Network Working Group
Internet-Draft
Expires: August 10, 2005

A. Newton
VeriSign, Inc.
Y. Shafranovich
SolidMatrix Technologies, Inc.
February 9, 2005

Distributed Black/White Lists
draft-newton-shafranovich-distributed-blacklists-00

Status of this Memo

Copyright Notice

Abstract

Many traditional, centrally-managed blacklists and whitelists describe Internet end-points by characteristics such as connectivity type or network function, and these characteristics are often used to infer behavior from which authorization is derived. However, it is often the case that connectivity type or network function are not related to good or bad behavior. This document describes a means of creating blacklists and whitelists representative of Internet end-points based on observed behavior by many participants in a

distributed monitoring network. The authors hope that distributed lists will mitigate some of the problems associated with existing centrally managed lists. While the concept, architecture, and data model are general enough to be applied to any type of network service, the authors of this document are specifically addressing the problem of spam in blogs.

Table of Contents

1. Introduction

For years, blacklists have been used as an authorization policy mechanism for public network services, mostly email. These centrally-managed blacklists lists can be categorized into two groups:

- lists containing Internet end-points based on certain characteristics, such as how they are connected to the Internet (e.g. dial-up or residential broadband) or a type of network function they may serve (e.g. proxy or relay)
- lists containing Internet end-points that have been observed to exhibit certain behavior (e.g. sending unsolicited email).

Additionally, recently a smaller but evergrowing number of whitelists have been developed and deployed to assist network administrators in determining authorization rights for public network services. Centrally managed whitelists usually contain positive information about Internet end-points that is being vouched for by the party that administers the list. In some cases this information is collected by the administrating party independently of the end points listed, but in many cases the party administering the list charges a fee for inclusion, thus essentially operating an accreditation service.

Some blacklists and whitelists are do not necessarally list bad or good information, but rather seek to provide reputation information about Internet end points. Unfortunatly, as the case with blacklists, reputation services tend to suffer from many of the same problems stemming from accountability issues.

The purpose of such lists is to erradicate certain undesirable side-effects of a highly successful network, usually unsolicited email. However, these lists have a great tendacy to inhibit universal network access, in many cases outweighing their perceived benefits. For example:

- While it is true that many senders of unsolicited email (spam) use dial-up network connections, it is not reasonable to assume that all dial-up network connections are used to send spam: the two are unrelated.
- Constrained by the need for human verification, many lists specializing in observed unwanted behavior tend to mark whole networks as bad versus specific end-points, though there is no evidence that every end-point in a network has exhibited undesirable behavior.
- There is often little guidance available on the criteria used to create these lists and seldom useful information on how to correct errors in these lists.
- In the case of whitelists, a fee chargable for accreditation and inclusion into a whitelist may inhibit certain Internet users from

obtaining network access. For example, individuals and non-commercial users, especially ones from poorer countries may not have the resources to pay an admission fee for inclusion into a whitelist. If multiple whitelists become popular, the financial burden will greatly descrease accessibility of Internet services to those users.

For these reasons and more, these centrally-managed lists have failed to make an impact on the spam problem and to be universally adopted. This is all too evident given that spam continues to be a growing problem not only in email, but slowly spreading to other network services as well.

This document describes an architecture and data model for Distributed Black/White Lists (DxL). The intent is to leverage an peer to peer web-of-trust as opposed to a centrally managed list, hopefully providing greater accuracy and understood accountability. It should be noted, however, that the concept, architecture, and data-model for DxLs could be applied to other network services. However, the authors chose to target the design of DxLs toward a relatively new type of web application called blogging.

2. Document Terminology

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 [1].

3. Motivations

Many of the problems arising in the use of blacklists and whitelists is the fact that they are centrally managed by a third-party which may not be accountable to or trusted by a network administrator who wishes to use such lists. List users may also wish to express their opinion on specific list entries or entire lists, but due to the central nature of these lists that is not currently possible. Additionally, many Internet users and network operators already have existing relationships in place with others which can be utilized to pass along blacklist and whitelist information, instead of establishing new ones with the parties administering central lists.

In the real world, existing relationships and social networks are often used to pass along reputation information, and the digital world should in theory be no different. Thefore, in order to step around the problem of trusting the party administering the central list, we choose to distribute DxL information in a peer to peer fashion. This gives users the ability to use their existing relationships to establish a web of trust for the purposes of authorizing access to public network services (which in this case are ability to leave comments and trackbacks on blog posts, and passing referer information). We also chose to allow lists to be combined and passed on as new lists, thus allowing trust information to be propogates via a social network.

Aditionally, in order to enforce accountability and transparency, we chose to require URLs pointing to the original list from which the information originates, URLs pointing to a removal page, and creation/update data for all entries. While these may not be checked for validity in all cases, nevertheless their presence indicates to the list creators and users that these are mattersnot to be ignored. Additionally, we believe that users will take the validity of this information into account when trusting or not trusting specific lists.

In order to allow flexibility for this system, we choose to add weights to the list entries indicating the "black" or "white" value. Many existing lists provides a binary "yes/no" decision in regards to their entries which may not be flexible enough for all cases. Additionally, a weight mechanism allows users to adjust weight ratings on lists coming from other users based on their trust level.

Though this document may be the first formulization of a distributed black/white list using XML, the concept of a peer-to-peer style distribution of these lists has been seen in <http://unknowngenius.com/blog/archives/2004/11/19/spam-karma-merciless-spam-killing-machine/>

and

<http://www.jayallen.org/projects/mt-blacklist/latest/index#futurefeatures>
.

4. Architecture

Unlike DNS-based blacklists [9] (known as DNSBLs) which operate over DNS, a DxL is an XML document and is retrieved over the Internet by using a protocol such as HTTP. This is modelled after RSS, which is commonly found in the "blogosphere". Once retreived, a DxL is cached for a period of time and checked for updates upon expiration. Note, that this is not the only possible implementation or exchange mechanism available for this data.

A DxL can be composed of entries derived from a private list based on direct observation and other DxLs, known as component DxLs. Hence, a DxL propogates data from many sources.

5. Data Model

This section describes the data model of a DxL. The formal syntax for a DxL is described in Section 6.

Each DxL has the following attributes:

- DxL URI - a URI pointing to the DxL
- description - a short, textual description describing the DxL
- description URI - a URI pointing to a longer description of the DxL
- expiration date and time
- creation date and time
- last updated date and time

Each of these attributes is optional.

Each item in a DxL describes an observed instance with the following trace data:

- either an IPv4 or IPv6 address
- a protocol identifier: either a domain name or a URI (a domain name is RECOMMENDED given that URIs are free to manufacture)
- protocol content: domain names, URIs, or regular expressions (regex) describing parts of content (domain names are RECOMMENDED) - regular expressions must be typed with one of the following identifiers:
  - Perl - denotes a Perl style regular expression
  - POSIX-enhanced - denotes a POSIX enhanced style regular expression
  - POSIX-basic - denotes a POSIX basic style regular expression
- proxy - a simple note indicating it was possible to detect that the end-point served as a protocol-level proxy
- user agent
- application: text in the form of XXX.YYY where XXX is an application name and YYY is a sub-application name - describes the application or network service type specific to the trace data. These values are defined as:
  - web.referrer - web-based referrals
  - blog.comments
  - blog.trackbacks

The following are two examples of trace data from observed incidents:

1. A comment is left on a blog. The blog software records the comment as coming from 192.0.2.1. The "URL" field was submitted with the URI "http://example.org/foo" and the "comment" field was submitted with the text "Buy all your foos at foo.example.org for the lowest prices". The trace data would consist of the following:
   - an IPv4 address of 192.0.2.1

    * a protocol URI of http://example.org/foo
    * a content domain of foo.example.org or example.org

2. An entry is left in a referrer log on a web server. The entry shows the request coming from 192.0.10.1 with a referral URI of http://example.com/bar. The trace data would consist of the following:
    * an IPv4 address of 192.0.10.1
    * a protocol URI of http://example.com/bar or a protocol domain name of example.com

Each item in a DxL as the following meta-data associated with it:

- URI of DxL source - taken directly from the Dxl URI of the DxL document where the item originated
- description
- description URI
- removal URI - points to a location where instructions may be found for removing an item from the source DxL
- method - describes what process was used to determine inclusion of the item if it originated from a component DxL. These methods are:
    * intersection - the item was found in a component DxL and by direct observation of this DxL publisher
    * union - the item was found in a component DxL and was not directly observed by the publisher of this DxL
    * direct - the item was found only by direct obersvation
- hops - a non-negative integer indicating the number of times the item has been derived from a component DxL. Zero indicates the item is in the DxL of the publisher who made the observation.
- weight - a value between -1.0 and 1.0 indicating a value judgement on the item. Values less than 0 are considered negative (i.e. a blacklisted item) and values greater than 0 are considered positive (i.e. a whitelisted item). Zero is considered neutral. If value judgements are simply to be boolean (either positive or negative), the values 1.0 and -1.0 SHOULD be used.
- expiration date and time
- created date and time
- last updated date and time

The following is an example of a DxL document:

```
<?xml version="1.0" encoding="UTF-8"?>
<dxl xmlns="urn:ietf:params:xml:ns:dxl0.1"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:ns:dxl0.1 dxl.xsd"
  expires="2005-01-31T12:00:00Z" description="grumpOps DxL"
  descriptionUri="http://hxr.us/grumpops/about-dxl"
  dxlUri="http://hxr.us/grumpops/dxl.xml">
  <item>
```

```
      <traceData application="blog.trackback">
        <ip4>192.0.2.1</ip4>
        <submitterDomain>online-poker.com</submitterDomain>
        <content>
          <domain>www.online-poker.com</domain>
          <domain>online-poker.com</domain>
          <uri>http://www.online-poker.com/bogus</uri>
        </content>
        <proxy>false</proxy>
        <userAgent>SpamBuddy/1.0</userAgent>
      </traceData>
      <sourceDxlUri>http://hxr.us/grumpops/dxl.xml</sourceDxlUri>
      <description>a persistent spammer</description>
      <descriptionUri>http://hxr.us/grumpops/dxl?item=abc123</descriptionUri>
      <removalUri>http://hxr.us/grumpops/dxl-removal?item=abc123</removalUri>
      <method>intersection</method>
      <hops>0</hops>
      <weight>1.0</weight>
      <expires>2005-01-30T12:00:00Z</expires>
      <created>2005-01-20T12:00:00Z</created>
      <lastUpdated>2005-01-25T12:00:00Z</lastUpdated>
    </item>
    <item>
      <traceData application="web.referrer">
        <ip6>ff:ee::00</ip6>
        <submitterUri>http://vegas-hotels.com/</submitterUri>
        <content>
          <domain>www.vegas-hotels.com</domain>
          <domain>visit.vegas-hotels.com</domain>
          <uri>http://www.vegas-hotels.com/offer</uri>
          <uri>http://www.vegas-hotels.com/redeem</uri>
        </content>
        <proxy>true</proxy>
        <userAgent>SpamBuddy/1.0</userAgent>
      </traceData>
      <sourceDxlUri>http://shaftek.org/dxl.xml</sourceDxlUri>
      <description>a very persistent spammer</description>
      <descriptionUri>http://shaftek.org/dxl?item=def456</descriptionUri>
      <removalUri>http://shaftek.org/dxl-removal?item=def456</removalUri>
      <method>intersection</method>
      <hops>1</hops>
      <weight>0.7</weight>
      <expires>2005-01-31T12:00:00Z</expires>
      <created>2005-01-22T12:00:00Z</created>
      <lastUpdated>2005-01-25T12:00:00Z</lastUpdated>
    </item>
  </dxl>
```

6. Formal XML Syntax

The following describes the formal XML syntax for DxL instances using XML Schema (see [2], [3], [5], and [4]). Implementors should note that this is only a formalization of the syntax for creation of interoperable processes and that an XML Schema capable parser is not required.

This formal definition uses the XML Schema 'anyType' is places where formal syntax definitions already exist:

- the syntax for domains is defined in [8]
- the syntax for IPv4 addresses is defined in [7]
- the syntax for IPv6 addresses is defined in [6]

In these cases, the formal syntax defers to the appropriate original defintion.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
        xmlns:dxl="urn:ietf:params:xml:ns:dxl0.1"
        targetNamespace="urn:ietf:params:xml:ns:dxl0.1"
        elementFormDefault="qualified" >

  <annotation>
    <documentation>
      A schema for describing
      distributed black/white lists (DxL)
    </documentation>
  </annotation>

  <element name="dxl">
    <complexType>
      <sequence>
        <element name="item" type="dxl:item"
          minOccurs="1" maxOccurs="unbounded"/>
        <any namespace="##other" processContents="skip"
          minOccurs="0" maxOccurs="unbounded" />
      </sequence>
      <attribute name="expires" type="dateTime"/>
      <attribute name="created" type="dateTime"/>
      <attribute name="lastUpdated" type="dateTime"/>
      <attribute name="dxlUri" type="anyURI"/>
      <attribute name="description" type="string"/>
      <attribute name="descriptionUri" type="anyURI"/>
    </complexType>
  </element>

  <complexType name="item">
    <sequence>
```

```
          <element name="traceData">
            <complexType>
              <sequence>
                <choice>
                  <element name="ip4" type="anyType">
                    <annotation>
                      <documentation>as defined by RFC 0791</documentation>
                    </annotation>
                  </element>
                  <element name="ip6" type="anyType">
                    <annotation>
                      <documentation>as defined by RFC 3513</documentation>
                    </annotation>
                  </element>
                </choice>
                <choice minOccurs="0">
                  <element name="submitterDomain" type="anyType">
                    <annotation>
                      <documentation>as defined by RFC 1035</documentation>
                    </annotation>
                  </element>
                  <element name="submitterUri" type="anyURI"/>
                </choice>
                <element name="content" minOccurs="0">
                  <complexType>
                    <choice minOccurs="0" maxOccurs="unbounded">
                      <element name="domain" type="anyType">
                        <annotation>
                          <documentation>as defined by RFC 1035</documentation>
                        </annotation>
                      </element>
                      <element name="uri" type="anyURI"/>
                      <element name="regex">
                        <complexType>
                          <simpleContent>
                            <extension base="string">
                              <attribute name="type" type="NMTOKEN"
use="required"/>
                            </extension>
                          </simpleContent>
                        </complexType>
                      </element>
                    </choice>
                  </complexType>
                </element>
                <element name="proxy" type="boolean" minOccurs="0"/>
                <element name="userAgent" type="token" minOccurs="0"/>
                <any namespace="##other" processContents="skip"
```

```
                minOccurs="0" maxOccurs="unbounded"/>
```

```
            </sequence>
            <attribute name="application" type="dxl:application"/>
          </complexType>
        </element>

        <element name="sourceDxlUri" type="anyURI" minOccurs="0"/>
        <element name="description" type="string" minOccurs="0"/>
        <element name="descriptionUri" type="anyURI" minOccurs="0"/>
        <element name="removalUri" type="anyURI" minOccurs="0"/>
        <element name="method" type="NMTOKEN" minOccurs="0"/>
        <element name="hops" type="nonNegativeInteger" minOccurs="0"/>
        <element name="weight" type="dxl:weight" minOccurs="0"/>
        <element name="expires" type="dateTime" minOccurs="0"/>
        <element name="created" type="dateTime" minOccurs="0"/>
        <element name="lastUpdated" type="dateTime" minOccurs="0"/>
        <any namespace="##other" processContents="skip"
          minOccurs="0" maxOccurs="unbounded" />

      </sequence>
    </complexType>

    <simpleType name="weight">
      <restriction base="decimal">
        <minInclusive value="-1.0"/>
        <maxInclusive value="1.0"/>
        <fractionDigits value="3"/>
      </restriction>
    </simpleType>

    <simpleType name="application">
      <restriction base="string">
        <pattern value="\w*(\.\w*)?"/>
      </restriction>
    </simpleType>

  </schema>
```

7. References

7.1 Normative References

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

[2] World Wide Web Consortium, "Extensible Markup Language (XML) 1.0", W3C XML, February 1998, <http://www.w3.org/TR/1998/REC-xml-19980210>.

[3] World Wide Web Consortium, "Namespaces in XML", W3C XML Namespaces, January 1999, <http://www.w3.org/TR/1999/REC-xml-names-19990114>.

[4] World Wide Web Consortium, "XML Schema Part 2: Datatypes", W3C XML Schema, October 2000, <http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>.

[5] World Wide Web Consortium, "XML Schema Part 1: Structures", W3C XML Schema, October 2000, <http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>.

[6] Hinden, R. and S. Deering, "Internet Protocol Version 6 (IPv6) Addressing Architecture", RFC 3513, April 2003.

[7] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

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

7.2 Informative References

[9] Levine, J., "DNS Based Blacklists and Whitelists for E-Mail", draft-irtf-asrg-dnsbl-01.txt (work in progress), November 2004.

Authors' Addresses

Andrew L. Newton
VeriSign, Inc.
21345 Ridgetop Circle
Sterling, VA 20166
USA

Phone: +1 703 948 3382
EMail: anewton@verisignlabs.com; andy@hxr.us
URI: http://www.verisignlabs.com/

Yakov Shafranovich
SolidMatrix Technologies, Inc.

EMail: YakovS@solidmatrix.com; ietf@shaftek.org
URI: http://www.shaftek.org/

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.