Version 11, February 22, 2013

Working Group 5
DNSSEC Implementation Practices for ISPs

Final Report on Measurement of DNSSEC Deployment

Table of Contents

Figures

Tables

1 Results in Brief

1.1 Executive Summary

Working Group 5's (WG5's) ultimate goal is to recommend best practices for Domain Name System Security Extension (DNSSEC) implementation. Discussions quickly revealed that the measurement of DNSSEC validation in Internet Service Providers (ISPs) is more complex than initially thought, leading to the need to rethink methods for measuring validation in ISPs.

ISP members noted that their networks, rather than being monolithic, may be highly complex and tend to be divided into divisions such as business wireline, residential wireline, wireless, and Web-hosting, all of which may provide different levels of Domain Name System (DNS) service. Given this, members were clear about the need to test and weigh validation levels carefully, particularly as several have developed relatively robust validation capabilities but choose not to activate them at this time for business reasons.

In light of ISPs' network complexity and other factors, Working Group 5 also devised a program in which testing of resolvers' DNSSEC capabilities was performed by Federal Communications Commission (FCC) contractor SamKnows and by Shinkuro, Inc. These two firms conducted surveys of U.S. ISPs and of all open recursive resolvers in the Internet Protocol version 4 (IPv4) address space, respectively, using a Java-based software that assesses a given resolver's DNSSEC capabilities.

The Shinkuro and SamKnows tests solely measured the extent to which an ISP had implemented DNSSEC protocols on a particular DNS server. They did not reveal any of the underlying business motivations for why ISPs might or might not have done so. In addition, these tests did not measure or identify network service impairments that might be caused by the use of DNSSEC, such as site inaccessibility due to a misconfiguration of DNSSEC in authoritative services. Finally, the tests did not gauge any ISP's future plans to deploy DNSSEC, instead merely gauging the extent of deployment at a given moment.

Based on these measurements, we found there is initial deployment of validating resolvers and substantial deployment of DNSSEC-aware resolvers that support end-system validation. We also found that significant proportions of these resolvers in both the validation and DNSSEC-aware categories support these functions only in part. The details of this partial support are documented in this report.

2 Introduction

Working Group 5, "DNSSEC Implementation Practices for ISPs," was asked by the FCC's Communications Security, Reliability and Interoperability Council (CSRIC) to examine "best practices for deploying and managing the Domain Name System Security Extensions (DNSSEC) by Internet service providers (ISPs). In addition, the Working Group shall recommend proper metrics and measurements that allow for evaluation of the effectiveness of DNSSEC deployment by ISPs." The Working Group's first report did not address the CSRIC's call to "recommend proper metrics and measurements," nor did it address how tests themselves would be conducted. These topics are the subjects of this final report.

This Working Group included input from a broad range of experts from both major ISPs and non-ISP organizations, who commented knowledgeably on DNSSEC's importance to the security of the DNS. The result is a final report that addresses the full range of ISP and other concerns about DNSSEC deployment, and helps clarify existing and potential obstacles to same along with potential solutions.

While some ISPs have deployed DNSSEC validation internally and for their customers, most have not. Other ISPs support their customers' use of DNSSEC without ISP participation in validation. As a result, another task of this Working Group was to recommend how ISPs might best accomplish DNSSEC deployment.

2.1 CSRIC Structure

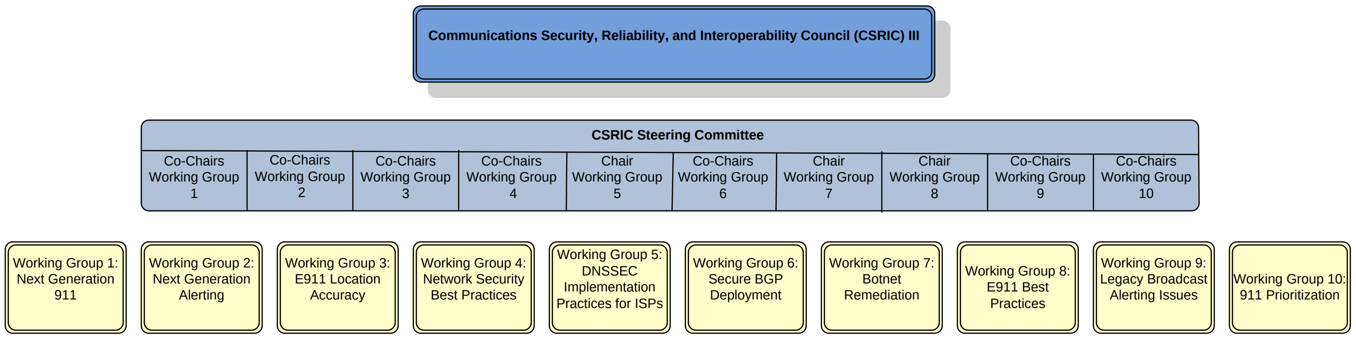

Communications Security, Reliability, and Interoperability Council (CSRIC) III
CSRIC Steering Committee
Co-Chairs Working Group 1
Co-Chairs Working Group 2
Co-Chairs Working Group 3
Co-Chairs Working Group 4
Chair Working Group 5
Co-Chairs Working Group 6
Chair Working Group 7
Chair Working Group 8
Co-Chairs Working Group 9
Co-Chairs Working Group 10
Working Group 1: Next Generation 911
Working Group 2: Next Generation Alerting
Working Group 3: E911 Location Accuracy
Working Group 4: Network Security Best Practices
Working Group 5: DNSSEC Implementation Practices for ISPs
Working Group 6: Secure BGP Deployment
Working Group 7: Botnet Remediation
Working Group 8: E911 Best Practices
Working Group 9: Legacy Broadcast Alerting Issues
Working Group 10: 911 Prioritization

Figure 1 – CSRIC structure

2.2 Working Group 5 Team Members

Working Group 5 consists of the members listed below.

Name	**Company**
Chair: Steve Crocker	Shinkuro
Daniel Awduche	Verizon
Warren Kumari	Google
Matt Larson	Verisign
Jason Livingood	Comcast
Chris Martin	Verizon
Daniel Mason	CenturyLink
Chris Mikkelson	CenturyLink
Doug Montgomery	NIST
Russ Mundy	Sparta/TIS Labs
Nodir Nazarov	Cablevision
Eric Osterweil	Verisign
Rod Rasmussen	Internet Identity
Brian Rexroad	AT&T
Scott Rose	NIST
Todd Szymanski	Sprint
Matt Williams	Cox
Suzanne Woolf	Internet Systems Consortium

Table 1 – List of working group members

The Working Group also had the benefit of assistance from:

Jeffrey Dewhurst
Mark Feldman
Olafur Gudmundsson
Paul Kretkowski
Bob Novas

3 Objective, Scope and Methodology

3.1 Objective

From the description of Working Group 5 on fcc.gov:[1]

"The Domain Name System Security Extensions (DNSSEC) are widely recognized as the best hope for improving the long-term security of the Internet's critical domain name system. Standards for DNSSEC are now mature and implementation has begun in the government as well as the enterprise sector."

"This Working Group shall recommend the best practices for deploying and managing the Domain Name System Security Extensions (DNSSEC) by Internet service providers (ISPs). *In addition, the Working Group shall recommend proper metrics and measurements that allow for evaluation of the effectiveness of DNSSEC deployment by ISPs.* [Emphasis by WG5.] In addition to any other metrics, the Working Group shall address the following: availability of a zone, verification of received data, and validation of verified data. Finally, the Working Group shall recommend ways for the ISP community to demonstrate their intent to deploy DNSSEC, possibly by way of a voluntary opt-in framework."

3.2 Limitations of This Final Report

This Working Group's scope of research was limited by time (specifically, a February reporting deadline). In addition, the following issues were considered outside this Working Group's scope:

a) Alternative approaches and countermeasures to protect against domain-name fraud
b) More holistic approaches to security outside of DNSSEC implementation, to avoid duplicating other CSRIC groups' work. This includes DNS server compromise; usability of small-office/home-office (SOHO) routers and other open DNS resolvers in attacks; increasing frequency, sophistication and size of DNS amplification attacks; DNSSEC inhibition of DNS redirection ("sinkholes") for security purposes; and alternative methods of detection and blocking of cache poisoning, since many of these have been explored in the report of CSRIC Working Group 4.[2]
c) The potential roles of alternative DNS resolver providers (e.g. resolvers operated by enterprises on behalf of their users, or resolvers within end systems)
d) Implementation of DNSSEC in the ISPs' own authoritative zones, which is not directly related to general DNSSEC validation by recursive resolvers

3.3 Methodology

The Working Group proceeded along three stages, each of which consisted of one or more steps, in conducting its research and analysis:

[1] Available for download at http://www.fcc.gov/pshs/advisory/csric3/wg-descriptions_2-17-12.pdf . The announcement pertaining to all of the working groups is at http://www.fcc.gov/encyclopedia/communications-security-reliability-and-interoperability-council-iii
[2] Download WG4's report from http://transition.fcc.gov/bureaus/pshs/advisory/csric3/CSRICIII_9-12-12_WG4-FINAL-Report-DNS-Best-Practices.pdf

Methodology

- Form working group with expertise
- Query working-group members regarding hurdles, challenges, etc.

Analysis

- List specific issues, formulate approach for each
- Note details of issue resolution
- Conduct testing

Findings

- Collate results of testing and other analysis
- Gauge consensus and generate recommendations

This Working Group included a broad range of participants among both ISPs (D. Awduche, M. Burns, J. Livingood, C. Martin, D. Mason, C. Mikkelson, N. Nazarov, B. Rexroad, T. Szymanski, M. Williams) and non-ISP experts who have been part of the DNSSEC deployment effort (S. Crocker, W. Kumari, M. Larson, D. Montgomery, R. Mundy, R. Rasmussen, S. Woolf). Mark Feldman, Olafur Gudmundsson and Bob Novas provided additional technical expertise related to DNSSEC.

The Working Group was queried via a series of teleconferences and e-mail exchanges designed to elucidate the issues surrounding ISPs' measurement of DNSSEC support, and to discuss the results of tests measuring ISPs' DNSSEC capabilities.

4 Background

4.1 Brief Overview of Internet Service Providers (ISPs)

In our discussions, we learned that ISP operations are more complex than we had initially thought, causing significant changes to our thinking about how to measure ISP validation levels. Indeed, many of the largest U.S. ISPs are far from operationally monolithic; most if not all run more than one network to serve different groups of users more efficiently, and customers may access just one of those networks without using any of the others.

Rather than speak of ISPs as being a single, unitary entity, then, we began considering ways to accurately test as many of the ISPs' sub-networks as practical, realizing that some may themselves be heterogeneous due to varying hardware, software, connectivity, completeness of absorption of recent acquisitions, and/or other factors.

The four most common sub-network types, which we believe account for the great majority of major ISPs' traffic, are those geared toward:

- Wireline business customers
- Wireline residential customers
- Wireless customers
- Customers using the ISP’s hosting service

Some ISPs might not have all four of these sub-networks, while others may have one or more additional sub-networks (e.g., for managed-service customers) We attempted to account for these complexities in devising and executing Shinkuro tests of open recursive resolvers in the IPv4 space, and collating results from SamKnows.

4.2 What Should Be Measured?

Discussions and testing since the previous report's publication caused this Working Group to realize that ISPs' networks may be sufficiently complex that it would be difficult, given time and confidentiality constraints, to:

- Accurately model each ISP's network, some of which may not be publicly accessible;
- Perform validation testing of various sub-networks; and
- Collate the results into a shorthand that would be both fair to ISPs and easily understood by non-specialist readers.

These factors caused the Working Group to replace the previous report's A-through-D rating system with a more nuanced, less judgmental system that is described in Section 4.3.2. Facets of the group's discussions are captured in the following sections.

4.2.1 Issue: The Need to Measure Multiple Resolvers

At the outset of this round of discussions, Working Group 5 realized that ISP networks are far from homogeneous in terms of their hardware, software and protocols, making it difficult to assign any ISP a single rating that accurately reflects its DNSSEC implementation.

For example, AT&T reported that it has four categories of service, each of which is homogeneous, while other ISPs may—because of the presence of legacy technologies, internal organizational complexity, incomplete mergers and acquisitions, or other factors—have dozens of internal divisions with differing levels of DNSSEC validation.

The Working Group requested that ISPs broadly describe their networks and their different internal classes of DNS service since those may vary. ISP members agreed to describe their networks broadly and perform testing on a representative internal division to the extent that the group decided. For simplicity's sake, some WG5 members advocated choosing just a few different resolver types (or business divisions) that account for a majority of a given ISP's traffic, and only measuring the validation level of a representative sample of those resolvers on the way to assigning a rating.

4.2.2 Issue: Need to Provide Relative Population Measurement

Some resolvers provide service for a larger proportion of Internet users than others, which will hold true even among a given ISP's resolvers. Consequently, measurement of DNSSEC support depends on the number of users that are using those resolvers for DNS services. It is difficult if not impossible to accurately measure the number of unique users of a given resolver for multiple reasons:

- A user may not exclusively use one resolver service. As users move between consumer wireline, Wi-Fi, enterprise, and mobility networks, the same user may be counted multiple times in different services.
- Many networks assign IP addresses dynamically. In some circumstances, addresses can be reassigned in hours, minutes, or even seconds. Dynamic address assignment creates a level of indirection between the end user and DNS resolvers, since it is often difficult to trace IP address assignments to specific users. By the same token, it is also generally very difficult to perform this function for large numbers of users.

An alternative approach to approximate user population mentioned was to measure the number of queries managed by a given DNS resolver, and weight the use of said resolver in DNSSEC deployment measurements.

4.2.3 Issue: The Need to Measure Validation Repeatedly

For various reasons, a given sub-network's validation levels must be measured more than once. For example, DNSSEC validation could have been turned off or could have failed to function properly due to a technical or operational error, or the fact that a resolver is currently withstanding a "TCP storm" or other type of attack that causes the resolver to stop accepting Transmission Control Protocol (TCP) traffic. In cases such as this, an ISP sub-network's rating could be adversely affected at one point in time, while a subsequent test just moments later might yield a much better rating. Because of this, WG5 advocated for serial testing of a given resolver.

We also considered the fact that the Internet is not static. Services, servers, software, and support functions constantly evolve to increase capacity, enrich features, and protect services. Both nearterm and long-term trend data are necessary to create meaningful measurements.

4.2.4 Issue: Treatment of TCP Queries

Preliminary testing of some ISP resolvers uncovered a specific issue with regard to TCP-based queries: Some ISPs purposefully configure their resolvers not to accept queries delivered via TCP instead of User Datagram Protocol (UDP).

In the long term, WG5 believes that ISPs should accept queries via both UDP and TCP in order to be fully standards-compliant and to be in line with best practices.

4.2.5 Issue: ISPs May Be "Permissive"

Some ISPs are permissive regarding DNSSEC. This means that the ISP performs the full gamut of DNSSEC-validation tasks but if a target server returns invalid DNSSEC results, the ISP nonetheless displays that server's content to a user rather than a "SERVFAIL" message. This type of resolver will not return the Authenticated Data (AD) bit in its response, thus allowing a DNSSEC-aware client to determine the proper behavior, while behaving in a legacy fashion for non-DNSSEC aware clients.

4.2.6 Issue: Middlebox Challenges

Many residential and business customers connect to their ISP through a wireless access router (a type of middlebox) to enable wireless connectivity and sharing of the network connection among multiple devices. The wireless router typically provides a DNS proxy, whose address is provided to customer devices via the Dynamic Host Configuration Protocol (DHCP). Similar to ISP DNS infrastructure, this access router and DNS proxy must meet applicable standards to support DNSSEC, but unfortunately many fail to do so. (This very simple example ignores the possibility that the ISP, middlebox, and intervening proxies may all be provided or controlled by different parties.)

This presents a challenge to both DNSSEC adoption as well as to some approaches (e.g., crowdsourced tests) to measuring an ISP’s support for DNSSEC. For the latter, measurements can be affected by the differing types of access routers that customers may be using. To compensate for this phenomenon, a variety of measurement approaches were proposed to account for these varied access routers.

4.2.7 Issue: Measurement of DNSSEC Abuses

In order to measure the value of any security function, the merits of that security function need to be weighed against the abuses and impairments that are introduced as part of that security function. For example, a computer that cannot be accessed may be very secure, but it is also not very useful. In this context, given the lack of documented cases where DNSSEC can provide legitimate protections, the security value of DNSSEC must be weighed against the security protections in current practice as well as potential abuses of DNSSEC.

There are known and yet-undiscovered activities associated with abuse of DNSSEC, as cited in WG5's previous report. In particular, DNSSEC is known to be used to facilitate DNS reflection/amplification attacks, and there is an increasing trend involving this activity. It is currently common practice to use DNS redirection as a means to thwart advanced persistent threat (APT) malware, but as DNSSEC becomes more readily deployed, this method of protection will fail. Further, the malware may use DNSSEC validation to determine whether DNS redirection is in use as a means to detect and thwart it.

Countermeasures for these growing threats are needed, or DNSSEC's merits may be swamped by the attack liability it creates. Consequently, it is only prudent to measure and track the trends in DNS abuse as they relate to DNSSEC. A watch should be conducted to detect and identify as-yet unknown abuses.

4.3 What Was Measured

Taking into account the above factors as well as time and technical limitations, our primary tool for measuring ISP DNSSEC validation was a Java-based program called DNSSEC Resolver Check, which was used by Shinkuro, Inc. and by FCC contractor SamKnows. Each took a slightly different snapshot of ISP traffic, and in Shinkuro's case, DNSSEC Resolver Check became part of a more comprehensive program of testing for the presence and DNSSEC status of open recursive resolvers across the IPv4 space.

4.3.1 DNSSEC Resolver Check

The two parties ran a Java-based program called DNSSEC Resolver Check (created by Olafur Gudmundsson of Shinkuro, Inc.). This program, available at https://github.com/ogud/DNSSEC-resolver-check, enables users to check the resolver they are currently using, or any other resolver they designate, for its level of validation capability. (DNSSEC Resolver Check automatically communicates its results back to a server at Shinkuro's offices, where those results were collected and analyzed.)

DNSSEC Resolver Check imposes a series of up to 13 tests that measure how able the target resolver is to perform DNSSEC validation. The tests are performed in sequence; test 1 is performed in all cases, tests 2–12 are performed if a resolver passes test 1, and test 13 is only performed if a resolver passes tests 1–12.

Table 3 below shows the various tests that DNSSEC Resolver Check performs along with their potential results:

Test	Test Description: Target Resolver ...	Test Query Used (Name/ RR Type)	Possible Results
T1	Correctly answers a DNS query	com./SOA	**Pass:** An answer is returned **Fail:** No answer is returned
T2	Supports Extension mechanisms for DNS (EDNS)	org/DNSKEY	**Pass:** An answer is returned w/OPTION pseudo (OPT) RR **Fail:** No answer is returned or no OPT RR seen
T3	Correctly supports unknown resource record (RR) types	tlsa.ogud.com/ TLSA	**Pass:** An answer is returned w/unknown type **Fail:** No answer is returned
T4	Listens on TCP	net/SOA via TCP	**Pass:** An answer is returned **Fail:** No answer is returned
T5	Understands Delegation Name (DNAME) processing	grade.goal.ogud .com/TXT	**Pass:** Correct DNAME processing in final response **Fail:** No DNAME response or incorrect CNAME processing
T6	Can process responses >512 bytes	N/A	**Pass:** An answer packet >512 bytes is received **Fail:** Answer packet is truncated or no answer is received
T7	Validates DNSSEC signatures	iab.org/SOA	**Pass:** Answer is returned w/the AD bit set, indicating authentication check performed **Fail:** Answer is returned w/o the AD bit set, or no answer is returned
T8	Correctly queries for DS RR type	ietf.org/DS	**Pass (AD):** Answer returned w/the DS RR and AD bit set **Pass (no AD):** Answer returned w/the DS RR **Fail:** No answer returned or "No Data" error message
T9	Returns correct signed DNAME response	grade.goal.ogud .com/TXT	**Pass (AD):** Answer returned w/the DNAME RR and AD bit set **Pass (no AD):** Answer returned w/correct processing **Fail:** No answer returned or "No Data" error message
T10	Understands authenticated denial of existence using Next Secure NSEC RRs	us/SPF	**Pass (AD):** Answer returned w/NSEC RRs and AD bit set **Pass (no AD):** Answer returned w/NSEC RRs **Fail:** No answer returned or "No Data" error message
T11	Understands authenticated denial of existence using Next Secure 3 (NSEC3) RRs	de/SPF	**Pass (AD):** Answer returned w/NSEC RRs and AD bit set **Pass (no AD):** Answer returned w/NSEC RRs **Fail:** No answer returned or "No Data" error message
T12	Supports large UDP packets	shinkuro.net/A	**Pass (AD):** Answer returned in UDP packet w/AD bit set **Pass (no AD):** Answer returned in UDP packet but w/o AD bit set **Fail:** No answer returned or truncated response
T13	Returns bogus (invalid) DNSSEC responses	dnssec-failed .org/SOA	**Pass:** SERVFAIL error message returned **Fail:** Response returned w/o AD bit set

Table 2 – DNSSEC Resolver Check tests and test sequence

4.3.2 Applying Descriptors to Resolvers

Once the results of tests 1–13 were known, several "descriptors" may be applied to the resolver. These descriptors consist of a Major Behavior descriptor and, if necessary, an additional set of subdescriptors.

(As a separate matter, we noted that the series of 13 DNSSEC Resolver Check tests does not test all the possible ways for a resolver to fail to meet the formal requirements for compliance with DNSSEC-related Requests for Comment (RFCs) (e.g., RFC 4035). For example, the tests conducted on WG5's behalf did not include a check of whether a given resolver supports all the current DNSSEC algorithms.)

Behavior	**Rationale Based on Test Results**
Validator	The resolver passed all 13 DNSSEC Resolver Check tests with the AD bit set indicating that it performed validation.
Partial Validator(…)	The resolver was a validator but did not pass all 13 tests, with Additional Information subdescriptors within parentheses indicate the deficiencies. (See the following table for subdescriptors.)
DNSSEC Aware	The resolver passed all of the tests relevant to supporting DNSSEC validation by its clients. Specifically, this means the resolver passed all of tests T1–T12, but did not have the AD bit set in at least one of tests T7–T12.
Partially DNSSEC Aware(…)	The resolver was DNSSEC-aware but did not pass all of the tests. Additional Information subdescriptors within parentheses indicate the deficiencies. (Again, see the following table for subdescriptors.)
Non-DNSSEC-Capable DNS Resolver	The resolver is not DNSSEC aware and does not support validation by its clients. Abbreviated "NDR".
Not a DNS Resolver	The entity probed is not a DNS resolver. Abbreviated "NAR".
Error (...)	The entity probed passed T1 but failed subsequent tests, typically by timing out or exhibiting anomalous behavior. The specific reason is appended within parentheses. Abbreviated "ERR".

Table 3: Major Behavior descriptors

The following table shows the Additional Information subdescriptors to be shown inside parentheses following a Partial Validator or Partially DNSSEC Aware descriptor.

Notation	**Means**
Unknown	Failed to pass T3 (support for unknown RR types).
DNAME	Failed to pass T5 (support for DNAMEs). DNAMEs are important and, for certain zones, essential, but their use is still somewhat limited. We expect that Partial Validator or Partially DNSSEC Aware implementations that do not yet support DNAMEs will do so relatively soon.
NSEC3	Failed to pass T11 (support for NSEC3). While NSEC3 is crucial and is used by .org, .gov and .com, it was not part of the original DNSSEC specification, and some otherwise compliant software implementations do not recognize NSEC3 records. We expect that this limitation would be corrected relatively quickly.
TCP	Failed to pass T4 but did pass T12. The two tests are linked because they relate to different strategies for handling large responses. Some resolvers are purposefully configured not to accept queries over TCP.
SlowBig	Failed to pass T12 but did pass T4. T12 tests whether large responses[3] can be returned successfully. A resolver may fail this test either because it is not configured to respond with answers that large, or because the path between the querying system and the resolver cannot pass packets that large or the receiving system does not support packet reassembly. Querying systems can fall back to TCP.
NoBig	Failed to pass either T4 or T12. This combines the TCP and SlowBig annotations but since the effect of this combination is total failure to handle large responses, we have assigned it a separate designator.
Permissive	Failed to pass T13.

Table 4: Additional Information subdescriptors

4.3.2.1 Deployment Descriptions

Here are how passing or failing specific combinations of tests leads to the Major Behavior descriptors and Additional Information subdescriptors described above.

The primary output of the DNSSEC Resolver Check app is a 13-character string called a DNSSEC Deployment Descriptor (DDD). (Note: sometimes the app outputs only the string “NAR”. This means that the app has determined the probed entity is Not A Resolver.) The DNSSEC Deployment Descriptor strings look like this:

PPPPPPAAAAAAP

[3] "Large" in this context means between 1,645 and 2,089 bytes. The variation is due to the variation in behavior among resolvers; some resolvers include authoritative delegation information in their answers, others do not.

Each character in the string, from left to right, corresponds to one of the numbered tests of DNSSEC-aware behavior; the leftmost letter corresponds to Test 1, the second letter to Test 2, and so on. Each character in the description string is a letter from the following list:

S - test skipped
T - test timed out
A - test passed and the AD flag bit is set in the response
P - test passed and the AD flag bit is not set
F - test failed and the AD flag bit is not set
X - test failed and the AD flag bit is set

To translate from the DDD string to a more understandable and useful shorthand, which we refer to above as a Major Behavior (or simply "Behavior"), we apply the process described below. We'll first present a somewhat loose description of the translation process in English, and then a more formal description using a Deterministic Finite Automaton as a pattern recognizer. (The descriptions relate to the tests described above in Table 3, "DNSSEC Resolver Check tests and test sequence".)

If the DDD string consists of the text "NAR" then the Behavior is NAR (as noted above, Not A Resolver) and the translation is finished. In addition, if the entity probed cannot answer T1, which is a DNS query for .com's Start of Authority (SOA) record, then it is also determined to be NAR and the translation is finished.

Next, if any of tests 2 through 13 timed out (e.g., are T in the DDD string), then the Behavior is Timeout and the translation is finished.

The entity is determined to be a "modern" resolver if it supports extension mechanism for DNS 0 (EDNS0) (i.e., it passes T2). A modern resolver should also understand DNAMEs (pass T5), answer over TCP (pass T4), return answers longer than 512 bytes (pass T6) and support New/Unknown types (pass T3), but we allow resolvers to fail these tests. In these cases, however, we append a modifier to the behavior to indicate the specific failure(s).

In order to have the DNSSEC Aware Behavior descriptor applied, the probed entity must pass all the following tests:

- Return an answer for a query of a signed SOA record without setting the AD bit (pass T7 with a P)
- Return Delegation Signer (DS) records without setting the AD bit (pass T8 with a P)
- Return NSEC records without setting the AD bit (pass T10 with a P)

In addition, the resolver should handle DNAMEs correctly (pass T9), correctly return NSEC3 records (pass T11 with a P) and support large records (pass T12), but we allow resolvers to fail these tests. In these cases, however, we append an Additional Information subdescriptor to the behavior to indicate the specific failure(s).

If the entity passes the mandatory tests for the DNSSEC Aware Behavior with an A instead of a P, then the Validator Behavior descriptor applies.

Next, we provide a more formal description of the algorithm using a Finite State Automaton (FSA). Table 6 is the state transition table for the automaton that recognizes the behaviors mentioned above. Each column in the table labeled 1–13 corresponds to a letter in the 13-character behavior string. In order for a DDD string to match a Behavior or Additional Information pattern in this table, each letter of the string must match the corresponding column in the table. Note, however, that a period indicates that any test result is acceptable in the cell in which the period appears.

Table 5 – Translation state transition table

Test State	1	2	3	4	5	6	7	8	9	10	11	12	13	Not Match	Match
0	NAR													1	=NAR.
1	FT	.	.	.	.	.	.	.	.	.	.	.	.	2	=NAR.
2	.	PFX	PFX	PFX	PFX	PF	APF	APF	APF	APF	APF	APF	.	=Timeout.	3
3	.	PF	PF	PF	PF	.	.	.	.	.	.	.	.	=Anomalous.	4
4	P	P	.	.	.	P	AP	AP	.	AP	.	.	.	=NOTDNSSEC.	5
5	.	.	.	.	.	.	P	P	.	P	.	.	.	6	7
6	.	.	.	.	.	.	.	.	.	.	.	.	.	=Validator	6a
6a	.	.	.	.	P	.	.	.	A	.	.	.	.	DNAME	6b
6b	.	.	.	F	.	.	.	.	.	.	.	A	.	6c	TCP
6c	.	.	.	.	.	.	.	.	.	.	.	.	F	6d	Permissive
6d	.	.	.	.	.	.	AF	AF	AF	AF	AF	AF	AF	8a	Mixed
7	.	.	.	.	.	.	.	.	.	.	.	.	.	7a	=DNSSEC Aware
7a	.	.	.	.	P	.	.	.	AP	.	.	.	.	DNAME	7b
7b	.	.	.	F	.	.	.	.	.	.	.	.	.	8a	TCP
8a	.	.	.F	.	.	.	.	.	.	.	.	.	.	8b	Unknown
8b	.	.	.	.	.	.	.	.	.	.	F	.	.	8c	NSEC3
8c	.	.	.	P	.	.	.	.	.	.	.	F	.	8d	SlowBig
8d	.	.	.	F	.	.	.	.	.	.	.	F	.	.	NoBig
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.

The two rightmost columns of the table represent the action to be taken if the pattern match either fails (“Not Match”) or succeeds (“Match”). If the action to be taken can be found in the leftmost column of the table (labeled “State”), then it is the next state of the FSA. Otherwise, it is a value to be emitted as part of the Behavior and the other action is the next state. If the action to be taken is a value to be emitted and starts with an equals sign (“=”), then it is the Major Behavior; otherwise, it is an Additional Information modifier. If an action to be emitted ends with a period (“.”), then the action is terminal.

Applying the translation correctly results in Behavior descriptors such as Validator, DNSSEC Aware, etc. as described above. (If in addition one or more modifiers apply, we prepend the behavior name with “Partial”.)

4.3.3 SamKnows' Use of DNSSEC Resolver Check

The following shows the resolver behaviors derived from three weeks of data from SamKnows.[4] All the results shown are backed by at least four identical results for the same probe; in other words, the probes ran between 1–13 tests against each resolver address, and if a resolver address produced four or more identical result strings, that resolver was assigned a Behavior.

[4] The SamKnows DNSSEC testing and other test activities described in this document are *not* part of the Measuring Broadband America Report. Any such expansion would require vetting by the FCC with the current Broadband Collaborative, with which the FCC is collaborating on the other aspects of SamKnows testing.

There were a total of 14,180 results to collate and assign descriptors to. About 2% of the results are Exceptions and all of them have an X for test 12. These can be treated as F.

Arranged from most supportive DNSSEC Behavior to least, these were the Behaviors assigned to the 14,180 SamKnows results:

- 1,677 (11.83%) were Validators
- 840 (5.92%) were Partial Validators
- 4,703 (33.17%) were DNSSEC Aware
- 2,362 (16.66%) were Partial DNSSEC Aware
- 3,935 (27.75%) were Old-Style Resolvers
- 430 (3.03%) were Not a Resolver
- 233 (1.64%) were Exceptions and were not described

Below are the numbers of results for the Partial Validator and Partial DNSSEC Aware Behaviors that required an Additional Information subdescriptor:

Table 6 – SamKnows numeric results

	840 Partial Validators, Including:
762	Partial Validator[DNAME]
39	Partial Validator[DNAME; MIXED]
13	Partial Validator[SlowBig]
12	Partial Validator[TCP]
4	Partial Validator[MIXED]
3	Partial Validator[DNAME; MIXED; NoBig]
3	Partial Validator[DNAME; SlowBig]
2	Partial Validator[NoBig]
1	Partial Validator[TCP; Permissive]
1	Partial Validator[DNAME; NoBig]
	2,362 Partial DNSSEC Aware, Including:
1,837	Partial DNSSEC Aware[NoBig]
264	Partial DNSSEC Aware[TCP]
214	Partial DNSSEC Aware[SlowBig]
30	Partial DNSSEC Aware[DNAME]
6	Partial DNSSEC Aware[DNAME; SlowBig]
5	Partial DNSSEC Aware[DNAME; NoBig]
5	Partial DNSSEC Aware[NSEC3]
1	Partial DNSSEC Aware[TCP; NSEC3]

Figure 2 below graphically represents the proportions of the SamKnows results:

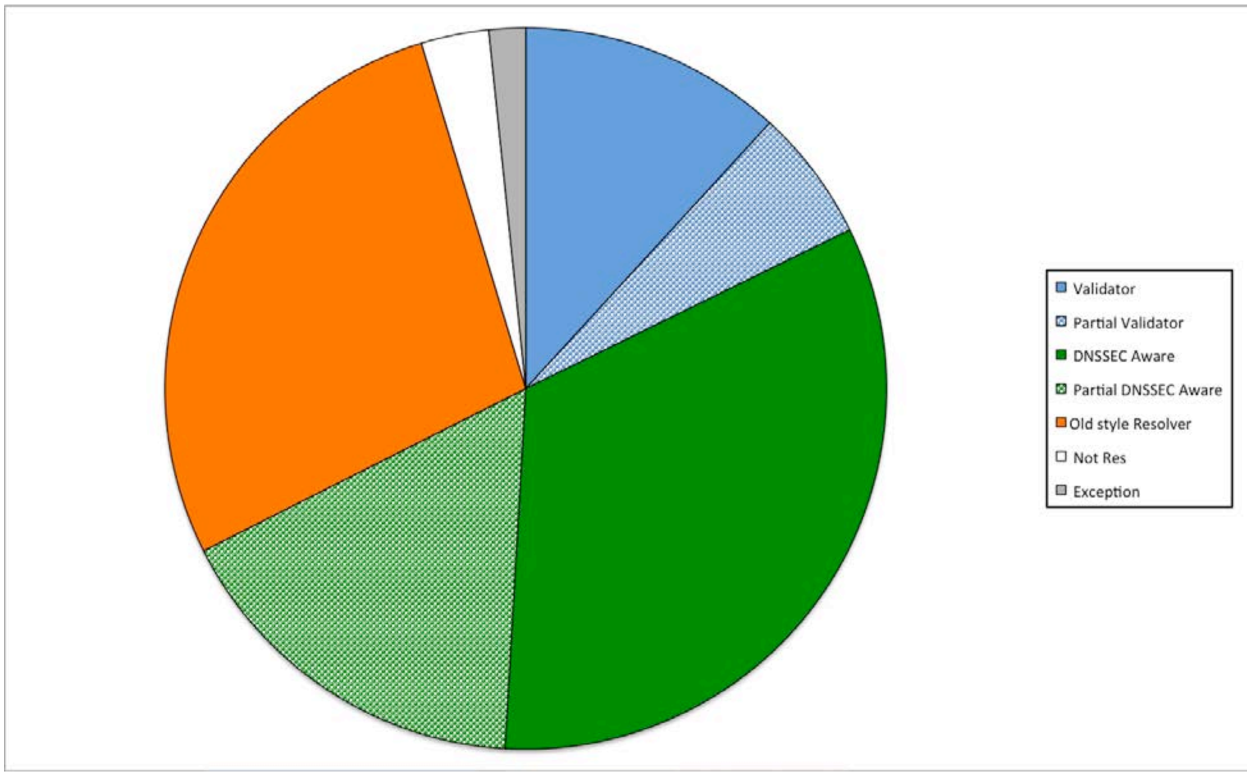


Figure 2 – Proportions of behaviors from SamKnows results

Note that, based on the types of failures that contribute to Table 7 and Figure 2, changes in how the tested resolver implementations handled DNAME and TCP would quickly add to the ranks of those that were either Validators or DNSSEC Aware.

4.3.4 Shinkuro Testing Program

In addition to tests using DNSSEC Resolver Check, Shinkuro, Inc. conducted an experiment to discover and categorize open recursive resolvers at addresses in the Internet IPv4 space, and the following describes the approach, tool chain used and results to date from a scan of the Internet conducted 24 January 2013 through 25 January 2013. The scan was actually conducted in four successive timeframes due to glitches in how responses were parsed, with the results of these four parts then “stitched” together.

Although the Shinkuro program incorporates DNSSEC Resolver Check testing, the experiment was implemented via a "pipeline" of several programs, and its overall approach is shown schematically in Figure 3, below.

Each program reads input, interacts with the Internet, and then produces output that is consumed by the next stage in the pipeline. The pipeline runs in a continuous batch mode; each stage produces a number of output elements into a file, then closes that file, opens a new file, and again produces a number of output elements into that file. The pipeline continues in that fashion until all the inputs are satisfied, and the next stage in the pipeline can run whenever there is a file available from the previous stage.

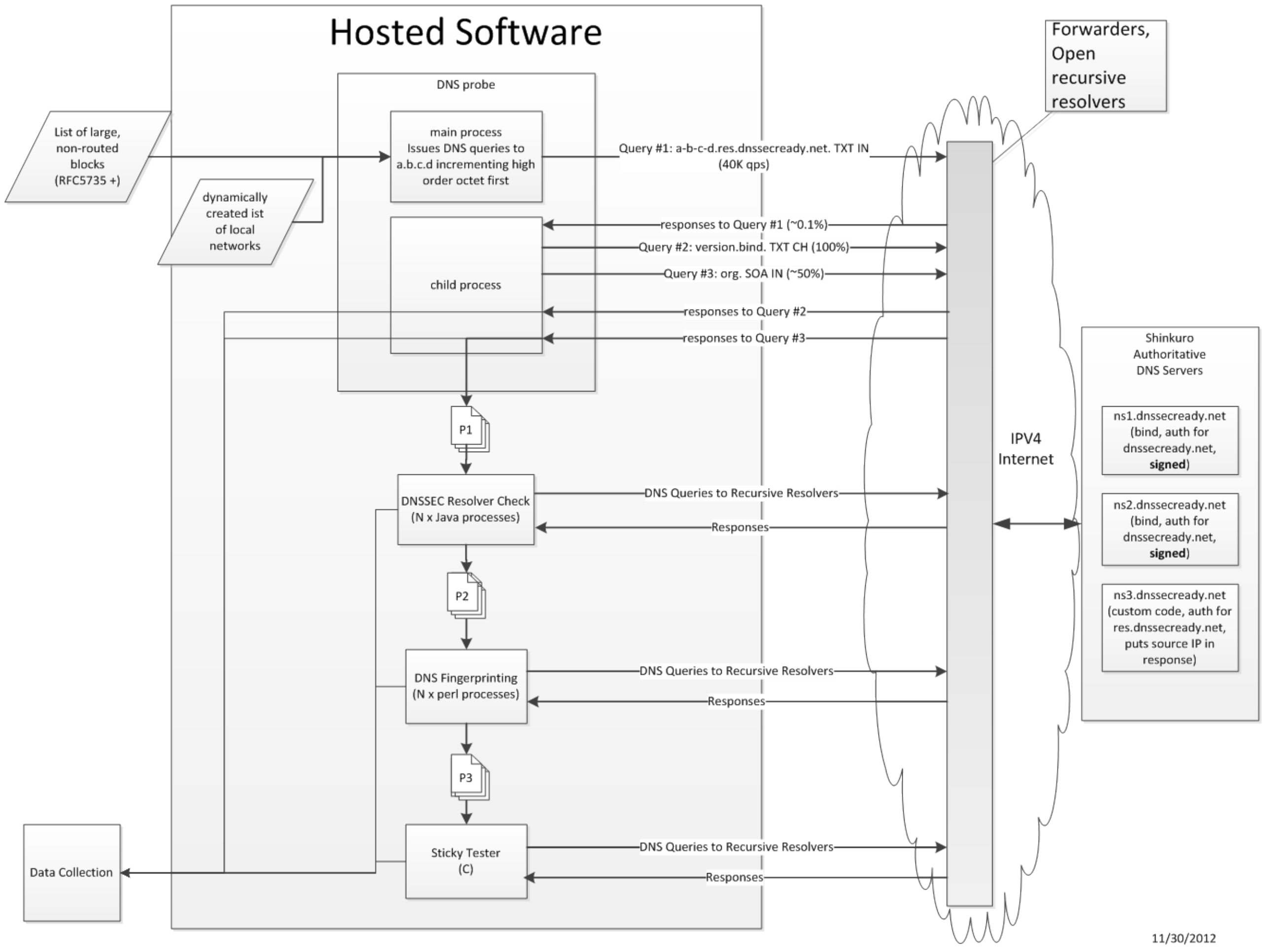

Hosted Software
DNS probe
List of large, non-routed blocks (RFC5735 +)
dynamically created ist of local networks
main process Issues DNS queries to a.b.c.d incrementing high order octet first
Query #1: a-b-c-d.res.dnssecready.net. TXT IN (40K qps)
child process
responses to Query #1 (~0.1%)
Query #2: version.bind. TXT CH (100%)
Query #3: org. SOA IN (~50%)
responses to Query #2
responses to Query #3
Forwarders, Open recursive resolvers
P1
DNSSEC Resolver Check (N x Java processes)
DNS Queries to Recursive Resolvers
Responses
P2
DNS Fingerprinting (N x perl processes)
DNS Queries to Recursive Resolvers
Responses
P3
Sticky Tester (C)
DNS Queries to Recursive Resolvers
Responses
Data Collection
IPV4 Internet
Shinkuro Authoritative DNS Servers
ns1.dnssecready.net (bind, auth for dnssecready.net, signed)
ns2.dnssecready.net (bind, auth for dnssecready.net, signed)
ns3.dnssecready.net (custom code, auth for res.dnssecready.net, puts source IP in response)
11/30/2012

Figure 3 – DNSSEC Internet probe experiment

4.3.4.1 The dnsprobe_5 application and the DNS Queries it sends

The first part of the experiment is implemented by dnsprobe_5, shown at the top center of Figure 3 above. The main process of dnsprobe_5 generates IP addresses in order from some start address to an end address. They are generated such that the high-order octet of the IPv4 address increments fastest (e.g., address 1.0.0.0 is followed by 2.0.0.0, 3.0.0.0, etc.). The main thread checks each address against two filters, the first being a list of non-routed Internet addresses that is static, as shown in Table 6. The second filter is generated dynamically from the getifaddrs() function, which creates a list that describes the local system's network interfaces. The Internet addresses occupied by these interfaces are also filtered.

Table 8 – Non-routed Internet addresses

0.0.0.0/8 RFC 1122
7.0.0.0/8
9.0.0.0/8
10.0.0.0/8 RFC 1918, 11.0.0.0/8
19.0.0.0/8
21.0.0.0/8, 22.0.0.0/8
25.0.0.0/8, 26.0.0.0/8
28.0.0.0/8
33.0.0.0/8
48.0.0.0/8
51.0.0.0/8
56.0.0.0/8
102.0.0.0/8
104.0.0.0/8
126.0.0.0/8, 127.0.0.0/8 RFC 1122
169.254.0.0/16 RFC 3927
172.16.0.0/12 RFC 1918
191.0.0.0/8, 192.0.0.0/24 RFC 5736
192.0.2.0/24 RFC 5737
192.88.99.0/24 RFC 3068
192.168.0.0/16 RFC 1918
198.18.0.0/15 RFC 2544
198.51.100.0/24 RFC 5737
200.160.2.0/24, 200.160.3.0/24 NIC.br NOC
200.160.5.0/24 NIC.br NOC
203.0.113.0/24 RFC 5737
224.0.0.0/4 RFC 3171
240.0.0.0/4 RFC 1112
255.255.255.255/32 RFC 919

4.3.4.2 dnsprobe_5 Query 1: "a-b-c-d.res.dnssecready.net TXT IN"

The main thread sends a DNS query to port 53 on each address that is not filtered by the above-described filters. This query is a DNS TXT query of the form `a-b-c-d.res.dnssecready.net`, where a-b-c-d is an IP address encoded with dashes separating octets instead of dots. If it were a dig command, it would look like this:

```
    dig @a.b.c.d a-b-c-d.res.dnssecready.net TXT IN
```

On a decent core (3+ GHz Intel Sandy Bridge), the dnsprobe_5 code is capable of sending these queries at up to approximately 40,000 queries/second, corresponding to a bandwidth utilization of approximately 30.4 million bits/second.

These queries arrive at their target destination and, if a DNS-capable entity is listening on port 53, may elicit DNS activity (if not prevented by local access control lists [ACLs]). In order to resolve the probe query, a DNS recursive resolver accesses authorities for "." (the DNS root) and "net" if necessary, and then the authority for "dnsssecready.net," which is hosted by two Shinkuro nameservers, ns1 and ns2.dnssecready.net, which run a standard Berkeley Internet Name Domain (BIND) 9 implementation.

The dnssecready.net domain is signed, and its nameservers log whether the accessing resolver requests DNS Public Key (DNSKEY) records. The querying recursive resolver then queries the authority for "res.dnssecready.net", since the address queried, "a-b-c-d.res.dnssecready.net" is unlikely to be in the resolver's cache. The custom-coded ns3.dnssecready.net nameserver inserts the IP address of the source IP (the recursive resolver's IP address) in the data of the TXT response to the query. As a result, when the DNS probe experiment finally receives the response from the DNS entity at the response address, that response contains the IP address of the host to which the query was issued (encoded in the query itself in the a-b-c-d part of the name), as well as the IP address of the host that queried ns3.res.dnssecready.net to resolve that name.

The dnsprobe_5 program asynchronously receives any responses on a separate (e.g., forked) sub-process.

Here are three examples of typical responses that have been received:

```
Header: 109.228.2.0:53 NOERROR, QR RD RA (0x8180), 1, 1, 0, 0
 Question: 109-228-002-000.res.dnssecready.net. IN TXT
 Answer: 109-228-002-000.res.dnssecready.net. 100 IN TXT "109.228.2.0"

Header: 184.73.0.0:53 NOERROR, QR RD RA (0x8180), 1, 1, 0, 0
 Question: 184-073-000-000.res.dnssecready.net. IN TXT
 Answer: 184-073-000-000.res.dnssecready.net. 100 IN TXT "64.233.168.85"

Header: 46.19.98.194:53 NOERROR, QR RD RA (0x8180), 1, 1, 1, 0
 Question: 185-008-000-000.res.dnssecready.net. IN TXT
 Answer: 185-008-000-000.res.dnssecready.net. 100 IN TXT "46.19.96.18"
 Authority: res.dnssecready.net. 683 IN NS ns3.dnssecready.net.
```

Three addresses are highlighted in each response. These are defined as follows: The first address and port are those from which the response was received, which we term the ***response address*** and the ***response port***. The second address is encoded with dashes in place of the normal dots, and is the address to which the original query was sent. We term this the ***probe address***. The last address is the address that queried the authoritative nameserver for res.dnssecready.net to resolve the TXT query. We term this the ***query address***.

In the first response, the probe, query and response addresses are all equal, and the response port is port 53. The first response is from what we consider to be a legitimate recursive resolver, as modeled by "A. Recursive Resolver" in Figure 5. The defining characteristic of this outcome is that the response address and probe address are the same, indicating that the single host is a DNS recursive resolver in its own right.

In the second response, the probe address equals the response address, the response port is port 53, and the query address differs from the probe or response addresses. The second response is from what we consider to be a legitimate forwarder/resolver, as represented in Figure 5 by "B. Forwarder/Resolver". In this model, the query is processed by a forwarder at the probe address, forwarding to a DNS recursive resolver at the query address. This is revealed by the two addresses mentioned above being different—the address originally queried by the probe is encoded in the "a-b-c-d" part of the name in the query, and "a-b-c-d.res.dnssecready.net." differs from the query address (that is, the address of the host that queried ns3.res.dnssecready.net to resolve the address of a-b-c-d.res.dnssecready.net). But the response is received by dnsprobe_5 on the query address, which is the same as the probe address. If the probe address and query address are the same, then it's likely that the machine we're probing is a resolver. If they are different, it may be because the machine we're probing may use a different address to send its query to our name server (e.g. a multi-homed resolver), or it may be because it sends its queries to another resolver before the query reaches our name server. In the latter case, the probe address likely corresponds to a "forwarder." There are three distinct regions in the observed data where the probe address and the query address differ:

- One probe address corresponds to 1 query address. This is likely to be the case of a multi-homed resolver.
- Several probe addresses correspond to 1 query address. We're not sure what this case is. It is for further research.
- Many probe addresses correspond to 1 query address. This is most likely to be case of multiple forwarders forwarding to a resolver.

The distinction between forwarders and resolvers becomes clearer when we see many responses that have the same query address but different probe addresses. For example, home routers on some networks will accept queries from outside the home, forward them to the ISP's resolver, and then return the result. In this situation, our scan of the IPv4 address space resulted in finding a substantial number of addresses, presumably corresponding to ISP customers' Internet Protocol (IP) addresses, all of which return results from the same resolver. It is further possible to check whether the resolver is directly accessible; if it is not, but the home routers are forwarding queries from outside the ISP's net, the resolvers are perhaps unintentionally accessible. We have not yet run through the data to prepare a comprehensive report on this and other phenomena.

In the third response, the probe address differs from the response address, meaning that whatever entity is responding to the probe is doing so from a different address than that to which the probe was directed. The response port is port 53, which is the correct DNS port. Note that the query address differs from both the probe and response addresses. We see that a query directed to 185.8.0.0 was forwarded to a resolver at 46.19.96.18 and the resulting response was returned to our dnsprobe_5 software by a host at yet another address—46.19.98.194, still from port 53. We assume this is some sort of unintended leakage from a legitimate DNS infrastructure.

This third type of response, shown in Figure 5 as "C. Unknown (Quake?)", is not easily related to any known DNS entity. There are several types of response that fall into this category; for example, we have observed DNS responses and also non-DNS responses from the host queried but from a different port, or indeed from a different host entirely (we match response to query using the a-b-c-d part of the query name). Sometimes the port corresponds to a port used by the popular *Quake* interactive game server, leading us to refer to this type of response using "Quake?" as shorthand.

dnsprobe_5 handles all of these types of responses similarly: by outputting a line in a text file that indicates the line is for a response to a Query 1 (the line starts with "R01,") and recording the IP address and port that sent the response, the value of the DNS flags extracted from the message, and the bits of the response message in their entirety.

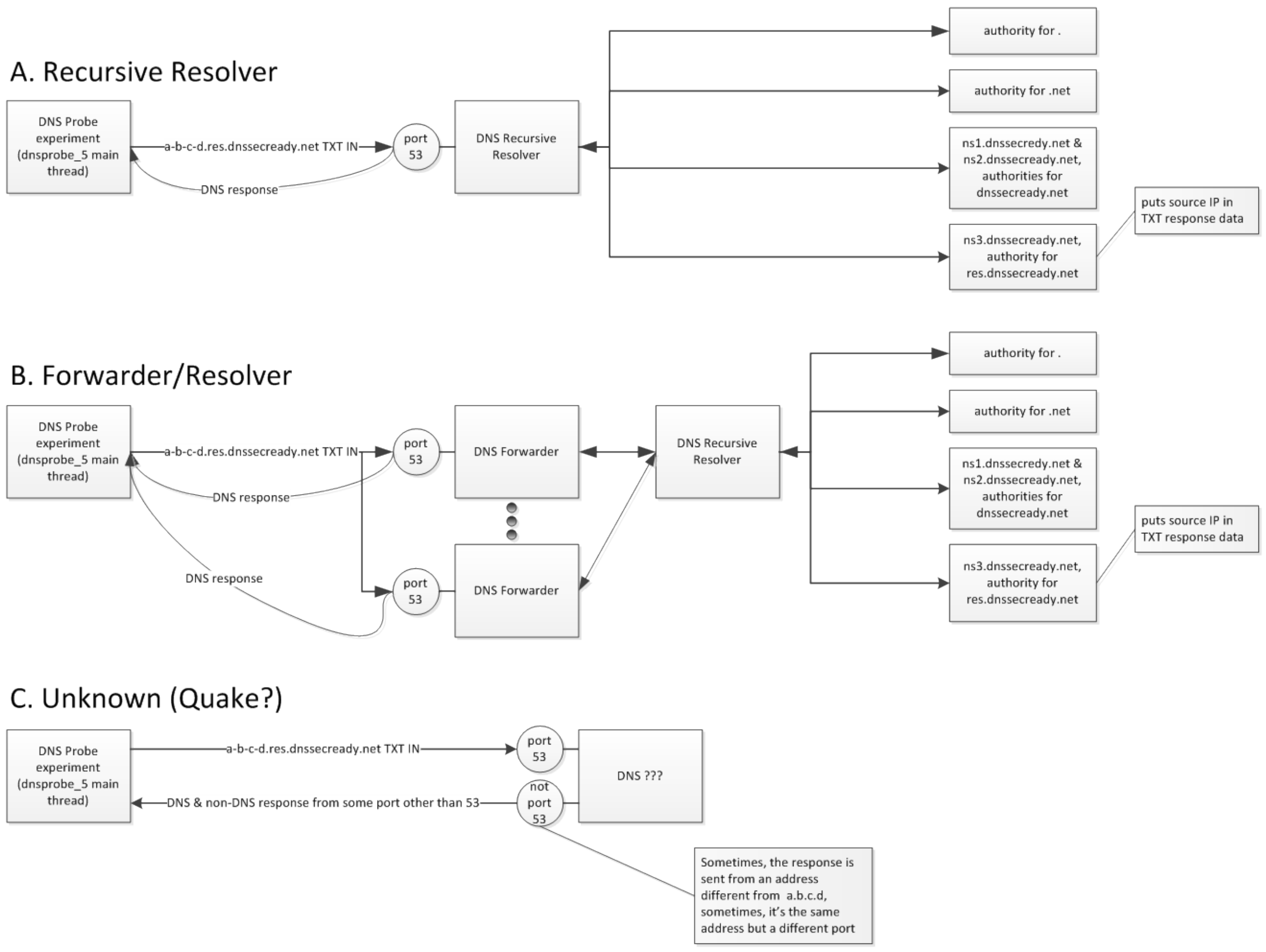

A. Recursive Resolver
DNS Probe experiment (dnsprobe_5 main thread)
a-b-c-d.res.dnssecready.net TXT IN
port 53
DNS response
DNS Recursive Resolver
authority for .
authority for .net
ns1.dnssecredy.net & ns2.dnssecready.net, authorities for dnssecready.net
ns3.dnssecready.net, authority for res.dnssecready.net
puts source IP in TXT response data
B. Forwarder/Resolver
DNS Probe experiment (dnsprobe_5 main thread)
a-b-c-d.res.dnssecready.net TXT IN
port 53
DNS response
DNS Forwarder
DNS Recursive Resolver
DNS response
port 53
DNS Forwarder
authority for .
authority for .net
ns1.dnssecredy.net & ns2.dnssecready.net, authorities for dnssecready.net
ns3.dnssecready.net, authority for res.dnssecready.net
puts source IP in TXT response data
C. Unknown (Quake?)
DNS Probe experiment (dnsprobe_5 main thread)
a-b-c-d.res.dnssecready.net TXT IN
port 53
DNS ???
DNS & non-DNS response from some port other than 53
not port 53
Sometimes, the response is sent from an address different from a.b.c.d, sometimes, it's the same address but a different port

Figure 4 – DNS models for responses to the a.b.c.d.dnssecready.net TXT query

Table 8 shows a list of all possible outcomes, all of which have been observed.

Table 8 – Observed outcomes to Query 1

Outcome	**From Response Port = 53**	**From Response Port != 53**
No response	No responding DNS entity	No responding DNS entity
Probe != Query != Response	Internal leakage?	Internal leakage?
(Probe == Query) != Response	Internal leakage?	Internal leakage?
Probe != (Query == Response)	Internal leakage?	Internal leakage?

(Probe == Response) != Query	Forwarder (Figure B)	Weird forwarder?
Probe == Query == Response	Recursive resolver (Figure A)	Weird recursive resolver?

4.3.4.3 dnsprobe_5 Query 2 – "version.bind TXT CH"

If dnsprobe_5 receives a response to Query 1 and if the conditions shown in Table 8 are met, then dnsprobe_5 sends a second (and third) query.

Table 9 – Conditions for sending Queries 2 and 3

Condition	Value
Response RCode	0 - NoError
Size of the response	More than 53 bytes
Number of queries in response	1 or more
IP address encoded in the query name (as a-b-c-d)	Equals address of host that sent the response
Port that sent the response	53

It is important to require that the response to a Query Type 1 be received from the same host as that to which the query was sent. (Type 1 queries are the only query for which we can check this condition.) This gives at least some assurance that the same will hold true for the second and third queries since we assume that those queries will be answered, from a procedural standpoint, in much the same way. We can then correlate the responses from these queries with information garnered from the responses to Type 1 queries.

The second query, if it were a dig command, would look like this:

```
dig @a.b.c.d version.bind TXT CH
```

DNS resolvers typically respond to such a query by reporting their version number, if this feature is enabled.

If dnsprobe_5 receives a response to this query, and if the response RCode is NoError (0) and the response has an answer, then dnsprobe_5 outputs a line in a text file that indicates the line is for a response to Query 2 (the line starts with "R02,"). The software then records the IP address and port that sent the response, the value of the DNS flags extracted from the message, and the bits of the response message in their entirety. In effect, this line has the same format as an "R01" type line.

4.3.4.4 dnsprobe_5 Query 3 – "org. SOA IN"

Following Query 2, if dnsprobe_5 receives a response to Query 1 and if the conditions shown in Table 8 are met, then dnsprobe_5 sends a third query, which if it were a dig command would look like this:

```
dig @a.b.c.d org. SOA IN
```

As above, if dnsprobe_5 receives a response to this query, and if the response RCode is NoError (0) and the response has an answer, then dnsprobe_5 outputs a line in a text file that indicates the line is for a response to a Query 3 (the line starts with "R03,") and records the IP address and port that sent the response, the values of the DNS flags extracted from the message, and the bits of the response message in their entirety. So again, effectively, the line has the same format as an "R01" type line.

4.3.4.5 dnsprobe_5 output files

Whenever a suitable response is received, dnsprobe_5 writes lines to an output file as described above. In addition, dnsprobe_5 periodically closes the current output file and opens a new output file, as described below.

When dnsprobe_5 opens an output file, it gives the file a name of the form:

```
output_yymmdd_hhmmss.wrk
```

Whenever dnsprobe_5 writes to the current output file, it checks how many lines it has written to the file; if it has written more than a given number (as established by an input parameter to the application), then dnsprobe_5 closes the output file, renames it to a name of the form:

```
output_yymmdd_hhmmss.log
```

and opens a new output file with a name in the first form shown above.

This allows a subsequent program in the pipeline to scan for files with the .log extension and process them in sorted order.

4.3.4.6 The run_grader_process.py Application

The next application in the pipeline is run_grader_process.py. This is python code that uses the twisted framework to allow the application to run without blocking while awaiting event completions. The primary purpose of run_grader_process.py is to run multiple copies of the "DNSSEC Resolver Check" application concurrently and coherently record the results. DNSSEC Resolver Check is actually a Java application that synchronously issues multiple DNS queries to assess a resolver.

In terms of the overall continuous batch pipeline operation, run_grader_process.py scans a folder for files named .log (which are output by dnsprobe_5 as described above), sorts the filenames, processes the first file in sort order, renames the processed file into a different, archive folder so the file is not reprocessed, and repeats. If there are no files, run_grader_process.py delays for a short time and then repeats the scan.

To process a log file, run_grader_process.py reads each line in the file. There are three kinds of lines in a log file—one for each type of query response from dnsprobe_5. The run_grader_process.py application handles each type of line as follows.

4.3.4.6.1 Query Type 1 Responses

For lines that contain a response to a Query Type 1, run_grader_process.py compares the probe address with the query address. If the two are equal, we assume this is the address of a DNS recursive resolver and run_grader_process.py caches the query address in a "resolver cache" dictionary, with the intent of recording the resolver's DNSSEC Resolver Check score when it becomes available.

If the probe address is different than the query address, but the response address equals the probe address and the response port is port 53, then run_grader_process.py caches both the probe and query addresses in a "forwarder cache." In this cache, the probe address points to the query address, so that if a subsequent R03 line is read with the same probe address, and if there happens to be a Behavior descriptor already available for that resolver at the response address, it will be used instead of reassessing the resolver for the probe address over and over again for each forwarder probed.

We ignore all other outcomes, since they cannot be from a legitimate open recursive resolver.

4.3.4.6.2 Query Type 2 Responses

The response to a Type 2 query contains, if anything, the claimed version of the resolver. If a Type 1 query determines that the probe address is a recursive resolver, run_grader_process.py caches the version against the probe address. If a Type 1 query determined that the probe address was the address of a forwarder, run_grader_process.py caches the version against the query address from the forwarder cache mentioned above. The cached version is used subsequently when the Behavior descriptor (and any other notation) is output for the probe address.

4.3.4.6.3 Query Type 3 Responses

The response to a Type 3 query indicates that there is some semi-persistent DNS-like behavior behind the probed address. If run_grader_process.py receives a response to a Type 3 query, it checks the cached forwarders to see if the probed address is that of a forwarder and if the forwarder's resolver has already been assigned a descriptor. If so, it outputs the previously determined descriptor (and version, if cached), with a grading time set to 0 to the grading output. Otherwise, it outputs a line to a grading (.grd) file that contains the IP address to assess, and an encoded message for the grader to send to the grade collector and continue.

4.3.4.7 Running the DNSSEC Resolver Check Application

When an input .log file has been processed in the above described fashion, fcc_grader_process.py adds the file to its caches of forwarders and versions, and a .grd file that contains the IP addresses of the resolvers to assign descriptors to. run_grader_process.py then processes the lines in the .grd file, starting an DNSSEC Resolver Check java application to assign descriptors for each address until *N* Java applications are running. *N* is set by an input parameter to run_grader_process.py, and is typically 200–300 processes. Once that number of processes is running, and as long as there is more input, run_grader_process.py waits for each grader to finish, records the grading result, and starts another grader, keeping *N* graders running concurrently. Processing continues in this fashion until all input has been consumed.

4.3.4.8 DNSSEC Resolver Check Output

As each DNSSEC Resolver Check application finishes, the probed IP address is recorded, along with the resolver IP address (if different than the probed address), the version of the resolver, if any was reported by the appropriate Type 2 query, its Behavior and any other descriptor, and the time it took to assess the resolver. The descriptor is also cached against the resolver IP address (or the probed address if there is no resolver cached).

4.3.4.9 Preliminary Results

The following breaks down the results from Shinkuro's survey thus far.

1	Total Size of the IPv4 Address Space	4,294,967,296
2	Number of Addresses Probed	3,421,239,040
3	Number of Queries Received at Name Server	27,280,190
4	Number of Answered Queries Received	17,082,533
5	Number of Answered Queries Not Received	10,197,657
6	Number of Probes Answered Without the Name Server	8,554,194

Table 10: Summary of probes and responses

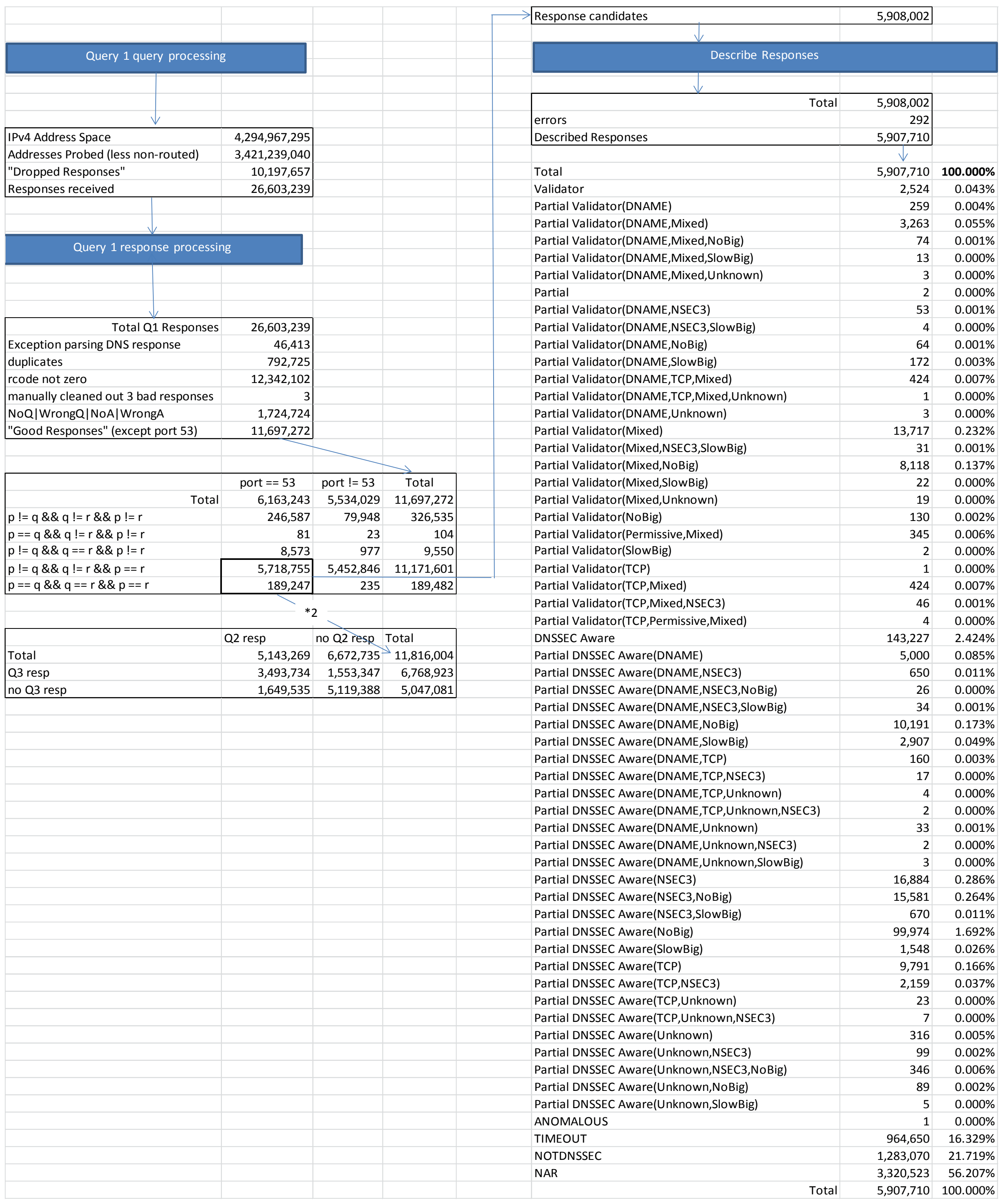
Line 3 shows the number of probes that resulted in queries (~27.2M) arriving at Shinkuro's name server, which responded to all these queries. Line 4 shows the number of queries (>17M) that were returned from the probe address with the correct answer. Line 5 shows the remainder (>10M) that were not returned from the probe address despite being sought from Shinkuro's name server. This is a very high figure, which is prompting us to recheck our results and methodology. This process should be repeated and checked more closely in the future.

Note that Shinkuro's survey software tried each address once and did not retry that address if it received no response. We think the test results might be different if we were to retry each address three times before giving up on it.

Line 6 shows the number of queries (>8M) that resulted in some form of locally generated response not correlated with any query to Shinkuro's name server. 99.9% of these were SERVFAIL.

In addition to retrying the survey for those addresses represented in line 5, running the DNSSEC Resolver Check program, and analyzing the pattern of probe addresses versus query addresses, it would also be possible to query the probe addresses to see the responses to the question “What software version are you running?” and also run the fingerprint program (fpdns), to attempt to learn the same information via a series of tests.

It is also possible to correlate the addresses of resolvers with ISPs by matching the query address to IP address blocks assigned to each ISP.



IPv4 Address Space	4,294,967,295
Addresses Probed (less non-routed)	3,421,239,040
"Dropped Responses"	10,197,657
Responses received	26,603,239

Total Q1 Responses	26,603,239
Exception parsing DNS response	46,413
duplicates	792,725
rcode not zero	12,342,102
manually cleaned out 3 bad responses	3
NoQ|WrongQ|NoA|WrongA	1,724,724
"Good Responses" (except port 53)	11,697,272

	port == 53	port != 53	Total
Total	6,163,243	5,534,029	11,697,272
p != q && q != r && p != r	246,587	79,948	326,535
p == q && q != r && p != r	81	23	104
p != q && q == r && p != r	8,573	977	9,550
p != q && q != r && p == r	5,718,755	5,452,846	11,171,601
p == q && q == r && p == r	189,247	235	189,482

	Q2 resp	no Q2 resp	Total
Total	5,143,269	6,672,735	11,816,004
Q3 resp	3,493,734	1,553,347	6,768,923
no Q3 resp	1,649,535	5,119,388	5,047,081

Response candidates	5,908,002

Total	5,908,002
errors	292
Described Responses	5,907,710

Total	5,907,710	**100.000%**
Validator	2,524	0.043%
Partial Validator(DNAME)	259	0.004%
Partial Validator(DNAME,Mixed)	3,263	0.055%
Partial Validator(DNAME,Mixed,NoBig)	74	0.001%
Partial Validator(DNAME,Mixed,SlowBig)	13	0.000%
Partial Validator(DNAME,Mixed,Unknown)	3	0.000%
Partial	2	0.000%
Partial Validator(DNAME,NSEC3)	53	0.001%
Partial Validator(DNAME,NSEC3,SlowBig)	4	0.000%
Partial Validator(DNAME,NoBig)	64	0.001%
Partial Validator(DNAME,SlowBig)	172	0.003%
Partial Validator(DNAME,TCP,Mixed)	424	0.007%
Partial Validator(DNAME,TCP,Mixed,Unknown)	1	0.000%
Partial Validator(DNAME,Unknown)	3	0.000%
Partial Validator(Mixed)	13,717	0.232%
Partial Validator(Mixed,NSEC3,SlowBig)	31	0.001%
Partial Validator(Mixed,NoBig)	8,118	0.137%
Partial Validator(Mixed,SlowBig)	22	0.000%
Partial Validator(Mixed,Unknown)	19	0.000%
Partial Validator(NoBig)	130	0.002%
Partial Validator(Permissive,Mixed)	345	0.006%
Partial Validator(SlowBig)	2	0.000%
Partial Validator(TCP)	1	0.000%
Partial Validator(TCP,Mixed)	424	0.007%
Partial Validator(TCP,Mixed,NSEC3)	46	0.001%
Partial Validator(TCP,Permissive,Mixed)	4	0.000%
DNSSEC Aware	143,227	2.424%
Partial DNSSEC Aware(DNAME)	5,000	0.085%
Partial DNSSEC Aware(DNAME,NSEC3)	650	0.011%
Partial DNSSEC Aware(DNAME,NSEC3,NoBig)	26	0.000%
Partial DNSSEC Aware(DNAME,NSEC3,SlowBig)	34	0.001%
Partial DNSSEC Aware(DNAME,NoBig)	10,191	0.173%
Partial DNSSEC Aware(DNAME,SlowBig)	2,907	0.049%
Partial DNSSEC Aware(DNAME,TCP)	160	0.003%
Partial DNSSEC Aware(DNAME,TCP,NSEC3)	17	0.000%
Partial DNSSEC Aware(DNAME,TCP,Unknown)	4	0.000%
Partial DNSSEC Aware(DNAME,TCP,Unknown,NSEC3)	2	0.000%
Partial DNSSEC Aware(DNAME,Unknown)	33	0.001%
Partial DNSSEC Aware(DNAME,Unknown,NSEC3)	2	0.000%
Partial DNSSEC Aware(DNAME,Unknown,SlowBig)	3	0.000%
Partial DNSSEC Aware(NSEC3)	16,884	0.286%
Partial DNSSEC Aware(NSEC3,NoBig)	15,581	0.264%
Partial DNSSEC Aware(NSEC3,SlowBig)	670	0.011%
Partial DNSSEC Aware(NoBig)	99,974	1.692%
Partial DNSSEC Aware(SlowBig)	1,548	0.026%
Partial DNSSEC Aware(TCP)	9,791	0.166%
Partial DNSSEC Aware(TCP,NSEC3)	2,159	0.037%
Partial DNSSEC Aware(TCP,Unknown)	23	0.000%
Partial DNSSEC Aware(TCP,Unknown,NSEC3)	7	0.000%
Partial DNSSEC Aware(Unknown)	316	0.005%
Partial DNSSEC Aware(Unknown,NSEC3)	99	0.002%
Partial DNSSEC Aware(Unknown,NSEC3,NoBig)	346	0.006%
Partial DNSSEC Aware(Unknown,NoBig)	89	0.002%
Partial DNSSEC Aware(Unknown,SlowBig)	5	0.000%
ANOMALOUS	1	0.000%
TIMEOUT	964,650	16.329%
NOTDNSSEC	1,283,070	21.719%
NAR	3,320,523	56.207%
Total	5,907,710	100.000%

Figure 5 – Results summary

5 Analysis, Findings and Recommendations

5.1 Analysis

The measurement of DNSSEC validation in ISPs is more complex than Working Group 5 initially thought, given the following factors:

- ISPs may have several internal networks, some of which may offer differing levels of DNS service.
- Some ISPs may have taken steps toward DNSSEC validation but may not have proceeded to full validation.
- The initial scheme outlined in this Working Group's March report has been improved to account for certain subtleties of DNSSEC validation, such as the setting of the AD bit or whether an ISP permits TCP queries.

The survey and description process have been fairly effective in discovering and characterizing the level of DNSSEC support across the Internet. The description system described in the first report has evolved into a more nuanced and less judgmental description process. The description process still distinguishes among three main levels of DNSSEC support for ISP resolvers:

- *Validator*. A Validator requests signed answers and checks those answers to make sure the signatures are correct.
- *DNSSEC Aware Resolver*. A DNSSEC Aware Resolver correctly returns signed answers and DNSSEC-related information such as a DS record, but does not check the signatures on the answers. If the user’s end system or another intermediate system such as the resolver built into the enterprise’s or customer’s edge router performs the validation, it will be able to fetch the necessary data via the ISP’s resolver.
- *NA*. An older resolver or one that does not support DNSSEC, does not check signatures and does not make it possible for the user’s end system to check signatures. If the user’s end system needs to check signatures, it will have to fetch the necessary records from some other resolver.

Within the first two categories, some Validators and some DNSSEC Aware Resolvers do not implement all of the features associated with full functionality. We developed a series of subdescriptors to annotate these particular descriptions. For example, “Partial Validator (DNAME, TCP)” describes a validator that does not correctly process DNAME records and does not permit TCP connections.

5.2 Findings

Working Group 5 devised a program in which different types of testing was performed by various parties. These tests helped determine ISPs' level of validation using slightly different methodologies and collection methods, helping to account for the above-mentioned complexities in validation. The Working Group's findings are as follows.

- In both the survey conducted with the SamKnows probes and with the Shinkuro full IPv4 space survey, many resolvers are DNSSEC Aware. Of these, a large number have limitations in specific areas, e.g. support for large packets, DNAME, etc.

- Similarly, in both surveys, a small but significant number of resolvers carry out validation. Together with the statistics on DNSSEC Aware resolvers, there is a substantial level of support for DNSSEC operation.
- The significant level of incomplete implementations suggests there is room for maturation in operational deployments. For example, many Validators and many DNSSEC Aware Resolvers do not support DNAME records. This is not causing a large number of problems at present because DNAME is not widely used, but its usage level is likely to rise. Accordingly, the limitations in existing deployments will become problematic.
- Substantial levels of support for DNSSEC operation exist in resolvers today, but there are still significant numbers of resolvers that do not support DNSSEC operation.
- As a side effect of the survey process, the Working Group discovered that many edge routers appear to be accepting DNS queries from anywhere on the Internet instead of just within their intended premises. This may have the effect of providing access to resolvers that are intended to serve only the addresses within the ISP’s network.

5.3 Recommendations

Working Group 5 forwards and still supports its recommendations from its initial report to CSRIC in March 2012. In that document, we asked that the FCC urge:

- ISPs to implement their DNS recursive nameservers so that they are at a minimum DNSSEC-aware, as soon as possible.
- Key industry segments, such as banking, credit cards, e-commerce, healthcare and other businesses, to sign their respective domain names. We encourage FCC to ask industry-leading companies in key sectors to commit to doing so, in order to create competitive pressure for others to follow. These industries may be prioritized based on the prevalence of threats to each one, which would mean focusing on financially related sites first, followed by other sites that hold private user data.
- Software developers such as web-browser developers to study how and when to incorporate DNSSEC validation functions into their software. For example, a browser developer might create a visual indicator for whether or not DNSSEC is in use, or perhaps only a visual warning if DNSSEC validation fails.

In addition, Working Group 5 also recommends that the FCC encourage ISP participation in the testing scheme outlined in this document, and the continued deployment of DNSSEC by ISPs and other members of the Internet ecosystem.

- The survey and description process reported here is just the beginning of a process that FCC should urge continuation of in the future. The Working Group believes there is room for refinement of this process, which will be valuable for measuring the uptake of DNSSEC support over the next few years, and that it should be expanded and continued. In addition, FCC should encourage ISPs to use this process to improve their own DNSSEC deployments.
- There is controversy as to whether DNSSEC exacerbates amplified DDoS attacks. FCC should encourage concerned parties to document and examine these attacks along with possible defensive solutions.

6 List of Acronyms

ACL	Access control list
AD	Authenticated Data
APT	Advanced persistent threat
BIND	Berkeley Internet Name Domain
CNAME	Canonical Name
CSRIC	Communications Security, Reliability and Interoperability Council
DDD	DNSSEC Deployment Descriptor
DHCP	Dynamic Host Configuration Protocol
DNAME	Delegation Name
DNS	Domain Name System
DNSKEY	DNS Public Key
DNSSEC	Domain Name System Security Extensions
DS	Delegation Signer
EDNS	Extension mechanisms for DNS
EDNS0	Extension mechanism for DNS 0 [zero]
FCC	Federal Communications Commission
FSA	Finite State Automaton
IP	Internet Protocol
IPv4	Internet Protocol version 4
ISP	Internet Service Provider
NAR	Not a Resolver
NDR	Non-DNSSEC Capable DNS Resolver
NSEC	Next Secure (data format)
NSEC3	Next Secure 3 (data format)
OPT	OPTION pseudo-RR
RFC	Request for Comment
RR	Resource Record
SOA	Start of Authority
SOHO	Small office/home office
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
WG5	Working Group 5