DNS Extensions (DNSEXT)
Internet-Draft
Updates: 2181 (if approved)
Intended status: Standards Track
Expires: June 18, 2009
December 15, 2008

A. Hubert
Netherlabs Computer Consulting BV.
R. van Mook
Equinix

Measures for making DNS more resilient against forged answers
draft-ietf-dnsext-forgery-resilience-10.txt

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on June 18, 2009.

Copyright Notice

Copyright (c) 2008 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.

Abstract

The current Internet climate poses serious threats to the Domain Name System. In the interim period before the DNS protocol can be secured more fully, measures can already be taken to harden the DNS to make 'spoofing' a recursing nameserver many orders of magnitude harder.

Even a cryptographically secured DNS benefits from having the ability to discard bogus responses quickly, as this potentially saves large amounts of computation.

By describing certain behaviour that has previously not been standardised, this document sets out how to make the DNS more resilient against accepting incorrect responses. This document updates RFC 2181.

Table of Contents

1. Requirements and definitions

1.1. Definitions

This document uses the following definitions:

Client: typically a 'stub-resolver' on an end-user's computer

Resolver: a nameserver performing recursive service for clients, also known as a caching server, or a full service resolver ([RFC1123], paragraph 6.1.3.1)

Stub resolver: a very limited Resolver on a client computer, that leaves the recursing work to a full resolver

Query: a question sent out by a resolver, typically in a UDP packet

Response: the answer sent back by an authoritative nameserver, typically in a UDP packet

Third party: any entity other than the resolver or the intended recipient of a question. The third party may have access to an arbitrary authoritative nameserver, but has no access to packets transmitted by the Resolver or authoritative server

Attacker: malicious third party.

Spoof: the activity of attempting to subvert the DNS process by getting a chosen answer accepted

Authentic response: the correct answer that comes from the right authoritative server

Target domain name: domain for which the attacker wishes to spoof in an answer

Fake data: response chosen by the attacker

1.2. Key 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. Introduction

This document describes several common problems in DNS implementations which, although previously recognized, remain largely unsolved. Besides briefly recapping these problems, this document contains rules that, if implemented, make complying resolvers vastly more resistant to the attacks described. The goal is to make the existing DNS as secure as possible within the current protocol boundaries.

The words below are aimed at authors of resolvers: it is up to operators to decide which nameserver implementation to use, or which options to enable. Operational constraints may override the security concerns described below. However, implementations are expected to allow an operator to enable functionality described in this document.

Almost every transaction on the Internet involves the Domain Name System, which is described in [RFC1034], [RFC1035] and beyond.

Additionally, it has recently become possible to acquire SSL/TLS certificates with no other confirmation of identity than the ability to respond to a verification email sent via SMTP ([RFC2821]) - which generally uses DNS for its routing.

In other words, any party that (temporarily) controls the Domain Name System is in a position to reroute most kinds of Internet transactions, including the verification steps in acquiring an SSL/TLS certificate for a domain. This in turn means that even transactions protected by SSL/TLS could be diverted.

It is entirely conceivable that such rerouted traffic could be used to the disadvantage of Internet users.

These and other developments have made the security and trustworthiness of DNS of renewed importance. Although the DNS community is working hard on finalising and implementing a cryptographically enhanced DNS protocol, steps should be taken to make sure that the existing use of DNS is as secure as possible within the bounds of the relevant standards.

It should be noted that the most commonly used resolvers currently do not perform as well as possible in this respect, making this document of urgent importance.

A thorough analysis of risks facing DNS can be found in [RFC3833].

This document expands on some of the risks mentioned in RFC 3833, especially those outlined in the sections on 'ID Guessing and Query

Prediction' and 'Name Chaining'. Furthermore, it emphasises a number of existing rules and guidelines embodied in the relevant DNS protocol specifications. The following also specifies new requirements to make sure the Domain Name System can be relied upon until a more secure protocol has been standardised and deployed.

It should be noted that even when all measures suggested below are implemented, protocol users are not protected against third parties with the ability to observe, modify or inject packets in the traffic of a resolver.

For protocol extensions that offer protection against these scenarios, see [RFC4033] and beyond.

3. Description of DNS spoofing

When certain steps are taken it is feasible to 'spoof' the current deployed majority of resolvers with carefully crafted and timed DNS packets. Once spoofed, a caching server will repeat the data it wrongfully accepted, and make its clients contact the wrong, and possibly malicious, servers.

To understand how this process works it is important to know what makes a resolver accept a response.

The following sentence in section 5.3.3 of [RFC1034] presaged the present problem:

> The resolver should be highly paranoid in its parsing of responses. It should also check that the response matches the query it sent using the ID field in the response.

DNS data is to be accepted by a resolver if and only if:

1. The question section of the reply packet is equivalent to that of a question packet currently waiting for a response

2. The ID field of the reply packet matches that of the question packet

3. The response comes from the same network address the question was sent to

4. The response comes in on the same network address, including port number, as the question was sent from

In general, the first response matching these four conditions is accepted.

If a third party succeeds in meeting the four conditions before the response from the authentic nameserver does so, it is in a position to feed a resolver fabricated data. When it does so, we dub it an attacker, attempting to spoof in fake data.

All conditions mentioned above can theoretically be met by a third party, with the difficulty being a function of the resolver implementation and zone configuration.

4. Detailed Description of Spoofing Scenarios

The previous paragraph discussed a number of requirements an attacker must match in order to spoof in manipulated (or fake) data. This section discusses the relative difficulties and how implementation defined choices impact the amount of work an attacker has to perform to meet said difficulties.

Some more details can be found in section 2.2 of [RFC3833].

4.1. Forcing a query

Formally, there is no need for a nameserver to perform service except for its operator, its customers or more generally its users. Recently, open recursing nameservers have been used to amplify denial of service attacks.

Providing full service enables the third party to send the target resolver a query for the domain name it intends to spoof. On receiving this query, and not finding the answer in its cache, the resolver will transmit queries to relevant authoritative nameservers. This opens up a window of opportunity for getting fake answer data accepted.

Queries may however be forced indirectly, for example by inducing a mail server to perform DNS lookups.

Some operators restrict access by not recursing for unauthorised IP addresses, but only respond with data from the cache. This makes spoofing harder for a third party as it cannot then force the exact moment a question will be asked. It is still possible however to determine a time range when this will happen, because nameservers helpfully publish the decreasing TTL of entries in the cache, which indicate from which absolute time onwards a new query could be sent to refresh the expired entry.

The time to live of the target domain name's RRSets determines how often a window of opportunity is available, which implies that a short TTL makes spoofing far more viable.

Note that the attacker might very well have authorised access to the target resolver by virtue of being a customer or employee of its operator. In addition, access may be enabled through the use of reflectors as outlined in [RFC5358].

4.2. Matching the question section

DNS packets, both queries and responses, contain a question section. Incoming responses should be verified to have a question section that is equivalent to that of the outgoing query.

4.3. Matching the ID field

The DNS ID field is 16 bits wide, meaning that if full use is made of all these bits, and if their contents are truly random, it will require on average 32768 attempts to guess. Anecdotal evidence suggests there are implementations utilising only 14 bits, meaning on average 8192 attempts will suffice.

Additionally, if the target nameserver can be forced into having multiple identical queries outstanding, the 'Birthday Attack' phenomenon means that any fake data sent by the attacker is matched against multiple outstanding queries, significantly raising the chance of success. Further details in Section 5.

4.4. Matching the source address of the authentic response

It should be noted that meeting this condition entails being able to transmit packets on behalf of the address of the authoritative nameserver. While two Best Current Practice documents ([RFC2827] and [RFC3013] specifically) direct Internet access providers to prevent their customers from assuming IP addresses that are not assigned to them, these recommendations are not universally (nor even widely) implemented.

Many zones have two or three authoritative nameservers, which make matching the source address of the authentic response very likely with even a naive choice having a double digit success rate.

Most recursing nameservers store relative performance indications of authoritative nameservers, which may make it easier to predict which nameserver would originally be queried - the one most likely to respond the quickest.

Generally, this condition requires at most two or three attempts before it is matched.

4.5. Matching the destination address and port of the authentic response

Note that the destination address of the authentic response is the source address of the original query.

The actual address of a recursing nameserver is generally known; the port used for asking questions is harder to determine. Most current resolvers pick an arbitrary port at startup (possibly at random) and use this for all outgoing queries. In quite a number of cases the source port of outgoing questions is fixed at the traditional DNS assigned server port number of 53.

If the source port of the original query is random, but static, any authoritative nameserver under observation by the attacker can be used to determine this port. This means that matching this conditions often requires no guess work.

If multiple ports are used for sending queries, this enlarges the effective ID space by a factor equal to the number of ports used.

Less common resolving servers choose a random port per outgoing query. If this strategy is followed, this port number can be regarded as an additional ID field, again containing up to 16 bits.

If the maximum ports range is utilized, on average, around 32256 source ports would have to be tried before matching the source port of the original query as ports below 1024 may be unavailable for use, leaving 64512 options.

It is in general safe for DNS to use ports in the range 1024-49152 even though some of these ports are allocated to other protocols. DNS resolvers will not be able to use any ports that are already in use. If a DNS resolver uses a port it will release that port after a short time and migrate to a different port. Only in the case of high volume resolver is it possible that an application wanting a particular UDP port suffers a long term block-out.

It should be noted that a firewall will not prevent the matching of this address, as it will accept answers that (appear) to come from the correct address, offering no additional security.

4.6. Have the response arrive before the authentic response

Once any packet has matched the previous four conditions (plus possible additional conditions), no further responses are generally accepted.

This means that the third party has a limited time in which to inject its spoofed response. For calculations we will assume a window in order of at most 100ms (depending on the network distance to the authentic authoritative nameserver).

This time period can be far longer if the authentic authoritative

nameservers are (briefly) overloaded by queries, perhaps by the attacker.

5. Birthday attacks

The so called birthday paradox implies that a group of 23 people suffices to have a more than even chance of having two or more members of the group share a birthday.

An attacker can benefit from this exact phenomenon if it can force the target resolver to have multiple equivalent (identical QNAME, QTYPE and QCLASS) outstanding queries at any one time to the same authoritative server.

Any packet the attacker sends then has a much higher chance of being accepted because it only has to match any of the outstanding queries for that single domain. Compared to the birthday analogy above, of the group composed of queries and responses, the chance of having any of these share an ID rises quickly.

As long as small numbers of queries are sent out, the chance of successfully spoofing a response rises linearly with the number of outstanding queries for the exact domain and nameserver.

For larger numbers this effect is less pronounced.

More details are available in US-CERT [vu-457875].

6. Accepting only in-domain records

Responses from authoritative nameservers often contain information that is not part of the zone for which we deem it authoritative. As an example, a query for the MX record of a domain might get as its responses a mail exchanger in another domain, and additionally the IP address of this mail exchanger.

If accepted uncritically, the resolver stands the chance of accepting data from an untrusted source. Care must be taken to only accept data if it is known that the originator is authoritative for the QNAME or a parent of the QNAME.

One very simple way to achieve this is to only accept data if it is part of the domain the query was for.

7. Combined difficulty

Given a known or static destination port, matching ID field, source and destination address requires on average in the order of 2 * 2^15 = 65000 packets, assuming a zone has 2 authoritative nameservers.

If the window of opportunity available is around 100ms, as assumed above, an attacker would need to be able to briefly transmit 650000 packets/s to have a 50% chance to get spoofed data accepted on the first attempt.

A realistic minimal DNS response consists of around 80 bytes, including IP headers, making the packet rate above correspond to a respectable burst of 416Mb/s.

As of mid-2006, this kind of bandwidth was not common but not scarce either, especially among those in a position to control many servers.

These numbers change when a window of a full second is assumed, possibly because the arrival of the authentic response can be prevented by overloading the bonafide authoritative hosts with decoy queries. This reduces the needed bandwidth to 42 Mb/s.

If in addition the attacker is granted more than a single chance and allowed up to 60 minutes of work on a domain with a time to live of 300 seconds, a meagre 4Mb/s suffices for a 50% chance at getting fake data accepted. Once equipped with a longer time, matching condition 1 mentioned above is straightforward - any popular domain will have been queried a number of times within this hour, and given the short TTL, this would lead to queries to authoritative nameservers, opening windows of opportunity.

7.1. Symbols used in calculation

Assume the following symbols are used:

I: Number distinct IDs available (maximum 65536)

P: Number of ports used (maximum around 64000 as ports under 1024 are not always available, but often 1)

N: Number of authoritative nameservers for a domain (averages around 2.5)

F: Number of 'fake' packets sent by the attacker

R: Number of packets sent per second by the attacker

W: Window of opportunity, in seconds. Bounded by the response time of the authoritative servers (often 0.1s)

D: Average number of identical outstanding queries of a resolver (typically 1, see Section 5)

A: Number of attempts, one for each window of opportunity

7.2. Calculation

The probability of spoofing a resolver is equal to the amount of fake packets that arrive within the window of opportunity, divided by the size of the problem space.

When the resolver has 'D' multiple identical outstanding queries, each fake packet has a proportionally higher chance of matching any of these queries. This assumption only holds for small values of 'D'.

In symbols, if the probability of being spoofed is denoted as P_s:

```
           D * F
P_s =    ---------
         N * P * I
```

It is more useful to reason not in terms of aggregate packets but to convert to packet rate, which can easily be converted to bandwidth if needed.

If the Window of opportunity length is 'W' and the attacker can send 'R' packets per second, the number of fake packets 'F' that are candidates to be accepted is:

```
                       D * R * W
F = R * W  ->   P_s  = ---------
                       N * P * I
```

Finally, to calculate the combined chance 'P_cs' of spoofing over a chosen time period 'T', it should be realised that the attacker has a new window of opportunity each time the TTL 'TTL' of the target domain expires. This means that the number of attempts 'A' is equal to 'T / TTL'.

To calculate the combined chance of at least one success, the following formula holds:

$$P\_cs = 1 - (1 - P\_s)^{A} = 1 - \left(1 - \frac{D * R * W}{N * P * I} \right)^{(T / TTL)}$$

When common numbers (as listed above) for D, W, N, P and I are inserted, this formula reduces to:

$$P\_cs = 1 - \left(1 - \frac{R}{1638400} \right)^{(T / TTL)}$$

From this formula it can be seen that, if the nameserver implementation is unchanged, only raising the TTL offers protection. Raising N, the number of authoritative nameservers, is not feasible beyond a small number.

For the degenerate case of a zero-second TTL, a window of opportunity opens for each query sent, making the effective TTL equal to 'W' above, the response time of the authoritative server.

This last case also holds for spoofing techniques which do not rely on TTL expiry, but use repeated and changing queries.

8. Discussion

The calculations above indicate the relative ease with which DNS data can be spoofed. For example, using the formula derived earlier on an RRSet with a 3600 second TTL, an attacker sending 7000 fake response packets/s (a rate of 4.5Mb/s), stands a 10% chance of spoofing a record in the first 24 hours, which rises to 50% after a week.

For an RRSet with a TTL of 60 seconds, the 10% level is hit after 24 minutes, 50% after less than 3 hours, 90% after around 9 hours.

For some classes of attacks, the effective TTL is near zero, as noted above.

Note that the attacks mentioned above can be detected by watchful server operators - an unexpected incoming stream of 4.5mbit/s of packets might be noticed.

An important assumption however in these calculations is a known or static destination port of the authentic response.

If that port number is unknown and needs to be guessed as well, the problem space expands by a factor of 64000, leading the attacker to need in excess of 285Gb/s to achieve similar success rates.

Such bandwidth is not generally available, nor expected to be so in the foreseeable future.

Note that some firewalls may need reconfiguring if they are currently setup to only allow outgoing queries from a single DNS source port.

8.1. Repetitive spoofing attempts for a single domain name

Techniques are available to use an effectively infinite number of queries to achieve a desired spoofing goal. In the math above, this reduces the effective TTL to 0.

If such techniques are employed, using the same 7000 packets/s rate mentioned above, and using 1 source port, the spoofing chance rises to 50% within 7 seconds.

If 64000 ports are used, as recommended in this document, using the same query rate, the 50% level is reached after around 116 hours.

9. Forgery countermeasures

9.1. Query matching rules

A resolver implementation MUST match responses to all of the following attributes of the query:

- o Source address against query destination address
- o Destination address against query source address
- o Destination port against query source port
- o Query ID
- o Query name
- o Query class and type

before applying DNS trustworthiness rules (see [RFC2181], section 5.4.1).

A mismatch and the response MUST be considered invalid.

9.2. Extending the Q-ID space by using ports and addresses

Resolver implementations MUST:

- o Use an unpredictable source port for outgoing queries from the range of available ports (53, or 1024 and above) that is as large as possible and practicable;
- o Use multiple different source ports simultaneously in case of multiple outstanding queries;
- o Use an unpredictable query ID for outgoing queries, utilizing the full range available (0-65535)

Resolvers that have multiple IP addresses SHOULD use them in an unpredictable manner for outgoing queries.

Resolver implementations SHOULD provide means to avoid usage of certain ports.

Resolvers SHOULD favour authoritative nameservers with which a trust relation has been established; Stub-resolvers SHOULD be able to use TSIG ([RFC2845]) or IPsec ([RFC4301]) when communicating with their recursive resolver

In case a cryptographic verification of response validity is available (TSIG, SIG(0)), resolver implementations MAY waive above rules, and rely on this guarantee instead.

Proper unpredictability can be achieved by employing a high quality (pseudo-)random generator, as described in [RFC4086].

9.2.1. Justification and Discussion

Since an attacker can force a full DNS resolver to send queries to the attacker's own name servers, any constant or sequential state held by such a resolver can be measured, and it must not be trivially easy to reverse engineer the resolver's internal state in a way that allows low-cost high-accuracy prediction of future state.

A full DNS resolver with only one or a small number of upstream-facing endpoints is effectively using constants for IP source address and UDP port number, and these are very predictable by potential attackers, and must therefore be avoided.

A full DNS resolver that uses a simple increment to get its next DNS query ID is likewise very predictable and so very spoofable.

Finally, weak random number generators have been shown to expose their internal state, such that an attacker who witnesses several sequential "random" values can easily predict the next ones. A crypto-strength random number generator is one whose output cannot be predicted no matter how many successive values are witnessed.

9.3. Spoof detection and countermeasure

If a resolver detects that an attempt is being made to spoof it, perhaps by discovering that many packets fail the criteria as outlined above, it MAY abandon the UDP query and re-issue it over TCP. TCP, by the nature of its use of sequence numbers, is far more resilient against forgery by third parties.

10. Security Considerations

This document provides clarification of the DNS specification to decrease the probability that DNS responses can be successfully forged. Recommendations found above should be considered complementary to possible cryptographical enhancements of the domain name system, which protect against a larger class of attacks.

This document recommends the use of UDP source port number randomization to extend the effective DNS transaction ID beyond the available 16 bits.

A resolver that does not implement the recommendations outlined above can easily be forced to accept spoofed responses, which in turn are passed on to client computers - misdirecting (user) traffic to possibly malicious entities.

This document directly impacts the security of the Domain Name System, implementers are urged to follow its recommendations.

Most security considerations can be found in Section 4 and Section 5, while proposed countermeasures are described in Section 9.

For brevity's sake, in lieu of repeating the security considerations references, the reader is referred to these sections.

Nothing in this document specifies specific algorithms for operators to use; it does specify algorithms implementations SHOULD or MUST support.

It should be noted that the effects of source port randomization may be dramatically reduced by NAT devices which either serialize or limit in volume the UDP source ports used by the querying resolver.

DNS recursive servers sitting behind at NAT or a statefull firewall may consume all available NAT translation entries/ports when operating under high query load. Port randomization will cause translation entries to be consumed faster than with fixed query port

. To avoid this NAT boxes and statefull firewalls can/should purge outgoing DNS query translation entries 10-17 seconds after the last outgoing query on that mapping was sent. [RFC4787] compliant devices need to treat UDP messages with port 53 differently than most other UDP protocols.

To minimize the potential that port/state exhaustion attacks can be staged from the outside, it is recommended that services which generate number of DNS queries for each connection, should be rate

limited. This applies in particular to e-mail servers

11. IANA Considerations

This document does not make any assignments and has no actions for IANA.

12. Acknowledgments

Source port randomisation in DNS was first implemented and possibly invented by Dan. J. Bernstein.

Although any mistakes remain our own, the authors gratefully acknowledge the help and contributions of:

Stephane Bortzmeyer,

Alfred Hoenes,

Peter Koch,

Sean Leach,

Norbert Sendetzky,

Paul Vixie,

Florian Weimer,

Wouter Wijngaards,

Dan Wing

13. References

13.1. Normative References

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

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

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

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

[RFC2821] Klensin, J., "Simple Mail Transfer Protocol", RFC 2821, April 2001.

[RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, May 2000.

[RFC2845] Vixie, P., Gudmundsson, O., Eastlake, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, May 2000.

[RFC3013] Killalea, T., "Recommended Internet Service Provider Security Services and Procedures", BCP 46, RFC 3013, November 2000.

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

[RFC4086] Eastlake, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, June 2005.

13.2. Informative References

[RFC1123] Braden, R., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, October 1989.

[RFC3833] Atkins, D. and R. Austein, "Threat Analysis of the Domain Name System (DNS)", RFC 3833, August 2004.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4787] Audet, F. and C. Jennings, "Network Address Translation (NAT) Behavioral Requirements for Unicast UDP", BCP 127, RFC 4787, January 2007.

[RFC5358] Damas, J. and F. Neves, "Preventing Use of Recursive Nameservers in Reflector Attacks", BCP 140, RFC 5358, October 2008.

[vu-457875]
United States CERT, "Various DNS service implementations generate multiple simultaneous queries for the same resource record", VU 457875, November 2002.

Authors' Addresses

Bert Hubert
Netherlabs Computer Consulting BV.
Braillelaan 10
Rijswijk (ZH) 2289 CM
The Netherlands

Email: bert.hubert@netherlabs.nl

Remco van Mook
Equinix
Auke Vleerstraat 1
Enschede 7521 PE
The Netherlands

Email: remco@eu.equinix.com