DNS privacy problem statement
draft-bortzmeyer-perpass-dns-privacy-01

Abstract

This document describes the privacy issues associated with the use of the DNS by Internet users. It is intended to be a problem statement and it does not prescribe solutions.

Discussions of the document should take place on the perpass mailing list [perpass]

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 May 20, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

The Domain Name System is specified in [RFC1034] and [RFC1035]. It is one of the most important infrastructure components of the Internet and one of the most often ignored or misunderstood. Almost every activity on the Internet starts with a DNS query (and often several). Its use has many privacy implications and we try to give here a comprehensive and correct list.
Let us start with a small reminder of the way the DNS works (with some simplifications). A client, the stub resolver, issues a DNS query to a server, the resolver (also called caching resolver or full resolver). For instance, the query is "What are the AAAA records for www.example.com?". AAAA is the qtype (Query Type) and www.example.com the qname (Query Name). To get the answer, the resolver will query first the root nameservers, which will, most of the times, send a referral. Here, the referral will be to .com nameservers. In turn, they will send a referral to the example.com nameservers, which will provide the answer. The root name servers, the name servers of .com and those of example.com are called authoritative name servers. It is important, when analyzing the privacy issues, to remember that the question asked to all these name servers is always the original question, not a derived question. Unlike what many "DNS for dummies" articles say, the question sent to the root name servers is "What are the AAAA records for www.example.com?", not "What are the name servers of .com?". So, the DNS leaks more information than it should.

Because the DNS uses caching heavily, not all questions are sent to the authoritative name servers. If the stub resolver, a few seconds later, ask to the resolver "What are the SRV records of _xmpp-server._tcp.example.com?", the resolver will remember that it knows the name servers of example.com and will just query them, bypassing the root and .com. Because there is typically no caching in the stub resolver, the resolver, unlike the authoritative servers, sees everything.

Almost all the DNS queries are today sent over UDP, which have practical consequences if someone thinks of encrypting this traffic.

I should be noted to that DNS resolvers often forward requests to a bigger machine, with a larger and more shared cache, the forwarders. From the point of view of privacy, forwarders are like resolvers, except that the caching in the resolver before them decreases the amount of data they can see.

We will use here the terminology of \[RFC6973\].

2. Requirements notation

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 \[RFC2119\].

3. Risks
This draft is limited to the study of privacy risks for the end-user (the one performing DNS requests). Privacy risks for the holder of a zone (the risk that someone gets the data) are discussed in [RFC5936] and in [I-D.koch-perpass-dns-confidentiality]. Non-privacy risks (such as cache poisoning) are out of scope.

3.1. Data in the DNS request

The DNS request includes many fields but two seem specially relevant for the privacy issues, the qname and the source IP address. "source IP address" is used in a loose sense of "source IP address + may be source port", because the port is also in the request and can be used to sort out several users sharing an IP address (CGN for instance).

The qname is the full question sent by the user. It gives information about what the user does ("What are the MX records of example.net?" means he probably wants to send email to someone at example.net, which may be a domain used by only a few persons and therefore very revealing). Some qnames are more sensitive than others. For instance, querying the A record of google-analytics.com reveals very little (everybody visits Web sites which use Google Analytics) but querying the A record of www.verybad.example where verybad.example is the domain of an illegal or very offensive organization may create more problems for the user. Another example if the qname embedding the software you use. For instance, some BitTorrent clients query a SRV record for _bittorrent-tracker._tcp.domain.example.

For the communication between the stub resolver and the resolver, the source IP address is the one of the user’s machine. Therefore, all the issues and warnings about collection of IP addresses apply here. For the communication between the resolver and the authoritative name servers, the source IP address has a different meaning, it does not have the same status as the source address in a HTTP connection. It is now the IP address of the resolver which, in a way "hides" the real user. However, it does not always work. Sometimes [I-D.vandergaast-edns-client-subnet] is used. Sometimes the end user has a personal resolver on her machine. In that case, the IP address is as sensitive as it is for HTTP.

3.2. On the wire

DNS traffic can be seen by an eavesdropper like any other traffic. It is typically not encrypted. (DNSSEC, specified in [RFC4033] explicitely excludes confidentiality from its goals.) So, if an initiator starts a HTTPS communication with a recipient, while the HTTP traffic will be encrypted, the DNS exchange prior to it won’t be. When the other protocols will become more or more privacy-aware
and secured against surveillance, the DNS risks to become "the weakest link" in privacy.

What also makes the DNS traffic different is that it may take a different path than the communication between the initiator and the recipient. For instance, an eavesdropper may be unable to tap the wire between the initiator and the recipient but may have access to the wire going to the resolver, or to the authoritative name servers.

The best place, from an eavesdropper’s point of view, is clearly between the stub resolvers and the resolvers, because you are not limited by DNS caching.

The attack surface between the stub resolver and the rest of the world can vary widely depending upon how the end user’s computer is configured. By order of increasing attack surface:

The resolver can be on the end user’s computer. In (currently) a small number of cases, individuals may choose to operate their own DNS resolver on their local machine. In this case the attack surface for the stub resolver to caching resolver connection is limited to that single machine.

The resolver can be in the IAP (Internet Access Provider) premises. For most residential users and potentially other networks the typical case is for the end user’s computer to be configured (typically automatically through DHCP) with the addresses of the DNS resolver at the IAP. The attack surface for on-the-wire attacks is therefore from the end user system across the local network and across the IAP network to the IAP’s resolvers.

The resolver may also be at the local network edge. For many/most enterprise networks and for some residential users the caching resolver may exist on a server at the edge of the local network. In this case the attack surface is the local network. Note that in large enterprise networks the DNS resolver may not be located at the edge of the local network but rather at the edge of the overall enterprise network. In this case the enterprise network could be thought of as similar to the IAP network referenced above.

The resolver can be a public DNS service. Some end users may be configured to use public DNS resolvers such as those operated by Google Public DNS or OpenDNS. The end user may have configured their machine to use these DNS resolvers themselves – or their IAP may choose to use the public DNS resolvers rather than operating their own resolvers. In this case the attack surface is the entire public Internet between the end user’s connection and the public DNS service.
3.3. In the servers

Using the terminology of [RFC6973], the DNS servers (resolvers and authoritative servers) are enablers: they facilitate communication between an initiator and a recipient without being directly in the communications path. As a result, they are often forgotten in risk analysis. But, to quote again [RFC6973], "Although [...] enablers may not generally be considered as attackers, they may all pose privacy threats (depending on the context) because they are able to observe, collect, process, and transfer privacy-relevant data." In [RFC6973] parlance, enablers become observers when they start collecting data.

Many programs exist to collect and analyze DNS data at the servers. From the "query log" of some programs like BIND, to tcpdump and more sophisticated programs like PacketQ and DNSmezzo. The organization managing the DNS server can use this data itself or it can be part of a surveillance program like PRISM [prism] and pass data to an outside attacker.

Sometimes, these data are kept for a long time and/or distributed to third parties, for research purposes [dit1], for security analysis, or for surveillance tasks. Also, there are observation points in the network which gather DNS data and then make it accessible to third-parties for research or security purposes ("passive DNS [passive-dns]").

3.3.1. In the resolvers

The resolvers see the entire traffic since there is typically no caching before them. They are therefore well situated to observe the traffic. To summary: your resolver knows a lot about you. The resolver of a big IAP, or a big public resolver can collect data from many users. You may get an idea of the data collected by reading the privacy policy of a big public resolver [1].

3.3.2. In the authoritative name servers

Unlike the resolvers, they are limited by caching. They see only a part of the requests. For aggregated statistics ("what is the percentage of LOC queries"), it is sufficient but it may prevent an observer to observe everything. Nevertheless, the authoritative name servers see a random part of the traffic and it may be sufficient to defeat some privacy expectations.
Also, the end user has typically some legal/contractual link with the resolver (he chose the IAP, or he chose to use a given public resolver) while he is often not even aware of the role of the authoritative name servers and their observation abilities.

It is an interesting question whether the privacy issues are bigger in the root or in a big TLD. The root sees the traffic for all the TLD (and the huge amount of traffic for non-existing TLD) but a big TLD has less caching before it.

As noted before, using a local resolver or a resolver close to the machine decreases the attack surface for an on-the-wire eavesdropper. But it may decrease privacy against an observer located on an authoritative name server since the authoritative name server will see the IP address of the end client, and not the address of a big resolver shared by many users. This is no longer true if [I-D.vandergaast-edns-client-subnet] is used because, in this case, the authoritative name server sees the original IP address.

As of today, all the instances of one root name server, L-root, receive together around 20 000 queries per second. While most of it is junk (errors on the TLD name), it gives an idea of the amount of big data which pours into name servers.

Many domains, including TLD, are partially hosted by third-party servers, sometimes in a different country. The contracts between the domain manager and these servers may or may not take privacy into account. But it may be surprising for an end-user that requests to a given ccTLD may go to servers managed by organisations outside of the country.

3.3.3. Rogue servers

A rogue DHCP server can send you to a rogue resolver. Most of the times, it seems to be done to divert traffic, by providing lies for some domain names. But it could be used just to capture the traffic and gather information about you. Same thing for malwares like DNSchanger[dnschanger] which changes the resolver in the machine’s configuration.

4. Actual "attacks"

Many research papers about malware detection use DNS traffic to detect "abnormal" behaviour that can be traced back to the activity of malware on infected machines. Yes, this reasearch was done for the good but, technically, it is a privacy attack and it demonstrates the power of the observation of DNS traffic. See [dns-footprint], [dagon-malware] and [darkreading-dns].
5. Legalities

To our knowledge, there are no specific privacy laws for DNS data. Interpreting general privacy laws like [data-protection-directive] (European Union) in the context of DNS traffic data is not an easy task and it seems there is no court precedent here.

6. Possible technical solutions

We mention here only the solutions that could be deployed in the current Internet. Disruptive solutions, like replacing the DNS with a completely new resolution protocol, are interesting but are kept for a future work. Remember that the focus of this document is on describing the threats, not in detailing solutions. This section is therefore non-normative and is NOT a technical specification of solutions. For the same reason, there are not yet actual recommendations in this document.

Raising seriously the bar against the eavesdropper will require SEVERAL actions. Not one is decisive by itself but, together, they can have an effect. The most important suggested here are:

- qname minimization,
- encryption of DNS traffic,
- padding (sending random queries from time to time).

We detail some of these actions later, classified by the kind of observer (on the wire, in a server, etc). Some actions will help against several kinds of observers. For instance, padding, sending gratuitous queries from time to time (queries where you’re not interested in the replies, just to disturb the analysis), is useful against all sorts of observers. It is a costly technique, because it increases the traffic on the network but it seriously blurs the picture for the observer.

6.1. On the wire

6.1.1. Reducing the attack surface

See Section 6.2.1 since the solution described there apply against on-the-wire eavesdropping as well as against observation by the resolver.

6.1.2. Encrypting the DNS traffic
To completely defeat an eavesdropper, there is only one solution: encryption. But, from the end user point of view, even if you check that your communication between your stub resolver and the resolver is encrypted, you have no way to ensure that the communication between the resolver and the authoritative name servers will be. There are two different cases, communication between the stub resolver and the resolver (no caching but only two parties so solutions which rely on an agreement may work) and communication between the resolver and the authoritative servers (less data because of caching, but many parties involved, so any solution has to scale well). Encrypting the "last mile", between the user's stub resolver and the resolver may be sufficient since the biggest danger for privacy is between the stub resolver and the resolver, because there is no caching involved there.

The only encryption mechanism available for DNS which is today an IETF standard is IPsec in ESP mode. It's deployment in the wide Internet is very limited, for reasons which are out of scope here. Still, it may be a solution for "the last mile" and, indeed, many VPN solutions use it this way, encrypting the whole traffic, including DNS to the safe resolver. In the IETF standards, a possible candidate could be DTLS [RFC6347]. It enjoyed very little actual deployment and its interaction with the DNS has never been considered, studied or of course implemented. There are also non standard encryption techniques like DNScrypt [dnscrypt] for the stub resolver <-> resolver communication or DNScurve [dnscurve] for the resolver <-> authoritative server communication. It seems today that the possibility of massive encryption of DNS traffic is very remote.

Another solution would be to use more TCP for the queries, together with TLS [RFC5246]. DNS can run over TCP and it provides a good way to leverage the software and experience of the TLS world. There have been discussions to use more TCP for the DNS, in light of reflection attacks (based on the spoofing of the source IP address, which is much more difficult with TCP). For instance, a stub resolver could open a TCP connection with the resolver at startup and keep it open to send queries and receive responses. The server would of course be free to tear down these connections at will (when it is under stress, for instance) and the client could reestablish them when necessary. Remember that TLS sessions can survive TCP connections so there is no need to restart the TLS negotiation each time. This DNS-over-TLS-over-TCP is already implemented in the Unbound resolver. It is safe only if pipelining multiple questions over the same channel. Name compression should also be disabled, or CRIME-style [crime] attacks can apply.

Encryption alone does not guarantee perfect privacy, because of the available metadata. For instance, the size of questions and
responses, even encrypted, provide hints about what queries have been sent. (DNScrypt uses random-length padding, and a 64 bytes block size, to limit this risk, but this raises other issues, for instance during amplification attacks.) Observing the periodicity of encrypted questions/responses also discloses the TTL, which is yet another hint about the queries. Non-cached responses are disclosing the RTT between the resolver and authoritative servers. This is a very useful indication to guess where authoritative servers are located. Web pages are made of many resources, leading to multiple requests, whose number and timing fingerprint which web site is being browsed. So, observing encrypted traffic is not enough to recover any plaintext queries, but is enough to answer the question "is one of my employees browsing Facebook?". Finally, attackers can perform a denial-of-service attack on possible targets, check if this makes a difference on the encrypted traffic they observe, and infer what a query was.

6.2. In the servers

6.2.1. In the resolvers

It does not seem there is a possible solution against a leaky resolver. A resolver has to see the entire DNS traffic in clear.

The best approach to limit the problem is to have local resolvers whose caching will limit the leak. Local networks should have a local caching resolver (even if it forwards the unanswered questions to a forwarder) and individual laptops can have their very own resolver, too.

One mechanism to potentially mitigate on the wire attacks between stub resolvers and caching resolvers is to determine if the network location of the caching resolver can be moved closer to the end user’s computer (reducing the attack surface). As noted earlier in Section 3.2, if an end user’s computer is configured with a caching resolver on the edge of the local network, an attacker would need to gain access to that local network in order to successfully execute an on the wire attack against the stub resolver. On the other hand, if the end user’s computer is configured to use a public DNS service as the caching resolver, the attacker needs to simply get in the network path between the end user and the public DNS server and so there is a much greater opportunity for a successful attack. Configuring a caching resolver closer to the end user can also reduce the possibility of on the wire attacks.

6.2.2. In the authoritative name servers
A possible solution would be to minimize the amount of data sent from the resolver. When a resolver receives the query "What is the AAAA record for www.example.com?", it sends to the root (assuming a cold resolver, whose cache is empty) the very same question. Sending "What are the NS records for .com?" would be sufficient (since it will be the answer from the root anyway). To do so would be compatible with the current DNS system and therefore could be deployable, since it is an unilateral change to the resolvers.

To do so, the resolver needs to know the zone cut [RFC2181]. There is not a zone cut at every label boundary. If we take the name www.foo.bar.example, it is possible that there is a zone cut between "foo" and "bar" but not between "bar" and "example". So, assuming the resolver already knows the name servers of .example, when it receives the query "What is the AAAA record of www.foo.bar.example", it does not always know if the request should be sent to the name servers of bar.example or to those of example. [RFC2181] suggest an algorithm to find the zone cut, so resolvers may try it.

Note that DNSSEC-validating resolvers already have access to this information, since they have to find the zone cut (the DNSKEY record set is just below, the DS record set just above).

It can be noted that minimizing the amount of data sent also partially addresses the case of a wire sniffer.

One should note that the behaviour suggested here (minimizing the amount of data sent in qnames) is NOT forbidden by the [RFC1034] (section 5.3.3) or [RFC1035] (section 7.2). Sending the full qname to the authoritative name server is a tradition, not a protocol MUST.

Another note is that the answer to the NS query, unlike the referral sent when the question is a full qname, is in the Answer section, not in the Authoritative section. It has probably no practical consequences.

6.2.3. Rogue servers

Traditional security measures (do not let malware change the system configuration) are of course a must. A protection against rogue servers announced by DHCP could be to have a local resolver, and to always use it, ignoring DHCP.
7. Security considerations

Hey, man, the entire document is about security!

8. Acknowledgments

Thanks to Nathalie Boulvard and to the CENTR members for the original work which leded to this draft. Thanks to Olaf Kolman, Francis Dupont and Ondrej Sury for the interesting discussions. Thanks to Dan York and Frank Denis for good written contributions.

9. References

9.1. Normative References


9.2. Informative References


[I-D.koch-perpass-dns-confidentiality]


dagon-malware

dns-footprint

darkreading-dns

dnschanger

dnsencrypt
Denis, F., "DNSCrypt", .

dnscurve
Bernstein, D., "DNScurve", .


crime

ditl
, "A Day in the Life of the Internet (DITL)", 2002.

data-protection-directive
, "European directive 95/46/EC on the protection of individuals with regard to the processing of personal data and on the free movement of such data", November 1995.

[passive-dns]

Author’s Address
Stephane Bortzmeyer
AFNIC
Immeuble International
Saint-Quentin-en-Yvelines 78181
France

Phone: +33 1 39 30 83 46
Email: bortzmeyer+ietf@nic.fr
URI:   http://www.afnic.fr/