DNS Privacy (dprive) Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 26, 2017

S. Bortzmeyer
AFNIC

November 22, 2016

Next step for DPRIVE: resolver-to-auth link draft-bortzmeyer-dprive-step-2-03

Abstract

This document examines the possible future work for the DPRIVE (DNS privacy) working group, specially in securing the resolver-to-authoritative name server link with TLS under DNS.

It is not intended to be published as a RFC.

REMOVE BEFORE PUBLICATION: this document should be discussed in the IETF DPRIVE group, through its mailing list. The source of the document, as well as a list of open issues, is currently kept at Github [1].

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 26, 2017.

Copyright Notice

Table of Contents

1. Introduction and background

To improve the privacy of the DNS user ([RFC7626]), the standard solution is to encrypt the requests with TLS ([RFC7858]). Just encrypting, without authenticating the remote server, leaves the user's privacy vulnerable to active man-in-the-middle attacks. [RFC7858] and [I-D.ietf-dprive-dtls-and-tls-profiles] describe how to authenticate the DNS resolver, in the stub-to-resolver link. We have currently no standard way to authenticate the authoritative name server, in the resolver-to-auth link.

The two cases are quite different: a stub resolver has only a few resolvers, and there is typically a pre-existing relationship. But a resolver speaks to many authoritative name servers, without any prior relationship. This means that, for instance, having a static key for the resolver makes sense while it would be clearly unrealistic for

the authoritative server.

Another difference is that resolvers are typically known by IP address (obtained by DHCP or manual configuration) while authoritative name servers are known by name (obtained from zone

delegation). This makes things easier for techniques similar to DANE: the manager of the ns1.example.net name server can always add a TLSA record under example.net while she may have problems modifying the zone under in-addr.arpa or ip6.arpa.

Note that, despite the fact that resolvers are in general configured by IP address, [I-D.ietf-dprive-dtls-and-tls-profiles] does not use it. It describes several techniques to get the domain name of the resolver, and authenticates using this name.

(On the other hand, the resolver knows also the IP address of the authoritative server, so it can authenticate based on this, the fourth and fifth solutions in Section 3.)

A third difference between the stub-to-resolver and resolver-to-auth links is that, in the second case, it is more difficult to report authentication problems to the end-user. (For better or for worse, the DNS is not end-to-end.)

The original charter of the DPRIVE working group, in force at the time of this draft, says "The primary focus of this Working Group is to develop mechanisms that provide confidentiality between DNS Clients and Iterative Resolvers" and adds "but it may also later consider mechanisms that provide confidentiality between Iterative Resolvers and Authoritative Servers". This document is here for this second step, "between Iterative Resolvers and Authoritative Servers". It will probably require a rechartering of the group.

2. TLS or not TLS

At first glance, the obvious protocol choice for encrypting the resolver-to-auth link is to use DNS over TLS or DTLS ([RFC7858], [I-D.ietf-dprive-dnsodtls]):

Already standardised

Relies on a well-know security protocol (and inventing a new security protocol is quite dangerous)

On the other hand, the DNS has some special properties. While a stub resolver talks to only a few few resolvers (and therefore can afford a few TCP+TLS connections), a resolver may hesitate in front of the task of managing hundreds of connections to remote authoritative servers. Some may think that a specially-designed protocol like [I-D.krecicki-dprive-dnsenc] is better.

If we choose TLS, we may require a minimum version of 1.3. TLS 1.3 [I-D.ietf-tls-tls13] will probably be standardised before this

document, and, specially combined with TCP Fast Open [RFC7413], it promises a minimum latency when contacting a new server.

3. Possible solutions

We can express the problem this way: if we want to use TLS-over-DNS to secure the link between the resolver and the authoritative server, it would be important to have a standard way to authenticate the authoritative server. Basically, the client will get the public key of the server in the TLS session, how will it know that this key is the right one?

Here is a comprehensive list of the six possible solutions to this problem. First, the two where the key (or a hash of it) is available somewhere else than in the TLS session.

3.1. Encode key in name

We could encode the key in the authoritative server name (as in DNScurve [dnscurve] [I-D.dempsky-dnscurve]). Here is an example of a domain using DNScurve: the names of the authoritative name servers are a Base-32 encoded form of the server's Curve25519 public key.

```
% dig +short NS yp.to
uz5dz39x8xk8wyq3dzn7vpt670qmvzx0zd9zg4ldwldkv6kx9ft090.ns.yp.to.
uz5hjgptn63q5qlch6xlrw63tf6vhvvu6mjwn0s31buw1lhmlk14kd.ns.yp.to.
uz5uu2c7j228ujjccp3ustnfmr4pgcg5ylvt16kmd0qzw7bbjgd5xq.ns.yp.to.
```

Securely transmitting the key would therefore be a by-product of

delegation. Among the limits of this solution, the length of these names limit the number of possible name servers, if we want to keep the delegation short. Also, it requires a cryptographic algorithm where keys are short (which means no RSA).

If we want to add cryptographic agility to this solution, we will need a few bytes before the key itself, to indicate, for instance, the algorithm it uses. It will reduce the possible key size even more.

3.2. Key in DNS

We could publish keys in the DNS, secured with DNSSEC (as in DANE [RFC6698]). This raises an interesting bootstrap problem: we need the key to get information privately with the DNS but we need the DNS to do so. A possible solution is to have an "unsecure" mode to retrieve the initial key material. The algorithm could be:

(0)The resolver remembers the keys of the authoritative name servers (in the same way it remembers the lowest RTT among a NS RRset),

(1)When the resolver needs to talk to a server (say ns1.example.net) for which it does not know the key, it does a TLSA request for _853._tcp.ns1.example.net,

(2)If the resolution of this request requires to talk to the very server we search the key for, the resolver connects to this server with TLS to port 853, does not authenticate, and sends the query. This step offers no authentication ("opportunistic"?).

(See also [I-D.ietf-dprive-dtls-and-tls-profiles], section 9.2.) The real algorithm will need to be more complicated since there are several servers per zone. A resolver may use the knowledge of TLS authentication it has to choose an authoritative name server among a NS RRset.

Another solution is for the authoritative name server to use [I-D.ietf-tls-dnssec-chain-extension] to send all the necessary DNSSEC records over TLS.

3.3. PKIX

We could use the X.509 security model [RFC5280]). The certificates for authoritative name servers would be signed by regular CAs, with the name of the server in the Subject Alternative Name (or may be its IP address in this field, but, as far as the author knows, few CAs issue certificates for IP addresses).

One of the problems is that resolvers will probably have different sets of trusted CA so an authoritative name server will not know in advance what percentage of the resolvers may authenticate it.

Of course, this technique would not work if the server used raw public keys ([RFC7250].

3.4. "reverse" DNS

The resolver could start from the IP address of the authoritative name server, and get a key from the name in the in-addr.arpa or ip6.arpa zone. For instance, if the authoritative server will be queried at 2001:db8:42:cafe::1:53, the request will be done for 3.5.0 .0.1.0.0.0.0.0.0.0.0.0.0.0.e.f.a.c.2.4.0.0.8.b.d.0.1.0.0.2.ip6.arpa.

There is also a boostrapping problem here too, but since there are only five RIR that manage the reverse DNS some sort of hard-coded or semi-hard-coded setup might be possible.

Unfortunately the operators for the space are often different from the operators of the DNS, so this is often not a reasonable solution administratively.

3.5. CGA

Another solution starting from the IP address of the authoritative name server would be to use CGA ([RFC3972]). The IPv6 address encodes the public key in the lower 64-bits of the address. So we could use the IPv6 address as the public key of the servers.

This only works for IPv6, doesn't have (much) cryptographic agility, and raises serious layering violation issues.

3.6. Lax security

Finally, we could simply not check the keys at all, accepting anything. This would break privacy promises, when there is an active attacker, able to pose as the authoritative name server. But it is still better, privacy-wise, than the current situation where DNS requests are sent in clear text.

4. Miscellaneous

A resolver may use a combination of these solutions. For instance, trying PKIX authentication (it does not require an extra lookup, except may be OCSP), if it fails, search a TLSA record, if there is none, depending on the resolver's policy, accept anyway. Clearly, trying all six solutions is unrealistic so some guidance on how to combine them will be necessary.

Trying the various solutions in sequence may seriously increase the latency, specially if there are timeouts involved. Using parallel attemps ("happy eyeballs", [RFC6555]) seem therefore crucial.

All these solutions can be improved by things like automatic key pinning ([RFC6797]).

An authoritative name server cannot know if the resolver authentified it, and how. In the future, it may be interesting to have a EDNS option to signal a successful authentication, or a failure, but this is out of scope currently.

If it is a concern that the same authoritative name servers are used for ordinary DNS and for encrypted DNS, there are several solutions. We may use front-end systems dispatching requests to port 53 and 853 to different servers. Or we may introduce a new delegation RR type, PNS (for Privacy Name Server), located only in the child zone (to avoid depending on a change of the provisioning software in the parent).

5. IANA Considerations

There is currently nothing to do for IANA. The future chosen solution may require some IANA action, such as a registry.

6. Security Considerations

For the time being, refer to each subsection under Section 3 to have an analysis of its security.

A general problem with all (or most?) encryption protocols will be the state to be kept in the server. It may make some denial-of-service attacks easier, of the protocol is not properly designed.

7. Acknowledgments

Shane Kerr for the ideas on authentication by IP address.

8. References

8.1. Normative References

[RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <http://www.rfc-editor.org/info/rfc7858>.

[I-D.ietf-dprive-dtls-and-tls-profiles] Dickinson, S., Gillmor, D., and T. Reddy, "Authentication and (D)TLS Profile for DNS-over-(D)TLS", draft-ietf-dprive-dtls-and-tls-profiles-07 (work in progress), October 2016.

8.2. Informative References

[RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, DOI 10.17487/RFC3972, March 2005, <http://www.rfc-editor.org/info/rfc3972>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S.,

Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <http://www.rfc-editor.org/info/rfc5280>.

[RFC6555] Wing, D. and A. Yourtchenko, "Happy Eyeballs: Success with Dual-Stack Hosts", RFC 6555, DOI 10.17487/RFC6555, April 2012, <http://www.rfc-editor.org/info/rfc6555>.

[RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August 2012, <http://www.rfc-editor.org/info/rfc6698>.

[RFC6797] Hodges, J., Jackson, C., and A. Barth, "HTTP Strict Transport Security (HSTS)", RFC 6797, DOI 10.17487/RFC6797, November 2012, <http://www.rfc-editor.org/info/rfc6797>.

[RFC7250] Wouters, P., Ed., Tschofenig, H., Ed., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7250, DOI 10.17487/RFC7250, June 2014, <http://www.rfc-editor.org/info/rfc7250>.

[RFC7413] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP Fast Open", RFC 7413, DOI 10.17487/RFC7413, December 2014, <http://www.rfc-editor.org/info/rfc7413>.

[RFC7626] Bortzmeyer, S., "DNS Privacy Considerations", RFC 7626, DOI 10.17487/RFC7626, August 2015, <http://www.rfc-editor.org/info/rfc7626>.

[I-D.dempsky-dnscurve] Dempsky, M., "DNSCurve: Link-Level Security for the Domain Name System", draft-dempsky-dnscurve-01 (work in progress), February 2010.

[I-D.krecicki-dprive-dnsenc] Krecicki, W., "Stateless DNS Encryption", draft-krecicki-dprive-dnsenc-01 (work in progress), October 2015.

[I-D.ietf-tls-tls13] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", draft-ietf-tls-tls13-18 (work in progress), October 2016.

[I-D.ietf-tls-dnssec-chain-extension]
Shore, M., Barnes, R., Huque, S., and W. Toorop, "A DANE Record and DNSSEC Authentication Chain Extension for TLS", draft-ietf-tls-dnssec-chain-extension-01 (work in progress), July 2016.

[I-D.ietf-dprive-dnsodtls]
Reddy, T., Wing, D., and P. Patil, "Specification for DNS over Datagram Transport Layer Security (DTLS)", draft-ietf-dprive-dnsodtls-12 (work in progress), September 2016.

[dnscurve]
Bernstein, D., "DNSCurve: Usable security for DNS", June 2009, <http://dnscurve.org/>.

8.3. URIs

[1] https://github.com/bortzmeyer/ietf-dprive-step-2

Author's Address

Stephane Bortzmeyer
AFNIC
1, rue Stephenson
Montigny-le-Bretonneux 78180
France

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