

Internet Engineering Task Force  
Internet-Draft  
Intended status: Informational  
Expires: January 2, 2017

C. Deccio  
Verisign Labs  
July 1, 2016

**Organizational Domains and Use Policies for Domain Names**  
**draft-deccio-dbound-organizational-domain-policy-03**

Abstract

Various Internet protocols and applications require some mechanism for identifying policy surrounding the use Domain Name System (DNS) names. In this document we describe an extensible system in which domain name policies can be discovered at various levels in the DNS tree.

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 2, 2017.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents 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

<a href="#">1.</a>	<a href="#">Introduction</a>	<a href="#">2</a>
<a href="#">1.1.</a>	<a href="#">Reserved Words</a>	<a href="#">3</a>
<a href="#">2.</a>	<a href="#">Solution Concepts</a>	<a href="#">3</a>
<a href="#">3.</a>	<a href="#">Designating Organizational Domain and Use Policy</a>	<a href="#">4</a>
<a href="#">3.1.</a>	<a href="#">ODUP Name</a>	<a href="#">4</a>
<a href="#">3.2.</a>	<a href="#">ODUP Statement</a>	<a href="#">5</a>
<a href="#">3.2.1.</a>	<a href="#">Version</a>	<a href="#">5</a>
<a href="#">3.2.2.</a>	<a href="#">Directives</a>	<a href="#">5</a>
<a href="#">3.3.</a>	<a href="#">Directive Considerations</a>	<a href="#">6</a>
<a href="#">3.3.1.</a>	<a href="#">all Directive</a>	<a href="#">6</a>
<a href="#">3.3.2.</a>	<a href="#">Directive Defaults</a>	<a href="#">7</a>
<a href="#">3.3.3.</a>	<a href="#">org Directive</a>	<a href="#">7</a>
<a href="#">3.3.4.</a>	<a href="#">bound Directive</a>	<a href="#">7</a>
<a href="#">3.3.4.1.</a>	<a href="#">Argument to bound directive</a>	<a href="#">8</a>
<a href="#">3.3.5.</a>	<a href="#">fetch Directive</a>	<a href="#">9</a>
<a href="#">4.</a>	<a href="#">Identifying Organizational Domain and Use Policy</a>	<a href="#">9</a>
<a href="#">5.</a>	<a href="#">Policy-negative Realm</a>	<a href="#">11</a>
<a href="#">5.1.</a>	<a href="#">PSL/ODUP Equivalence Example</a>	<a href="#">12</a>
<a href="#">6.</a>	<a href="#">Examples</a>	<a href="#">13</a>
<a href="#">6.1.</a>	<a href="#">ODUP Resolution</a>	<a href="#">13</a>
<a href="#">7.</a>	<a href="#">Example Application Use</a>	<a href="#">16</a>
<a href="#">7.1.</a>	<a href="#">Public Suffix List Replacement</a>	<a href="#">16</a>
<a href="#">7.2.</a>	<a href="#">HTTP Cookies</a>	<a href="#">17</a>
<a href="#">7.3.</a>	<a href="#">DMARC</a>	<a href="#">17</a>
<a href="#">8.</a>	<a href="#">Contributors</a>	<a href="#">17</a>
<a href="#">9.</a>	<a href="#">IANA Considerations</a>	<a href="#">18</a>
<a href="#">10.</a>	<a href="#">Security Considerations</a>	<a href="#">18</a>
<a href="#">11.</a>	<a href="#">References</a>	<a href="#">18</a>
<a href="#">11.1.</a>	<a href="#">Normative References</a>	<a href="#">18</a>
<a href="#">11.2.</a>	<a href="#">Informative References</a>	<a href="#">19</a>
<a href="#">Appendix A.</a>	<a href="#">Pseudo-code for ODUP Resolution</a>	<a href="#">20</a>
	<a href="#">Author's Address</a>	<a href="#">22</a>

## [1.](#) Introduction

The concepts of domains, subdomains, and administrative delegation are fundamental to the Domain Name System (DNS) [[RFC1034](#)] [[RFC1035](#)]. The DNS namespace is hierarchical, and the management of subdomain space is delegated by one entity to another throughout. However, policies governing the use of domain names do not always align with those lines of delegation. There are currently no generally reliable methods to reconcile domain names with policy for their use.

As a prominent example, HTTP cookies [[RFC6265](#)] have traditionally used ancestral relationships to determine allowable scope of information sharing and authorization. For example, the server at

Deccio

Expires January 2, 2017

[Page 2]

www.example.com might set a cookie with a Domain attribute of example.com, because it is a subdomain of that name. However, this simplistic authorization does not account for some cases, such as cookies with a Domain attribute corresponding to a so-called "public suffix". [RFC6265] indicates that many user agents reject such cookies due to security and privacy concerns. Even so, the public suffix designation is not apparent from either the names themselves or the delegations leading down to them from the root. Instead, a resource, such as Mozilla's Public Suffix List (PSL) [PSL], is used to identify public suffixes.

Another challenge with domain names is the ability to identify their respective Organizational Domains, for applications like Domain-based Message Authentication, Reporting, and Conformance (DMARC) [RFC7489]. [RFC7489] includes heuristics to identify an organizational domain using a public suffix list (e.g., Mozilla's PSL [PSL]) "in the absence of more accurate methods". Other applications have similarly relied on a public suffix list to define an organizational domain, whether for policy, forensics, or simple identification. However, defining a "more accurate" method is desirable.

In this document we describe a framework for more accurately identifying policy and organizational domains on a per-domain name basis. The system described adds customization and flexibility to existing systems while being fully backwards compatible with previous mechanisms and default behaviors.

### **1.1.    Reserved Words**

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

## **2.    Solution Concepts**

The major questions to be addressed by a solution in this space are the identification of policy for a given domain name and the identification of an organizational domain for a given domain name. The questions are related in that the policy for a domain name might depend on its organizational domain name, either because that policy is defined by its organizational domain or because it depends on organizational boundaries.

In previous solutions, policy has almost always been bound by direct ancestral relationships. While that constraint might be lifted by future solutions, the solution documented herein respects the hierarchical order in the DNS: policies between domain names in which



one is not a sub-domain of the other are non-existent and out of scope for this document.

In this context the organizational domain for a given domain name will always be an ancestor of the domain name, if not the domain name itself. Prior to this draft, one of the heuristics for determining the organizational domain was using a public suffix list: the organizational domain is the series of labels comprising the public suffix portion of the name, plus one label ([RFC6265]). With that methodology, for any given domain name, not a public suffix, there was exactly one possible organizational domain. In this work, an organizational domain can be designated at (mostly) arbitrary levels in the domain name's ancestry.

The notion of a "public suffix" (i.e., as included in a public suffix list, such as Mozilla's PSL [PSL]) has brought to light the nature of policy associated with "registry controlled domains" ([RFC6265]), sometimes viewed as extensions of top-level domains (TLDs), which have similar policy. While domain names might appear in current versions of such lists for a variety of reasons, the impact of their designation is consistent: they are treated as special-purpose names, limited in their use by applications. For example, HTTP cookies with a Domain attribute corresponding to a public suffix are rejected by most user agents. Similarly, public suffixes cannot be organizational domains, e.g., for DMARC use. We refer to the set of names above the boundary separating public suffixes from their descendants as "policy-negative".

### **3. Designating Organizational Domain and Use Policy**

In this section we describe the technical implementation for designating organization domain and use policy (ODUP) for domain names.

#### **3.1. ODUP Name**

The "\_odup" label, when used as part of a domain name, carries special meaning in the context of defining organizational domains and policy, and the resulting domain name is referred to as an ODUP name. In an ODUP name the sequence of labels after the \_odup label (i.e., higher in the DNS namespace tree) comprise the organizational domain, and the concatenation of the sequence of labels before it (i.e., lower in the DNS namespace tree) with those after it comprise the policy domain.

For example, given the ODUP name foo.bar.\_odup.example.com, the organizational domain is example.com, and the policy domain is foo.bar.example.com.



### **3.2.    ODUP Statement**

A TXT record at an ODUP name contains an ODUP statement for the policy domain. The formal syntax for an ODUP statement, using [\[RFC5234\]](#), is the following:

```
statement      =  version *( SP qual-directive )

version        =  "v=odup1"

qual-directive =  qualifier directive [ arg-sep arg ]

qualifier      =  %x2B / %x2D
                  ; addition sign (+) or hyphen (-)

directive      =  1*( directive-char )
                  ; one or more <directive-char>s

directive-char =  ALPHA / DIGIT / %x2D
                  ; letter, digit, or hyphen

arg-sep        =  %x3A
                  ; colon (:)

arg            =  1*( VCHAR )
                  ; one or more <VCHAR>s
```

#### **3.2.1.    Version**

The ODUP statement begins with a version declaration. At this time, the only defined version is "odup1", so the record **MUST** begin with "v=odup1".

#### **3.2.2.    Directives**

Following the version declaration and a space is a series of zero or more space-separated directives, each prepended with a qualifier: either "+" or "-". Additionally, each directive may optionally include a single argument, which is affixed to the end of the directive, immediately after a colon (":").

Only two policy-related directives are defined in this document. Unless otherwise specified, no arguments are defined for the directives.

- o httpcookie - whether the domain name may (+) or may not (-) be used as the value of the Domain attribute of an HTTP Cookie.





- o all - for directives not otherwise specified, the default policy is allow (+) or deny (-).

The following directives do not carry policy information, but designate organizational domains or boundaries, or offer other non-policy information. Only the "+" qualifier is valid with these directives:

- o org - specifies that the policy domain itself is an organizational domain (See [Section 3.3.3](#)).
- o bound - designates an organizational boundary below the policy domain. Domain names below that boundary are organizational domains (See [Section 3.3.4](#)). The bound directive may optionally include a numeric argument specifying the number of labels below the "\_odup" label within the domain name owning the ODUP statement. This is used to detect wildcard synthesis by ODUP resolvers (see [Section 3.3.4.1](#)).
- o fetch - designates a means for retrieving an ODUP policy realm in its entirety for local use. An argument is required, which designates one or more comma-separated URLs to use for retrieving the realm (See [Section 3.3.5](#)).

### **[3.3. Directive Considerations](#)**

#### **[3.3.1. all Directive](#)**

An ODUP statement MUST NOT have more than one "all" directive (with its accompanying qualifier). If no "all" directive is supplied, an implicit "+all" is applied.

The "all" directive SHOULD appear last in an ODUP statement. This is for readability purposes only, as directive order otherwise doesn't matter.

Because the "all" directive specifies the default policy for a given policy domain, other directives in the same statement simply specify exceptions to that default policy. Thus, the qualifier for directives other than "all" SHOULD be the opposite as that used with "all". The only side effect to non-adherence to this recommendation is the existence of superfluous directives (i.e., because they are already implied with the default policy). Implementors MAY add such directives nonetheless, for added readability.

Deccio

Expires January 2, 2017

[Page 6]

### **3.3.2. Directive Defaults**

The presence of each directive with its qualifier indicates intended policy associated with the name in question. Because the this draft introduces new behaviors, the positive ("+") value for any directive MUST match default behavior preceding this document, in order to maintain backwards compatibility. In this manner, the default policy ("all") matches previous behavior.

### **3.3.3. org Directive**

The "org" directive designates a specific domain name as an organizational domain. Because the "org" directive designates a domain name as an organizational domain, subdomains of ODUP names whose statements include the "org" directive SHOULD NOT exist, and subdomains MUST NOT be queried for policy.

For example, given the following ODUP statement:

```
example._odup.com. IN TXT "v=odup1 +org"
```

example.com is an organizational domain, and further policy information would be sought in the "\_odup.example.com" DNS domain. The domain name foo.example.\_odup.com is irrelevant.

An "org" directive MUST NOT appear together with a "bound" directive in the same ODUP statement. Additionally, the "org" directive SHOULD NOT be accompanied by any other directives; any other directives MUST be ignored by software interpreting the statement.

### **3.3.4. bound Directive**

Rather than specifying an organizational domain directly, like the "org" directive does, the "bound" directive specifies organizational domains indirectly by designating a boundary below which the organization domain exists. The actual boundary might not be immediately below the policy name specified by the "bound" statement at the corresponding ODUP name. Rather, it is a potentially uneven line dividing the subdomain space below that name. This boundary is drawn immediately below the lowest name in an unbroken line of descendants of the ODUP name for which there are no ODUP statements that include "org". Note that this line of descendants includes ODUP names with statements including "bound", statements with policy directives other than "org" and "bound", and existing ODUP names with no statements at all. It does not, however, include non-existent ODUP names.



For example, suppose the ODUP information for `www.sub.example.com` is being sought, and within the `com` organizational domain we find the following ODUP statement:

```
example._odup.com.  IN TXT "v=odup1 +bound"
```

Having this knowledge, if a subsequent query for `sub.example._odup.com/TXT` results in a name error (NXDOMAIN), then the boundary exists immediately below `example.com`, such that `sub.example.com` is the organizational domain. Likewise, if the query for `sub.example._odup.com/TXT` results in an ODUP statement with the "org" directive, such also indicates that the boundary is immediately below `example.com`. However, if the query results in a NODATA response, then the boundary is below `sub.example.com`, and additional queries are required to determine its precise location.

Subdomains of ODUP names with statements that include a "bound" directive are not restricted in the same way as they are in connection with the use of the "org" directive (See [Section 3.3.3](#)). However, subdomains of ODUP names having "bound" statements SHOULD NOT include statements that don't include "org" or "bound". Any such statements MUST be ignored.

"bound" directives SHOULD only be used in the policy-negative realm (see [Section 5](#)). In other places, the same effect can typically be accomplished through use of the "org" directive.

Finally, a "bound" directive MUST NOT appear together with an "org" directive in the same ODUP statement.

#### **[3.3.4.1](#). Argument to bound directive**

Wildcard ODUP names (i.e., those whose leftmost, or lowest, label is `*[RFC4592]`) are just as valid for ODUP names as they are generally for the DNS. However, for wildcard ODUP names whose corresponding statements include "bound", the boundary might not be detected properly if it is not known that the record is synthesized from a wildcard. Specifically, the wildcard synthesis would always yield a positive answer, and the lack of an NXDOMAIN response would cause an application identifying boundaries to continue to look for the longest match.

The numeric argument to the bound directive is used when an ODUP statement is used at a wildcard DNS name. The value is the number of non-`*` labels below the "\_odup" label. For example, the argument to the "bound" directive for the ODUP name `*.b.a._odup.example.com`. would always be 2.



### **3.3.5.    fetch Directive**

The "fetch" directive is used to designate one or more Uniform Resource Identifiers (URIs) [[RFC3986](#)] from which the ODUP statements for a given organizational domain can be downloaded for local reference. The URIs, delimited by semi-colons, comprise the argument to the "fetch" directive.

The statements for an organizational domain are downloaded either via zone transfer [[RFC1034](#)], HTTP [[RFC7230](#)] [[RFC7231](#)], or [[RFC2818](#)] HTTPS. In the case of an HTTP(s) download, the file is maintained in text format, structured as DNS zone master file [[RFC1034](#)] and retrieved using the GET method [[RFC7231](#)]. In any case, the contents of the download correspond to the records comprising the "\_odup" subdomain of the organizational domain.

The URI for a zone transfer uses "axfr" as its scheme. The host component is optional; if not specified, any of the authoritative servers advertised in the NS record set may be used. The path component is blank. The following are valid URIs for use in the fetch directive for example.com:

```
axfr://
```

```
axfr://ns1.example.com
```

```
http://www.example.com/odup.zone
```

## **4.    Identifying Organizational Domain and Use Policy**

The process of identifying organizational domains and policies is referred to as ODUP resolution. It involves methodically issuing DNS queries for DNS records of type TXT at ODUP names. The desired outcome for ODUP resolution is an organizational domain, a policy domain, and a policy. The process is iterative and completes in  $O(n)$  (upper bound) time, where  $n$  corresponds to the number of labels in the name. The algorithm finds the ODUP statement corresponding to the policy name that mostly closely matches the domain name being looked up within the lowest designated organizational domain. Each iteration is as follows, beginning with the single right-most label (i.e., TLD) as the organizational domain. We use `www.example.com` to illustrate policy lookup in each step.

1.    Reset the longest matching record and the longest existing name.
2.    Begin with the ODUP name formed by prepending the "\_odup" label to the current organizational domain, and with each subsequent





iteration increase the number of policy domain labels below the "\_odup" label by one to form an ODUP name.

Example: If the current organizational domain is com, then the iterative ODUP names would be: \_odup.com, example.\_odup.com, www.example.\_odup.com.

3. Issue a query using the ODUP name formed in the previous step for a record of type TXT.
4. If the query results in response code 0 (NOERROR), then save this ODUP name as the longest existing name.
5. If the query yields a response with an ODUP policy, and either the ODUP statement includes an "org" or "bound" directive or the previous longest matching record includes neither an "org" nor a "bound" directive, then save the record as the longest matching record.
6. If the query yields a response with an ODUP policy that includes an "org" directive, then go to Step 11.
7. If the query yields a response with an ODUP policy that includes a "bound" directive, and the response is derived from a wildcard, go to Step 11.
8. If the query results in response code 3 (name error or NXDOMAIN), go to Step 11.

Example: If a query for \_odup.com results in a name error, then we don't continue to example.\_odup.com.

9. If all the labels have been exhausted, then go to Step 11.

Example: If the current organizational domain is com, and we have already looked for policy at the ODUP names \_odup.com, example.\_odup.com, and www.example.\_odup.com, then we go to Step 11.

10. Return to step Step 2, increasing the number of labels of policy domain.

Example: If the current organizational domain is com, and we have just looked for policy at \_odup.com, then we would now look for policy at example.\_odup.com.



11. If there was no positive response (i.e., longest matching record is not set), then return an empty policy under the current organizational domain.
12. If the longest matching record includes an "org" directive, then return to Step 1, using the longest matching policy domain as the organizational domain.

Example: If the ODUP statement at `example._odup.com` included an "org" directive, then we would next look for policy at `_odup.example.com`, then `www._odup.example.com`.

13. If the longest matching record includes a "bound" directive, and the number of labels in the longest existing name is less than those in the name whose policy is being looked up, then return to Step 1, using the longest existing name domain, plus one label, as the organizational domain.

Example: If the ODUP statement at `_odup.com` included a "bound" directive, then we would next look for policy at `_odup.example.com`, then `www._odup.example.com`.

14. Return the record corresponding to the longest matching policy domain.

[Appendix A](#) contains the pseudo-code for this algorithm.

## 5. Policy-negative Realm

The policy-negative realm is the portion of the DNS namespace that corresponds to so-called public suffixes. The ODUP policies defining the policy-negative realm fall under the `_odup` subdomain of each TLD and consist of ODUP statements that use the "bound", "org", and "-all" directives.

The result is that the ODUP statements defining the policy-negative realm comprise a PSL-like list. In fact, the statements comprising the policy-negative realm can be derived from Mozilla's [\[PSL\]](#) and vice-versa. See, for example, the tables in [Section 5.1](#). This is useful for several reasons. The current applications and libraries using the PSL have a way to work in parallel, one being derived from the other, for backwards compatibility and possible transition. It also enables consumer applications to work (fully or partially) offline by simply downloading all the policy-negative statements when those are sufficient for the purposes of the application.

To enable local reference of ODUP statements within the policy-negative realm, "fetch" statements SHOULD be specified for ODUP



statements found at ODUP names immediately under TLDs, e.g., `_odup.com`. Because the nature of TLDs has changed [[NewgTLDs](#)], some TLDs might not have a complete negative policy, and thus might be "below" the policy negative realm. Such TLDs MAY choose to use a default policy (which is not negative) by not publishing an ODUP statement at all, in which case the inclusion of the "fetch" directive does not apply.

Even when a self-contained listing of the policy-negative names is locally available but insufficient for the needs of the consumer application (i.e., a more specific assurance of policy is necessary), this resource can provide an efficient starting place for ODUP resolution. Rather than starting with the TLD as the organizational domain, the shortest organizational domain below the policy-negative boundary is used first. This increases efficiency by avoiding network latency associated with unnecessary DNS lookups.

Working from a locally available version of the policy-negative realm is not only more efficient, but it also minimizes information disclosure to delegating authorities and is thus more privacy-adhering.

Directives other than "bound", "org", "fetch", and "-all" MUST NOT be used in the policy-negative realm.

### [5.1.](#)    PSL/ODUP Equivalence Example

```
+---+-----+
|   | PSL Entry |
+---+-----+
| 1 | uk       |
| 2 | co.uk    |
| 3 | *.ck     |
| 4 | !www.ck  |
+---+-----+
```

Select PSL Entries



Note that "fetch" directives are excluded to maximize space.

	ODUP Name	ODUP Statement
1	_odup.uk.	"v=odup1 +bound -all"
2	co._odup.uk.	"v=odup1 +bound -all"
3	*._odup.ck.	"v=odup1 +bound:0 -all"
4	www._odup.ck.	"v=odup1 +org"

Corresponding ODUP Names and Statements

## 6. Examples

We provide several examples and in this section of ODUP designation, resolution, and potential application.

### 6.1. ODUP Resolution

Example DNS entries are listed in Table 1 and illustrated in Figure 1. The resulting policies are listed in Table 3.

Org. Domain	ODUP Name	ODUP Statement
uk.	_odup.uk.	"v=odup1 +bound -all"
uk.	co._odup.uk.	"v=odup1 +bound -all"
ck.	_odup.ck.	"v=odup1 +bound -all"
ck.	*._odup.ck.	"v=odup1 +bound:0 -all"
ck.	www._odup.ck.	"v=odup1 +org"
a.uk.	c.b._odup.a.uk.	"v=odup1 +org"
a.uk.	e._odup.a.uk.	"v=odup1 -httpcookie"
c.b.a.uk.	_odup.c.b.a.uk.	"v=odup1 -httpcookie"

Example ODUP statements and corresponding ODUP names and organizational domains. Note that "fetch" directives are excluded to maximize space.

Table 1: Example DNS entries





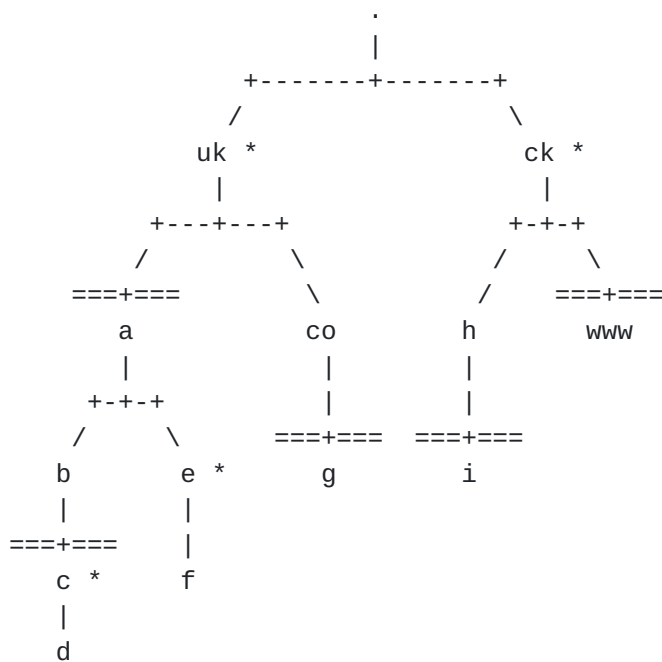


Illustration of the namespace associated with ODUP names and statements listed in Table 1. Each node represents a DNS label in its place in the DNS namespace hierarchy. Double horizontal lines (====) represent organizational boundaries, and asterisks (\*) represent explicit policy statements.

Figure 1

+-----+-----+-----+-----+					
Domain Name	DNS Queries	Responses	Step		
+-----+-----+-----+-----+					
uk	_odup.uk/TXT	+bound -all	14		
a.uk	_odup.uk/TXT	+bound -all	10		
	a._odup.uk/TXT	NXDOMAIN	13		
	_odup.a.uk/TXT	NODATA	11		
b.a.uk	_odup.uk/TXT	+bound -all	10		
	a._odup.uk/TXT	NXDOMAIN	13		
	_odup.a.uk/TXT	NODATA	10		
	b._odup.a.uk/TXT	NODATA	11		
c.b.a.uk	_odup.uk/TXT	+bound -all	10		
	a._odup.uk/TXT	NXDOMAIN	13		
	_odup.a.uk/TXT	NODATA	10		
	b._odup.a.uk/TXT	NODATA	10		
	c.b._odup.a.uk/TXT	+org	12		
	_odup.c.b.a.uk/TXT	-httpcookie	14		
d.c.b.a.uk	_odup.uk/TXT	+bound -all	10		
	a._odup.uk/TXT	NXDOMAIN	13		
	_odup.a.uk/TXT	NODATA	10		

Deccio

Expires January 2, 2017

[Page 14]

	b._odup.a.uk/TXT	NODATA	10
	c.b._odup.a.uk/TXT	+org	12
	_odup.c.b.a.uk/TXT	-httpcookie	10
	d._odup.c.b.a.uk/TXT	NXDOMAIN	14
e.a.uk	_odup.uk/TXT	+bound -all	10
	a._odup.uk/TXT	NXDOMAIN	13
	_odup.a.uk/TXT	NODATA	10
	e._odup.a.uk/TXT	-httpcookie	14
f.e.a.uk	_odup.uk/TXT	+bound -all	10
	a._odup.uk/TXT	NXDOMAIN	13
	_odup.a.uk/TXT	NODATA	10
	e._odup.a.uk/TXT	-httpcookie	10
	f.e._odup.a.uk/TXT	NXDOMAIN	14
co.uk	_odup.uk/TXT	+bound -all	10
	co._odup.uk/TXT	+bound -all	14
g.co.uk	_odup.uk/TXT	+bound -all	10
	co._odup.uk/TXT	+bound -all	10
	g.co._odup.uk/TXT	NXDOMAIN	13
	_odup.g.co.uk/TXT	NXDOMAIN	11
ck	_odup.ck/TXT	+bound -all	14
h.ck	_odup.ck/TXT	+bound -all	10
	h._odup.ck/TXT	+bound:0 -all	14
i.h.ck	_odup.ck/TXT	+bound -all	10
	h._odup.ck/TXT	+bound:0 -all	13
	_odup.i.h.ck/TXT	NXDOMAIN	11
www.ck	_odup.ck/TXT	+bound -all	10
	www._odup.ck/TXT	+org	12
	_odup.www.ck/TXT	NXDOMAIN	11

The sequence of queries, responses, and corresponding steps (i.e., from [Section 4](#)) followed for ODUP resolution of each name. The version information (i.e., "v=odup1") has been stripped from the policy for readability.

Table 2: DNS Queries for ODUP Resolution Examples



Domain Name	Org. Domain	Policy Domain	Policy ([D]efault, [E]xplicit, [I]nherited)
uk.	uk.	uk.	-all (E)
a.uk.	a.uk.	a.uk.	+all (D)
b.a.uk.	a.uk.	a.uk.	+all (I)
c.b.a.uk.	c.b.a.uk.	c.b.a.uk.	-httpcookie +all (E)
d.c.b.a.uk.	c.b.a.uk.	c.b.a.uk.	-httpcookie +all (I)
e.a.uk.	a.uk.	e.a.uk.	-httpcookie +all (E)
f.e.a.uk.	a.uk.	e.a.uk.	-httpcookie +all (I)
co.uk.	uk.	co.uk.	-all (I)
g.co.uk.	g.co.uk.	g.co.uk.	+all (D)
ck.	ck.	ck.	-all (E)
h.ck.	ck.	h.ck.	-all (I)
i.h.ck.	i.h.ck.	i.h.ck.	+all (D)
www.ck.	www.ck.	www.ck.	+all (D)

Policies resulting from the ODUP names and corresponding statements listed in Table 1 and illustrated in Figure 1. The version information (i.e., "v=odup1") has been stripped from the policy for readability. [D]efault policy means that there was no explicit policy designated for the organization, so the default ("v=odup1 +all") is used. [E]xplicit policy means that a policy is defined for the domain name in question. [I]nherited policy means that it is inherited from the closest ancestor with default or explicit policy.

Table 3: Effective ODUP Policies

## 7. Example Application Use

While designating the manner in which ODUP is applied in applications is beyond the scope of this document, for added clarity and utility, we provide some examples of how it might be consumed, using the examples data [Section 6](#) for reference.

### 7.1. Public Suffix List Replacement

As mentioned in [Section 5](#), the ODUP statements within the policy-negative realm provide a drop-in replacement for the public suffix list(s) from which they are initially derived. Whether used offline (i.e., having been downloaded and optionally converted to a legacy list format) or learned through DNS queries, the applications that previously used a public suffix list, can use the contents of the \_odup domain to achieve the same baseline behavior. That includes HTTP cookie policy [[RFC6265](#)], identifying organizational domains for DMARC [[RFC7489](#)], and other use.

Deccio

Expires January 2, 2017

[Page 16]

Beyond this baseline behavior provided by installation of the policy-negative realm, ODUP provides the opportunity for additional functionality, described subsequently.

## **7.2.    HTTP Cookies**

[RFC6265] directs user agents to reject HTTP cookies whose Domain attribute specifies a scope that does not include the origin server. The origin server is within scope if its name is equal to or a subdomain of the value of the Domain attribute. Basically, this allows an origin server to set a cookie with a Domain attribute value of any domain name in its ancestry (but below the public suffix boundary).

With ODUP policy HTTP cookies can be built on organizational boundaries, including not only the policy-negative realm, but also at other designated points. For example, given the organizational boundary between b.a.uk and c.b.a.uk, the origin server d.c.b.a.uk can set a cookie for d.c.b.a.uk and c.b.a.uk, but not for b.a.uk. Likewise, when a user-agent has a cookie with domain b.a.uk, it will not send it when communicating with d.c.b.a.uk or c.b.a.uk because organizational boundaries supercede "domain-matching" ([RFC6265]).

In addition, cookie policies between organizational boundaries can be specified using the "httpcookie" directive. For example, neither f.e.a.uk nor e.a.uk can be the value of the Domain attribute of an HTTP cookie, even though they are not public suffixes. However, this doesn't mean that user agents communicating with f.e.a.uk or e.a.uk cannot set or return cookies for a.uk, so it is not particularly useful, except within the policy-negative realm.

## **7.3.    DMARC**

Whereas [RFC7489] includes only provides a single possible organizational domain for any given domain name, ODUP allows the organizational domain to be specified, such that it can be designated and identified from anywhere within the ancestral namespace.

## **8.    Contributors**

The namespace convention used for ODUP names resembles and was inspired by that developed in [I-D.levine-orgboundary], for which we acknowledge John Levine's efforts in this area.





## **9.    IANA Considerations**

This document contains no requests for IANA.

## **10.   Security Considerations**

Applications depending on the mechanisms herein described to learn organizational domains and policies for domain names have their own security considerations. We reference [[RFC6265](#)] and [[RFC7489](#)] as examples of those.

Because the ODUP resolution mechanisms themselves rely on DNS queries (and zone transfers and/or HTTP, for the policy-negative realm), its security is only as secure as that of the underlying lookup/download mechanisms themselves. DNSSEC and HTTPS can be useful in ensuring authenticity of policy statements through the respective lookup mechanisms.

The iterative query process utilized by ODUP resolution can be potentially revealing of queries to higher DNS authorities, in part because of the necessity of finding the longest matched ODUP names. This is in contrast to the principles of minimum disclosure to maximize DNS privacy. This is mitigated by stopping at NXDOMAIN responses, as described in [Section 4](#). Additionally, referencing a locally downloaded version of the policy-negative realm for partial or "sufficient" ODUP resolution, as suggested in [Section 5](#) can reduce queries to the \_odup zone.

## **11.   References**

### **11.1.   Normative References**

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, [RFC 3986](#), DOI 10.17487/RFC3986, January 2005, <<http://www.rfc-editor.org/info/rfc3986>>.
- [RFC4592] Lewis, E., "The Role of Wildcards in the Domain Name System", [RFC 4592](#), DOI 10.17487/RFC4592, July 2006, <<http://www.rfc-editor.org/info/rfc4592>>.



- [RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, [RFC 5234](#), DOI 10.17487/RFC5234, January 2008, <<http://www.rfc-editor.org/info/rfc5234>>.
- [RFC6265] Barth, A., "HTTP State Management Mechanism", [RFC 6265](#), DOI 10.17487/RFC6265, April 2011, <<http://www.rfc-editor.org/info/rfc6265>>.
- [RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", [RFC 7231](#), DOI 10.17487/RFC7231, June 2014, <<http://www.rfc-editor.org/info/rfc7231>>.

### **11.2. Informative References**

- [I-D.levine-orgboundary] Levine, J., "Publishing Organization Boundaries in the DNS", [draft-levine-orgboundary-02](#) (work in progress), July 2013.
- [NewgTLDs] ICANN, "New Generic Top-Level Domains", 2016, <<http://newgtlds.icann.org/>>.
- [PSL] Mozilla Foundation, "Public Suffix List", 2016, <<https://publicsuffix.org/>>.
- [RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, [RFC 1034](#), DOI 10.17487/RFC1034, November 1987, <<http://www.rfc-editor.org/info/rfc1034>>.
- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, [RFC 1035](#), DOI 10.17487/RFC1035, November 1987, <<http://www.rfc-editor.org/info/rfc1035>>.
- [RFC2818] Rescorla, E., "HTTP Over TLS", [RFC 2818](#), DOI 10.17487/RFC2818, May 2000, <<http://www.rfc-editor.org/info/rfc2818>>.
- [RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", [RFC 7230](#), DOI 10.17487/RFC7230, June 2014, <<http://www.rfc-editor.org/info/rfc7230>>.



[RFC7489] Kucherawy, M., Ed. and E. Zwicky, Ed., "Domain-based Message Authentication, Reporting, and Conformance (DMARC)", [RFC 7489](https://www.rfc-editor.org/rfc/rfc7489), DOI 10.17487/RFC7489, March 2015, <<http://www.rfc-editor.org/info/rfc7489>>.

#### **Appendix A.   Pseudo-code for ODUP Resolution**

```
function resolveODUP(N = [l(n)...l(1)], orgBoundary)
{
    Require: N = [l(n)...l(1)], n >= 1
           (a sequence of labels representing domain name, N)
    Require: orgBoundary, 1 <= orgBoundary <= n
           (an integer representing an index into N)
    Return: a three-tuple consisting of:
        - policyDomain, the domain name from the which the policy
          was derived
        - orgDomain, the organizational domain of N
        - policy, the raw policy resulting from TXT DNS queries

    // orgDomain consists of labels 1 through orgBoundary
    orgDomain := [l(orgBoundary)...l(1)]

    subDomainLabels := n - orgBoundary
    longestMatch := NULL
    longestMatchBoundary := NULL
    existingLabels := 0
    for all i in [0...subDomainLabels]
    {
        if i = 0
        {
            subDomain := []
        }
        else
        {
            subDomain := [l(orgBoundary + i)...l(orgBoundary + 1)]
        }
        testDomain := subDomain | ["_odup"] | orgDomain
        queryResult := queryDNS(testDomain, TXT)

        if queryResult is either NOERROR or NODATA AND i > 0
        {
            // Update existingLabels because the name exists.
            // Only do this for labels other than _odup
            // (hence the test for i > 0).
            existingLabels := existingLabels + 1
        }

        if queryResult is an answer
```

Deccio

Expires January 2, 2017

[Page 20]

```
{
    // Update longestMatch by giving org and bound highest
    // priority and ignoring policy statements below "bound".
    if queryResult includes "+org" OR
        queryResult includes "+bound" OR
        longestMatch doesn't include "+bound"
    {
        longestMatch := queryResult
        longestMatchBoundary := i
    }

    if queryResult includes "+org"
    {
        // If this was an organizational domain designation,
        // then don't go any further; the organization will
        // dictate policy
        break
    }
    if queryResult includes "+bound" AND
        was synthesized from wildcard
    {
        // If this was a boundary designation, and the answer
        // was synthesized from a wildcard, no further
        // lookups must be performed
        break
    }
}
else if queryResult is NXDOMAIN response
{
    // An NXDOMAIN result means that no further lookups are
    // necessary, as there is no subtree
    break
}
}

if longestMatch is not NULL
{
    // If a policy has been found, then look for +org or +bound
    // directives, which will cause resolveODUP() to be called
    // recursively. A +org directive indicates that the
    // organizational domain and policy are (at least) one level
    // lower than the value of longestMatchBoundary.
    if longestMatch includes "+org"
    {
        return
            resolveODUP(N, orgBoundary + longestMatchBoundary)
    }
    // A +bound directive indicates that the organizational domain
```



Deccio

Expires January 2, 2017

[Page 21]

```
// and policy are (at least) one level lower than the value of
// existingLabels.
else if longestMatch includes "+bound" AND
    orgBoundary + existingLabels <= n
    {
        return resolveODUP(N, orgBoundary + existingLabels)
    }
}

// With no +org or +bound directives present, the orgDomain and
// policy remain as they were looked up, and are returned with
// the policy domain
return [l(orgBoundary + longestMatchBoundary)...l(1)],
    orgDomain, longestMatch
}

// Return an empty policy
return orgDomain, orgDomain, ""

}
```

#### Author's Address

Casey Deccio  
Verisign Labs  
12061 Bluemont Way  
Reston, VA 20190  
USA

Phone: +1 703-948-3200  
Email: [cdeccio@verisign.com](mailto:cdeccio@verisign.com)

