Domain Name System Operations
Internet-Draft
Intended status: Informational
Expires: June 19, 2017

P. Vixie
Farsight Security, Inc.
V. Schryver
Rhyolite Software
December 16, 2016

DNS Response Policy Zones
draft-vixie-dns-rpz-02

Abstract

This document describes a method for expressing DNS response policy inside a specially constructed DNS zone, and for recursive name servers to use such poicy to return modified results to DNS clients. The modified DNS results can stop access to selected HTTP servers, redirect users to "walled gardens," block objectionable email, and otherwise defend against attack. These "DNS Firewalls" are widely used in fighting Internet crime and abuse.

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 June 19, 2017.

Table of Contents

1. Introduction

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].

This document describes DNS Firewalls, a method of expressing DNS [RFC1034] policy information inside specially constructed DNS zones,

known as Response Policy Zones (RPZs). RPZs allow DNS reputation data producers and subscribers to cooperate in the application of policies to modify DNS responses in real time. Using the policy information, DNS resolution for low-reputation DNS data can be made to deliberately fail or to return local data such as an alias to a "walled garden".

A site's DNS response policy consists of the set of rules expressed in all of the RPZs that it uses. Each rule, expressed as an RRset, consists of a trigger and an action. A full description of the expressible policies is given in Section 3 (actions) and Section 4 (triggers), while Section 6 explains how the rules are applied.

Configuration examples are given using ISC BIND Version 9 (BIND9) [ISC-ARM] syntax, because work to add RPZ to that platform was started earliest (in 2009). The RPZ specification itself is free to implement and free to use in operation. It has been implemented in other name server software. We expect that in time, additional recursive DNS implementations will also implement DNS Firewalls as described by this RPZ specification.

1.1. Discussion Venue

The discussion venue for this document is the DNS Firewalls mailing list. http://lists.redbarn.org/mailman/listinfo/dnsfirewalls offers subscriptions and archives. See also https://dnsrpz.info/

[NOTE TO EDITOR: This section must be removed before this Internet Draft is published as an RFC.]

2. Zone Format

A DNS Response Policy Zone (RPZ) is a DNS zone. Like any DNS zone, an RPZ consists of RRsets or sets of resource records (RRs) with a common owner name and type. RRsets other than SOA and NS specify actions and triggers. The owner name (left hand side) of each RRset expresses a policy trigger, while the RDATA (right hand side) encodes the action to taken when the trigger matches. Depending on the type of trigger (see Section 4), a particular characteristic of the DNS query or response is checked.

Because an RPZ is a valid DNS zone, its contents can be transferred between DNS servers in whole (AXFR) [RFC5936] or incrementally as changes occur (IXFR) [RFC1995], authenticated and protected by TSIG transaction signatures [RFC2845] and expedited by real time change notifications (NOTIFY) [RFC1996], all subject to familiar DNS access controls. An RPZ need not support query access since query access is never required. It is the zone transfer of RPZ content from

producers to subscribers which effectively publishes the policy data, and it is the subscriber's server configuration which promotes RPZ payload data into DNS control plane data.

Any valid DNS zone (including an RPZ) is required to have an SOA record and at least one NS record at its apex, which is why the SOA and NS records of an RPZ cannot themselves be used to encode DNS response policy.

The RPZ's SOA record is real, with a serial number used for NOTIFY and IXFR, and timers used for AXFR and IXFR. The MNAME field or domain name of the primary source of the zone and the RNAME field or mailbox of the person responsible for the zone are often used by RPZ providers to label their policy zones.

As for an RPZ's apex NS record(s), since query access is never required, they will never be used. Similarly, no parent delegation is required for normal operation of the RPZ. Thus, by convention, a single NS record having the deliberately bogus RDATA of "LOCALHOST." is used as a placeholder.

The format of RPZs has undergone several revisions since work began (see Section 8). All POLICY described here are from RPZ Format 1 unless otherwise noted. Policy triggers from a higher format number than a recursive name server's implementation level are expected to be invisible to that implementation. Policy actions from a higher format number are likely to be misinterpreted as CNAME local data by older implementations.

3. Policy Actions

An RPZ resource record can specify any of six results or actions.

3.1. The "NXDOMAIN" Action

A single resource record (RR) consisting of a CNAME whose target is the root domain (.) will cause a response of NXDOMAIN to be returned. This is the most commonly used RPZ action.

3.2. The "NODATA" Action

A single RR consisting of a CNAME whose target is the wildcard top-level domain (*.) will cause a response of NODATA (ANCOUNT=0) to be returned regardless of query type.

3.3. The "PASSTHRU" Action

It is sometimes necessary to exempt some DNS responses from the response policy rule that covers an entire domain or a large IP address block. Exempting some clients of a DNS resolver from all RPZ rewriting can also be useful for research into attackers and for debugging. The PASSTHRU action is intended to override other, usually more general policies. It should be written so that it appears at a higher precedence than the policies it must override. See Section 5.1 about the precedence rules.

This policy zone record

```
$ORIGIN RPZ.EXAMPLE.ORG.
ok.example.com      CNAME   rpz-passthru.
```

would exempt requests for ok.example.com from the NXDOMAIN policy or action of the following record:

```
$ORIGIN RPZ.EXAMPLE.ORG.
example.com         CNAME   .
*.example.com       CNAME   .
```

The deprecated original encoding of the PASSTHRU action was a CNAME with a target equal to the QNAME field of the DNS request. That encoding could not be used with some desirable triggers.

3.4. The "DROP" Action

The "DROP" policy that consists of discarding the request and response is specified by a CNAME whose target is "rpz-drop". For example, with

```
$ORIGIN RPZ.EXAMPLE.ORG.
example.com         CNAME   rpz-drop.
```

nothing is sent to a DNS client trying to resolve example.com, not even a DNS error response.

3.5. The "TCP-Only" Action

The "TCP-Only" policy is specified by a CNAME whose target is "rpz-tcp-only". It changes UDP responses to short, truncated DNS responses that require the DNS client to try again with TCP. It is used to mitigate distributed DNS reflection attacks and is similar to the "slip" parameter of DNS Response Rate Limiting (RRL) [ISC-RRL].

3.6. The "Local Data" Action

An RRset that is neither a special RPZ encoding of an action nor one of several problematic record types specifies local data used to generate synthetic DNS responses. The special RPZ encodings are CNAMEs with targets of NXDOMAIN (.), NODATA (*.), a top level domain starting with "rpz-", or a child of a top level domain starting with "rpz-". Problematic record types include NS and DNSSEC (see [RFC4034]), because their appearance in responses would be invalid or confuse DNS clients. Local data DNAME RRsets are also commonly rejected by RPZ subscribers for internal implementation and other reasons. If any local data policy actions are present, then any request for an RR type that is not present in the local data is answered as NODATA (ANCOUNT=0) as if the recursive DNS server using RPZ were authoritative for the query name.

The most common local data is a CNAME RR pointing to a walled garden. Such CNAME RRs are distinguishable from other rpz actions, because the CNAME target name will not be the root (.), nor the root wildcard (*.), nor be a subdomain of a top level domain that starts with "rpz-".

A special form of local data involves a CNAME RR with a wildcarded target name. Wildcards are not valid as CNAME targets in ordinary DNS zones. This special form causes the QNAME to be prepended to the wildcarded target to communicate the triggering QNAME value to the walled garden DNS server. For example a policy action of "CNAME *.EXAMPLE.COM" and a query name of "EVIL.ORG." will result in a synthetic response of "EVIL.ORG CNAME EVIL.ORG.EXAMPLE.COM." The purpose for this special form is query logging in the walled garden's DNS server.

4. Policy Triggers

There are five types of RPZ triggers, and they are encoded by RRset owner names (left hand sides) in an RPZ.

Two of these types of trigger match characteristics of the DNS query: "Client IP address" and "QNAME". They are independent of cache contents or recursion results, but must be checked conceptually when the response is ready, including after any needed recursion. Recursion can sometimes be skipped, but only if the RPZ result would not be changed (see Section 5.1).

The other three types of triggers are based on characteristics of the DNS response, that is, on the RDATA to be returned to the client, or in some cases, on NS-related RDATA used while recursively obtaining the final response, regardless of whether or not those NS records or

additional data are themselves to be returned to the client. These three trigger types are: "Response IP address", "NSDNAME", and "NSIP".

All policies are conceptually applied after recursion, so that the recursive DNS resolver's cache contains either nothing or "truth," even if this truth is hidden by current policy. If the policy changes, the original, unmodified data is available for processing under the changed policy.

4.1. The "Client IP Address" trigger (.rpz-client-ip)

The IP addresses of DNS clients sending requests can be used as triggers, which can be useful for disabling RPZ rewriting for DNS clients used to test or investigate. Client IP address policy RRsets have owner names that are subdomains of "rpz-client-ip" relativized to the RPZ apex name, preceded by an encoded IP address or block of addresses.

For example, the following would drop all requests from clients in 192.0.2.0/24 and give truthful answers to requests from a client at 2001:db8::3.

```
$ORIGIN RPZ.EXAMPLE.ORG.
24.0.2.0.192.rpz-client-ip      CNAME rpz-drop.
128.3.zz.db8.2001.rpz-client-ip CNAME rpz-passthru.
```

4.1.1. IP address encoding in triggers

The IPv4 address (or address block) "B1.B2.B3.B4/prefix" is encoded in an RPZ trigger as "prefix.B4.B3.B2.B1", with the address octets reversed just as in the IN-ADDR.ARPA naming convention. (See [RFC1034].) The prefix length ("prefix") must be between 1 and 32. All four bytes, B4, B3, B2, and B1, must be present and must be written in decimal ASCII.

For example, the address block 192.0.2.0/24 would be encoded as "24.0.2.0.192".

The IPv6 address (or address block beginning at) "W1:W2:W3:W4:W5:W6:W7:W8" is encoded in a format similar to the standard IPv6 text representation (see [RFC5952]), again reversed: "prefix.W8.W7.W6.W5.W4.W3.W2.W1". Each of W8,...,W1 is a one- to four-digit hexadecimal ASCII number representing 16 bits of the IPv6 address with no leading zeroes. All 8 words must be present unless a "zz" label is present. The "zz" label is analogous to the double-colon (::) in the standard IPv6 address representation. The "zz" label is expanded to zero-fill the middle portion of the IPv6

address. Exactly one "zz" label must be present if there are two or more consecutive zero words in the address. The prefix length ("prefix") must be between 1 and 128

For example, the address 2001:db8::3 (with implied prefix length 128) would be encoded as "128.3.zz.db8.2001".

4.2. The "QNAME" trigger ("example.com")

The QNAME policy trigger matches on requested domains, the QNAME field in the question section of DNS requests and responses. (See [RFC1035].) The owner name of an RPZ QNAME policy RRset is the relativized name of the domain name about which policy is being expressed. For example, if the RPZ apex name is RPZ.EXAMPLE.ORG, an RRset at example.com.RPZ.EXAMPLE.ORG would affect responses to requests about example.com.

Wildcards also work, and so the owner name "*.example.com.RPZ.EXAMPLE.ORG" would trigger on queries to any subdomain of example.com. To control the policy for both a name and its subdomains, two policy RRsets must be used, one for the domain itself and another for a wildcard subdomain. In the following example, queries for both example.com and all subdomains of example.com will result in synthetic NXDOMAIN responses.

```
$ORIGIN RPZ.EXAMPLE.ORG.
example.com          CNAME   .
*.example.com        CNAME   .
```

4.3. The "Response IP address" trigger (.rpz-ip)

The response IP policy trigger matches response contents (RDATA): it matches IP addresses that would otherwise appear in A and AAAA records in the answer section of a DNS response. IP addresses in the authority and additional sections are not considered. Response IP policy RRsets have owner names that are subdomains of "rpz-ip" relativized to the RPZ apex name, and an encoded IP address or block of addresses. The IP address encodes are identical to those described in Section 4.1.1for Client IP Address triggers.

For example, to force an NXDOMAIN response whenever a truthful response would contain an answer section A RRset having an address in 192.0.2.0/24 unless address 192.0.2.2 is present, the RPZ would contain these records:

```
$ORIGIN RPZ.EXAMPLE.ORG.
24.0.2.0.192.rpz-ip   CNAME   .
32.2.2.0.192.rpz-ip   CNAME   rpz-passthru.
```

In another example, to answer NODATA (ANCOUNT=0) whenever a truthful response would contain an answer AAAA RRset having an address 2001:db8:101::/48 unless address 2001:db8:101::3 was present, the RPZ would contain these records:

```
$ORIGIN RPZ.EXAMPLE.ORG.
48.zz.101.db8.2001.rpz-ip       CNAME   *.
128.3.zz.101:db8.2001.rpz-ip    CNAME   rpz-passthru.
```

Please refer to Section 5.1 to understand how the above exception mechanims work.

4.4. The "NSDNAME" trigger (.rpz-nsdname)

The NSDNAME policy trigger matches name server names (NS RR) of any name server which is in the data path for an RRset present in the answer section of a DNS response. The data path is all delegation points from (and including) the root zone to the closest enclosing NS RRset for the owner name of the answering RRset.

In other words, an NSDNAME trigger is checked by first considering the named servers (domain names in the NS records) for the query domain (QNAME), then the name servers for the parent of the query domain name, and so on until the name servers for the root (.) have been checked or there fewer periods (.) in the domain name than the value of a local "min-ns-dots" parameter. See Section 4.4.1.1 below.

NSDNAME policies are encoded as RRsets in subdomains of "rpz-nsdname" but otherwise much like QNAME policies (xref target="qname"/>). For example, to force an NXDOMAIN answer whenever a name server for the requested domain or one of its parents is ns.example.com, the RPZ would contain the following:

```
$ORIGIN RPZ.EXAMPLE.ORG.
ns.example.com.rpz-nsdname   CNAME .
```

4.4.1. Implementation considerations for NSDNAME triggers

Note that these considerations apply also to NSIP triggers described in Section 4.5 below.

4.4.1.1. Performance issues

The process of traversing the data path from the level nearest the queried record to the top (root domain) level can be expensive, especially when it comes to checking the many NS records for the top level domains and the root. Because the name servers for the root and the TLDs are rarely used as RPZ triggers, some RPZ

implementations have a "min-ns-dots" parameter that stops NSDNAME and NSIP checking early.

Despite their costs, NSDNAME and NSIP triggers can be more effective than QNAME and IP triggers. Miscreants can more easily change their direct domain names and IP addresses (which are detected by QNAME and IP triggers) than they can their change NS names and addresses (detected by NSDNAME and NSIP triggers) in parent domains such as TLDs.

4.4.1.2. Caching of NS records and related address data

Some implementations of DNS RPZ will attempt to exhaustively discover all ancestral zone cuts above the query name and learn the NS RRset at the apex of each delegated zone. Other implementations will know only the zone cut information which has naturally come into the cache, which will often include only name server names and addresses from the parent. Apex ("below the cut") name server names and addresses often do not exactly match those from the parent. Such inconsistencies can lead to instability in DNS RPZ behavior. This potential inconsistency must be taken into account when designing a security policy or testing DNS RPZ.

For NSDNAME and NSIP triggers, the BIND9 and Unbound RPZ implementations (as of 2016) match the NS, A, and AAAA RRsets already in their caches unless there are none, in which case they recurse. This strategy works well in practice, because their caches were likely recently populated while generating the unmodified response and checking QNAME and response IP address triggers. In addition, the authoritative apex NS RRset (which, if obtained, would supersede the cached NS RRset from the delegating zone) of a domain operated by a malefactor is often peculiar. Even when it is reasonable, the authoritative DNS servers for such a domain are often extremely slow or broken. For example, RRs like "example.com NS ." claiming root as the authoritative server for a second or lower level domain are popular choices in miscreant apex NS RRsets, as are imaginative name servers A and AAAA RRsets.

4.5. The "NSIP" trigger (.rpz-nsip)

The NSIP policy trigger matches name server addresses, that is A or AAAA RRs referenced by an NS RRset. NSIP is much like NSDNAME (described above) except that the matching is by name server address rather than name server name. NSIP policies are expressed as subdomains of "rpz-nsip" and have the same subdomain naming convention as described for response IP policy triggers above (Section 4.1.1).

In a process very similar to that for an NSDNAME trigger (Section 4.4), an NSIP trigger is checked by first considering all of the IP addresses for all the named servers for the QNAME, then proceeding similarly parent of the QNAME, and so on until the name servers for the root (.) have been checked or a policy has been matched.

Policies are applied in order of precedence (see Section 5.1) at each level in the data path. The data path traversal process stops immediately when there is at least one policy match at a given level.

For example, to force an NXDNAME answer whenever one of the name servers for the requested domain (QNAME) or one of its parents has an address in the 192.0.2.0/24 block, the RPZ would contain the following:

```
$ORIGIN RPZ.EXAMPLE.ORG.
24.0.2.0.192.rpz-nsip     CNAME    .
```

4.5.1. Implementation considerations for NSIP triggers

The performance and caching considerations for the implementation of NSIP triggers are essentially identical to those described for NSDNAME triggers (Section 4.4.1).

5. Application of the Policy

To enable the use of RPZs, the recursive name server's control plane is connected to the DNS RPZ data plane by supplying an ordered list of RPZs in the name server's configuration. For each DNS RPZ in this list, it should be possible to specify an optional overriding policy action to be used for any policy triggers found in that RPZ. These override policies should include NXDOMAIN, NODATA, PASSTHRU, DROP, TCP-ONLY, CNAME domain, GIVEN, and DISABLED. The first five of these actions are defined in Section 3 above. "CNAME domain" is a restricted case of the "Local Data" action (also defined in Section 3) in which the rewritten response is a CNAME RR targeting "domain." GIVEN explicitly reaffirms the default, which is to respect all policy actions found in this DNS RPZ. The overriding DISABLED action causes triggered actions in the zone to have no effect except to log what would have happened, provided sufficient logging is enabled; this is useful for debugging or previewing a policy zone.

By default, policies are applied only on DNS requests that ask for recursion (RD=1). Recursive DNS servers generally send their requests to authority servers without asking for recursion (RD=0), while stub resolvers ask for recursion (RD=1). Thus, RPZ at a

recursive server by default only affects requests from stub resolvers. This default can be overridden in some implementations with configuration statements such as "recursive-only no".

Also by default, RPZ policies are only applied to responses to DNS requests that do not request DNSSEC metadata (DO=0) or for which no DNSSEC metadata exists. This defaults can be overridden in some implementations with a configuration statement such "break-dnssec yes". See Section 10 about the implications of responding with modified DNS responses when the DNS client seems to be expecting DNSSEC signatures.

If a policy rule matches and results in a modified answer, then that modified answer will include in its authority section the SOA RR of the policy zone whose policy was used to generate the modified answer. This SOA RR includes the name of the DNS RPZ and the serial number of the policy data which was connected to the DNS control plane when the answer was modified.

Conceptually, policies MUST be applied after recursion is complete and all data needed to formulate a response is available. However, implementations MAY short-circuit the process such as not waiting for recursion when it is clear which modification will be made to the response. Nevertheless, it SHOULD be possible to configure the resolver to continue checking and filling its cache by recursion as if it had not already made its decision, in order to deny operators of authority servers for listed domains information about whether they are listed, that is, in order to minimize giving hints to miscreants about when to change their DNS data. In BIND9, for example, this behavior is controlled with the "qname-wait-recurse" configuration option.

When the QNAME is resolved with CNAME or DNAME, there are no response IP address that might match a response IP address trigger, but NSIP and NSDNAME triggers might be matched. Unless the original query type is ANY, CNAME, or DNAME, the resolver will start over and try to resolve the target of the CNAME. RPZ is also restarted and the CNAME target is matched against CNAME policy rules resolved IP addresses (if any) are matched with response IP address policy triggers, and so forth. This process is repeated as the resolver follows the CNAME chain.

5.1. Precedence Rules.

More than one policy trigger among the various DNS RPZs connected to the name server's control plane can match a given DNS response, but only a single policy rule can affect the response. In theory and for standardization, all possible rules are considered simultaneously and

the following precedence rules are used to choose the single best RPZ rule. In implementations, policy triggers are usually considered in a sequence that mirrors the process of generating the DNS response, because checking RPZ triggers is conveniently made a part of that process. For example, client IP RPZ address triggers are often checked early as the DNS request is being received and the client IP address is checked in the access control list (ACL) that determines which DNS client IP addresses can ask for recursion. The QNAME is available for RPZ trigger matching before any response IP addresses are known and so QNAME poliocy triggers are usually checked immediately after client IP address triggers and before response IP address triggers. NSIP and NSDNAME triggers are often checked last. As far as the DNS client can determine, it MUST seem that all matching triggers are found and weighed using the precedence rules, but in practice, shortcuts are taken. For example, according to the precedence rules, a matched QNAME trigger in the first policy zone makes all response IP address, NSIP, and NSDNAME triggers moot. There is no need to look for those matches, because they will not affect the response.

The following list is ordered so that rules listed early override rules listed later.

RPZ Ordering
: Changes triggered by records in RPZs specified earlier in the ordered set of DNS RPZs are applied instead of triggers in policy zone specified later.

Within An RPZ
: Among policies from a single RPZ, client IP address policies are chosen instead of QNAME policies, QNAME policies are preferred to IP, IP policies are preferred to NSDNAME, and NSDNAME policies are preferred to NSIP.

Exact name match
: As in exact versus wildcard domain name matching at authority servers, exact domain name QNAME or NSDNAME triggers are preferred to wildcards.

Name Length
: The preceding rule implies QNAME policies are preferred to NSDNAME policies.

  Among triggered QNAME or NSDNAME policies within an RPZ, choose the policy that matches the triggering domain name that appears earlier in the sorting using the DNSSEC canonical DNS name order described in section 6.1 of [RFC4034]. The last labels of domain names are most significant in that ordering. A domain name that

is a parent of another appears before the child. Labels are compared as if they were words in a dictionary so that a label that is a prefix of a second label appears before the second. Characters in labels are sorted by their values as US-ASCII characters except that uppercase letters are treated as if they were lowercase.

Prefix Length

A preceding rule implies that IP policies within an RPZ are preferred to NSIP policies.

Among triggered IP or NSIP policies, use the policy (not the matched IP address) with the longest internal policy zone prefix length. The internal prefix length of an IPv6 address trigger is the numeric value of the first label that defines it as described in Section 4. The internal prefix length of an IPv4 trigger is the numeric value of its first label plus 112. This adjustment makes IPv4 triggers work the same as their equivalent IPv4-compatible IPv6 address triggers. It also tends to favor IPv4 triggers over IPv6 triggers. (See section 2.5.5.1 of [RFC4291] about IPv4-compatible IPv6 addresses.)

Tie Breaker

Given equal internal prefix lengths, use the IP or NSIP policy that matches the first IP address. Addresses with equal trigger internal prefix lengths are ordered by their representations as domain names described in Section 4, including the leading, unadjusted prefix length. Because this tie breaking considers the matched IP addresses instead of the triggered policy rules, the first or least significant label of an IPv6 address is always "128", and the first label of an IPv4 address is always "32".

6. Subscriber Behavior

RPZs must be primary or secondary zones at subscriber recursive resolvers. They can be searched only in a recursive server's own storage, because additional network transactions for DNS resolvers are extremely undesirable.

Response policy zones are loaded in the usual way. For primary zones this may mean loading the contents of a local file into memory, or connecting to a database. For secondary zones this means transferring the zone from the primary server using zone transfer such as IXFR [RFC1995] or AXFR [RFC5936]. It is strongly recommended that all secondary zone transfer relationships be protected with transaction signatures (DNS TSIG) and that real time change notification be enabled using the DNS NOTIFY protocol [RFC1996].

DNS resolvers often have limited or no notion of a DNS zone or zone file. They sometimes have special local zones, but generally have no implementations of IXFR, AXFR, or NOTIFY. Therefore, an external module or daemon that maintains local copies of policy zones can be useful.

7. Producer Behavior

A DNS RPZ producer should make every effort to ensure that incremental zone transfer (IXFR [RFC1995]) rather than full zone transfer (AXFR [RFC5936]) is used to move new policy data toward subscribers. Also, real time zone change notifications (DNS NOTIFY [RFC1996]) should be enabled and tested. DNS RPZ subscribers are "stealth slaves" as described in RFC 1996, and each such server must be explicitly listed in the master server's configuration as necessary to allow zone transfers from the stealth slave, as well to ensure that zone change notifications are sent to it. Because DNS NOTIFY is a lazy protocol, it may be necessary to explicitly trigger the master server's "notify" logic after each change of the DNS RPZ. These operational guidelines are to limit policy data latency, since minimal latency is critical to both prevention of crime and abuse, and to withdrawal of erroneous or outdated policy.

In the data feed for disreputable domains, each addition or deletion or expiration can be handled using DNS UPDATE [RFC2136] to trigger normal DNS NOTIFY and subsequent DNS IXFR activity which can keep the subscribing servers well synchronized to the master RPZ. Alternatively, on some primary name servers (such as ISC BIND) it is possible to generate an entirely new primary RPZ file and have the server compute the differences between each new version and its predecessor. In ISC BIND this option is called "ixfr-from-differences" and is known to be performant even for million-rule DNS RPZ's with significant churn on a minute by minute basis.

It is good operational practice to include test records in each DNS RPZ to help that DNS RPZ's subscribers verify that response policy rewriting is working. For example, a DNS RPZ might include a QNAME policy record for BAD.EXAMPLE.COM and an IP policy record for 192.0.2.1. A subscriber can verify the correctness of their installation by querying for BAD.EXAMPLE.COM which does not exist in real DNS. If an answer is received it will be from the DNS RPZ. That answer will contain an SOA RR denoting the fully qualified name of the DNS RPZ itself.

8. History and Evolution

RPZ was previously described in a technical note from Internet Systems Consortium [ISC-RPZ]. A more up to date description appeared in chapter 6 of the "BIND 9 Administrator Reference Manual" [ISC-ARM] as of 2010.

RPZ was designed by Paul Vixie and Vernon Schryver in 2009. The initial implementation and first patch adding it to BIND were written by Vernon Schryver in late 2009. Patches for various versions of BIND9 including 9.4, 9.6, and 9.7 were distributed from FTP servers at redbarn.org and rhyolite.com starting in 2010.

If all RPZ triggers and actions had been foreseen at the start in 2009, they would probably have been encoded differently. Instead RPZ grew incrementally, and upward compatibility required support of the original encodings. The initial specification or Format 1 contained only QNAME triggers. Changes for Format 2 included adding triggers based on response contents (RDATA), the targets of NS records (NSDNAME), and contents of A and AAAA records that resolve NS records (NSIP). Format 3 included "rpz-passthru" for the PASSTHRU action and wildcards in CNAME targets to synthesize local data.

Today, with a number of commercial RPZ providers with many users and no functional problems with the encodings, any lack of aesthetic appeal is balanced by the ever increasing weight of the installed base. For example, it is impossible to replace the original QNAME trigger encoding NXDOMAIN and NODATA policy action encodings with encodings that involve rpz-* pseudo-TLDs at RPZ providers without breaking the many existing RPZ subscriber installations. The original, deprecated PASSTHRU encoding of a CNAME pointing to the trigger QNAME might still be in use in local, private policy zones, and so it is still recognized by RPZ subscriber implementations.

The initial RPZ idea was only to deny the existence of objectionable domain names, and so there were only QNAME triggers and NXDOMAIN actions. Given that single kind of trigger, encoding it as the owner name of a policy record was clearly best. A CNAME pointing to the root domain (.) is a legal and valid but not generally useful record, and so that became the encoding for the NXDOMAIN action. The encoding of the NODATA action as "CNAME *." followed similar reasoning. Requests for more kinds of triggers and actions required a more general scheme, and so they are encoded as CNAMES with targets in bogus TLDs owner names with DNS labels that start with "rpz_".

9. IANA Considerations

No actions are required from IANA as result of the publication of this document.

10. Security Considerations

RPZ is a mechanism for providing "untruthful" DNS results from recursive servers. Nevertheless, RPZ does not exacerbate the existing vulnerability of recursive servers to falsified DNS data. RPZ merely formalizes and facilitates modifying DNS data on its way from DNS authority servers to clients. However, the use of DNSSEC (see [RFC4033] and [RFC4034]) prevents such changes to DNS data by RPZ.

Therefore, by default, DNS resolvers using RPZ avoid modifying DNS results when DNSSEC signatures are available and are requested by the DNS client. However, when the common "break-dnssec" configuration setting is used, RPZ-using resolvers rewrite responses. They omit DNSSEC RRsets, because the modified responses cannot be verified by DNSSEC signatures. This renders the results invalid according to DNSSEC. In such a case, a querying client which checks DNSSEC will ignore the invalid result, and will effectively be blocked from miscreant domains; this behaviour is functionally similar to that caused by an RPZ NXDOMAIN policy action.

The policy zones might be considered sensitive, because they contain information about miscreants. Like other DNS zones in most situations, RPZs are transferred from sources to subscribers as cleartext vulnerable to observation. However, TSIG transaction signatures [RFC2845] SHOULD be used to authenticate and protect RPZ contents from modification.

Recursive servers using RPZ are often configured to complete recursion even if a policy trigger provides a rewritten answer without needing recursion. This impedes miscreants observing requests from their own authority servers from inferring whether RPZ is in use and whether their RRs are listed. "qname-wait-recurse" is a common configuration switch that controls this behavior. See Section 5.

11. Acknowledgements

The authors gratefully acknowledge the substantial contributed material and editorial scrutiny of Anne Bennett. She directed the reorganization and clarification of the entire document.

Eric Ziegast, Jeroen Massar, and Ben April provided improvements to the document and caught errors.

12. References

12.1. Normative References

[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>.

[RFC1995] Ohta, M., "Incremental Zone Transfer in DNS", RFC 1995, DOI 10.17487/RFC1995, August 1996, <http://www.rfc-editor.org/info/rfc1995>.

[RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, DOI 10.17487/RFC1996, August 1996, <http://www.rfc-editor.org/info/rfc1996>.

[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>.

[RFC2136] Vixie, P., Ed., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, DOI 10.17487/RFC2136, April 1997, <http://www.rfc-editor.org/info/rfc2136>.

[RFC2845] Vixie, P., Gudmundsson, O., Eastlake 3rd, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, DOI 10.17487/RFC2845, May 2000, <http://www.rfc-editor.org/info/rfc2845>.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <http://www.rfc-editor.org/info/rfc4033>.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <http://www.rfc-editor.org/info/rfc4034>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <http://www.rfc-editor.org/info/rfc4291>.

[RFC5936] Lewis, E. and A. Hoenes, Ed., "DNS Zone Transfer Protocol (AXFR)", RFC 5936, DOI 10.17487/RFC5936, June 2010, <http://www.rfc-editor.org/info/rfc5936>.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <http://www.rfc-editor.org/info/rfc5952>.

12.2. Informative References

[ISC-ARM] Internet Systems Consortium, "BIND 9 Administrator Reference Manual, https://ftp.isc.org/isc/bind9/cur/9.10/doc/arm/Bv9ARM.ch06.html#rpz", 2016.

[ISC-RPZ] Vixie, P. and V. Schryver, "DNS Response Policy Zones (DNS RPZ, Format 3), https://ftp.isc.org/isc/dnsrpz/isc-tn-2010-1.txt", 2010.

[ISC-RRL] Vixie, P. and V. Schryver, "DNS Response Rate Limiting (DNS RRL), https://ftp.isc.org/isc/pubs/tn/isc-tn-2012-1.txt", 2012.

Appendix A. Examples

An existing data feed capable of producing an RHSBL can be trivially used to generate a DNS RPZ. If the desired policy is to alias targeted domains to a local walled garden, then for each domain name, generate the following records, one for the name itself and perhaps also one for its subdomains:

```
bad.example.com       CNAME   walled-garden.example.net.
*.bad.example.com     CNAME   walled-garden.example.net.
```

If it is desirable to return NXDOMAIN for each domain (and its subdomains in this example), try this:

```
bad.example.com       CNAME   .
*.bad.example.com     CNAME   .
```

Try this if there are walled gardens for mail versus everything else:

```
bad.example.com       MX      0 wgmail.example.net.
bad.example.com       A       192.0.2.66
*.bad.example.com     MX      0 wgmail.example.net.
*.bad.example.com     A       192.0.2.66
```

An extended example follows:

```
$ORIGIN rpz.example.net.
$TTL 1H
@                     SOA LOCALHOST. named-mgr.example.net. (
                          1 1h 15m 30d 2h) NS LOCALHOST.

; QNAME policy records.
; There are no periods (.) after the relative owner names.
nxdomain.example.com  CNAME   .           ; NXDOMAIN policy
nodata.example.com    CNAME   *.          ; NODATA policy

; redirect to walled garden
bad.example.com       A       10.0.0.1
                      AAAA    2001:db8::1

; do not rewrite OK.EXAMPLE.COM (PASSTHRU)
ok.example.com        CNAME   rpz-passthru.
bzone.example.com     CNAME   garden.example.net.

; redirect X.BZONE.EXAMPLE.COM to
; X.BZONE.EXAMPLE.COM.GARDEN.EXAMPLE.NET
*.bzone.example.com   CNAME   *.garden.example.net.

; rewrite all answers for 192.0.2.0/24 except 192.0.2.1
24.0.2.0.192.rpz-ip   CNAME   .
32.1.2.0.192.rpz-ip   CNAME   rpz-passthru.

; rewrite to NXDOMAIN all responses; for domains for which
; NS.EXAMPLE.COM is an authoritative DNS server or a server
; for a parent) or that have an authoritative server
; in 2001:db8::/32
ns.example.com.rpz-nsdname    CNAME   .
32.zz.db8.2001.rpz-nsip       CNAME   .
```

Authors' Addresses

Paul Vixie
Farsight Security, Inc.

Email: paul@redbarn.org

Vernon Schryver
Rhyolite Software

Email: vjs@rhyolite.com