Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 9, 2017

F. Fieau, Ed.
E. Stephan
Orange

July 8, 2016

Limited Use of Remote Keys for Interconnected CDNs
draft-cdni-fieau-lurk-https-delegation-00

Abstract

Sharing private keys amongst administrative entities raises numerous issues for end-users, intermediaries and content origins. The IETF envisions the standardization of protocols to limit the exchange of private keys (LURK). This document presents CDN Interconnection (CDNI) use cases based on the Limited Use of Remote Keys (LURK) protocol and architecture and identifies a set of requirements.

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 January 9, 2017.

Table of Contents

1. Introduction

CDNI requirements [RFC7337] and use cases [RFC6770] specify the requirements and use cases of HTTP content delivery between CDNs. The intent of this document is to pursue the work by proposing a solution for delegating content delivery over secure protocols like

HTTPS or DTLS. Conversely to HTTP delivery, HTTPS [RFC2818] allows content delivery between Content Service Provider (CSP) and End-users with integrity and confidentiality.

From the CDNI perspective, all parties - UAs, CSPs origin servers, and CDNs - are involved in the chain of trust and preserving this chain of trust in HTTPS raises concerns.

Indeed, regarding TLS certificates, CSPs and CDN providers are reluctant to share private keys mainly because of legal and security issues about private keys. Therefore the CDNI working group must specify a solution that avoids the exchange of private keys between CDNs.

This document leverages the work of the LURK initiative [LURK_Mailing_List] and discusses the introduction of a Limited Use of Remote Keys (LURK) solution in the CDNI architecture. It presents 2 use cases which complement those described in [I-D.mglt-lurk-tls-use-cases] and identifies a set of requirements for CDNI. Finally it discusses different options and identifies a set of open issues.

The proposed use cases are based on DNS redirection. The user agent is redirected to a dCDN to establish a TLS session to get HTTPS content. Additional use cases are expected in future versions of the draft.

This version focus on the delivery over HTTPS only.

2. Terminology

This document uses terminology from CDNI framework documents such as CDNI requirements [RFC7337] and CDNI interface specifications documents - CDNI metadata interface [I-D.ietf-cdni-metadata], CDNI Control interface [I-D.ietf-cdni-control-triggers] and Logging interface [I-D.ietf-cdni-logging].

3. CDNI architecture using LURK

This document introduces a Key Server in the CDNI architecture. The general LURK architecture is described in the Session Key interface draft [I-D.cairns-tls-session-key-interface].

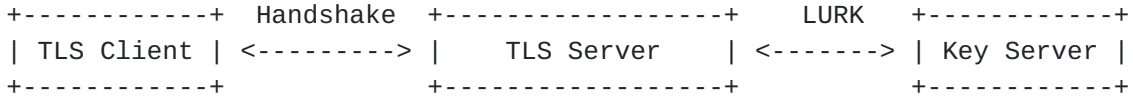

Figure 1: General Key Server Architecture

In CDNI, a LURK interface allows private keys to remain under the authority of its owner - typically the CSP or the uCDN - while delivering the content over HTTPS.

The LURK interface can be introduced at different locations in the CDNI architecture. The location of the LURK function depends on the use cases. Additionally, this architecture may support variants where the Key server is owned by a third party.

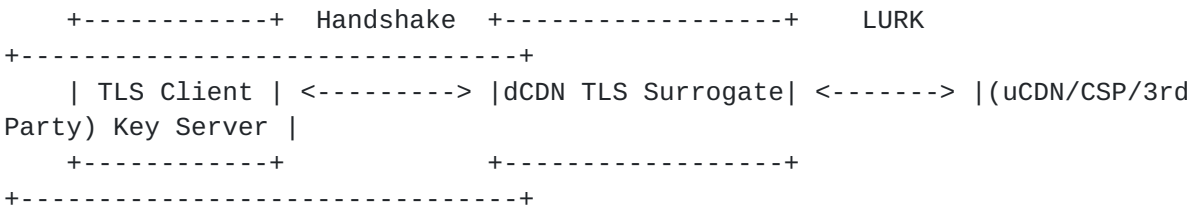

Figure 2: Key Server in CDNI Architecture

This document complements the LURK architecture of [I-D.mglt-lurk-tls-use-cases] with CDNI use cases. In those use cases, content delivery is decoupled from both content acquisition and signaling information needed to route and control the request.

Currently CDNI signaling exchanges occur over the Request Routing Redirection interface (information to route the request across CDNs), the Metadata interface (per content or group of content information about the acquisition and the delivery), the Logging interface (exchange of monitoring information about the delivery) and the Control interface (information for controlling the lifecycle of the aforementioned interfaces).

The LURK interface for CDNI adds two types of exchange: one for acquiring the certificate associated with the origin domain and the other for establishing delivery confidentiality and integrity.

This document presents two use cases of HTTPS delivery which rely on a LURK function to avoid exchanging private keys between CDNs:

A. uCDN Key server: the CSP has provided his certificate and private keys to the uCDN. the uCDN provides the LURK key server and interface.

B. CSP Key server: the CSP provides the LURK Key Server and interface.

The table below presents the use cases developed in the Section 3.

Use Case name	UA Redirection	uCDN redirection	CSP Cert delegation
UC A: uCDN KS	DNS CNAME	recursive	uCDN
UC B: CSP KS	DNS CNAME	recursive	No

Figure 3: Use cases description table

The use cases' call flows are respectful of the CDNI redirection [I-D.ietf-cdni-redirection] for the description of redirection methods in CDNI.

They share the same steps.

The figure below illustrates the framework use cases where the surrogate doesn't handle the CSP private keys and where the surrogate remotely accesses materials to sign and encrypt information exchanged over the TLS connection.

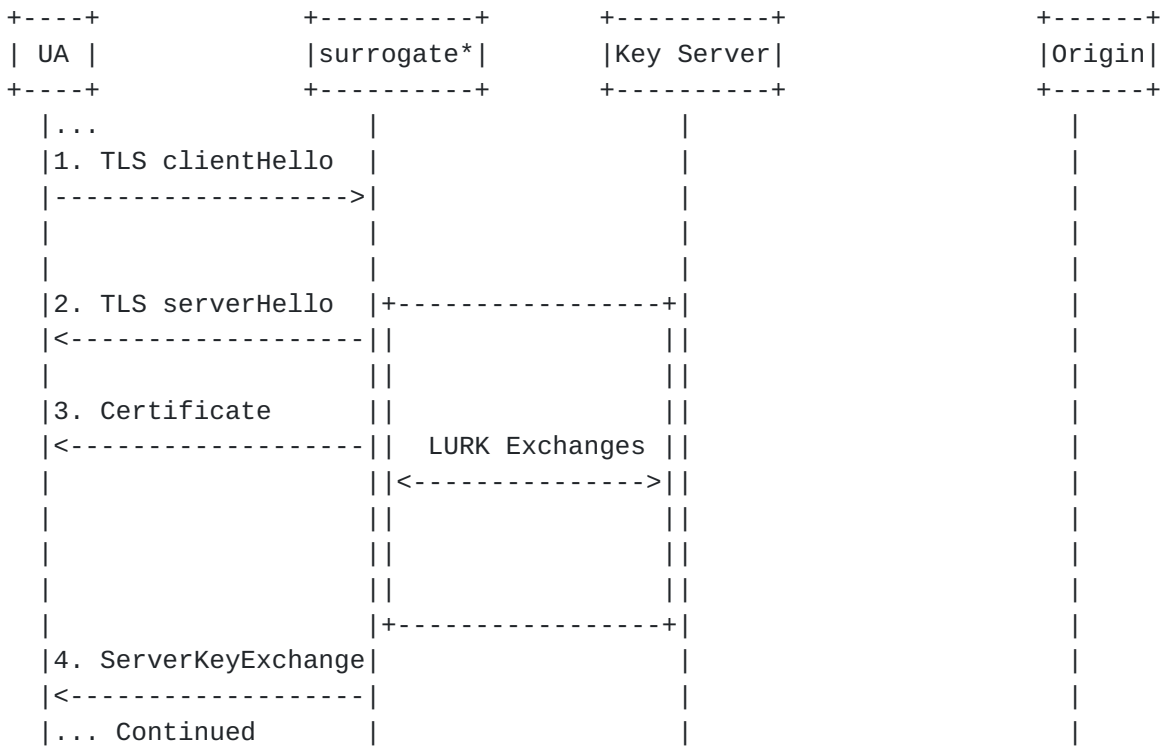

Figure 4: LURK exchanges in CDNI architecture

4. LURK use cases for CDNI

Two use cases are considered in this document to implement LURK in CDNI:

Use Case A. uCDN Key Server

Use Case B. CSP Key Server

4.1. Use case A: uCDN Key Server

In this use case, the dCDN has a LURK interface to a Key Server hosted at the uCDN side. It may be typically a case of CDNI regional delivery delegation.

This following example is based on ECDHE cypher suite. The UA is first redirected from the uCDN to a dCDN using a recursive DNS redirection. Then the UA initiates a TLS connection with the dCDN to get his content. Since the dCDN does not store the private keys for the requested certificate, it queries the uCDN Key Server (KS) to get the credential to establish the TLS session with the User Agent. Finally the dCDN can deliver the content in HTTPS to the UA using the CSP certificate.

The UA sends an HTTPS request to the CSP to get a content.

a. to d. As seen on figure 5, once the DNS resolution is over, i.e., UA was able to resolve dCDN surrogate IP@ steps [a.] to [d.], the user agent connects to the dCDN surrogate. Note that DNS cache is not shown on the figure.

1. The UA sends a ClientHello, as defined in the TLS protocol [RFC5246]. The SNI field of the ClientHello includes the CSP domain name.

2. The surrogate receives the ClientHello from the UA, and sends a ServerHello to the UA.

3. The surrogate parses the SNI field, and determines the Key Server interface of the CSP domain name. The surrogate uses this piece of information to determine the certificate of the delivery. The surrogate sends the public certificate to the UA. The UA validates the certificate.

4. The surrogate determines the ECDHE parameters, and requests the Key Server to sign these parameters with the CSP private key. The request includes the domain name received by the surrogate in the SNI field of the ClientHello.

5. The Key Server returns the necessary credentials to the dCDN surrogate over a secure tunnel.

6. and 7. the UA and the surrogate exchange public DH parameters and compute session keys.

8. The UA and the surrogate establish a secure connection. The UA issues its request for content over HTTPS.

The surrogate then processes the original request.

Below is an example of the handshake establishment:

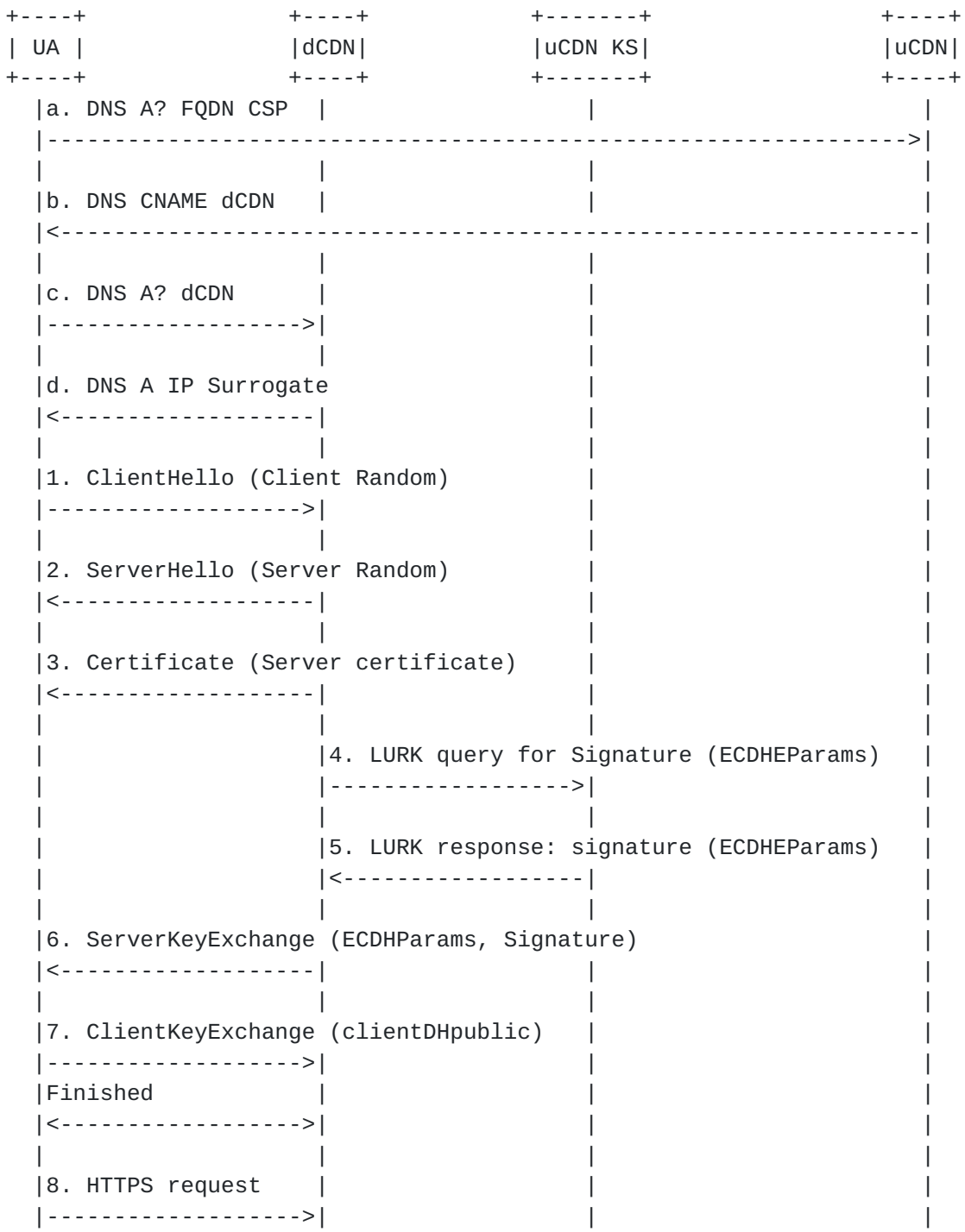

Figure 5: Key Server hosted by uCDN

4.2. Use case B: CSP Key Server

This section describes a use case in CDNI where the CSP provides the Key Server (KS).

In this example, the CSP delegates the HTTPS content delivery to an uCDN that in turn delegates the HTTPS delivery to a dCDN. For obvious legal or security reasons, the CSP does not want his private keys to be stored on all CDNs involved in the interconnection. In that case, the dCDN relies on credentials received from a CSP Key Server (KS) to deliver HTTPS content.

The dCDN cannot here request the KS directly. Instead, the dCDN LURK request will be relayed by the uCDN to the Key Server. Prior to that, the uCDN has to check whether the received LURK request is valid, e.g., complies with Content Distribution policies [I-D.ietf-cdni-metadata].

In the following example, the CSP stores his certificates and private keys on a Key Server that is able to compute and provide credentials for TLS establishment.

1. The UA sends a ClientHello to the dCDN's surrogate, as defined in the TLS protocol [RFC5246]. The SNI field of the ClientHello includes the CSP domain name.

2. The dCDN's surrogate receives the ClientHello from the UA, and sends a ServerHello to the UA..

3. The surrogate parses the SNI field, and determines the Key Server interface of the CSP domain name and determine the certificate of the delivery. The surrogate sends the public certificate to the UA. The UA checks the certificate signature with the public key.

4. The dCDN's surrogate requests the uCDN to sign parameters with the private key.

5. The uCDN parses the SNI field, and determines the Key Server interface of the CSP domain name. The uCDN relaies the LURK request received from the dCDN's surrogate, to the Key Server to sign parameters with the private key.

6. The Key Server returns the necessary credentials to the uCDN surrogate.

7. The uCDN returns the necessary credentials to the dCDN's surrogate.

8. and 9. the UA and the surrogate exchange public DH parameters and compute session keys.

10. The UA and the surrogate establish a secure connection. The UA issues its request for content over HTTPS.

The surrogate then processes the original request.

Figure 6: Cascaded delegation with LURK

4.3. Other use cases

Other use cases will be described in a further version of this draft.

5. Requirements

This section is a first attempt to identify the requirements introduced by the delivery of HTTPS content. Some of the requirements are new, whereas others may update existing CDNI requirements [RFC7337].

5.1. General

LURK-GEN-1: The specification should not require any change in User Agent. This is already stated in [RFC7337]/GEN-2

Discussion: Despite this is obvious, it is important to notice that recent limitation in TLS (version, cyphers...) or HTTP (cyphers) documents may constrain User Agent.

LURK-GEN-2: The user agent should be able to see that the requested content is delivered by the CDN using the CSP domain. Delivery domain name SHOULD be the same as the CSP portal domain name (or a sub-domain request name)

LURK-GEN-3: The Certificate delivered by the dCDN and CSP must match the domain of the URL [RFC2818]. As an example, it means that no certificate error occurs on the UA when the dCDN has redirected the UA a dCDN.

LURK-GEN-4: The dCDN's surrogates which implements LURK interface should support the delivery of content of the protocol HTTPS.

LURK-GEN-5: The dCDNs must be able to present the CSP certificate and credentials to the user agent when establishing a TLS session

5.2. CDNI Control Interface requirements

LURK-CI-1: The CDNI Control Interface may allow the bootstrapping of a Key Server interface. For example this may include:

- discovery the Key Server interface endpoint.

- credentials for Key Server and CDN mutual authentication.

- TLS version, Certificate types, TLS crypto suites.

Proposal: add this requirement to the section CDNI Control Interface of [RFC7337].

5.3. CDNI Footprint and Capabilities Advertisement interface requirements

LURK-FC-1: The CDNI Footprint and Capabilities Advertisement interface should allow the downstream CDN to communicate to the upstream CDN about its capabilities regarding the support of client side of the Key Server interface.

5.4. CDNI Logging Interface requirements

This section identifies additional requirements related to the CDNI Logging interface (LI).

TLS-LI-1: the CDNI Logging interface should allow a dCDN to report to the uCDN logging for deliveries which fail during the establishment of the secure connection, e.g., ability to report certificate validation error.

TLS-LI-2: The CDNI Logging interface should allow dCDN to report to uCDN logging incomplete deliveries due to encryption errors. Discussion: this requirement is mostly a subcase of LI-2 about incomplete deliveries.

LURK-LI-3: the CDNI Logging interface should allow a dCDN to report to the uCDN secure connections failures when using LURK interface.

As an example, the UA can't establish a secure connection to a dCDN because either, the credentials provided by a Key Server are invalid, or because the Key Server has refused to provide them to the dCDN.

5.5. Key Server Interface requirements

LURK-KS-1: A Key Server interface must not allow the exchange private keys. This is the centrality of the LURK interface.

LURK-KS-2: The Key Server and the requesting CDN must authenticate each other, according to the information provided by the CDNI Control interface.

LURK-KS-3: The dCDN Key Server interface must be able to send a LURK request to a Key Server.

As an example (UC A, step 4), the dCDN surrogate determines ECDHE parameters (...), and requests the Key Server to sign these

parameters with the CSP private key. The request includes the domain name received by the surrogate in the SNI field of the ClientHello.

6. Discussions

6.1. HTTPS and TLS

HTTPS contents are delivered with the dCDN credentials. The introduction of a Key Server in the CDNI architecture allows the HTTPS content to be delivered with the CSP origin server certificate. It conforms to the end-to-end HTTPS framework [RFC7230].

6.2. Cyphersuites in CDNI

CDNI WG and LURK WG should use a common set of cyphersuites, or CDNI WG should specify or points to the set of suites to support.

TLS and HTTP2 recommend different set of cypher suites. CDNI WG should clarify which one should be supported in the case of a HTTPS delivery based on LURK credentials: HTTP2 Cipher Suite, refer to appendix A of [RFC7540] and "backwards-compatibility-security-restrictions" in [I-D.draft-ietf-tls-tls13"].

6.3. PFS

Should the delivery be PFS proof?

6.4. TLS version

Which version of TLS should be supported by LURK: TLS/LTS, TLS1.2, TLS1.3?

6.5. Scalability

For each TLS session initialization on the dCDNs, the dCDNs need to request the KS to get the necessary credentials. To be scalable, the KS may need to be replicated.

6.6. Certificates and security

At least one private key per CSP is stored on the KS. Therefore the use of a KS avoids the complexity of duplicating private keys on uncontrolled servers. This also allows the uCDN to maintain a single domain name for the CDN interconnection.

6.7. Revocation

In the case of an HTTPS delegation revocation, a dCDN has no longer the delegation right to deliver a content for a given CSP. The uCDN would then deny access to CSP certificates and credentials derived from private keys, and therefore the dCDN would not be able to establish the TLS session without triggering a warning on UA Side.

6.8. Certificate provisioning

The dCDN may have to retrieve at least once the CSP public certificate related to the targeted domain name. The certificates may be cached on the dCDN for a given duration. .

Is certificate provisioning in the scope of CDNI as it seems implementation dependant? Which interface provides the certificates to the uCDN (Control, Metadata, LURK, ..)?

6.9. CSP side

With regard to the use case B, the CSP may provide a Key server. As a consequence the requirement [RFC7337]/GEN-3 should be updated accordingly.

6.10. Acquisition

Usage of the LURK interface when acquiring content: when the dCDN acquires content directly from the origin server, would it have to request first the KS to get the uCDN credentials ?

6.11. CDNI Control Interface vs CDNI Metadata Interface

What should be carried by the CI or the MI ?

7. Open issues

7.1. TLS session resuming

Not yet addressed in this document, contributors are welcome.

7.2. Stack Evolution

Contents might be delivered over other protocols than TCP and than HTTP/1.1 in a close future. The CDNI WG must discuss the support of HTTPS delivery over TLS/LTS, TLSv3, DTLS, QUIC and HTTP2.

8. IANA Considerations

This document has no IANA considerations.

9. Security Considerations

The entire document is about security.

10. Acknowledgments

Many thanks to Benoit Gaussen who contributed to this draft.

11. References

11.1. Normative References

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <http://www.rfc-editor.org/info/rfc2818>.

[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>.

[RFC6770] Bertrand, G., Ed., Stephan, E., Burbridge, T., Eardley, P., Ma, K., and G. Watson, "Use Cases for Content Delivery Network Interconnection", RFC 6770, DOI 10.17487/RFC6770, November 2012, <http://www.rfc-editor.org/info/rfc6770>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <http://www.rfc-editor.org/info/rfc7230>.

[RFC7337] Leung, K., Ed. and Y. Lee, Ed., "Content Distribution Network Interconnection (CDNI) Requirements", RFC 7337, DOI 10.17487/RFC7337, August 2014, <http://www.rfc-editor.org/info/rfc7337>.

[RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <http://www.rfc-editor.org/info/rfc7540>.

11.2. Informative References

[I-D.cairns-tls-session-key-interface]
Cairns, K., Mattsson, J., Skog, R., and D. Migault, "Session Key Interface (SKI) for TLS and DTLS", draft-cairns-tls-session-key-interface-01 (work in progress), October 2015.

[I-D.erb-lurk-rsalg]
Erb, S. and R. Salz, "A PFS-preserving protocol for LURK", draft-erb-lurk-rsalg-01 (work in progress), May 2016.

[I-D.ietf-cdni-control-triggers]
Murray, R. and B. Niven-Jenkins, "CDNI Control Interface / Triggers", draft-ietf-cdni-control-triggers-15 (work in progress), May 2016.

[I-D.ietf-cdni-logging]
Faucheur, F., Bertrand, G., Oprescu, I., and R. Peterkofsky, "CDNI Logging Interface", draft-ietf-cdni-logging-27 (work in progress), June 2016.

[I-D.ietf-cdni-metadata]
Niven-Jenkins, B., Murray, R., Caulfield, M., and K. Ma, "CDN Interconnection Metadata", draft-ietf-cdni-metadata-18 (work in progress), June 2016.

[I-D.ietf-cdni-redirection]
Niven-Jenkins, B. and R. Brandenburg, "Request Routing Redirection interface for CDN Interconnection", draft-ietf-cdni-redirection-18 (work in progress), April 2016.

[I-D.ietf-tls-tls13]
Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", draft-ietf-tls-tls13-13 (work in progress), May 2016.

[I-D.mglt-lurk-tls-use-cases]
Migault, D., Ma, K., Salz, R., Mishra, S., and O. Dios, "LURK TLS/DTLS Use Cases", draft-mglt-lurk-tls-use-cases-02 (work in progress), June 2016.

[LURK_Mailing_List]
"LURK Mailing List", <https://mailarchive.ietf.org/arch/search/?email_list=lurk>.

Authors' Addresses

Frederic Fieau (editor)
Orange
40-48, avenue de la Republique
Chatillon 92320
France

Email: frederic.fieau@orange.com

Emile Stephan
Orange
2, avenue Pierre Marzin
Lannion 22300
France

Email: emile.stephan@orange.com