

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 21, 2017

B. Schwartz
Google
February 17, 2017

A DNS Resource Record for TLS Server Name Indication (DNS SNI)
draft-schwartz-dns-sni-02

Abstract

The SNI record type allows a domain owner to specify the "server name" to indicate in TLS connections, if it is different from the domain name. This allows domains that use shared hosting and wildcard or multi-domain (UCC) certificates to change the only domain name shown in cleartext, to prevent a passive adversary from identifying exactly which domain a user is accessing.

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 August 21, 2017.

Copyright Notice

Copyright (c) 2017 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. Code Components extracted from this document must include Simplified BSD License text as described in [Section 4.e](#) of

Internet-Draft

DNSSNI

February 2017

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

| | | |
|------------------------|--|--------------------|
| 1. | Introduction | 2 |
| 1.1. | Threat example: shared hosting service | 3 |
| 2. | Concept | 4 |
| 2.1. | Need for a new record type | 4 |
| 2.1.1. | Using SRV | 5 |
| 2.1.2. | Using TLSA | 5 |
| 2.1.3. | Using TXT | 5 |
| 3. | Client Requirements | 5 |
| 4. | Server Requirements | 6 |
| 5. | Use Cases | 6 |
| 6. | Security Considerations | 6 |
| 7. | Performance Considerations | 7 |
| 8. | Considerations for optimizing privacy | 9 |
| 9. | Extensibility | 10 |
| 10. | IANA Considerations | 10 |
| 11. | Acknowledgements | 11 |
| 12. | References | 11 |
| 12.1. | Normative References | 11 |
| 12.2. | Informative References | 12 |
| | Author's Address | 12 |

[1.](#) Introduction

Modern Internet standards are designed to avoid unnecessarily revealing sensitive information about a user's behavior to a passive adversary. One key piece of sensitive information is the name of the domain that a user is visiting. Most commonly, this is the DNS name of a website, so the sensitive information indicates browsing history at domain granularity.

Several protocol standards contain elements that protect the user's browsing history from a passive adversary. The DNS over TLS standard [[RFC7858](#)] allows clients to perform DNS queries without revealing the request or response to an attacker. This allows clients to access DNS-named services without publicly revealing the name of the services that they are about to access.

After a DNS lookup, the HTTPS standard [[RFC2818](#)] allows clients to

perform HTTP requests to a server without revealing the contents of the HTTP request or response, using TLS. This hides the website's host name from being observed by a passive adversary. Previous versions of TLS transferred the server's certificate in cleartext, revealing some information about the identity of the destination, but

TLS 1.3 [[I-D.ietf-tls-tls13](#)] enables encrypting the certificate as well.

However, modern use of HTTPS requires providing a TLS Server Name Indication (SNI, [[RFC6066](#)]), which conveys the website's DNS name in cleartext. Sending the Server Name Indication in cleartext reveals the user's sensitive information to a passive adversary, defeating the desired privacy benefit.

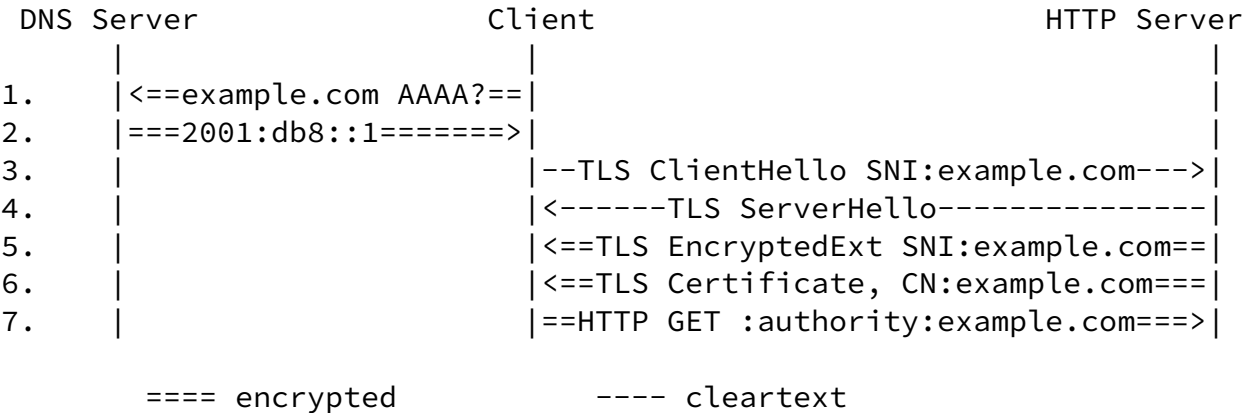


Figure 1: HTTP request with DPRIVE, HTTP/2, and TLS-1.3

This document describes a new DNS record that allows a domain owner to tell clients precisely what Server Name Indication to send. For domains that use wildcard or multi-domain (UCC) certificates, this name may differ from the website's DNS name, preventing the passive adversary from learning which domain the user is accessing with high confidence. Instead, the adversary learns only the IP addresses with confidence, so all services that share an IP address and certificate can become part of a single anonymity set that is larger than any single service. Clients can make use of this record without adding delay to their connection process, and clients who do not make use of this record are not negatively impacted.

[1.1](#). Threat example: shared hosting service

Consider a shared hosting service, example.com, that offers each user their own website, hosted at a unique subdomain like alicesphotos.example.com. For confidentiality and access control, example.com provides its users with service over HTTPS, using a wildcard certificate for *.example.com. All users' subdomains are hosted by the same set of servers.

Prior to the introduction of TLS Server Name Indication, this arrangement was sufficient to prevent a passive adversary who monitors example.com's incoming traffic from determining which user is visiting which subdomain. However, modern TLS clients publish the

Fully Qualified Domain Name in the TLS ClientHello as the Server Name Indication, so the adversary can easily identify which client IP address is accessing which subdomain. Mapping IP addresses to individual users is often straightforward for an adversary with access to ISP records or a popular website's logs.

The user's choice of which subdomain to visit may be sensitive. Subdomains of shared hosting services often contain controversial material, and users may reasonably wish not to make public the names of the subdomains that they read.

The user's identity as the author of a particular subdomain is typically even more sensitive. Authors of controversial sites on shared hosting often choose not to reveal their true identity or location, to protect their physical safety. However, the author of a domain is typically the only visitor who is likely to upload a large amount of data. By combining the TLS SNI with a simple upload threshold, the adversary can determine the author's client IP address with high confidence.

[2.](#) Concept

We define a new RRTYPE: "SNI". To control the TLS Server Name Indication that clients set, a TLS endpoint operator MAY publish one or more SNI records for their domain name (the "base domain").

Each SNI record MUST contain a prefix of the form _(port)._(transport), the same prefix used for a TLSA record [[RFC6698](#)]. The SNI record's RDATA MUST contain the intended Server

Name. For example, if the operator of subdomain1.example.com wishes to inform HTTPS visitors to set a Server Name Indication of "subdomain2.example.com", they would publish a record of the form

```
_443._tcp.subdomain1.example.com. IN SNI "subdomain2.example.com"
```

The RDATA of an SNI record consists of UTF-8 octets for use as a HostName in the TLS Server Name Indication field, in conformance with [Section 3 of \[RFC6066\]](#). DNS resolvers SHOULD treat SNI records as resource records of unknown type [\[RFC3597\]](#).

[2.1.](#) Need for a new record type

Instead of defining a new record type, we could convey this information using an existing record type. However, using any existing record type has some significant drawbacks.

[2.1.1.](#) Using SRV

We could achieve an equivalent effect using SRV records [\[RFC2782\]](#), with a new protocol type.

This would function very similarly to an SNI record, but it would be a violation of the SRV specification, which requires the client to redirect to the domain in the RDATA and perform additional DNS queries. SRV records also include a port, which is not meaningful in this context. Using SRV records in this way might also make future extensions (e.g. to other forms of RDATA) somewhat less appealing.

[2.1.2.](#) Using TLSA

We could use the existing TLSA record type [\[RFC6698\]](#). Clients could extract the server's supported names from a "full certificate association" [Section 4.1 \[RFC6698\]](#) TLSA response.

This approach doesn't work well for wildcard certificates, because the client cannot tell what SNI to set. Using TLSA in this way would also create problems for servers that use SNI for more than certificate selection, and for servers that have multiple overlapping

certificates. It might also be very inefficient: transferring an entire certificate just to learn its names is wasteful, and TLSA records that contain hashes would have to be discarded.

[2.1.3.](#) Using TXT

We could encode this information in a TXT record, but that would violate the intended purpose of TXT records: to convey information to human readers.

[3.](#) Client Requirements

When initiating a TLS connection to a domain and port, a client application SHOULD query the SNI record for the prefixed domain at the same time as the A or AAAA query. The application MUST discard or ignore any SNI record whose RDATA is not a valid TLS HostName or an empty string. If an application's query returns multiple valid SNI records, the application SHOULD choose one at random.

The client application SHOULD use the RDATA of the selected SNI record as the Server Name Indication in the TLS ClientHello. If the selected SNI record is present with an empty RDATA (RDLEN = 0), then the client SHOULD omit the Server Name Indication.

Client applications MUST NOT allow the contents of the SNI record to influence their certificate validation behavior. The client's

certificate validation MUST be based on the user's specified destination, not the RDATA of the SNI DNS record. If the server's certificate is not valid for the user's specified destination, the client application MUST terminate the TCP connection and SHOULD retry the TLS connection without the SNI RDATA, using the client's default Server Name Indication behavior.

[4.](#) Server Requirements

Domain owners that publish an SNI record MUST ensure that all relevant servers, when receiving this SNI value, reply with a certificate whose set of valid domains includes the base domain. The servers MUST satisfy this criterion for all SNI records that have been published and not yet expired. Relevant servers are those whose IP addresses are in the correct family and are listed in a non-

expired address record for this domain.

Domain owners that publish an SNI record MUST ensure that their servers do not depend on the TLS Server Name Indication to select which service to provide to the user. Instead, participating servers SHOULD select the service using the tunneled protocol's service identifier, e.g. the :authority pseudo-header in HTTP/2, bearing in mind that this identifier is controlled by the client and may be untrusted.

[5.](#) Use Cases

A host that serves many subdomains with a single wildcard certificate could set the SNI of all subdomains to the same fixed subdomain, in order to prevent a passive adversary from learning which subdomain a user is accessing.

A host that uses multiple-domain (UCC) certificates, and which controls the DNS entries for domains that it hosts, could set several SNI records on the domains that share one certificate, with each record indicating a different domain name that is also provided in that certificate. This would prevent a passive adversary from learning with confidence which of those domains a client is accessing.

A TLS service operator who does not use shared hosting could set an empty SNI record, to tell the client not to use SNI.

[6.](#) Security Considerations

The SNI record is provided over DNS, and MAY be used with or without DNSSEC. The SNI record is therefore untrusted, and clients MUST NOT rely on its presence or correctness.

A passive adversary who monitors the client's DNS queries could learn which domain the client is accessing, defeating the confidentiality benefits of this record type. Therefore, clients wishing to conceal browsing history from a passive adversary SHOULD also use DNS over TLS [[RFC7858](#)]. However, clients MAY use SNI records with standard DNS, in order to provide protection against a weak passive adversary who can monitor only the link between the client and the TLS server, not the link between the client and the DNS server.

An active adversary who controls the DNS could remove the SNI record, or add a malicious record. Removing the record would cause the client to revert to its default behavior in the absence of an SNI record. Adding a malicious record could cause the client to set an incorrect or nonexistent Server Name. This will likely cause the initial connection attempt to fail, but does not cause any additional loss of confidentiality, because the client will still perform standard TLS certificate verification. (Like the previous adversary, this active adversary can also monitor DNS queries, so they can still learn which domain the client is attempting to access.)

7. Performance Considerations

Client applications that intend to make use of the SNI record SHOULD issue a DNS query for the SNI record immediately following the A or AAAA record query. The client SHOULD NOT wait for an A or AAAA response before issuing the SNI query, as this would introduce an extra round-trip delay to the DNS server. A client application SHOULD NOT delay initiating a TCP connection while waiting for the SNI DNS response.

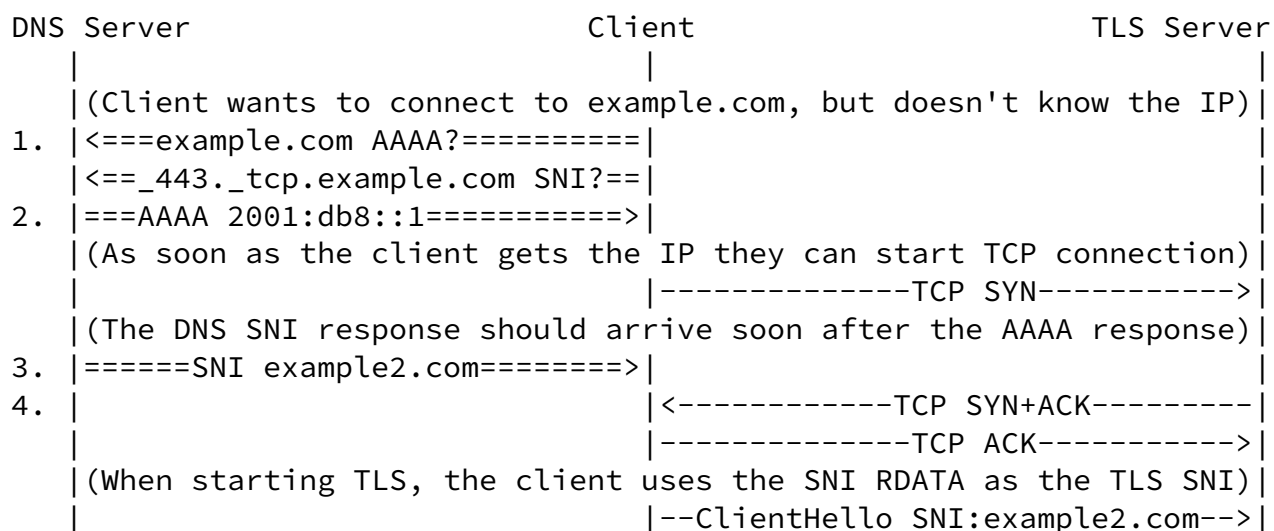


Figure 2: TLS connection with SNI record, typical case

If the client can begin sending data to the server (e.g. SYN-ACK is

received) before the SNI RDATA has been received, the client MAY issue a ClientHello using its default behavior (i.e. without using the SNI RDATA). Note that if the client's DNS resolver has similar performance for SNI and A/AAAA queries, the client will generally receive the SNI response before the destination server receives the TCP SYN packet, thus avoiding this situation.

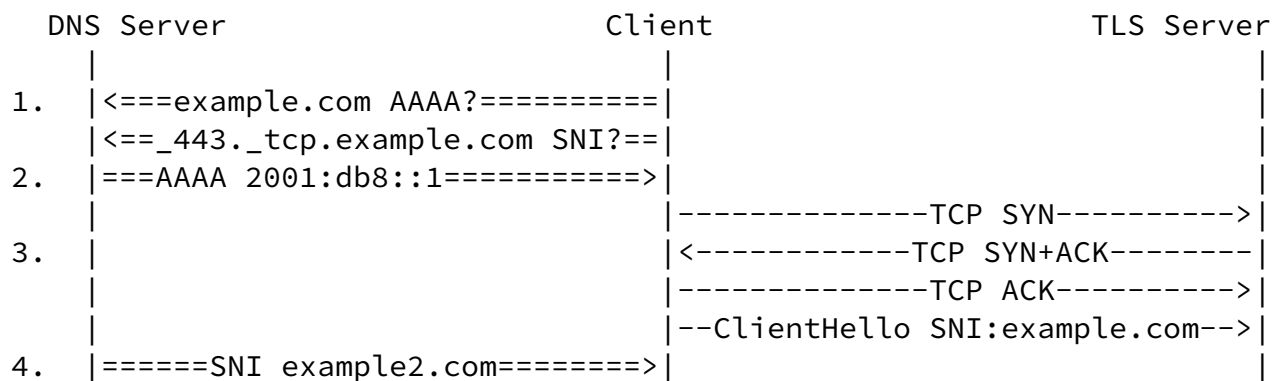


Figure 3: TLS connection with SNI record, slow case (OPTIONAL)

If the client is using TCP Fast Open [[RFC7413](#)], and the destination IP address is known (e.g. AAAA record is in cache) but the SNI RDATA is not known, the client MAY send a TLS ClientHello using its default behavior. However, the client SHOULD also issue a DNS query for the SNI record, in order to populate the local cache for subsequent connection attempts.

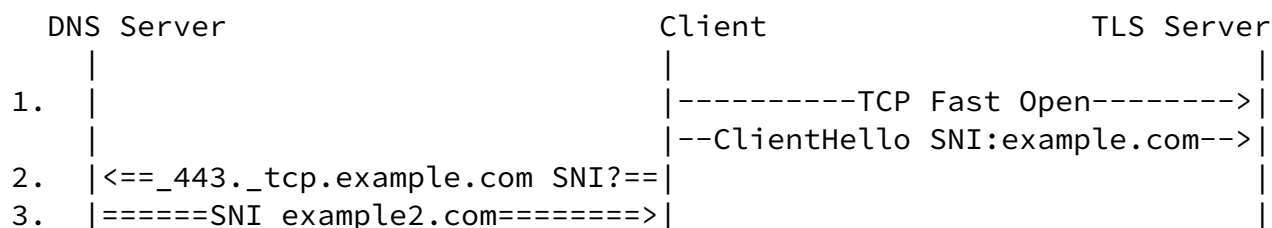


Figure 4: TCP Fast Open, AAAA cached but not SNI (OPTIONAL)

If the client is using TCP Fast Open and the destination IP address is not known, the client SHOULD issue A/AAAA and SNI requests simultaneously. If the A or AAAA response is received before the corresponding SNI response, the client SHOULD wait briefly (e.g. 1 millisecond) to allow receipt of the SNI response before issuing a TLS ClientHello.

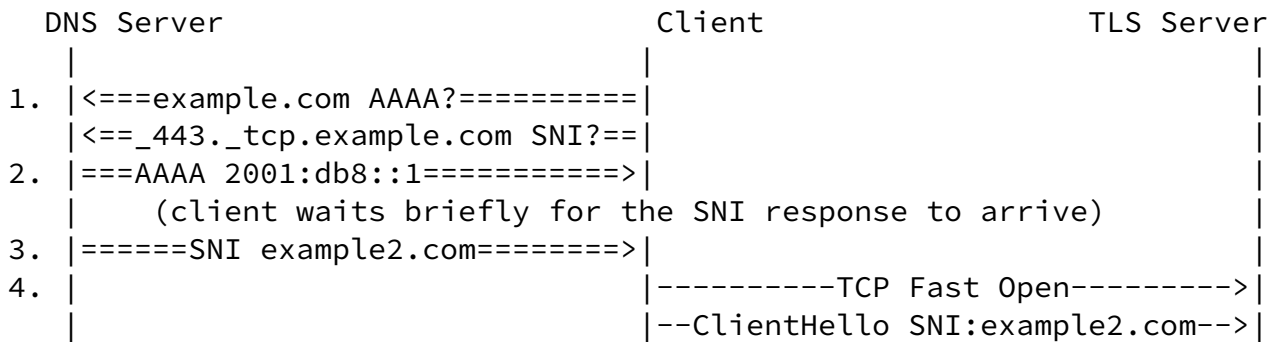


Figure 5: TCP Fast Open, nothing cached (delay is RECOMMENDED)

TODO: Check if 1ms is a reasonable delay to receive an immediately subsequent packet on current hardware.

8. Considerations for optimizing privacy

By publishing an SNI record, a service operator can reduce the ability of a passive network adversary to determine with certainty which domain each user is accessing. Reducing leakage of information to this adversary, and reducing this adversary's confidence in the information they do collect, improves user privacy.

Regardless of SNI, a passive adversary still learns the client's and destination's IP addresses. The adversary may have foreknowledge of which domains are hosted at that address. An active adversary could attempt a similar connection in order to learn more about the services hosted at the destination. The purpose of this proposal is to ensure that if there is more than one service hosted at the destination IP, a passive network adversary does not directly learn which service the client is using. Services that share an IP address and TLS certificate with a wider range of other services can benefit more from the use of an SNI record.

An adversary may be able to distinguish different encrypted services on the same IP address by the size, frequency, and timing of encrypted packets. Use of the SNI record does not prevent attacks of this kind.

If the adversary has full knowledge of the contents of the global DNS (a strong adversary), then users of a domain only experience a privacy benefit if another domain at the same IP address uses the same SNI RDATA. The privacy benefit increases as more domains share the same SNI RDATA and the same IP addresses.

If all clients will use the SNI record, then the greatest privacy

benefit is achieved by configuring all services that share a certificate to use the same SNI. However, a service operator may

also choose to publish multiple SNI records for a domain, listing several other services that share the same certificate. The client will then choose from among these Server Name Indications at random. This approach maintains protection of clients that are aware of the SNI record, but also benefits users who are not using the SNI record: if the adversary cannot distinguish these two groups of users, then it cannot tell with certainty which domain any user is accessing. Note that due to response latency issues, some clients might not be using the SNI record contents, even if they have issued an SNI DNS query (see Performance Considerations).

Use of the SNI record does not prevent an active adversary from learning which certificate the server is using. A simple active adversary can reuse the client's observed Server Name Indication, to learn which certificate is sent by the server in reply. Therefore, domain owners seeking to protect users against an active adversary MUST ensure that the server's certificate includes a sufficiently large and diverse set of domains to achieve their privacy goals.

In TLS 1.2 [[RFC5246](#)], the server sends its certificate in cleartext, visible to a passive adversary. A passive adversary can also observe the certificate ID if the client performs an OCSP lookup [[RFC6960](#)]. Therefore, servers wishing to hide the certificate MUST implement TLS 1.3 and OCSP Stapling [[RFC6961](#)]. Note that this precaution does not protect against even a simple active adversary.

[9.](#) Extensibility

The SNI record as described here contains two main mechanisms for extensibility. First, future extensions may describe records whose RDATA is not structured as a valid domain name (e.g. containing whitespace or "=" characters). Clients conformant to this specification MUST ignore such records.

Second, clients that connect to a server and experience a negotiation failure SHOULD retry the connection without making use of the SNI record. This allows future changes to the semantics of SNI records with RDATA that are valid domain names, without causing a serious problem for clients conformant to this version of the specification.

10. IANA Considerations

This document uses a new DNS RR type, SNI, which will require a value assignment from IANA.

Schwartz

Expires August 21, 2017

[Page 10]

Internet-Draft

DNSSNI

February 2017

11. Acknowledgements

Thanks to Ben Vitale, Warren Kumari, and Vinicius Fortuna for their constructive feedback.

12. References

12.1. Normative References

- [RFC3597] Gustafsson, A., "Handling of Unknown DNS Resource Record (RR) Types", [RFC 3597](#), DOI 10.17487/RFC3597, September 2003, <<http://www.rfc-editor.org/info/rfc3597>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", [RFC 5246](#), DOI 10.17487/RFC5246, August 2008, <<http://www.rfc-editor.org/info/rfc5246>>.
- [RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", [RFC 6066](#), DOI 10.17487/RFC6066, January 2011, <<http://www.rfc-editor.org/info/rfc6066>>.
- [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>>.
- [RFC6960] Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", [RFC 6960](#), DOI 10.17487/RFC6960, June 2013,

<<http://www.rfc-editor.org/info/rfc6960>>.

- [RFC6961] Pettersen, Y., "The Transport Layer Security (TLS) Multiple Certificate Status Request Extension", [RFC 6961](#), DOI 10.17487/RFC6961, June 2013, <<http://www.rfc-editor.org/info/rfc6961>>.
- [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>>.
- [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>>.

Schwartz

Expires August 21, 2017

[Page 11]

Internet-Draft

DNSSNI

February 2017

[12.2](#). Informative References

- [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.
- [RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", [RFC 2782](#), DOI 10.17487/RFC2782, February 2000, <<http://www.rfc-editor.org/info/rfc2782>>.
- [RFC2818] Rescorla, E., "HTTP Over TLS", [RFC 2818](#), DOI 10.17487/RFC2818, May 2000, <<http://www.rfc-editor.org/info/rfc2818>>.

Author's Address

Benjamin M. Schwartz
Google, Inc.
111 8th Ave
New York, NY 10011
USA

Email: bemasc@google.com

Schwartz

Expires August 21, 2017

[Page 12]