `.org`.

Figure 8.4: Use of DNSSEC Algorithms in `.com` on February 14, 2017

Figure 8.5: New versus existing CloudFlare customers activating DNSSEC

8.3.2 CloudFlare-Specific Metrics

CloudFlare was the first large-scale adopter of ECDSA. As shown in the previous section, CloudFlare was responsible for the largest deployment of ECDSA P-256 in `.com`, `.net` and `.org` for almost a year, from November 2015 until October 2016. Because of this special pioneering status, we specifically looked at developments in this deployment. CloudFlare’s DNSSEC service is an optional feature of its DNS DDoS protection service, which allows customers to protect the name servers for their domain. To use this service, domain owners configure CloudFlare name servers as authoritative name servers for their domain. This means that use of CloudFlare’s DNS service can be detected in the dataset by looking for NS records that point to authoritative name servers operated by CloudFlare [109]. Figure 8.5 shows the percentage of domains operated by CloudFlare that enable DNSSEC on a given day and indicates if these are new or existing CloudFlare customers. In other words: is DNSSEC something that only new customers enable, or do existing customers also turn it on? As a detection criterion, we label domains that were using CloudFlare name servers at least four weeks prior to DNSSEC signing as ‘existing’ customers, and all other domains as ‘new’. Figure 8.5 covers the time period from CloudFlare announcing the service [46] until the end of the dataset. As the figure shows, during the initial weeks of the service being offered, enablements are dominated by existing customers (75%). After that phase, though, the majority of enablements consistently comes from new customers.

As discussed in Section 8.3.1, there are barriers to adoption of DNSSEC-signing using ECDSA P-256. For instance, Figure 8.2 showed that up to 38% of domains signed through CloudFlare do not have a full secure delegation. This is not the entire story, however. Using the analysis previously discussed, we have also tracked the total number of CloudFlare users over time. The analysis shows that only a fraction of these users have enabled DNSSEC. Figure 8.6 shows the

Figure 8.6: Percentage of CloudFlare-operated domains that are signed

TLD	#Domains	#Signed	(%)	%ECDSA	#CloudFlare	%Signed*	%ECDSA*
.com	126.6M	0.63M	(0.50%)	14.73%	1.40M	1.59%	72.5%
.net	15.1M	0.10M	(0.69%)	17.49%	0.15M	1.65%	63.7%
.org	10.5M	0.08M	(0.72%)	17.23%	0.11M	1.73%	63.3%

*Potential deployment including CloudFlare

Table 8.2: CloudFlare’s DNSSEC potential in .com, .net and .org

percentage of domains operated by CloudFlare that are DNSSEC-signed. As the figure shows, while there is growth, the percentage is still small ($< 3\%$). If all CloudFlare users were to fully deploy its DNSSEC service, however, the picture would change dramatically. In fact, as Table 8.2 illustrates, if all domains operated by CloudFlare fully deploy DNSSEC, the number of signed domains more than doubles in all three TLDs and ECDSA P-256 would become the dominant signing algorithm in these TLDs overnight.

8.3.3 Adoption of ECDSA in .nl and Other ccTLDs

The second dataset selected for this study covers the .nl ccTLD, which – unlike .com, .net and .org – did not support secure delegations with ECDSA at the start of the dataset. This allows us to study ECDSA adoption before and after the ccTLD started supporting secure delegations for this algorithm. Figure 8.7 shows partial and full adoption of ECDSA in the .nl ccTLD. Three things stand out. First, initial ECDSA deployment is dominated by CloudFlare, just like in .com, .net and .org. From the middle of June 2016, however, the first domains signed with ECDSA from other operators appear, and toward the end of the dataset these far outnumber domains operated by CloudFlare. Each jump in the number of domains signed with ECDSA is attributable to an individual operator. Apart from the growth spurt starting in October 2016

Figure 8.7: ECDSA P-256 adoption in .nl

(which is the same Norwegian hosting provider observed in .com, .net and .org), all hosting providers responsible for this growth are local, Dutch service providers. Interestingly, the growth in ECDSA adoption coincides with the release of PowerDNS 4.0, an authoritative DNS server implementation popular among Dutch hosting companies. In this PowerDNS release, the default signing algorithm was changed to ECDSA P-256 [131].

The second observation is that deployment of ECDSA is quite modest in .nl, especially compared to the total number of signed domains, at just 0.28%. As we will discuss in Section 8.4, this is unlikely to change in the near future.

Third, we look at partial deployments. All of the partial deployments that exist before .nl starts supporting secure delegations with ECDSA on March 1, 2016 are operated through CloudFlare. As the figure shows, the number of partial deployments operated by CloudFlare does not change much over time, while the number of full deployments operated by CloudFlare steadily grows. Remarkably, over half of the domains that had a partial deployment at the start of the dataset on February 9, 2016 still have a partial deployment more than a year later on February 14, 2017. While we cannot be certain, we speculate that the registrants of these domains enabled CloudFlare’s DNSSEC service at a (much) earlier point in time and simply forgot to request a secure delegation when that became possible for .nl. Also note that from the middle of January 2017, a significant fraction of domains signed with ECDSA by providers other than CloudFlare has a partial deployment. Almost all of these partial deployments can be attributed to a single operator.

Finally, new in this chapter compared to the publication it is based on [189], we looked at deployment figures for ECDSA in six additional ccTLDs. Table 8.3 shows the result. Generally speaking, there are two observations. First, the ccTLDs with higher DNSSEC adoption typically have lower adoption of EC-

	ccTLD					
	.at	.ca	.dk	.fi	.nu	.se
%Signed	0.30%	0.01%	1.81%	0.38%	25.99%	48.59%
%ECDSA P-256	0.99%	41.25%	88.47%	75.13%	14.58%	2.64%

Table 8.3: Adoption of ECDSA in other ccTLDs

DSA. Second, in ccTLDs with low DNSSEC adoption, a large proportion of domains that do deploy DNSSEC use ECDSA. While there is not enough data to conclude that this is a trend, this does appear to suggest that newer DNSSEC deployments tend to favour ECDSA for signing. Future study is required to reveal if this becomes an actual trend.

8.3.4 Adoption of ECDSA in .gov

As Section 8.2.2 mentions, we also measured adoption for the .gov TLD. The reason for doing this is twofold. First, US Federal Government agencies were required to deploy DNSSEC for domains they operate in the .gov TLD, at the latest by the end of 2009 [22]. This means that all .gov domains should be DNSSEC-signed, which, as Table 8.1 in Section 8.2.2 shows, is not the case (91.4% are signed). Second, the US National Institute for Standards and Technology recommends switching to ECDSA [23] for DNSSEC.

Taking these two things into consideration, and also considering the relatively small number of domains for US federal agencies, one might expect to see significant ECDSA adoption in .gov. Disappointingly, however, in February 2017 not a single .gov domain uses ECDSA for signing, almost two years after the date that NIST recommends switching to this algorithm.

And the worries do not stop there. The same NIST publication [23] states that zones that sign using RSA and that use the deprecated SHA-1 hash algorithm should migrate to using SHA-256. The reason for this is that there are serious concerns about the security of the SHA-1 hash algorithm. Yet in February 2017, almost half of .gov domains⁴ are still exclusively signed using RSA with the SHA-1 hash algorithm. The final concern is about key sizes. As discussed earlier in Chapter 5, NIST recommends deprecating 1024-bit RSA keys in favour of 2048-bit RSA keys. Yet again, in February 2017 over 8% of .gov domains exclusively use 1024-bit RSA signing keys. And even worse, six .gov domains use 512-bit RSA keys, a key size that is considered highly insecure and trivial to break [178].

⁴Ironically including `nist.gov`.

Scheme	TLD or ccTLD									
	.com	.net	.org	.at	.ca	.dk	.fi	.nl	.nu	.se
KSK/ZSK	97.7%	98.4%	98.4%	74.0%	97.4%	47.8%	99.5%	53.4%	85.4%	99.1%
CSK	2.3%	1.6%	1.6%	26.0%	2.6%	52.2%	0.5%	46.6%	14.6%	0.9%

Table 8.4: ‘Traditional’ KSK/ZSK key management versus single key CSK

Given the sorry state of DNSSEC deployment in the .gov TLD, it is clear there is a need for an overhaul. US federal agencies, including NIST itself, would do well to follow NIST’s recommendations and to update their DNSSEC deployments to sign using ECDSA.

8.3.5 Key Management Schemes in ECDSA Deployments

Finally, we have examined whether zones that have been signed using ECDSA use the ‘traditional’ split key model with a KSK and ZSK, or whether they use a Combined Signing Key (CSK) (the advantages of which we explained in Chapter 5, Section 5.4.4). We examined this across all datasets shown in Table 8.1, for the last day included in the datasets (February 14, 2017).

Table 8.4 shows the result. As the table shows, there are big differences between the TLDs included in the study. Closer examination shows that there is insufficient evidence of any trends of operators favouring one key management model over the other. In the four ccTLDs with a significant fraction of ECDSA-signed domains using the CSK key management model, this fraction can be attributed to a single or only a few operators. Because of the additional benefit in terms of reduction of amplification potential of using the CSK model (§5.4.4), it is worthwhile to monitor future development of adoption of this key management model.

8.4 Recommendations for Operators

The most important takeaway for operators from the study presented in this chapter, is that the adoption of new algorithms in the complex DNSSEC ecosystem takes a lot of time. The uptake of ECDSA did not start to grow significantly until late 2016, four years after it was standardised for use in DNSSEC [94]. This is true throughout the DNSSEC ecosystem; in related work, Huston and Michaelson [102] show that support for ECDSA among DNS resolver operators is also not yet universal. It is much higher than on the signing side, though, with 82% of validating resolvers supporting ECDSA. We note that the Internet community is trying to learn from this slow deployment as input for the roll-out of future standards [218].

Domain name operators wishing to secure their domains with DNSSEC should be mindful of deployment hurdles. If they want to follow current advice to sign using ECDSA (Chapters 5 and 6), they need to take care when selecting a DNS operator and domain name registrar. These need to support DNSSEC in full, including the newly standardised ECC algorithms.

Finally, there may also be hard times ahead for those 3% of domain name owners that have already fully deployed DNSSEC using RSA. The transition from RSA to ECC signing algorithms is very complex. This so-called algorithm rollover [121] requires concurrently signing a zone with both algorithms. It is highly likely that this will require a manual intervention by operators, since only one recently released DNSSEC implementation supports automated algorithm rollovers [5]. If it indeed becomes best practice to use ECC algorithms, then these operators are in for a hard and potentially costly transition. This is especially true for `.nl`, with over 2.5M signed domains using RSA. The Dutch may once again be confronted with their own *‘law of the handicap of a headstart’*⁵. The fact that such an algorithm rollover can be performed successfully, however, has been demonstrated at scale at least once. As discussed in Section 8.3.1, the Norwegian operator Domainnameshop has switched a large number of domains over from RSA to ECDSA in late 2016. A closer investigation of the data surrounding that switch shows that this was an actual algorithm rollover. That is: domains were concurrently signed with both the old and the new algorithm for some time, and DNSSEC was enabled during the entire period.

8.5 Concluding Remarks

In this chapter we analysed the deployment of elliptic curve digital signature algorithms for DNSSEC in ten top-level domains. In general, deployment of ECDSA is on the rise after a slow start at the end of 2015. The largest deployment (in absolute numbers), in `.com`, covers about 14.7% of signed domains. And there is significant potential for growth, especially in `.com`, `.net` and `.org`. Universal deployment of DNSSEC by CloudFlare users alone can more than double the number of DNSSEC-signed domains in `.com`, `.net` and `.org`, and make ECDSA P-256 the dominant signing algorithm overnight.

While certainly not a niche technology, DNSSEC deployment is far from ubiquitous, with an estimated 3% of domains worldwide deploying DNSSEC. This means – given the benefits in terms of security and stability that elliptic curve signature algorithms offer – that there is also an opportunity to follow a ‘first time right approach’, and to use ECC for the 97% of domains that have not yet deployed DNSSEC.

⁵https://en.wikipedia.org/wiki/Law_of_the_handicap_of_a_head_start

The final takeaway from this chapter is that the study it presents also clearly demonstrates the richness of the datasets collected by the large-scale active DNS measurement system introduced in Chapter 7. Without the datasets collected by that system, it would not have been possible to present this longitudinal study of the adoption of new cryptographic algorithms in DNSSEC. As we have shown, there are valuable lessons to be learned from such a longitudinal study, that can inform, for example, future introductions of new cryptographic algorithms in DNSSEC.

Conclusions

In this final chapter, we draw conclusions from the research presented in this thesis. We start by outlining our main conclusions, followed by an in-depth discussion of the conclusions relating to the research questions set out in Chapter 1. We end the chapter with an outlook on future research directions that build on, and continue the work presented in this thesis.

9.1 Main Conclusions

In Chapter 1 we set two goals for this thesis. The first goal we set related to problems in DNSSEC and was:

Goal 1: *to study technical problems in DNSSEC that limit its large-scale deployment, and to identify and validate solutions to these problems.*

The two key terms in this goal were that we focused on *technical* problems and that these problems should be so severe that they would *limit* large-scale deployment of DNSSEC. In Chapters 3 and 4 of this thesis, we identified two technical problems with that kind of severity. We showed that failure to deal with fragmented DNS responses can cause up to 10% of DNS resolvers on the Internet to suffer from problems resolving DNSSEC-signed domains. Therefore,

DNSSEC-signed domains can become unreachable for users of these resolvers. This is an unacceptable risk for large operators such as the Googles of this world, who, as we mentioned, were hesitant to deploy another technology, IPv6, because this could cause reachability problems for only 0.1% of their users [47].

The second problem is amplification: DNS can be abused in amplification DDoS attacks, and since the digital signatures that DNSSEC uses make DNS messages larger, this increases the potential power of such attacks. While amplification was an oft-cited criticism of DNSSEC, no ground truth about the impact existed. In this thesis we establish this ground truth and show that DNSSEC can potentially be abused for devastating DDoS attacks. This is also a serious problem, as no domain operator will want to see their domain abused in attacks, as this can lead to reputation damage.

The second part of our first research goal was to come up with validated solutions to the problems we identified. In this thesis, we proved that the use of elliptic curve signature algorithms instead of RSA in DNSSEC fully solves the fragmentation problem, and significantly reduces the amplification potential in DNSSEC. Yet we also found one major disadvantage to the use of elliptic curve cryptography in DNSSEC: slower signature validation. By studying this disadvantage in more detail, however, we were able to show that the impact of slower signature validation on the global DNS infrastructure remains well within acceptable limits. This paves the way for deploying elliptic curve signature algorithms in DNSSEC and takes away serious problems that have limited the deployment of DNSSEC. In the previous chapter we have shown that the deployment of ECC in DNSSEC has actually started in earnest.

Next, we set a second research goal for this thesis, relating to the important role the DNS plays in the security and stability of Internet services:

Goal 2: *to develop ways to perform large-scale long-term active measurements of the data contained in the DNS.*

We discussed that there are existing methods to collect large-scale DNS data based on passive measurements. Yet we also showed that these kinds of datasets have limitations that can be overcome by using active measurements. No active measurement system, however, existed, with a scale comparable to current passive DNS measurement deployments. In this thesis we have introduced OpenINTEL, a large-scale long-term active measurement platform for the DNS. We have shown that through careful design, it is possible to perform daily active measurements of a significant portion of the global DNS namespace, including even the largest TLD (.com, with 127M names). We also showed that we are able to do this without imposing an undue burden on the global DNS infrastructure. Through a series of case studies we illustrated the applications of actively collected DNS data for longitudinal studies of protocol and security

measure adoption, as well as for studying and resolving real-world security incidents. And because we recognise the value of the unique datasets collected by this measurement system, we have taken steps to ensure access to this data for other academic researchers (§7.8), through an open access programme (for public datasets) and by providing access under NDA (for non-public datasets).

9.2 Revisiting the Research Questions

To provide a more detailed view on the results of the work in this thesis, this section revisits each of the research questions defined in Chapter 1. The section is split in two. The first part looks at the research questions relating to the first goal of the thesis, the second part at the questions relating to the second goal.

9.2.1 Research Questions Relating to Goal 1

The first research question relating to the first goal was:

RQ 1: *which technical problems exist in DNSSEC, that are severe enough to limit large-scale deployment of the technology? And to what extent do these problems occur in practice?*

We studied this research question in Chapters 3 and 4. First, in Chapter 3 we studied problems caused by IP fragmentation of DNS responses. Since, as we explained in Chapter 2, DNSSEC makes DNS responses larger due to the inclusion of digital signatures, this also increases the likelihood that these responses exceed the Maximum Transmission Unit (MTU) size and are fragmented at the IP layer. In §3.2 we found that up to 36% of DNSSEC responses observed during the measurements we conducted exceeded the minimum MTU of IPv6 and would thus potentially be fragmented. We also found that up to 10% of DNS resolvers on the Internet may have problems receiving fragmented DNS responses. Furthermore, more than half (57%) of DNS resolvers observed in the measurements received a fragmented response at some point. In a worst case scenario, problems receiving fragmented responses can lead to unreachability of DNSSEC-signed domains from the perspective of clients behind DNS resolvers that experience these problems. Even worse is that DNS resolvers may experience these problems for DNSSEC-signed domains regardless of whether or not these resolvers perform DNSSEC validation. This is because almost all modern DNS resolver implementations will, by default, ask for DNSSEC information in a query, even if they do not validate DNSSEC signatures. In the measurements we observed that on average 75% of resolvers request DNSSEC information¹.

¹Again this does not mean they validate this data. Studies suggest that only a small fraction of resolvers actually validate DNSSEC signatures [45, 82, 101, 102, 205, 220].

This then, is a serious problem: as a domain operator, if you sign your domain with DNSSEC, it can suffer from serious reachability problems for a substantial fraction of resolvers. This is a major barrier to adoption: becoming unreachable for such a large fraction of resolvers is unacceptable for all but pioneering domain operators. Fortunately, our results also show that as a domain operator it is possible to take effective and simple measures to mitigate this problem, by actively limiting the response size on authoritative name servers (§3.5.3). In addition to this, since the study in Chapter 3 was published, implementers of DNS resolver software have significantly improved fallback behaviour to deal with non-arrival of fragmented responses. The EDNS0 specification [52] has also been updated, to explicitly include the risks of intermediate systems blocking fragmented responses when implementing and configuring systems that use EDNS0 (a necessary building block for DNSSEC).

The second problem in DNSSEC we studied is the increase in amplification potential. That is: DNSSEC's larger responses can make DNSSEC-signed domains attractive to abuse in amplification denial-of-service attacks. Chapter 4 established ground truth on how DNSSEC actually increases the potential for amplification attacks. We measured response sizes for 70% (2.5 million) of DNSSEC-signed domains in operation in 2014. As a baseline, we also measured a random sample of equal size of non-DNSSEC signed domains, and compared the results. We found that – on average – DNSSEC brings a six- to twelve-fold increase in amplification potential (§4.5.1), for the ANY query type (with outliers reaching amplification factors up to 179×). The ANY query type is commonly abused in DNS amplification attacks [165]. For this reason, the IETF DNS operations working group is working on a clarification of the DNS specifications that allows DNS operators to curb the response size to ANY queries [6]. We also found that the most common DNS queries, A and AAAA queries (§4.5.3), as well as other common query types (NS, MX, TXT, §4.5.2) show an increase in amplification potential for DNSSEC-signed domains, as was to be expected. The good news is, however, that while there is an increase, this is not necessarily problematic. We defined what can be achieved with the 'classic' DNS protocol from 1983 [141] as an acceptable upper limit to amplification. All the common DNS query types have amplification factors below this limit for the DNSSEC-signed domains we measured. This means that in principle, DNSSEC deployment does not have to cause a problematic increase in amplification, provided that measures are taken against ANY queries. Only one caveat remained: the DNSSEC protocol uses public key cryptography. This requires distribution of public key material so digital signatures can be validated. DNSSEC uses the DNSKEY query type for this. In our study we found that this particular query type has significant amplification potential on its own, with on average 38% of DNSKEY responses for the domains we measured exceeding the acceptable upper limit we defined

(§4.5.2). The amplification mainly depends on the size and number of keys used to sign the DNS zone. Since the `DNSKEY` query type is integral to the DNSSEC protocol, this poses a problem, especially since key sizes are only likely to grow in the future, to increase cryptographic security. While we discussed a number of mitigation approaches to amplification attacks in §4.6, the only structural solution is to reduce the growth in DNS message size that DNSSEC imposes.

Given the serious technical problems in DNSSEC, that we found while answering RQ 1, we revisit the second research question:

***RQ 2:** if answering RQ 1 identified technical problems in DNSSEC, what are effective solutions to these problems?*

In Chapter 4, we already suggested that using different signature schemes, based on elliptic curve cryptography could be used to reduce DNSSEC message sizes. We studied this in Chapter 5. Using the same methodology and input datasets as in Chapters 3 and 4, we analysed what impact a change in cryptographic algorithm has on fragmentation and amplification. First we measured to what extent fragmentation and amplification are still a problem (§5.2). We find that, even after taking mitigation measures suggested in §3.5.3, fragmentation of responses still occurs (albeit far less frequently). As for amplification, the picture did not change significantly between the measurements for Chapter 4 (in 2014) and for Chapter 5 (a year later, in 2015). We then investigated six scenarios, covering both established elliptic curve algorithms [94] and algorithms that have only recently been standardised for use in DNSSEC [173]. In addition to this, the scenarios also cover two DNSSEC policies, one in which the ‘traditional’ split into a Key Signing Key (KSK) and Zone Signing Key (ZSK) is used, and a simpler one in which a single combined signing key (CSK) is used. The results of these scenarios are highly encouraging. Even in the most conservative scenario (using ECDSA P-384 with a ‘traditional’ KSK/ZSK signing policy), fragmentation is reduced to negligible levels (§5.4.2). Choosing a simpler signing policy with a single key, combined with a 256-bit ECC curve can even reduce message sizes such that even `DNSKEY` responses would fit in a ‘classic’ 512-byte DNS message. The analysis of the scenarios also shows that adoption of ECC would result in a significant reduction in amplification potential (§5.4.3). Only responses to `ANY` queries would still yield amplification levels above the acceptable upper limit we defined in §4.3.2. For all other query types, including the `DNSKEY` query, the amplification stays below this limit, even for the most conservative ECC scenario. While there are still a number of barriers to adoption of ECC in DNSSEC, which we discuss in Chapter 5, these are mostly transient. There is one exception to this: the validation of ECC signatures can be up to almost two orders of magnitude slower than RSA signatures, which may have an impact on validating DNS resolvers (§5.5.1).

This means that, while elliptic curve cryptography is a promising solution to the technical problems in DNSSEC, one question remained: can the slower signature validation for ECC introduce new problems, by imposing a burden on validating DNS resolvers? This brings us to the third research question:

***RQ 3:** what is the impact of the solutions identified as responses to RQ 2 on the global DNS infrastructure?*

We studied this question in Chapter 6. We wanted to isolate the impact of signature validation on the CPU use of a validating DNS resolver. To do this, we performed an in-depth analysis of all factors in the operation of a validating resolver that determine the number of signature validations it has to perform (§6.4.1). Based on this analysis, we created a linear model that predicts the expected number of signature validations required, given the number of queries the validating resolvers sends upstream to authoritative name servers (§6.4.2). We validated the model using real-world data from four sources in two networks and for two popular open source DNS resolver implementations. In order to be able to estimate CPU requirements, we performed a set of benchmarks of ECC signature algorithms, using commonly deployed cryptographic library implementations. Using the model and the benchmarks, we then evaluated two scenarios for DNSSEC deployment and estimated the CPU requirements due to signature validation for both of these. The first scenario was to switch all domains signed at the time of the study to an elliptic curve signature algorithm. In §6.6.3 we found that this would require in the order of 124 to 224 signature validations per second (depending on the DNS resolver software implementation). Regardless of which elliptic curve algorithm is chosen, the CPU requirements for those numbers of validations can easily be handled by a single modern CPU core. The second scenario modelled a growth to universal (100%) DNSSEC deployment across the whole DNS. We used a worst-case scenario in which popular domains (i.e., for which the most queries are seen) deploy DNSSEC first. For this scenario we also found favourable results. In almost all cases, the CPU requirements for signature validation can easily be handled on a single modern CPU core. There is one exception; if the most CPU-heavy algorithm, ECDSA P-384, is used for signing, and the BIND DNS resolver implementation is used, then the load due to signature validations can no longer be handled on a single CPU core. The reason for this is that BIND validates up to three times as many signatures given the same workload as the other implementation we evaluated (Unbound). Given that the current trend appears to be that operators deploying ECC-based signature algorithms choose ECDSA P-256, this limitation with BIND is not an insurmountable problem.

Now that we have shown that elliptic curve signature algorithms can be deployed without imposing an unacceptable burden on the global DNS infrastructure, one question remains:

***RQ 4:** to what extent are the solutions identified in RQ 2 adopted in practice?*

Chapter 8 studies the adoption of the Elliptic Curve Digital Signature Algorithm (ECDSA) in practice, using the datasets collected by the measurement system discussed in Chapter 7. We analysed almost two years worth of data for the `.com`, `.net` and `.org` top-level domains and over a year of data for the `.nl` ccTLD. In addition to this, we used single day measurements for six additional TLDs. In our data, we found that adoption of ECDSA was virtually non-existent in the analysed top-level domains until November 2015 (§8.3). In that month, CloudFlare announced they would be offering DNSSEC as part of their service and would be using ECDSA as signing algorithm [46]. This immediately lead to an uptake in ECDSA for DNSSEC, and until mid-2016 ECDSA adoption is dominated by CloudFlare. From that point on, however, our data shows new players deploying DNSSEC with ECDSA, and towards the end of the study these new players far outnumber CloudFlare in terms of the number of domains that deploy ECDSA. Another key finding of this study is that we see clear evidence of hurdles to adoption of ECDSA as signing algorithm. Specifically, the need in DNSSEC to signal secure delegations through the so-called *Registrant-Registrar-Registry* (RRR) channel leads to problems. If either registrar or registry does not support ECDSA, signing a zone with ECDSA is possible but pointless, since without a secure delegation in the parent zone of the domain, signatures cannot be validated. In our analysis we clearly see that in cases where the whole RRR channel is controlled by a single entity (e.g., a hosting provider), full deployment of DNSSEC with ECDSA proceeds without problems. But in the case of CloudFlare, domain owners have to signal the secure delegation through their registrar themselves, which does not happen in over 40% of cases of domains signed using CloudFlare’s DNSSEC service.

9.2.2 Research Questions Relating to Goal 2

In this subsection we revisit the research questions relating to the second goal of this thesis: to establish the means to perform large-scale longitudinal active DNS measurements. The first question, obviously, was whether it is possible at all to perform such measurements:

***RQ 5:** is it technically possible to perform an active measurement of a large portion of the global DNS namespace at regular, daily intervals? Can such a measurement scale to encompass even the largest top-level domain (`.com`)?*

In §7.3 we introduced a set of goals we wanted to achieve with regards to active DNS measurements. We then outlined the challenges we faced while meeting these goals. The biggest challenge was measuring the largest TLD, `.com` with 127M names in it. By first experimenting at a smaller scale – measuring all names in the smaller `.org` TLD – and extrapolating from that, we were able to establish that it is feasible to measure at the scale of `.com`. But while we have now shown that it is possible to measure at this scale, we also need to assess the impact of such a measurement, which brings us to the next question:

***RQ 6:** what is the performance impact of a large-scale daily active measurement of significant portions of the global DNS namespace?*

As discussed in §7.6, we did a performance analysis of the active DNS measurement system that we designed. We showed that the measurement system generates a decidedly non-trivial amount of traffic, exceeding the DNS traffic load generated by a medium-sized network operator. Yet we were also able to show that even the top talkers – that is: the systems that receive the most traffic from the measurement system – do not receive excessive amounts of traffic. Both independent verification based on public statistics, and private communication with the operator of the large `.com` and `.net` TLDs, whose servers receive a significant amount of the daily traffic generated by our measurement system, show that these servers do not suffer from excessive load due to our measurement. In addition to verifying the impact of our measurement, we have also put in place operational measures that alert us if the measurement exceeds certain limits. This allows us to take action if we run the risk of generating excessive amounts of traffic to the global DNS infrastructure. Note that we have also documented our measurement practices in Appendix C.

The next question was how to deal with the large datasets that result from measuring large portions of the DNS:

***RQ 7:** how do we efficiently store and analyse large-scale DNS datasets that are collected over long periods of time?*

While discussing our design decisions, we argued that the de-facto standard for analysing datasets of this magnitude is to use the tools in the Hadoop ecosystem. To this end, we ensured that the datasets resulting from the measurement were directly suitable for this toolchain (§7.4.3). We store all measurement data in the Apache Avro file format, which is well-supported in the Hadoop ecosystem. In addition to this, to improve analysis performance, we convert this data to the columnar Parquet storage format *in situ* on our Hadoop cluster.

Now we have shown that it is possible to collect, store and analyse large-scale active DNS measurement data, we arrive at the final research question:

RQ 8: *can we demonstrate uses of large-scale longitudinal active DNS datasets with real-world examples?*

Through five case studies we provided real world examples of how large-scale active DNS measurements can be used for both longitudinal research on the adoption of protocols and security measures, and how this type of data can also have a direct operational impact during security incidents.

The first two studies (§7.7.1 and §7.7.2) looked at the use of cloud e-mail providers, specifically Google, Microsoft Office 365 and Yahoo. We showed longitudinal developments in the use of these e-mail services, clearly demonstrating significant growth. That the use of these commercial cloud service providers has advantages became particularly clear when we examined how domains that use these services are protected against e-mail forgery using the Sender Policy Framework (SPF) [114]. An impressive 95.1% of .com domains using Microsoft Office 365 had deployed this measure, compared to a general deployment in .com of just 30.3%. The third case study (§7.7.3) showed how the datasets can be used in real-world security incidents, in this case to help mitigate a so-called ‘CEO fraud’ phishing attack. The fourth case study (§7.7.4) looked at changes in DNS operations after the large DDoS attack on service provider Dyn in October 2016. Analysis showed that after the attack, a significant number of customers who had until that point exclusively used Dyn’s services changed their behaviour. They spread their DNS operations to multiple providers in the aftermath of the attack, and thus reduced their vulnerability to an attack on Dyn. This case study illustrated the risk of single points of failure in core Internet infrastructures. As the case study demonstrates, our longitudinal DNS datasets are well suited for studying such single points of failure. Lastly, as discussed in the previous subsection on the first goal of this thesis, in Chapter 8 we used the active DNS datasets to perform a longitudinal study of the adoption of new cryptographic algorithms in DNSSEC. That study illustrates the value of the datasets for studying the long-term evolution of the Internet.

With an estimated 1.4 trillion ($1.4 \cdot 10^{12}$) data points covering almost 60% of the global DNS namespace at the time of writing of this thesis, our actively collected longitudinal DNS datasets are a valuable resource for network and network security research. For this reason, we have ensured that academic researchers can access this data for their own research, partly as open access data (in case of public datasets), and partly under NDA. The datasets collected by our active measurement platform have already been used for third party studies, for example in a case study included in the national Cyber Security Assessment for the Netherlands [145].

9.3 Directions for Future Research

In the final section of this chapter we set out directions for future research. These are structured around three themes: improving our understanding of DNSSEC in practice, continued insight into adoption of elliptic curve signature schemes and improving general Internet security by measuring the DNS.

9.3.1 Improving Understanding of DNSSEC in Practice

It has been almost a decade since DNSSEC deployment on the Internet took off in earnest, after the cache poisoning vulnerability was published in 2008 [112]. Over this period several best practice documents have been published by the IETF [88, 120, 121, 143, 209] with evolving insights on how to manage a DNSSEC-signed domain. In addition to this, there have been myriads of HOWTO's, blogposts and other documents to guide operators of DNS resolvers and operators of authoritative name servers in deploying DNSSEC. In order to improve deployment practices it is important to understand how operators deal with DNSSEC in actual practice. In other words, to answer the question: how are best practice recommendations used in practice?

Based on this question, we identify the following directions for future research:

- **Revisiting fragmentation** – problems with DNSSEC and IP fragmentation were discussed in Chapter 3. Since the study in that chapter was published, standards and best practices have been updated and fallback behaviour in DNS resolver implementations has been improved. It is worthwhile to regularly revisit the study discussed in Chapter 3, although this may prove to be increasingly hard as, e.g., elliptic curve signature algorithms are deployed and message sizes are reduced as a consequence. What is easier to measure is to what extent mitigation measures are applied by operators and resolver implementations. As discussed in Chapter 3, DNS resolvers advertise the maximum response size they are capable of receiving in EDNS0-enabled queries. These advertisements can be measured by studying traffic on authoritative name servers. Both conscious changes by operators and fallback behaviour in DNS resolver software will lead to lower maximum response sizes being advertised by resolvers. A longitudinal study based on data recorded on authoritative name servers will be able to show if there is evidence of changing behaviour in EDNS0 maximum response size advertisements. Several suitable longitudinal datasets exist to support such a study, for example the ENTRADA

system operated by SIDN² [216] and the ‘Day-in-the-Life’ datasets made available on an annual basis by DNS-OARC [59]. Similarly, it is possible to measure mitigation actions taken by operators of authoritative name servers for DNSSEC-signed domains. Using a similar methodology to that employed for the study into DNSSEC amplification in Chapter 4, it is possible to perform a census of EDNS0 maximum response sizes advertised by authoritative name servers. Additionally, it is possible to detect whether or not these authoritative servers have been configured to return minimal responses. Such studies can help improve best practice documentation and can also help gain an understanding how long it takes for such best practices to see widespread adoption.

- **Uptake of validation** – in Chapter 2, we discussed a number of past and ongoing studies that have looked at uptake of DNSSEC validation by DNS resolver operators. As we noted there, past studies mostly provide snapshots of the state of validation uptake, except for the long-term measurement run by Huston and Michaelson at APNIC [101]. This latter measurement is cited very frequently as a source of ground truth on DNSSEC validation and is updated regularly (e.g., it has been used to gauge support for validation of ECDSA signatures [102]). The operators of this measurement also claim that it provides a view on DNSSEC validation from the perspective of the user, i.e., how many users are protected by DNSSEC validation? But there are problems with this measurement. These are mainly caused by the measurement being based on advertisements. This approach introduces various forms of bias, e.g., because of advertisement placement strategies employed by the operator of the advertising network. Lian et al. [132] employed a similar approach as APNIC and expressly mention that bias is a serious concern with this type of measurement approach. The operators of the APNIC measurement have introduced compensation factors to deal with bias, but it is unclear how stable these compensation factors are and what the basis for compensation is. Another problem with measuring from the user’s perspective is that it will not catch a shift in validation behaviour for resolvers with smaller user populations, as these will in essence disappear in the masses. This is highly relevant, if we consider that, for example, a number of popular Linux distributions have enabled DNSSEC validation by default in newer releases of their distribution. We expect to see an effect of this as these distributions are deployed to replace older installations. For this reason, we argue in favour of a future longitudinal study from the perspective of authoritative name servers that serve DNSSEC-signed domains to monitor

²The ccTLD registry for the .nl domain.

uptake of validation. Suitable datasets for such studies include the SIDN ENTRADA platform mentioned earlier [216]; we note that SIDN already publishes some statistics based on this platform that deal with DNSSEC validation³.

- **DNSSEC signing** – the best practices for DNSSEC signing [120, 121, 143] suggest using certain policies when signing zones. Furthermore, most DNSSEC signing software comes with a set of default policies for signing domains. Existing work has looked at snapshots of the state of DNSSEC to identify, for instance, key types and sizes used [203]. We intend to go a step further and perform a longitudinal study of DNSSEC, based on data collected by the OpenINTEL active DNS measurement system discussed in Chapter 7. The goal of this study will be to understand the technical policy choices made in practice (such as signature refresh intervals, key rollover policies, signature timing jitter, ...). We do not just want to limit ourselves to describing policies used in practice, though. Our aim will also be to see if some policies are more robust than others (that is: lead to fewer validation errors), and even to see if we can fingerprint the DNSSEC signing software that was used to sign a domain.

9.3.2 Future Adoption of Elliptic Curve Algorithms

In Chapter 8 we studied the uptake of the ECDSA signing algorithm in DNSSEC. Yet there are two other elliptic curve signature schemes that have only recently been standardised for use in DNSSEC [173]. We have started measuring adoption of these algorithms as well. This provides an opportunity to monitor adoption of a new algorithm from the very first moment it became available for use in DNSSEC.

Additionally, we also noted that migrating a domain that is already DNSSEC-signed using an RSA signature scheme to an elliptic curve-based signature scheme can be a complex process. This so-called algorithm rollover (described in [121]) requires operators to concurrently sign their domain with the two algorithms according to very specific rules. As discussed in Chapter 8, we have already seen evidence of at least one large operator performing such a rollover. Operators, however, have another option: they can briefly disable DNSSEC, and then re-enable it with a new signing algorithm, foregoing the complex and risky algorithm rollover at the cost of temporarily losing the protection DNSSEC offers. We intend to use data collected by the OpenINTEL active measurement platform to monitor for domains switching from RSA to ECC signing schemes, and to study what approach operators take.

³<https://stats.sidnlabs.nl/#/dnssec>

9.3.3 Improving Internet Security by Measuring the DNS

Although we already illustrated some of the security applications of the active DNS datasets collected by the measurement system discussed in Chapter 7, this type of data has many other applications. Traditionally, passively collected DNS data is used in security research. The body of work around this is extensive; surveying it is outside the scope of this thesis, but a good overview can be found in work by Dodopoulos [60]. We foresee the following future security research directions, using the data that is collected by the OpenINTEL active DNS measurement platform introduced in this thesis:

- **Detecting domains for malicious purposes using active measurements** – just like in legitimate Internet services, the DNS plays a role in malicious activities. Either as an instrument, e.g., in amplification DDoS attacks, or as an intermediary, e.g., in the sending of spam or phishing e-mails. We have already started research to find domains crafted for these two purposes, and find that, for example, we sometimes observe new domains crafted for sending spam weeks before they appear on blacklists. This corresponds with findings reported in recent work by Kountouras et al. [122]. We intend to continue this work and to use additional sources of data, such as data from the large-scale e-mail filtering system operated by SURFnet⁴ to gauge the viability of early detection of domains set up for abuse using our actively collected DNS data.
- **Gap analysis between active and passive DNS measurements** – we already discussed the main difference between passively and actively collecting DNS data. Passive DNS will typically only capture data for domains in which there is an actual interest from clients. This means that there is also potential information in the gap in data between our active DNS measurements and data that is captured by passive DNS systems. There are likely to be domains for which our active measurements have data points, while these domains do not show up (yet) in passive DNS datasets. This raises the question why these domains are there. We hypothesize that at least some of these domains may be set up for malicious purposes and have not shown up in passive DNS sensors because they are not yet being used. If this is the case, this opens a window of opportunity to take action against these domains and their operators. This certainly warrants further study.

⁴<https://www.surf.nl/en/services-and-products/surfmailfilter/index.html>

DNS Server Implementations

Table A.1 below lists common open and closed source DNS implementations. The first column gives the name of the implementation. The second and third column indicate whether the implementation supports running as an authoritative name server or as a recursive caching name server (resolver). The last column gives a short description and a URL linking to more information.

	Name	Auth.	Res.	Remarks
Open Source	BIND	Yes	Yes	BIND was the first DNS server implementation, and is still the most widely used name server software. https://www.isc.org/downloads/bind/
	NSD	Yes	No	NSD implements a stand-alone authoritative-only DNS server. https://www.nlnetlabs.nl/projects/nsd/
	Unbound	No	Yes	Unbound was built from the ground up as a DNSSEC validating DNS resolver implementation. https://unbound.net/
	Knot DNS	Yes	No	Knot DNS is developed by the registry for the Czech ccTLD CZ.NIC as a DNSSEC-capable authoritative name server. https://www.knot-dns.cz/
	Knot DNS Resolver	No	Yes	Knot DNS Resolver is a DNSSEC validating DNS resolver implementation from the Knot DNS team. https://www.knot-resolver.cz/
	PowerDNS	Yes	No	PowerDNS is an authoritative name server implementation built to run on top of databases. https://www.powerdns.com/
	PowerDNS Recursor	No	Yes	PowerDNS Recursor is a DNSSEC validating DNS resolver implementation from the PowerDNS team. https://www.powerdns.com/
Closed Source	Microsoft DNS	Yes	Yes	Microsoft DNS is the name of the DNS implementation included in the Windows Server operating system. https://technet.microsoft.com/en-us/library/bb629410.aspx
	Infoblox	Yes	Yes	Infoblox is a vendor of IP Address Management (IPAM) systems. They provide a popular DNS solution for enterprises. https://www.infoblox.com/products/dns/
	Nominum Vantio AuthServe	Yes	No	DNSSEC-capable authoritative-only DNS implementation for enterprise applications. http://www.nominum.com/product/vantio-authserve/
	Nominum Vantio CacheServe	No	Yes	Resolver-only DNS implementation for enterprise applications. http://www.nominum.com/product/caching-dns/

Table A.1: Popular DNS server implementations that support DNSSEC

Open Dataset Management

B.1 Introduction

The practice of science is often likened to ‘*standing on the shoulders of giants*’, where no work is published in isolation and builds on previous discoveries and insights. The work in this thesis is no exception, it has benefited extensively from earlier studies. One of the ways in which scientific work can benefit from earlier studies is by using open access datasets published as part of this earlier work. It is becoming common practice to release research results as open access. The Netherlands, for example, has launched a National Plan for Open Science, undersigned by a broad consortium of national players in science, including the Ministry for Education, Culture and Science, funding bodies, e-science infrastructure partners, universities and major research institutes [198]. Another example is the long-standing practice of the ACM SIGCOMM Internet Measurement Conference to award a separate prize to authors of papers that release the data that accompanies their paper as open access data¹.

A second major consideration when collecting datasets is to preserve the integrity of scientific results and to facilitate reproducibility of results. Careful curation of results allows other researchers to independently validate the results of research.

For these two reasons, we have strived to release all data related to this thesis as open access, or, if public release was not possible due to privacy concerns or contractual obligations, to provide the means to other researchers to access this data under restrictions. All measurement data collected for the research in this thesis is either publicly available in open access repositories, or has been stored in our research group’s repository to facilitate reproducibility. In addition to this, we have made the papers that form the basis of this thesis available in the institutional open access repository². Finally, parts of the tooling used to perform measurements and analyses for this thesis have been released as open

¹<http://www.sigcomm.org/events/imc-conference/>

²<https://doc.utwente.nl/>

source software. All other tooling has been stored in the research group’s version control system to allow others to build on this work.

This appendix details our approach to providing access to the datasets accompanying the research presented in this thesis. Section B.2 explains our general approach to publishing datasets as open access data. Next, Section B.3 discusses how we make (parts of) non-public data accessible to other researchers. Finally, in Section B.4, we provide an overview of the datasets per thesis chapter and their status.

B.2 Open Access Publishing Practices

Our preferred method of releasing data is as open access data. We follow best practices for releasing open access data [19]³. This means that we provide a separate technical report that accompanies the datasets and that describes how the data was collected, the period over which data was collected and the vantage point from which the data was collected. Where possible, we use self-describing data storage formats, such as the Apache Avro format used for the data collected by the active DNS measurement system discussed in Chapter 7. We release datasets under a Creative Commons License that allows others to use the data for non-commercial purposes, allows them to share the data under the same conditions and requires attribution⁴. Furthermore, we clearly state attribution requirements for when the datasets are used; typically this requires citing the paper that the datasets were collected for. Finally, for ongoing measurements, such as those performed by the active DNS measurement system discussed in Chapter 7, we make sure that data is updated on a regular basis as new measurement data is being collected.

B.3 Accessibility of Non-Open Datasets

In some cases it is not possible to release datasets as open access. There are two reasons applicable to datasets discussed in this thesis. The first reason is contractual obligations. As, for example, discussed in Chapter 7, some of the TLD zone files that serve as a basis for our active DNS measurements are obtained under contracts that prohibit redistribution. The second reason is privacy⁵. This is for example the case for packet captures of traffic to DNS resolvers. This type

³The author of this thesis actively contributed to this best practice document.

⁴<https://creativecommons.org/licenses/by-nc-sa/4.0/>

⁵See also Appendix C for more information on how we review ethical considerations of measurements.

of data is highly privacy-sensitive, as revealing what DNS queries a client performs can reveal highly personal behaviour of individual users. Consequently, this type of data cannot be stored long term, as the risk to users' privacy is too high. This conflicts with the desire to aid reproducibility. Therefore, to make reproducibility as easy as possible, we make the data collection and aggregation tools we use available as open source⁶ and store aggregate results that can be shared.

If it is not possible to release datasets as open access, we strive to offer access through at least one of the following options:

- **Restricted access under NDA** – fellow researchers are provided access to datasets under a non-disclosure agreement.
- **Provide aggregate results on request** – within reasonable limits we provide aggregate data derived from the dataset in question. This is, of course, subject to available time on our side, and the aggregate data must not violate the reasons for why the data could not be released publicly.

B.4 Per-Chapter Datasets

Table B.1 lists the datasets per chapter of this thesis. The first column lists the chapter, the second the dataset and the third a brief description of the data. The fourth column specifies the access conditions for the data, these are: **AD** - aggregate data available upon request, **OA** - open access data available (URL provided) and **RA** - restricted access available for academic researchers (e.g., under NDA). The fifth column, finally, specifies the reasons for not releasing data as open access (if applicable). These reasons are: **C** - contractual obligations prohibit public release and **P** - privacy concerns prohibit public release. Requests for access to non-public data may be addressed to the authors of the associated papers and to staff members of the University of Twente's EWI-DACS research group.

For each dataset, the way the data is stored is also listed as footnotes in Table B.1. In general, two types of storage are discerned: open access in publicly accessible repositories and private storage in our research group's long term data storage repository. To ensure that data remains accessible for the foreseeable future, access to the research group repository is available to all staff members of the research group.

⁶e.g., <https://github.com/SURFnet/eemo/>

Chapter	Dataset	Brief Description	Access	Reasons
Chapter 3	Authoritative DNS packet captures	Data captured on SURFnet and University of Pennsylvania authoritative name servers to measure occurrences of fragmentation, and to measure occurrences of fragment-related problems.	AD [‡] , RA [†]	C, P
Chapter 4	Amplification measurements	Active measurement data for 2.5 million DNSSEC-signed domains and 2.5 million unsigned domains that provides detailed information about the amplification in responses to a set of DNS query types.	OA [★]	
Chapter 5	Authoritative DNS packet captures	Data captured on SURFnet authoritative name servers to measure occurrences of fragmentation, and to measure occurrences of fragment-related problems.	AD [‡] , RA [†]	C, P
Chapter 5	Amplification measurements	Active measurement data for 0.5 million DNSSEC-signed domains that provides detailed information about the amplification in responses to a set of DNS query types.	RA [†]	C
Chapter 6	DNS resolver packet captures	Data captured on SURFnet and university DNS resolvers.	AD [‡]	P
Chapter 7	Active DNS measurement data	Large-scale longitudinal DNS datasets covering a large number of TLDs, with responses to a set of common DNS queries	AD [‡] , OA [◆] , RA [♣]	C
Chapter 8	Active DNS measurement data	Specific longitudinal datasets for DNSSEC-signed domains, derived from the datasets collected by the system discussed in Chapter 7.	AD [‡] , RA [♣]	C

★https://www.simpleweb.org/wiki/index.php/Traces#DNSSEC_and_its_Potential_for_DDoS_Attacks

◆<https://data.openintel.nl/data/>

♣Through restricted access to OpenINTEL

[†]Original data stored in research group repository

[‡]Aggregate data stored in research group repository

Table B.1: Per-chapter datasets and their access status

Responsible Measurement Practices

The work in this thesis builds on both passive and active measurements of the Internet. Over the years, the Internet measurement research community has established best practices for responsible measurement [12, 43, 150]. All the work presented in this thesis has been performed with such best practices in mind. In this appendix, we provide an overview of the specific measures taken to ensure that both our passive and our active measurements were performed in a manner that the Internet measurement research community generally considers responsible. In all cases, we perform an ethical review, which we discuss in Section C.1. Next, Section C.2 discusses how we approach passive measurements. Section C.3, finally, explains how we approach active measurements.

C.1 Ethical Considerations

Before collecting data, we performed an ethical review to assess potential ethical issues. For these reviews, we followed best current practices, [18] and [57]¹, and documented the ethical considerations in the papers (and in this thesis in the chapters derived from these papers). If the outcome of an ethical review was that there are privacy concerns with regards to the collected data, we restrict access to the collected data and do not release it as open access.

C.2 Passive Measurements

We consider two specific issues when performing passive measurements:

Minimising Privacy Impact

In this thesis, we performed a number of passive measurements, where packet captures were used to record DNS data. Especially traffic captured on DNS

¹The author of this thesis actively contributed to this best practice document.

resolvers is particularly privacy sensitive, as users' DNS queries are highly revealing of their Internet use. For this reason, we take specific measures to ensure that the privacy of the users of resolvers on which we captured traffic was safeguarded. These measures were that we:

- *Discard privacy sensitive data as early as possible* – to give an example: in none of our measurements we needed the actual domain names that individual users were querying for. We therefore did not record this data. For the analysis of query name popularity in Chapter 6, we analysed this popularity on the link between the resolver and upstream authoritative name servers on the Internet. This hides queries from individual users. Furthermore, in that particular example, we were only interested in the ranking of domains, and thus discarded the list of query names as soon as we had verified that the measurement had been executed correctly.
- *Only store aggregate or anonymised results* – in order to ensure reproducibility, measurement results are stored and archived. To protect the privacy of users, we strove to only store aggregate results that are sufficient to reproduce our results, yet do not reveal privacy sensitive information. In practice, this means that in none of the measurements we performed did we store any data that can be traced back to individual users.
- *Obtain permission from relevant stakeholders* – finally, we obtained permission to perform measurements from relevant stakeholders, including privacy and security officers for the networks in which data was collected for the research in this thesis.

Minimising Performance Impact

A second consideration to take into account while performing passive measurements is performance impact. In most cases, our passive measurements were performed on the DNS servers themselves. Given that in some cases we wanted to specifically measure traffic at peak loads, we had to ensure that the system resources required to perform the measurements did not adversely affect the quality of the service we were measuring. We did this by taking two measures:

- *Continuous performance monitoring* – during the measurement, we continuously monitored the performance of the systems on which we measured. Using metrics from the monitoring system, we then compared the performance of the system under measurement to the performance without a running measurement. We observed clear evidence of the impact of measurements, but were able to ensure that the quality of service on the

systems on which we measured did not degrade as a consequence of the measurements.

- *Offload filtering and analysis* – we used tools developed in-house² that captured traffic on the systems on which we wanted to measure and then securely forwarded this traffic to external hosts for filtering and analysis. This reduces the load on the systems under measurement.

C.3 Active Measurements

In the case of the active measurements we performed or planned to perform, we take five issues into consideration:

Re-use Existing Data and Share Results

It is a well-established best practice to first check if existing data from own or externally performed measurements can be re-used. Our research is no exception; in some cases we were able to re-use existing data. Equally, it is recommended practice to share data if possible so other researchers can benefit. We discuss our approach to dataset management in Appendix B.

Clearly Advertise Intent

When performing active measurements, it is important to always clearly advertise the intent of the measurement to anyone who may see traffic from the measurements. In case of the active measurements we performed, we communicated our intent by:

- *Having a web page with a clear description* – for example, in case of the large-scale active DNS measurements we perform, as discussed in Chapter 7, we set up a dedicated web page³ that explains why we perform the measurement, how we perform the measurement and how we can be reached in case of questions or problems. It is also important to clearly state when the information on the page was last updated and to post regular updates so anyone who ends up on the page is reassured that the project is actively looked after.
- *Making measurements traceable* – this can be done by ensuring that reverse DNS entries for IP addresses from which the measurement is performed clearly link to information about the measurement. In case specific IP

²And released as open source, <https://github.com/SURFnet/eemo/>

³<https://www.openintel.nl/>

```
inetnum: xxx.xxx.xxx.0 - xxx.xxx.xxx.255
netname: SURFnet-OpenINTEL
descr: SURFnet bv
descr: OpenINTEL Active DNS Measurements
descr: See http://www.openintel.nl/ for more information
country: NL
admin-c: RVR19-RIPE
...
```

Example C.1: Excerpt from the WHOIS information for one of the IP blocks used by our active DNS measurement system

prefixes are used for the measurement, such as for our large-scale active DNS measurements, we also ensured that the address block registration at the Regional Internet Registry (RIR) makes clear that this block is used to perform active measurements. Example C.1 shows an excerpt from the registration of one of these blocks. Note that the description clearly states the intent, and directs people that consult this registration to the project's webpage. Also note that the administrative contact handle **RVR19-RIPE** expands to up-to-date contact information for the author of this thesis.

- *Communicating with incident response teams* – before starting measurements, we informed the incident response teams for the networks from which we performed measurements about our intentions. This ensures that the duty officer in the incident response team can act adequately if abuse reports regarding the measurement are received. We also explicitly ask the incident response teams to provide feedback on the measurement, as we can benefit from their expertise.

Promptly Respond to Questions and Complaints

Although we received very few queries regarding the measurements we performed and still perform, we have made a point of responding promptly to questions and complaints. In the few cases where we have received questions, we have responded within 24 hours. The OpenINTEL project webpage, for example, also clearly states which time zone we reside in, so people that contact us have an idea when they can expect to receive a response.

Provide Means to Opt Out

Whenever this was possible, we provided a means to opt out of being part of the measurements we perform. For example, the software used for the measure-

ments we performed for Chapters 4 and 5 had specific provisions to filter out domains before the start of the measurement. In case of our large-scale active DNS measurement platform, we can exclude specific name servers (based on IP address) from the measurement.

While we have not received specific opt-out requests or abuse complaints, we observed that traffic from our OpenINTEL measurement platform was blocked by a large domain name registrar in early 2017. We interpreted this block as an opt-out request and promptly excluded the IPs for their name servers from the measurement. We then reached out to them to discuss this block. As a result of this communication, the domain name registrar removed the block and we were able to re-include domains registered through this company in our measurement. One important lesson learned in this process is that it takes time and effort to reach the right people within large organisations; the total time elapsed between blocking and unblocking of the measurement spanned two months.

Actively Monitor Traffic Loads

For all the active measurements we performed for the work presented in this thesis we monitored traffic loads. We ensured that both traffic to individual operators and overall traffic loads remained well within reasonable levels. In addition to this, as we also discussed in Chapter 7, for our large-scale active DNS measurement platform we also set flow rate monitoring alerts on the egress links for the network from which we perform our measurement. This allows us to take direct action if traffic to, e.g., top talkers exceeds pre-set limits.

Bibliography

- [1] “Largest Synchronized Internet Security Effort Underway to Patch Newly Found DNS Bug,” 2008. (link)
- [2] “Forum Standaardisatie - DNSSEC [in Dutch],” 2012. (link)
- [3] “BSI TR-03108-1: Secure E-Mail Transport,” BSI, Bonn, Germany, Tech. Rep., 2016.
- [4] “Forum Standaardisatie - STARTLS en DANE [in Dutch],” 2016. (link)
- [5] “OpenDNSSEC 2.0 Release Notes,” 2016. (link)
- [6] J. Abley, Ó. Guðmundsson, and M. Majkowski, “Providing Minimal-Sized Response to DNS Queries with QTYPE=ANY,” 2016. (link)
- [7] L. Adamic and B. Huberman, “Zipf’s Law and the Internet,” *Glottometrics*, vol. 3, pp. 143–150, 2002.
- [8] B. Ager, H. Dreger, and A. Feldmann, “Predicting the DNSSEC Overhead Using DNS Traces,” in *Proceedings of the 40th Annual IEEE Conference on Information Sciences and Systems, CISS 2006*. Princeton, NJ, USA: IEEE Computer Society, 2007, pp. 1484–1489.
- [9] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig, “Web Content Cartography: Categories and Subject Descriptors,” in *Proceedings of the 11th ACM SIGCOMM Internet Measurement Conference (IMC ’11)*. Berlin, Germany: ACM Press, 2011, pp. 585–600.
- [10] Akamai, “Security Bulletin: DNSSEC Amplification DDoS,” Akamai, Tech. Rep., 2016. (link)
- [11] J. Akkerhuis, L. Chapin, P. Fältström, G. Kowack, L.-J. Liman, and B. Manning, “Scaling the Root,” Tech. Rep., 2009.
- [12] M. Allman and V. Paxson, “Issues and etiquette concerning use of shared measurement data,” in *Proceedings of the 7th ACM SIGCOMM Internet Measurement Conference (IMC ’07)*. San Diego, CA, USA: ACM Press, 2007, pp. 135–140.
- [13] M. Anagnostopoulos, G. Kambourakis, P. Kopanos, G. Louloudakis, and S. Gritzalis, “DNS Amplification Attack Revisited,” *Computers & Security*, vol. 39, no. November 2013, pp. 475–485, 2013.

- [14] D. Anstee, A. Cockburn, and G. Sockrider, “Worldwide Infrastructure Security Report,” Burlington, MA, USA, Tech. Rep., 2014. (link)
- [15] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, “RFC 4033 - DNS Security Introduction and Requirements,” 2005. (link)
- [16] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, “RFC 4034 - Resource Records for the DNS Security Extensions,” 2005. (link)
- [17] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, “RFC 4035 - Protocol Modifications for the DNS Security Extensions,” 2005. (link)
- [18] M. Bailey, D. Dittrich, E. Kenneally, and D. Maughan, “The Menlo Report,” *IEEE Security & Privacy Magazine*, vol. 10, no. 2, pp. 71–75, 2012.
- [19] V. Bajpai, A. W. Berger, P. Eardley, J. Ott, and J. Schönwälder, “Report of Dagstuhl Seminar 16012 - Global Measurements: Practice and Experience,” *Dagstuhl Reports*, vol. 6, no. 1, pp. 15–33, 2016.
- [20] F. Baker and P. Savola, “BCP 84 - Ingress Filtering for Multihomed Networks,” 2004. (link)
- [21] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, “Recommendation for Key Management - Part 1: General (rev. 4),” *NIST SP800-57*, 2016.
- [22] E. Barker and W. C. Barker, “Guideline for Using Cryptographic Standards in the Federal Government: Directives, Mandates and Policies,” *NIST SP800-175A*, 2016.
- [23] E. Barker and Q. Dang, “Recommendation for Key Management - Part 3: Application-Specific Key Management Guidance (rev. 1),” *NIST SP800-57*, 2015.
- [24] R. Barnes, “RFC 6394 - Use Cases and Requirements for DNS-Based Authentication of Named Entities (DANE),” 2011. (link)
- [25] S. M. Bellovin, “Using the Domain Name System for System Break-ins,” in *Proceedings of the 5th USENIX Security Symposium*. Salt Lake City, UT, USA: USENIX Association, 1995.
- [26] I. N. Bermudez, M. Mellia, M. M. Munafo, R. Keralapura, and A. Nucci, “DNS to the Rescue: Discerning Content and Services in a Tangled Web,” in *Proceedings of the 12th ACM SIGCOMM Internet Measurement Conference (IMC '12)*. Boston, MA, USA: ACM Press, 2012, pp. 413–426.
- [27] D. J. Bernstein, P. Birkner, M. Joye, and T. Lange, “Twisted Edwards Curves,” in *Progress in Cryptology - AFRICACRYPT 2008*. Springer Berlin Heidelberg, 2008, vol. 2, pp. 389–405.

- [28] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B. Y. Yang, “High-Speed High-Security Signatures,” *Journal of Cryptographic Engineering*, vol. 2, no. 2, pp. 77–89, 2012.
- [29] D. J. Bernstein and T. Lange, “Failures in NIST’s ECC Standards,” Tech. Rep., 2016. (link)
- [30] D. J. Bernstein and T. Lange, “SafeCurves: Choosing Safe Curves for Elliptic-Curve Cryptography,” 2017. (link)
- [31] D. Bernstein, “High-Speed High-Security Cryptography: Encrypting and Authenticating the Whole Internet,” in *27th Chaos Communication Congress (27C3)*, Berlin, 2010. (link)
- [32] L. Bilge, S. Sen, D. Balzarotti, E. Kirda, and C. Kruegel, “Exposure: A Passive DNS Analysis Service to Detect and Report Malicious Domains,” *ACM Transactions on Information and System Security*, vol. 16, no. 4, 2014.
- [33] D. Bradbury, “Hackers Step up Their Efforts to Exploit DNS Flaw,” 2008. (link)
- [34] R. Braden, “RFC 1122 - Requirements for Internet Hosts – Communication Layers,” 1989. (link)
- [35] R. Braden, “RFC 1123 - Requirements for Internet Hosts – Application and Support,” 1989. (link)
- [36] S. Bradner, L. Conroy, and K. Fujiwara, “RFC 6116 - The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM),” 2011. (link)
- [37] N. Brownlee and K. Claffy, “Internet Measurement,” *IEEE Internet Computing*, vol. 8, no. 5, pp. 30–33, 2004.
- [38] J. Callas, L. Donnerhake, H. Finney, D. Shaw, and R. Thayer, “RFC 4880 - OpenPGP Message Format,” 2007. (link)
- [39] M. Campagna, “Certicom Corp.’s Statement About IPR Related to draft-ietf-dnsext-ecdsa,” 2011. (link)
- [40] B. M. Carson, J. G. Montgomery, and B. Strauch, “Legally Speaking - Domain Names, Cybersquatting, and the ICANN System,” *Against the Grain*, vol. 13, no. 3, pp. 71–74, November 2001.
- [41] E. Casalicchio, M. Caselli, and A. Coletta, “Measuring the Global Domain Name System,” *IEEE Network*, vol. 27, no. 1, pp. 25–31, January 2013.
- [42] S. Castro, D. Wessels, M. Fomenkov, and K. Claffy, “A Day at the Root of the Internet,” *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 5, pp. 41–46, 2008.

- [43] V. G. Cerf, “RFC 1262 - Guidelines for Internet Measurement Activities,” 1991. (link)
- [44] Y. Cheng, J. Chu, S. Radhakrishnan, and A. Jain, “RFC 7413 - TCP Fast Open,” 2014. (link)
- [45] T. Chung, R. van Rijswijk-Deij, B. Chandrasekaran, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson, “A Longitudinal, End-to-End View of the DNSSEC Ecosystem,” in *Accepted for the 26th USENIX Security Symposium (USENIX Security '17)*. Vancouver, BC, Canada: USENIX Association, 2017.
- [46] CloudFlare, “Announcing Universal DNSSEC: Secure DNS for Every Domain,” 2015. (link)
- [47] L. Colitti, “IPv6 at Google,” in *NANOG 49*, San Francisco, CA, USA, 2010. (link)
- [48] L. Constantin, “Attackers Use DNSSEC Amplification to Launch Multi-vector DDoS Attacks,” 2016. (link)
- [49] A. Conta, S. Deering, and M. Gupta, “Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification,” 2006.
- [50] A. Cowperthwaite and A. Somayaji, “The Futility of DNSSec,” in *Proceedings of the 5th Annual Symposium on Information Assurance (ASIA10)*. Albany, NY, USA: University at Albany, State University of New York, 2010, pp. 2–8.
- [51] D. Dagon, M. Antonakakis, P. Vixie, T. Jinmei, and W. Lee, “Increased DNS Forgery Resistance Through 0x20-bit Encoding,” in *Proceedings of the 15th ACM Conference on Computer and Communications Security - CCS '08*. New York, New York, USA: ACM Press, 2008, pp. 211–222.
- [52] J. Damas, M. Graff, and P. Vixie, “RFC 6891 - Extension Mechanisms for DNS (EDNS0),” 2013. (link)
- [53] S. Deering and R. Hinden, “RFC 2460 - Internet Protocol, Version 6 (IPv6) Specification,” 1998. (link)
- [54] L. Deri, L. L. Trombacchi, M. Martinelli, and D. Vannozzi, “Towards a Passive DNS Monitoring System,” in *Proceedings of the 27th Annual ACM Symposium on Applied Computing (SAC '12)*. Trento, Italy: ACM Press, 2012, pp. 629–630.
- [55] L. P. Deutsch, “RFC 606 - Host Names On-line,” 1973. (link)
- [56] T. Dierks and E. Rescorla, “RFC 5246 - The Transport Layer Security (TLS) Protocol Version 1.2,” 2008. (link)

- [57] S. Dietrich, J. van der Ham, A. Pras, R. van Rijswijk-Deij, D. Shou, A. Sperotto, A. van Wynsberghe, and L. Zuck, "Ethics in Data Sharing: Developing a Model for Best Practice," in *2nd Cyber-security Research Ethics Dialogs & Strategy (CREDS II)*, co-located with the 35th IEEE Symposium on Security and Privacy (IEEE S&P 2014). San Jose, CA, USA: IEEE Computer Society, 2014.
- [58] DK Hostmaster A/S, "Netsite Signs Customer's Domain Names with DNSSEC," 2017. (link)
- [59] DNS-OARC, "DNS-OARC Day-in-the-Life Traces," 2016. (link)
- [60] R. Dodopoulos, "DNS-based Detection of Malicious Activity," Master thesis, Eindhoven University of Technology (TU/e), 2015.
- [61] V. Dukhovni and W. Hardaker, "RFC 7671 - The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance," 2015. (link)
- [62] V. Dukhovni and W. Hardaker, "RFC 7672 - SMTP Security via Opportunistic DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS)," 2015. (link)
- [63] Z. Durumeric, E. Wustrow, and J. A. Halderman, "ZMap: Fast Internet-wide Scanning and Its Security Applications," in *Proceedings of the 22nd USENIX Security Symposium*. Washington D.C., USA: USENIX Association, 2013, pp. 605-619.
- [64] D. Eastlake and C. Kaufman, "RFC 2065 - Domain Name System Security Extensions," 1997. (link)
- [65] D. Eastlake, "RFC 2535 - Domain Name System Security Extensions," 1999. (link)
- [66] D. Eastlake, "RFC 6895 - Domain Name System (DNS) IANA Considerations," 2013. (link)
- [67] D. Eastlake and M. Andrews, "RFC 7873 - Domain Name System (DNS) Cookies," 2016. (link)
- [68] ECRYPT-II Project, "eBATS: ECRYPT Benchmarking of Asymmetric Systems," 2016. (link)
- [69] P. Ferguson and D. Senie, "BCP 38 - Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing," 2000. (link)
- [70] O. Filip and E. Petr, "DNS Cache Poisoning, presentation at ICANN 37," Nairobi, Kenya, 2010. (link)

- [71] S. Floyd and V. Paxson, "Difficulties in Simulating the Internet," *IEEE/ACM Transactions on Networking*, vol. 9, no. 4, pp. 392–403, 2002.
- [72] H. Gao, V. Yegneswaran, J. Jiang, Y. Chen, P. Porras, S. Ghosh, and H. Duan, "Reexamining DNS From a Global Recursive Resolver Perspective," *IEEE/ACM Transactions on Networking*, vol. 24, no. 1, pp. 43–57, 2016.
- [73] L. Garber, "Denial-of-service Attacks Rip the Internet," *IEEE Computer Magazine*, vol. 33, no. 4, pp. 12–17, April 2000.
- [74] M. Geva, A. Herzberg, and Y. Gev, "Bandwidth Distributed Denial of Service: Attacks and Defenses," *IEEE Security & Privacy*, vol. 12, no. 1, pp. 54–61, 2014.
- [75] R. Gieben and W. Mekking, "RFC7129 - Authenticated Denial of Existence in the DNS," 2014. (link)
- [76] S. Goldberg, M. Naor, D. Papadopoulos, and L. Reyzin, "NSEC5 from Elliptic Curves," *Cryptology ePrint Archive*, 2016. (link)
- [77] S. Goldberg, M. Naor, D. Papadopoulos, L. Reyzin, S. Vasant, and A. Ziv, "NSEC5 : Provably Preventing DNSSEC Zone Enumeration," in *Proceedings of the Network and Distributed Systems Symposium 2015 (NDSS 2015)*. Internet Society, 2015.
- [78] S. Goldberg, M. Naor, D. Papadopoulos, L. Reyzin, S. Vasant, and A. Ziv, "Stretching NSEC3 to the Limit : Efficient Zone Enumeration Attacks on NSEC3 Variants," Boston University, Tech. Rep., 2015.
- [79] D. Goodin, "Record-Breaking DDoS Reportedly Delivered By >145k Hacked Cameras," 2016. (link)
- [80] Ó. Guðmundsson, "RFC 7218 - Adding Acronyms to Simplify Conversations about DNS-Based Authentication of Named Entities (DANE)," 2014. (link)
- [81] Ó. Guðmundsson, "ECDSA Is Your Friend," in *RIPE 70*, Amsterdam, The Netherlands, 2015. (link)
- [82] Ó. Guðmundsson and S. Crocker, "Observing DNSSEC Validation in the Wild," in *Proceedings of SATIN 2011 Workshop*. London, UK: NPL, 2011.
- [83] T. Halvorson, M. F. Der, I. Foster, S. Savage, L. K. Saul, and G. M. Voelker, "From .academy to .zone: An Analysis of the New TLD Land Rush," in *Proceedings of the 15th ACM SIGCOMM Internet Measurement Conference (IMC '15)*. Tokyo, Japan: ACM Press, 2015, pp. 381–394.
- [84] T. Halvorson, K. Levchenko, S. Savage, and G. M. Voelker, "XXXtortion?" in *Proceedings of the 23rd International Conference on the World Wide Web - WWW '14*. Seoul, Korea: ACM Press, 2014, pp. 901–912.

- [85] M. Hamburg, “Ed448-Goldilocks, a New Elliptic Curve,” *Cryptology ePrint Archive*, no. 625, 2015. (link)
- [86] A. Hankel, R. van den Hoed, E. Hoekstra, and R. van Rijswijk, “Measuring Software Energy Efficiency: Presenting a Methodology and Case Study on DNS Resolvers,” in *Proceedings of the 18th Mediterranean Electrotechnical Conference (MELECON 2016)*. Limassol, Cyprus: IEEE Computer Society, 2016.
- [87] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography*. Springer, 2004.
- [88] W. Hardaker, Ó. Guðmundsson, and S. Krishnaswamy, “RFC 8027 - DNSSEC Roadblock Avoidance,” 2016. (link)
- [89] L. Hendriks, R. de O. Schmidt, R. van Rijswijk-Deij, and A. Pras, “On the Potential of IPv6 Open Resolvers for DDoS Attacks,” in *Proceedings of the 18th Passive and Active Measurement Conference (PAM 2017)*, ser. Lecture Notes in Computer Science, vol. 10176. Sydney, Australia: Springer, 2017, pp. 17–29.
- [90] E. Hernaez, “UDP Fragmentation Breaks SIP in Today’s IP-PBXs,” 2009. (link)
- [91] A. Herzberg and H. Shulman, “Fragmentation Considered Poisonous, or: One-domain-to-rule-them-all.org,” in *2013 IEEE Conference on Communications and Network Security, CNS 2013*, 2013, pp. 224–232.
- [92] A. Herzberg and H. Shulman, “Cipher-Suite Negotiation for DNSSEC: Hop-by-Hop or End-to-End?” *IEEE Internet Computing*, vol. 19, pp. 80–84, 2015.
- [93] S. Hilton, “Dyn Analysis Summary of Friday October 21 Attack,” 2016. (link)
- [94] P. Hoffman and W. Wijngaards, “RFC 6605 - Elliptic Curve Digital Signature Algorithm (DSA) for DNSSEC,” 2012. (link)
- [95] P. Hoffman, “RFC 3207 - SMTP Service Extension for Secure SMTP over Transport Layer Security,” 2002. (link)
- [96] P. Hoffman and J. Schlyter, “RFC 6698 - The DNS-Based Authentication of Named Entities (DANE): Transport Layer Security (TLS) Protocol: TLSA,” 2012. (link)
- [97] P. Hoffman, A. Sullivan, and K. Fujiwara, “RFC 7719 - DNS Terminology,” 2015. (link)
- [98] S. Hollenbeck, “RFC 5730 - Extensible Provisioning Protocol (EPP),” 2009. (link)
- [99] C. Hornig, “RFC 894 - A Standard for the Transmission of IP Datagrams over Ethernet Networks,” 1984. (link)

- [100] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman, “RFC 7858 - Specification for DNS over Transport Layer Security (TLS),” 2016. (link)
- [101] G. Huston, “Measuring DNSSEC Use,” in *APNIC 36 Conference*, Xi’An, China, 2013. (link)
- [102] G. Huston and G. Michaelson, “ECDSA P-256 Support in DNSSEC-Validation Resolvers,” in *ICANN 55 DNSSEC Workshop*, Marrakech, Morocco, 2016. (link)
- [103] ICANN, “ICANN Centralized Zone Data Service.” (link)
- [104] ICANN, “Registrar Accreditation Agreement,” 2013. (link)
- [105] International Organization for Standardisation, “ISO 3166 - Country Codes.” (link)
- [106] International Telecommunication Union, “ITU-T E.164 - The International Public Telecommunication Numbering Plan,” 2010.
- [107] Internet Architecture Board, “RFC 2826 - IAB Technical Comment on the Unique DNS Root,” 2000. (link)
- [108] Internet Systems Consortium, “ISC Internet Domain Survey,” 2016. (link)
- [109] M. Jonker, A. Sperotto, R. van Rijswijk-Deij, R. Sadre, and A. Pras, “Measuring the Adoption of DDoS Protection Services,” in *Proceedings of the 16th ACM SIGCOMM Internet Measurement Conference (IMC ’16)*. Santa Monica, CA, USA: ACM Press, 2016.
- [110] S. Josefsson and I. Liusvaara, “RFC 8032 - Edwards-Curve Digital Signature Algorithm (EdDSA),” 2017. (link)
- [111] J. Jung, E. Sit, H. Balakrishnan, and R. Morris, “DNS Performance and the Effectiveness of Caching,” *IEEE/ACM Transactions on Networking*, vol. 10, no. 5, pp. 589–603, October 2002.
- [112] D. Kaminsky, “Black Ops 2008: It’s the End of the Cache As We Know It,” in *Black Hat USA*, 2008. (link)
- [113] Kaspersky Lab, “Kaspersky DDoS Intelligence Report for Q1 2016,” 2016. (link)
- [114] S. Kitterman, “RFC 7208 - Sender Policy Framework (SPF) for Authorising Use of Domains in Email, Version 1,” 2014. (link)
- [115] A. Klein, “BIND 9 DNS Cache Poisoning,” Tech. Rep., 2007. (link)
- [116] J. Klensin, “RFC 5321 - Simple Mail Transfer Protocol,” 2008. (link)

- [117] N. Koblitz and A. J. Menezes, “A Riddle Wrapped in an Enigma,” *IACR Cryptology ePrints*, vol. 2015, no. 1018, 2015. (link)
- [118] Y. Koç, A. Jamakovic, and B. Gijsen, “A Global Reference Model of the DNS,” *International Journal of Critical Infrastructure Protection*, vol. 5, no. 3-4, pp. 108–117, 2012.
- [119] P. Koch, M. Larson, and P. Hoffman, “RFC 8109 - Initializing a DNS Resolver with Priming Queries,” 2017. (link)
- [120] O. Kolkman and R. Gieben, “RFC 4641 - DNSSEC Operational Practices,” 2006. (link)
- [121] O. Kolkman, W. Mekking, and R. Gieben, “RFC 6781 - DNSSEC Operational Practices, Version 2,” 2012. (link)
- [122] A. Kountouras, P. Kintis, C. Lever, Y. Chen, Y. Nadji, D. Dagon, M. Antonakakis, and R. Joffe, “Enabling Network Security Through Active DNS Datasets,” in *Proceedings of Research in Attacks, Intrusions, and Defenses (RAID 2016)*, Paris, France, 2016.
- [123] L. Krämer, J. Krupp, D. Makita, T. Nishizoe, T. Koide, K. Yoshioka, and C. Rossow, “AmpPot: Monitoring and Defending Against Amplification DDoS Attacks,” in *Research in Attacks, Intrusions, and Defenses*. Kyoto, Japan: Springer, 2015, pp. 615–636.
- [124] C. Kreibich, N. Weaver, B. Nechaev, and V. Paxson, “Netalyzr: Illuminating the Edge Network,” in *Proceedings of the 10th ACM SIGCOMM Internet Measurement Conference (IMC '10)*. Melbourne, Australia: ACM Press, 2010, pp. 246–259.
- [125] M. Kührer, T. Hupperich, C. Rossow, and T. Holz, “Exit from Hell? Reducing the Impact of Amplification DDoS Attacks,” in *Proceedings of the 23rd USENIX Security Symposium*. San Diego, CA, USA: USENIX Association, 2014, pp. 111–125.
- [126] W. Kumari, Ó. Guðmundsson, and G. Barwood, “RFC 7344 - Automating DNSSEC Delegation Trust Maintenance,” 2014. (link)
- [127] J. Latour, Ó. Guðmundsson, P. Wouters, and M. Pounsett, “Third Party DNS Operator to Registrars/Registries Protocol,” 2017. (link)
- [128] B. Laurie, G. Sisson, R. Arends, and D. Blacka, “RFC 5155 - DNS Security (DNSSEC) Hashed Authenticated Denial of Existence,” 2008. (link)
- [129] D. Lee, “Global Internet Slows Down After ‘Biggest Attack in History’,” 2013. (link)

- [130] M. Lentz, D. Levin, J. Castonguay, N. Spring, and B. Bhattacharjee, “D-mystifying the D-root Address Change,” in *Proceedings of the 13th ACM SIGCOMM Internet Measurement Conference (IMC '13)*. Barcelona, Spain: ACM Press, 2013, pp. 57–62.
- [131] P. Lexis, “PowerDNS Authoritative Server 4.0.0 Released,” 2016. (link)
- [132] W. Lian, E. Rescorla, H. Shacham, and S. Savage, “Measuring the Practical Impact of DNSSEC Deployment,” in *Proceedings of the 22nd USENIX Security Symposium (USENIX Security 2013)*. Washington, D.C.: USENIX, 2013, pp. 573–588.
- [133] D. Lindsay, *International Domain Name Law - ICANN and the UDRP*. Portland, OR: Hart Publishing, 2007.
- [134] S. Mansfield-Devine, “The Imitation Game: How Business E-mail Compromise Scams Are Robbing Organisations,” *Computer Fraud and Security*, no. 11, pp. 5–10, 2016.
- [135] D. McGrew, K. Igoe, and M. Salter, “RFC 6090 - Fundamental Elliptic Curve Cryptography Algorithms,” 2011. (link)
- [136] J. Menn, J. Finkle, and D. Volz, “Cyber Attacks Disrupt PayPal, Twitter, Other Sites,” 2016. (link)
- [137] D. Migault, C. Girard, and M. Laurent, “A Performance View on DNSSEC Migration,” in *Proceedings of the 6th International Conference on Network and Service Management (CNSM 2010)*. Niagara Falls, Canada: IFIP, 2010, pp. 469–474.
- [138] P. Mockapetris, “RFC 882 - Domain Names - Concepts and Facilities,” 1983. (link)
- [139] P. Mockapetris, “RFC 883 - Domain Names - Implementation and Specification,” 1983.
- [140] P. Mockapetris, “RFC 1034 - Domain Names - Concepts and Facilities,” 1987. (link)
- [141] P. Mockapetris, “RFC 1035 - Domain Names - Implementation and Specification,” 1987. (link)
- [142] K. Moriarty, B. Kaliski, J. Jonsson, and A. Rusch, “RFC 8017 - PKCS #1: RSA Cryptography Specifications Version 2.2,” 2016. (link)
- [143] S. Morris, J. Ihren, J. Dickinson, and W. Mekking, “RFC 7583 - DNSSEC Key Rollover Timing Considerations,” 2015. (link)
- [144] E. Naone, “A Patch to Fix the Net,” 2008. (link)

- [145] National Cyber Security Centre, “Cyber Security Assessment Netherlands (CSAN) 2016,” The Hague, The Netherlands, Tech. Rep., 2016.
- [146] NIST, “FIPS PUB 186-4 - Digital Signature Standard (DSS),” *Processing Standards Publication*, 2009. (link)
- [147] E. Osterweil, D. Massey, and L. Zhang, “Observations from the DNSSEC Deployment,” in *Proceedings of the 3rd IEEE Workshop on Secure Network Protocols (NPsec 2007)*. Beijing, China: IEEE Computer Society, 2007, pp. 19–24.
- [148] E. Osterweil, D. McPherson, S. DiBenedetto, C. Papadopoulos, and D. Massey, “Behavior of DNS’ Top Talkers, a .com/.net View,” in *Proceedings of the 13th Passive and Active Measurement Conference (PAM 2012)*, ser. Lecture Notes in Computer Science, vol. 7192. Vienna, Austria: Springer, 2012, pp. 211–220.
- [149] E. Osterweil, M. Ryan, D. Massey, and L. Zhang, “Quantifying the Operational Status of the DNSSEC Deployment,” in *Proceedings of the 8th ACM SIGCOMM Internet Measurement Conference (IMC ’08)*. Vouliagmeni, Greece: ACM Press, 2008, pp. 231–242.
- [150] C. Papadopoulos and J. Heidemann, “Towards Best Practices for Active Network Measurement,” in *CAIDA AIMS Workshop*, San Diego, CA, USA, 2009.
- [151] D. Papadopoulos, D. Wessels, S. Huque, M. Naor, J. Včelak, L. Reyzin, and S. Goldberg, “Can NSEC5 Be Practical for DNSSEC Deployments?” *IACR Cryptology ePrints*, 2017. (link)
- [152] V. Pappas, D. Wessels, D. Massey, S. Lu, A. Terzis, and L. Zhang, “Impact of Configuration Errors on DNS Robustness,” *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 3, pp. 275–290, 2009.
- [153] R. Perdisci, I. Corona, and G. Giacinto, “Early Detection of Malicious Flux Networks via Large-Scale Passive DNS Traffic Analysis,” *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 5, pp. 714–726, 2012.
- [154] R. Perlner and D. Cooper, “Quantum Resistant Public Key Cryptography: a Survey,” in *Proceedings of the 8th Symposium on Identity and Trust on the Internet - IDtrust ’09*, 2009, pp. 85–93.
- [155] J. Postel, “RFC 791 - Internet Protocol,” Marina del Rey, CA, USA, 1981. (link)
- [156] J. Postel, “RFC 792 - Internet Control Message Protocol,” 1981. (link)
- [157] J. Postel, “RFC 805 - Compute Mail Meeting Notes,” 1982. (link)
- [158] J. Postel, “RFC 881 - The Domain Names Plan and Schedule,” 1983. (link)

- [159] J. Postel, “RFC 897 - Domain Name System Implementation Schedule,” 1984. (link)
- [160] J. Postel, “RFC 921 - Domain Name System Implementation Schedule - Revised,” 1984. (link)
- [161] K. Rikitake, K. Nakao, S. Shimojo, and H. Nogawa, “DNSSEC Feasibility Issues and the Transport Validation Assessment,” in *Proceedings of IPSJ Computer Security Symposium 2006 (CSS2006)*, ser. IPSJ Symposium Series. IPSJ, 2006, pp. 313–319.
- [162] K. Rikitake, K. Nakao, S. Shimojo, and H. Nogawa, “UDP Large-Payload Capability Detection for DNSSEC,” *IEICE Transactions on Information and Systems*, vol. E91-D, no. 5, pp. 1261–1273, 2008.
- [163] C. Rossow, “Amplification Hell: Revisiting Network Protocols for DDoS Abuse,” in *Proceedings of the 2014 Network and Distributed Systems Security Symposium (NDSS 2014)*. San Diego: Internet Society, 2014.
- [164] J. J. Santanna and A. Sperotto, “Characterizing and Mitigating The DDoS-as-a-Service Phenomenon,” in *Proceedings of the 8th International Conference on Autonomous Infrastructure, Management and Security (AIMS)*, ser. Lecture Notes in Computer Science, vol. 8508. Brno, Czech Republic: Springer, 2014.
- [165] J. J. Santanna, R. van Rijswijk-Deij, R. Hofstede, A. Sperotto, M. Wierbosch, L. Z. Granville, and A. Pras, “Booters - An Analysis of DDoS-as-a-Service Attacks,” in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*. Ottawa, CA: IEEE, May 2015, pp. 243–251.
- [166] K. Scarfone and P. Hoffman, “Guidelines on Firewalls and Firewall Policy,” *NIST SP800-41*, 2009.
- [167] B. Schneier, “NSA Surveillance: How to Stay Secure,” September 2013. (link)
- [168] K. Schomp, T. Callahan, M. Rabinovich, and M. Allman, “On Measuring the Client-Side DNS Infrastructure,” in *Proceedings of the 13th ACM SIGCOMM Internet Measurement Conference (IMC ’13)*. Barcelona, Spain: ACM Press, 2013, pp. 77–90.
- [169] P. K. Sen, “Estimates of the Regression Coefficient Based on Kendall’s Tau,” *Journal of the American Statistical Association*, vol. 63, no. 324, pp. 1379–1389, December 1968.
- [170] N. Smart, “ECRYPT II Yearly Report on Algorithms and Keysizes 2011-2012,” European Commission, Brussels, Belgium, Tech. Rep., 2012. (link)
- [171] W. Stewart, “Living Internet,” 2015. (link)

- [172] M. StJohns, “RFC 5011 - Automated Updates of DNS Security (DNSSEC) Trust Anchors,” 2007. (link)
- [173] O. Surý and R. Edmonds, “RFC 8080 - Edwards-Curve Digital Security Algorithm (EdDSA) for DNSSEC,” 2017. (link)
- [174] J. Szurdi, B. Kocso, G. Cseh, M. Felegyhazi, and C. Kanich, “The Long Tail of Typosquatting Domain Names,” in *Proceedings of the 23rd USENIX Security Symposium*. San Diego, CA, USA: USENIX Association, 2014, pp. 191–206.
- [175] H. Theil, “A Rank-Invariant Method of Linear and Polynomial Regression Analysis. I, II, III,” *Koninklijke Nederlandse Academie van Wetenschappen*, vol. 53, pp. 386–392, 512–525, 1397–1412, 1950.
- [176] M. Ülle and T. Martens, “E-voting in Estonia 2005. The First Practice of Country-Wide Binding Internet Voting in the World,” in *Electronic Voting 2006: 2nd International Workshop, Co-organized by Council of Europe, ESF TED, IFIP WG 8.6 and E-Voting.CC, August, 2nd - 4th, 2006*, ser. Lecture Notes in Informatics, vol. P-86. Castle Hofen, Bregenz, Austria: Gesellschaft für Informatik (GI), 2006, pp. 15–26.
- [177] United States Senate, “The Anticybersquatting Consumer Protection Act,” Washington DC, USA, 1999.
- [178] L. Valenta, S. Cohnsey, A. Liao, J. Fried, S. Bodduluri, and N. Heninger, “Factoring as a Service,” *IACR Cryptology ePrints*, no. 1000, 2015. (link)
- [179] F. Valsorda and Ó. Guðmundsson, “Compact DNSSEC Denial of Existence or Black Lies,” 2016. (link)
- [180] G. van den Broek, R. M. van Rijswijk, A. Sperotto, and A. Pras, “DNSSEC Meets Real World: Dealing with Unreachability Caused by Fragmentation,” *IEEE Communications Magazine*, vol. 52, no. April, pp. 154–160, 2014.
- [181] R. van Rijswijk-Deij, “MTU Woes and the Merits of Re-signing,” 2010. (link)
- [182] R. van Rijswijk-Deij, “MTU Woes Again...” 2011. (link)
- [183] R. van Rijswijk-Deij, “Dealing with Fragmentation in EDNS0: Proposal for a Recommendation,” in *RIPE 65*, Amsterdam, The Netherlands, 2012.
- [184] R. van Rijswijk-Deij, “Deploying DNSSEC - Validation on Recursive Caching Name Servers,” SURFnet, Utrecht, The Netherlands, Tech. Rep., 2012.
- [185] R. van Rijswijk-Deij, “DNSSEC and Fragmentation: A Prickly Combination,” in *ICANN 45 DNSSEC Workshop*, Toronto, Canada, 2012.
- [186] R. van Rijswijk-Deij, “DNSSEC: Dealing with Hosts That Don’t Get Fragments,” in *RIPE 64*, Ljubljana, Slovenia, 2012. (link)

- [187] R. van Rijswijk-Deij, “DNSSEC: What Every Sysadmin Should Know to Keep Things Working,” in *Proceedings of the 26th USENIX Large Installation System Administration Conference (LISA '12)*. San Diego, CA, USA: USENIX Association, 2012.
- [188] R. van Rijswijk-Deij, K. Hageman, A. Sperotto, and A. Pras, “The Performance Impact of Elliptic Curve Cryptography on DNSSEC Validation,” *IEEE/ACM Transactions on Networking*, vol. 25, no. 2, 2017.
- [189] R. van Rijswijk-Deij, M. Jonker, and A. Sperotto, “On the Adoption of the Elliptic Curve Digital Signature Algorithm (ECDSA) in DNSSEC,” in *Proceedings of the 12th International Conference on Network and Service Management (CNSM 2016)*. Montréal, Canada: IFIP, 2016.
- [190] R. van Rijswijk-Deij, M. Jonker, A. Sperotto, and A. Pras, “The Internet of Names: A DNS Big Dataset Actively Measuring 50% of the Entire DNS Name Space, Every Day,” in *Proceedings of ACM SIGCOMM 2015*. London, UK: ACM Press, 2015.
- [191] R. van Rijswijk-Deij, M. Jonker, A. Sperotto, and A. Pras, “A High-Performance, Scalable Infrastructure for Large-Scale Active DNS Measurements,” *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 7, pp. 1877–1888, 2016.
- [192] R. van Rijswijk-Deij and E. Poll, “Using Trusted Execution Environments in Two-Factor Authentication: Comparing Approaches,” in *Proceedings of the Open Identity Summit 2013 (OID-2013)*, ser. Lecture Notes in Informatics, vol. P-223. Kloster Banz, Germany: Springer, 2013.
- [193] R. van Rijswijk-Deij, A. Sperotto, and A. Pras, “DNSSEC and Its Potential for DDoS Attacks,” in *Proceedings of the 14th ACM SIGCOMM Internet Measurement Conference (IMC '14)*. Vancouver, BC, Canada: ACM Press, 2014.
- [194] R. van Rijswijk-Deij, A. Sperotto, and A. Pras, “Large-Scale DNS and DNSSEC Data Sets for Network Security Research (TR-CTIT-14-13),” University of Twente, Enschede, Tech. Rep., 2014.
- [195] R. van Rijswijk-Deij, A. Sperotto, and A. Pras, “Making the Case for Elliptic Curves in DNSSEC,” *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 5, 2015.
- [196] R. van Rijswijk-Deij and J. van Dijk, “tiqr : a Novel Take on Two-Factor Authentication,” in *Proceedings of the 25th Large Installation System Administration Conference (LISA '11)*. Boston, MA: USENIX Association, 2011, pp. 81–97.
- [197] R. van Rijswijk-Deij, R. van Rein, D. Yoshikawa, and P. Brand, “Hardening the Internet - The Impact and Importance of DNSSEC,” SURFnet, Utrecht, The Netherlands, Tech. Rep., 2009.

- [198] W. van Wezenbeeck, H. Touwen, A. Versteeg, and A. van Wesenbeeck, “Nationaal Plan Open Science,” Ministry of Education, Culture and Science, The Hague, The Netherlands, Tech. Rep., 2017.
- [199] R. Vaughn and G. Evron, “DNS Amplification Attacks (preliminary release),” in *DEFCON 2006*, 2006.
- [200] VeriSign, “VeriSign Domain Name Industry Brief,” 2016. (link)
- [201] P. Vixie, “DNS and BIND Security Issues,” in *Proceedings of the 5th USENIX Security Symposium*. Salt Lake City, UT, USA: USENIX Association, 1995.
- [202] P. Vixie and V. Schryver, “ISC Technical Note - DNS Response Rate Limiting (DNS RRL),” 2012. (link)
- [203] M. Wander, “Measurement Survey of Server-Side DNSSEC Adoption,” in *ICANN 56 DNSSEC Workshop*, Helsinki, Finland, 2016. (link)
- [204] M. Wander, L. Schwittmann, C. Boelmann, and T. Weis, “GPU-Based NSEC3 Hash Breaking,” in *Proceedings of the 13th International Symposium on Network Computing and Applications (NCA 2014)*. IEEE Computer Society, 2014, pp. 137–144.
- [205] M. Wander and T. Weis, “Measuring Occurrence of DNSSEC Validation,” in *Proceedings of the 14th Passive and Active Measurement Conference (PAM 2013)*, ser. Lecture Notes in Computer Science, M. Roughan and R. Chang, Eds., vol. 7799. Hong Kong, China: Springer, 2013, pp. 125–134.
- [206] X. S. Wang, A. Balasubramanian, A. Krishnamurthy, and D. Wetherall, “Demystifying Page Load Performance with WProf,” in *Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI ’13)*. Lombard, IL, USA: USENIX Association, 2013, pp. 473–485.
- [207] Y. A. Wang, C. Huang, J. Li, and K. W. Ross, “Queen: Estimating Packet Loss Rate Between Arbitrary Internet Hosts,” in *Proceedings of the 10th International Conference on Passive and Active Measurements (PAM ’09)*, ser. Lecture Notes in Computer Science, vol. 5448. Seoul, Korea: Springer, 2009, pp. 57–66.
- [208] N. Weaver, C. Kreibich, B. Nechaev, and V. Paxson, “Implications of Netalyzr’s DNS Measurements,” in *Proceedings of the SATIN 2011 Workshop*. London, UK: NPL, 2011.
- [209] S. Weiler and D. Blacka, “RFC 6840 - Clarifications and Implementation Notes for DNS Security (DNSSEC),” 2013. (link)
- [210] F. Weimer, “Passive DNS Replication,” in *Proceedings of the 17th FIRST Conference (FIRST 2005)*, Singapore, Singapore, 2005.

- [211] D. Wessels, M. Fomenkov, N. Brownlee, and K. Claffy, “Measurements and Laboratory Simulations of the Upper DNS Hierarchy,” in *Proceedings of the 5th International Workshop on Passive and Active Measurements (PAM 2004)*, ser. Lecture Notes in Computer Science, vol. 3015. Antibes Juan-les-Pins, France: Springer, 2004, pp. 147–157.
- [212] T. White, *Hadoop - The Definitive Guide*, 4th ed. O’Reilly Media, 2015.
- [213] W. C. A. Wijngaards and B. J. Overeinder, “Securing DNS: Extending DNS Servers with a DNSSEC Validator,” *IEEE Security and Privacy*, vol. 7, no. 5, pp. 36–43, 2009.
- [214] S. Winter, “How to Prevent RADIUS Packet Fragmentation When Using EAP?” in *IETF 71*, Philadelphia, PA, USA, 2008. (link)
- [215] P. Wouters, “RFC 7929 - DNS-Based Authentication of Named Entities (DANE) Bindings for OpenPGP,” 2016. (link)
- [216] M. Wullink, G. C. M. Moura, M. Müller, and C. Hesselman, “ENTRADA: A High-Performance Network Traffic Data Streaming Warehouse,” in *Proceedings of the 2016 IEEE/IFIP Network Operations and Management Symposium (NOMS 2016)*. Istanbul, Turkey: IEEE Computer Society, 2016, pp. 913–918.
- [217] H. Yang, E. Osterweil, D. Massey, S. Lu, and L. Zhang, “Deploying Cryptography in Internet-Scale Systems: A Case Study on DNSSEC,” *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 5, pp. 656–669, 2011.
- [218] D. York, O. Surý, P. Wouters, and Ó. Guðmundsson, “Observations on Deploying New DNSSEC Cryptographic Algorithms,” 2016. (link)
- [219] Y. Yu, D. Wessels, M. Larson, and L. Zhang, “Authority Server Selection in DNS Caching Resolvers,” *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 2, p. 80, 2012.
- [220] Y. Yu, D. Wessels, M. Larson, and L. Zhang, “Check-Repeat: A New Method of Measuring DNSSEC Validating Resolvers,” in *Proceedings of the 5th IEEE International Traffic Monitoring and Analysis Workshop (TMA 2013)*. IEEE, April 2013, pp. 3147–3152.
- [221] R. Zakon, “RFC 2235 - Hobbes’ Internet Timeline,” 1997. (link)
- [222] S. T. Zargar, J. Joshi, and D. Tipper, “A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks,” *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 2046–2069, January 2013.
- [223] L. Zhu, D. Wessels, A. Mankin, and J. Heidemann, “Measuring DANE TLSA Deployment,” in *Proceedings of the 7th International Workshop on Traffic Measurements and Analysis (TMA 2015)*. Barcelona, Spain: Springer, 2015, pp. 219–232.

About the Author

Roland van Rijswijk-Deij was born in Arnhem, The Netherlands, on March 17th, 1977. He received an M.Sc. degree in Computer Science from the University of Twente, Enschede, The Netherlands in 2001. Roland has a background in embedded systems and applied cryptography. He previously worked for British Telecom (2000, traineeship), Royal Philips Electronics (2001-2002), Advanced Encryption Technology (AET) Europe (2002-2006) and In-Traffic (2006-2008).

Since 2008, Roland works for SURFnet, the National Research and Education Network in The Netherlands. At SURFnet, Roland is responsible for SURFnet's DNS and DNSSEC infrastructure. He also initiates and leads innovation projects in the area of Internet security and stability. Past innovation projects initiated by Roland have focused on DNS, DNSSEC, detecting and mitigating DDoS attacks,

IPv6 and many other topics. Roland regularly presents his work in international networking venues, such as TNC, Internet2 conferences, IETF meetings, ICANN meetings, RIPE meetings and NANOG.

From 2013 to 2017 Roland pursued a Ph.D. degree next to his job at SURFnet. First in the Digital Security group at the Radboud University in Nijmegen, The Netherlands, and from 2014 to 2017 in the Design and Analysis of Communication Systems (DACS) group at the University of Twente in Enschede, The Netherlands.

A more complete overview of Roland's curriculum vitae can be found on his LinkedIn page, <https://www.linkedin.com/in/rolandvanrijswijk>.

Roland can be reached via his private e-mail address: roland@mazuki.nl

List of Publications

Below is a list of peer-reviewed academic publications that Roland has authored or co-authored in reverse chronological order:

- T. Chung, R. van Rijswijk-Deij, B. Chandrasekaran, D. Choffnes, D. Levin, B.M. Maggs, A. Mislove and C. Wilson. *A Longitudinal, End-to-End View of the DNSSEC Ecosystem*. Accepted for the 26th USENIX Security Symposium (USENIX Security '17). Vancouver, BC, Canada: USENIX Association [45].
- R. van Rijswijk-Deij, K. Hageman, A. Sperotto and A. Pras. *The Performance Impact of Elliptic Curve Cryptography on DNSSEC Validation*. In IEEE/ACM Transactions on Networking, Volume 25, Issue 2 (April 2017) [188].
- L. Hendriks, R. de Oliveira Schmidt, R. van Rijswijk-Deij and A. Pras. *On the Potential of IPv6 Open Resolvers for DDoS Attacks*. Proceedings of the 18th Passive and Active Measurement Conference (PAM 2017). Sydney, Australia [89].
- M. Jonker, A. Sperotto, R. van Rijswijk-Deij, R. Sadre and A. Pras. *Measuring the Adoption of DDoS Protection Services*. Proceedings of ACM IMC 2016. Santa Monica, CA, USA: ACM Press [109].
- R. van Rijswijk-Deij, M. Jonker and A. Sperotto. *On the Adoption of the Elliptic Curve Digital Signature Algorithm (ECDSA) in DNSSEC*. Proceedings of the 12th International Conference on Network and Service Management (CNSM 2016). Montréal, Canada: IFIP [189].
- A. Hankel, R. van den Hoed, E. Hoekstra and R. van Rijswijk-Deij. *Measuring Software Energy Efficiency: Presenting a Methodology and Case Study on DNS Resolvers*. Proceedings of the 18th Mediterranean Electrotechnical Conference (MELECON 2016). Limassol, Cyprus: IEEE Computer Society [86].
- R. van Rijswijk-Deij, M. Jonker, A. Sperotto and A. Pras. *A High-Performance, Scalable Infrastructure for Large-Scale Active DNS Measurements*. IEEE Journal on Selected Areas in Communications (JSAC), Volume 34, Issue 7 (May 2016) [191].
- R. van Rijswijk-Deij, M. Jonker, A. Sperotto and A. Pras. *The Internet of Names: A DNS Big Dataset – Actively Measuring 50% of the Entire DNS Name Space, Every Day*. Proceedings of ACM SIGCOMM 2015. London, UK: ACM Press [190].
- R. van Rijswijk-Deij, A. Sperotto and A. Pras. *Making the Case for Elliptic Curves in DNSSEC*. ACM SIGCOMM Computer Communication Review, Volume 45, Issue 5 (October 2015) [195].
- J.J. Santanna, R. van Rijswijk-Deij, R. Hofstede, A. Sperotto, M. Wierbosch, L.Z. Granville and A. Pras. *Booters - an Analysis of DDoS-as-a-Service Attacks*. Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM 2015). Ottawa, Canada: IEEE Computer Society [165].

- R. van Rijswijk-Deij, A. Sperotto and A. Pras. *DNSSEC and Its Potential for DDoS Attacks - a Comprehensive Measurement Study*. Proceedings of ACM IMC 2014. Vancouver, BC, Canada: ACM Press [193].
- G. van den Broek, R. van Rijswijk-Deij, A. Sperotto and A. Pras. *DNSSEC Meets Real World: Dealing with Unreachability Caused by Fragmentation*. IEEE Communications Magazine, Vol. 52 (April 2014), pp. 154-160 [180].
- S. Diettrich, J. van der Ham, A. Pras, R. van Rijswijk-Deij, D. Shou, A. Sperotto, A. van Wynsberghe and L. Zuck. *Ethics in Data Sharing: Developing a Model for Best Practice*. Proceedings of the 2nd Cybersecurity Research Ethics Dialogs & Strategy (CREDS-II), co-located with the 35th IEEE Symposium on Security and Privacy (IEEE S&P 2014). San Jose, CA, USA: IEEE Computer Society [57].
- R. van Rijswijk-Deij and E. Poll. *Using Trusted Execution Environments in Two-Factor Authentication: Comparing Approaches*. Proceedings of the Open Identity Summit 2013 (OID 2013). Kloster Banz, Germany: Gesellschaft für Informatik [192].
- R. van Rijswijk-Deij and J. van Dijk. *tiqr: a Novel Take on Two-Factor Authentication*. Proceedings of the 25th international conference on Large Installation System Administration (LISA '11). Boston, MA, USA: USENIX Association, 2011 [196].

Notable Awards

Both Roland's academic work and work he has performed while at SURFnet have been recognised internationally. Below is a list of notable awards in chronological order:

- **Runner-up ACLU Privacy-by-Design Award 2011** – for outstanding work on a privacy-friendly open source two-factor authentication solution for smart phones called 'tiqr' (<https://tiqr.org/>).
- **ACM SIGCOMM IMC Community Contribution Award 2014** – for his paper 'DNSSEC and Its Potential for DDoS Attacks' [193].
- **Internet Research Task Force (IRTF) Applied Networking Research Prize (ANRP) 2015** – also for his paper 'DNSSEC and Its Potential for DDoS Attacks' [193].
- **Internet Research Task Force (IRTF) Applied Networking Research Prize (ANRP) 2017** – for his paper 'The Performance Impact of Elliptic Curve Cryptography on DNSSEC Validation' [188].

IMPROVING DNS SECURITY

A MEASUREMENT-BASED APPROACH

ROLAND VAN RIJSWIJK-DEIJ

ISBN 978-90-365-4329-3

ISSN 1381-3617 (CTIT Ph.D. thesis Series No. 17-430)

DOI 10.3990/1.9789036543293

Copyright ©2017 Roland van Rijswijk-Deij

This work is licensed under a Creative Commons

Attribution-NonCommercial-ShareAlike 4.0 International License.