intdash®

White Paper

iSCP (intdash Stream Control Protocol)

30th May, 2018

aptpod

iSCP (intdash Stream Control Protocol)

intdashの各種機能を実現するために、当社が独自に開発し各モジュールの内部通信に使用されているプロトコルが **iSCP (intdash Stream Control Protocol)** です。iSCPは、ストリーミングデータの中継用ブローカーとエッジデバイス側モジュールとの間のデータストリーミングに使用され、完全回収や効率伝送、流量抑制といったintdashの主要機能を実現します。

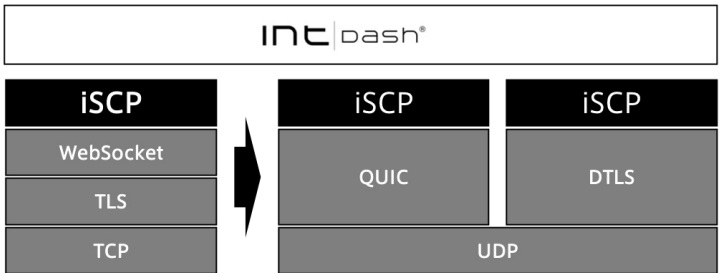
プロトコルスタック

iSCPの現在のバージョンでは、トランスポートとして WebSocket Secure を使用しています。WebSocketを採用するメリットは、次のとおりです。

- TCP/TLSとほぼ同様の双方向ソケット通信が可能
- 経路圧縮機能をプロトコルレベルでサポートしている
- HTTPポートを使用しており、プロキシやファイアウォールに遮断されにくい
- Webシステム向けサービスの活用により開発コストが抑えられる

デメリットとしては、WebSocketヘッダが余分なオーバーヘッドとして付与される点が存在するもののiSCPの備える送信バッファリングの機能により、ヘッダによる伝送効率の低下は抑制されます。

ただし、WebSocketはTCPをベースとするプロトコルであり、遠隔制御など遅延に対する厳しい要件のあるユースケースに対しては、最適解ではありません。当社では、今後ますますこのような厳しい要件は増えていくと予想しており、UDPベースの新しいプロトコルであるQUICやUDP/DTLSをトランスポートとして利用するための調査や検討も開始しています。



データフォーマット

伝送時のデータ構造としては、発生したデータそれ自体を表す **Unit** 構造と完全回収のための **Section** 構造、効率伝送のための **Flush** 構造が重要となります。

Unit構造

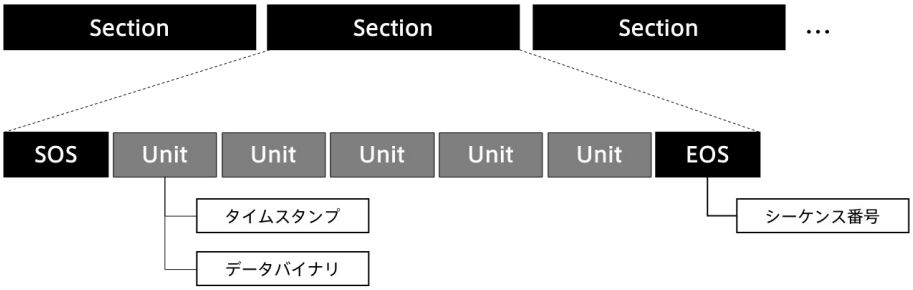
発生したそれぞれの時系列データは、Unitとよばれる構造に格納されます。Unit構造は、ストリームを識別するためのストリームID、発生時刻を表す経過時間およびデータ本体であるペイロードとその他細かなメタ情報から構成されます。

Section構造

各時系列データを格納したUnitは、Section Marker（SOSおよびEOS[1]）とよばれる目印を前後に付与されることでSectionとなります。このSectionが、WebSocket Secureによって形成されたセキュアな通信路を通って連続的にストリーミングされます。

Sectionは、前述の通り完全回収のための構造であり、付与されるSection Markerには、欠損の確認に使用されるシーケンシャルな番号が格納されます。また、Sectionを受け取ったサーバは、永続化の完了後にエッジデバイスへACKを返却することにより、発生データの完全回収のための一連のシーケンスが完了します。

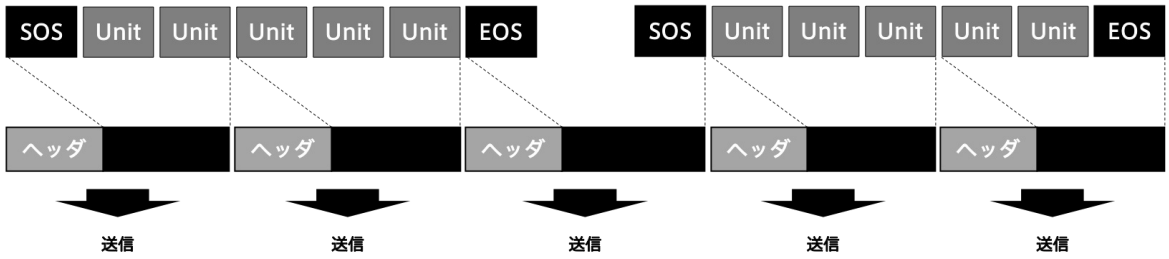
Sectionを長くとることにより、ACKを返却・確認する回数が減る、永続化のためのI/O処理回数が減るなどエッジデバイスやサーバの処理負荷に軽減が見込めます。一方で、欠損が生じた場合に再送しなければならない単位が大きくなる、エッジデバイスの再送キューやサーバ側でのバッファが長くなりやすいといったデメリットもあり、どの程度のUnitをSectionにまとめるかは、アプリケーションの要件に応じて設定をする必要があります。



Flush構造

WebSocket Secure上をSectionが伝送される際、実際にはUnitやSection Markerの羅列が送信されることになります。Flushは、伝送効率の向上のためにこれらのUnitやSection Merkerを一定期間バッファリングし、一括送信する際の送信単位です。

Flushを長くとることにより、ネットワークへの送受信要求回数が減る、ヘッダによるオーバーヘッドが減るなど伝送効率の向上が見込めます。一方で、バッファリング遅延が加算されるため、Section同様、Flushの長さについても、アプリケーションの要件に応じて設定をする必要があります。

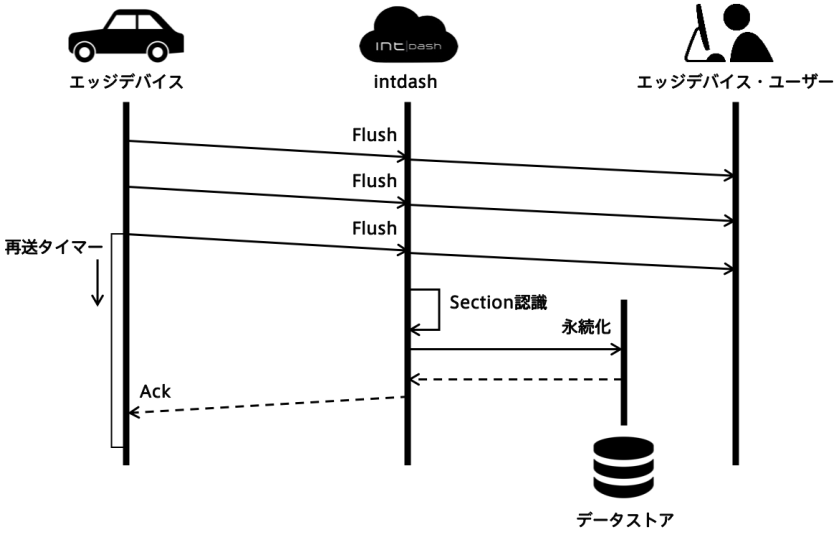


完全回収シーケンス

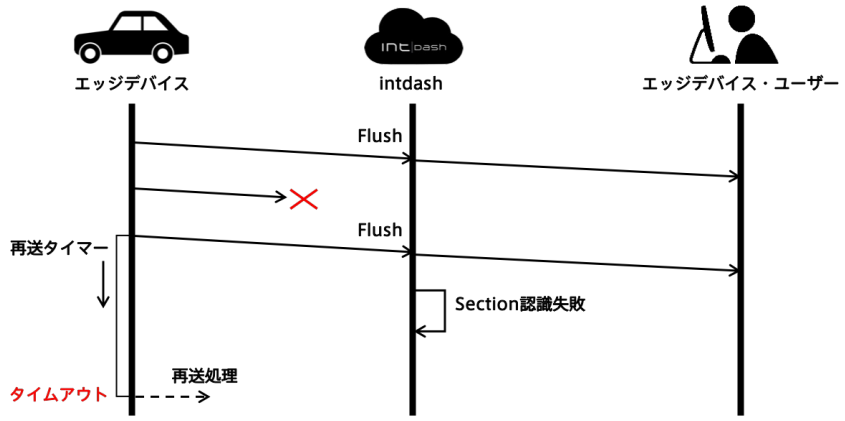
iSCPにおける完全回収は、Section単位でデータストアへの到達確認を実施することで実現されます。

まずエッジデバイス側では、発生したデータがバッファリングされ、Flushにまとめられてサーバーへの送信されます。Flushを受け取ったサーバは、受信側エッジデバイスへすぐさま転送すると同時にFlushにまとめられたUnitおよびSection Markerを順次検査し、Sectionを抽出します。Sectionが抽出されると、内包されたUnitを全てデータストアに永続化した後、Section ACKをエッジデバイスに返送します。

一方、エッジデバイス側では、サーバへ送信済みのSectionについて、所定のタイムアウト時間が経過するまでSection ACKの返却を監視し、タイムアウトしてもSection ACKが返却されなかったSectionについては、内包されるUnitを再送ストリームに引き継ぎます。再送ストリームでは、定期的に再送に回されたUnitを監視し、再送が必要な Unitが存在すれば再送を実施します。



ここで、iSCPは永続化されるデータの完全性は保証するものの、リアルタイムにストリーミングされるデータについては、完全性を保証しない点に注意が必要です。iSCPの現在のバージョンでは、トランスポートへWebSocket Secureの使用を想定しているため、TCP/TLSと同等の欠損補償および順序保証は期待できますが、iSCPのレイヤにおいては、伝送遅延の抑制のために何も付加的な処理を行っていません。また、後述のダイナミックフィルタによって、意図的にデータ流量が抑制されている可能性があることにも注意が必要です。



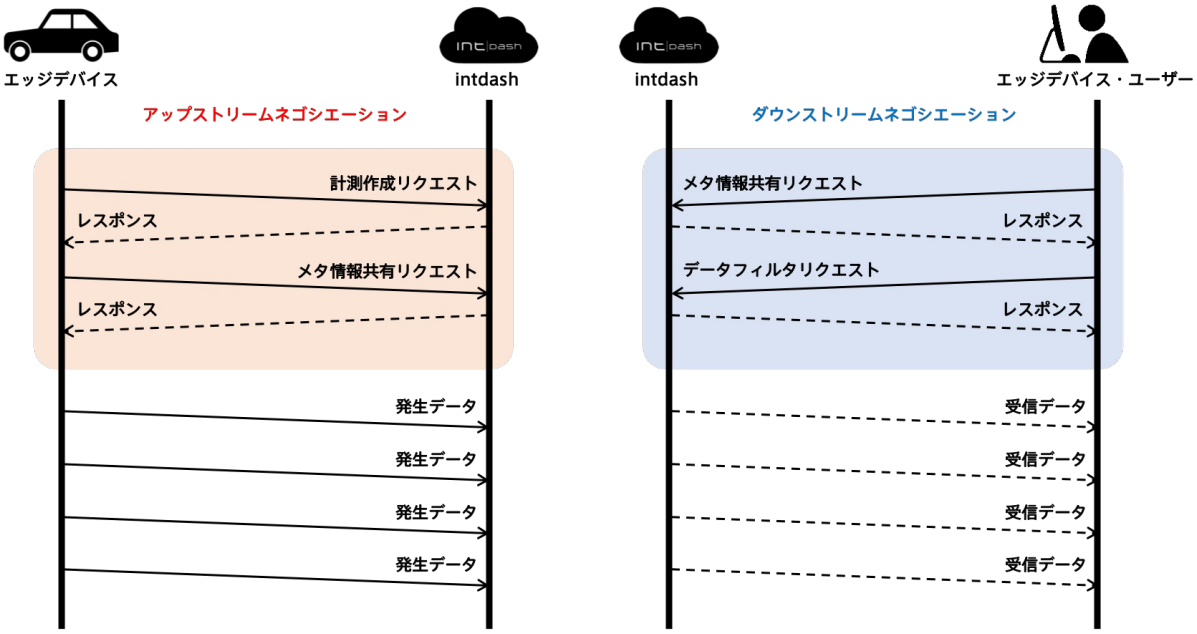
ネゴシエーションシーケンス

iSCPでは、送受信の開始前に実施されるネゴシエーションの手順を定義しています。ネゴシエーションにより、通信中に変更されないメタ情報を送受信者間で共有し、ストリーミング時の送信データ量を削減します。

ネゴシエーションには、エッジデバイスからブローカーに送信する方向で使用されるアップストリームネゴシエーションと、ブローカーからエッジデバイスへ送信する方向で使用されるダウンストリームネゴシエーションの2種類が存在します。

アップストリームネゴシエーションでは、必要に応じて新しい計測が生成されます。既存の計測データとしてストリーミングを開始する場合には、計測の作成は必須ではありません。計測の作成が終わると、静的なメタ情報の共有シーケンスに入り、計測ID/送信元エッジID/送信先エッジID（複数）/永続化要否フラグ/再送フラグを共有してストリームを開きます。

ダウンストリームネゴシエーションでは、まず静的メタ情報の共有シーケンスにより、送信元エッジID/送信先エッジIDを共有してストリームを開きます。その後、受信データのフィルタリング設定シーケンスにより、ストリームごとにサーバから送信されるデータの種別を設定します。



ネゴシエーションにより開かれたアップストリーム／ダウンストリームは、受信側でストリームIDにより区別されるため、1つのコネクションに多重化して送信できます。