TTC技術レポート
Technical Report

TR-1107

WebRTC に関する技術報告書
接続管理編

Technical Report on WebReal-Time Communication (WebRTC): Connection Management

第 1 版

2024 年 7 月 4 日制定

一般社団法人
情報通信技術委員会

THE TELECOMMUNICATION TECHNOLOGY COMMITTEE

目　次

＜参考＞ 4
I. 本技術レポートの概要 5
II. RFC8829 原文の著作権について 6
III. RFC8829 の和訳 7
1. はじめに 7
1.1 JSEP の一般的な設計 7
1.2 考慮されるその他のアプローチ 8
1.3 バンドルのみの"m="セクションに関する矛盾 8
2. 用語 9
3. 意味と構文 9
3.1 シグナリングモデル 9
3.2 セッションの説明とステートマシン 9
3.3 セッション記述形式 11
3.4 セッション記述制御 12
3.5 ICE 12
3.6 ビデオサイズのネゴシエーション 15
3.7 Simulcast (同時放送) 17
3.8 分岐との相互作用 17
4. インタフェース 19
4.1 PeerConnection 19
4.2 RtpTransceiver 26
5. SDP 相互作用の手順 27
5.1 要件の概要 28
5.2 オファーの構築 29
5.3 アンサーの生成 36
5.4 オファーまたはアンサーの変更 41
5.5 ローカル記述の処理 42
5.6 リモート記述の処理 42
5.7 ロールバックの処理 42
5.8 セッション記述の解析 43
5.9 ローカル記述の適用 47
5.10 リモート記述の適用 48
5.11 アンサーの適用 50
6. RTP/RTCP の処理 52
7. 例 52
7.1 簡単な例 52
7.2 詳細な例 56
7.3 早期トランスポートウォームアップの例 62
8. セキュリティに関する考慮事項 68
9. IANA に関する考慮事項 69
10. 参考資料 69
10.1 標準参照 69
10.2 参考文献 72
付録 A. SDP ABNF 構文 74

＜参考＞

１．国際勧告等との関連

本技術レポートは、RFC8829 を調査したものである。

２．上記国際勧告等に対する追加項目等

なし

３．改定の履歴

版　数	発　行　日	改　版　内　容
第 1 版	2024 年 7 月 4 日	制定

４．参考文献

[RFC8829]　Rescorla, E., " JavaScript Session Establishment Protocol (JSEP)", RFC 8829, DOI 10.17487/RFC8829, January 2021, <https://www.rfc-editor.org/info/rfc8829>.

５．工業所有権

本技術レポートに関わる「工業所有権等の実施許諾に係る声明書」の提出状況は、TTC ホームページでご覧になれます。

６．技術レポート作成部門

企業ネットワーク専門委員会

I. 本技術レポートの概要

近年、テレワークの推進により、Web 会議システムが急速に普及してきた。Web 会議システムにおいてはパソコンやスマートフォン、タブレットなどデバイスを選ばず、Web ブラウザからアクセスすることにより、いつでもどこでも会議を行うことができるというメリットがある。Web 会議システムの通信プロトコルはシステムによって WebSocket であったり独自仕様であったりと様々なプロトコルが使用されている。その中でも近年特に注目されているのが WebRTC である。

WebRTC はブラウザ同士の双方向通信のために 2012 年に規格が策定され、様々な Web ブラウザで実装されてきた。その後テレワークの推進により、さらに注目を浴び、2021 年に IETF による標準化が行われた。

本報告書では TR-1095、TR-1101、TR-1102 に引き続き、IETF によって標準化された WebRTC に関する以下の RFC について日本語に翻訳する。

- RFC8829：WebRTC の接続管理

TR-1095、TR-1101、TR-1102 にて翻訳した RFC を以下に記載する。

技術レポート	RFC	タイトル
TR-1095 (技術報告書)	RFC8825	IETF によって標準化された WebRTC の仕様の概要
	RFC8834	WebRTC で使用される RTP の取り決め
TR-1101 (データ転送編)	RFC8835	WebRTC で使用されるデータ転送プロトコル
	RFC8828	WebRTC 実装における IP アドレスの処理方法 (プライバシーとメディアパフォーマンスのトレードオフの処理方法)
	RFC8831	WebRTC コンテキストにおける Stream Control Transmission Protocol (SCTP) の使用方法
	RFC8832	WebRTC データチャネル確立プロトコル
TR-1102 (セキュリティ編)	RFC8826	WebRTC のセキュリティに関する考慮事項
	RFC8827	WebRTC セキュリティアーキテクチャ

II. RFC8829 原文の著作権について

III. RFC8829 の和訳

RFC 8829　WebRTC の接続管理

概要

このドキュメントでは、JavaScript アプリケーションが W3C RTCPeerConnection API で指定されたインタフェースを介してマルチメディアセッションのシグナリングプレーンを制御できるようにするメカニズムについて説明し、これが既存のシグナリングプロトコルとどのように関連するかについて説明する。

1.　はじめに

このドキュメントでは、W3C Web Real-Time Communication (WebRTC) RTCPeerConnection インタフェース [W3C.webrtc] を使用して、マルチメディアセッションのセットアップ、管理、およびティアダウン（各要素ごとに分解解析）を制御する方法について説明する。

1.1　JSEP の一般的な設計

WebRTC のコールセットアップは、メディアプレーンの制御に重点を置いて設計されており、シグナルプレーンの動作は可能な限りアプリケーションに委ねられている。その理論的根拠は、異なるアプリケーションが、既存の SIP コールシグナルプロトコルなどの異なるプロトコルを使用することを好む場合や、おそらく新しいユースケースのために、特定のアプリケーションにカスタムされたものを使用する場合があるということ。このアプローチでは、交換する必要がある重要な情報は、メディアプレーンを確立するために必要なトランスポートとメディア構成情報を指定するマルチメディアセッションの説明である。

これらの考慮事項を考慮して、このドキュメントでは、JavaScript からシグナリングステートマシンを完全に制御できる JavaScript Session Establishment Protocol (JSEP) について説明する。前述のように、JSEP は、WebRTC API を含むランタイム内で JavaScript アプリケーションが実行されるモデル (「JSEP 実装」) を想定している。JSEP 実装は、コアのシグナリングフローとはほぼ完全に分離されており、代わりに、(1) ローカルおよびリモートセッションの記述を渡すこと、および (2) Interactive Connectivity Establishment (ICE) ステートマシン [RFC8445] との対話という 2 つのインタフェースを使用する JavaScript によって処理される。このドキュメントでは、JSEP 実装と JavaScript アプリケーションの組み合わせを「JSEP エンドポイント」と呼ぶ。

このドキュメントでは、JSEP の使用は常に 2 つの JSEP エンドポイント間で行われるように記述されている。ただし、多くの場合、実際には JSEP エンドポイントとゲートウェイや Multipoint Control Unit (MCU) などの何らかのサーバ間で行われることに注意すること。この区別は、JSEP エンドポイントからは見えない。API 経由で与えられる指示に従うだけである。

JSEP のセッション記述の扱いは単純である。オファー/アンサーの交換が必要な場合、開始側は createOffer API を呼び出すことでオファーを作成する。その後、アプリケーションはそのオファーを使用して、setLocalDescription API を介してローカル構成を設定する。

オファーは最終的に、優先されるシグナリングメカニズム(例:WebSocket)を介してリモート側に送信される。そのオファーを受信すると、リモート側は setRemoteDescription API を使用してそれをインストールする。

オファー/アンサーの交換を完了するために、リモート側は createAnswer API を使用して適切なアンサーを生成し、setLocalDescription API を使用してそれを適用し、シグナルチャネルを介して発信側にアンサーを返す。発信側がそのアンサーを取得すると、setRemoteDescription API を使用してインストールし、初期設定が完了する。このプロセスは、追加のオファー/アンサー交換で繰り返すことができる。

ICE [RFC8445] に関して、JSEP は ICE ステートマシンをシグナリングステートマシン全体から切り離す。ICE ステートマシンは、実装のみが候補やその他のトランスポート情報の必要な知識を持っているため、JSEP 実装に残る必要がある。この分離を実行すると、セッション記述をトランスポートから分離するプロトコルの柔軟性が高まる。たとえば、従来の SIP では、各オファーまたはアンサーは、セッション記述とトランスポート情報の両方を含む自己完結型である。ただし、[RFC8840] では、SIP を Trickle ICE [RFC8838] で使用できる。この場合、セッション記述はすぐに送信でき、トランスポート情報は利用可能な場合に送信できる。トランスポート情報を個別に送信すると、すべてのトランスポート情報を待つのではなく、任意のトランスポート情報が利用可能になるとすぐに ICE チェックを開始できるため、ICE と DTLS の起動を高速化できる。JSEP は ICE とシグナリングステートマシンを分離しているため、どちらのモデルにも対応できる。

これはシグナリングを抽象化するが、JSEP のアプローチでは、アプリケーションがシグナリングプロセスを認識する必要がある。アプリケーションは、コールを設定するためにセッション記述の内容を理解する必要はないが、適切なタイミングで適切な API を呼び出し、セッション記述と ICE 情報を選択したシグナリングプロトコルの定義済みメッセージに変換し、相手側から受信したメッセージに対して逆変換を実行する必要がある。

アプリケーションの作業を容易にする 1 つの方法は、開発者からこの複雑さを隠す JavaScript ライブラリを提供すること。このライブラリは、ステートマシンとシリアライゼーションコードとともに所定のシグナリングプロトコルを実装し、アプリケーション開発者に高レベルの呼び出し指向インタフェースを提供する。たとえば、JSEP API 上で SIP [RFC3261] および Extensible Messaging and Presence Protocol (XMPP) [RFC6120] シグナリングプロトコルの実装を提供するライブラリが存在する。したがって、JSEP は経験豊富な開発者により大きな制御を提供し、初心者開発者にさらなる複雑さを強いることはない。

1.2 考慮されるその他のアプローチ

JSEP の代わりに検討されたアプローチの 1 つは、軽量のシグナリングプロトコルを含めることであった。API にセッション記述を提供する代わりに、API はこのプロトコルからメッセージを生成して消費する。より高レベルの API を提供する一方で、これにより JSEP 実装内のシグナリングの制御が強化され、シグナリンググレア([RFC3264] の 4 章を参照)などの概念を理解して処理する必要が生じた。

検討されたが選択されなかった 2 番目のアプローチは、メディア制御オブジェクトの管理とセッション記述を分離し、代わりに各コンポーネントを直接制御する API を提供することであった。これは、アプリケーションプログラマにこのレベルの複雑さを公開することを要求することは有益ではないという議論に基づいて拒否された。その結果、(1) 単純な例であっても、必要なすべての相互作用を調整するためにかなりの量のコードを必要とする API が作成され、(2) 合意と文書化が必要な大規模な API サーフェスが作成される。さらに、これらの API ポイントは任意の順序で呼び出すことができ、その結果、セッション記述の評価と適用方法を指定する JSEP アプローチよりも複雑なメディアサブシステムとの対話のセットになった。

検討された JSEP のバリエーションの 1 つは、基本的なセッション記述指向 API を維持しながら、オファーとアンサーを生成するメカニズムを JSEP 実装から移動することだった。このアプローチでは、実装内で createOffer/createAnswer メソッドを提供する代わりに、getCapabilities API を公開し、独自のセッション記述を生成するために必要な情報をアプリケーションに提供する。これにより、アプリケーションが行う必要のある作業量が増加する。機能からセッション記述を生成する方法、特に任意のオファーとサポートされている機能から正しい答えを生成する方法を知っている必要がある。これは確かに上記のようなライブラリを使用することで対処できるが、単純な例であっても基本的にはそのライブラリの使用を強制する。createOffer/createAnswer を提供することで、この問題を回避できる。

1.3 バンドルのみの"m="セクションに関する矛盾

WebRTC 仕様文書が承認されて以来、IETF は JSEP を指定する文書と BUNDLE を指定する文書(それぞれ、

この RFC と [RFC8843])の間の不整合を認識するようになった。解決のために公開をさらに遅らせるのではなく、文書は当初承認された通りに公開されている。IETF はこれらの技術に関する作業を再開する予定であり、これらの文書の改訂版は、解決策が得られ次第公表される予定である。

具体的な問題には、この RFC の 4.1.1 項で議論されているように、バンドル専用として指定された"m="セクションの処理が含まれる。現在、JSEP と BUNDLE の間、およびこれらの仕様と既存のブラウザ実装の間には相違がある。

- JSEP は、前述の"m="セクションは、最初のオファーではポート 0 を使用し、"a=bundle-only"属性を追加する必要があるが、アンサーや後続のオファーでは追加しないように規定している。
- BUNDLE は、これらの"m="セクションは、前のポイントで説明したようにマークする必要があるが、すべてのオファーとアンサーでマークする必要があると規定している。
- 現在のほとんどのブラウザは、どの"m="セクションもポート 0 でマークせず、代わりにすべてのバンドルされた"m="セクションに同じポートを使用している。JSEP の動作に従うものもある。

2. 用語

このドキュメントのキーワード"MUST"、"MUST NOT"、"REQUIRED"、"SHALL"、"SHALL NOT"、"SHOULD"、"SHOULD NOT"、"RECOMMENDED"、"NOT RECOMMENDED"、"MAY"、および"OPTIONAL"は、ここに示すように、すべて大文字で表示される場合にのみ、BCP 14 [RFC2119] [RFC8174] に記載されているように解釈される。

3. 意味と構文

3.1 シグナリングモデル

JSEP は、セッションの両側がセッションの実施方法を知るために、[RFC3264] (オファー/アンサー) で記述されている方法でセッション記述を交換する一般的な必要性以外に、特定のシグナリングモデルやステートマシンを指定しない。JSEP は、オファーとアンサーを作成し、セッションに適用するためのメカニズムを提供する。ただし、JSEP の実装は、アドレッシング、再送信、フォーク、グレア処理など、これらのオファーとアンサーがリモート側に伝達される実際のメカニズムから完全に分離されている。これらの問題は完全にアプリケーションに委ねられている。アプリケーションは、どのオファーとアンサーがいつ実装に渡されるかを完全に制御する。

図 1　JSEP シグナリングモデル

3.2 セッションの説明とステートマシン

メディアプレーンを確立するために、JSEP の実装では、リモート側に送信する内容と、受信したメディアの処理方法を示す特定のパラメータが必要である。これらのパラメータは、オファーとアンサーでセッションの説明を交換することによって決定され、このプロセスには JSEP API で処理する必要がある特定の詳細がある。

セッションの説明がローカル側とリモート側のどちらに適用されるかは、その説明の意味に影響する。た

とえば、リモートパーティに送信されるコーデックのリストは、ローカル側が受信するものを示し、リモート側がサポートするコーデックのセットと交差するときに、リモート側が送信するものを指定する。ただし、すべてのパラメータがこの規則に従うわけではない。一部のパラメータは宣言型であり、リモート側はそれらを受け入れるか、完全に拒否する必要がある。このようなパラメータの例として、DTLS [RFC6347] のコンテキストで使用される TLS フィンガープリント [RFC8122] がある。これらのフィンガープリントは提供されたローカル証明書に基づいて計算され、ネゴシエーションの対象にはならない。

さらに、さまざまな RFC はオファーとアンサーの形式に異なる条件を付けている。たとえば、オファーは任意の数の"m="セクション(すなわち、[RFC4566] の 5.14 節に記載されているメディアの説明)を提案する場合があるが、アンサーにはオファーとまったく同じ数が含まれている必要がある。

最後に、正確なメディアパラメータはオファーとアンサーが交換された後にのみ認識されるが、オファー側は ICE チェックを受け取る場合があり、場合によってはアンサーを受け取る前にメディア(例:コネクション確立後の再オファーの場合)を受け取る場合がある。この場合に着信メディアを適切に処理するには、オファー側のメディアハンドラが、アンサーが到着する前にオファーの詳細を認識している必要がある。

したがって、セッション記述を適切に処理するために、JSEP 実装では次のことが必要である。

1. セッション記述がローカル側とリモート側のどちらに関連しているかを知ること。
2. セッション記述がオファーかアンサーかを知ること。
3. アンサーとは無関係にオファーを指定できるようにすること。

JSEP は、setLocalDescription メソッドと setRemoteDescription メソッドの両方を追加し、セッション記述オブジェクトに提供されるセッション記述のタイプを示す type フィールドを含めることで、これに対処する。これは、最初に setLocalDescription (sdp [offer]) を呼び出し、その後 setRemoteDescription (sdp [answer]) を呼び出すオファー側と、最初に setRemoteDescription (sdp [offer]) を呼び出し、その後 setLocalDescription (sdp [answer]) を呼び出すアンサー側の両方について、上記の要件を満たす。

オファー/アンサーの交換中、未処理のオファーは、受け入れまたは拒否される可能性があるため、オファー側とアンサー側で「保留中」と見なされる。これが再オファーの場合、各側には、最後のオファー/アンサー交換の結果を反映する「現在の」ローカルおよびリモートの説明もある。4.1.14 項、4.1.16 項、4.1.13 項、および 4.1.15 項に、保留中および現在の説明の詳細が記載されている。

JSEP では、アンサーをアプリケーションによって暫定的なものとして扱うこともできる。暫定的なアンサーは、アンサー側がセッションの開始を許可するために、最初のセッションパラメータをオファー側に通知する方法を提供し、最終的なアンサーを後で指定できるようにする。この最終的なアンサーの概念は、オファー/アンサーモデルにとって重要である。このようなアンサーを受信すると、正確なセッション構成がわかったので、呼び出し元によって割り当てられた余分なリソースを解放できる。これらの「リソース」には、追加の ICE コンポーネント、Traversal Using Relays around NAT (TURN) 候補、ビデオデコーダなどが含まれる。一方、暫定的なアンサーでは、そのような割り当て解除は行われない。その結果、独自のコーデック選択、トランスポートパラメータなどを持つ複数の異なる暫定的なアンサーを、コールセットアップ中に受信して適用できる。最終的なアンサー自体が、受信したどの暫定的なアンサーとも異なる場合があることに注意すること。

[RFC3264] では、シグナリングレベルでの制約は、特定のセッションに対して未処理のオファーは 1 つだけであるが、JSEP レベルでは、任意の時点で新しいオファーを生成できることである。たとえば、シグナリングに SIP を使用する場合、1 つのオファーが送信され、その後 SIP CANCEL を使用してキャンセルされると、最初のオファーに対するアンサーが受信されなかったにもかかわらず、別のオファーを生成できる。これをサポートするために、JSEP メディア層は、JavaScript アプリケーションがシグナリングにオファーを必要とするたびに、createOffer メソッドを介してオファーを提供できる。アンサー側は、0 個以上の暫定的なア

ンサーを返送し、最終的なアンサーを送信することで、最終的にオファー/アンサーの交換を終了できる。このためのステートマシンは次のとおりである。

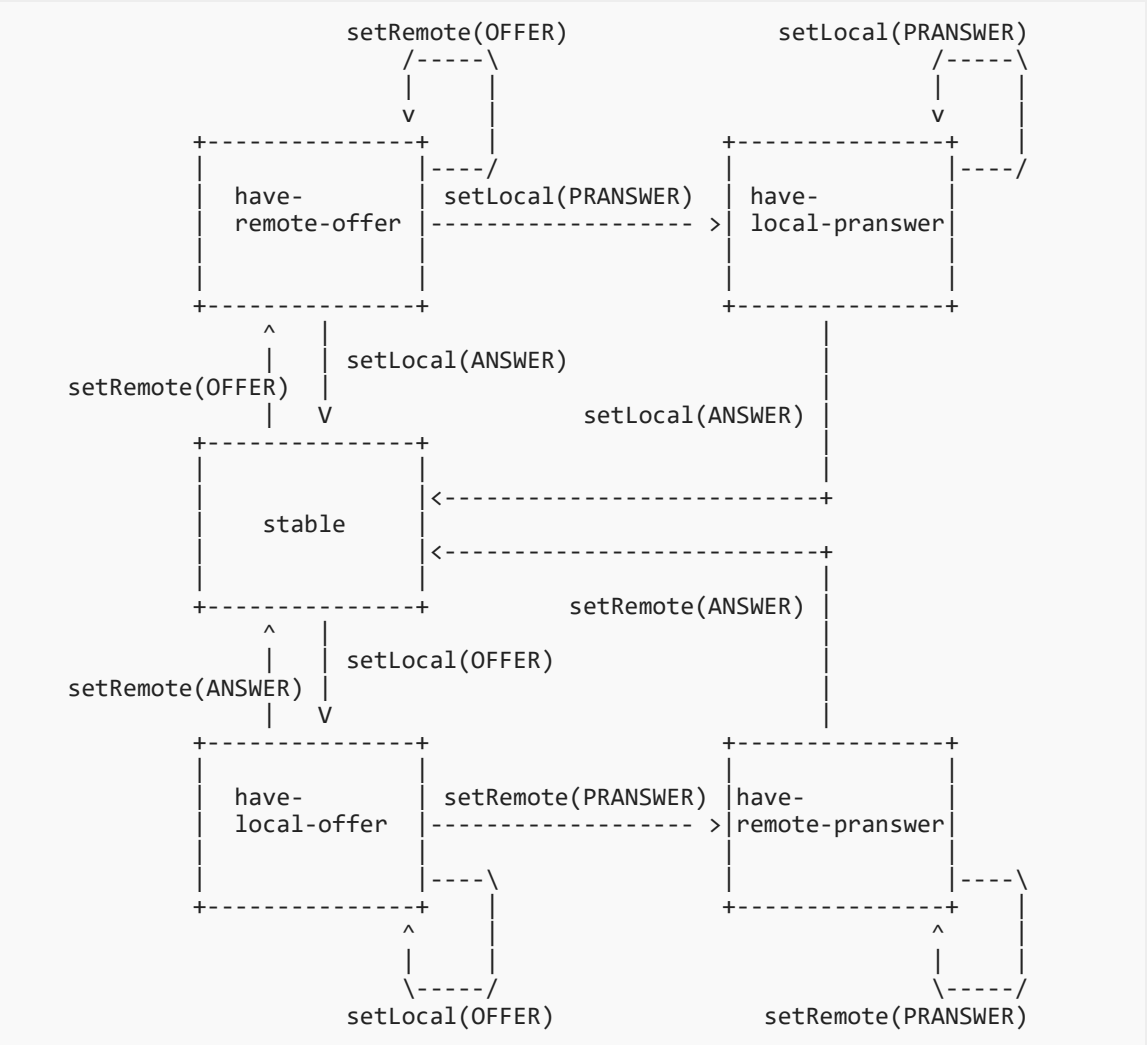

図 2　JSEP ステートマシン

これらの状態遷移以外に、暫定的なアンサー ("pranswer") と最終的なアンサー ("answer") の処理に違いはない。

3.3　セッション記述形式

JSEP のセッション記述では、内部表現に Session Description Protocol (SDP) 構文を使用する。この形式は JavaScript からの操作には最適ではないが、広く受け入れられており、新機能で頻繁に更新されている。セッション記述の代替エンコードは、少なくともこの新しいエンコードが SDP の人気を上回るまでは、SDP の変更に歩調を合わせる必要がある。

ただし、将来の柔軟性を提供するために、SDP 構文は SessionDescription オブジェクト内にカプセル化されており、SDP から構築して SDP にシリアル化できる。将来の仕様でセッション記述のための JSON 形式が合意されれば、このオブジェクトがその JSON を生成して使用できるように簡単にすることができる。

以下で詳しく説明するように、ほとんどのアプリケーションは、これらのさまざまな API 呼び出しによって生成および消費されるセッション記述を不透明な blob として扱うことができるはずである。つまり、アプリケーションはそれらを読み込んだり変更したりする必要はない。

3.4　セッション記述制御

アプリケーションがさまざまな共通セッションパラメータを制御できるように、JSEP にはセッション記述の生成方法を JSEP 実装に指示するコントロールサーフェスが用意されている。これにより、ほとんどの場合、JavaScript でセッション記述を変更する必要がなくなる。

これらのオブジェクトを変更すると、後続の createOffer/createAnswer 呼び出しによって生成されるセッション記述が変更される。

3.4.1 RtpTransceivers

RtpTransceivers を使用すると、1 つの"m="セクションに関連付けられた RTP メディアをアプリケーションで制御できる。各 RtpTransceiver には RtpSender と RtpReceiver があり、アプリケーションはこれを使用して RTP メディアの送受信を制御できる。アプリケーションは、RtpTransceiver を停止するなどして、RtpTransceiver を直接変更することもできる。

RtpTransceiver は通常、"m="セクションと 1:1 マッピングを持つ。ただし、RtpTransceiver が作成されてもまだ"m="セクションに関連付けられていない場合、または RtpTransceiver が停止していて、"m="セクションから切り離されている。RtpTransceiver のメディア識別 (mid) プロパティが null でない場合、RtpTransceiver は"m="セクションに関連付けられていると言われている。関連付けられた"m="セクションは、オファーの作成時またはリモートオファーの適用時に形成される、トランシーバ間のマッピングと"m="セクションインデックスを使用して決定される。

RtpTransceiver は複数の"m="セクションに関連付けられることはなく、セッション記述が適用されると、"m="セクションは常に正確に 1 つの RtpTransceiver に関連付けられる。ただし、後述の 5.2.2 項で説明するように、"m="セクションが拒否された特定のケースでは、その"m="セクションは"リサイクル"され、新しい MID 値を持つ新しい RtpTransceiver に関連付けられる。

RtpTransceivers は、アプリケーションによって明示的に作成することも、新しい"m="セクションを追加するオファーを使用して setRemoteDescription を呼び出すことによって暗黙的に作成することもできる。

3.4.2.RtpSenders

RtpSenders を使用すると、RTP メディアの送信方法をアプリケーションで制御できる。RtpSender は、概念的には"m="セクションで記述された送信 RTP ストリームを担当する。これには、添付された MediaStreamTrack のエンコード、RTP メディアパケットの送信、送信 RTP ストリームの RTP Control Protocol (RTCP) の生成/処理が含まれる。

3.4.3.RtpReceivers

RtpReceivers を使用すると、アプリケーションは RTP メディアの受信方法を検査できる。RtpReceiver は、概念的には"m="セクションで記述された着信 RTP ストリームを担当する。これには、受信した RTP メディアパケットの処理、着信ストリームのデコードによるリモート MediaStreamTrack の生成、着信 RTP ストリームの RTCP の生成/処理が含まれる。

3.5　ICE

3.5.1 ICE ギャザリングの概要

JSEP は、アプリケーションの必要に応じて ICE 候補を収集する。ICE 候補の収集は収集フェーズと呼ばれ、これは、ローカルセッションの説明に新しいまたはリサイクルされた"m="セクションを追加するか、説明に新しい ICE 資格情報を追加することによってトリガーされ、ICE の再起動を示す。新しい ICE 資格情報の使用は、アプリケーションによって明示的にトリガーされることも、ICE 構成の変更に応じて JSEP 実装に

よって暗黙的にトリガーされることもある。

新しい収集フェーズを必要とするように ICE 構成が変更されると、'needs-icerestart'ビットが設定される。このビットが設定されると、createOffer API の呼び出しによって新しい ICE 資格情報が生成される。このビットは、オファーまたはアンサー、つまりローカルまたはリモートで開始された ICE 再起動のいずれかから、新しい ICE 資格情報を使用して setLocalDescription API を呼び出すことによってクリアされる。

新しい収集フェーズが開始されると、ICE エージェントは状態変更イベントを通じて収集が発生していることをアプリケーションに通知する。その後、新しい各 ICE 候補が使用可能になると、ICE エージェントは onicecandidate イベントを介してその候補をアプリケーションに提供する。これらの候補は、現在および/または保留中のローカルセッションの説明にも自動的に追加される。最後に、すべての候補が収集されると、最終的な onicecandidate イベントがディスパッチされ、収集プロセスが完了したことが通知される。

収集フェーズでは、新規/リサイクル/再起動"m="セクションで必要な候補のみが収集されることに注意すること。他の"m="セクションでは、既存の候補が引き続き使用される。また、"m="セクションがバンドルされている場合 (バンドルネゴシエーションの成功またはバンドルのみとしてマークされている場合) 、[RFC8843] で説明されているように、その MID 項目が BUNDLE タグである場合に限り、候補が収集され、その"m="セクションと交換される。

3.5.2 ICE 候補トリクルリング

候補トリクルリングは、最初のオファーがディスパッチされた後に、発信者が着信者に候補を段階的に提供できる手法である。"Trickle ICE"のセマンティクスは [RFC8838] で定義されている。このプロセスにより、呼び出し元がすべての候補を収集するのを待たずに、呼び出し先が呼び出しに基づいて動作を開始し、ICE (およびおそらく DTLS) 接続をすぐに設定できるようになる。これにより、呼び出しを開始する前に収集が実行されない場合のメディアセットアップが高速になる。

JSEP は、前述のように、ICE 候補収集プロセスの制御とフィードバックを提供する API を提供することで、オプションの候補トリクルをサポートしている。候補トリクルをサポートするアプリケーションは、最初のオファーをすぐに送信し、新しい候補者の通知を受けたときに個々の候補者を送信できる。この機能をサポートしていないアプリケーションは、単に収集が完了したことを示すのを待ってから、その時点ですべての候補者を含むオファーを作成して送信できる。

トリクルされた候補者を受信すると、受信側アプリケーションは ICE エージェントにそれらを提供する。これにより、ICE エージェントは新しいリモート候補を使用して接続チェックを開始する。

3.5.2.1 ICE 候補フォーマット

JSEP では、ICE 候補は IceCandidate オブジェクトによって抽象化され、セッション記述と同様に内部表現に SDP 構文が使用される。

候補の詳細は、[RFC8839] の 5.1 節で定義されている"candidate-attribute"フィールドと同じ SDP 構文を使用して、IceCandidate フィールドで指定される。次の例に示すように、このフィールドには"a="プレフィックスが含まれていないことに注意すること。

```
candidate:1 1 UDP 1694498815 192.0.2.33 10000 typ host
```

IceCandidate オブジェクトには、[RFC8839] の 5.4 節で定義されているように、関連付けられている ICE ユーザ名フラグメント (ufrag) を示すフィールドが含まれている。この値は、この IceCandidate がどのセッション記述 (およびそれによるどの収集フェーズ) に属するかを決定するために使用され、ICE 再起動時のあいまいさの解決に役立つ。受信した IceCandidate にこのフィールドが存在しない場合 (JSEP 以外のエンドポイントと通信する場合など) 、最後に受信したセッション記述が想定される。

IceCandidate オブジェクトには、どの"m="セクションに関連付けられているかを示すフィールドも含まれている。これは、"m="セクションインデックスまたは MID のいずれかの方法で識別できる。"m="セクションインデックスは 0 から始まるインデックスで、インデックス N はこの IceCandidate によって参照されるセッション記述の N+1 番目の"m="セクションを参照する。MID は、[RFC5888] の 4 章で定義されている「メディアストリーム識別」値であり、関連する RtpTransceiver オブジェクト(これは、後述の 5.10 節で説明するように、MID 属性をサポートしない JSEP 以外のエンドポイントと対話するときに、アンサー側によってローカルに生成された可能性がある)の MID を使用して、セッション記述の "m="セクションを識別するより堅牢な方法を提供する。https://www.rfc-editor.org/rfc/rfc5888MID フィールドが受信した IceCandidate に存在する場合は、識別に使用する必要がある[MUST]。それ以外の場合は、代わりに"m="セクションインデックスが使用される。

実装は、前述のように、いくつかのフィールドが欠落しているオブジェクトを受信する準備が必要である[MUST]。

3.5.3 ICE 候補ポリシー

通常、ICE 候補を収集する場合、JSEP 実装では、ホスト、サーバ再帰、リレーなど、初期候補の可能なすべての形式が収集される。ただし、プライバシーや関連する懸念のために、アプリケーションが収集プロセスをより具体的に制御したい場合がある。たとえば、(この選択には対応する運用コストが伴うことを念頭に置いて) できるだけ少ない位置情報をリークするために、リレー候補のみを使用したい場合がある。これを実現するために、JSEP では、セッションで使用される ICE 候補をアプリケーションで制限できる。このフィルタリングは、[RFC8828] で説明されているように、アプリケーションに許可される IP アドレスに関して実装が強制することを選択した制限に加えて適用されることに注意すること。

また、セッションがアクティブな間に使用される候補のタイプをアプリケーションが変更する場合もある。主な例として、着信者が任意の発信者に位置情報が漏れるのを避けるために、最初はリレー候補のみを使用したい場合があるが、ユーザが通話を受けることを示すと、(運用コストを下げるために) すべての候補を使用するように変更する場合がある。このシナリオでは、JSEP 実装は、前述のローカルポリシーとの相互作用に応じて、候補ポリシーをセッションの途中で変更できるようにする必要がある[MUST]。

次に、収集フェーズ中、実装は、現在のポリシーで禁止されている候補をアプリケーションに公開したり、接続チェックのソースとして使用したり、他の ICE 候補の raddr/rport 属性などの他のフィールドを介して間接的に公開したりしてはならない[MUST NOT]。後で、アプリケーションによって別のポリシーが指定された場合、アプリケーションは ICE の再起動を介して新しい収集フェーズを開始することによって、そのポリシーを適用できる。

3.5.4 ICE 候補プール

JSEP アプリケーションは通常、setLocalDescription に提供される情報を介して ICE の収集を開始するように JSEP 実装に通知する。これは、ローカルの説明が必要となる ICE コンポーネントの数と、どの候補を収集する必要があるかを示しているためである。ただし、アプリケーションが使用する ICE コンポーネントの数を事前に把握しているケースを加速するために、実装に対して潜在的な ICE 候補のプールを収集して、迅速なメディアセットアップを確実にするように依頼する場合がある。

最終的に setLocalDescription が呼び出され、JSEP 実装が必要な ICE 候補を収集する準備をするとき、候補がプールで使用可能かどうかを確認することから始める必要がある[SHOULD]。プールに候補がある場合、ICE 候補イベントを介してすぐにアプリケーションに渡される必要がある[SHOULD]。予想よりも多くの ICE コンポーネントが使用されているか、候補を収集するための十分な時間がプールになかったためにプールが枯渇した場合、残りの候補は通常どおり収集される。これは最初のオファー/アンサー交換でのみ発生し、その後候補プールは空になり、使用されなくなる。

この概念が役立つ例の 1 つは、将来のある時点での着信を想定し、初期メディアのクリッピングを回避するために接続の確立にかかる時間を最小限に抑えたいアプリケーションである。候補者をプールに事前に収集することで、通話の受信とほぼ同時に、これらの候補者からの接続チェックを交換して送信を開始できる。ただし、これらの事前に収集された候補者を保持することで、それらが必要になる可能性がある限り保持され、アプリケーションは使用している STUN/TURN サーバのリソースを消費することに注意すること。("STUN"は"Session Traversal Utilities for NAT"の略である。)

3.5.5 ICE のバージョン

この仕様は正式には [RFC8445] に依存しているが、公開時点では、WebRTC 実装の大部分は [RFC5245] で説明されている ICE のバージョンをサポートしている。[RFC8445] で定義されている"ice2"属性は、リモートエンドポイントで使用されているバージョンを検出し、古い仕様から新しい仕様へのスムーズな移行を提供するために使用できる。実装は、"ice2"属性を持たないリモート記述を受け入れることができる必要がある[MUST]。

3.6 ビデオサイズのネゴシエーション

ビデオサイズのネゴシエーションは、受信側が"a=imageattr" SDP 属性 [RFC6236] を使用して、受信可能なビデオフレームサイズを示すプロセスである。受信側では、ビデオデコーダが処理できる内容にハードリミットが設定されている場合や、ポリシーによってある程度の最大値が設定されている場合がある。これらの制限を"a=imageattr"属性で指定することで、JSEP エンドポイントは、リモート送信側が許容できる解像度でビデオを送信するように試みることができる。ただし、この属性を理解していない JSEP 以外のエンドポイントと通信する場合、信号による制限を超える可能性があり、JSEP 実装では、ビデオを破棄するなどして、これを適切に処理する必要がある[MUST]。

なお、一部のコーデックでは、アスペクト比が 1.0 以外のサンプルの伝送がサポートされている(すなわち、非正方形ピクセル)。JSEP の実装では、非正方形ピクセルは伝送されないが、正しいアスペクト比でそのようなビデオを受信してレンダリングする必要がある[SHOULD]。ただし、サンプルのアスペクト比は、以下で説明するサイズのネゴシエーションには影響しない。正方形であるかどうかにかかわらず、すべての寸法はピクセル単位で測定される。

3.6.1 imageattr 属性の作成

受信側はまず、既知のローカル制限(例:ハードウェアデコーダ機能またはローカルポリシー)を組み合わせて、受信できる絶対最小サイズと最大サイズを決定する。既知のローカル制限がない場合は、"a=imageattr"属性を省略する必要がある[SHOULD]。これらのローカル制限によってビデオを受信できない場合、つまり、許可された解像度がない縮退した場合は、"a=imageattr"属性を省略し[MUST]、必要に応じて"m="セクションを sendonly/inactive としてマークする必要がある[MUST]。

それ以外の場合、"a=imageattr"属性は"recv"方向で作成され、前述の交差から形成される結果の解像度スペースを使用して、その最小値と最大値"x="および"y="の値を指定する。

ここでの規則は 1 つの設定セットを表しているため、"a=imageattr" "q="値は重要ではない。"1.0"に設定する必要がある[MUST]。

"a=imageattr"フィールドはペイロードタイプ固有である。サポートされているすべてのビデオコーデックが同じ機能を持つ場合は、ワイルドカードペイロードタイプ (*) を使用した単一の属性の使用が推奨される[RECOMMENDED]。ただし、サポートされているビデオコーデックの制限が異なる場合は、ペイロードタイプごとに特定の"a=imageattr"属性を挿入する必要がある[MUST]。

例として、48x48 から 720p までの任意の解像度をデコードできるマルチフォーマットビデオデコーダを備えたシステムを考える。この場合、実装によって次の属性が生成される。

```
a=imageattr:* recv [x=[48:1280],y=[48:720],q=1.0]
```

この宣言は、受信側が 48x48 から 1280x720 ピクセルまでの任意のイメージ解像度をデコードできることを示す。

3.6.2 imageattr 属性の解釈

[RFC6236] は"a=imageattr"を勧告フィールドと定義している。これは、送信者が使用できるビデオ形式を絶対的に制限するのではなく、優先される値を示すことを意味している。

この仕様は、より具体的な動作を規定している。特定の解像度 (「トラック解像度」) のビデオを生成している MediaStreamTrack が、同じまたは低い解像度 (「エンコーダ解像度」) でトラックビデオをエンコードしている RtpSender に接続されており、送信者を参照し、有効な"a=imageattr recv"属性を含むリモート記述が適用されている場合、送信者が属性で指定されたサイズ基準を超える解像度を送信しないように、以下のルールに従う必要がある[MUST]。これらのルールは、トラックがその解像度を変更する場合や別のトラックに置き換えられる場合を含め、属性がリモート記述に存在し続ける限り従う必要がある[MUST]。

RtpSender の構成方法によっては、特定の解像度で 1 つのエンコードを生成する場合もあれば、Simulcast (3.7 節)がネゴシエートされている場合は、複数のエンコードを生成し、それぞれが独自の特定の解像度で生成する場合もある。さらに、構成によっては、各エンコードが必要に応じて解像度を下げる柔軟性を備えている場合や、特定の出力解像度にロックされている場合もある。

RtpSender によって生成される各エンコードについて、リモート記述の対応する"m="セクションの"a=imageattr recv"属性のセットが処理され、何を送信するかが決定される。エンコード用に選択されたメディア形式を参照する属性のみが考慮される。このような各属性は、"q="値が最も高い属性から順に個別に評価される。複数の属性が同じ"q="値を持つ場合、それらは含まれている"m="セクションに表示される順序で評価される。JSEP エンドポイントには、メディア形式ごとに最大 1 つの"a=imageattr recv"属性が含まれるが、JSEP エンドポイントは、そのような複数の属性を含む"m="セクションを持つ JSEP 以外のエンドポイントからセッション記述を受け取る場合があることに注意すること。

各"a=imageattr recv"属性には、次のルールが適用される。この処理が成功すると、それに応じてエンコードが送信され、そのエンコードに対してそれ以上の属性は考慮されない。それ以外の場合は、前述の順序で次の属性が評価される。指定された属性のいずれも正常に処理できない場合、エンコーディングは送信されてはならず[MUST NOT]、アプリケーションにエラーが発生する必要がある[SHOULD]。

- 属性からの制限は、エンコーダの解像度と比較される。以下に記載されている特定の制限のみが考慮される。画像の縦横比などの他の値は無視する必要がある[MUST]。回転されたビデオを生成する MediaStreamTrack を検討する場合、回転されていない解像度をチェックに使用する必要がある[MUST]。これは、受信側が受信側の回転(例:Coordination of Video Orientation (CVO) [TS26.114])の実行をサポートしているかどうかに関係なく必要である。これは、マッチングロジックが大幅に簡略化されるためである。
- 受信側が非正方形ピクセルを受信することを示す"1.0"以外の値に設定された"sar=" (サンプルアスペクト比) 値が属性に含まれている場合、これは満たされず、属性は使用できない[MUST NOT]。
- エンコーダの解像度が属性で許可されている最大サイズを超え、エンコーダがその解像度を調整できる場合、エンコーダは制限を満たすためにダウンスケーリングを適用する必要がある[SHOULD]。ダウンスケーリングは、丸めによる些細な違いを無視して、エンコードの画像アスペクト比を変更してはならない[MUST NOT]。たとえば、エンコーダの解像度が 1280x720 で、属性に最大 640x480 が指定されている場合、予想される出力解像度は 640x360 になる。ダウンスケーリングを適用できない場合は、属性を使用してはならない[MUST NOT]。
- エンコーダの解像度が属性で許可されている最小サイズ未満の場合は、属性を使用してはならない

[MUST NOT]。エンコーダはアップスケーリングを適用してはならない[MUST NOT]。JSEP 実装は、おそらくソフトウェアデコーダへのフォールバックを介して、任意の小さな解像度の受信を許可することによって、この状況を回避する必要がある[SHOULD]。

- エンコーダの解像度が最大サイズと最小サイズの範囲内であれば、アクションは必要ない。

3.7 Simulcast (同時放送)

JSEP は MediaStreamTrack の同時放送送信をサポートしており、単一の"m="セクションのコンテキスト内でソースメディアの複数のエンコードを送信できる。現在の JSEP API は、アプリケーションが同時放送メディアを送信できるように設計されているが、1 つのエンコードしか受信できない。これにより、各送信クライアントが複数のエンコーディングをサーバに送信し、サーバが受信クライアントごとに転送する適切なエンコーディングを選択するマルチユーザのシナリオが可能になる。

アプリケーションは、RtpSender で複数のエンコーディングを設定することで、同時放送のサポートを要求する。オファーまたはアンサーの生成時に、これらのエンコーディングは、以下に説明するように、対応する"m="セクションの SDP マーキングを介して示される。Simulcast を理解し、受信を希望する受信者は、サポートを示す SDP マーキングも含み、JSEP エンドポイントはこれらのマーキングを使用して、特定の RtpSender に対して Simulcast が許可されているかどうかを判断する。Simulcast サポートがネゴシエートされていない場合、RtpSender は最初に設定されたエンコーディングのみを使用する。

正確な simulcast パラメータは、送信側アプリケーションによって異なる。前述の SDP マーキングは、リモート側が複数の simulcast エンコードを受信してデマックスできるようにするために提供されているが、各エンコードに使用される特定の解像度とビットレートは、JSEP では純粋に送信側の決定である。

JSEP は現在、simulcast の受信を設定するメカニズムを提供していない。これは、リモートエンドポイントによって simulcast が提供された場合、JSEP エンドポイントによって生成されたアンサーは simulcast の受信をサポートしていることを示さないため、リモートエンドポイントは"m="セクションごとに 1 つのエンコードのみを送信することを意味する。

さらに、JSEP は、JSEP エンドポイントからの simulcast を要求する着信オファーを処理するメカニズムを提供しない。つまり、JSEP エンドポイントが最初のオファーを受信する場合の同時キャストの設定には、アウトオブバンドシグナリングまたは SDP 検査が必要になる。ただし、JSEP エンドポイントが最初のオファーで同時キャストを設定する場合、確立されたすべての同時キャストストリームは、再オファーの受信時に引き続き機能する。この仕様の将来のバージョンでは、最初のオファーの着信シナリオを処理するための API が追加される可能性がある。

JSEP を使用して RtpSender から複数のエンコードを送信する場合は、[RFC8853] と [RFC8851] の手法が使用される。具体的には、RtpSender に複数のエンコードが設定されている場合、RtpSender の"m="セクションには、[RFC8853] の 5.1 節で定義されているように、"a=simulcast"属性が含まれ、各目的のエンコードを一覧表示する"send" simulcast ストリームの説明が含まれ、"recv" simulcast ストリームの説明は含まれない。"m="セクションには、[RFC8851] の 4 章で指定されているように、各エンコードの"a=rid"属性も含まれる。制限識別子(RID (rid-id または RtpStreamIds とも呼ばれる))を使用すると、すべてが同じ"m="セクションの一部である場合でも、個々のエンコードを明確にすることができる。https://www.rfc-editor.org/rfc/rfc8851

3.8 分岐との相互作用

一部のコールシグナリングシステムでは、SDP オファーが複数のデバイスに提供される可能性のあるさまざまなタイプの分岐が可能である。たとえば、SIP [RFC3261] では、「並列検索」と「順次検索」の両方が定義されている。これらは、主に JSEP の範囲外のシグナリングレベルの問題であるが、関連するメディアプレーンの設定にある程度の影響を与える。シグナリング層でフォークが発生した場合、シグナリングを担当する JavaScript アプリケーションは、任意の時点でどのメディアを送受信すべきか、どのリモートエンドポ

イントと通信すべきかを決定する必要がある。JSEP は、メディアエンジンが RTP とメディアをアプリケーションの要求どおりに実行できるようにするために使用される。アプリケーションがメディアエンジンに実行させることができる基本的な操作は次のとおりである。

- 指定されたリモートピアとのメディア交換を開始するが、オファーで予約されているすべてのリソースを保持する。
- 指定されたリモートピアとのメディア交換を開始し、オファー内の使用されていないリソースを解放する。

3.8.1 順次分岐

順次分岐では、複数のリモート呼び出し先にコールがディスパッチされ、各呼び出し先がコールを受け入れることができるが、一度に 1 つのアクティブセッションしか存在しない;受信したメディアのミキシングは行われない。

JSEP はシーケンシャルフォークをうまく処理するため、アプリケーションは目的のリモートエンドポイントを選択するためのポリシーを簡単に制御できる。呼び出し元の 1 つからアンサーが到着すると、アプリケーションはそれを (1) 暫定的なアンサーとして適用し、将来別のアンサーを使用する可能性を残すか、(2) 最終的なアンサーとして適用し、セットアップフローを終了するかを選択できる。

「先勝ち」の状況では、最初のアンサーが最終的なアンサーとして適用され、アプリケーションは後続のアンサーを拒否する。SIP の用語では、これは ACK+BYE になる。

「後勝ち」の状況では、すべてのアンサーが暫定的なアンサーとして適用され、以前のコールレッグは終了される。ある時点で、アプリケーションはセットアッププロセスを終了する。この時点で、アプリケーションは保留中のリモート記述を最終的なアンサーとして再適用できる。

3.8.2 並列フォーク

並列フォークでは、複数のリモート呼び出し先にコールがディスパッチされ、各呼び出し先がコールを受け入れることができ、その結果として複数の同時アクティブシグナリングセッションを確立できる。複数の呼び出し先が同時にメディアを送信する場合、これを処理する可能性については、[RFC3960] の 3.1 節で説明されている。現在のほとんどの SIP デバイスは、一度に単一のデバイスとのメディアの交換のみをサポートしており、複数の初期メディアのオーディオソースを混在させようとしないため、混乱する状況になる可能性がある。たとえば、ヨーロッパのリングバックトーンを北米のリングバックトーンと混在させることを検討すること。結果として得られるサウンドはどちらのトーンとも似ておらず、ユーザを混乱させる。シグナリングアプリケーションが一度に 1 つのリモートエンドポイントとのメディア交換のみを行う場合、メディアエンジンの観点からは、これはシーケンシャルフォークのケースとまったく同じである。

JavaScript アプリケーションが複数のピアと同時にメディアを交換することを望む並列フォークのケースでは、フローは少し複雑になるが、JavaScript アプリケーションは [RFC3960] に記載されている戦略に従い、UPDATE を使用する。UPDATE アプローチを使用すると、シグナリングはメディアを交換するピアごとに個別のメディアフローを設定できる。JSEP では、UPDATE で使用されるこのオファーは、新しい PeerConnection (4.1 節参照)を作成し、この新しい PeerConnection に同じローカルメディアストリームが追加されていることを確認するだけで形成される。その後、新しい PeerConnection オブジェクトは、シグナリングが [RFC3960] で説明されている UPDATE 戦略を実行するために使用できる SDP オファーを生成する。

メディアストリームを共有した結果、アプリケーションは N 個の並列 PeerConnection セッションを持つことになり、それぞれにローカルとリモートの記述と独自のローカルとリモートのアドレスがある。これらのセッションからのメディアフローは、setDirection (4.2.3 項を参照) を使用して管理することも、アプリケーションがすべてのセッションのメディアを混在させて再生することもできる。もちろん、アプリケーションが 1 つのセッションだけを保持したい場合は、不要になったセッションを終了するだけで済む。

4. インタフェース

ここでは、JSEP 機能を実装するために必要な基本的な操作について説明する。W3C API で公開されている実際の API は、構文が多少異なる場合があるが、これらの概念は簡単にマップできる。

4.1 PeerConnection

4.1.1 コンストラクタ

PeerConnection コンストラクタを使用すると、アプリケーションは、候補を収集するときに使用する STUN/TURN サーバやクレデンシャル、初期 ICE 候補ポリシーやプールサイズ、使用するバンドルポリシーなど、メディアセッションのグローバルパラメータを指定できる。

ICE 候補ポリシーが指定された場合、3.5.3 項で説明されているように機能するため、JSEP 実装は許可された候補 (実装内部フィルタリングを含む) のみをアプリケーションに表示し、それらの候補のみを接続チェックに使用する。使用可能なポリシーのセットは次のとおりである。

すべて: 実施方針で許可されたすべての候補を集めて使用する。

リレー: リレー候補を除くすべての候補が除外される。これにより、受信した候補からリモートピアによって確認される可能性のある位置情報が不明瞭になる。アプリケーションがリレーサーバをどのように展開して選択するかによって、これは場所をメトロレベル、場合によってはグローバルレベルまで不明瞭にする可能性がある。

通常、これは望ましいポリシーであり、通常はアプリケーション TURN サーバリソースの使用も大幅に削減するため、デフォルトの ICE 候補ポリシーは"all"に設定する必要がある[MUST]。

ICE 候補プールにサイズが指定されている場合、これは候補を事前に収集する ICE コンポーネントの数を示す。事前収集は、STUN/TURN サーバリソースを潜在的に長期間使用することになるため、これはアプリケーションの要求時にのみ発生する必要があり[MUST]、そのため、デフォルトの候補プールサイズは 0 である必要がある[MUST]。

アプリケーションは、バンドル ([RFC8843] で定義されている多重化メカニズム) の使用に関する優先ポリシーを指定できる。ポリシーに関係なく、アプリケーションは常に単一のトランスポートへのバンドルのネゴシエートを試み、すべての"m="セクションにわたって単一のバンドルグループを提供する。この単一のトランスポートの使用は、アンサー側がバンドルを受け入れることによって決まる。ただし、以下のリストからポリシーを指定することで、アプリケーションはメディアストリームをどの程度積極的にバンドルしようとするかを正確に制御でき、バンドルを認識しないエンドポイントとの相互運用方法に影響する。非バンドル対応エンドポイントとネゴシエートする場合、バンドル専用ストリームとしてマークされていないストリームのみが確立される。

使用可能なポリシーのセットは次のとおりである。

balanced: 各タイプの最初の"m="セクション(オーディオ、ビデオ、またはアプリケーション)にはトランスポートパラメータが含まれ、アンサー側はそのセクションをアンバンドルできる。各タイプの 2 番目と後続の"m="セクションはバンドル専用とマークされる。その結果、N 個の異なるメディアタイプがある場合、N 個のメディアストリームの候補が収集される。このポリシーは、多重化への欲求と、基本的な音声とビデオが従来のケースでもネゴシエートできるようにする必要性とのバランスを取っている。アンサー側として機能する場合、オファーにバンドルグループがない場合、実装は各タイプの最初の"m="セクションを除くすべてを拒否する。

max-compat: すべての"m="セクションにトランスポートパラメータが含まれる;バンドル専用としてマークされるものはない。このポリシーでは、バンドルを認識しないエンドポイントですべてのストリーム

を受信できるが、メディアストリームごとに個別の候補を収集する必要がある。

max-bundle: 最初の"m="セクションにのみトランスポートパラメータが含まれる;最初のストリーム以外のすべてのストリームはバンドル専用としてマークされる。このポリシーは、従来のエンドポイントとの互換性を犠牲にして、候補の収集を最小限に抑え、多重化を最大化することを目的としている。アンサー側として機能する場合、実装では、最初の"m="セクション以外の"m="セクションは、その"m="セクションと同じバンドルグループにない限り拒否される。

パフォーマンスとレガシーエンドポイントとの互換性の間の最適なトレードオフを提供するため、デフォルトのバンドルポリシーは"balanced"に設定する必要がある[MUST]。

アプリケーションは、次のいずれかのポリシーを使用して、RTP/RTCP 多重化 [RFC5761] の使用に関する優先ポリシーを指定できる。

negotiate:JSEP の実装では、RTP と RTCP の両方の候補が収集されるが、"a=rtcp-mux"も提供されるため、多重化エンドポイントまたは非多重化エンドポイントとの互換性が可能になる。

require: JSEP の実装では、RTP 候補のみが収集され、生成されるオファーの新しい"m="セクションに"a=rtcpmux-only"表示が挿入される。これにより、オファー側が収集する必要がある候補の数が半分になる。"a=rtcp-mux"属性を含まない"m="セクションを含む説明を適用すると、エラーが返される。

デフォルトの多重化ポリシーは"require"に設定する必要がある[MUST]。実装は、多重化ポリシーを"negotiate"に設定するアプリケーションの試みを拒否することを選択できる[MAY]。

4.1.2 addTrack

addTrack メソッドは、MediaStream 引数を使用して同じ MediaStream 内の他のトラックとトラックを関連付けることで、PeerConnection に MediaStreamTrack を追加し、オファーまたはアンサーの作成時に同じ"LS" (リップシンク) グループに追加できるようにする。同じ"LS"グループにトラックを追加すると、[RFC5888] の 7 章で説明されているように、適切なリップシンクのためにこれらのトラックの再生が同期される必要があることを示す。https://www.rfc-editor.org/rfc/rfc5888addTrack は、次のようにトランシーバの数を最小化しようとする。PeerConnection が"haveremote-offer"状態の場合、トラックは setRemoteDescription の最新の呼び出しによって作成され、ローカルトラックを持たない最初の互換性のあるトランシーバに接続される。それ以外の場合は、4.1.4 項で説明するように、新しいトランシーバが作成される。

4.1.3 removeTrack

removeTrack メソッドは、PeerConnection から MediaStreamTrack を削除する。このとき、RtpSender 引数を使用して、どの送信者のトラックを削除するかを指定する。送信者のトラックはクリアされ、送信者は送信を停止する。今後、createOffer を呼び出すと、送信者に関連付けられた"m="セクションが recvonly (transceiver.direction が sendrecv の場合)または inactive (transceiver.direction が送信専用の場合)としてマークされる。

4.1.4 addTransceiver

addTransceiver メソッドは、PeerConnection に新しい RtpTransceiver を追加する。MediaStreamTrack 引数が指定されている場合、トランシーバはそのメディアタイプで設定され、トラックはトランシーバに接続される。それ以外の場合、アプリケーションは明示的にタイプを指定する必要がある[MUST]。このモードは、recvonly トランシーバを作成する場合だけでなく、後でトラックを接続できるトランシーバを作成する場合にも便利である。

作成時に、アプリケーションは、トランシーバの方向属性、トランシーバが関連付けられている

MediaStream のセット(3.7 節に記載されている同時放送に使用される)、およびメディアのエンコードのセットを指定することもできる。

4.1.5 onaddtrack Event

onaddtrack イベントは、setRemoteDescription 呼び出しの結果として新しいリモートトラックが通知されたときにアプリケーションにディスパッチされる。新しいトラックは、トラックが属する MediaStream (s) とともに、イベント内の MediaStreamTrack オブジェクトとして提供される。

4.1.6 createDataChannel

createDataChannel メソッドは、新しいデータチャネルを作成し、それを PeerConnection に接続する。この PeerConnection のデータチャネルが現在存在しない場合は、新しいオファー/アンサー交換が必要となる。特定の PeerConnection 上のすべてのデータチャネルは、同じ SCTP/DTLS アソシエーション ("SCTP"は"Stream Control Transmission Protocol"の略) を共有するため、同じ"m="セクションを共有するため、後続のデータチャネルの作成は JSEP の状態に影響を与えない。

createDataChannel メソッドには、PeerConnection (例:maxPacketLifetime)で使用されるが、SDP に反映されず、JSEP の状態に影響を与えない多くの引数も含まれている。

4.1.7 ondatachannel Event

ondatachannel イベントは、新しいデータチャネルがリモート側によってネゴシエートされたときにアプリケーションにディスパッチされる。これは、基礎となる SCTP/DTLS アソシエーションが確立された後、いつでも発生する可能性がある。新しいデータチャネルオブジェクトはイベントで提供される。

4.1.8 createOffer

createOffer メソッドは、この PeerConnection に追加されたメディアの説明、この実装でサポートされているコーデック、RTP、および RTCP オプション、および ICE エージェントによって収集された候補を含む、セッションのサポートされている構成を含む [RFC3264] ごとのオファーを含む SDP の blob を生成する。オプションパラメータを指定すると、生成されるオファーをさらに制御できる。このオプションパラメータを使用すると、接続を再確立する目的で、アプリケーションが ICE 再起動をトリガーできる。

最初のオファーでは、生成された SDP には、セッションに必要なすべての機能が含まれる (サポートされているが、デフォルトでは必要ない機能は省略できる)。SDP 行ごとに、SDP の生成は、指定された SDP 行を定義する仕様から最初のオファーを生成するために定義されたプロセスに従う。最初のオファー生成の正確な処理については、下記の 5.2.1 項で詳しく説明する。

セッションが確立された後に createOffer が呼び出された場合、createOffer は、セッションに加えられた変更に基づいて現在のセッションを変更するオファーを生成する。

たとえば、RtpTransceivers を追加または停止したり、ICE の再起動を要求したりする。既存のストリームごとに、各 SDP 行の生成は、特定の SDP 行を指定する RFC から更新されたオファーを生成するために定義されたプロセスに従う必要がある[MUST]。新しいストリームごとに、SDP の生成は、前述のように、最初のオファーを生成するプロセスに従う必要がある[MUST]。変更が行われていない場合、または要求された変更の影響を受けない SDP 行の場合、オファーには最後のオファー/アンサー交換によってネゴシエートされたパラメータのみが含まれる[SHOULD]。後続のオファー生成の正確な処理については、後述の 5.2.2 項で詳しく説明する。

createOffer によって生成されたセッション記述は、setLocalDescription によってすぐに使用できる必要がある[MUST]。システムのリソースが制限されている場合(例えば、有限個のデコーダ)、createOffer はシステムの現在の状態を反映するオファーを返す必要がある。これにより、setLocalDescription がそれらのリソースを

取得しようとしたときに成功する。

このメソッドを呼び出すと、新しい ICE 資格情報の生成などを行うことができるが、PeerConnection の状態を変更したり、候補の収集をトリガーしたり、メディアのフローを開始または停止したりすることはない。具体的には、オファーは適用されず、setLocalDescription が呼び出されるまで、保留中のローカル記述にはならない。

4.1.9 createAnswer

createAnswer メソッドは、setRemoteDescription への最新の呼び出しで指定されたパラメータと互換性のあるセッションのサポートされている設定を使用して、[RFC3264] ごとに SDP アンサーを含む SDP の blob を生成する。setRemoteDescription は、createAnswer を呼び出す前に呼び出されている必要がある[MUST]。createOffer と同様に、返される blob には、この PeerConnection に追加されたメディアの説明、このセッションでネゴシエートされたコーデック/RTP/RTCP オプション、および ICE エージェントによって収集された候補が含まれる。生成されたアンサーをさらに制御するために、options パラメータを指定できる。

アンサーとして、生成された SDP には、メディアプレーンの確立方法を指定する特定の設定が含まれる。SDP 行ごとに、SDP の生成は、指定された SDP 行を定義する仕様からアンサーを生成するために定義されたプロセスに従う必要がある[MUST]。アンサー生成の正確な処理については、後述の 5.3 節で詳しく説明する。

createAnswer によって生成されたセッション記述は、setLocalDescription によってすぐに使用できる必要がある[MUST];createOffer と同様に、返される説明はシステムの現在の状態を反映する必要がある[SHOULD]。

このメソッドを呼び出すと、新しい ICE 資格情報の生成などを行うことができるが、PeerConnection の状態を変更したり、候補の収集をトリガーしたり、メディアの状態を変更したりすることはない。具体的には、setLocalDescription が呼び出されるまで、アンサーは適用されず、現在のローカル記述にはならない。

4.1.10 SessionDescriptionType

セッション記述オブジェクト (RTCSessionDescription) のタイプは、"offer"、"pranswer"、"answer"、または"rollback"である。これらのタイプは、記述パラメータの解析方法とメディア状態の変更方法に関する情報を提供する。

「offer」は、説明をオファーとして解析する必要があることを示す[MUST]。この説明には、多くの可能なメディア構成が含まれる場合がある。「オファー」として使用される説明は、PeerConnection が「安定した」状態にあるとき、または以前に提供されたがアンサーのない「オファー」の更新として適用されるときにいつでも適用できる。

「pranswer」は、説明をアンサーとして解析する必要があるが[MUST]、最終的なアンサーとして解析する必要はないため、割り当てられたリソースを解放してはならないことを示す[MUST NOT]。非アクティブなメディア方向がアンサーで指定されていない場合、メディア送信が開始されることがある。"pranswer"として使用される説明は、"offer"への応答として、または以前に送信された"pranswer"の更新として適用される場合がある。

"answer"は、説明がアンサーとして解析されなければならないこと[MUST]、オファー/アンサーの交換が完了したと見なされなければならないこと[MUST]、および不要になったリソース(デコーダ、候補)が解放されなければならないことを示す[SHOULD]。"answer"として使用される説明は、"offer"への応答として、または以前に送信された"pranswer"の更新として適用される場合がある。

暫定的なアンサーと最終的なアンサーの唯一の違いは、最終的なアンサーによって、オファーの結果として割り当てられた未使用のリソースが解放されることである。そのため、アプリケーションは、アンサーを暫定的なものとして適用するか最終的なものとして適用するかについてある程度の裁量を使用でき、必要に応じてセッション記述のタイプを変更できる。たとえば、シリアル分岐のシナリオでは、アプリケーションは、各リモートエンドポイントから 1 つずつ、複数の「最終的な」アンサーを受け取る場合がある。アプリ

ケーションは、最初のアンサーを暫定的なアンサーとして受け入れ、その基準(例:ボイスメールの代わりにライブユーザ)を満たすアンサーを受け取った場合にのみ最終的なアンサーとして適用することを選択できる。

"rollback"は、4.1.10.2 項で説明されているように、ステートマシンを以前の"stable"状態にロールバックする必要がある[MUST]ことを示す特別なセッション記述タイプである。内容は空でなければならない[MUST]。

4.1.10.1 Use of Provisional Answers

ほとんどのアプリケーションでは、"pranswer"タイプを使用してアンサーを作成する必要はない。オファーに対してすぐにアンサーを送信することをお勧めするが、セッショントランスポートをウォームアップしてメディアクリッピングを防ぐために、JSEP アプリケーションで推奨される処理は、オファーを受信した直後に null の MediaStreamTrack を含む"sendonly"最終アンサーを作成して送信することである。これにより、メディアが発信者によって送信されるのを防ぎ、メディアが着信者によって応答されるとすぐに送信できるようになる。その後、着信者が実際にコールを受け入れると、アプリケーションは実際の MediaStreamTrack を接続して新しい"sendrecv"オファーを作成し、以前のオファー/アンサーペアを更新して双方向メディアフローを開始できる。これは、"sendonly"暫定アンサーの後に"sendrecv"アンサーを使用して行うこともできるが、最初の暫定アンサーはオファー/アンサーの交換を開いたままにするため、発信者はこの間に更新されたオファーを送信できない。

例として、オーディオとビデオをできるだけ早く設定したい一般的な JSEP アプリケーションを考えてみる。呼び出し先は、オーディオとビデオの MediaStreamTracks でオファーを受信すると、これらのトラックを sendonly として受け入れる即アンサーを送信する (つまり、呼び出し元は呼び出し先にまだメディアを送信しないことと、呼び出し先がまだ独自のメディアを追加していないことを意味する。

MediaStreamTracks の場合、呼び出し先もメディアを送信しない)。その後、ユーザにコールの受け入れと必要なローカルトラックの取得を求める。ユーザが同意すると、アプリケーションは取得したトラックを接続する。これは、ICE ハンドシェイクと DTLS ハンドシェイクがこの時点までに完了している可能性が高いため、すぐに送信を開始できる。また、アプリケーションは、コールの受け入れを示す新しいオファーをリモート側に送信し、音声とビデオを双方向メディアに移動する。これらの行に沿った詳細なフロー例を 7.3 節に示す。

もちろん、一部のアプリケーション、特にレガシーシグナリングプロトコルへのゲートウェイを試みるアプリケーションでは、この二重のオファー/アンサーエクスチェンジを実行できない場合がある。このような場合でも、pranswer はトランスポートをウォームアップするメカニズムをアプリケーションに提供できる。

4.1.10.2 Rollback

状況によっては、setLocalDescription または setRemoteDescription に対する変更を「元に戻す」ことが望ましい場合がある。コールが進行中で、一方がセッションパラメータの一部を変更したい場合を考える。その側は更新されたオファーを生成し、setLocalDescription を呼び出す。ただし、リモート側は、setRemoteDescription の前でも後でも、新しいパラメータを受け入れたくないと判断し、拒否メッセージをオファー側に返信する。ここで、オファー側と、場合によってはアンサー側も、「安定した」状態と以前のローカル/リモートの説明に戻る必要がある。これをサポートするために、セッションに対して提案された変更を破棄し、ステートマシンを「安定した」状態に戻す「ロールバック」の概念を導入する。ロールバックは、setLocalDescription または setRemoteDescription に空の内容を持つ「ロールバック」タイプのセッション記述を提供することによって実行される。

4.1.11 setLocalDescription

setLocalDescription メソッドは、指定されたセッション記述をローカル構成として適用するよう

PeerConnection に指示をする。type フィールドは、説明をオファー、暫定アンサー、最終アンサー、またはロールバックとして処理する必要があるかどうかを示す。オファーとアンサーは、SDP 行ごとに存在するさまざまなルールを使用して、異なる方法でチェックされる。

この API は、ローカルメディアの状態を変更する;特に、メディアを受信およびデコードするためのローカルリソースを設定する。アプリケーションが 1 つのメディア形式から異なる互換性のない形式への変更を提供するシナリオを正常に処理するには、PeerConnection が現在のローカル記述と保留中のローカル記述の両方の使用を同時にサポートできる必要がある[MUST] (例:いずれかの説明に存在するコーデックをサポートする)。この二重処理は、PeerConnection が"have-local-offer"状態になったときに開始され、(1) 最終的なアンサーが返された時点で PeerConnection が保留中のローカル記述を完全に採用できるか、(2) ロールバックされて現在のローカル記述に戻るかのいずれかで setRemoteDescription が呼び出されるまで継続される。

この API は、候補の収集プロセスを間接的に制御する。ローカル記述が指定され、現在使用されているトランスポートの数がローカル記述で必要なトランスポートの数と一致しない場合、PeerConnection は必要に応じてトランスポートを作成し、候補プールからの候補を使用して、各トランスポートの候補の収集を開始する (使用可能な場合) 。

(1) setRemoteDescription が以前にオファーで呼び出された場合、(2) setLocalDescription がアンサー (暫定または最終) で呼び出された場合、(3) メディアの方向が互換性がある場合、(4) メディアを送信できる場合、これによりメディア送信が開始される。

4.1.12 setRemoteDescription

setRemoteDescription メソッドは、指定されたセッション記述を目的のリモート構成として適用するよう PeerConnection に指示をする。setLocalDescription と同様に、説明の type フィールドは、その処理方法を示す。

この API は、ローカルメディアの状態を変更する。特に、メディアを送信およびエンコードするためのローカルリソースを設定する。

(1) setLocalDescription が以前にオファーで呼び出された場合、(2) setRemoteDescription がアンサー (暫定または最終) で呼び出された場合、(3) メディアの方向に互換性があり、(4) メディアを送信できる場合、これによりメディア送信が開始される。

4.1.13 currentLocalDescription

currentLocalDescription メソッドは、ローカル記述が設定されてから ICE エージェントによって生成されたローカル候補に加えて、現在ネゴシエートされたローカル記述 (つまり、最後に成功したオファー/アンサー交換のローカル記述) を返す。

オファー/アンサー交換がまだ完了していない場合は、null オブジェクトが返される。

4.1.14 pendingLocalDescription

pendingLocalDescription メソッドは、ローカル記述が設定されてから ICE エージェントによって生成されたローカル候補に加えて、現在ネゴシエーション中のローカル記述のコピー (つまり、対応するリモートアンサーのないローカルオファーセット) を返す。

PeerConnection の状態が"stable"または"have-remote-offer"の場合は、null オブジェクトが返される。

4.1.15 currentRemoteDescription

currentRemoteDescription メソッドは、リモートの説明が設定されてから processIceMessage を介して提供されたリモート候補に加えて、現在ネゴシエートされているリモートの説明のコピー (つまり、最後に成功したオファー/アンサー交換からのリモートの説明) を返す。

オファー/アンサー交換がまだ完了していない場合は、null オブジェクトが返される。

4.1.16 pendingRemoteDescription

pendingRemoteDescription メソッドは、リモート記述が設定されてから processIceMessage を介して提供されたリモート候補に加えて、現在ネゴシエート中のリモート記述のコピー (つまり、対応するローカルアンサーのないリモートオファーセット) を返す。

PeerConnection の状態が"stable"または"have-local-offer"の場合は、null オブジェクトが返される。

4.1.17 canTrickleIceCandidates

canTrickleIceCandidates プロパティは、リモート側がトリクルされた候補の受信をサポートするかどうかを示す。次の 3 つの値が考えられる。

null: 相手側から SDP を受信していないため、トリクルに対応できるかは不明である。これは、setRemoteDescription が呼び出される前の初期値である。

true: 相手側から SDP を受信し、トリクルをサポートできることを示している。

false: トリクルをサポートできないことを示す SDP を相手側から受信した。

3.5.2 項で説明したように、JSEP 実装は常にアプリケーションに個別に候補を提供し、トリクル ICE に必要なものと一致している。ただし、アプリケーションは canTrickleIceCandidates プロパティを使用して、ピアが実際にトリクル ICE を実行できるかどうかを判断できる。

つまり、最初のオファーまたはアンサーを送信し、その後に候補者が収集されるときにそれを送信することが安全かどうかである。"true"はリモート Trickle ICE サポートを明確に示す唯一の値であるため、canTrickleIceCandidates と"true"を比較するアプリケーションは、デフォルトで最初のオファーに対して Half Trickle を試行し、その後の Trickle ICE 互換エージェントとの相互作用に対して Full Trickle を試行する。

4.1.18 setConfiguration

setConfiguration メソッドを使用すると、最初はコンストラクタパラメータによって設定されていた PeerConnection のグローバル設定をセッション中に変更できる。このメソッドの呼び出しの効果は、呼び出されたタイミングによって異なり、変更された特定のパラメータによって異なる。

- 使用する STUN/TURN サーバへの変更は、次の収集フェーズに影響する。ICE 収集フェーズがすでに開始または完了している場合は、3.5.1 項に記載されている「needs-ice-restart」ビットが設定される。これにより、次の createOffer の呼び出しで新しい ICE 資格情報が生成され、ICE の再起動を強制して新しい収集フェーズを開始し、新しいサーバが使用される。ICE 候補プールのサイズが 0 以外で、ローカル記述がまだ適用されていない場合、既存の候補は破棄され、新しい候補が新しいサーバから収集される。
- ICE 候補ポリシーを変更すると、次の収集フェーズに影響する。ICE 収集フェーズがすでに開始または完了している場合は、「needs-ice-restart」ビットが設定される。いずれにしても、ポリシーの変更は候補プールに影響しない。これは、プールされた候補は収集フェーズが発生するまでアプリケーションで使用できないため、必要なフィルタリングはプールされた候補に対して実行できる。
- ローカル記述を適用した後に ICE 候補プールサイズを変更してはならない[MUST NOT]。ローカル記述がまだ適用されていない場合、ICE 候補プールサイズの変更はすぐに有効になる;増加した場合、追加の候補が事前に収集される;減少した場合、不要になった候補は破棄される。
- バンドルおよび RTCP 多重化ポリシーは、PeerConnection の構築後に変更してはならない[MUST NOT]。

このメソッドを呼び出すと、ICE エージェントの状態が変更される可能性がある。

4.1.19 addIceCandidate

addIceCandidate メソッドは、IceCandidate オブジェクト (3.5.2.1 項) を介して ICE エージェントに更新を提供する。IceCandidate の候補フィールドが null 以外の場合、IceCandidate は新しいリモート ICE 候補として扱われ、Trickle ICE に定義されたルールに従って、現在および/または保留中のリモート記述に追加される。それ以外の場合、IceCandidate は、[RFC8838] の 14 章で定義されているように、候補の終わりを示すものとして扱われる。

いずれの場合も、上記の 3.5.2.1 項で説明したように、提供された IceCandidate の"m="セクションインデックス、MID、および ufrag フィールドを使用して、IceCandidate がどの"m="セクションと ICE 候補生成に属するかが判断される。候補の終了表示の場合、"m="セクションインデックスと MID フィールドの null 値は、指定された ICE 候補生成内のすべての"m="セクションに表示が適用されることを意味すると解釈される。ただし、新しいリモート候補に対して両方のフィールドが null の場合、これは以下に示すように無効な条件として扱われる必要がある[MUST]。

IceCandidate フィールドに無効な値が含まれている場合、または IceCandidate オブジェクトの処理中にエラーが発生した場合は、指定された IceCandidate を無視して[MUST]エラーを返す必要がある[MUST]。

それ以外の場合は、新しいリモート候補または候補の終わりの表示が ICE エージェントに提供される。新しいリモート候補の場合は、ICE 収集プロセスを初期化するために setLocalDescription がすでに呼び出されていると仮定して、接続チェックが新しい候補に送信される。

4.1.20 onicecandidate

onicecandidate イベントは、(1) 3.5.1 項で概説されており、3.5.3 項で説明されている制限に従う ICE 収集中に、ICE エージェントが新しい許可されたローカル ICE 候補を検出した場合、または (2) ICE 収集フェーズが完了した場合の 2 つの状況でアプリケーションにディスパッチされる。このイベントには、3.5.2.1 項で定義されているように、1 つの IceCandidate オブジェクトが含まれる。

最初のケースでは、新しく発見された候補が IceCandidate オブジェクトに反映され、そのすべてのフィールドが非 null でなければならない[MUST]。この候補は、Trickle ICE に定義されたルールに従って、現在および/または保留中のローカル記述にも追加される。

2 番目のケースでは、イベントの IceCandidate オブジェクトは、現在の収集フェーズが完了したことを示すために、その候補フィールドを null に設定する必要がある[MUST]。つまり、このフェーズではそれ以上の onicecandidate イベントはない。ただし、どの ICE 候補生成が終了するかを示すために、IceCandidate の ufrag フィールドを指定する必要がある[MUST]。IceCandidate の"m="セクションインデックスと MID フィールドは、イベントが特定の"m="セクションに適用されることを示すために指定することも、現在の ICE 候補生成のすべての"m="セクションに適用されることを示すために null に設定することもできる[MAY]。このイベントは、[RFC8838] の 13 章で定義されているように、候補の終了表示を生成するためにアプリケーションで使用できる。

4.2 RtpTransceiver

4.2.1 stop

stop メソッドは RtpTransceiver を停止する。これにより、以降の createOffer の呼び出しで、関連付けられた"m="セクションの 0 ポートが生成される。詳細は以下を参照。

4.2.2 stopped

stopped プロパティは、停止の呼び出しによって、または関連する"m="セクションを拒否するアンサーを適用することによって、トランシーバが停止したかどうかを示す。いずれの場合も"true"に設定され、それ以

外の場合は"false"に設定される。

停止した RtpTransceiver は、発信 RTP または RTCP を送信したり、着信 RTP または RTCP を処理したりしない。再起動することはできない。

4.2.3 setDirection

setDirection メソッドは、トランシーバの方向を設定する。これは、今後の createOffer および createAnswer の呼び出しで、関連付けられた"m="セクションの direction プロパティに影響する。direction に使用できる値は、[RFC4566] の 6 章で定義されている同じ名前の direction 属性を反映して、"recvonly"、"sendrecv"、"sendonly"、および"inactive"である。

オファーを作成すると、再オファーの場合でも、トランシーバの方向が出力に直接反映される。アンサーを作成する場合、以下の 5.3 節で説明するように、トランシーバの方向はオファーされた方向と交差する。

setDirection はトランシーバの direction プロパティをすぐに設定するが (4.2.4 項) 、このプロパティはトランシーバの RtpSender が送信するか、その RtpReceiver が受信するかにすぐには影響しない。有効な方向は currentDirection プロパティによって表され、アンサーが適用されたときにのみ更新される。

4.2.4 direction

direction プロパティは、setDirection に渡された最後の値を示す。setDirection が一度も呼び出されていない場合は、トランシーバが初期化された方向に設定される。

4.2.5 currentDirection

currentDirection プロパティは、トランシーバに関連付けられた"m="セクションの最後にネゴシエートされた方向を示す。具体的には、最後に適用されたアンサー (暫定アンサーを含む) で関連付けられた"m="セクションの方向属性 [RFC3264] を示し、リモートアンサーの場合は"send"と"recv"の方向が逆になる。たとえば、リモートアンサーで関連付けられた"m="セクションの方向属性が"recvonly"の場合、currentDirection は"sendonly"に設定される。

このトランシーバを参照するアンサーがまだ適用されていない場合、またはトランシーバが停止している場合、currentDirection は"null"に設定される。

4.2.6 setCodecPreferences

setCodecPreferences メソッドは、トランシーバのコーデック設定を設定する。これは、createOffer および createAnswer の将来の呼び出しで、関連付けられた"m="セクションのコーデックの存在と順序に影響する。setCodecPreferences は、実装が送信を決定するコーデックに直接影響しないことに注意すること。オファーまたはアンサーを介して、実装が受信を優先するコーデックにのみ影響する。setCodecPreferences によってコーデックが除外された場合でも、次のオファー/アンサー交換で破棄されるまで送信に使用されることがある。

RtpTransceiver のコーデック設定によって、後続の createOffer と createAnswer の呼び出しによってコーデックが除外されることがある。この場合、関連する"m="セクション内の対応するメディア形式が除外される。コーデック設定では、本来存在しないメディア形式を追加することはできない。

RtpTransceiver のコーデック設定では、後続の createOffer と createAnswer の呼び出しでコーデックの順序を決定することもできる。この場合、関連する"m="セクションのメディア形式の順序は、指定された設定に従う。

5. SDP 相互作用の手順

ここでは、SDP オブジェクトを作成および解析する際に従うべき具体的な手順について説明する。

5.1 要件の概要

JSEP 実装は、オファーとアンサーの作成と処理を管理する以下に示す仕様に準拠する必要がある。

5.1.1 使用要件

ローカルとリモートの両方の JSEP 実装によって処理されるすべてのセッション記述は、次の仕様のサポートを示す必要がある[MUST]。これらのいずれかがない場合、この省略はエラーとして扱われる必要がある[MUST]。

- [RFC8445] で指定されている ICE を使用する必要がある。リモートエンドポイントは lite 実装を使用する場合があることに注意すること。実装は、ICE-lite を使用するリモートエンドポイントを適切に処理する必要がある。リモートエンドポイントは、古いバージョンの ICE を使用することもできる。実装は、[RFC5245] で指定されている ICE を使用するリモートエンドポイントを適切に処理する必要がある[MUST]。
- [RFC8827] で指定されているメディアタイプに応じて、DTLS [RFC6347] または DTLS-SRTP [RFC5763] を使用する必要がある[MUST]。

[RFC8827] で説明されているように、SRTP キーイングの SDP セキュリティ記述メカニズム [RFC4568] を使用してはならない[MUST NOT]。

5.1.2 プロファイル名と相互運用性

メディア"m="セクションの場合、JSEP 実装は [RFC5764] で指定された"UDP/TLS/RTP/SAVPF"プロファイルと [RFC7850] で指定された"TCP/DTLS/RTP/SAVPF"プロファイルをサポートする必要があり[MUST]、オファーで生成する各メディア"m="行に対してこれらのプロファイルの 1 つを示す必要がある[MUST]。データの場合"m="セクションでは、実装は"TCP/DTLS/SCTP"プロファイルと同様に"UDP/DTLS/SCTP"プロファイルをサポートする必要があり[MUST]、オファーで生成する各データ"m="行に対してこれらのプロファイルの 1 つを示す必要がある[MUST]。使用する正確なプロファイルは、[RFC8839] の 4.2.1.2 項で説明されているように、現在のデフォルトまたは選択された ICE 候補に関連付けられているプロトコルによって決定される。

残念ながら、互換性を確保するために、一部のエンドポイントでは、これらのプロファイルのいずれかをサポートする場合でも、他のプロファイル文字列が生成される。たとえば、エンドポイントは"RTP/AVP"を生成するが、"a=fingerprint"属性と"a=rtcp-fb"属性を提供し、"UDP/TLS/RTP/SAVPF"または"TCP/DTLS/RTP/SAVPF"をサポートする意思があることを示す。このようなエンドポイントとの互換性を簡素化するために、JSEP 実装は、受信オファーのメディア"m="セクションを処理するときに、次の規則に従う必要がある[MUST]。

- 次のいずれかに一致するオファー内のすべてのプロファイルを受け入れる必要がある。
  - "RTP/AVP"([RFC4566] の 8.2.2 項で定義されている)
  - "RTP/AVPF"([RFC4585] の 9 章で定義されている)https://www.rfc-editor.org/rfc/rfc4585
  - "RTP/SAVP"([RFC3711] の 12 章で定義されている)
  - "RTP/SAVPF"([RFC5124] の 6 章で定義されている)
  - "TCP/DTLS/RTP/SAVP"([RFC7850]の 3.4 節で定義されている)https://www.rfc-editor.org/rfc/rfc7850
  - "TCP/DTLS/RTP/SAVPF"([RFC7850]の 3.5 節で定義されている)
  - "UDP/TLS/RTP/SAVP"([RFC5764] の 9 章で定義されている)
  - "UDP/TLS/RTP/SAVPF"([RFC5764] の 9 章で定義されている)
- 生成されたアンサーの"m="行のプロファイルは、オファーで提供されたプロファイルと正確に一致する必要がある[MUST]。

- DTLS-SRTP が必須であるため、SAVP または AVP の選択は効果がない。DTLS-SRTP のサポートは、1 つ以上の"a=fingerprint"属性の存在によって決定される。"a=fingerprint"属性がないと、ネゴシエーションが失敗することに注意すること。
- AVPF または AVP を使用すると、RTCP フィードバックに使用されるタイミングルールが制御されるだけである。AVPF が提供されている場合、または"a=rtcp-fb"属性が存在する場合は、AVPF タイミング、つまり"trr-int=0"のデフォルト値を想定すること。それ以外の場合は、AVPF が AVP 互換モードで使用されていると仮定し、"trr-int=4000"の値を使用する。
- データ"m="セクションの場合、実装は"UDP/DTLS/SCTP"、"TCP/DTLS/SCTP"、または"DTLS/SCTP" (下位互換性のため) プロファイルの受信をサポートする必要がある[MUST]。

JSEP 実装による再オファーは、最初のオファー/アンサー交換で (正しくない) 古いプロファイル文字列が使用された場合でも、正しいプロファイル文字列を使用する必要があることに注意すること。このような再オファーを処理できないリモートエンドポイントは、JSEP エンドポイントの最初のオファーも処理できないため、JSEP の動作が簡素化され、ダウンサイドが最小限に抑えられる。

5.2 オファーの構築

createOffer が呼び出されると、[RFC8834] で指定された機能を含む新しい SDP 記述を作成する必要がある。このプロセスの正確な詳細を以下に説明する。

5.2.1 初回オファー

createOffer を初めて呼び出すと、その結果は初回オファーと呼ばれる。

初回オファーを生成する最初のステップは、[RFC4566] の 5 章で指定されているように、セッションレベルの属性を生成することである。具体的には、

- 最初の SDP 行は、[RFC4566] の 5.1 節で定義されている"v=0"でなければならない[MUST]。
- 2 番目の SDP 行は、[RFC4566] の 5.2 節で定義されている"o="行でなければならない[MUST]。<username>フィールドの値は"-"でなければならない[SHOULD]。<sess-id>は 64 ビット符号付き整数で表す必要があり[MUST]、値は 2^{63}-1 未満でなければならない[MUST]。<sess-id>は、最上位ビットが 0 に設定され、残りの 63 ビットが暗号的にランダムである 64 ビット量を生成することによって構築することが推奨される[RECOMMENDED]。<nettype><addrtype><unicastaddress>タプルの値は、このフィールドでローカル IP アドレスが漏れるのを防ぐために、IN IP4 0.0.0.0 のような意味のないアドレスに設定する必要がある[SHOULD]。この問題は [RFC8828] で議論されている。[RFC4566] で述べたように、"o="行全体が一意である必要があるが、これを実現するには<sess-id>に乱数を選択するだけで十分である。
- 3 番目の SDP 行は、[RFC4566] の 5.3 節で定義されている"s="行でなければならない[MUST]。"o="行に一致するように、単一のダッシュをセッション名として使用する必要がある (例:"s=-") [SHOULD]。これは、単一のスペースを提案する [RFC4566] のアドバイスとは異なるが、"o="と"s="の両方が JSEP では意味がないため、同じ意味のない値を持つ方が明確に見えることに注意すること。
- セッション情報 ("i=")、URI ("u=")、電子メールアドレス ("e=")、電話番号 ("p=")、繰り返し時間 ("r=")、およびタイムゾーン ("z=") の行は、このコンテキストでは有用ではなく、含めるべきではない[SHOULD NOT]。
- 暗号化キー ("k=") の行は十分なセキュリティを提供しないため、含めてはならない[MUST NOT]。
- [RFC4566] の 5.9 節で指定されているように、"t="行を追加する必要がある[MUST]。<start-time>と<stop-time>の両方を 0 に設定する必要がある (例:"t=0 0") [SHOULD]。
- [RFC8840] の 4.1.1 項と [RFC8445] の 10 章で指定されているように、"trickle"と"ice2"オプションを含

む"a=ice-options"行を追加する必要がある[MUST]。

- WebRTC ID が使用されている場合は、[RFC8827] の 5 章で説明されているように、"a=identity"行を追加する必要がある[MUST]。

次のステップでは、[RFC4566] の 5.14 節で指定されているように、"m="セクションを生成する。https://www.rfc-editor.org/rfc/rfc4566"m="セクションは、停止している RtpTransceiver を除いて、PeerConnection に追加された RtpTransceiver ごとに生成される。これは、RtpTransceiver が PeerConnection に追加された順序で実行される。そのような RtpTransceiver がない場合、"m="セクションは生成されない。[RFC3264] の 5 章で説明されているように、後で追加できる。

RtpTransceiver 用に生成された各"m="セクションについて、トランシーバと生成された"m="セクションのインデックス間のマッピングを確立する。

バンドル専用としてマークされていない場合、各"m="セクションには、ICE 資格情報の一意のセットと ICE 候補の一意のセットが含まれている必要がある[MUST]。バンドル専用の"m="セクションには、ICE 資格情報が含まれていてはならず[MUST NOT]、候補を収集してはならない[MUST NOT]。

DTLS の場合、すべての"m="セクションは、PeerConnection に指定されている任意の証明書とすべての証明書を使用する必要がある[MUST]。そのため、これらはすべて同じフィンガープリント値 [RFC8122] を持つ必要がある[MUST]。これらの値はセッションレベルの属性でなければならない[MUST]。

各"m="セクションは、[RFC4566] の 5.14 節で指定されているように生成されなければならない[MUST]。"m="行自体については、次の規則に従う必要がある。

- "m="セクションがバンドル専用としてマークされている場合は、<port>値を 0 に設定する必要がある[MUST]。それ以外の場合、<port>値はこの"m="セクションのデフォルト ICE 候補のポートに設定されるが、候補がまだ使用できない場合は、[RFC8840] の 4.1.1 項に示されているように、デフォルトのポート値 9 (破棄) を使用する必要がある[MUST]。
- DTLS の使用を適切に示すには、<proto>フィールドを [RFC5764] の 8 章で指定されているように、"UDP/TLS/RTP/SAVPF"に設定する必要がある[MUST]。
- 関連するトランシーバにコーデック設定が設定されている場合、メディア形式は対応する順序で生成され、コーデック設定に存在しないコーデックを除外する必要がある[MUST]。
- 上記の制限によって除外されない限り、メディア形式には、[RFC7874] の 3 章および [RFC7742] の 5 章で指定されている必須のオーディオ/ビデオコーデックが含まれている必要がある[MUST]。https://www.rfc-editor.org/rfc/rfc7742

"m="行の直後には、[RFC4566] の 5.7 節で指定されているように、"c="行が続く必要がある[MUST]。ここでも、候補がまだないため、"c="行には、[RFC8840] の 4.1.1 項で定義されているように、デフォルト値"IN IP4 0.0.0.0"が含まれている必要がある[MUST]。

[RFC8859] は、SDP 属性を異なるカテゴリにグループ化する。バンドル時の不要な重複を避けるために、カテゴリ IDENTICAL または TRANSPORT の属性は、バンドルされた"m="セクションで繰り返してはならず[MUST NOT]、[RFC8843] の 7.1.3 項のガイダンスを繰り返す。https://www.rfc-editor.org/rfc/rfc8843 これには、バンドルがネゴシエートされ、まだ必要な"m="セクションと、バンドル専用としてマークされた"m="セクションが含まれる。

IDENTICAL または TRANSPORT 以外のカテゴリである次の属性は、各"m="セクションに含まれている必要がある[MUST]。

- [RFC5888] の 4 章で指定されている"a=mid"行。すべての MID 値は、ランダムまたは PeerConnection 単位のカウンタを使用するなど、ユーザ情報をリークしない方法で生成する必要があり、[RFC8843] の 15.2 節で定義されている RTP ヘッダー拡張に効率的に適合できるように、3 バイト以下にする必要が

ある。https://www.rfc-editor.org/rfc/rfc8843 これは、説明が適用される場合にのみ発生するため、RtpTransceiver mid プロパティを設定しないことに注意すること。生成された MID 値は、この時点で「提案された」MID と見なすことができる。

- 関連付けられたトランシーバと同じ方向属性。
- [RFC4566] の 6 章および [RFC3264] の 5.1 節で指定されているように、"m="行、"a=rtpmap"および"a=fmtp"行の各メディア形式。
- RTP 再送信を使用する必要がある各プライマリコーデックについて、[RFC4588] の 8.1 節で指定されているように、プライマリコーデックのクロックレートで"rtx"を示す対応する"a=rtpmap"行と、プライマリコーデックのペイロードタイプを参照する"a=fmtp"行。
- サポートされている各 Forward Error Correction (FEC) メカニズムについて、[RFC4566] の 6 章で指定されているように、"a=rtpmap"行と"a=fmtp"行。サポートされなければならない FEC メカニズムは [RFC8854] の 7 章で指定されており、各メディアタイプの具体的な使用方法は 4 章と 5 章で概説されている。
- この"m="セクションが、オーディオなど、パケットあたりのメディアの持続時間を設定可能なメディアの場合、"a=maxptime"行は、[RFC4566] の 6 章で指定されているように、各パケットでカプセル化できるミリ秒単位で指定されたメディアの最大量を示す。この値は、"m="セクションに含まれるすべてのコーデックにわたって最大持続時間値の最小値に設定される。
- この"m="セクションがビデオメディア用であり、デコード可能なイメージのサイズに既知の制限がある場合は、3.6 節で指定されているように、"a=imageattr"行。
- サポートされている RTP ヘッダー拡張ごとに、[RFC5285] の 5 章で指定されている"a=extmap"行。サポートすべき[SHOULD]/しなければならない[MUST]ヘッダー拡張のリストは、[RFC8834] の 5.2 節で指定されている。暗号化を必要とするすべてのヘッダー拡張は、[RFC6904] の 4 章で示されているように指定する必要がある[MUST]。
- サポートされている RTCP フィードバックメカニズムごとに、[RFC4585] の 4.2 節で指定されているように、"a=rtcp-fb"行。サポートすべき[SHOULD]/しなければならない[MUST] RTCP フィードバックメカニズムのリストは、[RFC8834] の 5.1 節で指定されている。
- RtpTransceiver に sendrecv または sendonly の方向がある場合:

  ◦addTrack または addTransceiver を介して作成されたときにトランシーバに関連付けられた各 MediaStream について、[RFC8830] の 2 章で指定されているが、"appdata"フィールドを省略した"a=msid"行。

- RtpTransceiver の方向が sendrecv または sendonly で、アプリケーションがエンコーディングに rid-id を指定している場合、または RtpSenders のパラメータで複数のエンコーディングを指定している場合は、指定された各エンコーディングの"a=rid"行。"a=rid"行は [RFC8851] で指定されており、その方向は"send"でなければならない[MUST]。アプリケーションが rid-id を選択した場合、それを使用しなければならない[MUST]。それ以外の場合、rid-id は実装によって生成されなければならない[MUST]。rid-ids は、ランダムに、または perPeerConnection カウンタ([RFC8852] の 3.3 節の最後にあるガイダンスを参照)を使用するなど、ユーザ情報をリークしない方法で生成する必要があり、3 バイト以下にする必要がある[SHOULD]。これは、[RFC8852] の 3.3 節で定義されている RTP ヘッダー拡張に効率的に適合できるようにするためである。エンコーディングが指定されていない場合、またはエンコーディングが 1 つだけ指定されていても rid-id が指定されていない場合、"a=rid"行は生成されない。
- RtpTransceiver の方向が sendrecv または sendonly で、複数の"a=rid"行が生成されている場合、[RFC8853] の 5.1 節で定義されているように、方向が"send"の"a=simulcast"行が生成される。https://www.rfc-

editor.org/rfc/rfc8853 関連する rid-id のセットには、この"m="セクションの"a=rid"行で使用されるすべての rid-id が含まれている必要がある[MUST]。

- このPeerConnectionのバンドルポリシーが"max-bundle"に設定されていて、これが最初の"m="セクションではない場合、または(2)バンドルポリシーが"balanced"に設定されていて、これがこのメディアタイプの最初の"m="セクションではない場合、"a=bundle-only"行。

次の属性は、カテゴリ IDENTICAL または TRANSPORT であり、一意のアドレスを持つか、BUNDLE タグに関連付けられている"m="セクションにのみ表示される必要がある[MUST]。(初期オファーでは、これは"a=bundle-only"属性を含まない"m="セクションを意味する。)

- [RFC8839] の 5.4 節で指定されているように、"a=ice-ufrag"行と"a=ice-pwd"行。
- 必要なダイジェストアルゴリズムごとに、[RFC8122] の 5 章で指定されているように、エンドポイントの証明書ごとに 1 つ以上の"a=fingerprint"行。
- [RFC4145] の 4 章で指定されており、[RFC5763] の 5 章で DTLS-SRTP シナリオで使用するために明確にされている"a=setup"行。https://www.rfc-editor.org/rfc/rfc5763 オファーのロール値は"actpass"である必要がある。
- [RFC8842] の 5.2 節で指定されている"a=tls-id"行。
- [RFC3605] の 2.1 節で指定されている"a=rtcp"行。デフォルト値"9 IN IP4 0.0.0.0"が含まれている。候補がまだ収集されていないためである。
- [RFC5761] の 5.1.3 項で指定されている"a=rtcp-mux"行。
- RTP/RTCP 多重化ポリシーが"require"の場合、[RFC8858] の 4 章で指定されている"a=rtcp-mux-only"行。
- [RFC5506] の 5 章で指定されている"a=rtcp-rsize"行。

最後に、データチャネルが作成されている場合は、データの"m="セクションを生成する必要がある[MUST]。

<media>フィールドを"application"に設定し、<proto>フィールドを"UDP/DTLS/SCTP" [RFC8841] に設定する必要がある[MUST]。<fmt>値は、[RFC8841] の 4.2.2 項で指定されている"webrtc-datachannel"に設定する必要がある[MUST]。

データ"m="セクション内に、"a=mid"行が生成され、[RFC8841] の 5.1 節で定義されているように、SCTP ポート番号を参照する"a=sctp-port"行とともに、上記のように含まれている必要がある[MUST]。また、適切な場合は、[RFC8841] の 6.1 節で定義されているように、"a=max-message-size"行が含まれている必要がある。

前述のように、次のカテゴリ IDENTICAL または TRANSPORT の属性は、データ"m="セクションが一意のアドレスを持つか、BUNDLE タグ(例えば、それが唯一の"m="セクションである場合)に関連付けられている場合にのみ含まれる。

- "a=ice-ufrag"
- "a=ice-pwd"
- "a=fingerprint"
- "a=setup"
- "a=tls-id"

すべての"m="セクションが生成されたら、[RFC5888] で指定されているように、セッションレベルの"a=group"属性を追加する必要がある[MUST]。この属性は"BUNDLE"という意味を持ち、各"m="セクションの中間識別子を含まなければならない[MUST]。その結果、JSEP 実装ではすべての"m="セクションが 1 つのバンドルグループとして提供される。ただし、"m="セクションがバンドル専用かどうかは、バンドルポリシーによって異なる。

次の手順では、[RFC5888] の 7 章で定義されているセッションレベルのリップシンクグループを生成する。

https://www.rfc-editor.org/rfc/rfc5888 複数の RtpTransceiver によって参照される各 MediaStream について (これらの MediaStream を addTrack メソッドと addTransceiver メソッドの引数として渡すことによって) 、各 RtpTransceiver の MID 値を含むタイプ"LS"のグループを追加する必要がある[MUST]。

SDP がセッションレベルまたはメディアレベルのいずれかであることを許可する属性は、それらが同一であっても、一般的にメディアレベルである必要がある[SHOULD]。これは、特に最初は同一の属性のセットの 1 つが後で変更された場合に、個々のメディアセクションを理解しやすくすることによって、開発とデバッグを支援する。ただし、実装はセッションレベルで属性を集約することを選択する場合があり、JSEP 実装はいずれかの場所で属性を受け取る準備が必要である[MUST]。

上記で指定された属性以外の属性を含めることができるが[MAY]、[RFC8834] の要件と特に互換性がなく、含めることはできない[MUST NOT]。

- "a=crypto"
- "a=key-mgmt"
- "a=ice-lite"

バンドルを使用する場合、追加される属性は、それらの属性がバンドルとどのように相互作用するかについての [RFC8859] のアドバイスに従う必要があることに注意すること。

これらの要件は、場合によっては SDP の要件よりも厳しいことに注意すること。実装は、この仕様で SDP を生成するための要件に準拠していない場合でも、準拠 SDP を受け入れる準備が必要である[MUST]。

5.2.2 後続のオファー

createOffer が 2 回目 (またはそれ以降) に呼び出された場合、またはローカルの記述がすでにインストールされている後に呼び出された場合、最初のオファーの場合とは処理が多少異なる。

以前のオファーが setLocalDescription を使用して適用されなかった場合、つまり PeerConnection がまだ「安定した」状態にある場合は、次の制限に従って、最初のオファーを生成する手順に従う必要がある[MUST]。

- "o="行のフィールドは、<session-version>フィールドを除いて同じままである必要がある[MUST]。これは、オファーが createOffer への以前のコールの出力と異なる可能性がある場合に、createOffer への各コールで 1 ずつインクリメントする必要がある[MUST]。実装は、すべてのコールで<session-version>をインクリメントすることを選択できる[MAY]。生成された<session-version>の値は、現在のローカル記述の<session-version>とは無関係である。特に、現在のバージョンが N の場合、オファーが作成され、バージョン N+1 で適用された後、そのオファーがロールバックされて現在のバージョンが再び N になると、次の生成されたオファーはバージョン N+2 のままになる。

アプリケーションが createOffer を呼び出す代わりに currentLocalDescription を読み込んでオファーを作成した場合、収集された ICE 候補が追加されるため、返される SDP は setLocalDescription が最初に呼び出されたときとは異なる可能性があるが、<session-version>は変更されない。これが問題を引き起こす既知のシナリオはないが、これが懸念される場合は、createOffer を使用して一意の<session-version>を確保するだけで解決できる。

以前のオファーが setLocalDescription を使用して適用されたが、リモート側からの対応するアンサーがまだ適用されていない、つまり PeerConnection がまだ"have-localoffer"状態にある場合は、上記の"stable"状態の手順に従って、次の例外とともにオファーが生成さる。

- "s="行と"t="行は同じままでなければならない[MUST]。
- RtpTransceiver が追加されていて、現在のローカル記述または現在のリモート記述に 0 ポートの"m="セクションが存在する場合、"m="セクションがセッション記述に追加されているかのように (新しい MID 値を含む) 、追加された RtpTransceiver の"m="セクションを生成し、0 ポートの"m="セクションと

同じインデックスに配置することによって、その"m="セクションを再利用する必要がある[MUST]。

- RtpTransceiver が停止していて、"m="セクションに関連付けられていない場合、"m="セクションはその RtpTransceiver に対して生成されてはならない[MUST NOT]。これにより、前述のように、"m="セクションが再利用され、以前のオファー/アンサー交換で新しい RtpTransceiver に使用された RtpTransceiver を再び追加することができなくなる。
- RtpTransceiver が停止され、"m="セクションに関連付けられていて、"m="セクションが前述のように再利用されていない場合は、ポートを 0 に設定し、すべての"a=msid"行を削除して、"m="セクションを生成する必要がある[MUST]。
- 停止されていない RtpTransceiver の場合、"a=msid"行は、トランシーバの方向またはトラックの変更に関係なく、現在の説明に存在する場合は同じままである必要がある[MUST]。現在の説明に"a=msid"行が存在しない場合、"a=msid"行は、最初のオファーと同じ規則に従って生成される必要がある[MUST]。
- [RFC8839] の 4.2.1.2 項で説明され、5.1.2 項で明確にされているように、各"m="行と"c="行には、ポート、関連する RTP プロファイル、および"m="セクションのデフォルト候補のアドレスを入力する必要がある[MUST]。RTP 候補がまだ収集されていない場合は、前述のようにデフォルト値を使用する必要がある[MUST]。
- 各"a=mid"行は同じままでなければならない[MUST]。
- 各"a=ice-ufrag"行と"a=ice-pwd"行は、ICE 設定が変更されている(例えば、サポートされている STUN/TURN サーバや ICE 候補ポリシーの変更)か、IceRestart オプション (5.2.3.1 項) が指定されていない限り、同じままでなければならない[MUST]。"m="セクションが別の"m="セクションにバンドルされている場合でも、ICE 資格情報は含まれていてはならない[MUST NOT]。
- "m="セクションが別の"m="セクションにバンドルされていない場合、その"a=rtcp"属性行には、[RFC5761] の 5.1.3 項に示されているように、デフォルトの RTCP 候補のポートとアドレスを入力する必要がある[MUST]。RTCP 候補がまだ収集されていない場合は、上記の 5.2.1 項で説明されているように、デフォルト値を使用する必要がある[MUST]。
- "m="セクションが別の"m="セクションにバンドルされていない場合は、最新の収集フェーズ (3.5.1 項を参照) で収集された各候補について、[RFC8839] の 5.1 節で定義されているように、"a=candidate"行を追加する必要がある[MUST]。セクションの候補の収集が完了した場合は、[RFC8840] の 8.2 節で説明されているように、"a=end-of-candidates"属性を追加する必要がある[MUST]。"m="セクションが別の"m="セクションにバンドルされている場合は、"a=candidate"と"a=end-of-candidates"の両方を省略する必要がある[MUST]。
- まだ存在する RtpTransceiver の場合、"a=rid"行は同じままでなければならない[MUST]。
- 存在している RtpTransceiver の場合、すべての"a=simulcast"行は同じままでなければならない[MUST]。

前のオファーが setLocalDescription を使用して適用され、リモート側からの対応するアンサーが setRemoteDescription を使用して適用されている場合、つまり PeerConnection が"have-remote-pranswer"状態または"stable"状態にある場合、上記の"have-local-offer"状態で説明されている手順に従って、ネゴシエートされたセッションの説明に基づいてオファーが生成される。

さらに、新しいオファーの既存の、再利用されていない、拒否されていない"m="セクションごとに、必要に応じて、現在のローカルまたはリモートの説明の対応する"m="セクションの内容に基づいて、次の調整が行われる。

- "m="行と対応する"a=rtpmap"および"a=fmtp"行には、関連するトランシーバのコーデック設定によって除外されていないメディア形式のみが含まれている必要があり[MUST]、また、現在使用可能なすべての形式も含まれている必要がある[MUST]。以前に提供されていたが、現在は利用できないメディア

形式(例:共有ハードウェアコーデック)は除外してもかまわない[MAY]。

- 関連するトランシーバにコーデックの設定が設定されていない限り、"m="行のメディア形式は、最新のアンサーと同じ順序で生成する必要がある[MUST]。最新のアンサーに存在しなかったメディア形式は、既存のすべての形式の後に追加する必要がある[MUST]。
- RTP ヘッダー拡張には、最新のアンサーに存在するもののみを含める必要がある[MUST]。
- RTCP フィードバックメカニズムには、新しく追加されたメディア形式を参照している形式固有のメカニズムの場合を除き、最新のアンサーに存在するもののみを含める必要がある[MUST]。
- 最新のアンサーに"a=rtcp-mux"行が含まれている場合は、"a=rtcp"行を追加してはならない[MUST NOT]。
- "a=rtcp-mux"行は、最新のアンサーと同じである必要がある[MUST]。
- "a=rtcp-mux-only"行を追加してはならない[MUST NOT]。
- "a=rtcp-rsize"行は、最新のアンサーに存在しない限り、追加してはならない[MUST NOT]。
- [RFC8843] の 6 章で定義されている"a=bundle-only"行は、追加してはならない[MUST NOT]。代わりに、JSEP 実装は、[RFC8843] の 7.1.3 項で説明されているように、バンドルされた"m="セクションの IDENTICAL および TRANSPORT カテゴリのパラメータを単に省略しなければならない[MUST]。
- メディア"m="セクションがデータ"m="セクションにバンドルされている場合、特定の TRANSPORT および IDENTICAL 属性が、それ以外の場合はメディア"m="セクションにのみ適している場合でも、データ"m="セクションに表示されることがある(例:"a=rtcp-mux")。JSEP 実装では、最初のオファーでは常にメディア"m="セクション (存在する場合) がデータ"m="セクション (存在する場合) の前にリストされ、それらのメディア"m="セクションの少なくとも 1 つには"a=bundle-only"属性がないため、これは最初のオファーでは発生しない。したがって、最初のオファーでは、すべての"a=bundle-only" "m="セクションは、先行する非バンドル専用メディア"m="セクションにバンドルされる。

"a=group:BUNDLE"属性には、最新のアンサーのバンドルグループで指定された MID 識別子から、拒否としてマークされたすべての"m="セクションと、新しく追加または再有効化されたすべての"m="セクションを引いたものを含める必要がある[MUST]。つまり、バンドル属性には、新しい"m="セクションだけでなく、以前にバンドルされていたすべての"m="セクションが含まれている必要がある[MUST]。

"a=group:LS"属性は、最初のオファーの場合と同じ方法で生成されるが、識別された"m="セクションがまだ存在していて拒否されない限り、最新のアンサーに存在していたリップシンクグループは存在し続けなければならず[MUST]、以前から存在していた MID 識別子を含まなければならないという追加の規定がある[MUST]。また、グループには少なくとも 2 つの MID 識別子が含まれている。これにより、同期された"recvonly""m="セクションは、新しいオファーでも引き続き同期される。

5.2.3 オプションの処理

createOffer メソッドは、RTCOfferOptions オブジェクトをパラメータとして受け取る。次のオプションが存在する場合、SDP 記述を生成するときに特別な処理が実行される。

5.2.3.1 IceRestart (Ice 再起動)

IceRestart オプションが"true"の値で指定されている場合、オファーは、[RFC8839] の 4.4.3.1.1 項で指定されているように、新しい ICE ufrag 属性と pwd 属性を生成して ICE 再起動を示す必要がある[MUST]。このオプションが最初のオファーで指定されている場合、効果はない (新しい ICE ufrag と pwd がすでに生成されているため)。同様に、ICE の設定が変更されている場合、新しい ufrag 属性と pwd 属性が自動的に生成されるため、このオプションは無効となる。このオプションは主に、アプリケーションによって障害が検出された場合に接続を再確立するために役立つ。

5.2.3.2 VoiceActivityDetection (VoiceActivity の検出)

不連続伝送 ("DTX") とも呼ばれる無音抑制は、音声アクティビティが検出されないときに特殊なエンコードに切り替えることで、音声に使用される帯域幅を減らすことができるが、ある程度の忠実性は犠牲になる。

"VoiceActivityDetection"オプションが"true"の値で指定されている場合、オファーは、独自の内部無音抑制サポートを持つコーデックを除き、[RFC3389] の 5.1 節で指定されているように、提供される各オーディオコーデックのコンフォートノイズ ("CN") コーデックを含めることによって、受信するオーディオで無音抑制のサポートを示す必要がある[MUST]。独自の内部無音抑制サポートを持つコーデックの場合、そのコーデックの適切な fmtp パラメータを指定して、受信オーディオの無音抑制が必要であることを示す必要がある[MUST]。たとえば、Opus コーデック [RFC6716] を使用する場合、[RFC7587] で指定されている"usedtx=1"パラメータがオファーで使用される。

"VoiceActivityDetection"オプションが"false"の値で指定されている場合、JSEP 実装は"CN"コーデックを出力してはならない[MUST NOT]。独自の内部無音抑制サポートを持つコーデックの場合、受信オーディオの無音抑制が望ましくないことを示すために、そのコーデックの適切な fmtp パラメータを指定する必要がある[MUST]。たとえば、Opus コーデックを使用する場合、オファーには"usedtx=0"パラメータが指定される。さらに、実装は、"CN"コーデックまたは関連する fmtp パラメータがピアの説明に表示されるかどうかに関係なく、生成するメディアに無音抑制を使用してはならない[MUST NOT]。これらの規則の影響は、JSEP での無音抑制が双方の合意に依存していることであり、使用されるコーデックに関係なく一貫した処理が保証される。

"VoiceActivityDetection"オプションは、[RFC6464] の 4 章で説明されているクライアント/ミキサー音声レベルヘッダー拡張のシグナリングの"vad"値の設定には影響しない。https://www.rfc-editor.org/rfc/rfc6464

5.3 アンサーの生成

createAnswer が呼び出されると、提供されたリモート記述および [RFC8834] で指定された要件と互換性のある新しい SDP 記述を作成する必要がある[MUST]。この処理の正確な詳細について、以下で説明する。

5.3.1 初期アンサー

リモート記述が提供された後に createAnswer が初めて呼び出された場合、その結果は初期アンサーと呼ばれる。リモート記述がインストールされていない場合、アンサーは生成できず、エラーを返す必要がある[MUST]。

リモート記述の SDP は JSEP エンドポイントによって作成されていない可能性があり、5.2 節に記載されているすべての要件に準拠していない可能性があることに注意する。多くの場合、これは問題ではない。ただし、必須の SDP 属性が欠落しているか、上記の必須としてリストされている機能が存在しない場合、これはエラーとして扱われる必要があり[MUST]、該当する"m="セクションを拒否としてマークする必要がある[MUST]。

初期アンサーを生成する最初の手順は、セッションレベルの属性を生成することである。ここでの処理は、上記の 5.2.1 項で示されているものと同じだが、"a=ice-options"行に、[RFC8840] の 4.1.3 項で規定されている"trickle"オプションと、[RFC8445] の 10 章で規定されている"ice2"オプションを含むのは、そのようなオプションがオファーに存在した場合のみである。https://www.rfc-editor.org/rfc/rfc8840

次のステップは、[RFC5888] の 7 章で定義されているように、セッションレベルのリップシンクグループを生成することである。オファーに存在するタイプ"LS"の各グループについて、指定されたグループの MID 値によって参照されるローカル RtpTransceivers を選択し、そのうちのどれが共通のローカル MediaStream (作成に使用される addTrack/addTransceiver の呼び出しで指定される) を参照するか、または

addTrack/addTransceiver によって作成されなかったために参照する MediaStream がないかを決定する。そのような RtpTransceivers が少なくとも 2 つ存在する場合は、これらの RtpTransceivers の MID 値を持つタイプ"LS"のグループを追加する必要がある[MUST]。それ以外の場合、提供された"LS"グループは無視され、対応するグループがアンサーに生成されないようにする必要がある[MUST]。

簡単な例として、同じ MediaStream に含まれる単一のオーディオおよび単一のビデオトラックの次のオファーを考える。この例に関連しない SDP 行は、明確にするために削除されている。5.2 節で説明したように、各トラックの RtpTransceiver を参照するタイプ"LS"のグループが追加された。

```
a=group:LS a1 v1
m=audio 10000 UDP/TLS/RTP/SAVPF 0
a=mid:a1
a=msid:ms1
m=video 10001 UDP/TLS/RTP/SAVPF 96
a=mid:v1
a=msid:ms1
```

アンサー側がトラックを追加するときに 1 つの MediaStream を使用する場合、両方のトランシーバがこのストリームを参照するため、次のように、後続のアンサーにはオファーのものと同じ"LS"グループが含まれる。

```
a=group:LS a1 v1
m=audio 20000 UDP/TLS/RTP/SAVPF 0
a=mid:a1
a=msid:ms2
m=video 20001 UDP/TLS/RTP/SAVPF 96
a=mid:v1
a=msid:ms2
```

ただし、アンサー側がトラックを個別の MediaStream にグループ化した場合、そのトランシーバは異なるストリームを参照するため、後続のアンサーには"LS"グループは含まれない。

```
m=audio 20000 UDP/TLS/RTP/SAVPF 0
a=mid:a1
a=msid:ms2a
m=video 20001 UDP/TLS/RTP/SAVPF 96
a=mid:v1
a=msid:ms2b
```

最後に、アンサー側がトラックを追加しない場合、そのトランシーバは MediaStream を参照しないため、オファー側の設定が維持され、後続のアンサーには同じ"LS"グループが含まれる。

```
a=group:LS a1 v1
m=audio 20000 UDP/TLS/RTP/SAVPF 0
a=mid:a1
a=recvonly
m=video 20001 UDP/TLS/RTP/SAVPF 96
a=mid:v1
a=recvonly
```

7.2 節の例では、より複雑な"LS"グループ生成のケースを示している。

次のステップでは、[RFC3264] の 6 章で規定されているように、リモートオファーに存在する各"m="セクションに対して"m="セクションを生成する。https://www.rfc-editor.org/rfc/rfc3264 この議論では、メディアレベル属性としても有効なオファーのセッションレベル属性は、各"m="セクションに存在すると見なされる。提供される各"m="セクションには、5.10 節で説明されているように、関連付けられた RtpTransceiver がある。"m="セクションの数よりも多くの RtpTransceiver がある場合は、一致しない RtpTransceiver を後続のオファー

で関連付ける必要がある。

提供された各"m="セクションについて、次の条件のいずれかに当てはまる場合は、[RFC3264] の 6 章に示されているように、"m="行の<port>を 0 に設定して、アンサーの対応する"m="セクションを拒否としてマークする必要があり[MUST]、この"m="セクションのさらなる処理を省略できる。

- 関連付けられた RtpTransceiver が停止された
- コーデック設定でサポートされており、該当する場合は許可されたメディア形式が提供されていない
- バンドルポリシーが"max-bundle"で、これが最初の"m="セクションでも、最初の"m="セクションと同じバンドルグループでもない
- バンドルポリシーが"balanced"で、これがこのメディアタイプの最初の"m="セクションでも、このメディアタイプの最初の"m="セクションと同じバンドルグループでもない
- この"m="セクションがバンドルグループ内にあり、そのグループのオファー側タグ付きの"m="セクションが、上記のいずれかの理由で拒否されている。これには、[RFC8843] の 7.3.3 項で規定されているように、バンドルグループ内のすべての"m="セクションが拒否される必要がある。

それ以外の場合、アンサーの各"m="セクションは、[RFC3264] の 6.1 節で規定されているように生成されなければならない[MUST]。"m="行自体については、次の規則に従う必要がある[MUST]。

- <port>値は、通常、この"m="セクションのデフォルト ICE 候補のポートに設定されるが、候補がまだ使用できない場合は、[RFC8840] の 4.1.1 項に示されているように、デフォルトの<port>値 9 (破棄) を使用する必要がある[MUST]。
- <proto>フィールドは、オファー内の対応する"m="行の<proto>フィールドと正確に一致するように設定する必要がある[MUST]。
- 関連するトランシーバにコーデック設定が設定されている場合、提供されたものに関係なく、対応する順序でメディア形式を生成する必要があり[MUST]、コーデック設定に存在しないコーデックを除外する必要がある[MUST]。
- それ以外の場合、"m="行のメディア形式は、現在サポートされていない形式を除き、現在のリモート記述で提供されているものと同じ順序で生成する必要がある[MUST]。現在のリモート記述に存在しない現在利用可能なメディア形式は、既存のすべての形式の後に追加する必要がある[MUST]。
- どちらの場合も、アンサーのメディア形式には、オファーに存在する少なくとも 1 つの形式が含まれている必要があるが[MUST]、[RFC3264] の 6.1 節で言及されているように、ローカルでサポートされているがオファーには存在しない形式を含めることができる[MAY]。共通の形式が存在しない場合、前述のように"m="セクションは拒否される。

"m="行の直後には、[RFC4566] の 5.7 節で指定されているように、"c="行が必要である[MUST]。ここでも、候補がまだないため、"c="行には、[RFC8840] の 4.1.3 項で定義されているように、デフォルト値"IN IP4 0.0.0.0"が含まれている必要がある[MUST]。

オファーがバンドルをサポートしている場合、バンドルされるすべての"m="セクションは、同じ ICE 資格情報と候補を使用する必要がある[MUST]。バンドルされないすべての"m="セクションは、一意の ICE 資格情報と候補を使用する必要がある[MUST]。各"m="セクションには、次の属性 (IDENTICAL または TRANSPORT 以外の属性タイプ) が含まれている必要がある[MUST]。

- オファーに存在する場合に限り、[RFC5888] の 9.1 節で規定されている"a=mid"行。MID 値は、オファーで指定されたものと一致する必要がある[MUST]。
- [RFC3264] の 6.1 節で規定されている提供された方向に関するルールを適用し、関連付けられた RtpTransceiver の方向と交差することによって決定される方向属性。たとえば、"m="セクションが

"sendonly"として提供され、ローカルトランシーバが"sendrecv"に設定されている場合、アンサーの結果は"recvonly"方向になる。

- [RFC4566] の 6 章および [RFC3264] の 6.1 節で規定されている、"m="行の各メディア形式に対する、"a=rtpmap"行および"a=fmtp"行
- オファーに"rtx"が存在する場合、RTP 再送信を使用する必要がある各プライマリコーデックについて、[RFC4588] の 8.1 節で規定されている、プライマリコーデックのクロックレートに対応する"rtx"を示す"a=rtpmap"行と、プライマリコーデックのペイロードタイプを参照する"a=fmtp"行。
- サポートされている各 FEC メカニズムについて、[RFC4566] の 6 章で規定されている、"a=rtpmap"および"a=fmtp"行。サポートされなければならない[MUST]FEC メカニズムは、[RFC8854] の 7 章で規定されており、各メディアタイプの具体的な使用方法は、4 章と 5 章で概説されている。
- この"m="セクションが、オーディオなど、パケットあたりのメディアの継続時間が設定可能なメディアの場合、5.2 節で説明されている"a=maxptime"行。
- この"m="セクションがビデオメディアの場合で、デコード可能なイメージのサイズに既知の制限がある場合、3.6 節で指定されている"a=imageattr"行。
- オファーに存在するサポートされている各 RTP ヘッダー拡張に対して、[RFC5285] の 5 章で規定されている"a=extmap"行。サポートしなければならない[SHOULD/MUST]ヘッダー拡張のリストは、[RFC8834] の 5.2 節で規定されている。https://www.rfc-editor.org/rfc/rfc8834 暗号化を必要とするヘッダー拡張は、[RFC6904] の 4 章で示されているように指定する必要がある[MUST]。
- オファーに存在するサポートされている各 RTCP フィードバックメカニズムに対して、[RFC4585] の 4.2 節で規定されている"a=rtcp-fb"行。サポートしなければならない[SHOULD/MUST]RTCP フィードバックメカニズムのリストは、[RFC8834] の 5.1 節で規定されている。
- RtpTransceiver に sendrecv または sendonly の方向がある場合:
    - ◦addTrack または addTransceiver を介して作成されたときにトランシーバに関連付けられた各 MediaStream について、[RFC8830] の 2 章で規定されている"a=msid"行。ただし、"appdata"フィールドは省略する。

別の"m="セクションにバンドルされていない各"m="セクションには、次の属性 (IDENTICAL または TRANSPORT カテゴリ) が含まれている必要がある[MUST]。

- [RFC8839] の 5.4 節で規定されている"a=ice-ufrag"行と"a=ice-pwd"行。
- [RFC8122] の 5 章で規定されている、必要なダイジェストアルゴリズムごとに、各エンドポイントの証明書のための 1 つ以上の"a=fingerprint"行。
- "a=setup"行は、[RFC4145] の 4 章で規定されており、[RFC5763] の 5 章で DTLS-SRTP シナリオで使用するために明確にされている。https://www.rfc-editor.org/rfc/rfc4145https://www.rfc-editor.org/rfc/rfc5763 アンサーのロール値は"active"または"passive"である必要がある[MUST]。オファーに"actpass"値が含まれている場合、JSEP エンドポイントの場合と同様に、アンサー側は"active"ロールを使用する必要がある[SHOULD]。JSEP 以外のエンドポイントからのオファーは、"a=setup"の他の値を送信する場合があり[MAY]、その場合、アンサーはオファーの値と一致する値を使用する必要がある[MUST]。
- [RFC8842] の 5.3 節で規定されている"a=tls-id"行。
- オファーに存在する場合は、[RFC5761] の 5.1.3 項で規定されている"a=rtcp-mux"行。それ以外の場合は、[RFC3605] の 2.1 節で規定されている"a=rtcp"行で、デフォルト値"9 IN IP4 0.0.0.0"が含まれている (候補がまだ収集されていないため) 。
- オファーに存在する場合は、[RFC5506] の 5 章で規定されている"a=rtcp-rsize"行。

データチャネルの "m="セクションが提供されている場合は、データの "m="セクションも生成する必要が

ある[MUST]。<media>フィールドは"application"に設定する必要があり[MUST]、<proto>フィールドと<fmt>フィールドはオファーのフィールドと正確に一致するように設定する必要がある[MUST]。

データ"m="セクション内には、前述のように"a=mid"行が生成され、[RFC8841] の 5.1 節で定義されている、SCTP ポート番号を参照する"a=sctp-port"行とともに含まれている必要がある[MUST]。また、適切な場合は、[RFC8841] の 6.1 節で定義されている"a=max-message-size"行が含まれている必要がある[MUST]。

前述のように、IDENTICAL または TRANSPORT カテゴリの次の属性は、データ"m="セクションが別の"m="セクションにバンドルされていない場合にのみ含まれる。

- "a=ice-ufrag"
- "a=ice-pwd"
- "a=fingerprint"
- "a=setup"
- "a=tls-id"

メディア"m="セクションがデータ"m="セクションにバンドルされている場合、特定の TRANSPORT 属性と IDENTICAL 属性は、メディア"m="セクションにのみ適している場合でも、データ"m="セクションにも表示されることがある(例:"a=rtcp-mux")。

"BUNDLE"のセマンティクスを持つ"a=group"属性が提供される場合、対応するセッションレベルの"a=group"属性は、[RFC5888] で規定されているように追加される必要がある[MUST]。これらの属性は、"BUNDLE"のセマンティクスを持つ必要があり[MUST]、拒否されていない提供されたバンドルグループのすべての中間識別子を含む必要がある[MUST]。オファー内の"a=bundle-only"の存在に関係なく、アンサー内のすべての"m="セクションに"a=bundle-only"行があってはならない[MUST NOT]ことに注意する。

すべての"m="セクション間で共通の属性は、セッションレベルで有効であると明示的に定義されている場合、セッションレベルに移動できる[MAY]。

オファーの作成で禁止されている属性は、アンサーの作成でも禁止されている。

5.3.2 後続のアンサー

createAnswer が 2 回目 (またはそれ以降) に呼び出された場合、またはローカル記述が既にインストールされている後に呼び出された場合、初期アンサーの場合とは処理が多少異なる。

前のアンサーが setLocalDescription を使用して適用されなかった場合、つまり PeerConnection がまだ"have-remote-offer"状態にある場合は、次の制限に従って、初期アンサーを生成する手順に従う必要がある[MUST]。

- "o="行のフィールドは、<session-version>フィールドを除いて同じままでなければならない[MUST]。これは、セッション記述が以前に生成されたアンサーから何らかの方法で変更された場合にインクリメントする必要がある[MUST]。

セッション記述が以前に setLocalDescription に提供されていた場合は、次の例外とともに、上記の"have-remote-offer"状態の手順に従ってアンサーが生成される。

- "s="行と"t="行は同じままでなければならない[MUST]。
- 各"m="行および"c="行には、[RFC8839] の 4.2.1.2 項で説明されているように、"m="セクションのデフォルト候補のポートとアドレスを入力する必要がある[MUST]。前述のように、"m="行プロトコルの値はオファーで提供されたものと一致する必要があるため[MUST]、特定のケースでは、"m="行プロトコルがデフォルト候補のものと一致しない場合があることに注意する。
- 各"a=ice-ufrag"行および"a=ice-pwd"行は、"m="セクションが再起動する場合を除き、同じままでなければならず[MUST]、その場合は、[RFC8839] の 4.4.1.1.1 項で規定されているように、新しい ICE 資格情報を作成する必要がある[MUST]。"m="セクションが別の"m="セクションにバンドルされている場

合でも、ICE 資格情報は含まれていてはならない[MUST NOT]。

- 各"a=tls-id"行は、オファー側の"a=tls-id"行が変更されない限り、同じままでなければならない[MUST]。その場合、[RFC8842] の 5.2 節で説明されているように、新しい tls-id 値を作成する必要がある[MUST]。
- 関連付けがオファー側によって継続されている場合、各"a=setup"行は、既存の DTLS 関連付けと一致する"active"または"passive"ロール値を使用しなければならない[MUST]。
- RTCP 多重化を使用する必要があり[MUST]、"m="セクションが以前に RTCP 多重化を使用していた場合にのみ"a=rtcp-mux"行が挿入される。
- "m="セクションが別の"m="セクションにバンドルされておらず、RTCP 多重化がアクティブでない場合は、"a=rtcp"属性行にデフォルトの RTCP 候補のポートとアドレスを入力する必要がある[MUST]。RTCP 候補がまだ収集されていない場合は、上記の 5.3.1 項で説明されているように、デフォルト値を使用する必要がある[MUST]。
- "m="セクションが別の"m="セクションにバンドルされていない場合は、最新の収集フェーズ (3.5.1 項を参照) で収集された各候補について、[RFC8839] の 5.1 節で定義されているように、"a=candidate "行を追加する必要がある[MUST]。セクションの候補の収集が完了した場合は、[RFC8840] のセクション 8.2 で説明されているように、"a=end-of-candidates "属性を追加する必要がある[MUST]。"m="セクションが別の"m="セクションにバンドルされている場合は、"a=candidate "と"a=end-of-candidates "の両方を省略する必要がある[MUST]。
- 停止されていない RtpTransceiver の場合、"a=msid"行は、トランシーバの方向またはトラックの変更に関係なく、同じままでなければならない[MUST]。現在の記述に"a=msid"行が存在しない場合、"a=msid"行は、初期アンサーと同じルールに従って生成されなければならない[MUST]。

5.3.3 オプションの処理

createAnswer メソッドは、RTCAnswerOptions オブジェクトをパラメータとして受け取る。RTCAnswerOptions のパラメータセットは、RTCOfferOptions でサポートされているものとは異なる。オファー側が ICE 再起動の実行を選択したすべての"m="セクションで ICE 資格情報が自動的に変更されるため、IceRestart オプションは不要である。

RTCAnswerOptions では、次のオプションがサポートされている。

5.3.3.1 VoiceActivityDetection

アンサーの無音抑制は、5.2.3.2 項で説明されているように処理されるが、1 つの例外がある。オファーで無音抑制のサポートが示されていない場合、VoiceActivityDetection パラメータは無効であり、VoiceActivityDetection が"false"に設定されているかのようにアンサーを生成する必要がある[MUST]。これはコーデックごとに実行される(たとえば、オファー側が何らかの方法で CN のサポートを提供したが、Opus に"usedtx=0"を設定した場合、VoiceActivityDetection を"true"に設定すると、CN コーデックと"usedtx=0"でアンサーが返される。)。このルールの影響は、アンサー側が無音抑制を提供しないエンドポイントで無音抑制を使用しようとしないことで、JSEP 以外のエンドポイントでも無音抑制が双方向にサポートされるようになることである。

5.4 オファーまたはアンサーの変更

createOffer または createAnswer から返された SDP は、setLocalDescription に渡す前に変更してはならない[MUST NOT]。SDP を正確に制御する必要がある場合は、前述の createOffer/createAnswer オプションまたは RtpTransceiver API を使用する必要がある[MUST]。

オファーまたはアンサーを使用して setLocalDescription を呼び出した後、有効な JSEP オファーまたはアンサーを定義する上記のルールに従う限り、アプリケーションは SDP を変更して、相手に送信する前にその機

能を減らすことができる[MAY]。同様に、相手からオファーまたはアンサーを受信したアプリケーションは、setRemoteDescription を呼び出す前に、同じ制約に従って、受信した SDP を変更できる[MAY]。

いつものように、アプリケーションは相手に送信するものに対して単独で責任を負い、すべての受信した SDP は、その機能の範囲内で JSEP 実装によって処理される。すべての SDP が適切に形成されていると仮定するのは間違いである。ただし、この仕様のどの実装でも、この仕様の他の実装からの変更されていない SDP をリモートのオファーまたはアンサーとして処理できると仮定できるはずである。

5.5 ローカル記述の処理

SessionDescription を setLocalDescription に渡す場合、次の手順を実行する必要がある[MUST]。

- 記述のタイプが"rollback"の場合は、5.7 節で定義されている処理に従い、このセクションの残りの部分で説明されている処理をスキップする。
- それ以外の場合は、SessionDescription のタイプが PeerConnection の現在の状態と照合される。
  - タイプが"offer"の場合、PeerConnection の状態は"stable"または"have-local-offer"のいずれかである必要がある[MUST]。
  - タイプが"pranswer"または"answer"の場合、PeerConnection の状態は"have-remote-offer"または"have-local-pranswer"のいずれかである必要がある[MUST]。
- 現在の状態に対してタイプが正しくない場合、処理を停止して[MUST]、エラーを返す必要がある[MUST]。
- 次に、SessionDescription がチェックされ、5.4 節で説明されているように、その内容が createOffer/createAnswer の最後の呼び出しで生成された内容と同一であり、したがって、変更されていないことを確認する。それ以外の場合は、処理を停止して[MUST]、エラーを返す必要がある[MUST]。
- 次に、以下の 5.8 節で説明するように、SessionDescription がデータ構造に解析される。
- 最後に、以下の 5.9 節で説明するように、解析された SessionDescription が適用される。

5.6 リモート記述の処理

SessionDescription を setRemoteDescription に渡す場合、次の手順を実行する必要がある[MUST]。

- 記述のタイプが"rollback"の場合は、5.7 節で定義されている処理に従い、このセクションの残りの部分で説明されている処理をスキップする。
- それ以外の場合は、SessionDescription のタイプが PeerConnection の現在の状態と照合される。
  - タイプが"offer"の場合、PeerConnection の状態は"stable"または"have-remote-offer"のいずれかである必要がある[MUST]。
  - タイプが"pranswer"または"answer"の場合、PeerConnection の状態は"have-local-offer"または"have-remote-pranswer"のいずれかである必要がある[MUST]。
- 現在の状態に対してタイプが正しくない場合は、処理を停止して[MUST]、エラーを返す必要がある[MUST]。
- 次に、以下の 5.8 節で説明するように、SessionDescription がデータ構造に解析される。何らかの理由で解析が失敗した場合、処理を停止して[MUST]、エラーを返す必要がある[MUST]。
- 最後に、下記の 5.10 節で説明するように、解析された SessionDescription が適用される。

5.7 ロールバックの処理

PeerConnection が"stable"以外の状態の場合、ロールバックを実行できる。これは、オファーと暫定的なアンサーの両方をロールバックできることを意味する。ロールバックは、提案された変更をキャンセルする場合にのみ使用できる。"stable"状態から以前の"stable"状態へのロールバックはサポートされていない。"stable"状態でロールバックが試行された場合、処理を停止し[MUST]、エラーを返す必要がある[MUST]。これは、

アンサー側がそのアンサーで setLocalDescription を実行した後は、ロールバックできないことを意味する。

ロールバックの効果は、setLocalDescription または setRemoteDescription が呼び出されたかどうかに関係なく同じである必要がある[MUST]。

ロールバックを処理するために、JSEP 実装は現在のオファー/アンサートランザクションを破棄し、シグナリング状態を"stable"に設定し、保留中のローカル記述またはリモート記述 (4.1.14 項および 4.1.16 項を参照) を"null"に設定する。破棄されたローカル記述によって割り当てられたリソースまたは候補は破棄される。受信されたメディアは、以前のローカル記述およびリモート記述に従って処理される。

ロールバックは、ロールバックされたセッション記述の適用(5.10 節および 5.9 節を参照)によって、"m="セクションに関連付けられていた RtpTransceivers の関連付けを解除する。これは、以前に関連付けられていた一部の RtpTransceivers が、どの"m="セクションにも関連付けられなくなることを意味する。このような場合、RtpTransceiver の mid プロパティの値を"null"に設定し[MUST]、トランシーバとその"m="セクションインデックス間のマッピングを破棄する必要がある[MUST]。その後ロールバックされたリモートオファーを適用して作成された RtpTransceiver を停止し、PeerConnection から削除する必要がある[MUST]。ただし、addTrack メソッドを介して RtpTransceiver にトラックが接続されている場合は、RtpTransceiver を削除してはならない[MUST NOT]。これは、アプリケーションが addTrack を呼び出した後、オファーを使用して setRemoteDescription を呼び出し、そのオファーをロールバックしてから createOffer を呼び出し、追加されたトラックの"m="セクションが生成されたオファーに表示されるようにするためである。

5.8 セッション記述の解析

セッション記述オブジェクトに含まれる SDP は、[RFC4566] の 5 章で説明されているように、キーバリュー表現を含むテキスト行の列で構成される。SDP は 1 行ずつ読み取られ、デシリアライズされた情報を含むデータ構造に変換される。ただし、SDP では多くの種類の行を使用できるが、そのすべてが JSEP アプリケーションに関連するわけではない。各行について、実装はまず定義されている ABNF に従って構文的に正しいことを確認し、[RFC4566] および [RFC3264] で使用されているセマンティクスに準拠していることを確認してから、次に説明するように、指定された値を解析して格納または破棄する。

行が適切に形成されていない場合、または説明どおりに解析できない場合、値を破棄する場合でも、パーサーはエラーで停止し、セッション記述を拒否する必要がある[MUST]。これにより、実装があいまいな SDP を誤って解釈しないようにする。

5.8.1 セッションレベルの解析

まず、セッションレベルの行がチェックされ、解析される。これらの行は、[RFC4566] の 5 章で定義されているように、特定の順序で、特定の構文で記述する必要がある[MUST]。特定の行タイプ(例:"v="、"c=")は定義された順序で記述する必要がある[MUST]が、同じタイプの行 (通常は"a=") は任意の順序で記述できる。

次の属性以外の行は JSEP コンテキストでは意味がなく、チェックされると破棄される場合がある[MAY]。

- "c="行は構文をチェックする必要がある[MUST]が、その値は [RFC8445] の 5.4 節で定義されている ICE 不一致検出にのみ使用される。WebRTC には ICE が必要であるため、JSEP 実装ではこの条件が発生しないことに注意する。
- "i="、"u="、"e="、"p="、"t="、"r="、"z="、"k="行は構文をチェックする必要がある[MUST]が、それ以外の場合、値は使用されない。

残りの非属性行は次のように処理される。

- "v="行は、[RFC4566] の 5.1 節で規定されているように、バージョン 0 である必要がある[MUST]。
- "o="行は、[RFC4566] の 5.2 節で規定されているように解析される必要がある[MUST]。
- "b="行が存在する場合は、[RFC4566] の 5.8 節で規定されているように解析され、bwtype と bandwidth

の値を保存する必要がある[MUST]。

最後に、属性行が処理される。特定の処理は、次のセッションレベル属性 ("a=") 行に適用する必要がある[MUST]。

- すべての"a=group"行は、[RFC5888] の 5 章で規定されているように解析され、グループのセマンティクスと mid が格納される。
- 存在する場合、単一の"a=ice-lite"行は、[RFC8839] の 5.3 節で規定されているように解析され、ice-lite の存在を示す値が格納される。https://www.rfc-editor.org/rfc/rfc8839
- 存在する場合、単一の"a=ice-ufrag"行は、[RFC8839] の 5.4 節で規定されているように解析され、ufrag 値が格納される。
- 存在する場合、単一の"a=ice-pwd"行は、[RFC8839] の 5.4 節で規定されているように解析され、パスワード値が格納される。
- 存在する場合、単一の"a=ice-options"行は、[RFC8839] の 5.6 節で規定されているように解析され、指定されたオプションのセットが格納される。
- 任意の"a=fingerprint"行は、[RFC8122] の 5 章で規定されているように解析され、フィンガープリントとアルゴリズムの値のセットが格納される。https://www.rfc-editor.org/rfc/rfc8122
- 存在する場合、1 つの"a=setup"行は、[RFC4145] の 4 章で規定されているように解析され、setup 値が格納される。https://www.rfc-editor.org/rfc/rfc4145
- 存在する場合、1 つの"a=tls-id"行は、[RFC8842] の 5 章で規定されているように解析され、属性値が格納される。
- すべての"a=identity"行は、[RFC8827] の 5 章で規定されているように解析され、その後の検証のために identity 値が格納される。https://www.rfc-editor.org/rfc/rfc8827
- すべての"a=extmap"行は、[RFC5285] の 5 章で規定されているように解析され、その値が格納される。https://www.rfc-editor.org/rfc/rfc5285

JSEP に関連しない他の属性も存在する可能性があり、実装は認識したものを処理する必要がある[SHOULD]。[RFC4566] の 5.13 節で要求されているように、不明な属性行は無視しなければならない[MUST]。

すべてのセッションレベルの行が解析されると、"m="セクションの行で処理が続行される。

5.8.2 メディアセクションの解析

セッションレベルの行と同様に、メディアセクションの行は、[RFC4566] の 5 章で定義されている特定の順序と特定の構文で出現しなければならない[MUST]。

"m="行自体は、[RFC4566] の 5.14 節で規定されているように解析され、<media>、<port>、<proto>、および<fmt>の値を保存する必要がある[MUST]。

"m="行に続いて、次の属性以外の行に特定の処理を適用する必要がある[MUST]。

- セッションレベルでの"c="行と同様に、"c="行は [RFC4566] の 5.7 節に従って解析されなければならない[MUST]が、その値は使用されない。
- "b="行が存在する場合は、[RFC4566] の 5.8 節で規定されているように解析され、bwtype と bandwidth の値を保存する必要がある[MUST]。

特定の処理は、次の属性行にも適用されなければならない[MUST]。

- 存在する場合、単一の"a=ice-ufrag"行が [RFC8839] の 5.4 節で規定されているように解析され、ufrag 値が格納される。
- 存在する場合、単一の"a=ice-pwd"行が [RFC8839] の 5.4 節で規定されているように解析され、パスワー

ド値が格納される。

- 存在する場合、単一の"a=ice-options"行が [RFC8839] の 5.6 節で規定されているように解析され、指定されたオプションのセットが格納される。
- すべての"a=candidate "属性は、[RFC8839] の 5.1 節で規定されているように解析され、その値が格納される必要がある[MUST]。
- すべての"a=remote-candidates"属性は、[RFC8839] の 5.2 節で規定されているように解析される必要がある[MUST]が、その値は無視される。
- 存在する場合、単一の"a=end-of-candidates"属性は、[RFC8840] の 8.1 節で規定されているように解析される必要があり[MUST]、その有無にフラグが立てられ、格納される。
- すべての"a=fingerprint"行は、[RFC8122] の 5 章で規定されているように解析され、フィンガープリントとアルゴリズムのセットが格納される。https://www.rfc-editor.org/rfc/rfc8122

上記 5.1.2 項で説明したように、"m="<proto>値が RTP の使用を示す場合、次の属性行を処理する必要がある[MUST]。

- "m="<fmt>値は、[RFC4566] の 5.14 節で規定されているように解析され、個々の値を保存する必要がある[MUST]。
- "a=rtpmap"または"a=fmtp"行は、[RFC4566] の 6 章で規定されているように解析され、それらの値を保存する必要がある[MUST]。
- 存在する場合、1 つの"a=ptime"行は、[RFC4566] の 6 章で規定されているように解析され、その値を保存する必要がある[MUST]。
- 存在する場合、1 つの"a=maxptime"行は、[RFC4566] の 6 章で規定されているように解析され、その値を保存する必要がある[MUST]。
- 存在する場合、1 つの方向属性行(例:"a=sendrecv")は、[RFC4566] の 6 章で規定されているように解析され、その値を保存する必要がある[MUST]。
- すべての"a=ssrc"属性は、[RFC5576] の 4.1 節で規定されているように解析され、その値を保存する必要がある[MUST]。
- すべての"a=extmap"属性は、[RFC5285] の 5 章で規定されているように解析され、それらの値を保存する必要がある[MUST]。
- すべての"a=rtcp-fb"属性は、[RFC4585] の 4.2 節で規定されているように解析され、それらの値を保存する必要がある[MUST]。
- 存在する場合、単一の"a=rtcp-mux"属性は、[RFC5761] の 5.1.3 項で規定されているように解析され、その有無にフラグを立てて保存する必要がある[MUST]。
- 存在する場合、単一の"a=rtcp-mux-only"属性は、[RFC8858] の 3 章で規定されているように解析され、その有無にフラグを立てて保存する必要がある[MUST]。
- 存在する場合、単一の"a=rtcp-rsize"属性は、[RFC5506] の 5 章で規定されているように解析され、その有無にフラグを立てて保存する必要がある[MUST]。
- 存在する場合、単一の"a=rtcp"属性は、[RFC3605] の 2.1 節で規定されているように解析する必要がある[MUST]が、この情報は ICE を使用する場合に不要であるため、その値は無視される。
- 存在する場合、"a=msid"属性は、[RFC8830] の 3.2 節で規定されているように解析され、その値を格納する必要があり[MUST]、"appdata"フィールドは無視される。"a=msid"属性が存在しない場合は、セッションの"default" MediaStream に対してランダムな msid-id 値が生成され、この値が格納される。
- すべての"a=imageattr"属性は、[RFC6236] の 3 章で規定されているように解析され、その値を保存する必要がある[MUST]。

- すべての"a=rid"行は、[RFC8851] の 10 章で規定されているように解析され、その値を保存する必要がある[MUST]。
- 存在する場合、単一の"a=simulcast"行は、[RFC8853] で規定されているように解析され、その値を保存する必要がある[MUST]。

それ以外の場合、"m="<proto>値が SCTP の使用を示している場合は、次の属性行を処理する必要がある[MUST]。

- "m="<fmt>値は、[RFC8841] の 4.3 節で規定されているように解析され、アプリケーションプロトコルの値を保存する必要がある[MUST]。
- "a=sctp-port"属性が存在する必要があり[MUST]、[RFC8841] の 5.2 節で規定されているように解析され、その値を保存する必要がある[MUST]。
- 存在する場合、単一の"a=max-message-size"属性は、[RFC8841] の 6 章で規定されているように解析され、その値を保存する必要がある[MUST]。それ以外の場合は、指定されたデフォルトを使用する。

JSEP に関連しない他の属性も存在する可能性があり、実装は認識したものを処理する必要がある[SHOULD]。[RFC4566] の 5.13 節で要求されているように、不明な属性行は無視しなければならない[MUST]。

5.8.3 セマンティクスの検証

解析が正常に完了したと仮定すると、解析された記述が評価され、必須機能の適切なサポートだけでなく、内部の一貫性が保証される。具体的には、次のチェックが行われる。

- 各"m="セクションについて、5.1.1 項に列挙されている各必須機能の有効な値が存在している必要がある[MUST]。これらの値は、メディアレベルに存在するか、セッションレベルから継承される[MAY]。
  - ICE ufrag 値とパスワード値。これは、[RFC8839] の 5.4 節で規定されたサイズ制限に準拠する必要がある[MUST]。
  - tls-id 値。[RFC8842] の 5 章に従って設定される必要がある [MUST]。これが再オファーまたは再オファーへの応答であり、tls-id 値が現在使用されている値と異なる場合、DTLS 接続は継続されず、リモート記述は新しい ufrag およびパスワード値とともに ICE 再起動の一部である必要がある[MUST]。
  - DTLS 設定値。[RFC5763] の 5 章で規定された規則に従って設定する必要があり[MUST]、現在の DTLS 接続の選択されたロールが存在し、継続されている場合は、それと一致している必要がある[MUST]。
  - DTLS フィンガープリント値。少なくとも 1 つのフィンガープリントが存在する必要がある[MUST]。
- "a=simulcast"行で参照されるすべての rid-id は、"a=rid"行として存在する必要がある[MUST]。
- 各"m="セクションもチェックされ、禁止されている機能が使用されていないことを確認する。
- RTP/RTCP 多重化ポリシーが"require"の場合、各"m="セクションに"a=rtcpmux"属性を含める必要がある[MUST]。"m="セクションに"a=rtcp-mux-only"属性が含まれる場合、そのセクションに"a=rtcp-mux"属性も含める必要がある[MUST]。
- "m="セクションが前のアンサーに存在する場合、RTP/RTCP 多重化の状態は、以前にネゴシエートされたものと一致する必要がある[MUST]。

このセッション記述のタイプが"pranswer"または"answer"の場合、次の追加チェックが適用される。

- セッション記述は、[RFC3264] の 6 章で定義されている規則に従う必要がある[MUST]。これには、"m="セクションの数が、関連付けられたオファーの"m="セクションの数と正確に一致する必要がある

[MUST]という要件も含まれる。https://www.rfc-editor.org/rfc/rfc3264

- 各"m="セクションについて、メディアタイプとプロトコルの値は、関連付けられたオファーの対応する"m="セクションのメディアタイプとプロトコルの値と正確に一致する必要がある[MUST]。

前のチェックのいずれかが失敗した場合、処理を停止して[MUST]、エラーを返す必要がある[MUST]。

5.9 ローカル記述の適用

ローカル記述を適用するには、メディアエンジンレベルで次の手順を実行する。エラーが返された場合、セッションはこれらの手順を実行する前の状態に復元する必要がある[MUST]。

まず、"m="セクションが処理される。"m="セクションごとに、次の手順を実行する必要がある[MUST]。いずれかのパラメータが範囲外であるか適用できない場合、処理を停止して[MUST]、エラーを返す必要がある[MUST]。

- この"m="セクションが新しい場合は、決定的にバンドルされていない限り((1) これはオファーであり、"m="セクションはバンドル専用とマークされているか、または (2) これはアンサーであり、"m="セクションは別の"m="セクションにバンドルされている) 、[RFC8445] の 5.1.1 項で定義されているように、その候補の収集を開始する。https://www.rfc-editor.org/rfc/rfc8445
- または、ICE ufrag とパスワードの値が変更されている場合は、[RFC8445] の 9 章の説明に従って ICE エージェントをトリガーして ICE 再起動を開始し、"m="セクションの新しい候補の収集を開始する。この記述がアンサーである場合は、そのメディアセクションのチェックも開始する。
- "m="セクション<proto>値が RTP の使用を示している場合:
  - この"m="セクションに関連付けられた RtpTransceiver がない場合は、次の手順に従って 1 つを検索し、この"m="セクションに関連付ける。この状況はオファーを適用する場合にのみ発生することに注意する。
    - トランシーバ間のマッピングとオファーの作成時に確立された"m="セクションインデックスを使用して、この"m="セクションに対応する RtpTransceiver を見つける。
    - この RtpTransceiver の mid プロパティの値を"m="セクションの MID に設定する。
  - RTCP の多重化が指定されている場合は、[RFC5761] の 5.1.3 項で規定されているように、RTP ICE コンポーネントから RTP と RTCP をデマックスする準備をする。
  - 指定された RTP ヘッダー拡張ごとに、[RFC5285] の 6 章で説明されているように、拡張 ID と URI の間のマッピングを確立する。https://www.rfc-editor.org/rfc/rfc5285
  - MID ヘッダー拡張がサポートされている場合、[RFC8843] の 15 章で説明されているように、MID ヘッダー拡張に基づいてこの"m="セクション用の RTP ストリームをデマックスする準備をする。
  - [RFC3264] の 6.1 節で説明されているように、指定されたメディア形式ごとに、ペイロードタイプと実際のメディア形式の間のマッピングを確立する。さらに、[RFC8843] の 9.2 節で説明されているように、この"m="セクションでサポートされているメディア形式に基づいて、この"m="セクション用の RTP ストリームをデマックスする準備をする。https://www.rfc-editor.org/rfc/rfc8843
  - 指定された"rtx"メディア形式ごとに、[RFC4588] の 8.6 節および 8.7 節で説明されているように、RTX ペイロードタイプとそれに関連するプライマリペイロードタイプ間のマッピングを確立する。https://www.rfc-editor.org/rfc/rfc4588
  - 方向属性のタイプが"sendrecv"または"recvonly"の場合、メディアの受信とデコードを有効にする。

最後に、この記述のタイプが"pranswer"または"answer"の場合は、以降の 5.11 節で定義されている処理に従う。

5.10 リモート記述の適用

リモート記述を適用するには、次の手順を実行する。エラーが返された場合、セッションをこれらの手順を実行する前の状態に復元する必要がある[MUST]。

アンサーに"trickle"が属性としてリストされている"a=ice-options"属性が含まれている場合は、PeerConnection canTrickleIceCandidatesプロパティを"true"に更新する。それ以外の場合は、このプロパティを"false"に設定する。

セッションレベルで属性に対して次の手順を実行する必要がある[MUST];いずれかのパラメータが範囲外であるか適用できない場合、処理を停止して[MUST]、エラーを返す必要がある[MUST]。

- 指定された"CT"帯域幅値については、[RFC4566] の 5.8 節で規定されているように、この値をすべての"m="セクションの最大合計ビットレートの制限として設定する。https://www.rfc-editor.org/rfc/rfc4566 この全体的な制限内で、実装は、個々の"m="セクションに指定されている特定の制限を考慮して、"m="セクション間で使用可能な帯域幅を最適に割り当てる方法を動的に決定できる。
- 指定された"RR"または"RS"の帯域幅値については、[RFC3556] の 2 章で規定されているように処理する。
- すべての"AS"帯域幅値([RFC4566] の 5.8 節)は、セッションレベルでのこのコンストラクトの意味が十分に定義されていないため、無視する必要がある[MUST]。

"m="セクションごとに、次の手順を実行する必要がある[MUST]。いずれかのパラメータが範囲外であるか適用できない場合、処理を停止して[MUST]、エラーを返す必要がある[MUST]。

- ICE ufragまたはパスワードが以前のリモート記述から変更された場合:
  - 記述のタイプが"offer"の場合、実装は [RFC8839] の 4.4.1.1.1 項に記載されているように、ICE の再起動が必要であることに注意しなければならない[MUST]。
  - 記述のタイプが"answer"または"pranswer"の場合、現在のローカル記述が ICE の再起動であるかどうかを確認し、そうでない場合はエラーを生成する。PeerConnection の状態が"have-remote-pranswer"で、ICE の ufrag またはパスワードが以前の暫定的なアンサーから変更された場合は、"m="セクションの以前の ICE チェックリストの状態を破棄するように ICE エージェントに通知する。最後に、チェックを開始するように ICE エージェントに通知する。
- 現在のローカル記述が ICE の再起動を示しているが、ICE ufrag もパスワードも以前のリモート記述([RFC8445] の 9 章で規定されている)から変更されていない場合は、エラーが生成される。
- このメディアセクションに関連付けられた ICE コンポーネントが、接続チェックに提供された ICE リモート ufrag とパスワードを使用するように設定する。
- [RFC8445] の 6.1.2 項で説明されているように、提供された ICE 候補と収集されたローカル候補をペアにして、適切な資格情報で接続チェックを開始する。
- "a=end-of-candidates "属性が存在する場合は、[RFC8838] の 14 章で説明されているように、end-of-candidates の通知を処理する。
- "m="セクション<proto>の値が RTP の使用を示している場合:
  - "m="セクションが再利用されている場合 (5.2.2 項を参照) 、mid プロパティを"null"に設定して現在関連付けられている RtpTransceiver の関連付けを解除し、トランシーバと"m="セクションインデックス間のマッピングを破棄する。
  - "m="セクションがどの RtpTransceiver にも関連付けられていない場合 (前のステップで関連付けが解除されている可能性がある) 、次の手順に従って RtpTransceiver を見つけるか、または作成する。
    - "m="セクションが sendrecv または recvonly で、addTrack によって PeerConnection に追加され、

どの"m="セクションにも関連付けられておらず、停止されていない同じタイプの RtpTransceiver がある場合は、そのような最初の (5.2.1 項で説明されている正規の順序による) RtpTransceiver を見つける。

- 前の手順で RtpTransceiver が見つからなかった場合は、recvonly 方向で作成する。
- RtpTransceiver の mid プロパティの値を"m="セクションの MID に設定して、見つかったまたは作成された RtpTransceiver を"m="セクションに関連付け、トランシーバと"m="セクションのインデックス間のマッピングを確立する。"m="セクションに MID が含まれていない(つまり、リモートエンドポイントは MID 拡張をサポートしていない)場合は、5.2.1 項に記載されている"a=mid"のガイダンスに従って、RtpTransceiver の mid プロパティの値を生成する。

◦ ローカル実装でもサポートされている指定されたメディア形式ごとに、[RFC3264] の 6.1 節の説明に従って、指定されたペイロードタイプとメディア形式の間のマッピングを確立する。具体的には、これは、指定された各メディア形式を送信するときに送信した RTP パケットで使用されるペイロードタイプと、その順序で示される各形式の相対的な優先順位を実装が記録することを意味する。指定されたメディア形式がローカル実装でサポートされていない場合は、無視する必要がある[MUST]。

◦ 指定された"rtx"メディア形式ごとに、[RFC4588] の 4 章で説明されているように、RTX ペイロードタイプとそれに関連付けられたプライマリペイロードタイプ間のマッピングを確立する。https://www.rfc-editor.org/rfc/rfc4588 参照されたプライマリペイロードタイプが存在しない場合、これはエラーになる必要がある[MUST]。RTX ペイロードタイプは、ローカルメディア実装でサポートされていないプライマリペイロードタイプを参照する場合があり、その場合は RTX ペイロードタイプも無視する必要がある[MUST]ことに注意する。

◦ ローカル実装でサポートされる指定された fmtp パラメータごとに、関連するメディア形式で有効にする。

◦ "m="セクションで通知される指定された同期ソース (SSRC) ごとに、[RFC8843] の 9.2 節の説明に従って、その SSRC を使用してこの"m="セクション用の RTP ストリームをデマックスする準備をする。

◦ ローカル実装でもサポートされる指定された RTP ヘッダー拡張ごとに、[RFC5285] の 5 章の説明に従って、拡張 ID と URI の間のマッピングを確立する。具体的には、実装は指定された各ヘッダー拡張を送信するときに、送信 RTP パケットで使用される拡張 ID を記録することを意味する。指定された RTP ヘッダー拡張がローカル実装でサポートされていない場合は、無視する必要がある[MUST]。

◦ ローカル実装でサポートされている指定された RTCP フィードバックメカニズムごとに、関連するメディア形式で有効にする。

◦ 指定された"TIAS" ("Transport Independent Application Specific Maximum") 帯域幅値に対して、この値を [RFC3890] で規定されているメディア送信時に使用される最大 RTP ビットレートの制約として設定する。"TIAS"値が存在せず、"AS"値が指定されている場合は、次の式を使用して"TIAS"値を生成する。

```
TIAS = AS * 1000 * 0.95 - (50 * 40 * 8)
```

1000 は単位を kbps から (TIAS で必要とされる) bps に変更し、0.95 は 5%を RTCP に割り当てる。次にヘッダーオーバーヘッドの推定値を減算する。50 は毎秒 50 パケットに基づき、40 は典型的なヘッダーサイズ (バイト単位) に基づき、8 はバイトをビットに変換する。"TIAS"は帯域幅をより正確に制御できるため、"AS"よりも優先されることに注意する。

- "RR"または"RS"の帯域幅値に対して、[RFC3556] の 2 章で規定されているとおりに処理する。
- メディアレベルでのこのコンストラクトの意味が十分に定義されていないため、指定された"CT"の帯域幅値はすべて無視しなければならない[MUST]。
- "m="セクションのタイプが"audio"の場合:
  - 指定された"CN"メディア形式ごとに、[RFC3389] の 5 章で説明されているように、サポートされているすべてのメディア形式に対して、独自の内部無音抑制メカニズムを持つ形式を除き、同じクロックレートで無音抑制を設定する。このような形式(例:Opus)の無音抑制は、5.2.3.2 項で説明されているように、fmtp パラメータを介して制御される。
  - 指定された各"telephone-event"メディア形式について、サポートされているすべてのメディア形式で同じクロックレートのデュアルトーン多重周波数 (DTMF) 伝送を有効にする。これについては、[RFC4733] の 2.5.1.2 項を参照すること。対応する telephone-event 形式を持たないサポートされているメディア形式がある場合は、それらの形式の DTMF 伝送を無効にする。
  - 指定した"ptime"値に対して、送信時に指定したパケットサイズを使用するように使用可能なメディア形式を設定する。指定したサイズがメディア形式でサポートされていない場合は、代わりに次に近い値を使用する。

最後に、この記述のタイプが"pranswer"または"answer"の場合は、以下の 5.11 節で定義されている処理に従う。

5.11 アンサーの適用

ローカルまたはリモート記述を処理する上記の手順に加えて、タイプ"pranswer"または"answer"の記述を処理する場合は、次の手順を実行する。

各 "m="セクションについて、次の手順を実行する必要がある[MUST]。

- "m="セクションが拒否された場合(つまり、アンサーで<port>の値が 0 に設定されている)、[RFC8839] の 4.4.3.1 項に記載されているように、このセクションのメディアの受信または送信を停止し、拒否されていない”m=“セクションがこの”m=“セクションにバンドルされていない限り、関連付けられている ICE コンポーネントを破棄する。https://www.rfc-editor.org/rfc/rfc8839
- リモート DTLS フィンガープリントが変更された場合、または"a=tls-id"属性の値が変更された場合は、DTLS 接続を切断する。これには、PeerConnection の状態が"have-remote-pranswer"の場合も含まれる。DTLS 接続を切断する必要があるが、アンサーが ICE の再起動を示していない場合、または"have-remote-pranswer"の場合で新しい ICE 資格情報を示していない場合は、エラーが生成される必要がある[MUST]。tls-id 値またはフィンガープリントを変更せずに ICE の再起動を実行した場合、同じ DTLS 接続が新しい ICE チャネルで継続される。JSEP では、オファー側が tls-id 値を変更した場合に限り、アンサー側は tls-id 値を変更する必要があるが、JSEP ではないアンサー側は、オファーに ICE の再起動が含まれている限り、tls-id 値を変更できる。したがって、アンサーを受信する前に DTLS データを処理する JSEP 実装では、ClientHello または以前の DTLS 接続からのデータを受信する準備が必要である[MUST]。
- 有効な DTLS 接続が存在しない場合は、基になる ICE コンポーネントがアクティブになったら、指定されたロールとフィンガープリントを使用して DTLS 接続を開始する準備をする。
- "m="セクション<proto>値が RTP の使用を示している場合:
  - ”m=“セクションがオファー内の対応する”m=“セクションに存在しなかった RTCP フィードバックメカニズムを参照している場合、これはネゴシエーションの問題を示しており、エラーになる必要がある[MUST]。ただし、[RFC3264] の 7 章および [RFC5285] の 6 章で説明されているように、新しいメディア形式および新しい RTP ヘッダー拡張の値はアンサーで許可される。

https://www.rfc-editor.org/rfc/rfc3264https://www.rfc-editor.org/rfc/rfc5285

- "m="セクションで RTCP の多重化が有効になっている場合、RTCP ICE コンポーネントが存在する場合はそれを破棄し、[RFC5761] の 5.1.3 項で指定されているように、RTP ICE コンポーネント上で RTCP の多重化を開始または継続する。そうでない場合は、RTCP ICE コンポーネント上で RTCP を送信する準備をする。RTCP の多重化が以前に有効になっていたために RTCP ICE コンポーネントが存在しない場合、これはエラーになる必要がある[MUST]。
- "m="セクションで Reduced-Size RTCP が有効になっている場合、[RFC5506] で指定されているように、この"m="セクションの RTCP 送信で Reduced-Size RTCP を使用するように設定する。
- アンサーの方向属性が、JSEP 実装がメディアを送信する必要があることを示している場合(ローカルアンサーの場合は"sendonly"、リモートアンサーの場合は"recvonly"、いずれかの種類のアンサーの場合は"sendrecv")、[RFC3264] の 6.1 節および 7 章で説明されているように、ローカルでもサポートされているリモート記述から最も優先されるメディア形式として送信するメディア形式を選択し、基盤となるトランスポート層が確立されたら、その形式を使用して RTP メディアの送信を開始する。https://www.rfc-editor.org/rfc/rfc3264https://www.rfc-editor.org/rfc/rfc3264 この送信 RTP ストリームに対して SSRC がまだ選択されていない場合は、一意のランダムなものを選択する。メディアがすでに送信されている場合は、新しいコーデックのクロックレートが異なる場合を除き、同じ SSRC を使用する必要がある[SHOULD]。その場合は、[RFC7160] の 4.1 節で指定されているように、新しい SSRC を選択する必要がある[MUST]。
- リモート記述からのペイロードタイプマッピングは、上記で選択した送信メディア形式のペイロードタイプを含む、送信 RTP ストリームのペイロードタイプを決定するために使用される。ネゴシエートされた RTP ヘッダー拡張は、リモート記述からの拡張マッピングを使用して、送信 RTP ストリームに含める必要がある。MID ヘッダー拡張がネゴシエートされた場合は、[RFC8843] の 15 章に示されているように、送信 RTP ストリームに含める。RtpStreamId または RepairedRtpStreamId ヘッダー拡張がネゴシエートされ、rid-id が確立されている場合は、[RFC8851] の 4 章に示されているように、送信 RTP ストリームにこれらのヘッダー拡張を含める。https://www.rfc-editor.org/rfc/rfc8851
- "m="セクションのタイプが"audio"で、無音抑制が (1) リモート記述を処理した結果として送信メディア形式に設定され、(2) ローカル記述でその形式に対しても有効になっている場合は、5.2.3.2 項のガイダンスに従って、送信メディアに無音抑制を使用する。これらの条件が満たされない場合は、送信メディアに無音抑制を使用してはならない[MUST NOT]。
- ◦simulcast がネゴシエートされた場合、[RFC8853] の 5.3.3 項で規定された適切な数の送信元 RTP ストリームを送信する。
- 上記で選択した送信メディア形式に対応する"rtx"メディア形式がある場合、または FEC メカニズムがネゴシエートされている場合は、各送信元 RTP ストリームに対して一意のランダムな SSRC を持つ冗長 RTP ストリームを確立し、必要に応じて RTX/FEC パケットの送信を開始または継続する。
- 上記で選択した送信メディア形式に対応する同じクロックレートの"red"メディア形式がある場合、[RFC8854] の 3.2 節で説明されているように、復元性の目的で指定された形式を使用した冗長エンコードを許可する。https://www.rfc-editor.org/rfc/rfc8854RTX または FEC メディア形式とは異なり、"red"形式は冗長 RTP ストリームではなく送信元 RTP ストリームで送信されることに注意する。
- 指定されたメディア形式を使用するすべての送信元 RTP ストリームについて、メディアセクションで参照されている RTCP フィードバックメカニズムを有効にする。具体的には、[RFC4585] の

4.2 節で指定されているように、要求されたフィードバックタイプの送信を開始または継続し、受信したフィードバックに対応する。https://www.rfc-editor.org/rfc/rfc4585RTCP フィードバックを送信する場合は、[RFC8108] の 5.4.1 項の規則と推奨事項に従って、使用する SSRC を選択する。

- アンサーの方向属性が、JSEP 実装がメディアを送信するべきではないことを示している場合(ローカルアンサーの場合は"recvonly"、リモートアンサーの場合は"sendonly"、いずれかの種類のアンサーの場合は"inactive")、[RFC3264]の 5.1 節に記載されているように、すべての RTP メディアの送信を停止するが、RTCP の送信は継続する。

- "m="セクション<proto>値が SCTP の使用を示している場合:
  - SCTP アソシエーションが存在し、リモート SCTP ポートが変更された場合、既存の SCTP アソシエーションを破棄する。これには、PeerConnection の状態が"have-remote-pranswer"の場合も含まれる。
  - 有効な SCTP アソシエーションが存在しない場合、[RFC8841] の 10.2 節に記載されているように、ローカル記述からのローカル SCTP ポート値とリモート記述からのリモート SCTP ポート値を使用して、関連付けられた ICE コンポーネントおよび DTLS 接続を介して SCTP アソシエーションを開始する準備をする。https://www.rfc-editor.org/rfc/rfc8841

アンサーに有効なバンドルグループが含まれている場合は、[RFC8843] の 7.4 節に記載されているように、各バンドルのプライマリ ICE コンポーネントにバンドルされる"m="セクションの ICE コンポーネントを破棄し、それに応じてこれらの"m="セクションの多重化を開始する。https://www.rfc-editor.org/rfc/rfc8843

記述のタイプが"answer"で、ICE 候補プールにまだ候補が残っている場合は、それらを破棄する。

6. RTP/RTCP の処理

バンドルする場合、受信 RTP/RTCP を適切な"m="セクションに関連付けることは、[RFC8843] の 9.2 節で定義されている。バンドルしない場合、適切な"m="セクションは、RTP/RTCP を受信する ICE コンポーネントから明確になる。

適切な"m="セクションがわかったら、RTP/RTCP は"m="セクションに関連付けられた RtpTransceiver に配信され、RTP/RTCP のさらなる処理は RtpTransceiver レベルで行われる。これには、RID メカニズム [RFC8851] とそれに関連付けられた RtpStreamId および RepairedRtpStreamId 識別子を使用して、複数のエンコードされたストリームを区別し、特定の冗長 RTP ストリームによって修復する Source RTP ストリームを決定することが含まれる。

7. 例

このセクションの例では、いくつかの SDP フラグメントを示している。RFC の行長制限に対応するために、SDP 行の一部は複数の行に分割されており、先頭の空白はその行が前の行の続きであることを示している。さらに、読みやすさを向上させるためにいくつかの空白行が追加されているが、SDP では無効である。

IPv6 アドレスの例を含む、WebRTC コールフロー用の SDP のより多くの例は、[SDP4WebRTC] にある。

7.1 簡単な例

このセクションでは、Trickle ICE を使用せずに、2 つの JSEP エンドポイント間の最小限の音声/ビデオコールを設定する非常に簡単な例を示す。次のセクションの例では、JSEP セッションで発生する可能性のある、より詳細な例を示す。

次のコードフローは、Bob のエンドポイントへのセッションを開始する Alice のエンドポイントを示している。Alice のブラウザの JavaScript アプリケーションから Bob のブラウザの JavaScript へのメッセージは、それぞれ"AliceJS"と"BobJS"と略され、ウェブサーバを介して何らかのシグナリングプロトコルを介して流れると想定される。Alice 側と Bob 側の JavaScript は、オファーまたはアンサーを送信する前にすべての候補

者を待機するため、オファーとアンサーは完全である。Trickle ICE は使用されない。Alice と Bob のブラウザのユーザエージェント (JSEP 実装) は、それぞれ"AliceUA"と"BobUA"と略称され、どちらも"balanced"のデフォルトバンドルポリシーと"require"のデフォルト RTCP mux ポリシーを使用している。

```
//                  set up local media state
AliceJS->AliceUA:   create new PeerConnection
AliceJS->AliceUA:   addTrack with two tracks: audio and video
AliceJS->AliceUA:   createOffer to get offer
AliceJS->AliceUA:   setLocalDescription with offer
AliceUA->AliceJS:   multiple onicecandidate events with candidates

//                  wait for ICE gathering to complete
AliceUA->AliceJS:   onicecandidate event with null candidate
AliceJS->AliceUA:   get |offer-A1| from pendingLocalDescription

//                  |offer-A1| is sent over signaling protocol to Bob
AliceJS->WebServer: signaling with |offer-A1|
WebServer->BobJS:   signaling with |offer-A1|

//                  |offer-A1| arrives at Bob
BobJS->BobUA:       create a PeerConnection
BobJS->BobUA:       setRemoteDescription with |offer-A1|
BobUA->BobJS:       ontrack events for audio and video tracks

//                  Bob accepts call
BobJS->BobUA:       addTrack with local tracks
BobJS->BobUA:       createAnswer
BobJS->BobUA:       setLocalDescription with answer
BobUA->BobJS:       multiple onicecandidate events with candidates

//                  wait for ICE gathering to complete
BobUA->BobJS:       onicecandidate event with null candidate
BobJS->BobUA:       get |answer-A1| from currentLocalDescription

//                  |answer-A1| is sent over signaling protocol
//                  to Alice
BobJS->WebServer:   signaling with |answer-A1|
WebServer->AliceJS: signaling with |answer-A1|

//                  |answer-A1| arrives at Alice
AliceJS->AliceUA:   setRemoteDescription with |answer-A1|
AliceUA->AliceJS:   ontrack events for audio and video tracks

//                  media flows
BobUA->AliceUA:     media sent from Bob to Alice
AliceUA->BobUA:     media sent from Alice to Bob
```

|offer-A1|の SDP は次のようになる。

```
v=0
o=- 4962303333179871722 1 IN IP4 0.0.0.0
s=-
t=0 0
a=ice-options:trickle ice2
a=group:BUNDLE a1 v1
a=group:LS a1 v1

m=audio 10100 UDP/TLS/RTP/SAVPF 96 0 8 97 98
c=IN IP4 203.0.113.100
a=mid:a1
a=sendrecv
a=rtpmap:96 opus/48000/2
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
```

```
a=rtpmap:97 telephone-event/8000
a=rtpmap:98 telephone-event/48000
a=fmtp:97 0-15
a=fmtp:98 0-15
a=maxptime:120
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:2 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=msid:47017fee-b6c1-4162-929c-a25110252400
a=ice-ufrag:ETEn
a=ice-pwd:OtSK0WpNtpUjkY4+86js7ZQl
a=fingerprint:sha-256
              19:E2:1C:3B:4B:9F:81:E6:B8:5C:F4:A5:A8:D8:73:04:
              BB:05:2F:70:9F:04:A9:0E:05:E9:26:33:E8:70:88:A2
a=setup:actpass
a=tls-id:91bbf309c0990a6bec11e38ba2933cee
a=rtcp:10101 IN IP4 203.0.113.100
a=rtcp-mux
a=rtcp-rsize
a=candidate:1 1 udp 2113929471 203.0.113.100 10100 typ host
a=candidate:1 2 udp 2113929470 203.0.113.100 10101 typ host
a=end-of-candidates

m=video 10102 UDP/TLS/RTP/SAVPF 100 101 102 103
c=IN IP4 203.0.113.100
a=mid:v1
a=sendrecv
a=rtpmap:100 VP8/90000
a=rtpmap:101 H264/90000
a=fmtp:101 packetization-mode=1;profile-level-id=42e01f
a=rtpmap:102 rtx/90000
a=fmtp:102 apt=100
a=rtpmap:103 rtx/90000
a=fmtp:103 apt=101
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:3 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=rtcp-fb:100 ccm fir
a=rtcp-fb:100 nack
a=rtcp-fb:100 nack pli
a=msid:47017fee-b6c1-4162-929c-a25110252400
a=ice-ufrag:BGKk
a=ice-pwd:mqyWsAjvtKwTGnvhPztQ9mIf
a=fingerprint:sha-256
              19:E2:1C:3B:4B:9F:81:E6:B8:5C:F4:A5:A8:D8:73:04:
              BB:05:2F:70:9F:04:A9:0E:05:E9:26:33:E8:70:88:A2
a=setup:actpass
a=tls-id:91bbf309c0990a6bec11e38ba2933cee
a=rtcp:10103 IN IP4 203.0.113.100
a=rtcp-mux
a=rtcp-rsize
a=candidate:1 1 udp 2113929471 203.0.113.100 10102 typ host
a=candidate:1 2 udp 2113929470 203.0.113.100 10103 typ host
a=end-of-candidates
```

|answer-A1|の SDP は次のようになる。

```
v=0
o=- 6729291447651054566 1 IN IP4 0.0.0.0
s=-
t=0 0
a=ice-options:trickle ice2
a=group:BUNDLE a1 v1
a=group:LS a1 v1

m=audio 10200 UDP/TLS/RTP/SAVPF 96 0 8 97 98
c=IN IP4 203.0.113.200
a=mid:a1
a=sendrecv
a=rtpmap:96 opus/48000/2
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 telephone-event/8000
a=rtpmap:98 telephone-event/48000
a=fmtp:97 0-15
a=fmtp:98 0-15
a=maxptime:120
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:2 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=msid:61317484-2ed4-49d7-9eb7-1414322a7aae
a=ice-ufrag:6sFv
a=ice-pwd:cOTZKZNVlO9RSGsEGM63JXT2
a=fingerprint:sha-256
              6B:8B:F0:65:5F:78:E2:51:3B:AC:6F:F3:3F:46:1B:35:
              DC:B8:5F:64:1A:24:C2:43:F0:A1:58:D0:A1:2C:19:08
a=setup:active
a=tls-id:eec3392ab83e11ceb6a0990c903fbb19
a=rtcp-mux a=rtcp-rsize
a=candidate:1 1 udp 2113929471 203.0.113.200 10200 typ host
a=end-of-candidates

m=video 10200 UDP/TLS/RTP/SAVPF 100 101 102 103
c=IN IP4 203.0.113.200
a=mid:v1
a=sendrecv
a=rtpmap:100 VP8/90000
a=rtpmap:101 H264/90000
a=fmtp:101 packetization-mode=1;profile-level-id=42e01f
a=rtpmap:102 rtx/90000 a=fmtp:102 apt=100
a=rtpmap:103 rtx/90000 a=fmtp:103 apt=101
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:3 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=rtcp-fb:100 ccm fir
a=rtcp-fb:100 nack
a=rtcp-fb:100 nack pli
a=msid:61317484-2ed4-49d7-9eb7-1414322a7aae
```

7.2　詳細な例

このセクションでは、2 つの JSEP エンドポイント間のセッションのより複雑な例を示す。Trickle ICE は完全なトリクルモードで使用され、バンドルポリシーは"max-bundle"、RTCP mux ポリシーは"require"、単一の TURN サーバで構成される。最初は、Alice と Bob の両方がオーディオチャネルとデータチャネルを確立する。その後、Bob は 2 つのビデオフロー (1 つはビデオフィード用、もう 1 つは画面共有用で、どちらも FEC をサポートしている) を追加し、ビデオフィードを同時放送用に設定する。Alice はこれらのビデオフローを受け入れるが、独自のビデオフローを追加しないため、recvonly として扱われる。Alice は、ビデオデコーダの最大解像度も指定する。

```
//                  set up local media state
AliceJS->AliceUA:   create new PeerConnection
AliceJS->AliceUA:   addTrack with an audio track
AliceJS->AliceUA:   createDataChannel to get data channel
AliceJS->AliceUA:   createOffer to get |offer-B1|
AliceJS->AliceUA:   setLocalDescription with |offer-B1|

//                  |offer-B1| is sent over signaling protocol to Bob
AliceJS->WebServer: signaling with |offer-B1|
WebServer->BobJS:   signaling with |offer-B1|

//                  |offer-B1| arrives at Bob
BobJS->BobUA:       create a PeerConnection
BobJS->BobUA:       setRemoteDescription with |offer-B1|
BobUA->BobJS:       ontrack event with audio track from Alice

//                  candidates are sent to Bob
AliceUA->AliceJS:   onicecandidate (host) |offer-B1-candidate-1|
AliceJS->WebServer: signaling with |offer-B1-candidate-1|
AliceUA->AliceJS:   onicecandidate (srflx) |offer-B1-candidate-2|
AliceJS->WebServer: signaling with |offer-B1-candidate-2|
AliceUA->AliceJS:   onicecandidate (relay) |offer-B1-candidate-3|
AliceJS->WebServer: signaling with |offer-B1-candidate-3|

WebServer->BobJS:   signaling with |offer-B1-candidate-1|
BobJS->BobUA:       addIceCandidate with |offer-B1-candidate-1|
WebServer->BobJS:   signaling with |offer-B1-candidate-2|
BobJS->BobUA:       addIceCandidate with |offer-B1-candidate-2|
WebServer->BobJS:   signaling with |offer-B1-candidate-3|
BobJS->BobUA:       addIceCandidate with |offer-B1-candidate-3|

//                  Bob accepts call
BobJS->BobUA:       addTrack with local audio
BobJS->BobUA:       createDataChannel to get data channel
BobJS->BobUA:       createAnswer to get |answer-B1|
BobJS->BobUA:       setLocalDescription with |answer-B1|

//                  |answer-B1| is sent to Alice
BobJS->WebServer:   signaling with |answer-B1|
WebServer->AliceJS: signaling with |answer-B1|
AliceJS->AliceUA:   setRemoteDescription with |answer-B1|
AliceUA->AliceJS:   ontrack event with audio track from Bob

//                  candidates are sent to Alice
BobUA->BobJS:       onicecandidate (host) |answer-B1-candidate-1|
BobJS->WebServer:   signaling with |answer-B1-candidate-1|
BobUA->BobJS:       onicecandidate (srflx) |answer-B1-candidate-2|
BobJS->WebServer:   signaling with |answer-B1-candidate-2|
BobUA->BobJS:       onicecandidate (relay) |answer-B1-candidate-3|
BobJS->WebServer:   signaling with |answer-B1-candidate-3|

WebServer->AliceJS: signaling with |answer-B1-candidate-1|
```

```
AliceJS->AliceUA:   addIceCandidate with |answer-B1-candidate-1|
WebServer->AliceJS: signaling with |answer-B1-candidate-2|
AliceJS->AliceUA:   addIceCandidate with |answer-B1-candidate-2|
WebServer->AliceJS: signaling with |answer-B1-candidate-3|
AliceJS->AliceUA:   addIceCandidate with |answer-B1-candidate-3|

//                  data channel opens
BobUA->BobJS:       ondatachannel event
AliceUA->AliceJS:   ondatachannel event
BobUA->BobJS:       onopen
AliceUA->AliceJS:   onopen

//                  media is flowing between endpoints
BobUA->AliceUA:     audio+data sent from Bob to Alice
AliceUA->BobUA:     audio+data sent from Alice to Bob

//                  some time later, Bob adds two video streams
//                  note: no candidates exchanged, because of bundle
BobJS->BobUA:       addTrack with first video stream
BobJS->BobUA:       addTrack with second video stream
BobJS->BobUA:       createOffer to get |offer-B2|
BobJS->BobUA:       setLocalDescription with |offer-B2|

//                  |offer-B2| is sent to Alice
BobJS->WebServer:   signaling with |offer-B2|
WebServer->AliceJS: signaling with |offer-B2|
AliceJS->AliceUA:   setRemoteDescription with |offer-B2|
AliceUA->AliceJS:   ontrack event with first video track
AliceUA->AliceJS:   ontrack event with second video track
AliceJS->AliceUA:   createAnswer to get |answer-B2|
AliceJS->AliceUA:   setLocalDescription with |answer-B2|

//                  |answer-B2| is sent over signaling protocol
//                  to Bob
AliceJS->WebServer: signaling with |answer-B2|
WebServer->BobJS:   signaling with |answer-B2|
BobJS->BobUA:       setRemoteDescription with |answer-B2|

//                  media is flowing between endpoints
BobUA->AliceUA:     audio+video+data sent from Bob to Alice
AliceUA->BobUA:     audio+video+data sent from Alice to Bob
```

|offer-B1|の SDP は次のようになる。

```
v=0
o=- 4962303333179871723 1 IN IP4 0.0.0.0
s=-
t=0 0
a=ice-options:trickle ice2
a=group:BUNDLE a1 d1

m=audio 9 UDP/TLS/RTP/SAVPF 96 0 8 97 98
c=IN IP4 0.0.0.0
a=mid:a1 a=sendrecv
a=rtpmap:96 opus/48000/2
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 telephone-event/8000
a=rtpmap:98 telephone-event/48000
a=fmtp:97 0-15
a=fmtp:98 0-15
a=maxptime:120
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:2 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=msid:57017fee-b6c1-4162-929c-a25110252400
a=ice-ufrag:ATEn
a=ice-pwd:AtSK0WpNtpUjkY4+86js7ZQl
a=fingerprint:sha-256
              29:E2:1C:3B:4B:9F:81:E6:B8:5C:F4:A5:A8:D8:73:04:
              BB:05:2F:70:9F:04:A9:0E:05:E9:26:33:E8:70:88:A2
a=setup:actpass
a=tls-id:17f0f4ba8a5f1213faca591b58ba52a7
a=rtcp-mux a=rtcp-mux-only
a=rtcp-rsize

m=application 0 UDP/DTLS/SCTP webrtc-datachannel
c=IN IP4 0.0.0.0
a=mid:d1
a=sctp-port:5000
a=max-message-size:65536
a=bundle-only
```

|offer-B1-candidate-1|は次のようになる。

```
ufrag ATEn
index 0
mid   a1
attr  candidate:1 1 udp 2113929471 203.0.113.100 10100 typ host
```

|offer-B1-candidate-2|は次のようになる。

```
ufrag ATEn
index 0
mid   a1
attr  candidate:1 1 udp 1845494015 198.51.100.100 11100 typ srflx
                raddr 203.0.113.100 rport 10100
```

|offer-B1-candidate-3|は次のようになる。

```
ufrag ATEn
index 0
mid   a1
attr  candidate:1 1 udp 255 192.0.2.100 12100 typ relay
                 raddr 198.51.100.100 rport 11100
```

|answer-B1|の SDP は次のようになる。

```
v=0
o=- 7729291447651054566 1 IN IP4 0.0.0.0
s=-
t=0 0
a=ice-options:trickle ice2
a=group:BUNDLE a1 d1

m=audio 9 UDP/TLS/RTP/SAVPF 96 0 8 97 98
c=IN IP4 0.0.0.0
a=mid:a1
a=sendrecv
a=rtpmap:96 opus/48000/2
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 telephone-event/8000
a=rtpmap:98 telephone-event/48000
a=fmtp:97 0-15
a=fmtp:98 0-15
a=maxptime:120
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:2 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=msid:71317484-2ed4-49d7-9eb7-1414322a7aae
a=ice-ufrag:7sFv
a=ice-pwd:dOTZKZNVlO9RSGsEGM63JXT2
a=fingerprint:sha-256
              7B:8B:F0:65:5F:78:E2:51:3B:AC:6F:F3:3F:46:1B:35:
              DC:B8:5F:64:1A:24:C2:43:F0:A1:58:D0:A1:2C:19:08
a=setup:active
a=tls-id:7a25ab85b195acaf3121f5a8ab4f0f71
a=rtcp-mux
a=rtcp-mux-only
a=rtcp-rsize

m=application 9 UDP/DTLS/SCTP webrtc-datachannel
c=IN IP4 0.0.0.0
a=mid:d1
a=sctp-port:5000
a=max-message-size:65536
```

|answer-B1-candidate-1|は次のようになる。

```
ufrag 7sFv
index 0
mid   a1
attr  candidate:1 1 udp 2113929471 203.0.113.200 10200 typ host
```

|answer-B1-candidate-2|は次のようになる。

```
ufrag 7sFv
index 0
mid   a1
attr  candidate:1 1 udp 1845494015 198.51.100.200 11200 typ srflx
                raddr 203.0.113.200 rport 10200
```

|answer-B1-candidate-3|は次のようになる。

```
ufrag 7sFv
index 0
mid   a1
attr  candidate:1 1 udp 255 192.0.2.200 12200 typ relay
                 raddr 198.51.100.200 rport 11200
```

|offer-B2|の SDP を次に示す。ビデオの新しい"m="セクション (両方が FEC を提供し、一方が同時放送を提供している) に加えて、"o="行のバージョン番号のインクリメントに注意すること。選択されたローカル候補を示す"c="行への変更;と、収集された候補を a=候補行として含めること。

```
v=0
o=- 7729291447651054566 2 IN IP4 0.0.0.0
s=-
t=0 0
a=ice-options:trickle ice2 a=group:BUNDLE a1 d1 v1 v2
a=group:LS a1 v1

m=audio 12200 UDP/TLS/RTP/SAVPF 96 0 8 97 98
c=IN IP4 192.0.2.200
a=mid:a1 a=sendrecv
a=rtpmap:96 opus/48000/2
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 telephone-event/8000
a=rtpmap:98 telephone-event/48000
a=fmtp:97 0-15
a=fmtp:98 0-15 a=maxptime:120
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:2 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=msid:71317484-2ed4-49d7-9eb7-1414322a7aae
a=ice-ufrag:7sFv
a=ice-pwd:dOTZKZNVlO9RSGsEGM63JXT2
a=fingerprint:sha-256
              7B:8B:F0:65:5F:78:E2:51:3B:AC:6F:F3:3F:46:1B:35:
              DC:B8:5F:64:1A:24:C2:43:F0:A1:58:D0:A1:2C:19:08
a=setup:actpass
a=tls-id:7a25ab85b195acaf3121f5a8ab4f0f71
a=rtcp-mux
a=rtcp-mux-only
a=rtcp-rsize
a=candidate:1 1 udp 2113929471 203.0.113.200 10200 typ host
a=candidate:1 1 udp 1845494015 198.51.100.200 11200 typ srflx
            raddr 203.0.113.200 rport 10200
a=candidate:1 1 udp 255 192.0.2.200 12200 typ relay
             raddr 198.51.100.200 rport 11200
a=end-of-candidates

m=application 12200 UDP/DTLS/SCTP webrtc-datachannel
c=IN IP4 192.0.2.200
a=mid:d1
a=sctp-port:5000
a=max-message-size:65536

m=video 12200 UDP/TLS/RTP/SAVPF 100 101 102 103 104
c=IN IP4 192.0.2.200
a=mid:v1
a=sendrecv
a=rtpmap:100 VP8/90000
a=rtpmap:101 H264/90000
a=fmtp:101 packetization-mode=1;profile-level-id=42e01f
a=rtpmap:102 rtx/90000
a=fmtp:102 apt=100
a=rtpmap:103 rtx/90000
a=fmtp:103 apt=101
a=rtpmap:104 flexfec/90000
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:3 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=rtcp-fb:100 ccm fir
a=rtcp-fb:100 nack
a=rtcp-fb:100 nack pli
```

```
a=msid:71317484-2ed4-49d7-9eb7-1414322a7aae
a=rid:1 send
a=rid:2 send
a=rid:3 send
a=simulcast:send 1;2;3

m=video 12200 UDP/TLS/RTP/SAVPF 100 101 102 103 104
c=IN IP4 192.0.2.200
a=mid:v2
a=sendrecv
a=rtpmap:100 VP8/90000
a=rtpmap:101 H264/90000
a=fmtp:101 packetization-mode=1;profile-level-id=42e01f
a=rtpmap:102 rtx/90000
a=fmtp:102 apt=100
a=rtpmap:103 rtx/90000
a=fmtp:103 apt=101
a=rtpmap:104 flexfec/90000
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:3 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=rtcp-fb:100 ccm fir
a=rtcp-fb:100 nack
a=rtcp-fb:100 nack pli
a=msid:81317484-2ed4-49d7-9eb7-1414322a7aae
```

|answer-B2|の SDP を次に示す。ビデオ"m="セクションの受け入れ、一方向ビデオを示す a=recvonly の使用、受信解像度を制限する a=imageattr の使用に加えて、既存の DTLS ロールを維持するための setup:passive の使用に注意すること。

```
v=0
o=- 4962303333179871723 2 IN IP4 0.0.0.0
s=-
t=0 0
a=ice-options:trickle ice2
a=group:BUNDLE a1 d1 v1 v2
a=group:LS a1 v1

m=audio 12100 UDP/TLS/RTP/SAVPF 96 0 8 97 98
c=IN IP4 192.0.2.100
a=mid:a1
a=sendrecv
a=rtpmap:96 opus/48000/2
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 telephone-event/8000
a=rtpmap:98 telephone-event/48000
a=fmtp:97 0-15
a=fmtp:98 0-15
a=maxptime:120
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:2 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=msid:57017fee-b6c1-4162-929c-a25110252400
a=ice-ufrag:ATEn
a=ice-pwd:AtSK0WpNtpUjkY4+86js7ZQl
a=fingerprint:sha-256
              29:E2:1C:3B:4B:9F:81:E6:B8:5C:F4:A5:A8:D8:73:04:
              BB:05:2F:70:9F:04:A9:0E:05:E9:26:33:E8:70:88:A2
a=setup:passive
a=tls-id:17f0f4ba8a5f1213faca591b58ba52a7
a=rtcp-mux
a=rtcp-mux-only
a=rtcp-rsize
a=candidate:1 1 udp 2113929471 203.0.113.100 10100 typ host
a=candidate:1 1 udp 1845494015 198.51.100.100 11100 typ srflx
```

```
            raddr 203.0.113.100 rport 10100
a=candidate:1 1 udp 255 192.0.2.100 12100 typ relay
             raddr 198.51.100.100 rport 11100
a=end-of-candidates

m=application 12100 UDP/DTLS/SCTP webrtc-datachannel
c=IN IP4 192.0.2.100
a=mid:d1
a=sctp-port:5000
a=max-message-size:65536

m=video 12100 UDP/TLS/RTP/SAVPF 100 101 102 103
c=IN IP4 192.0.2.100
a=mid:v1
a=recvonly
a=rtpmap:100 VP8/90000
a=rtpmap:101 H264/90000
a=fmtp:101 packetization-mode=1;profile-level-id=42e01f
a=rtpmap:102 rtx/90000
a=fmtp:102 apt=100
a=rtpmap:103 rtx/90000
a=fmtp:103 apt=101
a=imageattr:100 recv [x=[48:1920],y=[48:1080],q=1.0]
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:3 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=rtcp-fb:100 ccm fir
a=rtcp-fb:100 nack
a=rtcp-fb:100 nack pli

m=video 12100 UDP/TLS/RTP/SAVPF 100 101 102 103
c=IN IP4 192.0.2.100
a=mid:v2
a=recvonly
a=rtpmap:100 VP8/90000
a=rtpmap:101 H264/90000
a=fmtp:101 packetization-mode=1;profile-level-id=42e01f
a=rtpmap:102 rtx/90000
a=fmtp:102 apt=100
a=rtpmap:103 rtx/90000
a=fmtp:103 apt=101
a=imageattr:100 recv [x=[48:1920],y=[48:1080],q=1.0]
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:3 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=rtcp-fb:100 ccm fir
a=rtcp-fb:100 nack
a=rtcp-fb:100 nack pli
```

7.3 早期トランスポートウォームアップの例

この例は、4.1.10.1 項で説明されている早期ウォームアップ手法を示している。ここで、Alice のエンドポイントは、音声/ビデオ通話を開始するためのオファーを Bob のエンドポイントに送信する。Bob はすぐに応答し、音声/ビデオの"m="セクションを受け入れるが、(彼の視点では) 送信専用としてマークする。これは、Alice がまだメディアを送信しないことを意味する。これにより、JSEP 実装は ICE と DTLS のネゴシエートをすぐに開始できる。次に、Bob のエンドポイントは彼にコールに応答するように促し、応答すると、彼のエンドポイントは 2 番目のオファーを送信する。これにより、オーディオとビデオの"m="セクションが有効になり、双方向メディア伝送が可能になる。このようなフローの利点は、最初の応答を受信するとすぐに実装が ICE と DTLS のネゴシエーションを続行し、セッショントランスポートを確立できることである。2 番目のオファーが送信される前にトランスポートのセットアップが完了した場合、コールに応答した直後にメディアを呼び出し先から送信できるため、認識されるダイヤル後の遅延を最小限に抑えることができる。2 番目のオファー/アンサーエクスチェンジでは、優先コーデックやその他のセッションパラメータを変更する

こともできる。

この例では、3.5.3 項で説明されている「リレー」ICE 候補ポリシーも利用して、必要な ICE の収集とチェックを最小限に抑えている。

```
//                  set up local media state
AliceJS->AliceUA:   create new PeerConnection with "relay" ICE policy
AliceJS->AliceUA:   addTrack with two tracks: audio and video
AliceJS->AliceUA:   createOffer to get |offer-C1|
AliceJS->AliceUA:   setLocalDescription with |offer-C1|

//                  |offer-C1| is sent over signaling protocol to Bob
AliceJS->WebServer: signaling with |offer-C1|
WebServer->BobJS:   signaling with |offer-C1|

//                  |offer-C1| arrives at Bob
BobJS->BobUA:       create new PeerConnection with "relay" ICE policy
BobJS->BobUA:       setRemoteDescription with |offer-C1|
BobUA->BobJS:       ontrack events for audio and video

//                  a relay candidate is sent to Bob
AliceUA->AliceJS:   onicecandidate (relay) |offer-C1-candidate-1|
AliceJS->WebServer: signaling with |offer-C1-candidate-1|

WebServer->BobJS:   signaling with |offer-C1-candidate-1|
BobJS->BobUA:       addIceCandidate with |offer-C1-candidate-1|

//                  Bob prepares an early answer to warm up the
//                  transport
BobJS->BobUA:       addTransceiver with null audio and video tracks
BobJS->BobUA:       transceiver.setDirection(sendonly) for both
BobJS->BobUA:       createAnswer
BobJS->BobUA:       setLocalDescription with answer

//                  |answer-C1| is sent over signaling protocol
//                  to Alice
BobJS->WebServer:   signaling with |answer-C1|
WebServer->AliceJS: signaling with |answer-C1|

//                  |answer-C1| (sendonly) arrives at Alice
AliceJS->AliceUA:   setRemoteDescription with |answer-C1|
AliceUA->AliceJS:   ontrack events for audio and video

//                  a relay candidate is sent to Alice
BobUA->BobJS:       onicecandidate (relay) |answer-B1-candidate-1|
BobJS->WebServer:   signaling with |answer-B1-candidate-1|
WebServer->AliceJS: signaling with |answer-B1-candidate-1|
AliceJS->AliceUA:   addIceCandidate with |answer-B1-candidate-1|

//                  ICE and DTLS establish while call is ringing

//                  Bob accepts call, starts media, and sends
//                  new offer
BobJS->BobUA:       transceiver.setTrack with audio and video tracks
BobUA->AliceUA:     media sent from Bob to Alice
BobJS->BobUA:       transceiver.setDirection(sendrecv) for both
                    transceivers BobJS->BobUA:       createOffer
BobJS->BobUA:       setLocalDescription with offer

//                  |offer-C2| is sent over signaling protocol
//                  to Alice
BobJS->WebServer:   signaling with |offer-C2|
WebServer->AliceJS: signaling with |offer-C2|

//                  |offer-C2| (sendrecv) arrives at Alice
```

```
AliceJS->AliceUA:   setRemoteDescription with |offer-C2|
AliceJS->AliceUA:   createAnswer
AliceJS->AliceUA:   setLocalDescription with |answer-C2|
AliceUA->BobUA:     media sent from Alice to Bob

//                  |answer-C2| is sent over signaling protocol
//                  to Bob
AliceJS->WebServer: signaling with |answer-C2|
WebServer->BobJS:   signaling with |answer-C2|
BobJS->BobUA:       setRemoteDescription with |answer-C2|
```

|offer-C1|の SDP は次のようになる。

```
v=0
o=- 1070771854436052752 1 IN IP4 0.0.0.0
s=-
t=0 0
a=ice-options:trickle ice2
a=group:BUNDLE a1 v1
a=group:LS a1 v1

m=audio 9 UDP/TLS/RTP/SAVPF 96 0 8 97 98
c=IN IP4 0.0.0.0
a=mid:a1
a=sendrecv
a=rtpmap:96 opus/48000/2
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 telephone-event/8000
a=rtpmap:98 telephone-event/48000
a=fmtp:97 0-15
a=fmtp:98 0-15
a=maxptime:120
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:2 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=msid:bbce3ba6-abfc-ac63-d00a-e15b286f8fce
a=ice-ufrag:4ZcD
a=ice-pwd:ZaaG6OG7tCn4J/lehAGz+HHD
a=fingerprint:sha-256
              C4:68:F8:77:6A:44:F1:98:6D:7C:9F:47:EB:E3:34:A4:
              0A:AA:2D:49:08:28:70:2E:1F:AE:18:7D:4E:3E:66:BF
a=setup:actpass
a=tls-id:9e5b948ade9c3d41de6617b68f769e55
a=rtcp-mux
a=rtcp-mux-only
a=rtcp-rsize

m=video 0 UDP/TLS/RTP/SAVPF 100 101 102 103
c=IN IP4 0.0.0.0
a=mid:v1
a=sendrecv
a=rtpmap:100 VP8/90000
a=rtpmap:101 H264/90000
a=fmtp:101 packetization-mode=1;profile-level-id=42e01f
a=rtpmap:102 rtx/90000
a=fmtp:102 apt=100
a=rtpmap:103 rtx/90000
a=fmtp:103 apt=101
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:3 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=rtcp-fb:100 ccm fir
a=rtcp-fb:100 nack
a=rtcp-fb:100 nack pli
a=msid:bbce3ba6-abfc-ac63-d00a-e15b286f8fce
a=bundle-only
```

|offer-C1-candidate-1|は次のようになる。

```
ufrag 4ZcD
index 0
mid   a1
attr  candidate:1 1 udp 255 192.0.2.100 12100 typ relay
                raddr 0.0.0.0 rport 0
```

|answer-C1|の SDP は次のようになる。

```
v=0
o=- 6386516489780559513 1 IN IP4 0.0.0.0
s=-
t=0 0
a=ice-options:trickle ice2
a=group:BUNDLE a1 v1
a=group:LS a1 v1

m=audio 9 UDP/TLS/RTP/SAVPF 96 0 8 97 98
c=IN IP4 0.0.0.0
a=mid:a1
a=sendonly
a=rtpmap:96 opus/48000/2
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 telephone-event/8000
a=rtpmap:98 telephone-event/48000
a=fmtp:97 0-15
a=fmtp:98 0-15
a=maxptime:120
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:2 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=msid:751f239e-4ae0-c549-aa3d-890de772998b
a=ice-ufrag:TpaA
a=ice-pwd:t2Ouhc67y8JcCaYZxUUTgKw/
a=fingerprint:sha-256
              A2:F3:A5:6D:4C:8C:1E:B2:62:10:4A:F6:70:61:C4:FC:
              3C:E0:01:D6:F3:24:80:74:DA:7C:3E:50:18:7B:CE:4D
a=setup:active
a=tls-id:55e967f86b7166ed14d3c9eda849b5e9
a=rtcp-mux
a=rtcp-mux-only
a=rtcp-rsize

m=video 9 UDP/TLS/RTP/SAVPF 100 101 102 103
c=IN IP4 0.0.0.0
a=mid:v1 a=sendonly
a=rtpmap:100 VP8/90000
a=rtpmap:101 H264/90000
a=fmtp:101 packetization-mode=1;profile-level-id=42e01f
a=rtpmap:102 rtx/90000
a=fmtp:102 apt=100
a=rtpmap:103 rtx/90000
a=fmtp:103 apt=101
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:3 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=rtcp-fb:100 ccm fir
a=rtcp-fb:100 nack
a=rtcp-fb:100 nack pli
a=msid:751f239e-4ae0-c549-aa3d-890de772998b
```

|answer-C1-candidate-1|は次のようになる。

```
ufrag TpaA index 0 mid   a1
attr  candidate:1 1 udp 255 192.0.2.200 12200 typ relay
                raddr 0.0.0.0 rport 0
```

|offer-C2|の SDP は次のようになる。

```
v=0
o=- 6386516489780559513 2 IN IP4 0.0.0.0
s=-
t=0 0
a=ice-options:trickle ice2
a=group:BUNDLE a1 v1
a=group:LS a1 v1

m=audio 12200 UDP/TLS/RTP/SAVPF 96 0 8 97 98
c=IN IP4 192.0.2.200
a=mid:a1
a=sendrecv
a=rtpmap:96 opus/48000/2
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 telephone-event/8000
a=rtpmap:98 telephone-event/48000
a=fmtp:97 0-15 a=fmtp:98 0-15
a=maxptime:120
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:2 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=msid:751f239e-4ae0-c549-aa3d-890de772998b
a=ice-ufrag:TpaA
a=ice-pwd:t2Ouhc67y8JcCaYZxUUTgKw/
a=fingerprint:sha-256
              A2:F3:A5:6D:4C:8C:1E:B2:62:10:4A:F6:70:61:C4:FC:
              3C:E0:01:D6:F3:24:80:74:DA:7C:3E:50:18:7B:CE:4D
a=setup:actpass
a=tls-id:55e967f86b7166ed14d3c9eda849b5e9
a=rtcp-mux
a=rtcp-mux-only
a=rtcp-rsize
a=candidate:1 1 udp 255 192.0.2.200 12200 typ relay
            raddr 0.0.0.0 rport 0
a=end-of-candidates

m=video 12200 UDP/TLS/RTP/SAVPF 100 101 102 103
c=IN IP4 192.0.2.200
a=mid:v1
a=sendrecv
a=rtpmap:100 VP8/90000
a=rtpmap:101 H264/90000
a=fmtp:101 packetization-mode=1;profile-level-id=42e01f
a=rtpmap:102 rtx/90000
a=fmtp:102 apt=100 a=rtpmap:103 rtx/90000
a=fmtp:103 apt=101
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:3 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=rtcp-fb:100 ccm fir
a=rtcp-fb:100 nack
a=rtcp-fb:100 nack pli
a=msid:751f239e-4ae0-c549-aa3d-890de772998b
```

|answer-C2|の SDP は次のようになる。

```
v=0
o=- 1070771854436052752 2 IN IP4 0.0.0.0
s=t=0 0
a=ice-options:trickle ice2
a=group:BUNDLE a1 v1
a=group:LS a1 v1

m=audio 12100 UDP/TLS/RTP/SAVPF 96 0 8 97 98
c=IN IP4 192.0.2.100
a=mid:a1
a=sendrecv
a=rtpmap:96 opus/48000/2
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 telephone-event/8000
a=rtpmap:98 telephone-event/48000
a=fmtp:97 0-15
a=fmtp:98 0-15
a=maxptime:120
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:2 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=msid:bbce3ba6-abfc-ac63-d00a-e15b286f8fce
a=ice-ufrag:4ZcD
a=ice-pwd:ZaaG6OG7tCn4J/lehAGz+HHD
a=fingerprint:sha-256
              C4:68:F8:77:6A:44:F1:98:6D:7C:9F:47:EB:E3:34:A4:
              0A:AA:2D:49:08:28:70:2E:1F:AE:18:7D:4E:3E:66:BF
a=setup:passive
a=tls-id:9e5b948ade9c3d41de6617b68f769e55
a=rtcp-mux
a=rtcp-mux-only a=rtcp-rsize
a=candidate:1 1 udp 255 192.0.2.100 12100 typ relay
            raddr 0.0.0.0 rport 0
a=end-of-candidates

m=video 12100 UDP/TLS/RTP/SAVPF 100 101 102 103
c=IN IP4 192.0.2.100
a=mid:v1
a=sendrecv
a=rtpmap:100 VP8/90000
a=rtpmap:101 H264/90000
a=fmtp:101 packetization-mode=1;profile-level-id=42e01f
a=rtpmap:102 rtx/90000
a=fmtp:102 apt=100 a=rtpmap:103 rtx/90000
a=fmtp:103 apt=101
a=extmap:1 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:3 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=rtcp-fb:100 ccm fir
a=rtcp-fb:100 nack
a=rtcp-fb:100 nack pli
a=msid:bbce3ba6-abfc-ac63-d00a-e15b286f8fce
```

8. セキュリティに関する考慮事項

IETF は、WebRTC 全体のセキュリティアーキテクチャを記述した個別のドキュメント [RFC8827] [RFC8826] を公開している。このセクションの残りの部分では、このドキュメントのセキュリティに関する考慮事項について説明する。

正式には、JSEP インタフェースは API だが、JSEP 実装の観点からはアプリケーション JavaScript が信頼できないため、インターネットプロトコルと考えた方がよい。したがって、[RFC3552] の脅威モデルが適用される。特に、JavaScript は、任意の順序で、悪意のあるものを含む任意の入力で API を呼び出すことができる。これは、setLocalDescription に渡される SDP を考慮する場合に特に関連する。API を正しく使用するに

は、createOffer または createAnswer から派生した SDP でアプリケーションが合格する必要があるが、アプリケーションが合格する保証はない。JSEP 実装は、JavaScript が代わりに偽のデータを渡すために準備する必要がある。

逆に、アプリケーションプログラマは、JavaScript がエンドポイントの動作を完全に制御していないことを認識する必要がある。特に言及されているケースの 1 つは、SDP から ICE 候補を編集したり、トリクルされた候補を抑制したりすることは、期待された動作をしないということである。実装は、相手側に送信されていない場合でも、これらの候補からのチェックを実行する。したがって、たとえば、サーバ再帰候補を削除することによって、リモートピアがパブリック IP アドレスを学習するのを防ぐことはできない。パブリック IP アドレスを隠したいアプリケーションは、代わりにリレー候補のみを使用するように ICE エージェントを設定する必要がある。

9. IANA に関する考慮事項

このドキュメントには IANA アクションはない。

10. 参考資料

10.1 標準参照

[RFC2119] Bradner, S., "Key words for use in RFCs to IndicateRequirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <https://www.rfc-editor.org/info/rfc3261>.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <https://www.rfc-editor.org/info/rfc3264>.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <https://www.rfc-editor.org/info/rfc3552>.

[RFC3605] Huitema, C., "Real Time Control Protocol (RTCP) attribute in Session Description Protocol (SDP)", RFC 3605, DOI 10.17487/RFC3605, October 2003, <https://www.rfc-editor.org/info/rfc3605>.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <https://www.rfc-editor.org/info/rfc3711>.

[RFC3890] Westerlund, M., "A Transport Independent Bandwidth Modifier for the Session Description Protocol (SDP)", RFC 3890, DOI 10.17487/RFC3890, September 2004, <https://www.rfc-editor.org/info/rfc3890>.

[RFC4145] Yon, D. and G. Camarillo, "TCP-Based Media Transport in the Session Description Protocol (SDP)", RFC 4145, DOI 10.17487/RFC4145, September 2005, <https://www.rfc-editor.org/info/rfc4145>.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, DOI 10.17487/RFC4566, July 2006, <https://www.rfc-editor.org/info/rfc4566>.

[RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, DOI 10.17487/RFC4585, July 2006, <https://www.rfc-editor.org/info/rfc4585>.

[RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, DOI 10.17487/RFC5124, February 2008, <https://www.rfc-editor.org/info/rfc5124>.

[RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, DOI 10.17487/RFC5285, July 2008, <https://www.rfc-editor.org/info/rfc5285>.

[RFC5761] Perkins, C. and M. Westerlund, "Multiplexing RTP Data and Control Packets on a Single Port", RFC 5761, DOI 10.17487/RFC5761, April 2010, <https://www.rfc-editor.org/info/rfc5761>.

[RFC5888] Camarillo, G. and H. Schulzrinne, "The Session Description Protocol (SDP) Grouping Framework", RFC 5888, DOI 10.17487/RFC5888, June 2010, <https://www.rfc-editor.org/info/rfc5888>.

[RFC6236] Johansson, I. and K. Jung, "Negotiation of Generic Image Attributes in the Session Description Protocol (SDP)", RFC 6236, DOI 10.17487/RFC6236, May 2011, <https://www.rfc-editor.org/info/rfc6236>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[RFC6716] Valin, JM., Vos, K., and T. Terriberry, "Definition of the Opus Audio Codec", RFC 6716, DOI 10.17487/RFC6716, September 2012, <https://www.rfc-editor.org/info/rfc6716>.

[RFC6904] Lennox, J., "Encryption of Header Extensions in the Secure Real-time Transport Protocol (SRTP)", RFC 6904, DOI 10.17487/RFC6904, April 2013, <https://www.rfc-editor.org/info/rfc6904>.

[RFC7160] Petit-Huguenin, M. and G. Zorn, Ed., "Support for Multiple Clock Rates in an RTP Session", RFC 7160, DOI 10.17487/RFC7160, April 2014, <https://www.rfc-editor.org/info/rfc7160>.

[RFC7587] Spittka, J., Vos, K., and JM. Valin, "RTP Payload Format for the Opus Speech and Audio Codec", RFC 7587, DOI 10.17487/RFC7587, June 2015, <https://www.rfc-editor.org/info/rfc7587>.

[RFC7742] Roach, A.B., "WebRTC Video Processing and Codec Requirements", RFC 7742, DOI 10.17487/RFC7742, March 2016, <https://www.rfc-editor.org/info/rfc7742>.

[RFC7850] Nandakumar, S., "Registering Values of the SDP 'proto' Field for Transporting RTP Media over TCP under Various RTP Profiles", RFC 7850, DOI 10.17487/RFC7850, April 2016, <https://www.rfc-editor.org/info/rfc7850>.

[RFC7874] Valin, JM. and C. Bran, "WebRTC Audio Codec and Processing Requirements", RFC 7874, DOI 10.17487/RFC7874, May 2016, <https://www.rfc-editor.org/info/rfc7874>.

[RFC8108] Lennox, J., Westerlund, M., Wu, Q., and C. Perkins, "Sending Multiple RTP Streams in a Single RTP Session", RFC 8108, DOI 10.17487/RFC8108, March 2017, <https://www.rfc-editor.org/info/rfc8108>.

[RFC8122] Lennox, J. and C. Holmberg, "Connection-Oriented Media Transport over the Transport Layer Security (TLS) Protocol in the Session Description Protocol (SDP)", RFC 8122, DOI 10.17487/RFC8122, March 2017, <https://www.rfc-editor.org/info/rfc8122>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8445] Keranen, A., Holmberg, C., and J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal", RFC 8445, DOI 10.17487/RFC8445, July 2018,

<https://www.rfc-editor.org/info/rfc8445>.

[RFC8826] Rescorla, E., "Security Considerations for WebRTC", RFC 8826, DOI 10.17487/RFC8826, January 2021, <https://www.rfc-editor.org/info/rfc8826>.

[RFC8827] Rescorla, E., "WebRTC Security Architecture", RFC 8827, DOI 10.17487/RFC8827, January 2021, <https://www.rfc-editor.org/info/rfc8827>.

[RFC8830] Alvestrand, H., "WebRTC MediaStream Identification in the Session Description Protocol", RFC 8830, DOI 10.17487/RFC8830, January 2021, <https://www.rfc-editor.org/info/rfc8830>.

[RFC8834] Perkins, C., Westerlund, M., and J. Ott, "Media Transport and Use of RTP in WebRTC", RFC 8834, DOI 10.17487/RFC8834, January 2021, <https://www.rfc-editor.org/info/rfc8834>.

[RFC8838] Ivov, E., Uberti, J., and P. Saint-Andre, "Trickle ICE: Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (ICE) Protocol", RFC 8838, DOI 10.17487/RFC8838, January 2021, <https://www.rfc-editor.org/info/rfc8838>.

[RFC8839] Petit-Huguenin, M., Nandakumar, S., Holmberg, C., Keränen, A., and R. Shpount, "Session Description Protocol (SDP) Offer/Answer Procedures for Interactive Connectivity Establishment (ICE)", RFC 8839, DOI 10.17487/RFC8839, January 2021, <https://www.rfc-editor.org/info/rfc8839>.

[RFC8840] Ivov, E., Stach, T., Marocco, E., and C. Holmberg, "A Session Initiation Protocol (SIP) Usage for Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (Trickle ICE)", RFC 8840, DOI 10.17487/RFC8840, January 2021, <https://www.rfc-editor.org/info/rfc8840>.

[RFC8841] Holmberg, C., Shpount, R., Loreto, S., and G. Camarillo, "Session Description Protocol (SDP) Offer/Answer Procedures for Stream Control Transmission Protocol (SCTP) over Datagram Transport Layer Security (DTLS) Transport", RFC 8841, DOI 10.17487/RFC8841, January 2021, <https://www.rfc-editor.org/info/rfc8841>.

[RFC8842] Holmberg, C. and R. Shpount, "Session Description Protocol (SDP) Offer/Answer Considerations for Datagram Transport Layer Security (DTLS) and Transport Layer Security (TLS)", RFC 8842, DOI 10.17487/RFC8842, January 2021, <https://www.rfc-editor.org/info/rfc8842>.

[RFC8843] Holmberg, C., Alvestrand, H., and C. Jennings, "Negotiating Media Multiplexing Using the Session Description Protocol (SDP)", RFC 8843, DOI 10.17487/RFC8843, January 2021, <https://www.rfc-editor.org/info/rfc8843>.

[RFC8851] Roach, A.B., Ed., "RTP Payload Format Restrictions", RFC 8851, DOI 10.17487/RFC8851, January 2021, <https://www.rfc-editor.org/info/rfc8851>.

[RFC8852] Roach, A.B., Nandakumar, S., and P. Thatcher, "RTP Stream Identifier Source Description (SDES)", RFC 8852, DOI 10.17487/RFC8852, January 2021, <https://www.rfc-editor.org/info/rfc8852>.

[RFC8853] Burman, B., Westerlund, M., Nandakumar, S., and M. Zanaty, "Using Simulcast in Session Description Protocol (SDP) and RTP Sessions", RFC 8853, DOI 10.17487/RFC8853, January 2021, <https://www.rfc-editor.org/info/rfc8853>.

[RFC8854] Uberti, J., "WebRTC Forward Error Correction Requirements", RFC 8854, DOI 10.17487/RFC8854, January 2021, <https://www.rfc-editor.org/info/rfc8854>.

[RFC8858] Holmberg, C., "Indicating Exclusive Support of RTP and RTP Control Protocol (RTCP) Multiplexing

Using the Session Description Protocol (SDP)", RFC 8858, DOI 10.17487/RFC8858, January 2021, <https://www.rfc-editor.org/info/rfc8858>.

[RFC8859] Nandakumar, S., "A Framework for Session Description Protocol (SDP) Attributes When Multiplexing", RFC 8859, DOI 10.17487/RFC8859, January 2021, <https://www.rfc-editor.org/info/rfc8859>.

10.2 参考文献

[RFC3389] Zopf, R., "Real-time Transport Protocol (RTP) Payload for Comfort Noise (CN)", RFC 3389, DOI 10.17487/RFC3389, September 2002, <https://www.rfc-editor.org/info/rfc3389>.

[RFC3556] Casner, S., "Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth", RFC 3556, DOI 10.17487/RFC3556, July 2003, <https://www.rfc-editor.org/info/rfc3556>.

[RFC3960] Camarillo, G. and H. Schulzrinne, "Early Media and Ringing Tone Generation in the Session Initiation Protocol (SIP)", RFC 3960, DOI 10.17487/RFC3960, December 2004, <https://www.rfc-editor.org/info/rfc3960>.

[RFC4568] Andreasen, F., Baugher, M., and D. Wing, "Session Description Protocol (SDP) Security Descriptions for Media Streams", RFC 4568, DOI 10.17487/RFC4568, July 2006, <https://www.rfc-editor.org/info/rfc4568>.

[RFC4588] Rey, J., Leon, D., Miyazaki, A., Varsa, V., and R. Hakenberg, "RTP Retransmission Payload Format", RFC 4588, DOI 10.17487/RFC4588, July 2006, <https://www.rfc-editor.org/info/rfc4588>.

[RFC4733] Schulzrinne, H. and T. Taylor, "RTP Payload for DTMF Digits, Telephony Tones, and Telephony Signals", RFC 4733, DOI 10.17487/RFC4733, December 2006, <https://www.rfc-editor.org/info/rfc4733>.

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, DOI 10.17487/RFC5245, April 2010, <https://www.rfc-editor.org/info/rfc5245>.

[RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, DOI 10.17487/RFC5506, April 2009, <https://www.rfc-editor.org/info/rfc5506>.

[RFC5576] Lennox, J., Ott, J., and T. Schierl, "Source-Specific Media Attributes in the Session Description Protocol (SDP)", RFC 5576, DOI 10.17487/RFC5576, June 2009, <https://www.rfc-editor.org/info/rfc5576>.

[RFC5763] Fischl, J., Tschofenig, H., and E. Rescorla, "Framework for Establishing a Secure Real-time Transport Protocol (SRTP) Security Context Using Datagram Transport Layer Security (DTLS)", RFC 5763, DOI 10.17487/RFC5763, May 2010, <https://www.rfc-editor.org/info/rfc5763>.

[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, DOI 10.17487/RFC5764, May 2010, <https://www.rfc-editor.org/info/rfc5764>.

[RFC6120] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 6120, DOI 10.17487/RFC6120, March 2011, <https://www.rfc-editor.org/info/rfc6120>.

[RFC6464] Lennox, J., Ed., Ivov, E., and E. Marocco, "A Real-time Transport Protocol (RTP) Header Extension for Client-to- Mixer Audio Level Indication", RFC 6464, DOI 10.17487/RFC6464, December 2011,

<https://www.rfc-editor.org/info/rfc6464>.

[RFC8828] Uberti, J. and G. Shieh, "WebRTC IP Address Handling Requirements", RFC 8828, DOI 10.17487/RFC8828, January 2021, <https://www.rfc-editor.org/info/rfc8828>.

[SDP4WebRTC] Nandakumar, S. and C. Jennings, "Annotated Example SDP for WebRTC", Work in Progress, Internet-Draft, draft-ietf- rtcweb-sdp-14, 17 December 2020, <https://tools.ietf.org/html/draft-ietf-rtcweb-sdp-14>.

[TS26.114] 3GPP, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; IP Multimedia Subsystem (IMS); Multimedia Telephony; Media handling and interaction (Release 16)", 3GPP TS 26.114 V16.3.0, September 2019, <https://www.3gpp.org/DynaReport/26114.htm>.

[W3C.webrtc] Jennings, C., Ed., Boström, H., Ed., and J. Bruaroey, Ed., "WebRTC 1.0: Real-time Communication Between Browsers", World Wide Web Consortium PR PR-webrtc-20201215, December 2020, <https://www.w3.org/TR/2020/PR-webrtc-20201215/>.

付録 A. SDP ABNF 構文

5.8 節で実行した構文検証では、ABNF 定義の次のリストを使用する。

属性	参考
ptime	[RFC4566] の 6 章
maxptime	[RFC4566] の 6 章
rtpmap	[RFC4566] の 6 章
recvonly	[RFC4566] の 9 章
sendrecv	[RFC4566] の 9 章
sendonly	[RFC4566] の 9 章
inactive	[RFC4566] の 9 章
fmtp	[RFC4566] の 9 章
rtcp	[RFC3605] の 2.1 節
setup	[RFC4145] の 4 章
fingerprint	[RFC8122] の 5 章
rtcp-fb	[RFC4585] の 4.2 節
extmap	[RFC5285] の 7 章
mid	[RFC5888] の 4 章
group	[RFC5888] の 5 章
imageattr	[RFC6236] の 3.1 節
Extmap (encrypt option)	[RFC6904] の 4 章
candidate	[RFC8839] の 5.1 節
remote-candidates	[RFC8839] の 5.2 節
ice-lite	[RFC8839] の 5.3 節
ice-ufrag	[RFC8839] の 5.4 節
ice-pwd	[RFC8839] の 5.4 節
ice-options	[RFC8839] の 5.6 節
msid	[RFC8830] の 3 章
rid	[RFC8851] の 10 章
simulcast	[RFC8853] の 5.1 節
tls-id	[RFC8842] の 4 章

表 A.1　SDP ABNF リファレンス