DNSOP
Internet-Draft
Intended status: Best Current Practice
Expires: September 08, 2014

W. Hardaker
Parsons
O. Gudmundsson
Shinkuro Inc.
S. Krishnaswamy
Parsons

March 07, 2014

DNSSEC Roadblock Avoidance
draft-ietf-dnsop-dnssec-roadblock-avoidance-00.txt

Abstract

This document describes problems that a DNSSEC aware resolver/application might run into within a non-compliant infrastructure. It outline potential detection and mitigation techniques. The scope of the document is to create a shared approache to detect and overcome network issues that a DNSSEC software/system may face.

Table of Contents

1. Introduction

This document describes problems with DNSSEC ([RFC4034], [RFC4035]) deployment due to non-compliant infrastructure. It poses potential detection and mitigation techniques.

1.1. Background

Deployment of DNSSEC today is hampered by network components that make it difficult or sometimes impossible for validating resolvers to effectively obtain the DNSSEC data they need. This can occur for many different reasons including

- o Because neighboring recursive resolvers are not fully DNSSEC compliant
- o Because resolvers are not even DNSSEC aware
- o Because "middle-boxes" active block/restrict outbound traffic to the DNS port (53) either UDP and/or TCP .
- o Network component in path does not allow UDP fragments
- o etc...

This document talks about ways a Host Validator can detect the state of the network it is attached to, and ways to hopefully circumvent the problems associated with the network defects it discovers. The tests described in this document may be performed on any validating resolver to detect and prevent problems. While these recomendations are mainly aimed at Host Validators it it prudent to perform these test from regular Validatating Resolvers before enabling just to make sure things work.

1.2. Notation

When we talk about a "Host Validator", this can either be a library that an application has linked in or an actual validating resolver running on the same machine.

A variant of this is a "Validating Forwarding Resolver", which is a resolver that is configured to use upstream Resolvers if possible. Validating Forward Resolver needs to perform the same set of tests before using an upstream recursive resolver.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Goals

The result of this work is intended to show how a Host Validator can detect the capabilities of a nearby recursive resolver, and work around any problems that could potentially affect DNSSEC resolution. This enables the Host Validator to make use of the caching functionality of the recursive resolver, which is desirable in that it decreases network traffic and improves response times.

A Host Validator has two choices: it can wait to determine that it has problems with a recursive resolver based on the results that it is getting from real-world queries issued to it, or it can proactively test for problems (Section Section 3) to build a work around list ahead of time (Section Section 5). There are pros and cons to both of these paths that are application specific, and this document does not attempt to provide guidance about whether proactive tests should or should not be used. Either way, DNSSEC roadblock avoidance techniques ought to be used when needed and if possible.

3. Detecting DNSSEC Non-Compilance

A Host Validator may choose to determine early-on what bottlenecks exist that may hamper its ability to perform DNSSEC look-ups. This section outlines tests that can be done to test certain features of the surrounding network.

NOTE: when performing these tests against an address, we make the following assumtption about that address: It is a unicast address or an anycast cluster where all servers have identical configuration and connectivity.

3.1. Determining DNSSEC support in neighboring recursive resolvers

Ideally, a Host Validator can make use of the caching present in neighboring recursive resolvers. This section discusses the tests that a neighboring recursive resolver MUST pass in order to be fully usable as a near-by DNS cache.

Unless stated otherwise, all of the following tests SHOULD have the recursive flag set when sending out a query and SHOULD be sent over UDP. Unless otherwise stated, the tests MUST NOT have the DO bit set or utilize any of the other DNSSEC related requirements, like EDNS0. The tests are designed to check for one feature at a time.

3.1.1. Supports UDP answers

Purpose: This tests basic DNS over UDP functionality to a resolver.

Test: A DNS request is sent to the resolver under test for an A record for a known existing domain, such as www.dnssec-tools.org.

SUCCESS: A DNS response was received that contains an a A record in the answer section. (The data itself does not need to be checked.)

Note: an implementation MAY chose to not perform the rest of the tests if this test fails, as clearly the resolver under test is severely broken.

3.1.2. Supports TCP answers

Purpose: This tests basic TCP functionality to a resolver.

Test: A DNS request is sent over TCP to the resolver under test for an A record for a known existing domain, such as www.dnssec-tools.org.

SUCCESS: A DNS response was received that contains an A record in the answer section. (The data itself does not need to be checked.)

3.1.3. Supports EDNS0

Purpose: Test whether a resolver properly supports the EDNS0 extension option.

Pre-requisite: "Supports UDP or TCP".

Test: Send a request to the resolver under test for an A record for a known existing domain, such as www.dnssec-tools.org, with an EDNS0 OPT record in the additional section.

SUCCESS: A DNS response was received that contains an EDNS0 option with version number 0.

3.1.4. Supports the DO bit

Purpose: This tests whether a resolver has minimal support of the DO bit.

Pre-requisite: "Supports EDNS0".

Test: Send a request to the resolver under test for an A record for a known existing domain such as www.dnssec-tools.org. Set the DO bit in the outgoing query.

SUCCESS: A DNS response was received that contains the DO bit set.

Note: this only tests that the resolver sets the DO bit in the response. Later checks will determine if the DO bit was actually made use of. Some resolvers successfully pass this test because they simply copy the unknown flags into the response. Don't worry, they'll fail the later tests.

3.1.5. Supports the AD bit

Purpose: This tests whether the resolver is a validating resolver.

Pre-requisite: "Supports the DO bit".

Test: Send a request to the resolver under test for an A record for a known existing domain in a DNSSEC signed zone which is verifiable to a configured trust anchor, such as www.dnssec-tools.org using the root's published DNSKEY or DS record as a trust anchor. Set the DO bit in the outgoing query.

SUCCESS: A DNS response was received that contains the AD bit set.

3.1.6. Returns RRsig for signed answer

Purpose: This tests whether a resolver will properly return RRSIG records when the DO bit is set.

Pre-requisite: "Supports the DO bit".

Test: Send a request to the resolver under test for an A record for a known existing domain in a DNSSEC signed zone, such as www.dnssec-tools.org. Set the DO bit in the outgoing query.

SUCCESS: A DNS response was received that contains at least one RRSIG record.

3.1.7. Supports querying for DNSKEY records

Purpose: This tests whether a resolver can query for and receive a DNSKEY record from a signed zone.

Pre-requisite: "Supports the DO bit."

Test: Send a request to the resolver under test for an DNSKEY record which is known to exist in a signed zone, such as dnssec-tools.org/DNSKEY. Set the DO bit in the outgoing query.

SUCCESS: A DNS response was received that contains a DNSKEY record in the answer section.

Note: Some DNSKEY RRset's are large and if the network path has problems with large answers this query may result in either false positive or false negative. In general the DNSKEY queried for is a small enough to fit into 1220 byte answer, to avoid false negative result when TCP is disabled. However, querying many zones will result in answers greater than 1220 bytes so ideally TCP MUST be available.

3.1.8. Supports querying for DS records

Purpose: This tests whether a resolver can query for and receive a DS record from a signed zone.

Pre-requisite: "Supports the DO bit."

Test: Send a request to the resolver under test for an DS record which is known to exist in a signed zone, such as dnssec-tools.org/DS. Set the DO bit in the outgoing query.

SUCCESS: A DNS response was received that contains a DS record in the answer section.

3.1.9. Supports negative answers with NSEC records

Purpose: This tests whether a resolver properly returns NSEC records for a non-existing domain in a DNSSEC signed zone.

Pre-requisite: "Supports the DO bit."

Test: Send a request to the resolver under test for an A record which is known to not existing, such as non-existent.test.dnssec-tools.org. Set the DO bit in the outgoing query.

SUCCESS: A DNS response was received that contains an NSEC record.

Note: The query issued in this test MUST be sent to a NSEC signed zone. Getting back appropriate NSEC3 records does not indicate a failure, but a bad test.

3.1.10. Supports negative answers with NSEC3 records

Purpose: This tests whether a resolver properly returns NSEC3 records ([RFC5155]) for a non-existing domain in a DNSSEC signed zone.

Pre-requisite: "Supports the DO bit."

Test: Send a request to the resolver under test for an A record which is known to be non-existent, such as non-existent.nsec3-ns.test .dnssec-tools.org. Set the DO bit in the outgoing query.

SUCCESS: A DNS response was received that contains an NSEC3 record.

Note: The query issued in this test MUST be sent to a NSEC3 signed zone. Getting back appropriate NSEC records does not indicate a failure, but a bad test.

3.1.11. Supports queries where DNAME records lead to an answer

Purpose: This tests whether a resolver can query for an A record in a zone with a known DNAME referral for the record's parent.

Test: Send a request to the resolver under test for an A record which is known to exist in a signed zone within a DNAME referral child zone, such as good-a.dname-good-ns.test.dnssec-tools.net.

SUCCESS: A DNS response was received that contains a DNAME in the answer section. An RRSIG MUST also be received in the answer section that covers the DNAME record.

3.1.12. Permissive DNSSEC

Purpose: To see if a validating resolver is ignoring DNSSEC validation failures.

Pre-requisite: Supports the AD bit.

Test: ask for data from a broken DNSSEC delegation such as badsign-a.test.dnssec-tools.org.

SUCCESS: A reply with the Rcode set to SERVFAIL

3.1.13. UDP size limits

TBD

3.1.14. Supports Unknown RRtypes

Purpose: Some DNS Resolvers/gateways only support some RRtypes. This causes problems for applications that need recently defined types.

Pre-requisite: "Supports UDP or TCP".

Test: Send a request for recently defined type or unknown type in the 20000-22000 range, that resolves to a server that will return answer for all types, such as alltypes.res.dnssecready.org

SUCCESS: A DNS response was retrieved that contains the type requested in the answer section.

3.2. Direct Network Queries

If need be, a Host Validator may need to make direct queries to authoritative servers or known Open Recursive Resolvers in order to collect data. To do that, a number of key network features MUST be functional.

3.2.1. Support for Remote UDP Over Port 53

Purpose: This tests basic UDP functionality to outside the local network.

Test: A DNS request is sent to a known distant authoritative server for a record known to be within that server's authoritative data. Example: send a query to the address of ns1.dnssec-tools.org for the www.dnssec-tools.org/A record.

SUCCESS: A DNS response was received that contains an a A record in the answer section.

Note: an implementation can use the local resolvers for determining the address of the name server that is authoritative for the given zone. The recursive bit MAY be set for this request, but does not need to be.

3.2.2. Support for Remote UDP With Fragmentation

Purpose: This tests if the local network can receive fragmented UDP answers

Pre-requisite: Local UDP > 1500 is possible

Test: A DNS request is sent over UDP to a known distant DNS address asking for a record that has answer larger than 2000 bytes. Example send a query for the dnssec-tools.org/DNSKEY record with the DO bit set in the outgoing query.

Success: A DNS response was received that contains the large answer.

Note: A failure in getting large answers over UDP is not a serious problem if TCP is working.

3.2.3. Support for Outbound TCP Over Port 53

Purpose: This tests basic TCP functionality to outside the local network.

Test: A DNS request is sent over TCP to a known distant authoritative server for a record known to be within that server's authoritative data. Example: send a query to the address of ns1.dnssec-tools.org for the www.dnssec-tools.org/A record.

SUCCESS: A DNS response was received that contains an a A record in the answer section.

Note: an implementation can used the local resolvers for determining the address of the name server that is authoritative for the given zone. The recursive bit MAY be set for this request, but does not need to be.

4. Aggregating The Results

Some conclusions can be drawn from the results of the above tests in an "aggregated" form. This section defines some labels to assign to a resolver under test given the results of the tests run against them.

4.1. Resolver capability description

This section will group and label certain common results TBD

Resolvers are classified into following broad behaviors:

Validator: The resolver passes all DNSSEC tests and had the AD bit appropriately set.

DNSSEC Aware: The resolver passes all DNSSEC tests, but does not appropriately set the AD bit on answers, indicating it is not validating. A Host Validator will function fine using this resolver as a forwarder.

Non-DNSSEC capable: The resolver is not DNSSEC aware and will make it hard for a Host Validator to operate behind it. It MAY be usable for querying for data that is in known insecure sections of the DNS tree.

Not a DNS Resolver: This is a bad address and not used anymore.

While it would be great if all resolvers fell cleanly into one of the broad categories above, that is not the case. For that reason it is

necessary to augment the classification with more descriptive result, this is done by adding the word "Partial" in front of Validator/ DNSSEC Aware classifications, followed by sub-descriptors of what is not working.

Unknown: Failed Unknown test

DNAME: Failed DNAME test

NSEC3: Failed NSEC3 test

TCP: TCP not available

SlowBig: UDP is size limited but TCP fallback works

NoBig: TCP not available and UDP is size limited

Permissive: Passes data known to fail validation

5. Roadblock Avoidance

[Editors note: the goal of this document is to tie the above tests and aggregations to avoidance practices; however right now the "tie" part of this text is known to be weak. The goal of this document is specifically to improve this tie as work on this document progresses]

Once we have determined what level of support is available in the neighboring network, we can determine what MUST be done in order to effectively act as a validating resolver. This section discusses some of the options available given the results from the previous sections.

The general fallback approach can be described by the following sequence:

```
    If the resolver is labeled as "Validator" or "DNSSEC aware"

        Send query through this resolver and perform local
        validation on the results.

        If validation fails, try the next resolver

    Else if the resolver is labeled "Not a DNS Resolver" or
       "Non-DNSSEC capable"

        Mark it as unusable and try next resolver
```

Else if no more resolvers are configured and if direct queries are supported try iterating from Root

Else return a useful error code

While attempting resolution through a particular recursive name server with a particular transport method that worked, any transport-specific parameters MUST be remembered in order to short-circuit any unnecessary fallback attempts.

Transport-specific parameters MUST also be remembered for each authoritative name server that is queried while performing an iterative mode lookup.

Any transport settings that are remembered for a particular name server MUST be periodically refreshed; they should also be refreshed when an error is encountered as described below.

For a stub resolver, problems with the name server MAY manifest themselves as the following types of error conditions:

o No response/error response or missing DNSSEC meta-data.

o Illegal Response, which prevents the validator from fetching all necessary records required for constructing an authentication chain. This could result when referral loops are encountered, when any of the antecedent zone delegations are lame, when aliases are erroneously followed for certain RRtypes (such as SOA, DNSKEYs or DS records), or when resource records for certain types (e.g. DS) are returned from a zone that is not authoritative for such records.

o Bogus Response, when the cryptographic assertions in the authentication chain do not validate properly.

For each of the above error conditions a validator MAY adopt the following dynamic fallback technique, preferring a particular approach if it is known to work for a given name server or zone from previous attempts.

o No response, error response, or missing DNSSEC meta-data

  * Re-try with different EDNS0 sizes (4096, 1492, None)

  * Re-try with TCP only

* Perform an iterative query starting from Root if the previous error was returned from a lookup that had recursion enabled.

* Re-try using an alternative transport method, if this alternative method is known (configured) to be supported by the nameserver in question.

- Illegal Response

  * Perform an iterative query starting from Root if the previous error was returned from a lookup that had recursion enabled.

  * Check if any of the antecedent zones up to the closest configured trust anchor are provably insecure.

- Bogus Response

  * Perform an iterative query starting from Root if the previous error was returned from a lookup that had recursion enabled.

For each fallback technique, attempts to multiple potential name servers should be skewed such that the next name server is tried when the previous one encounters an error or a timeout is reached, whichever is earlier.

The validator SHOULD remember, in its zone-specific fallback cache, any broken behavior identified for a particular zone for a duration of that zone's SOA negative TTL.

The validator MAY place name servers that exhibit broken behavior into a blacklist, and bypass these name servers for all zones that they are authoritative for. The validator MUST time out entries in this name server blacklist periodically, where this interval could be set to be the same as the DNSSEC BAD cache default TTL.

5.1. Partial Resolver Usage

It MAY be possible to use Non-DNSSEC Capable caching resolvers in careful ways if maximum optimization is desired. This section describes some of the advanced techniques that could be used to use a resolver in at least a minimal way. Most of the time this would be unnecessary, except in the case where none of the resolvers are fully compliant and thus the choices would be to use them at least minimally or not at all (and no caching benefits would be available).

5.1.1. Known Insecure Lookups

If a resolver is Non-DNSSEC Capable but a section of the DNS tree has been determined to be Provably Insecure [RFC4035], then queries to this section of the tree MAY be sent through Non-DNSSEC Capable caching resolver.

5.1.2. Partial NSEC/NSEC3 Support

TBD

6. Start-Up and Network Connectivity Issues

A number of scenarios will produce either short-term or long-term connectivity issues with respect to DNSSEC validation. Consider the following cases:

Time Synchronization: Time synchronization problems can occure when a device which has been off for a period of time and the clock is no longer in close synchronization with "real time" or when a device always has clock set to the same time during start-up. This will cause problems when the device needs to resolve their source of time synchronization, such as "ntp.example.com".

Changing Network Properties: A newly established network connection MAY change state shortly after a HTTP-based pay-wall authentication system has been used. This especially common in hotel networks, where DNSSEC, validation and even DNS are not functional until the user proceeds through a series of forced web pages used to enable their network. The tests in Section Section 3 will produce very different results before and after the network authorization has succeeded. APIs exist on many operating systems to detect initial network device status changes, such as right after DHCP has finished, but few (none?) exist to detect that authentication through a pay-wall has succeeded.

There are only two choices when situations like this happen:

Continue to perform DNSSEC processing, which will likely result in all DNS requests failing. This is the most secure route, but causes the most operational grief for users.

Turn off DNSSEC support until the network proves to be usable. This allows the user to continue using the network, at the sacrifice of security. It also allows for a denial of security-service attack if a man-in-the-middle can convince a device that DNSSEC is impossible.

6.1. What To Do

TBD

7. IANA Considerations

No IANA actions are require to support this document

8. Security Considerations

This document discusses problems that may occur while deploying the secure DNSSEC protocol and what mitigation's can be used to help detect and mitigate these problems. Following these suggestions will result in a more secure DNSSEC operational environment than if DNSSEC was simply disabled when it fails to perform as expected.

9. Acknowledgments

Multiple lessons learned from multiple implementations led to the development of this document, including (in alphabetical order) DNSSEC-Tools' DNSSEC-Check, DNSSEC_Resolver_Check, dnssec-trigger, FCC_Grade.

The following people contributed to portions of this document in some fashion: ....

10. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005.

[RFC5155] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence", RFC 5155, March 2008.

Authors' Addresses

Wes Hardaker
Parsons
P.O. Box 382
Davis, CA 95617
US

Email: ietf@hardakers.net

Olafur Gudmundsson
Shinkuro Inc.
4922 Fairmont Av, Suite 250
Bethesda, MD 20814
USA

Email: ogud@ogud.com

Suresh Krishnaswamy
Parsons
7110 Samuel Morse Dr
Columbia, MD 21046
US

Email: suresh@tislabs.com