RTCWEB
Internet-Draft
Intended status: Standards Track
Expires: January 24, 2015

M. Thomson
Mozilla

July 23, 2014

Application Layer Protocol Negotiation for Web Real-Time Communications (WebRTC)

draft-ietf-rtcweb-alpn-00

Abstract

Application Layer Protocol Negotiation (ALPN) labels are defined for use in identifying Web Real-Time Communications (WebRTC) usages of Datagram Transport Layer Security (DTLS). Labels are provided for identifying a session that uses a combination of WebRTC compatible media and data, and for identifying a session requiring confidentiality protection.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Web Real-Time Communications (WebRTC) [I-D.ietf-rtcweb-overview] uses Datagram Transport Layer Security (DTLS) [RFC6347] to secure all peer-to-peer communications.

Identifying WebRTC protocol usage with Application Layer Protocol Negotiation (ALPN) [RFC7301] enables an endpoint to positively identify WebRTC uses and distinguish them from other DTLS uses.

Different WebRTC uses can be advertised and behavior can be constrained to what is appropriate to a given use. In particular, this allows for the identifications of sessions that require confidentiality protection.

1.1. Conventions and Terminology

At times, this document falls back on shorthands for establishing interoperability requirements on implementations: the capitalized words "MUST", "SHOULD" and "MAY". These terms are defined in [RFC2119].

2. ALPN Labels for WebRTC

The following identifiers are defined for use in ALPN:

webrtc: The DTLS session is used to establish keys for a Secure Real-time Transport Protocol (SRTP) - known as DTLS-SRTP - as described in [RFC5764]. The DTLS record layer is used for WebRTC data channels [I-D.ietf-rtcweb-data-channel].

c-webrtc: The DTLS session is used for confidential WebRTC communications, where peers agree to maintain the confidentiality of the communications, as described in Section 3.

A more thorough definition of what WebRTC communications entail is included in [I-D.ietf-rtcweb-transports].

Both identifiers describe the same basic protocol: a DTLS session that is used to provide keys for an SRTP session in combination with WebRTC data channels. Either SRTP or data channels MAY be absent. The data channels send Stream Control Transmission Protocol (SCTP) [RFC4960] over the DTLS record layer, which can be multiplexed with SRTP on the same UDP flow. WebRTC requires the use of Interactive Communication Establishment (ICE) [RFC5245] to establish the UDP flow, but this is not covered by the identifier.

A more thorough definition of what WebRTC communications entail is included in [I-D.ietf-rtcweb-transports].

There is no functional difference between the identifiers except with respect to the promise that an endpoint makes with respect to the confidentiality of session content. An endpoint negotiating "c-webrtc" makes a promise to preserve the confidentiality of the data it receives.

Only one of these labels can be used for any given session. A peer acting in the client role MUST NOT offer both identifiers. A peer in the server role that receives a ClientHello containing both labels MUST reject the session, though it MAY accept the confidential option and protect content accordingly.

3. Media Confidentiality

Private communications in WebRTC depend on separating control (i.e., signaling) capabilities and access to media [I-D.ietf-rtcweb-security-arch]. In this way, an application can establish a session that is end-to-end confidential, where the ends in question are user agents (or browsers) and not the signaling application.

A browser is required to enforce this control using isolation controls similar to those used in cross-origin protections. These protections ensure that media is protected from applications. Applications are not able to read or modify the contents of a protected flow of media. Media that is produced from a session using the "c-webrtc" identifier MUST only be displayed to users.

Without some form of indication that is securely bound to the session, a WebRTC endpoint is unable to properly distinguish between session that requires confidentiality protection and one that does not.

A browser is required to enforce confidentiality using isolation controls similar to those used in content cross-origin protections (see Section 5.3 [1] of [HTML5]). These protections ensure that media is protected from applications. Applications are not able to read or modify the contents of a protected flow of media. Media that is produced from a session using the "c-webrtc" identifier MUST only be displayed to users.

Confidentiality protections of this sort are not expected to be possible for data that is sent using data channels. Thus, it is expected that data channels will not be employed for sessions that negotiate confidentiality. In the browser context, confidential data depends on having both data sources and consumers that are exclusively browser- or user-based. No mechanisms currently exist to take advantage of data confidentiality, though some use cases suggest that this could be useful, for example, confidential peer-to-peer file transfer.

Generally speaking, ensuring confidentiality depends on authenticating the communications peer. This mechanism explicitly does not define a specific authentication method; a WebRTC endpoint that accepts a session with this ALPN identifier MUST respect confidentiality no matter what identity is attributed to a peer.

RTP middleboxes and entities that forward media or data cannot promise to maintain confidentiality. Any entity that forwards content, or records content for later access by entities other than the authenticated peer, MUST NOT offer or accept a session with the "c-webrtc" identifier.

4. Security Considerations

Confidential communications depends on more than just an agreement from browsers.

Information is not confidential if it is displayed to those other than to whom it is intended. Peer authentication [I-D.ietf-rtcweb-security-arch] is necessary to ensure that data is only sent to the intended peer.

This is not a digital rights management mechanism. Even with an authenticated peer, a user is not prevented from using other mechanisms to record or forward media. This means that (for example)

screen recording devices, tape recorders, portable cameras, or a cunning arrangement of mirrors could variously be used to record or redistribute media once delivered. Similarly, if media is visible or audible (or otherwise accessible) to others in the vicinity, there are no technical measures that protect the confidentiality of that media. In other cases, effects might not be temporally localized: transmitted smells could linger for a period after communications cease.

The only guarantee provided by this mechanism and the browser that implements it is that the media was delivered to the user that was authenticated. Individual users will still need to make a judgment about how their peer intends to respect the confidentiality of any information provided.

On a shared computing platform like a browser, other entities with access to that platform (i.e., web applications), might be able to access information that would compromise the confidentiality of communications. Implementations MAY choose to limit concurrent access to input devices during confidential communications session.

For instance, another application that is able to access a microphone might be able to sample confidential audio that is playing through speakers. This is true even if acoustic echo cancellation, which attempts to prevent this from happening, is used. Similarly, an application with access to a video camera might be able to use reflections to obtain all or part of a confidential video stream.

5. IANA Considerations

The following two entries are added to the "Application Layer Protocol Negotiation (ALPN) Protocol IDs" registry established by [RFC7301].

The "webrtc" identifies mixed media and data communications using SRTP and data channels:

Protocol: WebRTC Media and Data

Identification Sequence: 0x77 0x65 0x62 0x72 0x74 0x63 ("webrtc")

Specification: This document (RFCXXXX)

The "c-webrtc" identifies confidential WebRTC communications:

Protocol: Confidential WebRTC Media and Data

Identification Sequence: 0x63 0x2d 0x77 0x65 0x62 0x72 0x74 0x63 ("c-webrtc")

Specification: This document (RFCXXXX)

6. References

6.1. Normative References

[I-D.ietf-rtcweb-data-channel]
Jesup, R., Loreto, S., and M. Tuexen, "WebRTC Data Channels", draft-ietf-rtcweb-data-channel-09 (work in progress), May 2014.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5764] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)", RFC 5764, May 2010.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

[RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, July 2014.

6.2. Informative References

[HTML5] Berjon, R., Leithead, T., Doyle Navara, E., O'Connor, E., and S. Pfeiffer, "HTML 5", CR CR-html5-20121217, August 2010, <http://www.w3.org/TR/2012/CR-html5-20121217/>.

[I-D.ietf-rtcweb-overview]
Alvestrand, H., "Overview: Real Time Protocols for Brower-based Applications", draft-ietf-rtcweb-overview-09 (work in progress), February 2014.

[I-D.ietf-rtcweb-security-arch]
Rescorla, E., "WebRTC Security Architecture", draft-ietf-rtcweb-security-arch-09 (work in progress), February 2014.

[I-D.ietf-rtcweb-transports]
Alvestrand, H., "Transports for RTCWEB", draft-ietf-rtcweb-transports-04 (work in progress), April 2014.

[RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, April 2010.

6.3. URIs

[1] http://www.w3.org/TR/2012/CR-html5-20121217/browsers.html#origin

Author's Address

Martin Thomson
Mozilla
331 E Evelyn Street
Mountain View, CA 94041
US

Email: martin.thomson@gmail.com