Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 25, 2013

A. Sullivan
Dyn, Inc.

October 22, 2012

Asserting DNS Administrative Boundaries Within DNS Zones
draft-sullivan-domain-origin-assert-02

Abstract

Some clients on the Internet make inferences about the administrative relationships among servers on the Internet based on the domain names of those servers. Perhaps unfortunately, it is not currently possible to detect the real administrative boundaries in the DNS, and therefore such inferences can go wrong in several ways. Mitigation strategies deployed so far will not scale. The solution to this is to provide a way to make an explicit assertion about the relationships between different domain names and perhaps the services provided at them.

Status of this Memo

Copyright Notice

Table of Contents

1. Motivation 3
2. Background, terminology, and organization of this memo 5
3. Overview of mechanism 5
  3.1. Identifying names as the scope for policy authority . . . 6
  3.2. Identifying names, schemes, and ports as the scope for policy authority 6
4. The SOPA Resource Record 7
  4.1. Thr SOPA Resource Record only for names 7
  4.2. Thr SOPA Resource Record with ports and schemes 8
5. Use of the SOPA RRTYPE 9
  5.1. Special target labels 10
    5.1.1. The root target 10
    5.1.2. Wildcards in targets 10
  5.2. What can be done with an SOPA RR 11
6. An example case 11
  6.1. Examples of using the SOPA record for determining boundaries 12
    6.1.1. One delegation, eight administrative realms, no root target 13
    6.1.2. One delegation, eight administrative realms, root targets 13
    6.1.3. Two delegations, seven or eight policy realms, root targets 14
7. Limitations of the approach and other considerations 15
  7.1. Handling truncation 16
8. Security Considerations 16
9. IANA Considerations 16
10. Acknowledgements 16
11. References 17
  11.1. Normative References 17
  11.2. Informative References 17
Appendix A. Discussion Venue 18
Appendix B. Change History 18
Author's Address 19

1. Motivation

Many network resources are accessed primarily by name. DNS names make up the bulk of those names. As a result, DNS names have become fundamental elements in building security policies and user agent behaviour. For example, domain names are used in attempts to determine the scope for data sharing of HTTP state management cookies [RFC6265] . The idea is to foil the attempts of attackers in (for example) attackersite.co.tld from setting cookies for everyone in co.tld.

Another example policy is a user interface convention that purports to display the "real" domain name differently from other parts of the fully-qualified domain name, in an effort to decrease the success of phishing attacks. In this strategy, for instance, a domain name like "www.bank.example.com.attackersite.tld" is formatted to highlight that the name is inside "attackersite.tld", in the hope of thereby reducing the user's impression of visiting "www.bank.example.com".

Issuers of X.509 certificates make judgements about administrative boundaries around domains when issuing the certificates. For some discussion of the relationship between DNS names and X.509 certificates, see [RFC6125].

One way to build a reasonable policy is to treat each different domain name distinctly. Under this approach, foo.example.org, bar.example.org, and baz.example.org are all just different domains. Such an approach can be awkward, however, when (as is often the case) the real administrative boundary is a shared one (in this example, example.org). Therefore, clients have attempted to make more sophisticated policies.

Historically, policies were sometimes based on the DNS tree. Early policies made a firm distinction between top-level domains and everything else; but this was too naive, and later attempts were based on inferences from the DNS names themselves. That did not work well, because there is no way in the DNS to discover the boundaries of administrative control around domain names.

Some have attempted to use the boundary of zone cuts (i.e. the location of the zone's apex, which is at the SOA record; see [RFC1034] and [RFC1035]). Unfortunately, that boundary is neither necessary nor sufficient for these purposes: it is possible for a large site to have many, administratively distinct subdomain-named sites without inserting an SOA record, and it is also possible that an administrative entity (like a company) might divide its domain up into different zones for administrative reasons unrelated to the purposes of sites named in that domain. It was also, prior to the

advent of DNSSEC, difficult to find zone cuts. Regardless, the location of a zone cut is an administrative matter to do with the operation of the DNS itself, and not useful for determining relationships among services offered at names in the DNS.

What appears to be needed is a mechanism to determine administrative boundaries in the DNS. That is, given services at two domain names, one needs to be able to answer whether the first and the second are under the same administrative control and same administrative policies. We may call this state of affairs "lying within the same policy realm". We may suppose that, if this information were to be available, it would be possible to make useful decisions based on the information.

A particularly important distinction for security purposes is the one between names that are mostly used to contain other domains, as compared to those that are mostly used to operate services. The former are often "delegation-centric" domains, delegating parts of their name space to others, and are frequently called "public suffix" domains or "effective TLDs". The term "public suffix" comes from a site, publicsuffix.org, that publishes a list of domains (henceforth, the "public suffix list") that are used to contain other domains. Not all, but most, delegation-centric domains are public suffix domains; and not all public suffix domains need to do DNS delegation, although most of them do. The reason for the public suffix list is to make the distinction between names that must never be treated as being in the same policy realm as another, and those that might be so treated. For instance, if "com" is on the public suffix list, that means that "example.com" lies in a policy realm distinct from that of com.

Unfortunately, the public suffix list has several inherent limitations. To begin with, it is a list that is separately maintained from the list of DNS delegations. As a result, the data in the public suffix list can diverge from the actual use of the DNS. Second, because its semantics are not the same as those of the DNS, it does not capture unusual features of the DNS that are a consequence of its structure (see [RFC1034] for background on that structure). Third, as the size of the root zone grows, keeping the list both accurate and synchronized with the expanding services will become difficult and unreliable. Perhaps most importantly, it puts the power of assertion about the operational policies of a domain outside the control of the operators of that domain, and in the control of a third party possibly unrelated to those operators.

There have been suggestions for improvements of the public suffix list, most notably in [I-D.pettersen-subtld-structure]. It is unclear the extent to which those improvements would help, because

they represent improvements on the fundamental mechanism of keeping metadata about the DNS tree apart from the DNS tree itself.

2. Background, terminology, and organization of this memo

The reader is assumed to be familiar with the DNS ([RFC1034] [RFC1035]) and DNSSEC ([RFC4033] [RFC4034] [RFC4035] [RFC5155]).

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 RFC 2119 [RFC2119].

Section 3 describes the mechanism in general terms, outlining two different possible approaches and outlining the compromises in each case. Definitions of the new RRTYPE is in Section 4, with two different definitions to allow for the two different approaches. There is some general discussion of the use of the RRTYPE is in Section 5. Then, Section 6 offers an example portion of a DNS tree that can be used to understand the ways in which the mechanism can be used to draw inferences about administrative relationships. Section 7 notes some limitations of the mechanism. Section 8 outlines how the mechanism might be used securely.

3. Overview of mechanism

This memo presents a way to assert a common administrative realm by placing a resource record (RR) at DNS names within the policy realm. The mechanism requires a new resource record type (RRTYPE) to enable these assertions, called SOPA (for Start Of Policy Authority, echoing the Start Of Authority or SOA record). While there are reported difficulties in deploying new RRTYPEs, the only RRTYPE that could be used to express all the necessary variables is the TXT record, and it is unsuitable because it can also be used for other purposes. The use of this mechanism does not require "underscore labels" to scope the interpretation of the RR, in order to make it possible to use the mechanism where the underscore label convention is already in use. The SOPA RRTYPE is class-independent.

While many policies of the sort discussed in Section 1 appear to be based on domain names, they are actually only partly based on them. Usually, there are implicit rules that come from the destination port [RFC6335] or scheme [RFC4395] (or both) in use.

It is possible to make those assumptions explicit, but at the cost of expressing in the resulting record a tighter relationship between the DNS and the services offered at DNS names. There are arguments to be

made for each approach. Section 3.1 and Section 3.2 explore the two different approaches; this memo does not make a decision about which strategy to adopt. If there are future developments of this memo, one strategy will be selected.

It is worth observing that a policy realm relationship ought to be symmetric: if example.com is in the same policy realm as example.net, then example.net should be (it would seem) in the same policy realm as example.com. In principle, then, if a SOPA RR at example.com provides a target at example.net, there should be a complementary SOPA RR at example.net with a target of example.com. Because of the distributed nature of the DNS, and because the other administrative divisions need not follow the policy realms, the only way to know whether two names are in the same policy realm is to query at each name, and to correlate the responses.

3.1. Identifying names as the scope for policy authority

The first approach provides, as the RDATA of a SOPA RR, a target name that lies in the same policy realm as the owner name in the RR. For convenience, in what follows this is sometimes called this the "names-only" strategy. Such an approach is an assertion on the part of the authoritative DNS server for the owner name in question that there is a policy relationship between that owner name and the target name. If a single name lies in the same policy realm as several other target names, an additional RR is necessary for each such relationship, with one exception. It is not uncommon for two different names to have policy relationships among all the children beneath them. Using the SOPA RR, it is possible to specify that the target is all the names beneath a name, using a wildcard target.

This approach follows the traditional way that relationships are expressed in the DNS, which is historically mostly between different names. Aliases, for instance, redirect names or trees, and not names or trees for specific RRTYPEs. It has the disadvantage, however, that it may not provide enough information about the relationship between two names to make all the inferences one might need about the relationship. For instance, the relationship between two hosts might depend on the protocol, port, and scheme in use: two domains might share policy for the purposes of connections on port 80, but for no other connections. [[anchor2: Could this be capured by using SRV or NAPTR records on both sides of the policy relationship? Too slow? --ajs@anvilwalrusden.com]]

3.2. Identifying names, schemes, and ports as the scope for policy authority

The second approach provides, as the RDATA of a SOPA RR, a target

name that lies in the same policy realm as the owner name in the RR, but can identify the relationship as pertaining only to certain ports and schemes. In what follows, for convenience this is sometimes called the "port-and-scheme" stragegy. It provides a way to cover ranges of ports with a single resource record, and to cover all schemes and ports with a single resource record. It does not, however, permit arbitrary combinations of destination point and schemes without using more than one RR.

This approach offers a mechanism to express relationships between services at a domain name instead of merely between names. As a disadvantage, however, it seems to step outside the usual scope of the DNS, which concerns itself with names and not services offered at those names. It might be argued that some RRTYPEs (notably SRV [RFC2782] and NAPTR [RFC3403]) do relate to services; but in those cases, it is an expression of services available at an already-named host. It would be a significant innovation, perhaps in a bad direction, to attempt to express these relationships in a single RR. Since the names are under different administration, also, it is entirely possible that the operators of the two domains do not agree on the port ranges and schemas to be supported, creating an intractable comparison problem for a client.

4. The SOPA Resource Record

Because of the two approaches outlined in Section 3.1 and Section 3.2, this section provides two different outlines of the SOPA resource record. This arrangement is pending the decision about which strategy to adopt, at which time the discussion below will be reduced to reflect that decision.

4.1. Thr SOPA Resource Record only for names

In this case, the SOPA resource record, type number [TBD1], includes the following fields:

Name: The owner name of the RR.

TTL: The time to live for the RR.

Class: The CLASS for the RR. As of this writing, on the contemporary Internet this is almost always "IN", but the SOPA RR is class-independent.

SOPA: The RRTYPE field.

Target: A DNS name relative to the root that is in the same policy realm as the SOPA RR's owner name. The name MUST be a DNS name according to the rules in [RFC1034] and [RFC1035], except that the left-most label of the target MAY be the wildcard character ("*"). In addition, the target may be "."; in that case, the RR asserts that there are no other names in the same policy realm. For

further discussion, see Section 5.1. The target MUST NOT be an alias [RFC2181], such as the owner name of a CNAME [RFC1034], DNAME [RFC6672], or other similar such resource records.

The SOPA RRTYPE's wire format is as follows:

[[anchor3: To follow if this idea (and this version of it) turns out worth pursuing. It can be derived from above, however. --ajs@anvilwalrusden.com]]

4.2. Thr SOPA Resource Record with ports and schemes

In this case, the SOPA resource record, type number [TBD1], includes the following fields:

Name: The owner name of the RR.

TTL: The time to live for the RR.

Class: The CLASS for the RR. As of this writing, on the contemporary Internet this is almost always "IN", but the SOPA RR is class-independent.

SOPA: The RRTYPE field.

Starting port: The port number that begins the range to which this SOPA RR applies. This is a 16 bit unsigned integer in network byte order. The range is 0-65535.

Ending port: The port number that ends the range for which this SOPA RR applies. This is a 16 bit unsigned integer in network byte order. The range is 0-65535.

Scheme: The scheme to which the SOPA RR applies. The scheme SHOULD be listed in the IANA Uniform Resource Identifier (URI) Schemes registry [1]. [[anchor4: This is "SHOULD" right now, but I think it should be "MUST". I can't think of a reason why not to make it MUST. --ajs@anvilwalrusden.com]] Alternatively, the field may contain the special value "*", in which case the SOPA RR applies to all schemes, with a limitation: some clients use certain schemes only for internal operations, and regardless of whether those schemes are included in an SOPA RR, they MAY be ignored.

Target: A DNS name relative to the root that is in the same policy realm as the SOPA RR's owner name. The name MUST be a DNS name according to the rules in [RFC1034] and [RFC1035], except that the left-most label of the target MAY be the wildcard character ("*"). In addition, the target may be "."; in that case, the RR asserts that there are no other names in the same policy realm. For further discussion, see Section 5.1. The target MUST NOT be an alias [RFC2181], such as the owner name of a CNAME [RFC1034], DNAME [RFC6672], or other similar such resource records.

The SOPA RRTYPE's wire format is as follows:

[[anchor5: To follow if this idea (and this version of it) turns out

worth pursuing. It can be derived from above, however.
--ajs@anvilwalrusden.com]]

5. Use of the SOPA RRTYPE

SOPA RRs may have, in effect, three different functions. The simplest is to make an assertion that two DNS names are in the same policy realm. Under the port-and-scheme strategy, if the Starting Port is 0, the Ending Port is 65535, the Scheme is "*", and the Target is anything other than ".", then the SOPA record makes a claim that the owner name and the target name are in the same policy realm in every case. This is also the claim whenever the names-only stragey is in use, and the RDATA has a target other than ".".

The second function, available only under the port-and-scheme strategy, is to make an assertion that two DNS names are in the same policy realm, but only for some subset of ports or schemes or both. There is a 1:1 port mapping presumed in the way the SOPA RR is structured, such that it is not possible to say that port N at the owner name is related to port M at the target name. This is an expedient for simplicity. For the same reasons of simplicity, the SOPA RR permits linking all schemes between names, or else one scheme; a given RR does not permit listing more than one scheme without using the wildcard selector.

The third function is to make an assertion that no other name lies in the same policy realm as the owner name. If there are names beneath that owner name, this is a way for a DNS operator to assert that the owner name is a public suffix. For more details, see Section 5.1.

There could be more than one SOPA resource record per owner name in a response. Each domain name in the RDATA is treated as a part of the same policy realm as the owner name in the original QNAME (subject to the qualifications of scheme and port contained in the SOPA RR in the port-and-scheme strategy). The QNAME from the query might not be the owner name of the SOPA RR: if the original QNAME was an alias, then the SOPA owner name will be different.

There are three possible responses to a query for the SOPA RRTYPE at an owner name that are relevant to determining the policy realm. The first is Name Error (RCODE=3, also known as NXDOMAIN). In this case, the owner name itself does not exist, and no further processing is needed.

The second is a No Data response [RFC2308] of any type. The No Data response means that the owner name in the QNAME does not recognize any other name as part of a common policy realm.

The final is a response with one or more SOPA resource records in the Answer section. Each SOPA resource record asserts a relationship between the owner name and the target name, according to the functions of the SOPA RRTYPE outlined above.

Any other response is no different from any other sort of response from the DNS, and is not in itself meaningful for determining the policy realm of a name (though it might be meaningful for finding the SOPA record).

5.1. Special target labels

5.1.1. The root target

An SOPA resource record with the single character "." (called the "root target") in the RDATA is a positive assertion that no other domain name falls inside the policy realm of the owner name. The record has a special use: it may be used to bootstrap operation. A client that has encountered the root target may remember the existence of the root target even after the expiry of the TTL on the RRset, until such time as a new query for the owner name may be made successfully. This rule permits implementations to cache positive statements of administrative isolation during disconnected periods, thereby starting a subsequent session with the values of prior affirmed policy. Apart from this bootstrapping use, and the ability of such an RR to have a TTL independent of the negative TTL value for the zone, this mechanism is semantically equivalent to a No Data response.

It would be absurd for the root target for any given schema to exist with any other SOPA resource record at that owner name. An authoritative name server MAY refuse to serve a zone containing such an inconsistency, MAY refuse to load a zone containing such an inconsistency, or MAY suppress every SOPA RR at an owner name and schema except that containing the root target. The name server side of a recursive resolver MAY discard every SOPA RR at an owner name except that containing the root target. Conforming servers MUST NOT serve the root target and any other SOPA RR at the same owner name. Clients receiving a SOPA RRset that includes the root target MUST accept that RR, and discard any other RR in the RRset.

5.1.2. Wildcards in targets

The special character "*" in the Target field is used to match any label, according to the wildcard label rules in section 4.3.3 of [RFC1034]. Note that, because of the way wildcards work in the DNS, is it not possible to place a restriction to the left of a wildcard; so, for instance, example.*.example.com does not work. The effect is

maintained in this memo. An authoritative name server MUST NOT serve an SOPA RR with erroneous wildcards, and clients receiving such an SOPA RR MUST discard the RR. If the discarded RR is the last RR in the answer section of the response, then the response is treated as a No Data response.

5.2. What can be done with an SOPA RR

Use of an SOPA RR enables a site administrator to assert or deny relationships between names. By the same token, it permits a a consuming client to detect these assertions and denials.

Some of these relationships are currently impossible to indicate in the DNS. For example, IDN character variants (see [RFC4290]) result in situations where multiple labels are sometimes intended to be treated as though they are the same. Without a mechanism for binding the names together even loosely, such a goal cannot be achieved.

The use of SOPA RRs could either replace the public suffix list or (more likely due to some limitations -- see Section 7) simplify and automate the management of the public suffix list. A client could use the responses to SOPA queries to refine its determinations about http cookie Domain attributes. In the absence of SOPA RRs at both owner names, a client might treat a Domain attribute as though it were omitted. More generally, SOPA RRs would permit additional steps similar to steps 4 and 5 in [RFC6265].

SOPA RRs might be valuable for certificate authorities when issuing certificates, because it would allow them to check whether two names are related in the way the party requesting the certificate claims they are.

6. An example case

For the purposes of discussion, it will be useful to imagine a portion of the DNS, using the domain example.tld. A diagram of the tree of this portion is in Figure 1. In the example, the domain example.tld includes several other names: www.example.tld, account.example.tld, cust1.example.tld, cust2.example.tld, test.example.tld, cust1.test.example.tld, and cust2.test.example.tld.

```
                             tld
                              |
                              |
                     ------example -----
                    /     /   |  \      \
                   /     /    |   \      \
                  /   www  account \      cust2
              test                  \
             /   \                   cust1
         cust1   cust2
```

Figure 1

In the example, the domain tld delegates the domain example.tld. There are other possible cut points in the example, and depending on whether the cuts exist there may be implications for the use of the examples. See Section 6.1, below.

The (admittedly artificial) example permits us to distinguish a number of different roles. To begin with, there are three parties involved in the operation of services:

- OperatorV, the operator of example.tld;
- Operator1, the operator of cust1.example.tld;
- Operator2, the operator of cust2.example.tld.

Since there are three parties, there are likely three administrative boundaries as well; but the example contains some others. For instance, the names www.example.tld and example.tld are in this case in the same policy realm. By way of contrast, account.example.tld might be treated as completely separate, because OperatorV might wish to ensure that the accounts system is never permitted to share anything with any other name. By the same token, the names underneath test.example.tld are actually the test-instance sites for customers. So cust1.test.example.tld might be in the same policy realm as cust1.example.tld, but test.example.tld is certainly not in the same administrative realm as www.example.tld.

Finally, supposing that Operator1 and Operator2 merge their operations, it seems that it would be useful for cust1.example.tld and cust2.example.tld to lie in the same policy realm, without including everything else in example.tld.

6.1. Examples of using the SOPA record for determining boundaries

This section provides some examples of different configurations of the example tree in Section 6, above. The examples are not exhaustive, but may provide an indication of what might be done with the mechanism. [[anchor6: This needs to have examples of using the

ports and scheme added to it. Suggestions welcome. Also, I think some examples could be made longer to make them clearer. Maybe complete zone files in presentation form in an appendix? --ajs@anvilwalrusden.com]]

6.1.1. One delegation, eight administrative realms, no root target

In this scenario, the example portion of the DNS name space contains all and only the following SOPA records for the names-only strategy:

```
example.tld 86400 IN SOPA www.example.tld
www.example.tld 86400 IN SOPA example.tld
```

For the scheme-and-port strategy, these are the records instead:

```
example.tld 86400 IN SOPA 0 65535 * www.example.tld
www.example.tld 86400 IN SOPA 0 65535 * example.tld
```

Tld is the top-level domain, and has delegated example.tld. The operator of example.tld makes no delegations. There are four operators involved: the operator of tld; OperatorV; Operator1, the operator of the services at cust1.example.tld and cust1.test.example.tld; and Operator2, the operator of the services at cust2.example.tld and cust2.test.example.tld.

In this arrangement, example.tld and www.example.tld positively claim to be within the same policy realm. Every other name stands alone. A query for an SOPA record at any of those other names will result in a No Data response, which means that none of them include any other name in the same policy realm. As a result, there are eight separate policy realms in this case: tld, {example.tld and www.example.tld}, test.example.tld, cust1.test.example.tld, cust2.test.example.tld, account.example.tld, cust1.example.tld, and cust2.example.tld.

6.1.2. One delegation, eight administrative realms, root targets

This example mostly works the same way as the one in Section Section 6.1.1, but there is a slight difference. In this case, in addition to the records listed in Section 6.1.1, both tld and test.example.tld publish root targets in their SOPA records. For names-only:

```
tld 86400 IN SOPA .
test.example.tld 86400 IN SOPA .
```

For scheme-and-port:

```
tld 86400 IN SOPA 0 65535 * .
test.example.tld 86400 IN SOPA 0 65535 * .
```

The practical effect of this is largely the same as the previous example, except that these expressions of policy last 86,400 seconds instead of the length of time on the negative TTL in the relevant SOA for the zone. Many zones have short negative TTLs because of expectations that newly-added records will show up quickly. This mechanism permits such names to express their administrative isolation for predictable periods of time. Moreover, because clients are permitted to retain these records during periods when DNS service is not available, a client could go offline for several weeks, and return to service with the presumption that test.example.tld is still not in any policy realm with any other name.

6.1.3. Two delegations, seven or eight policy realms, root targets

In this scenario, example.tld delegates the name test.example.tld. In this case, in addition to the SOPA record at test.example.tld, there is an SOA record for test.example.tld. So, there are the same SOPA records as in Section 6.1.2. The addition of the SOA record for test.example.tld does not affect the relationship between test.example.tld and example.tld. At this point, there are eight policy realms.

Next, the Operator1 determines that it is safe to treat the test instance and production instance as being in the same policy realm. To begin with, Operator1 asks OperatorV to add the following record to the test.example.tld zone for the names-only case:

```
cust1.test.example.tld 86400 IN SOPA cust1.example.tld
```

And for the scheme-and-port case:

```
cust1.test.example.tld 86400 IN SOPA 0 65535 * cust1.example.tld
```

This arrangement is not complete yet. Until a record is also added at cust1.example.tld, Operator1's intention is only half fulfilled. The service at cust1.test.example.tld treats cust1.example.tld as part of a common policy realm, but the converse is not the case. [[anchor7: I can't decide whether there's anything useful in this configuration. Thoughts? I also can't decide whether this is still 8 admin realms, 7 admin realms but broken, or 7 admin realms from one perspective and 8 from another. --ajs@anvilwalrusden.com]]

To complete the process, Operator1 asks OperatorV to add the following record to the example.tld zone in the names-only case:

```
      cust1.example.tld 86400 IN SOPA cust1.test.example.tld
```

In the scheme-and-port case:

```
      cust1.example.tld 86400 IN SOPA 0 65535 * cust1.test.example.tld
```

Once this is complete, both names treat the other as part of the same policy realm. In the end, the example segment of the DNS expresses the following seven policy realms: tld, {example.tld, www.example.tld}, test.example.tld, {cust1.test.example.tld, cust1.example.tld}, cust2.example.tld, account.example.tld, cust2.test.example.tld.

7. Limitations of the approach and other considerations

There are four significant problems with this proposal, all of which are related to using DNS to deliver the data.

The first is that new DNS RRTYPEs are difficult to deploy. While adding a new RRTYPE is straightforward, many provisioning systems do not have the necessary support and some firewalls and other edge systems continue to filter RRTYPEs they do not know.

The second is that it is difficult for an application to obtain data from the DNS. The TTL on an RRset, in particular, is usually not available to an application, even if the application uses the facilities of the operating system to deliver other parts of an unknown RRTYPE.

The third, which is mostly a consequence of the above two, is that there is a significant barrier to adoption: until browsers have mostly all implemented this, operations need to proceed as though nobody has. But browsers will need to support two mechanisms for some period of time if they are to implement this mechanism at all, and they are unlikely to want to do that. This may mean that there is no reason to implement, which also means no reason to deploy. This is made worse because, to be safe, the mechanism really needs DNSSEC, and performing DNSSEC validation at end points is still an unusual thing to do. This limitation may not be as severe for use-cases that are directed higher in the network (such as using this mechanism as an automatic feed to keep the public suffix list updated, or for the use of CAs when issuing certificates.

Finally, in many environments the system hosting the application has only proxied access to the Internet, and cannot query the DNS directly. It is not clear how such clients could ever possibly

retrieve the SOPA record for a name.

7.1. Handling truncation

It is possible to put enough SOPA records into a zone such that the resulting response will exceed DNS or UDP protocol limits. This is especially true in the case where one wishes to take advantage of the scheme-and-port approach, and one expresses many different such relationship. In such cases, a UDP DNS response will arrive with the TC (truncation) bit set. An SOPA response with the TC bit must be queried again in order to retrieve a complete response, in order to ensure that there is no missing root target (see Section 5.1.1), generally using TCP. This increases the cost of the query, increases the time to being able to use the answer, and may not work at all in networks where administrators mistakenly block port 53 using TCP.

8. Security Considerations

This mechanism enables publication of assertions about administrative relationships of different DNS-named systems on the Internet. If such assertions are accepted without checking that both sides agree to the assertion, it would be possible for one site to become an illegitimate source for data to be consumed in some other site. In general, assertions about another name should never be accepted without querying the other name for agreement.

Undertaking any of the inferences suggested in this draft without the use of the DNS Security Extensions exposes the user to the possibility of forged DNS responses.

9. IANA Considerations

IANA will be requested to register the SOPA RRTYPE if this proceeds.

10. Acknowledgements

The author thanks Adam Barth, Dave Crocker, Jeff Hodges, John Klensin, Murray Kucherawy, Gervase Markham, Patrick McManus, Henrik Nordstrom, Yngve N. Pettersen, Eric Rescorla, Thomas Roessler, Peter Saint-Andre, and Maciej Stachowiak for helpful comments.

11. References

11.1. Normative References

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

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

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.

[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.

11.2. Informative References

[I-D.pettersen-subtld-structure]
Pettersen, Y., "The Public Suffix Structure file format and its use for Cookie domain validation", draft-pettersen-subtld-structure-09 (work in progress), March 2012.

[RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.

[RFC2308] Andrews, M., "Negative Caching of DNS Queries (DNS NCACHE)", RFC 2308, March 1998.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.

[RFC3403] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database", RFC 3403, October 2002.

[RFC4290] Klensin, J., "Suggested Practices for Registration of Internationalized Domain Names (IDN)", RFC 4290, December 2005.

[RFC4395] Hansen, T., Hardie, T., and L. Masinter, "Guidelines and Registration Procedures for New URI Schemes", BCP 35, RFC 4395, February 2006.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, March 2011.

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, August 2011.

[RFC6672] Rose, S. and W. Wijngaards, "DNAME Redirection in the DNS", RFC 6672, June 2012.

URIs

[1] <http://www.iana.org/assignments/uri-schemes.html>

Appendix A. Discussion Venue

This Internet-Draft is discussed on the applications area working group mailing list: apps-discuss@ietf.org.

Appendix B. Change History

00 to 01:

* Changed the mnemonic from BOUND to AREALM
* Added ports and scheme to the RRTYPE
* Added some motivating text and suggestions about what can be done with the new RRTYPE
* Removed use of "origin" term, because it was confusing. The document filename preserves "origin" in the name in order that the tracker doesn't lose the change history, but that's just a vestige.

   * Removed references to cross-document information sharing and ECMAScript. I don't understand the issues there, but Maciej Stachowiak convinced me that they're different enough that this mechanism probably won't work.
   * Attempted to respond to all comments received. Thanks to the commenters; omissions and errors are mine.

01 to 02:

   * Changed mnemonic again, from AREALM to SOPA. This in response to observation by John Klensin that anything using "administrative" risks confusion with the standard administrative boundary language of zone cuts.
   * Add discussion of two strategies: name-only or scheme-and-port.
   * Increase prominence of utility to CAs. This use emerged in last IETF meeting.

Author's Address

Andrew Sullivan
Dyn, Inc.
150 Dow St
Manchester, NH 03101
U.S.A.

Email: asullivan@dyn.com