

# Deploying DNSSEC

## *Validation on recursive caching name servers*

Author(s): Roland van Rijswijk- Deij

Version: 2.0

Date: August 2012

## Management Summary

The Domain Name System (DNS) is a crucial part of the core infrastructure of the Internet. At nearly 30 years the DNS is showing its age. The Internet and – more importantly – the security requirements of its users have changed radically since the DNS was first introduced.

DNSSEC provides a welcome extension to the DNS adding authenticity to the DNS. This key feature not only solves current security issues in the DNS, it also opens up possibilities for new Internet security technologies.

The rollout of DNSSEC has two sides: rollout on the client side of DNS – what we call DNSSEC validation – and rollout on the server side – what we call DNSSEC signing. In this paper we provide information about the rollout on the client side. We will briefly explain what this entails in relation to the architecture of the DNS. We will then discuss requirements and deployment considerations. We finish by providing pointers on how to plan deployment of DNSSEC on the client side – your caching recursive name servers – and what you can expect from an operational point of view if you decide to deploy DNSSEC validation.

After reading this document you will understand the benefits of deploying DNSSEC validation, you will be able to plan your deployment and assess and mitigate the risks for your deployment.



# Contents

<b>Management Summary .....</b>	<b>2</b>
<b>1 Introduction .....</b>	<b>5</b>
1.1 Purpose .....	5
1.2 Required knowledge and intended audience .....	5
1.3 Reading guide .....	5
1.4 Acknowledgements .....	6
1.5 Endorsement SIDN .....	6
<b>2 Cost and benefits of deploying DNSSEC validation .....</b>	<b>7</b>
2.1 Introduction .....	7
2.2 Trust in DNS answers .....	7
2.3 Derived services .....	7
2.4 'Good netizenship' .....	8
2.5 Cost .....	8
2.6 Conclusion .....	8
<b>3 DNS architecture .....</b>	<b>9</b>
3.1 Introduction .....	9
3.2 Client/server view on DNS .....	9
3.3 The client side of the DNS .....	9
3.4 The server side of the DNS .....	10
3.5 Combining the client and server side of DNS .....	11
<b>4 Scope .....</b>	<b>12</b>
4.1 Introduction .....	12
4.2 Caching recursive name servers .....	12
4.3 End user systems .....	13
<b>5 Requirements before deployment .....</b>	<b>14</b>
5.1 Introduction .....	14
5.2 Software .....	14
5.3 Server systems .....	15
5.4 Network infrastructure .....	15
5.5 Pre-deployment checklist .....	17

<b>6</b>	<b>Planning your deployment</b>	<b>18</b>
6.1	Introduction	18
6.2	Acquiring and validating the root trust anchor	18
6.3	Software deployment and testing	18
6.4	Informing users	19
<b>7</b>	<b>Operational phase</b>	<b>20</b>
7.1	Introduction	20
7.2	Keeping time in sync	20
7.3	Trust-anchor maintenance	20
7.4	Dealing with validation failures	20
7.5	User support questions	21
<b>8</b>	<b>Conclusions</b>	<b>22</b>
<b>Appendix A</b>	<b>DNSSEC Fact Sheet for Users</b>	<b>23</b>
<b>Appendix B</b>	<b>How to configure BIND 9.x DNSSEC validation</b>	<b>24</b>
<b>Appendix C</b>	<b>How to configure Unbound DNSSEC validation</b>	<b>25</b>
<b>Appendix D</b>	<b>How to configure Windows Server 2012 DNSSEC validation</b>	<b>27</b>
<b>Appendix E</b>	<b>Checking your setup</b>	<b>29</b>

# 1 Introduction

## 1.1 Purpose

The Domain Name System (DNS) is a vital part of the core infrastructure of the Internet. From a user's perspective the Internet is broken if DNS malfunctions.

The core functionality of the DNS was designed and implemented in the early 1980s. The system has proven to be robust and resilient during the 30 odd years it has been in service. For more than a decade, however, there have been concerns about the trustworthiness of data in the DNS. These concerns mainly have to do with attacks on so-called recursive caching name servers. Using these attacks, it is possible to forge DNS data that is being served out to client systems. The end result can be misdirection of users to malicious web sites, redirection of e-mail and VoIP traffic and many other problems.

In 1997 action was undertaken to mitigate these vulnerabilities and to add trust to the DNS. This was done by introducing an extension to the DNS protocol set called DNSSEC. This protocol extension has gone through multiple iterations since then and is now considered to be mature.

Rollout of DNSSEC on the Internet has progressed rapidly since 2008 (when a security researcher called Dan Kaminsky published a new attack on DNS caches with serious implications). Much of the core infrastructure of the DNS has now been equipped with DNSSEC support. Also, most DNS software suites now fully support DNSSEC, bringing DNSSEC into the reach of most organisations connected to the Internet.

With these cornerstones in place, the challenge for the coming years becomes the rollout of DNSSEC support to organisations connected to the Internet. There are two aspects to this challenge:

- On the consuming side, owners of caching recursive name servers need to enable DNSSEC validation on their servers.
- On the authoritative side, owners of domain names need to sign the DNS zones associated with these domains using DNSSEC.

This white paper deals with the first challenge, enabling DNSSEC on caching recursive name servers. The goal of this document is to describe the decisions you need to make and the implications of enabling DNSSEC support for your organisation. It contains several checklists to help you in the decision process.

## 1.2 Required knowledge and intended audience

You are assumed to have a basic knowledge of the purpose and structure of the Domain Name System and a basic knowledge of network infrastructure. In-depth knowledge of DNS or DNSSEC is not required.

The intended audience of this document includes IT decision makers, consultants and system integration engineers.

## 1.3 Reading guide

- Chapter 2 describes the cost and benefits of deploying DNSSEC validation

- Chapter 3 explains the architecture of the Domain Name System in terms of the well-known client/server concept; this information will serve as a basis for what is explained in the remainder of the document
- Chapter 4 shows the impact of deploying DNSSEC validation in terms of the scope of the systems that will be involved
- Chapter 5 lists the requirements before you can start to deploy DNSSEC validation
- Chapter 6 helps to provide concrete steps in planning deployment of DNSSEC validation
- Chapter 7 analyses the operational impact of deploying DNSSEC validation
- Chapter 8 concludes this document with a summary of the conclusions that can be drawn from the information provided in this document

## 1.4 Acknowledgements

The author would like to thank Marco Davids from SIDN for reviewing this document and providing insights and suggestions for improvements.

## 1.5 Endorsement SIDN

SIDN recognises the important role DNSSEC plays in furthering the security of the Internet. As such, SIDN is one of the leading group of top-level domains involved in DNSSEC and has deployed DNSSEC in the .nl top-level domain.

The roll-out of DNSSEC validation at ISPs and within enterprise environments is an important step in the overall deployment of DNSSEC. SIDN feels that this white paper can help organisations in planning and performing this deployment and shows that it can be achieved with a minor investment in time and materials.



## 2 Cost and benefits of deploying DNSSEC validation

### 2.1 Introduction

This chapter will discuss the rationale for deploying DNSSEC validation. We will discuss the immediate benefits, what future services can you expect and finish with some words on 'good netizenship'. We will then give an indication of the expected cost.

### 2.2 Trust in DNS answers

The most important benefit of DNSSEC validation is that it protects users effectively against forged DNS responses. Forged DNS responses are both an economic concern (in terms of lost profits, fraud and damage to reputation) as well as a security issue for both users (identity theft, fraud) and enterprises (traffic redirection and interception).

As was already briefly mentioned in the introduction in chapter 1, plain DNS does not provide any guarantees that the data in response to queries is authentic. In fact, a whole host of attacks is known that can be used to forge DNS responses, which may in turn lead to end user systems being misdirected to potentially malicious systems.

The most important feature of DNSSEC is that it adds a mechanism to the DNS that can prove the authenticity of DNS responses. To achieve this, digital signatures on DNS records are included in the responses.

Recipients of DNS data, such as recursive caching name servers, can validate these digital signatures thus proving that the DNS data is authentic.



### 2.3 Derived services

A second and equally important benefit is that DNSSEC opens up possibilities for additional services based on the trust that DNSSEC introduces.

There is a whole range of initiatives that aim to leverage the authenticity DNSSEC brings to the DNS. Examples include:

- Storing SSL certificates or references to certificates in DNS to provide additional ways to check their authenticity outside of the scope of the PKI in which they were issued.
- Adding records to DNS zones that indicate which Certificate Authorities may issue certificates for a domain.
- Storing secure shell (SSH) fingerprints in DNS that help in authenticating server systems to system administrators.
- Storing (references to) Identity Federation meta-data in the DNS.

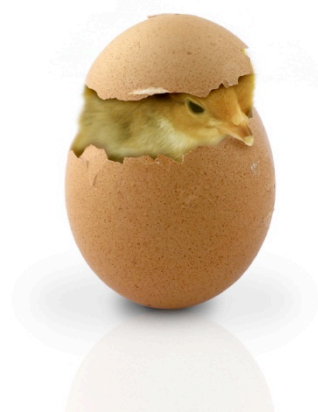
## 2.4 'Good netizenship'

The DNS is critical Internet infrastructure and as such, making sure that it is trustworthy is of key importance. DNSSEC is a technology that plays an important role in this respect; it adds authenticity to the DNS.

Like many technologies DNSSEC has suffered from the chicken-and-egg problem; why would anyone add DNSSEC signatures to their domain if nobody validates these signatures?

Fortunately, DNSSEC deployment across the Internet has taken off, especially on the authoritative signing side. The root zone has been signed and DNSSEC signing is now within the reach of the large majority of domain owners thanks to the efforts of the largest top-level domain registries.

For DNSSEC to be of use to end users, however, the validating side also requires attention. This means that operators of caching recursive name servers, including you, should enable validation on their servers. Some large organisations have already done this and have paved the way by ironing out the initial difficulties that enabling validation caused. The technology surrounding DNSSEC validation is mature and it is now up to you to show 'good netizenship' and to deploy DNSSEC validation on your caching recursive name servers.



## 2.5 Cost

The cost of deploying DNSSEC validation is usually low to very low. As will become clear in the rest of this document, it requires little to no investment in terms of hardware and software and only requires a small investment in time for your system administration department. We estimate that a typical deployment can be completed in as little as a week of effort for experienced system administrators (depending – of course – on the size of the deployment in terms of the number of servers and end users dependent on these servers).

## 2.6 Conclusion

Given the significant benefits and the low cost, we strongly advise you to deploy DNSSEC validation.



## 3 DNS architecture

### 3.1 Introduction

In this chapter, we explain DNS in terms of the well-known client/server paradigm. This information serves as a basis for understanding the rest of the document. If you are intimately familiar with the way the DNS works you can safely skip this chapter.

### 3.2 Client/server view on DNS

From a high-level architecture perspective the DNS can be classified as a highly distributed client/server database system. Figure 1 below shows this perspective:

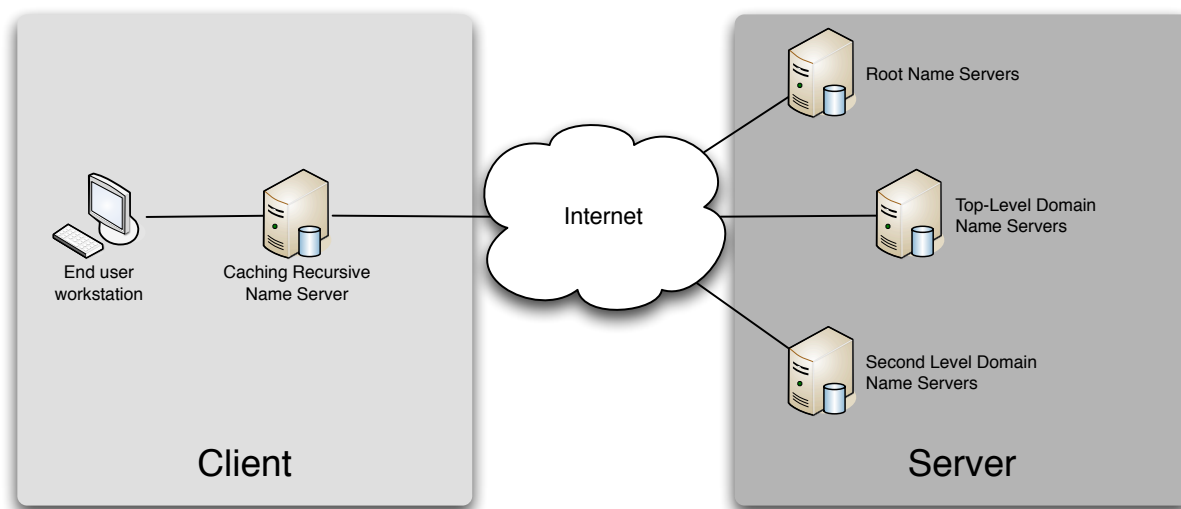


Figure 1 - Client/server view on DNS

The left-hand side of the figure shows the client side of DNS. This client side consists of end-user systems and caching recursive name servers often referred to as resolvers. The role of the client side in the DNS is described in §3.3. The right-hand side consists of authoritative name servers that are each responsible for one or more domains. The role of the server side in the DNS is described in §3.4.

This white paper deals with implementation of DNSSEC on the client side.

### 3.3 The client side of the DNS

On the client side of the DNS two roles can be identified:

- The role of end user systems. These are the systems that rely on DNS data to access resources on the Internet, for example the mapping of a logical name like `www.example.com` to an IPv4 or IPv6 address.
- The role of caching recursive name servers. These are the systems that gather the information required to answer a question from an end user system.

In most cases, both roles are performed by separate systems. A single caching recursive name server or set of such servers usually provide DNS services to multiple end user systems (as shown in Figure 2 below).

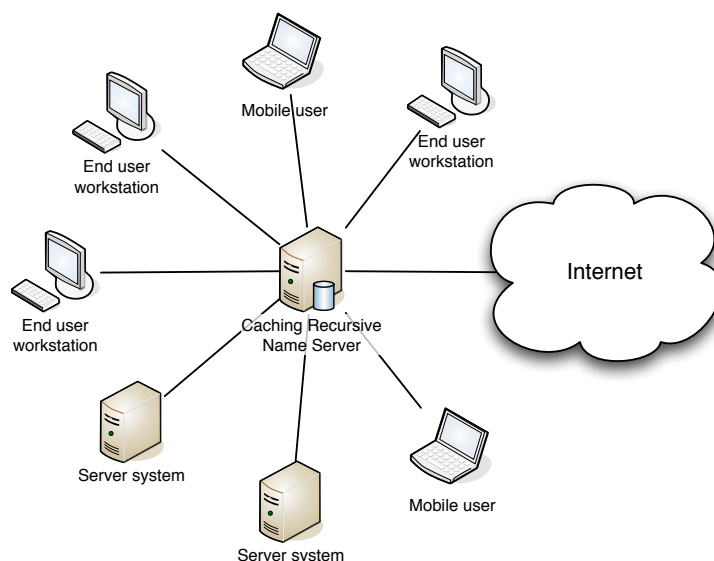


Figure 2 - The two roles on the client side of DNS

End user systems in this case are not only workstations but also other servers that rely on DNS information.

The advantage of having a caching recursive name server is that it maintains an internal database of answers to all the queries it performed on behalf of end user systems. This means that it can serve out a cached answer to an end user system in cases where it already knows the answer.

End users typically see the recursive caching name server configured as 'DNS server' on their workstation. Examples can be seen in Figure 3

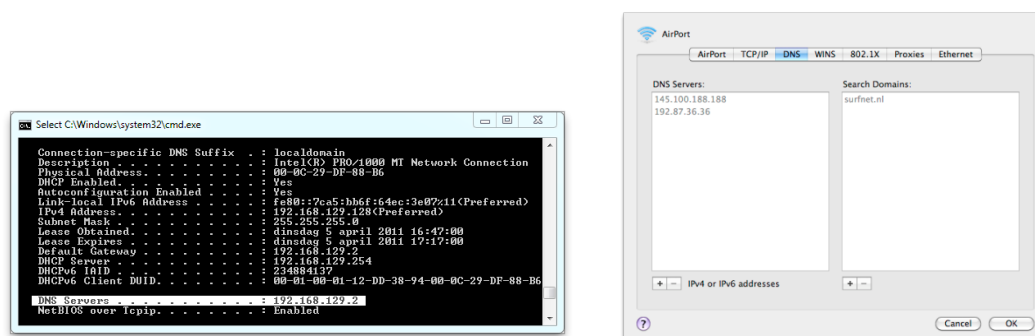


Figure 3 – Example DNS configuration on Windows 7 and Mac OS X

### 3.4 The server side of the DNS

On the server side the DNS can be viewed as a vast distributed database. DNS names are written in 'dot' notation (i.e. www 'dot' example 'dot' com – www.example.com). Each 'dot' is the boundary of a name space called a domain that is managed by so-called authoritative name servers. These authoritative name servers keep the data for a domain in what is called a zone. For example: for www.example.com this breaks down as follows:

- The root name servers (signified by a single dot at the end of the name which is usually omitted) are authoritative for the root zone; the root zone contains references to all top-level domain name servers. The root is managed by IANA.
- The .com top-level name servers; these servers are authoritative for the contents of the .com zone and they contain references to name servers for all sub-domains of .com. These servers are managed by VeriSign.
- The example.com name servers; these servers are authoritative for the contents of the example.com zone containing records for all hosts in the example.com domain. In the example above, www.example.com, these servers have information about the mapping of the host name 'www' to an IP address.

To determine the answer to a query, a client (in most cases a recursive caching name server) will traverse the DNS hierarchy from the root down as shown in Figure 4 below:

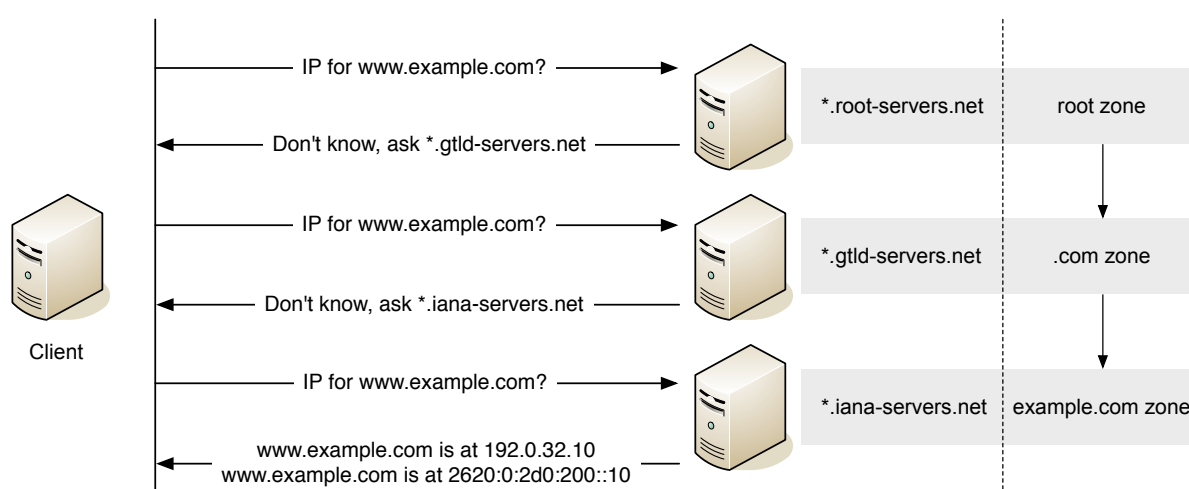


Figure 4 - Traversing the DNS for `www.example.com`

### 3.5 Combining the client and server side of DNS

Some organisations combine the client and server side of DNS on a single server or a group of servers. This may be for historic reasons (this was more common during the early years of the Internet) or it may be because the terminology surrounding DNS can be misleading and may give the impression that 'a name server is a name server'.

It is out of the scope of this paper to describe in detail why it is bad practice<sup>1</sup> to combine both sides on a single server, but if this is currently the case in your organisation you should urgently consider changing this. We do not recommend proceeding with the roll-out of DNSSEC validation on your servers if you have such a set up and urge you to separate the client and server roles of DNS first.

<sup>1</sup> For some arguments, see – for example – the BIND manual <https://www.isc.org/files/arm97.pdf> in section 1.4.6

## 4 Scope

### 4.1 Introduction

This chapter identifies the parts of your infrastructure that are affected by deploying DNSSEC validation and the impact it will have on these components.

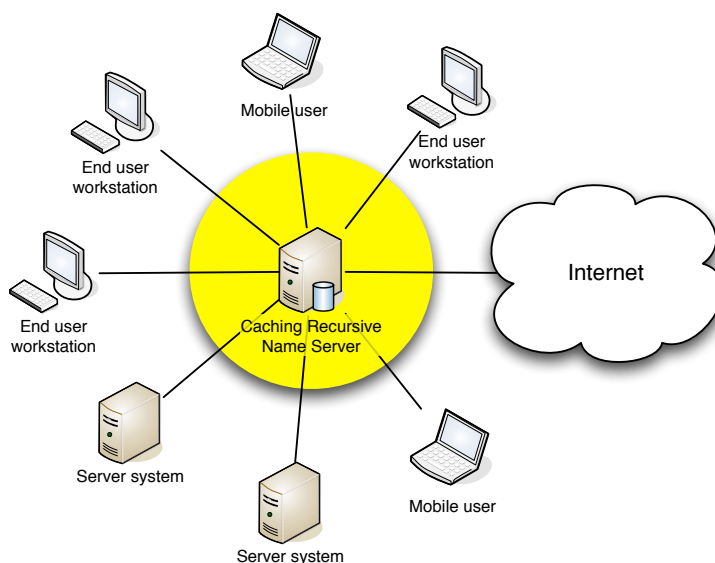
After reading this chapter you will:

- Know which parts of your infrastructure require attention when deploying DNSSEC validation
- Know the current drawbacks of deploying DNSSEC validation all the way to the end user

### 4.2 Caching recursive name servers

Caching recursive name servers are the most important part of a DNSSEC validation deployment. These servers will play the most important role because they will be validating the answers to DNS queries submitted by the clients depending on them for DNS resolution.

Figure 5 shows how caching recursive name servers are positioned relative to the other systems involved:



**Figure 5 - Position of caching recursive name servers**

It is very important to note that deployment of DNSSEC validation will only be successful if all caching recursive name servers used by a group of client systems have validation enabled. For instance, if your organisation operates two caching recursive name servers that are used by end user workstations and supplied to them when they are configured through DHCP, then both these servers will need to have DNSSEC validation enabled in order for the deployment to be useful. The reason for this is that clients will try to get queries answered at all configured recursive caching name servers; thus, if one or more of the configured caches does not perform DNSSEC validation then clients are not adequately protected.

### 4.3 End user systems

It is usually not necessary nor is it possible to deploy DNSSEC on end user systems. There are multiple reasons for this:

- End users may be behind (additional) firewalls that have problems with DNSSEC traffic
- CPE<sup>2</sup> equipment, especially in case of DSL or cable CPE equipment, is often not prepared for DNSSEC support
- The added value of validation on end users systems is relatively low; there is no real business case for attacking individual end user systems using DNS cache poisoning, it is much more lucrative to attack caching recursive name servers since these potentially server 100000s of users

The current recommendation therefore is that you should not invest in deploying DNSSEC on individual end user systems. Figure 6 shows the position of end users systems relative to the other systems involved:

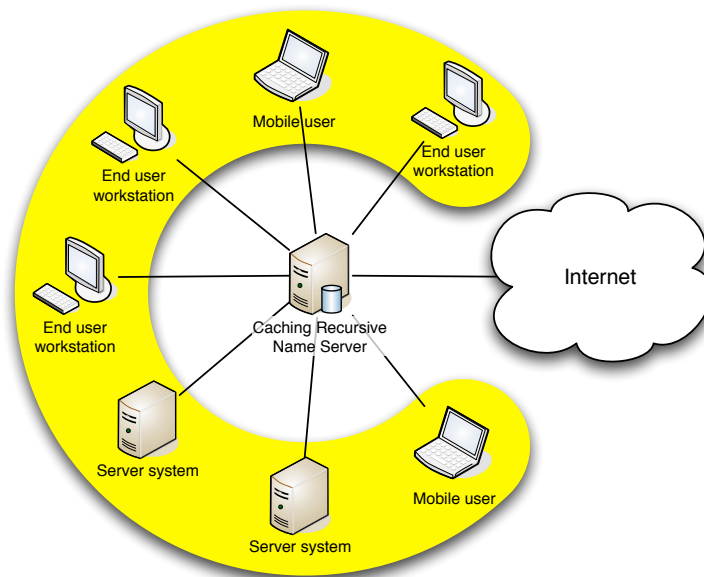


Figure 6 - Position of end user systems

---

<sup>2</sup> Customer Premise Equipment, e.g. a customer router or an ADSL modem

## 5 Requirements before deployment

### 5.1 Introduction

This chapter will guide you through the requirements that need to be met before you can deploy DNSSEC validation on your caching recursive name servers. At the end of the chapter you will find a useful pre-deployment checklist that you can use in your deployment plan.

### 5.2 Software

DNSSEC validation is supported by most mainstream DNS server solutions. This paragraph contains specific information about the three most commonly used DNS caching recursive name server packages.

#### 5.2.1 BIND

BIND (the Berkeley Internet Name Daemon) is the most popular DNS server on the Internet. It is maintained by the Internet Systems Consortium (ISC, <http://www.isc.org>).

BIND has had support for DNSSEC built-in since version 9 was released in 2000. DNSSEC has evolved over the years and the new features have all been included in BIND.



If your DNS infrastructure is based on BIND, it is recommended that you run at least version 9.7 of BIND to ensure that the latest version of the DNSSEC protocol suite is properly supported. With the introduction of BIND 9.7 many features were added that simplify management of a DNSSEC validating caching recursive name server. If you are running an older version we recommend that you consider upgrading to version 9.7 or up before deploying DNSSEC validation.

#### 5.2.2 Unbound

Unbound was developed from the ground up as a pure caching recursive name server with DNSSEC validation support. It is maintained by NLnet Labs (<http://www.nlnetlabs.nl>).



All versions of Unbound support DNSSEC validation out-of-the-box. We recommend that you use at least version 1.4 or up since features were included in this version that simplify management of a DNSSEC validating caching recursive name server.

### 5.2.3 Microsoft DNS

Microsoft Windows Server has DNSSEC support for both authoritative DNS as well as caching recursive name servers. DNSSEC support was introduced in Windows Server 2008 R2.

We do not recommend operating a DNSSEC validating caching recursive name server with Windows Server 2008 R2 because the DNSSEC implementation in Windows Server 2008 R2 is limited in functionality and does not support the most modern version of the DNSSEC protocol suite that is commonly used on the Internet.

If you want to operate a DNSSEC validating caching recursive name server, we recommend that you use Windows Server 2012. This version of Windows has full support for all modern algorithms in the DNSSEC protocol suite and also offers automated key tracking for the root zone (something that is also missing in Windows Server 2008 R2).

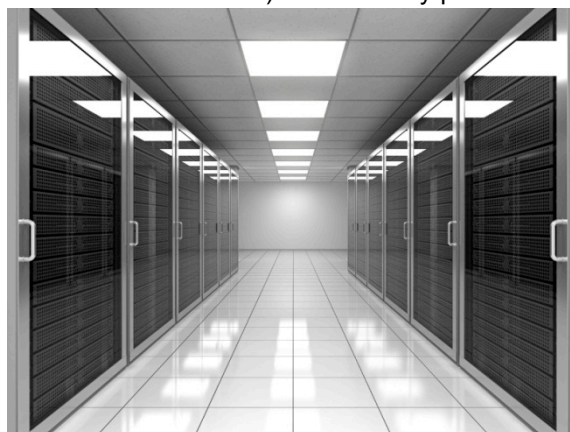
## 5.3 Server systems

### 5.3.1 Physical hardware

DNSSEC relies on public key cryptography to create the digital signatures that guarantee the authenticity of records. Public key cryptography is by its nature computationally intensive, meaning that it requires an above average amount of CPU power to run smoothly. In the past this has for instance lead to the introduction of hardware cryptography accelerators.

Research has shown that modern server hardware (manufactured after 2005) is sufficiently powerful to operate a DNSSEC validating caching recursive name server. This means that no additional hardware acceleration is required and off-the-shelf products can be used. It is very likely that no upgrades to your existing hardware platform are required.

Only if you have very old server hardware (pre 2005) should you consider investing in new hardware for your DNS infrastructure.



### 5.3.2 Virtual machines

Many organisations nowadays choose to deploy virtual machine infrastructures rather than relying on individual physical hardware for servers. It is not uncommon for caching recursive name servers to be operated on virtual machines.

Tests have shown that it is perfectly feasible to operate a DNSSEC validating caching recursive name server on a virtual machine. Depending on the amount of traffic the server processes assigning more than one CPU core to the virtual machine may be required.

## 5.4 Network infrastructure

When you deploy DNSSEC validation on your caching recursive name servers you need to take special care making sure that your network infrastructure is compatible. DNSSEC adds digital signatures to DNS response packets. This has a significant impact on the size of these packets.

Prior to the introduction of DNSSEC, DNS response packets rarely exceeded 512 bytes in size. For DNSSEC signed domains this is no longer the case; packets can be much larger and regularly exceed 1500 bytes in size. This has consequences for your network infrastructure in several ways that will be explained in the following paragraphs.



Even if you do not plan to deploy DNSSEC validation on your caching recursive name servers in the short term you may still experience problems. Almost all current versions of mainstream DNS software for caching recursive name servers automatically requests DNSSEC data even if no validation occurs. This means that your servers may experience problems regardless of whether or not DNSSEC validation is enabled.

#### 5.4.1 DNS over TCP

Traditionally, the DNS relies on the UDP protocol to transmit queries and responses. DNS has, however, always had the option to use the TCP protocol as well. If, for example, a DNS response exceeds the maximum allowed packet size or if a DNS zone transfer takes place TCP may be used.

Many security auditor firms and firewall vendors claim that DNS does not use TCP and as a consequence of that claim that traffic to port 53 – the DNS server port – using the TCP protocol should be blocked. Even though this has always been incorrect, this did not necessarily lead to problems since DNS packets rarely exceeded 512 bytes, which meant that it was very rare that a fallback to TCP was required.

With the introduction of DNSSEC this has changed. It is now very likely that larger DNS packets need to be exchanged and if the data to be transmitted does not fit in the maximum UDP packet size that the caching recursive name server is configured to support then a fallback to TCP will automatically occur. Thus, if TCP for DNS is blocked on your firewall this may lead to serious name resolution issues on your caching recursive name server.

We recommend that you always check with your firewall vendor and system administrators to ensure that DNS over TCP is allowed on your network and you should disregard security auditors claiming that DNS traffic never uses the TCP protocol.

#### 5.4.2 UDP packet size

DNSSEC relies on an extension to the DNS protocol introduced in 1999 called EDNS0. Among other things this extension makes it possible to use UDP packets larger than 512 bytes to transmit DNS responses.

By default many DNS software solutions are configured with a default EDNS0 packet size limit of 4KB. This means that the DNS software indicates to other DNS servers that it can receive DNS packets up to 4KB in size.

As a result of this, your network equipment will need to be able to deal with large UDP packets. This can be an issue for two reasons:



- Some firewall systems are configured such that DNS UDP packets larger than 512 bytes are discarded because they are assumed to be an attack.
- Some firewall systems refuse to accept fragmented UDP packets, again because these are assumed to be an attack.

In both cases you should attempt to reconfigure the firewall such that these restrictions are lifted. Both settings have very little – if any – security benefits and seriously impair the functioning of any caching recursive name server behind the firewall that performs DNSSEC validation or has EDNS0 enabled (which – as mentioned in the introduction to this section – almost all modern DNS solutions have by default).

If it is not possible to reconfigure or replace firewall systems that enforce these restrictions you can consider reconfiguring your caching recursive name server. In almost all cases (with the exception of Windows Server 2008 R2<sup>3</sup>) it is possible to configure the maximum packet size supported by the server. A consequence of this is that TCP fallback will occur if a response does not fit in the configured packet size. This will result in additional load on both your caching recursive name server as well as the authoritative name servers it queries on the Internet. This is generally considered not to be a good thing so if you can prevent this by replacing or reconfiguring your firewall you should seriously consider doing so.

There are good tools available to check the maximum DNS UDP packet size from your network, for instance <https://www.dns-oarc.net/oarc/services/replysizetest>.

## 5.5 Pre-deployment checklist

The table below is a checklist that can help you to plan your deployment by ticking off the requirements set out in this chapter:

Requirement	Check
Software supports DNSSEC	
BIND version 9.7 or up	
Unbound version 1.4 or up	
Microsoft Windows Server 2012 or up	
Server systems are modern enough	
Network infrastructure can deal with DNSSEC requirements	
DNS over TCP is allowed	
Large UDP packets are allowed	

---

<sup>3</sup> At the time of writing of this document, it was unclear whether this feature will be present in Microsoft Windows Server 2012

## 6 Planning your deployment

### 6.1 Introduction

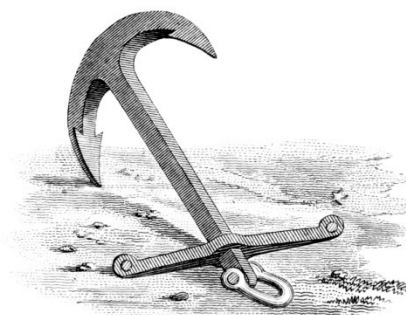
Once you have ensured that the requirements set out in chapter 5 are met you are ready to start planning your deployment. In this chapter we will discuss the most important issues that you should address in your plan.

### 6.2 Acquiring and validating the root trust anchor

As was already mentioned in section 2.2 the goal of DNSSEC is to add a trust mechanism to DNS. In DNSSEC, trust is expressed hierarchically from the root zone down to top-level domains and further down to registered domain names.

Recursive caching name servers need to be configured with what is called a trust anchor in order for them to be able to validate the digital signatures used by DNSSEC. It is thus of key importance that you check the validity and integrity of the root trust anchor before you configure it on your servers.

There are several ways to obtain the root trust anchor (some of which are described in the appendices of this white paper). Once you have obtained the trust anchor, you can use one or more of the following methods to ascertain its validity:



- Obtain a copy of the root zone trust anchor from IANA's<sup>4</sup> website. Information about DNSSEC in the root zone can be found on <https://www.iana.org/dnssec>. Please ensure that you acquire the trust anchor information over a TLS secured link; the default link on the page redirects to a HTTP site, changing the URL to HTTPS will get you to the TLS-secured site. Once you have obtained a copy, you can compare the data to the trust anchor you have.
- Validate one or more of the digital signatures created by the Community Representatives that attended one or more root zone ceremonies, see for instance <https://dnssec.surfnet.nl/?p=371>.
- Retrieve the root trust anchor from multiple independent sources (e.g. multiple root servers) and compare the results to ascertain that each server has returned the same trust anchor.

### 6.3 Software deployment and testing

The next step should be to deploy and configure the new software on your caching recursive name servers. Again, DNSSEC is meant to add trust to the DNS so make sure that you check the integrity of any new software that you need to deploy in order to support DNSSEC validation.

---

<sup>4</sup> IANA, the Internet Assigned Numbers Authority, is responsible for managing the root zone of the DNS.

It is also prudent to deploy the software on test servers first or to use completely new servers for your new validating resolver setup. Once you have done this, you can start testing your setup. Simple tests are described in the appendices to this white paper.

Finally, to conclude the testing phase, it may be helpful to test with a select group of users by configuring their clients to use the new or test servers.

## 6.4 Informing users

Once your setup is ready and has been tested to your satisfaction you are ready to roll it out in production. We strongly recommend that you inform your users of the rollout of DNSSEC.

To help you inform users, we have included a simple fact-sheet that you can distribute as Appendix A.



## 7 Operational phase

### 7.1 Introduction

In this chapter we will address the operational consequences of deploy DNSSEC validation.

### 7.2 Keeping time in sync

The DNSSEC protocol has a strong dependency on the system clock of your caching recursive name server. DNSSEC signatures have a set validity, starting at an inception time and ending at an expiration time. If the clock of your server is (severely) out of sync, your validating caching recursive name server may treat signature data incorrectly (either accepting it as valid when it has expired or incorrectly treating it is invalid).



We highly recommend that you ensure that the server clock is kept updated using an external time source such as a Network Time Protocol (NTP) server<sup>5</sup>. This is even more important if you run your validating caching recursive name server on a virtual machine.

### 7.3 Trust-anchor maintenance

As you may know, X.509 certificates and their associated keys have a limited validity and are renewed regularly (e.g. at 1 or 3 year intervals). Operators of DNSSEC signed zones may have a similar policy resulting in DNSSEC keys being rolled over and replaced by new keys. This also applies the root trust anchor that you will have used to configure your resolver. In fact, the current DNSSEC signing policy for the root zone requires that the key that you use as a trust anchor is rolled every 5 years<sup>6</sup>.

Unless you are using Microsoft Windows Server 2008 R2 (not recommended), your validating caching recursive name server software will automatically keep the root trust-anchor up-to-date. The only exception to this is when your server is offline for more than a couple of days. If a key rollover is initiated during this offline period then there is a chance that your server may either miss the rollover or may not accept the new key. For this reason, you should make sure that your operational manuals include guidelines for dealing with situations where servers are offline for extended periods of time.

### 7.4 Dealing with validation failures

When you enable DNSSEC validation on your caching recursive name server you are going to see validation failures in the log files. This is nothing to worry about; they are commonly caused by people making errors in their DNSSEC signed domain (such as forgetting to re-sign their zone information).

---

<sup>5</sup> It may be prudent to configure the NTP server using its IP address rather than its DNS name

<sup>6</sup> See <https://www.iana.org/dnssec/icann-dps.txt>, section 6.4

In most cases you can ignore validation failures. There are, however, some exceptions. If you see large numbers of validation failures occurring for one specific top-level domain this may indicate a serious issue with this top-level domain. In these cases, a whole top-level domain may become unreachable, which will quickly lead to a severely degrade Internet experience for your users. We recommend that you check resources like the DNSSEC Deployment mailing list to see if others are experiencing the same problems. If this is the case, you may decide to temporarily disable DNSSEC validation for the top-level domain that is experiencing problems. Most common software products, such as BIND and Unbound, support this.



## 7.5 User support questions

Experience reports by people currently operating validating caching recursive name servers show that enabling validation does not lead to an increase in user helpdesk calls. Nevertheless, it is sensible to train your helpdesk staff with respect to DNSSEC related questions. They should be especially alert to people calling in problems that all pertain to DNS name resolution errors in one top-level domain as this may indicate a DNSSEC issue with a top-level domain (see also §7.4).

## 8 Conclusions

DNSSEC rollout is progressing steadily on the Internet. Deployment of validating caching recursive name servers is an important part of this. As we have shown in this document deploying DNSSEC validation provides significant benefits at a minimal cost.

Name server software is mature enough to deploy validating caching recursive name servers very easily. We have provided information that should help you plan the rollout of validation on your own name server infrastructure. Finally, we believe that the benefits of deploying validating caching recursive name servers now outweigh the risks; past experience of early adopters has shown that the impact on infrastructure and user support is very low.

For these reasons, we strongly recommend that you start planning deployment of validation on your caching recursive name server infrastructure and join the growing group of organisations that are involved in the rollout of DNSSEC on the Internet.

## Appendix A DNSSEC Fact Sheet for Users

### What is DNS?

The Domain Name System (DNS) can be compared to a telephone directory. Just like your telephone cannot deal with written names and requires telephone numbers to work, computers connected to the Internet cannot deal with written names either. They require Internet Protocol (IP) addresses, which are unique numeric codes that identify computers on the Internet.

Humans are much better at dealing with names than with numbers, which is why telephone directories exist. The Domain Name System provides a similar service; it maps human readable names (like [www.isoc.org](http://www.isoc.org)) to IP addresses.

### What is DNSSEC?

By and of itself, DNS has no provisions for proving the authenticity of information. Just like you can forge a telephone number in a directory (e.g. by changing the information on your corporate intranet site) it is possible to falsify DNS data.

This is a problem since the value of DNS data has grown with the advent of the modern Internet. Many day-to-day activities like banking, shopping and telephone calls rely on the Internet – and thus the Domain Name System.

To improve the security of the DNS, the DNS Security Extensions (DNSSEC) have been introduced. DNSSEC proves the authenticity of DNS data using digital signatures.

### How does DNSSEC protect me?

When you surf the Internet, your browser uses a DNS server (also referred to as a name server or a DNS resolver) to resolve IP addresses from the names you enter in the browser or the names contained in – for example – the hyperlinks on a web page.

This DNS server uses the digital signatures that DNSSEC introduces to verify the authenticity of the data it receives from the Internet. If a record has been falsified, the DNS server will detect this because the digital signature verification will fail.

### What do I have to do as a user?

As a user, you do not have to make any changes to your system configuration. Your system administrator will make all the required changes on the DNS server used by your computer.

### What will I notice once DNSSEC has been enabled?

It is unlikely that you will notice anything once DNSSEC has been enabled. The only thing you will notice is when your DNS server notices an incorrect digital signature on DNS data. In this case, you will most likely see a blank page in your web browser and an error message stating that the server you are looking for cannot be found. Unlike the pop-ups you may occasionally see warning you of SSL certificate errors, there is no way to ignore DNSSEC validation errors.

## Appendix B How to configure BIND 9.x DNSSEC validation

### Prerequisites

Before you follow these instructions to configure your BIND 9.x caching recursive name server to enable DNSSEC validation, please take a minute to ensure that the following pre-requisites are met:

- You have BIND 9.7 or up installed
- Your BIND installation is properly configured as a caching recursive name server and is fully functional
- You have obtained and validated the root trust anchor
- You have a client machine for your caching recursive name server with a working browser

### Step 1 – Adding the root trust anchor

Open your BIND configuration file (usually *named.conf*) in an editor. Add the following section to the configuration (this is a new configuration section that is not part of any existing configuration sections):

```
managed-keys {  
    "." initial-key 257 3 8 "<root-trust-anchor-data>";  
};
```

Please make sure that the key flags and algorithm correctly match the data you obtained with the root trust anchor; the root trust anchor should always have the flags set to 257.

Substitute *<root-trust-anchor-data>* with the actual trust anchor, which should look something like this: "AwEAAgAIKIVZrp..."

### Step 2 – Enabling validation

In the *options* section of the BIND configuration, add the following two lines:

```
dnssec-enable yes;  
  
dnssec-validation yes;
```

### Step 3 – Restarting BIND

The configuration should now be correct. Save the configuration file and close the editor. You are now ready to restart BIND, please use the operating system-specific command to achieve this, for example by invoking:

```
# /etc/init.d/named restart
```

Check the log file to make sure that BIND starts up correctly. You may see warnings about the fact that no "managed keys" file exists. This is normal the first time you launch BIND; it will need to obtain the current root trust anchor first to match it against the configuration. The "managed keys" file will then be created automatically.

### Step 4 – Checking your setup

Please refer to Appendix E.



## Appendix C How to configure Unbound DNSSEC validation

### Prerequisites

Before you follow these instructions to configure your Unbound caching recursive name server to enable DNSSEC validation, please take a minute to ensure that the following pre-requisites are met:

- You have Unbound 1.4.0 or up installed
- Your Unbound installation is properly configured as a caching recursive name server and is fully functional
- You have obtained and validated the root trust anchor
- You have a client machine for your caching recursive name server with a working browser

### Step 1 – Creating a trust anchor file for the root trust anchor

Obtain and **validate** (see §6.2) the root trust anchor; you need to obtain the entire DNSKEY record for the root trust anchor. Create a blank file and copy the DNSKEY record for the root trust anchor into it. You can also directly perform this action with the following command:

```
$ dig DNSKEY . | grep 257 > ./root-anchor
```

Make sure that the file contains the correct anchor by checking its contents; it should contain a single DNS resource record of type DNSKEY:

```
$ cat ./root-anchor
.          151328  IN      DNSKEY  257 3 8
AwEAAgAIIkVZrpC6Ia7gEzahOR+9W29euxhJhVVLOyQbSEW0O8gcCjF
FVQUTf6v58fLjwBd0YI0EzrAcQqBGCzh/RStIoO8g0NfnfL2MTJRkxoX
bfDaUeVPQuYEhg37NZWAJQ9VnMVDxP/VHL496M/QZxkjf5/Efucp2gaD
X6RS6CXpoY68LsvPVjR0ZSwzzlapAzvN9dlzEheX7ICJBBtuA6G3LQpz
W5hOA2hzCTMjJPJ8LbqF6dsV6DoBQzgul0sGicGOYl7OyQdXfZ57relS
Qageu+ipAdTTJ25AsRTAoub8ONGcLmqRAmRLKBP1dfwhYB4N7knNnulq QxA+Uk1ihz0=
```

(the output above is an example)

### Step 2 – Configure Unbound for DNSSEC validation

Open the Unbound configuration (usually called *unbound.conf*) in an editor. Locate the configuration item called *module-config*. It should look like this:

```
module-config: "iterator"
```

Modify it to look like this:

```
module-config: "validator iterator"
```

Next, add an extra statement to the configuration file to point Unbound towards the file containing the root trust anchor (note that this file must be writeable for the UNIX user Unbound runs as):

```
auto-trust-anchor-file: /path/to/root-anchor
```

Replace */path/to/root-anchor* with the path to the actual file.

### Step 3 – Restarting Unbound

The configuration should now be correct. Save the configuration file and close the editor. You are now ready to restart Unbound, please use the operating system-specific command to achieve this, for example by invoking:

```
# /etc/init.d/unbound restart
```

Check the log file to make sure that Unbound starts up correctly.

#### **Step 4 – Checking your setup**

Please refer to Appendix E.

## Appendix D How to configure Windows Server 2012 DNSSEC validation

### Prerequisites

Before you follow these instructions to configure your Microsoft Windows Server 2012 caching recursive name server to enable DNSSEC validation, please take a minute to ensure that the following pre-requisites are met:

- You have installed the latest updates and service packs
- Your Windows Server 2012 installation is properly configured as a caching recursive name server and is fully functional
- You have a client machine for your caching recursive name server with a working browser

### Step 1 – Enabling DNSSEC validation

Microsoft Windows Server 2012 has a built-in command-line tool to configure the built-in DNS server component called **dnscmd**. This command allows administrators to enable DNSSEC validation with a single operation. To execute the command, launch Windows PowerShell **as Administrator**<sup>7</sup> and enter the following:

```
C:\>dnscmd /RetrieveRootTrustAnchors
```

You should see the following output (answer “yes” to the question):

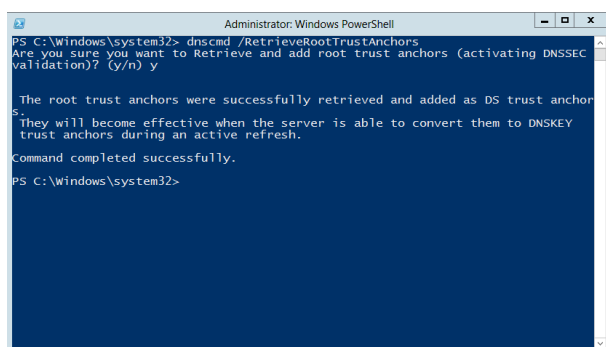


Figure 7 - Output after Windows has retrieved trust anchors

As the output indicates, it may take some time before DNSSEC validation is actually activated. You can manually trigger an active refresh by invoking the following command:

```
C:\>dnscmd /ActiveRefreshAllTrustPoints
```

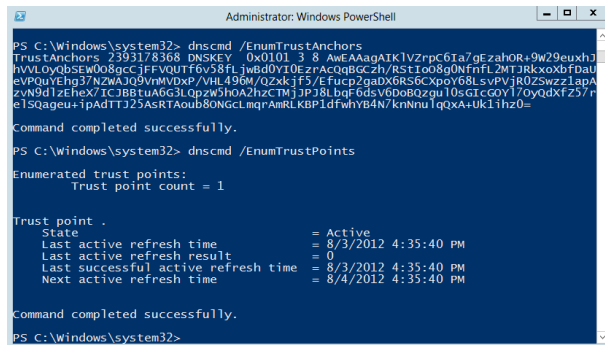
Finally, you can check whether Windows has retrieved the trust anchors by invoking the following commands:

```
C:\>dnscmd /EnumTrustAnchors
...
C:\>dnscmd /EnumTrustPoints
```

---

<sup>7</sup> By right clicking on Windows PowerShell and selecting “Run as Administrator”

The output should look like this:

A screenshot of a Windows PowerShell window titled 'Administrator: Windows PowerShell'. The window shows the output of two commands. The first command, 'dnscmd /EnumTrustAnchors', displays a long string of base64-encoded data representing trust anchors. The second command, 'dnscmd /EnumTrustPoints', shows that there is one trust point, which is active and has a last active refresh time of 8/3/2012 4:35:40 PM.

```
PS C:\Windows\system32> dnscmd /EnumTrustAnchors
TrustAnchors: 2393178368 DNSKEY 0x0101 3 8 AWEAAagATK1VZrpC6Ta7gEzahOR+9w29euxh3
hVVL0yQbSEu008gccjFFVQUTf6v58fLjwBd0YIOEzrAcqBGCzh/rstIo08g0NfnFL2MTJRkxoxbFdaU
eVPQuYEhg37NZWAJ09vrmVdXP/VHL496W/QZxkjF3/Efucp2gaDX6RS6CXpoyV68LsvPVjR0ZSwzz1apA
zyN9d1zeHeX7ICJBBtuA0G3LQpzw3h0A2h2CTMj3P38LbqF6dsV6D08Q2gu10s6IcGOY17OyQdXfZ3/r
e1SQageu+ipAdTTJ25ASRTAoub8ONGCLmqAmRLKBP1dfwYB4N7krNnu1qQxA+uk1ihz0=

Command completed successfully.

PS C:\Windows\system32> dnscmd /EnumTrustPoints
Enumerated trust points:
    Trust point count = 1

Trust point .
State = Active
Last active refresh time = 8/3/2012 4:35:40 PM
Last active refresh result = 0
Last successful active refresh time = 8/3/2012 4:35:40 PM
Next active refresh time = 8/4/2012 4:35:40 PM

Command completed successfully.
PS C:\Windows\system32>
```

Figure 8 - Output showing that DNSSEC trust anchors have been configured

## Step 2 – Checking your setup

You can use the steps described in Appendix E to check your setup.

## Appendix E Checking your setup

### Using command-line tools

You can check your setup using command-line tools such as *dig*, which is included with BIND and commonly available on UNIX systems. Alternative tools that you can use are the *host* command on UNIX and the *nslookup* command on Windows. The examples below assume that you are using *dig*.

#### Test 1 – positive test with a command-line tool

The goal of this test is to prove that your validating caching recursive name server correctly validates the DNS answers for DNSSEC-signed domains. In this test, we will send a query for the A record for *www.surfnet.nl* and check – by looking at the flags that were returned – whether the answer has been validated.

On the command-line, execute the following command:

```
$ dig +dnssec +noauthority +noadditional A www.surfnet.nl
```

The answer should look something like this:

```
; <<>> DiG version <<>> +dnssec +noauthority +noadditional A www.surfnet.nl
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24622
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 6, ADDITIONAL: 13

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;www.surfnet.nl.                IN      A

;; ANSWER SECTION:
www.surfnet.nl.                658     IN      A      194.171.26.203
www.surfnet.nl.                658     IN      RRSIG  A 8 3 3600 20110621225534 20110615070013
30160 surfnet.nl. 9+QfZyefAz1iq6lwr1l+rn1n+47pVPkfZ8QnhkJjv0PqYEJXAxlvpMDX
3+1nfCijNPac01Q82UJ2Y60ZGhBCufLrM3ezeOWl9q9CEiWak8ryTFnN
pYzIvtkSz6KHlUql6yelNAU+ijtc6qQEhgrQNljmhSqmmmp+Ir8f2T4iN XM4=

;; Query time: 14 msec
;; SERVER: server
;; WHEN: Fri Jun 17 16:55:20 2011
;; MSG SIZE rcvd: 1659
```

The key thing to look for in the answer is the AD (Authenticated Data) flag; this flag will only be set if the resolver validated the answer it got from the authoritative name server (it is marked bold with yellow background in the example above).

#### Test 2 – negative test with a command-line tool

The goal of this test is to prove that your validating caching recursive name server discards data with invalid signatures. In this test, we will send a query for the A record for *www.rhybar.cz* and check that the server provides no answer and returns the correct error code.

On the command-line, execute the following command:

```
$ dig +dnssec +noauthority +noadditional A www.rhybar.cz
```

The answer should look something like this:

```
; <<>> DiG 9.6.0-APPLE-P2 <<>> +dnssec +noauthority +noadditional A www.rhybar.cz
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 32343
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;www.rhybar.cz.                IN      A

;; Query time: 4 msec
;; SERVER: 145.100.188.188#53(145.100.188.188)
;; WHEN: Fri Jun 17 17:02:47 2011
;; MSG SIZE rcvd: 42
```

As you can see in the example output, no answers is provided to the query and the server returns the error code “SERVFAIL” (marked in bold with yellow background).

### Testing with a browser

There are several sites that offer you the possibility to check whether or not DNSSEC is enabled. Examples are:

- <http://dnssectest.sidn.nl/test.php>
- <http://www.nic.cz/dnssec/>
- <http://dnssec-or-not.org/>

Their output should show you whether or not the caching recursive name server that your browser uses properly validates DNSSEC answers. Note that these checks will only work if **all** name servers configured on your computer perform DNSSEC validation.

Example output of the first check in the list above looks like this:

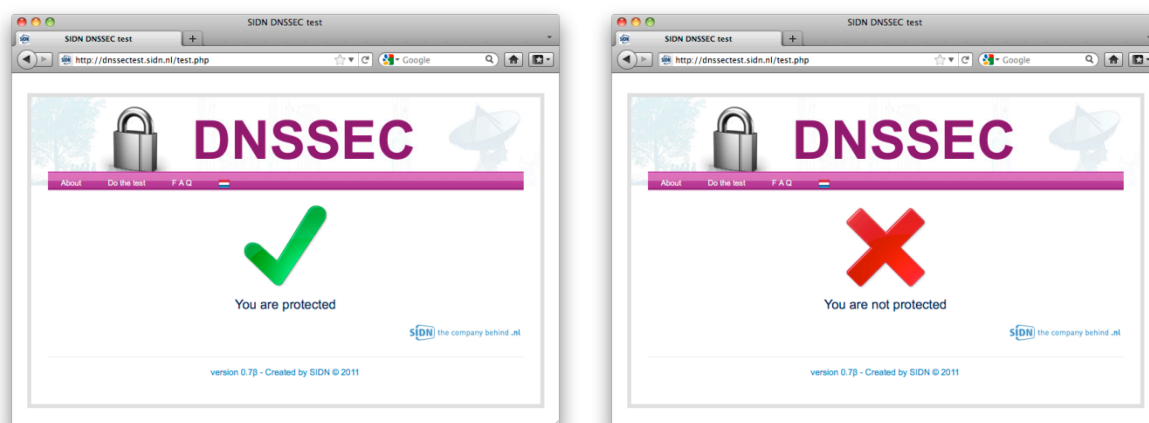


Figure 9 - Sample DNSSEC check output

Note the green checkmark in the left-hand screenshot indicating that DNSSEC validation works properly and the red cross in the right-hand screenshots indicating a problem (or no DNSSEC validation).