Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 6, 2014

A. Sullivan
Dyn, Inc.
J. Hodges
PayPal

July 5, 2013

Asserting DNS Policy Realm Boundaries: The SOPA Resource Record
draft-sullivan-domain-policy-authority-00

Abstract

Some Internet client entities on the Internet make inferences about the administrative relationships among services on the Internet based on the domain names at which they are offered. At present, it is not possible to ascertain organizational administrative boundaries in the DNS, therefore such inferences can be erroneous in various ways. Mitigation strategies deployed so far will not scale. The solution presented in this memo is to provide a means to make explicit assertions regarding certain administrative relationships between domain names.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 6, 2014.

Copyright Notice

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction and Motivation

Many Internet resources and services, especially at the application layer, are identified primarily by DNS domain names [RFC1034]. As a result, domain names have become fundamental elements in building security policies and also in affecting user agent behaviour. For example, domain names are used for defining the scope of HTTP state management "cookies" [RFC6265].

Another example is a user interface convention that purports to display an "effective domain name" differently from other parts of a 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 domain name actually ends in "attackersite.tld", in the hope of reducing user's potential 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 domain names and X.509 certificates, see [RFC6125].

The simplest policy, and the one most likely to work, 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. Unfortunately, this approach is too naive to be useful. Often, the real policy control is the same in several names (in this example, example.org and its children). Therefore, clients have attempted to make more sophisticated policies around some idea of such shared control. We call such an area of shared control a "policy realm", and the control held by the administrator the "policy authority".

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 also too naive, and later attempts were based on inferences from the domain names themselves. That did not work well, because there is no way in the DNS to discover the boundaries of policy 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]). 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 names 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.

These different issues often appear to be different kinds of problems. The issue of whether two names may set cookies for one another appears to be a different matter from whether two names get the same highlighting in a browser's address bar, or whether a particular name "owns" all the names underneath it. But the problems all boil down to the same fundamental problem, which is that of determining whether two different names in the DNS are under the control of the same entity and ought to be treated as having an important administrative relationship to one another.

What appears to be needed is a mechanism to determine policy boundaries in the DNS. That is, given 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 "being 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 -- which is also known as the "effective TLD (eTLD) list", and henceforth in this memo as 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. Prerequisites, Terminology, and Organization of this Memo

The reader is assumed to be familiar with the DNS ([RFC1034] [RFC1035]) and the omain Name System Security Extensions (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].

To begin, Section 3 discusses the cases where this technique might be useful. Section 4 describes the mechanism in general terms. A definition of the new RRTYPE is in Section 5. There is some discussion of the use of the RRTYPE is in Section 6. Section 7 attempts to show how the mechanism can address the use cases in general terms. Then, Section 8 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 8.2 notes some limitations of the mechanism. Section 9 outlines how the mechanism might be used securely.

3. Use Cases

In the most general sense, this memo presents a mechanism that can be used either as a replacement of the public suffix list [publicsuffix.org], or else as a way to build and maintain such a list. The mechanism outlined here is explicitly restricted to names having ancestor-descendant or sibling relationships, but only as a practical matter; nothing about the mechanism makes that restriction a requirement.

HTTP state management cookies

The mechanism can be used to determine the scope for data sharing of HTTP state management cookies [RFC6265]. Using this mechansim, it is possible to determine whether a service at one name may be permitted to set a cookie for a service at a different name. (Other protocols use cookies, too, and those approaches could benefit similarly.)

User interface indicators

User interfaces sometimes attempt to indicate the "real" domain name in a given domain name. A common use is to highlight the portion of the domain name believed to be the "real" name -- usually the rightmost three or four labels in a domain name

string.

Setting the document.domain property

The DOM same-origin policy might be helped by being able to identify a common policy realm.

Email authentication mechanisms

Mail authentication mechanisms such as DMARC [I-D.kucherawy-dmarc-base] need to be able to find policy documents for a given domain name given a subdomain.

SSL and TLS certificates

Certificate authorities need to be able to discover delegation-centric domains in order to avoid issuance of certificates at or above those domains.

HSTS and Public Key Pinning with includeSubDomains flag set

[[TODO]]

Linking domains together for reporting purposes

[[TODO]]

4. Overview of Start Of Policy Authority (SOPA)

This memo presents a way to assert that two domains lie in the same policy realm by placing a resource record (RR) at the domain names. 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 often only partly based on them. Often, there are implicit rules that stem from associated components of composite names such as URIs [RFC3986], e.g., the destination port [RFC6335] or URI scheme [RFC4395] (or both). It is possible to make those assumptions explicit, but at the cost of expressing in the resulting resource record a tighter relationship between the DNS and the services offered at domain names. SRV [RFC2782] records offer a mechanism for expressing such relationships, and a SOPA record in conjunction with an SRV record appears to provide the necessary mechanism to express such relationships. (SRV records use underscore labels, and this is an example of why underscore labels themselves need to be available for SOPA records.)

It is worth observing that a positive 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 a.example.com provides a target at b.example.com, there should be a complementary SOPA RR at b.example.com with a target of a.example.com. Because of the distributed nature of the DNS, and because other DNS administrative divisions need not be congruent to policy realms, the only way to know whether two domain names are in the same policy realm is to query at each domain name, and to correlate the responses.

4.1. Identifying a Target Name for Policy Authority

The RDATA of a SOPA RR contains a "target name", lying either in the same policy realm as the owner name of the RR, that lies outside of that policy realm. The SOPA record is therefore an assertion, on the part of the authoritative DNS server for the given owner name, that there is some policy relationship between the owner name and the target name. If a given target 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

policy target is all the names beneath a given owner name, by using a wildcard target.

5. The SOPA Resource Record

The SOPA resource record, type number [TBD1], contains two fields in its RDATA:

Relation: A one-octet field used to indicate the relationship between the owner name and the target.

Target: A field used to contain a domain name, relative to the root, that is in some relationship with the owner name.

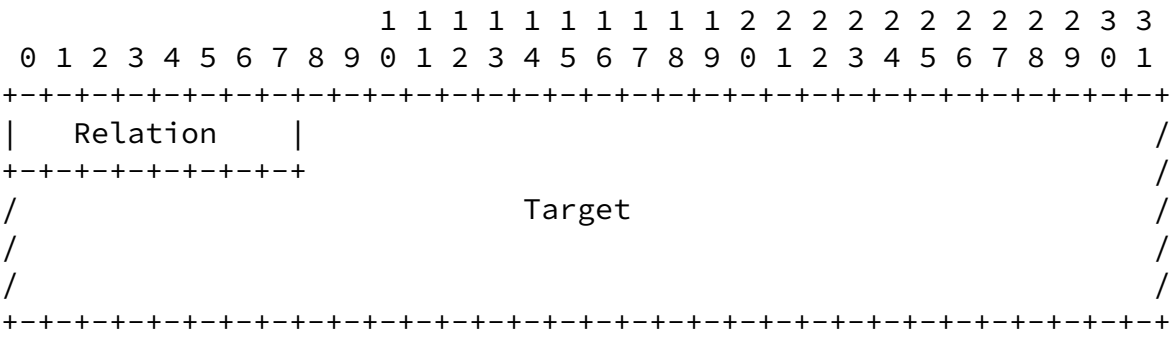

Figure 1

5.1. The Relation Field

The relation field is REQUIRED and contains an indicator of the relationship between the owner name and the target name. This memo specifies two possible values:

Value	Setting	Meaning
1	Excluded	The target is not in the same policy realm as the owner name
2	Included	The target is in the same policy realm as the owner name

Table 1

5.2. The Target Field

The target field contains a (fully-qualified) domain name, and is REQUIRED to be populated. The name MUST be a domain name according to the rules in [RFC1034] and [RFC1035], except that the left-most label of the target MAY be the wildcard character ("*"). The target MUST be sent in uncompressed form [RFC1035] [RFC3597]. The target MUST NOT be an alias [RFC2181], such as the owner name of a CNAME RR [RFC1034], DNAME RR [RFC6672], or other similar such resource records.

The target name MUST be either an ancestor, a descendent, or a sibling of the owner name in the record. This requirement is intended to limit the applicability of the SOPA RR to names in the same DNS hierarchy, thereby avoiding possible negative side effects of unbounded linkages across disparate DNS subtrees, including those subtrees rooted close to, or immediately below, the DNS root.

6. Expressing Different Policies with the SOPA RRTYPE

A SOPA RR has one of three different functions. The first is to claim that two domain names are not in the same policy realm ("exclusion"). The second is to claim that two domain names are in the same policy realm ("inclusion"). In both of these cases, it is possible to make the assertion over groups of DNS names.

The third function describes a portion of the tree that would be covered by targets containing a wildcard, but where the policy is the opposite of that expressed with the wildcard. This is expressed simply by including the relevant specific exception. For example, all the subdomains under example.com could be indicated in a target "*.example.com". To express a different policy for exception.example.com than for the rest of the names under example.com requires two SOPA RRs, one with the target "*.example.com" and he other with the target "exception.example.com". The most-specific match to a target always wins.

Is is important to note that any given fully-qualified domain name does not lie in any given other name's policy realm unless there is an explicit statement by appropriate SOPA resource record(s) to the contrary. If a candidate target name does not appear in the target of any SOPA record for some owner name, then that candidate target does not lie in the same policy realm as that owner name.

It is acceptable for there to be more than one SOPA resource record

per owner name in a response. Each domain name, in the RDATA of each returned SOPA record, is treated as a separate policy statement about

the original QNAME (query name). Note, however, that the QNAME from the query might not be the owner name of the SOPA RR: if the original QNAME was an alias, then the actual SOPA owner name in the DNS database will be different. In other words, even though a SOPA target name is not allowed to be an an alias, statements about an alias are followed and are accepted transitively from the alias to the canonical name.

6.1. The Exclusion Relation

A SOPA record with a relation field with value 1 states that the owner name and the target name are not in the same policy realm. While this would seem not to be obviously useful (given that positive declarations are required for two names to be in the same policy realm), a SOPA record with a relation field value of 1 can be useful in combination with a long TTL field, in order to ensure long term caching of the policy.

In addition, an important function of SOPA is to enable the explicit assertion that no other name lies in the same policy realm as the owner name (or, what is equivalent, that the owner name should be treated as a public suffix). In order to achieve this, the operator of the zone may use a wildcard target together with a relation field value of 1. See Section 6.4.

In addition, an exclusion relation can be used to override a more general inclusion relation (i.e. with a wildcard in the target) at the same owner name. For example,

```
example.tld.      86400 IN    SOPA  2  *.example.tld

www.example.tld.  86400 IN    SOPA  1  example.tld
```

A SOPA-using client that receives a SOPA resouce record with a relation value of 1 MUST treat the owner name and the target name as lying in different policy realms.

6.2. The Inclusion Relation

A SOPA record with a relation field set to 2 is an indicator that the target name lies in the same policy realm as the owner name. In order to limit the scope of security implications, the target name and the owner name MUST stand in some ancestor-decendant or sibling relationship to one another.

The left-most label of a target may be a wildcard record, in order to indicate that all descendant or sibling names lie in the same policy realm as the owner name. See Section 6.4.

A SOPA-using client that receives a SOPA resouce record where relation is set to 2 SHOULD treat the owner name and the target name as lying in the same policy realm. If a client does not, it is likely to experience unexpected failures because the client's policy expectations are not aligned with those of the service operator.

6.3. Interpreting DNS Responses

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. That is, a No Data response is to be interpreted as though there were a SOPA resource record with relation value 1 and a wildcard target. The TTL on the policy in this case is the negative TTL from the SOA record, in case it is available.

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).

6.4. 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 SHOULD NOT serve a SOPA RR with erroneous wildcards when it is possible to suppress them, and clients receiving such a 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.

It is possible for the wildcard label to be the only label in the target name. In this case, the target is "every name". This makes it trivial for an owner name to assert that there are no other names

in its policy realm.

Because it would be absurd for there to be more than one SOPA RR with the same wildcard target in a SOPA RRset, a server encountering more than one such wildcard target SHOULD only serve the RR for the exclusion relation, discarding others when possible. Discarding other RRs in the RRset is not possible when serving a signed RRset. A client receiving multiple wildcard targets in the RRset MUST use only the RR with relation set to 1.

When a SOPA RR with a wildcard target appears in the same RRset as a SOPA RR with a target that would be covered by the wildcard, the specific (non-wildcard) RR expresses the policy for that specific owner name/target pair. This way, exceptions to a generic policy can be expressed.

6.5. TTLs and SOPA RRs

The TTL field in the DNS is used to indicate the period (in seconds) during which an RRset may be cached after first encountering it (see [RFC1034]). As is noted in Section 3, however, SOPA RRs could be used to build something like the public suffix list, and that list would later be used by clients that might not themselves have access to SOPA DNS RRsets. In order to support that use as reliably as possible, a SOPA RR MAY continue to be used even after the TTL on the

RRset has passed, until the next time that a SOPA RRset from the DNS for the owner name (or a No Data response) is available. It is preferable to fetch the more-current data in the DNS, and therefore if such DNS responses are available, a SOPA-aware client SHOULD use them. Note that the extension of the TTL when DNS records are not available does not extend to the use of the negative TTL field from No Data responses.

7. What Can be Done With a SOPA RR

Use of a 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.

The use of SOPA RRs could either replace the public suffix list or (more likely due to some limitations -- see Section 8.2) 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.

7.1. Exclusion has Priority

In order to minimize the chance of policy associations where none exist, this memo always assumes exclusion unless there is an explicit policy for inclusion. Therefore, a client processing SOPA records can stop as soon as it encounters an exclusion record: if a parent record excludes a child record, it makes no difference whether the child includes the parent in the policy realm, and conversely. By the same token, an inclusion SOPA record that specifies a target, where the target does not publish a corresponding inclusion SOPA record, is not effective.

8. 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 2. 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.

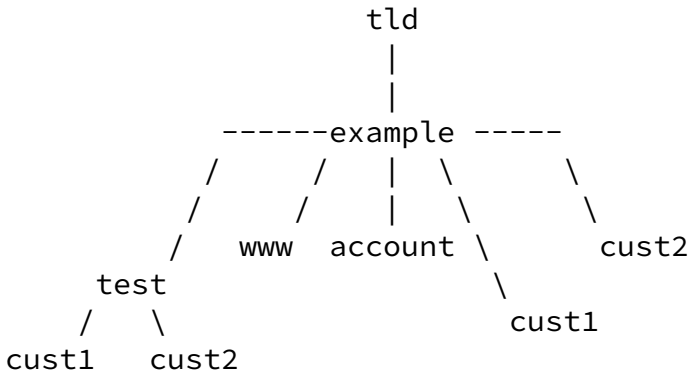

Figure 2

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 8.1, 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.

8.1. Examples of Using the SOPA Record for Determining Boundaries

This section provides some examples of different configurations of the example tree in Section 8, above. The examples are not exhaustive, but may provide an indication of what might be done with the mechanism.

8.1.1. Declaring a Public Suffix

Perhaps the most important function of the SOPA RR is to identify public suffixes. In this example, the operator of TLD publishes a single SOPA record:

```
tld.              86400 IN    SOPA  1  *
```

8.1.1.1. One Delegation, Eight Administrative Realms, Wildcard Exclusions

In this scenario, the example portion of the domain name space contains all and only the following SOPA records:

```
example.tld.      86400   IN  SOPA  2  www.example.tld

www.example.tld.  86400   IN  SOPA  2  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.

8.1.1.2. One Delegation, Eight Administrative Realms, Exclusion Wildcards

This example mostly works the same way as the one in Section Section 8.1.1.1, but there is a slight difference. In this case, in addition to the records listed in Section 8.1.1.1, both tld and test.example.tld publish exclusion of all names in their SOPA records:

```
tld.              86400   IN  SOPA  1  *

test.example.tld. 86400   IN  SOPA  1  *
```

The practical effect of this is largely the same as the previous example, except that these expressions of policy last (at least) 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 minimum 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.

8.2. 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. This is yet another reason why this mechanism is likely to be initially more useful for constructing and maintaining the public suffix list than for real-time queries.

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). This limitation could be reduced by using SOPA records to maintain something like the current public suffix list in an automatic fashion.

Fourth, 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.

8.2.1. 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. In such cases, a UDP DNS response will arrive with the TC (truncation) bit set. A SOPA response with the TC bit must be queried again in order to retrieve a complete response, 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.

9. 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.

10. IANA Considerations

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

11. Acknowledgements

The authors thank Adam Barth, Dave Crocker, Brian Dickson, Phillip Hallam-Baker, John Klensin, Murray Kucherawy, John Levine, Gervase Markham, Patrick McManus, Henrik Nordstrom, Yngve N. Pettersen, Eric Rescorla, Thomas Roessler, Peter Saint-Andre, and Maciej Stachowiak for helpful comments.

12. Informative References

[I-D.kucherawy-dmarc-base]
Kucherawy, M., "Domain-based Message Authentication, Reporting and Conformance (DMARC)", draft-kucherawy-dmarc-base-00 (work in progress), March 2013.

[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.

[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.

[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.

[RFC3597] Gustafsson, A., "Handling of Unknown DNS Resource Record (RR) Types", RFC 3597, September 2003.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[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.

[RFC4395] Hansen, T., Hardie, T., and L. Masinter, "Guidelines and

Registration Procedures for New URI Schemes", BCP 35, RFC 4395, February 2006.

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

[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.

[publicsuffix.org] Mozilla Foundation, "Public Suffix List", also known as: Effective TLD (eTLD) List.

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

draft-sullivan-domain-origin-assert-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.

02 to 03:

* Removed discussion of scheme-and-port, which was confusing.

* Add inclusion/exclusion/exception approach in response to comment by Phill H-B.

* Change mechanism for indicating "no others" to a wildcard mechanism.

* Added better discussion of use cases

03 to draft-sullivan-domain-origin-assert-00:

* Renamed file to get rid of "origin", which caused confusion.

* Added Jeff as co-author

* Remove exception relation; instead, more than one RR is allowed.

* Added discussion of SRV records

* updated title and title abbreviation

* modest rearrangement of test

* terminology polishing

Authors' Addresses

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

Email: asullivan@dyn.com

Jeff Hodges

PayPal
2211 North First Street
San Jose, California 95131
US

Email: Jeff.Hodges@PayPal.com