DNS Extensions (DNSEXT)
Internet-Draft
Updates: 1035 (if approved)
Intended status: Standards Track
Expires: January 2, 2008

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

July 2007

Measures for making DNS more resilient against forged answers
draft-ietf-dnsext-forgery-resilience-01.txt

Status of this Memo

Copyright Notice

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 make 'spoofing' a recursing nameserver many orders of magnitude harder.

Even a cryptographically secured DNS benefits from having the ability to discard bogus answers 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 answers. This document updates RFC1034.

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

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

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

Third party: any host other than the resolver or the intended recipient of a question. The third party may have access to a random 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 answer: the answer that would be accepted if no third party interferes

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

Fake data: answer chosen by the attacker

TBD: Do we need to talk about stub resolvers? Does this draft apply to them?

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 RFC contains rules that, if implemented, make complying resolvers vastly more resistant to the attacks described.

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 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 certificate for a domain. This in turn means that even transactions protected by SSL 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 resolver currently does 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 STDs and RFCs. 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 intercept, change or inject packets sent to the resolver.

For protocol extensions under development 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 caching 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 (and more specifically a caching server) accept an answer.

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 accepted by a resolver if and only if:

1. The question section of the reply packet is identical to that of a question packet currently waiting for an answer

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

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

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

5. It is the first answer to match the previous four conditions.

Note that the fifth condition can strictly speaking be derived from the first. It is included for clarity reasons only.

If a third party succeeds in meeting the first four conditions before the answer from the authentic answer 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, with the difficulty being a function of the resolver implementation and zone configuration.

4. Details

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. Matching the question

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.

In spite of this, many resolvers perform at least partial service for the whole world. This is partially out of lack of concern, and is reminiscent of the open relay SMTP service the net enjoyed up to the early 1990s. Some access providers may serve so many subnets that it is hard to enumerate these all in the DNS configuration.

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

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' 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.

4.2. 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 questions outstanding, the 'Birthday Attack' phenomenon means that any fake data sent by the attacker is matched against multiple outstanding questions, significantly raising the chance of success. Further details in Section 5.

4.3. Matching the source address of the authentic answer

Most domains have two or three authoritative nameservers, which make matching the source address of the authentic answer 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.

It should be noted that meeting this condition entails being able to transmit packets on behalf of the address of the authoritative nameserver. While several important 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.

4.4. Matching the destination address of the authentic answer

Note that the destination address of the authentic answer is the source address of the original question.

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

If the source port of the original question 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 questions, this enlarges the effective address space by a factor equal to the number of ports used.

Less common resolving servers choose a random port per outgoing question. 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 32128 source ports would have to be tried before matching the source port of the original question as ports below 1024 may be unavailable for use, leaving 64512 options.

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.

4.5. Have the answer arrive before the authentic answer

Once any packet has matched the previous four conditions, no further answers should be accepted.

This means that the third party has a limited time in which to inject its spoofed answer, typically in the order of at most 100ms.

This time period can be far longer if the authentic authoritative nameservers are (briefly) overloaded by queries, perhaps by the attacker.

5. Birthday attacks

A curious mathematical phenomenon means that a group of 22 people suffices to have a more than even chance at having two or more members of the group share a birthday.

An attacker can benefit from this phenomenon if it can force the target resolver to have multiple outstanding questions at any one time for the same domain 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 questions and answers, the chance of having any of these share an ID rises quickly.

As long as small numbers of questions are sent out, the chance of successfully spoofing an anwers rises linearly with the number of outstanding questions 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-zone answers

Answers 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 answer 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 that data.

One very simple way to achieve this is to only accept data if it is part of the domain we asked the question 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 domain 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 answer 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 answer can be prevented by overloading the bonafide authoritative hosts with decoy questions. This reduces the needed bandwith 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 questions 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 questions 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 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 questions, each fake packet has a proportionally higher chance of matching any of these questions. This assumption only holds for small values of 'D'.

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

$$P_s = \frac{D * F}{N * P * I}$$

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:

$$F = R * W \rightarrow P_s = \frac{D * R * W}{N * P * I}$$

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 question asked, making the effective TTL equal to 'W' above, the response time of the authoritative server.

8. Discussion

The calculations above indicate the relative ease with which DNS data can be spoofed. For example, using the formula derived earlier on a domain with a 3600 second TTL, an attacker sending 7000 fake answer 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 a domain 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.

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 answer.

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.

9. Countermeasures

NOTE: This section is expected to change, and is very much open to discussion!

Implementations MUST be able to send queries from ANY UDP port available to it.

Implementations SHOULD use good random source to select a Query ID for next query

Implementations SHOULD NOT use UDP source ports <1024 for sending queries

Implementations MUST use an as large as possible pool of UDP source ports for sending queries

Implementations SHOULD be configurable to use one or multiple ports for queries.

Implementations MAY be configurable to use one or more addresses for queries

Implementations MUST suppress multiple simultaneous identical queries to the SAME server.

Implementations MUST match answers to the following

- Remote address
- Local address
- Query port
- Query ID
- Question

before applying DNS credibility rules.

The document can not require the use of either multiple ports or addresses as that is an operational issue and should be addressed in a separate document in DNSOP.

NOTE! A previous version of requirements is listed below as an inspiration to further discussions:

Given the above, a resolver MAY/SHOULD/MUST:

- Use an unpredictable source port for outgoing queries from a range (53, or > 1024) of ports that is as large as possible
- Make use of all 16 bits of the ID field
- Assure that its choices of port and ID cannot be predicted by an attacker having knowledge of its (pseudo-)random generator
- Not have multiple equivalent questions outstanding to any authoritative server, unless all with identical ID and source port

A resolver SHOULD offer diagnostics that enable the operator to determine a spoofing attempt is under way.

Operators SHOULD attempt to restrict recursing service, either full or partial, to authorised users.

A resolver MAY use heuristics to detect an excess of unacceptable answers and take measures if it believes an attempt is made to spoof it.

Futhermore, zone operators are urged not to configure the Time To Live of domains to be lower than realistically needed for proper operations.

10. Security Considerations

This document directly impacts the operational security of the Domain Name System, readers are urged to implement its recommendations.

TBD!

11. Acknowledgements

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,

Sean Leach,

Norbert Sendetzky

12. 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.

[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.

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

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

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

[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
Virtu
Auke Vleerstraat 1
Enschede 7521 PE
The Netherlands

Email: remco@virtu.nl

Full Copyright Statement

Intellectual Property

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).