University of Alberta – Mint Program

DNSSEC Implementation in the RDC.AB.CA Domain

Mint 709

Patrick Hewitt
12-15-2014

Table of

Contents

Project Objective

The objective of this project is to implement Domain Name System Security Extensions (DNSSEC) in the RDC.AB.CA domain for which Red Deer College (RDC) maintains two authoritative Domain Name System (DNS) servers, NS and NS2. Further to this, the project will implement a simulation infrastructure to which the production architecture will be compared and the means and practice of the technology verified.

Overview of Implementation

The genesis of this project grew from the need to replace the two existing physical, end-of-life BlueCat Adonis appliances that had been serving up the RDC.AB.CA domain since 2008 and the fact that the Canadian Internet Registration Authority (CIRA) began accepting DNSSEC enabled .CA domains earlier in 2014[1]. With this need and opportunity in mind, our infrastructure team looked at a few options and decided, based on our experience with Microsoft Windows DNS within our internal network, the technology and security offered in Server Core Windows Server 2012 R2[2], our highly virtualized environment, and the comparative cost and time to implement, that the Adonis appliances would be replaced with two pairs of Server Core Windows Server 2012 R2 computers running in our VMware virtualized infrastructure. One pair would act as caching/resolving DNS servers and the other pair as authoritative DNS servers. At the same time I was in communication with CIRA and our domain register, egate DOMAINS Inc., about what was required once we had signed our domain in regards to publishing our delegation signer (DS) record in the parent .CA domain and providing our DNS public key (DNSKEY) information. Initially we were going to have to change registers as egate was not a register recognized by CIRA as being able to accept DNSSEC registrations, however, with a little cajoling on the part of CIRA and a little work by egate, switching registers no longer became necessary. With that, the new servers were built and configured over a couple months, the authoritative DNS servers were deployed to production on 21 November, the caching/resolving DNS servers were deployed and the RDC.AB.CA zone signed on 3 December, and the DS record and DNSKEY information provided to egate on 4 December. At the same time as the production infrastructure was built and deployed, a simulation infrastructure comprising three Server Core servers, cloned from the same template as was used with the productions servers, was built in an isolated subnet. Two of the servers (DNS1 and DNS2) were built to act as the authoritative DNS servers and the third server (DNSRecurs1) was built to act as a caching/resolving DNS server in the simulation environment. As part of the implementation process, the DNSstuff[3] website was utilized as a tool to help ensure that we were in compliance with appropriate DNS RFC's and best practices. I have attached documents that show the results of three scans from the website offering views into different times during the implementation: old infrastructure prior to introduction of new servers[4], new servers prior to signing[5], and the servers after signing[6]. To date, the DS record has yet to be published to the .CA domain by our register.

Our DNS Architecture

---

[1] "CIRA Calls on Canadian Website Owners to Help Improve .ca Security." *ITBusinessca*. N.p., n.d. Web. 13 Dec. 2014. http://www.itbusiness.ca/article/cira-calls-on-canadian-website-owners-to-help-improve-ca-security.

[2] "Server Core for Windows Server 2012 R2 and Windows Server 2012." *Server Core for Windows Server 2012 R2 and Windows Server 2012 (Windows)*. N.p., n.d. Web. 13 Dec. 2014. http://msdn.microsoft.com/en-us/library/hh846323(v=vs.85).aspx.

[3] http://www.dnsstuff.com/

[4] See accompanying 1dnsreport_rdc ab ca (Before New Build).pdf document.

[5] See accompanying 2dnsreport_rdc ab ca (After setup as AuthDNS).pdf document.

[6] See accompanying 3dnsreport_rdc ab ca (after signing rdc.ab.ca).pdf document.

We run a split DNS architecture where our authoritative DNS infrastructure runs in a perimeter network behind our external firewall, but is isolated from our internal Active Directory integrated DNS infrastructure[7]. The authoritative DNS servers in the perimeter network have their IP addresses translated to external IP addresses by our firewall, these addresses can be found in our CIRA registration by other DNS resolvers looking for authoritative addresses of the RDC.AB.CA domain. The old Adonis appliances ran a locked down version of Linux with a BIND DNS implementation allowing for both internal and external facing views which also allowed them to act as DMZ DNS caching servers to which our internal DNS was forwarded. In the new infrastructure build, due to some differences between Linux/BIND DNS and Windows DNS, the DMZ/caching DNS and the authoritative DNS are separated into two pairs of primary and secondary file based DNS servers. Though not explicitly part of this project, a pair of caching/resolving DMZ DNS servers were deployed to production so that the old Adonis appliances could be retired.

The Build

All servers for this project, four for the production environment and three for the simulation environment, were cloned from a Server Core Windows Server 2012 R2 template I built for the purposes of this project. Initially I was going to use Active Directory integrated DNS to take advantage of the replication and security properties available with Active Directory, however, over numerous builds and rebuilds using various configurations it became clear that I would not be able to hide the actual IP addresses of the servers and reveal only the translated IP addresses to those who would accessing our authoritative DNS from outside our network, the actual server IP addresses could not be completely obfuscated; as a result I used file based Windows DNS. All three pairs of DNS servers, production authoritative (NS, NS2), simulation authoritative (DNS1, DNS2), and production DMZ/caching (NSC1, NSC2), are file based primary/secondary configurations that do not accept dynamic updates, allow zone transfers only from primary to secondary, and with recursion disabled on the authoritative servers. In regards to the DNSSEC configuration both the key signing keys (KSK) and zone signing key (ZSK) use RSA/SHA-256 as the signing algorithm with the KSK having a key length or 2048 and ZSK having a key length of 1024. The rollover for the KSK is 755 days and for the ZSK it is 90 days and both are set for automatic rollover. NSEC3 is used rather than NSEC for signing the zone to limit opportunities for zone enumeration. In the simulation environment trust anchors were distributed to all DNS servers and in the production environment the DS resource record and DNSKEY information provided to our register. Firewalling was also an important component of the build, first in ensuring that the larger packet sizes required of DNSSEC were accommodated in addition to having TCP and UDP connectivity to the servers, and secondly, in ensuring that access to the new servers was locked down appropriately from internal and external access while also allowing the machines to be appropriately accessed by our infrastructure team.

Primer on DNSSEC

DNSSEC is an Internet Engineering Task Force (IETF) standard, first proposed in RFC2065 and meant to address security concerns in the original DNS specification. Specifically it utilizes digitally signed resource records to provide authentication and validation to DNS resolvers or applications that utilize DNS as a means to secure the DNS system. RFC2065 has been obsoleted by subsequent RFC's, including RFC2535 , RFC4033 , RFC4034 , RFC4035 which in turn are updated by RFC6014 and RFC6840, amongst others.

The authentication and validation of DNS is achieved through a process called zone signing whereby a series of new digitally signed resource records are created in each signed zone that allows for the creation of a chain of trust from the root DNS servers down through the various top level domains (TLD), to a specific authoritative DNS server. This secure process doesn't change the basic, and well founded, mechanism of the DNS query response process that has existed

[7] See accompanying RDC-DNS-Infrastructure.pdf document for diagram of our DNS infrastructure.

since the creation of DNS, nor does it protect or encrypt the DNS record data itself, but it does stop some exploits, such as DNS spoofing, from occurring and will validate that the data requested is from the domain owner for that data.

The new resource records added to the general DNS specification are the following: DNSKEY, DS, NSEC, NSEC3, NSEC3PARAM, and RRSIG. The DNSKEY, DNS public key, resource record is used by resolvers to authenticate the resource records for a zone that have been signed with a private key for that zone[8]. The DS, delegation signer, resource record is used in creating authentication chains between parent and child DNS zones, so for example, the DS resource record for RDC.AB.CA is added in the .CA zone and points to the apex DNSKEY resource record within the RDC.AB.CA zone[9]. Both the DS and DNSKEY resource records act as trust anchors and must be distributed to nonauthoritative DNS servers to allow for the validation of DNS responses. The NSEC, next secure, resource record lists the next resource record name and resource record types present and when taken as a group for the zone provides authoritative denial of existence for resource records within a zone[10]. The NSEC3 resource record is an alternative to the NSEC resource record. While it also provides the next resource record name and type and authoritative denial of existence, it also addresses concerns about the ability to walk a zone for all of its records[11] by the use of hashed record names. The NSEC3PARAM resource record provides authoritative servers with information on which resource records to provide when queried for non-existent names[12]. The RRSIG, resource record signature, resource record holds the stored digital signature for the various resource record types within a signed zone, so a SOA or CNAME or TXT resource record would each also have a RRSIG resource record containing the digital signature for the record that would then be used by a resolver to validate that record[13].

In addition to these new resource records, the DNS specification itself[14] needed to be modified to allow for an increased message size, limited to 512 bytes under the original DNS specification, and thus accommodate the signatures found in DNSSEC, or additionally, multiple IPv6 addresses or long TXT messages, while still maintaining backward compatibility with the original DNS specification and without resorting to the Transmission Control Protocol (TCP) and the increased overhead this would entail. These Extension Mechanisms for DNS (EDNS) utilize a hop-by-hop process whereby each party to a DNS resolution negotiate the additional flags and return codes, allowed for in the new specification[15], to complete the DNS resolution. With DNSSEC, not only is the larger message size required to allow for the new resource record signatures, but the additional flagging space within the specification is used to indicate the requirement for DNSSEC validation data. Four flags that are utilized by DNSSEC[16] are: the DO bit, used in a DNS query to indicate DNSSEC OK, or that the client is DNSSEC aware and DNSSEC data can be included in a DNS response (DO=1 send DNSSEC data, DO=0 do not send DNSSEC data); the AD bit, authenticated data, is used in a DNS response to indicate that the data is valid and has been authenticated using DNSSEC (AD=1 validated, AD=0 not validated); the CD bit, checking disabled, is used in a DNS query to indicate that a response should be provided regardless of whether validation has occurred or not (CD=0 response should be sent if validation is successful or not, CD=1 response should not be sent if validation is required and it failed); the AA bit, authoritative answer, is part of the DNS specification, but used by DNSSEC

---

[8] "RFC 4034 - Resource Records for the DNS Security Extensions." *RFC 4034 - Resource Records for the DNS Security Extensions*. N.p., n.d. Web. 13 Dec. 2014. https://tools.ietf.org/html/rfc4034#section-2.

[9] "RFC 4035 - Protocol Modifications for the DNS Security Extensions." *RFC 4035 - Protocol Modifications for the DNS Security Extensions*. N.p., n.d. Web. 13 Dec. 2014. https://tools.ietf.org/html/rfc4035#section-2.4.

[10] "RFC 4034 - Resource Records for the DNS Security Extensions." *RFC 4034 - Resource Records for the DNS Security Extensions*. N.p., n.d. Web. 13 Dec. 2014. https://tools.ietf.org/html/rfc4034#section-4.

[11] "RFC 5155 - DNS Security (DNSSEC) Hashed Authenticated Denial of Existence." *RFC 5155 - DNS Security (DNSSEC) Hashed Authenticated Denial of Existence*. N.p., n.d. Web. 13 Dec. 2014. http://tools.ietf.org/html/rfc5155#section-3

[12] "RFC 5155 - DNS Security (DNSSEC) Hashed Authenticated Denial of Existence." *RFC 5155 - DNS Security (DNSSEC) Hashed Authenticated Denial of Existence*. N.p., n.d. Web. 13 Dec. 2014. http://tools.ietf.org/html/rfc5155#section-4

[13] "RFC 4034 - Resource Records for the DNS Security Extensions." *RFC 4034 - Resource Records for the DNS Security Extensions*. N.p., n.d. Web. 13 Dec. 2014. https://tools.ietf.org/html/rfc4034#section-3.

[14] "RFC 1035 - Domain Names - Implementation and Specification." *RFC 1035 - Domain Names - Implementation and Specification*. N.p., n.d. Web. 13 Dec. 2014. http://tools.ietf.org/html/rfc1035.

[15] "RFC 6891 - Extension Mechanisms for DNS (EDNS(0))." *RFC 6891 - Extension Mechanisms for DNS (EDNS(0))*. N.p., n.d. Web. 13 Dec. 2014. http://tools.ietf.org/html/rfc6891.

[16] "Overview of DNSSEC." Overview of DNSSEC. N.p., n.d. Web. 13 Dec. 2014. http://technet.microsoft.com/library/jj200221.aspx.

to indicate if a response came directly from an authoritative DNS server because an authoritative response does not need to be validated, this would be redundant, so in a scenario where authentication is required by a client, AD=1, the client would also accept the AA=1 bit, even if the AD=0 bit is set (AA=1 directly from authoritative server, AA=0 not directly from authoritative server)[17]. We will now turn to an analysis of the production and simulation environments created for the signed RDC.AB.CA domain to better understand and verify the means and practice of DNSSEC.

The Analysis

The analysis will use a simulation environment comprised of a client computer and three file based DNS servers. DNS1 and DNS2 are meant to simulate our authoritative DNS servers NS and NS2, DNSRecurs1 is meant to simulate a recursive DNS resolver somewhere out on the Internet, and DNSSim is a client computer using the resolving DNS of DNSRecurs1 to resolve names that can be found on DNS1 and DNS2[18]. In analyzing the production environment I placed a client computer outside our network and assigned it one of our external IP addresses and used Google Open DNS (8.8.8.8 and 8.8.4.4) for the client DNS. In both simulation and production there is a single RDC.AB.CA signed zone. The client computers used in both simulation and production will be running Windows 8.1 as their operating system, PowerShell 4.0 and the Resolve-DnsName commandlet[19] will be used for running DNS queries, and Wireshark 1.12.2 will be used for packet capture. The stub resolver utilized by the Windows 8.1 operating system is DNSSEC aware but does not require validation by default, in other words, the CD bit is set to zero, CD=0, though this can be changed based on settings within the Name Resolution Policy Table (NRPT) set on the client or the policy within the domain it operates in.

We will start by querying the signed RDC.AB.CA zone from a client computer without requesting DNSSEC data be set, essentially a basic DNS query:

Simulation Network:

```
PS C:\> resolve-dnsname -name dns1.rdc.ab.ca -server dnsrecurs1 -type A

Name                                           Type   TTL   Section    IPAddress
----                                           ----   ---   -------    ---------
dns1.rdc.ab.ca                                 A      78210 Answer     192.168.5.10
```

The above is a screen shot that shows a query initiated on the simulation network client DNSSim for an A record using DNSRecurs1 as a resolver. The accompanying response shows the appropriate A record has been returned, dns1.rdc.ab.ca, with the correct associated IP address. The screen shot below shows the relevant part of the DNS response packet capture of this request (see accompanying Wireshark capture file sim-signed-no-validation.pcapng):

[17] "Overview of DNSSEC." Overview of DNSSEC. N.p., n.d. Web. 13 Dec. 2014. http://technet.microsoft.com/library/jj200221.aspx.
[18] See accompanying RDC-DNS-Infrastructure.pdf document for diagram of production and simulation environments.
[19] "Resolve-DnsName." *Resolve-DnsName*. N.p., n.d. Web. 13 Dec. 2014. http://technet.microsoft.com/en-us/library/jj590781.aspx.

```
  10 5.65053300 192.168.5.30   192.168.5.31   DNS      90 Standard query response 0xa476  A 192.168.5.10
User Datagram Protocol, Src Port: 53 (53), Dst Port: 57614 (57614)
Domain Name System (response)
  [Request In: 9]
  [Time: 0.000174000 seconds]
  Transaction ID: 0xa476
⊟ Flags: 0x8180 Standard query response, No error
    1... .... .... .... = Response: Message is a response
    .000 0... .... .... = Opcode: Standard query (0)
    .... .0.. .... .... = Authoritative: Server is not an authority for domain
    .... ..0. .... .... = Truncated: Message is not truncated
    .... ...1 .... .... = Recursion desired: Do query recursively
    .... .... 1... .... = Recursion available: Server can do recursive queries
    .... .... .0.. .... = Z: reserved (0)
    .... .... ..0. .... = Answer authenticated: Answer/authority portion was not authenticated by the server
    .... .... ...0 .... = Non-authenticated data: Unacceptable
    .... .... .... 0000 = Reply code: No error (0)
  Questions: 1
  Answer RRs: 1
  Authority RRs: 0
  Additional RRs: 0
⊟ Queries
  ⊟ dns1.rdc.ab.ca: type A, class IN
      Name: dns1.rdc.ab.ca
      [Name Length: 14]
      [Label Count: 4]
      Type: A (Host Address) (1)
      Class: IN (0x0001)
⊟ Answers
  ⊟ dns1.rdc.ab.ca: type A, class IN, addr 192.168.5.10
      Name: dns1.rdc.ab.ca
      Type: A (Host Address) (1)
      Class: IN (0x0001)
      Time to live: 78210
      Data length: 4
      Address: 192.168.5.10 (192.168.5.10)
```

This screen shot shows the response packet from DNSRecurs1 to DNSSim. You can note under the “Flags” section that the packet shows that DNSRecurs1 is not an authority for the RDC.AB.CA domain and that validation is not a requirement of this request. Additionally, the packet shows both the query and response of the interaction between DNSSim and DNSRecurs1 in the “Queries” and “Answers” sections. We will now look at a similar query in the production environment.

Production Network:

```
PS C:\> resolve-dnsname -name ns.rdc.ab.ca -type A -server ns.rdc.ab.ca

Name                                           Type   TTL   Section    IPAddress
----                                           ----   ---   -------    ---------
ns.rdc.ab.ca                                   A      3600  Answer     204.209.17.126
```

As with the simulation environment I did an A record lookup as indicated in the screen shot above, however, in this circumstance rather than querying an intermediary DNS resolver as I did in simulation, I queried the authoritative DNS server for RDC.AB.CA, NS, directly. I did this largely to ensure that I could identify the correct packet in the packet capture, but it is also instructive in seeing the differences in terms of how this affects EDNS flag bits. Below shows the relevant part of the packet capture of this request (see accompanying Wireshark capture file prod-signed-no-validation.pcapng):

```
  6 3.98848400 192.168.2.180      204.209.17.211      DNS       88 S
  8 5.42041400 192.221.162.195    192.168.2.180       DNS       91 S
  9 5.42058700 192.168.2.180      192.221.162.195     DNS      296 S

   .... .1.. .... .... = Authoritative: Server is an authority for
   .... ..0. .... .... = Truncated: Message is not truncated
   .... ...1 .... .... = Recursion desired: Do query recursively
   .... .... 0... .... = Recursion available: Server can't do recur
   .... .... .0.. .... = Z: reserved (0)
   .... .... ..0. .... = Answer authenticated: Answer/authority por
   .... .... ...0 .... = Non-authenticated data: Unacceptable
   .... .... .... 0000 = Reply code: No error (0)
Questions: 1
Answer RRs: 1
Authority RRs: 0
Additional RRs: 0
Queries
⊟ ns.rdc.ab.ca: type A, class IN
    Name: ns.rdc.ab.ca
    [Name Length: 12]
    [Label Count: 4]
    Type: A (Host Address) (1)
    Class: IN (0x0001)
Answers
⊟ ns.rdc.ab.ca: type A, class IN, addr 204.209.17.126
    Name: ns.rdc.ab.ca
    Type: A (Host Address) (1)
    Class: IN (0x0001)
    Time to live: 3600
    Data length: 4
    Address: 204.209.17.126 (204.209.17.126)
```

This screen shot shows the response packet from NS to the production client computer sitting on the outside of our network. In this circumstance, under the "Flags" section, NS is correctly identifying itself as an authority for the RDC.AB.CA domain, but that like simulation, validation is not a requirement of this request. Also like simulation, the packet shows both the query and response of the interaction between client and host in the "Queries" and "Answers" sections.

Now we will query the signed RDC.AB.CA zone and request that DNSSEC data be sent:

Simulation Network:

```
PS C:\> resolve-dnsname -name dns1.rdc.ab.ca -server dnsrecurs1 -type A -dnssecok

Name                                           Type   TTL   Section    IPAddress
----                                           ----   ---   -------    ---------
dns1.rdc.ab.ca                                 A      86081 Answer     192.168.5.10

Name          : dns1.rdc.ab.ca
QueryType     : RRSIG
TTL           : 86081
Section       : Answer
TypeCovered   : A
Algorithm     : 8
LabelCount    : 4
OriginalTtl   : 604800
Expiration    : 12/19/2014 9:36:31 AM
Signed        : 12/9/2014 8:36:31 AM
Signer        : rdc.ab.ca
Signature     : {90, 137, 243, 191...}


Name       : .
QueryType  : OPT
TTL        : 32768
Section    : Additional
Data       : {}
```

The above image shows the same query in the simulation network as the previous scenario with the exception of the requirement that now we are asking for DNSSEC data to be sent. This is achieved by utilizing the "-dnssecok" switch

which is appended to the end of the query. This switch has the effect of setting the DO bit to one, DO=1. The accompanying response provides the normal DNS query response information, but also provides the additional RRSIG data associated with the A resource record we have requested. So for example, you can now see who the signer for the record is, when the record was signed, the signature, etc. Below is a screen shot of the relevant part of the packet capture of this request (see accompanying Wireshark capture file sim-signed-no-validation-dnssecok.pcapng):

```
  11 12.7604880 192.168.5.31   192.168.5.30   DNS       85 Standard query 0x231a  A d
  12 12.7606570 192.168.5.30   192.168.5.31   DNS      398 Standard query response 0x
    .... ...1 .... .... = Recursion desired: Do query recursively
    .... .... 1... .... = Recursion available: Server can do recursive queries
    .... .... .0.. .... = Z: reserved (0)
    .... .... ..0. .... = Answer authenticated: Answer/authority portion was not aut
    .... .... ...0 .... = Non-authenticated data: Unacceptable
    .... .... .... 0000 = Reply code: No error (0)
  Questions: 1
  Answer RRs: 2
  Authority RRs: 0
  Additional RRs: 1
⊟ Queries
  ⊞ dns1.rdc.ab.ca: type A, class IN
⊟ Answers
  ⊟ dns1.rdc.ab.ca: type A, class IN, addr 192.168.5.10
      Name: dns1.rdc.ab.ca
      Type: A (Host Address) (1)
      Class: IN (0x0001)
      Time to live: 86081
      Data length: 4
      Address: 192.168.5.10 (192.168.5.10)
  ⊟ dns1.rdc.ab.ca: type RRSIG, class IN
      Name: dns1.rdc.ab.ca
      Type: RRSIG (46)
      Class: IN (0x0001)
      Time to live: 86081
      Data length: 285
      Type Covered: A (Host Address) (1)
      Algorithm: RSA/SHA-256 (8)
      Labels: 4
      Original TTL: 604800 (7 days)
      Signature Expiration: Dec 19, 2014 02:36:31.000000000 Mountain Standard Time
      Signature Inception: Dec  9, 2014 01:36:31.000000000 Mountain Standard Time
      Key Tag: 25569
      Signer's name: rdc.ab.ca
      Signature: 5a89f3bf0a912ee79cb1bd67e7e59b25e893b37e065f635b...
⊟ Additional records
  ⊟ <Root>: type OPT
      Name: <Root>
      Type: OPT (41)
      UDP payload size: 4000
      Higher bits in extended RCODE: 0x00
      EDNS0 version: 0
    ⊟ Z: 0x8000
        1... .... .... .... = DO bit: Accepts DNSSEC security RRs
        .000 0000 0000 0000 = Reserved: 0x0000
      Data length: 0
```

Here you can see that once again the information under the flags section shows that the response isn't authenticated, but within the "Additional records" section, the EDNS portion of the message, that the value of the DO bit has been set to one. We will now look at this scenario in the production environment.

Production Network:

The screen shot below from the client computer in production showing the PowerShell commandlet query response shows similar results as in simulation when the "-dnssecok" switch is applied, with the correct answer and the additional DNSSEC information provided with the response. Following that is a screen shot of the relevant part of the packet capture of response to this query (see accompanying Wireshark capture file prod-signed-no-validation-dnssecok.pcapng), as in simulation the DO bit shows as being set to one and like the previous example when directly querying an authoritative server the flag bits are showing AA=1.

```
PS C:\> resolve-dnsname -name ns.rdc.ab.ca -type A -server ns.rdc.ab.ca -dnssecok

Name                                           Type   TTL   Section    IPAddress
----                                           ----   ---   -------    ---------
ns.rdc.ab.ca                                   A      3600  Answer     204.209.17.126

Name         : ns.rdc.ab.ca
QueryType    : RRSIG
TTL          : 3600
Section      : Answer
TypeCovered  : A
Algorithm    : 8
LabelCount   : 4
OriginalTtl  : 3600
Expiration   : 12/22/2014 2:39:31 PM
Signed       : 12/12/2014 1:39:31 PM
Signer       : rdc.ab.ca
Signature    : {158, 24, 119, 47...}


Name       : .
QueryType  : OPT
TTL        : 32768
Section    : Additional
Data       : {}
```

```
11 5.51108000 192.168.2.180    204.209.17.211    DNS    268 Standard query response 0xd4e5  A 204.209.17
    .... .1.. .... .... = Authoritative: Server is an authority for domain
    .... ..0. .... .... = Truncated: Message is not truncated
    .... ...1 .... .... = Recursion desired: Do query recursively
    .... .... 0... .... = Recursion available: Server can't do recursive queries
    .... .... .0.. .... = Z: reserved (0)
    .... .... ..0. .... = Answer authenticated: Answer/authority portion was not authenticated by the server
    .... .... ...0 .... = Non-authenticated data: Unacceptable
    .... .... .... 0000 = Reply code: No error (0)
  Questions: 1
  Answer RRs: 2
  Authority RRs: 0
  Additional RRs: 1
  Queries
  Answers
⊟ ns.rdc.ab.ca: type A, class IN, addr 204.209.17.126
      Name: ns.rdc.ab.ca
      Type: A (Host Address) (1)
      Class: IN (0x0001)
      Time to live: 3600
      Data length: 4
      Address: 204.209.17.126 (204.209.17.126)
⊟ ns.rdc.ab.ca: type RRSIG, class IN
      Name: ns.rdc.ab.ca
      Type: RRSIG (46)
      Class: IN (0x0001)
      Time to live: 3600
      Data length: 157
      Type Covered: A (Host Address) (1)
      Algorithm: RSA/SHA-256 (8)
      Labels: 4
      Original TTL: 3600 (1 hour)
      Signature Expiration: Dec 22, 2014 07:39:31.000000000 Mountain Standard Time
      Signature Inception: Dec 12, 2014 06:39:31.000000000 Mountain Standard Time
      Key Tag: 42655
      Signer's name: rdc.ab.ca
      Signature: 9e18772f8d5c89a2b841512d2fd595f9dff474ac265c1d18...
  Additional records
⊟ <Root>: type OPT
      Name: <Root>
      Type: OPT (41)
      UDP payload size: 4000
      Higher bits in extended RCODE: 0x00
      EDNS0 version: 0
    ⊟ Z: 0x8000
        1... .... .... .... = DO bit: Accepts DNSSEC security RRs
        .000 0000 0000 0000 = Reserved: 0x0000
```

Finally we will query a signed zone and require validation:

To begin we will look to ensure that trust anchors are resolvable, valid, and active for this I will utilize the resolve-dnsname, get-dnsservertrustanchor[20], and get-dnsservertrustpoint[21] commandlets:

Simulation Network:
In the simulation network, the trust anchor for RDC.AB.CA was created initially on the primary authoritative DNS server, DNS1, when the zone was signed, then manually distributed to DNS2 shortly thereafter. The trust anchor was then subsequently distributed to DNSRecurs1 for the purposes of testing this scenario. Firstly, from the simulation computer client I show that the trust anchors are resolvable for each DNS server:

```
PS C:\> resolve-dnsname -name rdc.ab.ca.trustanchors -type dnskey -server dns1

Name                                Type   TTL   Section    Flags  Protocol Algorithm      Key
----                                ----   ---   -------    -----  -------- ---------      ---
rdc.ab.ca.trustanchors              DNSKEY 3600  Answer     257    DNSSEC   8              {3, 1, 0, 1...}
rdc.ab.ca.trustanchors              DNSKEY 3600  Answer     257    DNSSEC   8              {3, 1, 0, 1...}


PS C:\> resolve-dnsname -name rdc.ab.ca.trustanchors -type dnskey -server dns2

Name                                Type   TTL   Section    Flags  Protocol Algorithm      Key
----                                ----   ---   -------    -----  -------- ---------      ---
rdc.ab.ca.trustanchors              DNSKEY 3600  Answer     257    DNSSEC   8              {3, 1, 0, 1...}
rdc.ab.ca.trustanchors              DNSKEY 3600  Answer     257    DNSSEC   8              {3, 1, 0, 1...}


PS C:\> resolve-dnsname -name rdc.ab.ca.trustanchors -type dnskey -server dnsrecurs1

Name                                Type   TTL   Section    Flags  Protocol Algorithm      Key
----                                ----   ---   -------    -----  -------- ---------      ---
rdc.ab.ca.trustanchors              DNSKEY 3600  Answer     257    DNSSEC   8              {3, 1, 0, 1...}
rdc.ab.ca.trustanchors              DNSKEY 3600  Answer     257    DNSSEC   8              {3, 1, 0, 1...}
```

Now from each server I show that the trust anchors and points are active and valid:

```
[DNS2]: PS C:\> get-dnsservertrustanchor -name rdc.ab.ca

TrustAnchorName               TrustAnchorType      TrustAnchorState     TrustAnchorData
---------------               ---------------      ----------------     ---------------
rdc.ab.ca.                    DNSKEY               Valid                [64135][DnsSec][RsaSha256][AwEAAdDR9+2v28+A...
rdc.ab.ca.                    DNSKEY               Valid                [32156][DnsSec][RsaSha256][AwEAAbPZhJ/YM4lu...


[DNS2]: PS C:\> get-dnsservertrustpoint -name rdc.ab.ca

TrustPointName                          TrustPointState      LastActiveRefreshTime     NextActiveRefreshTime
--------------                          ---------------      ---------------------     ---------------------
rdc.ab.ca.                              Active               12/3/2014 12:45:29 PM     12/14/2014 3:58:43 PM
```

```
[DNS1]: PS C:\> get-dnsservertrustanchor -name rdc.ab.ca

TrustAnchorName               TrustAnchorType      TrustAnchorState     TrustAnchorData
---------------               ---------------      ----------------     ---------------
rdc.ab.ca.                    DNSKEY               Valid                [64135][DnsSec][RsaSha256][AwEAAdDR9+2v28+A...
rdc.ab.ca.                    DNSKEY               Valid                [32156][DnsSec][RsaSha256][AwEAAbPZhJ/YM4lu...


[DNS1]: PS C:\> get-dnsservertrustpoint -name rdc.ab.ca

TrustPointName                          TrustPointState      LastActiveRefreshTime     NextActiveRefreshTime
--------------                          ---------------      ---------------------     ---------------------
rdc.ab.ca.                              Active                                         12/14/2014 3:59:10 PM
```

[20] "Get-DnsServerTrustAnchor." *Get-DnsServerTrustAnchor*. N.p., n.d. Web. 14 Dec. 2014. http://technet.microsoft.com/en-us/library/jj649856.aspx#feedback.
[21] "Get-DnsServerTrustPoint." *Get-DnsServerTrustPoint*. N.p., n.d. Web. 14 Dec. 2014. http://technet.microsoft.com/en-us/library/jj649862.aspx

```
[DNSRecurs1]: PS C:\> get-dnsservertrustanchor -name rdc.ab.ca

TrustAnchorName               TrustAnchorType      TrustAnchorState     TrustAnchorData
---------------               ---------------      ----------------     ---------------
rdc.ab.ca.                    DNSKEY               Valid                [64135][DnsSec][RsaSha256][AwEAAdDR9+2v28+A...
rdc.ab.ca.                    DNSKEY               Valid                [32156][DnsSec][RsaSha256][AwEAAbPZhJ/YM4lu...


[DNSRecurs1]: PS C:\> get-dnsservertrustpoint -name rdc.ab.ca

TrustPointName                          TrustPointState      LastActiveRefreshTime     NextActiveRefreshTime
--------------                          ---------------      ---------------------     ---------------------
rdc.ab.ca.                              Active               12/14/2014 3:57:17 PM     12/14/2014 4:57:17 PM
```

Finally, I will set the NRPT policy for the work group based client computer used in the simulation environment using group policy and confirm this policy is enabled by using the get-dnsclientnrptpolicy commandlet[22]. This will have the effect of requiring the computer to require validation of DNS responses.

```
PS C:\> get-dnsclientnrptpolicy


Namespace                       : .rdc.ab.ca
QueryPolicy                     :
SecureNameQueryFallback         :
DirectAccessIPsecCARestriction  :
DirectAccessProxyName           :
DirectAccessDnsServers          :
DirectAccessEnabled             :
DirectAccessProxyType           : NoProxy
DirectAccessQueryIPsecEncryption :
DirectAccessQueryIPsecRequired  : False
NameServers                     :
DnsSecIPsecCARestriction        :
DnsSecQueryIPsecEncryption      :
DnsSecQueryIPsecRequired        : False
DnsSecValidationRequired        : True
NameEncoding                    : Utf8WithoutMapping
```

Given this requirement to get validation, the following is the query response on DNSSim sent to DNSRecurs1 for a record from the RDC.AB.CA domain followed by a screen shot of the packet capture of the event (see accompanying Wireshark capture file sim-signed-validation-required-dnssecok.pcapng):

```
PS C:\> resolve-dnsname -name dns2.rdc.ab.ca -type A -server dnsrecurs1

Name                                           Type   TTL   Section    IPAddress
----                                           ----   ---   -------    ---------
dns2.rdc.ab.ca                                 A      3600  Answer     192.168.5.20

Name         : dns2.rdc.ab.ca
QueryType    : RRSIG
TTL          : 3600
Section      : Answer
TypeCovered  : A
Algorithm    : 8
LabelCount   : 4
OriginalTtl  : 3600
Expiration   : 12/19/2014 9:36:31 AM
Signed       : 12/9/2014 8:36:31 AM
Signer       : rdc.ab.ca
Signature    : {70, 153, 233, 227...}


Name       : .
QueryType  : OPT
TTL        : 32768
Section    : Additional
Data       : {}
```

[22] "Get-DnsClientNrptPolicy." *Get-DnsClientNrptPolicy*. N.p., n.d. Web. 14 Dec. 2014. http://technet.microsoft.com/en-us/library/jj590779.aspx.

You can see from the above query/response that DNSSEC data was returned despite the fact that the "-dnssecok" switch wasn't applied. When the NRPT requirement is set to true the DO bit is set to one automatically, DO=1, and because the trust anchor had been distributed to DNSRecurs1 there was a valid chain of trust between DNSSim and the authoritative DNS servers thus the appropriate data was sent.

```
    5 4.80396400 192.168.5.30   192.168.5.31    DNS      398 Standard query response 0x6f0b
  Questions: 1
  Answer RRs: 2
  Authority RRs: 0
  Additional RRs: 1
- Queries
  + dns2.rdc.ab.ca: type A, class IN
- Answers
  - dns2.rdc.ab.ca: type A, class IN, addr 192.168.5.20
      Name: dns2.rdc.ab.ca
      Type: A (Host Address) (1)
      Class: IN (0x0001)
      Time to live: 1345
      Data length: 4
      Address: 192.168.5.20 (192.168.5.20)
  - dns2.rdc.ab.ca: type RRSIG, class IN
      Name: dns2.rdc.ab.ca
      Type: RRSIG (46)
      Class: IN (0x0001)
      Time to live: 1345
      Data length: 285
      Type Covered: A (Host Address) (1)
      Algorithm: RSA/SHA-256 (8)
      Labels: 4
      Original TTL: 3600 (1 hour)
      Signature Expiration: Dec 19, 2014 02:36:31.000000000 Mountain Standard Time
      Signature Inception: Dec  9, 2014 01:36:31.000000000 Mountain Standard Time
      Key Tag: 25569
      Signer's name: rdc.ab.ca
      Signature: 4699e9e3b55308bfd1f52c7f52add233b2b5a3c4599ad98d...
- Additional records
  - <Root>: type OPT
      Name: <Root>
      Type: OPT (41)
      UDP payload size: 4000
      Higher bits in extended RCODE: 0x00
      EDNS0 version: 0
    - Z: 0x8000
        1... .... .... .... = DO bit: Accepts DNSSEC security RRs
        .000 0000 0000 0000 = Reserved: 0x0000
      Data length: 0
```

The packet capture output is substantially the same as in the previous scenario, but the fact that response data is returned indicates that the chain of trust exists between DNSSim and the authoritative DNS servers for RDC.AB.CA and that the signed resource records can be validated. If I remove the trust anchor on DNSRecurs1 I get the following response:

```
PS C:\> resolve-dnsname -name dns1.rdc.ab.ca -type A -server dnsrecurs1
resolve-dnsname : dns1.rdc.ab.ca : Unsecured DNS packet
At line:1 char:1
+ resolve-dnsname -name dns1.rdc.ab.ca -type A -server dnsrecurs1
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ResourceUnavailable: (dns1.rdc.ab.ca:String) [Resolve-DnsName], Win32Exception
    + FullyQualifiedErrorId : DNS_ERROR_UNSECURE_PACKET,Microsoft.DnsClient.Commands.ResolveDnsName
```

We will now turn to look at this scenario in the production network.

Production Network:

As in simulation I will first determine the resolvability of the RDC.AB.CA trust anchors and ensure that they are active and valid on the NS and NS2 DNS servers. The screen shot below shows that from the production client I can resolve the trust anchors on NS and NS2, however, this doesn't work when attempting to do this via Google DNS.

```
PS C:\> resolve-dnsname -name rdc.ab.ca.trustanchors -type dnskey -server ns.rdc.ab.ca

Name                                Type   TTL   Section    Flags  Protocol Algorithm      Key
----                                ----   ---   -------    -----  -------- ---------      ---
rdc.ab.ca.trustanchors              DNSKEY 3600  Answer     257    DNSSEC   8              {3, 1, 0, 1...}
rdc.ab.ca.trustanchors              DNSKEY 3600  Answer     257    DNSSEC   8              {3, 1, 0, 1...}


PS C:\> resolve-dnsname -name rdc.ab.ca.trustanchors -type dnskey -server ns2.rdc.ab.ca

Name                                Type   TTL   Section    Flags  Protocol Algorithm      Key
----                                ----   ---   -------    -----  -------- ---------      ---
rdc.ab.ca.trustanchors              DNSKEY 3600  Answer     257    DNSSEC   8              {3, 1, 0, 1...}
rdc.ab.ca.trustanchors              DNSKEY 3600  Answer     257    DNSSEC   8              {3, 1, 0, 1...}


PS C:\> resolve-dnsname -name rdc.ab.ca.trustanchors -type dnskey -server 8.8.8.8
resolve-dnsname : rdc.ab.ca.trustanchors : DNS name does not exist
At line:1 char:1
+ resolve-dnsname -name rdc.ab.ca.trustanchors -type dnskey -server 8.8.8.8
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ResourceUnavailable: (rdc.ab.ca.trustanchors:String) [Resolve-DnsName], Win32Exception
    + FullyQualifiedErrorId : DNS_ERROR_RCODE_NAME_ERROR,Microsoft.DnsClient.Commands.ResolveDnsName
```

Following are screen shots showing that the trust anchors and points for RDC.AB.CA are valid and active for NS and NS2.

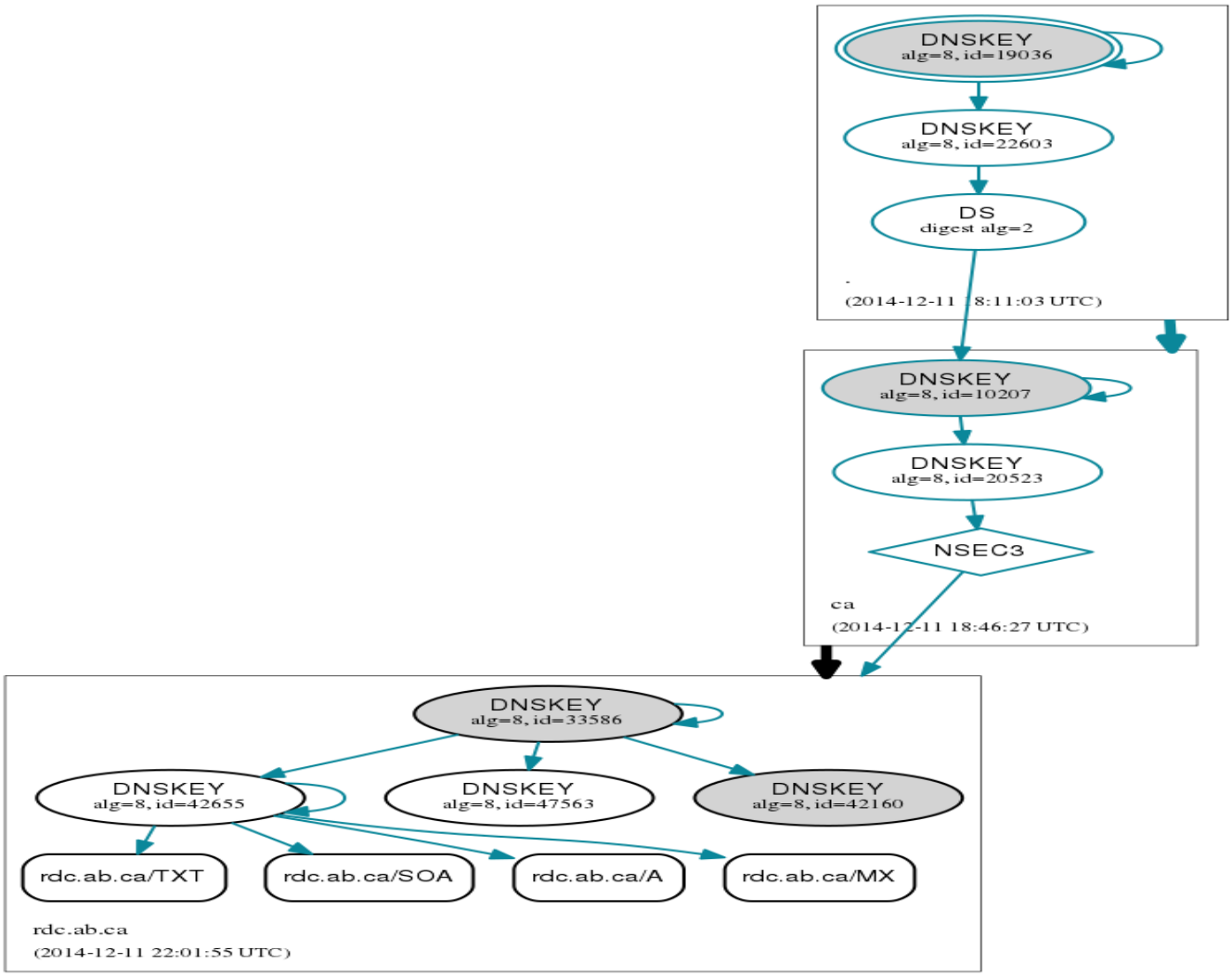
```
[ns]: PS C:\> get-dnsservertrustanchor -name rdc.ab.ca

TrustAnchorName               TrustAnchorType      TrustAnchorState     TrustAnchorData
---------------               ---------------      ----------------     ---------------
rdc.ab.ca.                    DNSKEY               Valid                [33586][DnsSec][RsaSha256][AwEAAbcl4jx7Oj+z...
rdc.ab.ca.                    DNSKEY               Valid                [42160][DnsSec][RsaSha256][AwEAAaM+kCQtm36j...


[ns]: PS C:\> get-dnsservertrustpoint -name rdc.ab.ca

TrustPointName                          TrustPointState      LastActiveRefreshTime      NextActiveRefreshTime
--------------                          ---------------      ---------------------      ---------------------
rdc.ab.ca.                              Active               12/4/2014 1:02:28 AM       12/15/2014 10:40:28 AM
```

```
[ns2]: PS C:\> get-dnsservertrustanchor -name rdc.ab.ca

TrustAnchorName               TrustAnchorType      TrustAnchorState     TrustAnchorData
---------------               ---------------      ----------------     ---------------
rdc.ab.ca.                    DNSKEY               Valid                [33586][DnsSec][RsaSha256][AwEAAbcl4jx7Oj+z...
rdc.ab.ca.                    DNSKEY               Valid                [42160][DnsSec][RsaSha256][AwEAAaM+kCQtm36j...


[ns2]: PS C:\> get-dnsservertrustpoint -name rdc.ab.ca

TrustPointName                          TrustPointState      LastActiveRefreshTime      NextActiveRefreshTime
--------------                          ---------------      ---------------------      ---------------------
rdc.ab.ca.                              Active               12/4/2014 1:10:12 AM       12/15/2014 10:42:18 AM
```

Once again to ensure that validation is required from my production client I have set its NRPT policy accordingly as seen in the first screen shot below. In the production scenario, I don't have an intermediary DNS resolver that I control, as in simulation, and as of 15 December the DS record I provided to our register has yet to be published to the .CA domain so necessarily there is no chain of trust to get to the authoritative DNS servers for RDC.AB.CA. This break in the chain of

trust can be visualized using an online DNS visualization tool provided by Sandia National Laboratories[23] and can be seen in the image in the second screen shot below:

```
PS C:\> get-dnsclientnrptpolicy

Namespace                       : .rdc.ab.ca
QueryPolicy                     :
SecureNameQueryFallback         :
DirectAccessIPsecCARestriction  :
DirectAccessProxyName           :
DirectAccessDnsServers          :
DirectAccessEnabled             :
DirectAccessProxyType           : NoProxy
DirectAccessQueryIPsecEncryption :
DirectAccessQueryIPsecRequired  : False
NameServers                     :
DnsSecIPsecCARestriction        :
DnsSecQueryIPsecEncryption      :
DnsSecQueryIPsecRequired        : False
DnsSecValidationRequired        : True
NameEncoding                    : Utf8WithoutMapping


PS C:\> (get-dnsclientnrptpolicy -namespace rdc.ab.ca).dnssecvalidationrequired
True
```



[23] http://dnsviz.net/

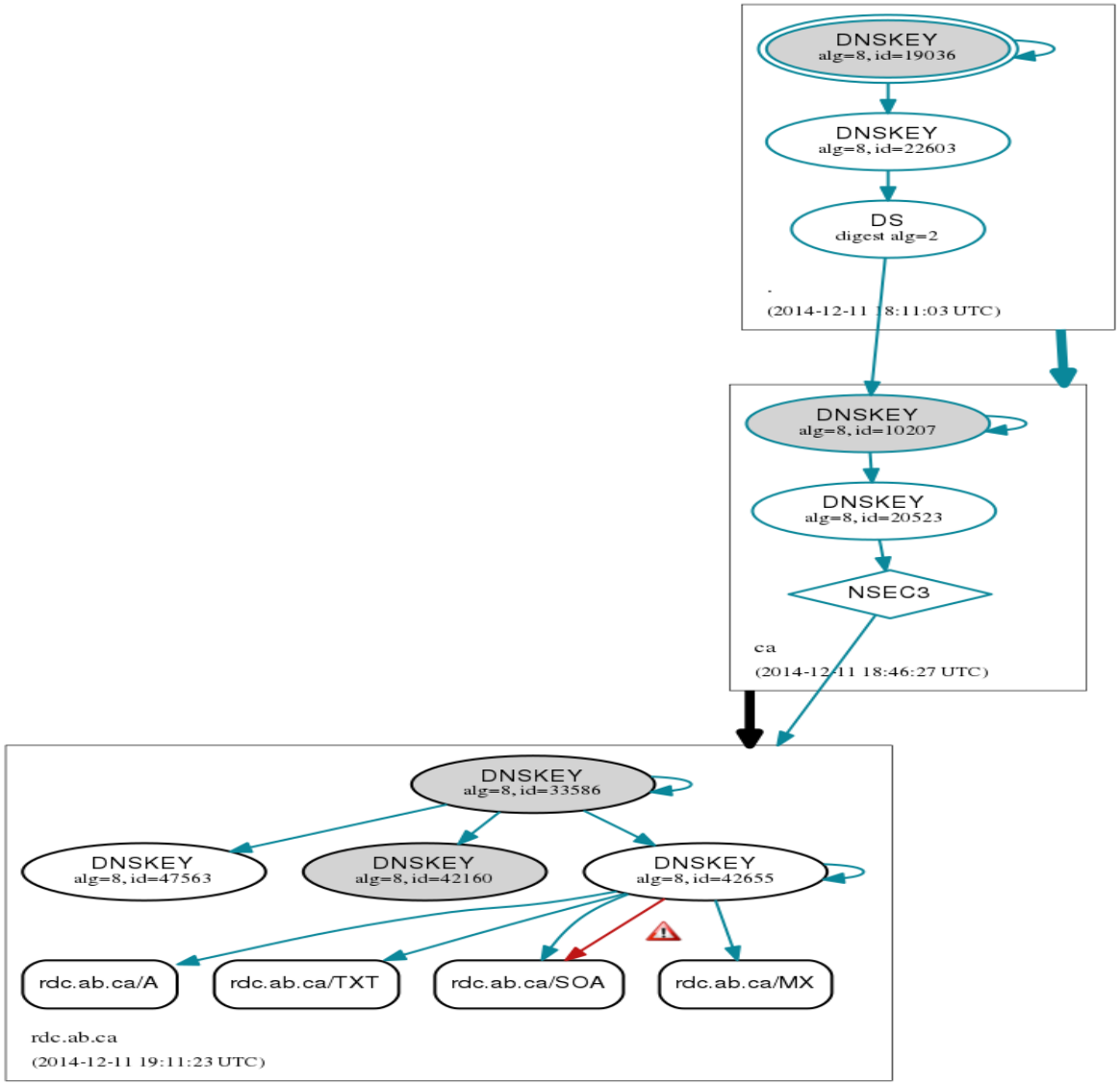
The black arrow going from the .CA domain to the RDC.AB.CA domain indicates an insecure link and a break in the chain of trust from the root servers down, additionally, our DNS public key has yet to added to the CIRA registry so resolvers have no means to validate our signed records.

So once I set the NRPT policy to require validation, I could not get the client to resolve a name in the RDC.AB.CA domain, even when querying the domain directly, as in, when validation is redundant because you are querying the authoritative zone directly, indicating that the requirement to authenticate the signed resource records overrides any other requirement; below are some screen shots showing the failure of lookups in the production domain. It is of note that I can resolve www.google.com indicating that the DNS client settings on the production client computer contact a resolver in possession of the DNS public key for google.com and that Google has signed DNS records that can be authenticated.

```
PS C:\> resolve-dnsname -name ns.rdc.ab.ca -type A -server ns.rdc.ab.ca
resolve-dnsname : No such host is known securely
At line:1 char:1
+ resolve-dnsname -name ns.rdc.ab.ca -type A -server ns.rdc.ab.ca
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (:) [Resolve-DnsName], SocketException
    + FullyQualifiedErrorId : System.Net.Sockets.SocketException,Microsoft.DnsClient.Commands.ResolveDnsName

PS C:\> resolve-dnsname -name ns2.rdc.ab.ca -type A -server ns.rdc.ab.ca
resolve-dnsname : No such host is known securely
At line:1 char:1
+ resolve-dnsname -name ns2.rdc.ab.ca -type A -server ns.rdc.ab.ca
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (:) [Resolve-DnsName], SocketException
    + FullyQualifiedErrorId : System.Net.Sockets.SocketException,Microsoft.DnsClient.Commands.ResolveDnsName

PS C:\> resolve-dnsname -name test.rdc.ab.ca -type A -server ns.rdc.ab.ca
resolve-dnsname : No such host is known securely
At line:1 char:1
+ resolve-dnsname -name test.rdc.ab.ca -type A -server ns.rdc.ab.ca
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (:) [Resolve-DnsName], SocketException
    + FullyQualifiedErrorId : System.Net.Sockets.SocketException,Microsoft.DnsClient.Commands.ResolveDnsName

PS C:\> resolve-dnsname -name www.google.com -type A -server 8.8.8.8

Name                                           Type   TTL   Section    IPAddress
----                                           ----   ---   -------    ---------
www.google.com                                 A      90    Answer     74.125.28.99
www.google.com                                 A      90    Answer     74.125.28.104
www.google.com                                 A      90    Answer     74.125.28.147
www.google.com                                 A      90    Answer     74.125.28.105
www.google.com                                 A      90    Answer     74.125.28.106
www.google.com                                 A      90    Answer     74.125.28.103
```

Outstanding Issues

The most obvious outstanding issue is the fact that as of 15 December our DS resource record and DNSKEY information has yet to be published to the .CA registry. Our register has yet to indicate an ETA for when this will occur and I am uncertain whether this is a learning curve issue with the register or if the process itself takes an extended period of time? Additionally, a bug may be occurring on our secondary server as expired RRSIGs have shown up in our secondary for the signed DNSSEC RDC.AB.CA zone in production. I discovered this using the DNS visualization tool mentioned earlier and can be seen in the screen shot below.



The visualization showed an invalid link to a Start of Authority (SOA) RRSIG, which upon subsequent investigation turned out to be an expired SOA RRSIG on the secondary. There is a known issue for this in Windows DNS[24] concerning the accumulation of expired RRSIGs on DNS secondary zones that should have been addressed in a June update rollup for Windows Server 2012 R2[25], however, this update has been applied to all new servers and the issue still occurred. The issue was corrected manually by forcing a transfer of a copy of the primary to the secondary zone, however, I will continue to monitor the issue to see if this was a one off or an ongoing concern.

[24] "DNS Queries Fail on Secondary DNS Server Running Windows Server 2012 R2 or Windows Server 2012." *DNS Queries Fail on Secondary DNS Server Running Windows Server 2012 R2 or Windows Server 2012*. N.p., n.d. Web. 13 Dec. 2014. http://support.microsoft.com/kb/2964090.

[25] "June 2014 Update Rollup for Windows RT 8.1, Windows 8.1, and Windows Server 2012 R2." *June 2014 Update Rollup for Windows RT 8.1, Windows 8.1, and Windows Server 2012 R2*. N.p., n.d. Web. 13 Dec. 2014. http://support.microsoft.com/kb/2962409

Conclusion and Moving Forward

This project was meant to implement DNSSEC in the RDC.AB.CA domain, which at this point is incomplete. The work on the authoritative DNS servers at RDC is complete, however, the publishing of the DNSSEC artefacts to the .CA domain by our register is incomplete so overall the implementation is incomplete. The additional objective of utilizing a simulation environment to compare with our production environment and verify the means and practice of the technology has been completed and aided in building and understanding the new production environment.

Moving forward, it is hoped the .CA registry will be updated and we can begin looking at a number of steps to further protect our infrastructure using DNSSEC technology. To begin with, later this month, the Active Directory integrated DNS in our internal network will be signed and once this is validated we can begin the process of modifying the NRPT policy settings through the use of Group Policy for our Windows client and server infrastructure to require validation, further protecting our internal infrastructure from DNS exploits. Additionally, once the DS record is finally published to .CA and we have validated and documented our DNSSEC policy and the process of key rollover we can look at signing our reverse lookup zones and publishing the DNSSEC artefacts from this process with the American Registry of Internet Numbers (ARIN) to allow for the validation of reverse lookups. In the longer term, we can look at the implementation of DNS-based Authentication of Named Entities (DANE), which is dependent on DNSSEC[26], to ensure that certificates we employ to encrypt traffic to our websites can be validated thus ensuring that important infrastructure accessed via the Internet, such as our web portal or virtual desktop environment, and the clients who use them are further protected.

[26] "DNS-based Authentication of Named Entities." *Wikipedia*. Wikimedia Foundation, n.d. Web. 13 Dec. 2014. http://en.wikipedia.org/wiki/DNS-based_Authentication_of_Named_Entities.

Bibliography

"CIRA Calls on Canadian Website Owners to Help Improve .ca Security." ITBusinessca. N.p., n.d. Web. 13 Dec. 2014.

"DNS-based Authentication of Named Entities." Wikipedia. Wikimedia Foundation, n.d. Web. 13 Dec. 2014.

"DNS Queries Fail on Secondary DNS Server Running Windows Server 2012 R2 or Windows Server 2012." DNS Queries Fail on Secondary DNS Server Running Windows Server 2012 R2 or Windows Server 2012. N.p., n.d. Web. 13 Dec. 2014.

"Get-DnsClientNrptPolicy." *Get-DnsClientNrptPolicy*. N.p., n.d. Web. 14 Dec. 2014.

"Get-DnsServerTrustAnchor." *Get-DnsServerTrustAnchor*. N.p., n.d. Web. 14 Dec. 2014.

"Get-DnsServerTrustPoint." *Get-DnsServerTrustPoint*. N.p., n.d. Web. 14 Dec. 2014.

"June 2014 Update Rollup for Windows RT 8.1, Windows 8.1, and Windows Server 2012 R2." June 2014 Update Rollup for Windows RT 8.1, Windows 8.1, and Windows Server 2012 R2. N.p., n.d. Web. 13 Dec. 2014.

"Overview of DNSSEC." Overview of DNSSEC. N.p., n.d. Web. 13 Dec. 2014.

"RFC 1035 - Domain Names - Implementation and Specification." RFC 1035 - Domain Names - Implementation and Specification. N.p., n.d. Web. 13 Dec. 2014.

"RFC 2065 - Domain Name System Security Extensions." RFC 2065 - Domain Name System Security Extensions. N.p., n.d. Web. 13 Dec. 2014.

"RFC 2535 - Domain Name System Security Extensions." RFC 2535 - Domain Name System Security Extensions. N.p., n.d. Web. 13 Dec. 2014.

"RFC 4033 - DNS Security Introduction and Requirements." RFC 4033 - DNS Security Introduction and Requirements. N.p., n.d. Web. 13 Dec. 2014.

"RFC 4034 - Resource Records for the DNS Security Extensions." RFC 4034 - Resource Records for the DNS Security Extensions. N.p., n.d. Web. 13 Dec. 2014.

"RFC 4035 - Protocol Modifications for the DNS Security Extensions." RFC 4035 - Protocol Modifications for the DNS Security Extensions. N.p., n.d. Web. 13 Dec. 2014.

"RFC 5155 - DNS Security (DNSSEC) Hashed Authenticated Denial of Existence." RFC 5155 - DNS Security (DNSSEC) Hashed Authenticated Denial of Existence. N.p., n.d. Web. 13 Dec. 2014.

"RFC 5155 - DNS Security (DNSSEC) Hashed Authenticated Denial of Existence." RFC 5155 - DNS Security (DNSSEC) Hashed Authenticated Denial of Existence. N.p., n.d. Web. 13 Dec. 2014.

"RFC 6014 - Cryptographic Algorithm Identifier Allocation for DNSSEC." RFC 6014 - Cryptographic Algorithm Identifier Allocation for DNSSEC. N.p., n.d. Web. 13 Dec. 2014.

"RFC 6840 - Clarifications and Implementation Notes for DNS Security (DNSSEC)." RFC 6840 - Clarifications and Implementation Notes for DNS Security (DNSSEC). N.p., n.d. Web. 13 Dec. 2014.

"RFC 6891 - Extension Mechanisms for DNS (EDNS(0))." RFC 6891 - Extension Mechanisms for DNS (EDNS(0)). N.p., n.d. Web. 13 Dec. 2014.

"Resolve-DnsName." Resolve-DnsName. N.p., n.d. Web. 13 Dec. 2014.

"Server Core for Windows Server 2012 R2 and Windows Server 2012." Server Core for Windows Server 2012 R2 and Windows Server 2012 (Windows). N.p., n.d. Web. 13 Dec. 2014.

"Step-by-Step: Demonstrate DNSSEC in a Test Lab." Step-by-Step: Demonstrate DNSSEC in a Test Lab. N.p., n.d. Web. 14 Dec. 2014.