MQTT V3.1 プロトコル仕様

著者:

International Business Machines Corporation (IBM)
Eurotech

はじめに

MQ Telemetry Transport (MQTT) は、ブローカー・ベースの軽量なパブリッシュ/サブスクライブ型メッセージ・プロトコルです。MQTT はオープンで単純、軽量であるように、そして容易に実装できるように設計されています。こうした特徴から、以下のような制約された環境で使用するのに理想的です (ただし、必ずしもこれらの環境での使用に限定されるわけではありません)。

- ネットワークのコストが高い、帯域幅が狭い、または信頼性が低い環境
- プロセッサーやメモリーのリソースが限られている組み込みデバイスで稼働する環境

このプロトコルには以下のような特徴があります。

- パブリッシュ/サブスクライブ型のメッセージ・パターンにより、1 対多でメッセージを配布することができ、またアプリケーションを分離することができます。
- メッセージ・トランスポートはペイロードの内容に依存しません。
- ベースとなるネットワーク接続に TCP/IP を使用します。
- メッセージ送信のサービス品質には、以下の 3 種類があります。
    - 「最高 1 回 (At most once)」: メッセージは、基となる TCP/IP ネットワークのベスト・エフォートに従って送信されます。メッセージの損失や重複が発生する可能性があります。このレベルはアンビエント・センサーのデータなどに利用することができます (アンビエント・センサーでは、個別の読み取りが失われても問題ありません。すぐに次の読み取りがパブリッシュされます)。
    - 「最低 1 回 (At least once)」: メッセージが必ず到着することが保証されますが、重複して到着する可能性があります。
    - 「正確に 1 回 (Exactly once)」: メッセージが 1 回のみ到着することが保証されます。このレベルは、例えば、メッセージの重複や損失が不正な課金の適用につながる課金システムなどで利用できます。

- トランスポートのオーバーヘッドが小さく (固定長ヘッダーは 2 バイトのみ)、プロトコルのやりとりが最低限に抑えられているため、ネットワーク・トラフィックを減らすことができます。
- 「遺言 (Last Will and Testament)」機能により、関心のあるパーティーにクライアントの異常切断を通知するメカニズムがあります。

Copyright Notice

目次

1. 概要

この仕様は 3 つのメイン・セクションに分かれています。

- すべてのパケット・タイプに共通するメッセージ・フォーマット
- パケット・タイプごとの仕様の詳細
- クライアントとサーバー間のパケット・フロー

トピック・ワイルドカードの使用方法については、付録で説明しています。

1.1. 変更

MQTT V3 から MQTT V3.1 における変更点は以下のとおりです。

- ユーザー名とパスワードを CONNECT パケットを使用して送信可能
- セキュリティーの問題に対応した、CONNACK パケットの新しい戻りコード
- 無許可の PUBLISH コマンドや SUBSCRIBE コマンドの情報がクライアントに通知されない点、およびコマンドを実行していない場合にも通常の MQTT フローが実行される点を明示
- MQTT のストリングでの UTF-8 のフルサポート (以前は US-ASCII サブセットのみ)

CONNECT パケットにより渡されるプロトコル・バージョン番号はこの改訂では変更されず、「3」のままとなっています。「残りの長さ」フィールドを正しく識別する限り、追加のセキュリティー情報は無視され、既存の MQTT V3 サーバー実装でも、この改訂をサポートするクライアントとの接続を受け入れることができます。

2. メッセージ・フォーマット

各 MQTT コマンド・メッセージのメッセージ・ヘッダーには固定ヘッダーが含まれます。可変ヘッダーやペイロードも必要なメッセージもあります。以下のセクションで、メッセージ・ヘッダーの各部分のフォーマットについて説明します。

2.1. 固定ヘッダー

各 MQTT コマンド・メッセージのメッセージ・ヘッダーには固定ヘッダーが含まれています。以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ				DUP フラグ	QoS レベル		RETAIN
バイト 2	残りの長さ							

バイト 1

メッセージ・タイプおよびフラグ (DUP、QoS level、および RETAIN) フィールドを含みます。

バイト 2

(最小 1 バイト) 「残りの長さ」フィールドを含みます。

各フィールドについては、以下のセクションで説明しています。すべてのデータ値はビッグ・エンディアンの順序で、上位バイトが下位バイトの前に来ます。ワイヤレベルでは、16 ビット・ワードは最上位バイト (MSB)、最下位バイト (LSB) の順で表現されます。

メッセージ・タイプ

位置: バイト 1、ビット 7-4

4 ビットの符号なし値として表されます。以下の表に、このバージョンのプロトコルの列挙を示します。

ニーモニック	列挙	説明
予約済み	0	予約済み
CONNECT	1	クライアントがサーバーへの接続を要求
CONNACK	2	接続確認応答
PUBLISH	3	メッセージのパブリッシュ
PUBACK	4	パブリッシュ確認応答
PUBREC	5	パブリッシュの受信 (送達保証 パート 1)
PUBREL	6	パブリッシュ のリリース (送達保証 パート 2)
PUBCOMP	7	パブリッシュの完了 (送達保証パート 3)
SUBSCRIBE	8	クライアント・サブスクライブ要求
SUBACK	9	サブスクライブ確認応答
UNSUBSCRIBE	10	クライアント・アンサブスクライブ要求
UNSUBACK	11	アンサブスクライブ確認応答
PINGREQ	12	PING 要求
PINGRESP	13	PING 応答
DISCONNECT	14	クライアント切断中
予約済み	15	予約済み

フラグ

バイト 1 の残りのビットには、DUP、QoS、および RETAIN フィールドが含まれます。ビット位置は、以下の表に示すフラグを表すようにエンコードされています。

ビット位置	名前	説明
3	DUP	重複送信
2-1	QoS	サービス品質 (Quality of Service)
0	RETAIN	RETAIN フラグ

DUP

位置: バイト 1、ビット 3

このフラグは、クライアントまたはサーバーが PUBLISH、PUBREL、SUBSCRIBE または UNSUBSCRIBE メッセージを再送しようとする際に設定されます。これは、QoS の値が 0 より大きいメッセージに適用され、確認応答が必須です。DUP ビットが設定されている場合、可変ヘッダーにメッセージ ID が含まれます。

受信側はこのフラグを、以前にメッセージを受信している可能性を示すヒントとしてのみ使用するべきです。重複を検出する目的でこのフラグに依存するべきではありません。

QoS

位置: バイト 1、ビット 2-1

このフラグは、 PUBLISH メッセージの送達保証レベルを示します。以下の表に、QoS レベルを示します。

QoS 値	ビット 2	ビット 1	説明		
0	0	0	最高 1 回	送出のみ	<=1
1	0	1	最低 1 回	確認応答による到達確認	>=1
2	1	0	正確に 1 回	送達保証	=1
3	1	1	予約済み		

RETAIN

位置: バイト 1、ビット 0

このフラグは PUBLISH メッセージにのみ使用されます。クライアントがサーバーに Retain フラグが (1) に設定されている PUBLISH メッセージを送信すると、サーバーは当該メッセージが現行のサブスクライバーに送信された後も、メッセージを保持します。

トピックに新規サブスクリプションが設定されると、そのトピックの最新の保存メッセージが、Retain フラグが設定された状態でサブスクライバーに送信されます。保存メッセージがない場合、送信されるものはありません。

これは、パブリッシャーが「例外ごとにレポート」ベースでメッセージを送信する場合のように、メッセージ間の時間があく可能性がある状況に有用です。これにより新しいサブスクライバーは、保存された値 (つまり、前回の正常値) を持つデータを即時受信できます。

サーバーが PUBLISH が到着した時点のサブスクリプション情報に基づいて、クライアントに PUBLISH を送信する際には、元の PUBLISH の Retain フラグに関わらず Retain フラグは設定されません。これによりクライアントは、受信

したメッセージが保存されていたものか、「ライブ」で受信したものかを区別できます。

保存されたメッセージは、サーバーの再始動を跨いで維持されます。

サーバーは、メッセージを保持している当該トピックに対して、Retain フラグが設定された長さが 0 のペイロードのメッセージを受信した場合、保存しているメッセージを削除できます。

残りの長さ

位置: バイト 2

現行メッセージ内の残りのバイト数を表します (可変ヘッダーおよびペイロードのデータを含む)。

可変長のエンコーディング・スキームで、127 バイト長までのメッセージに対して単一バイトを使用します。それより長いメッセージは以下のように扱われます。各バイトの 7 ビットに「残りの長さ」のデータをエンコードし、8 番目のビットは、後続バイトが「残りの長さ」データを含むかを示します。各バイトには、128 の値と「後続ビット」がエンコードされます。例えば、10 進の「64」を単一バイトとしてエンコードすると、10 進では 64 、16 進では 0x40 となります。10 進の 321 l (= 65 + 2*128) は 2 バイトでエンコードされ、最下位ビットとなります。最初のバイトは 65+128 = 193 です。最上位のビットは、最低後続の 1 バイトを示すために設定されます。2 バイト目は 2 です。

このプロトコルでは、「残りの長さ」を表すバイト数が最大 4 に制限されます。これにより、 アプリケーションは最大 268 435 455 (256 MB) までメッセージを送信できます。ワイヤレベルでののでのこの数値の表示は次のようになります。0xFF、 0xFF、0xFF、0x7F。

以下の表に、増大したバイト数で表示される「残りの長さ」の値を示します。

桁数	開始	終了
1	0 (0x00)	127 (0x7F)
2	128 (0x80, 0x01)	16 383 (0xFF, 0x7F)
3	16 384 (0x80, 0x80, 0x01)	2 097 151 (0xFF, 0xFF, 0x7F)
4	2 097 152 (0x80, 0x80, 0x80, 0x01)	268 435 455 (0xFF, 0xFF, 0xFF, 0x7F)

10 進 (X) を可変長エンコーディング・スキームでエンコードするためのアルゴリズムは、以下のようになります。

```
do
      digit = X MOD 128
```

```
      X = X DIV 128
      // if there are more digits to encode, set the top bit of this digit
      if ( X > 0 )
             digit = digit OR 0x80
      endif
      'output' digit
while ( X> 0 )
```

ここで MOD は剰余演算子 (C 言語では %)、DIV は整数除算 (C 言語では /)、OR は、ビット OR 演算 (C 言語では |) です。

「残りの長さ」フィールドをデコードするアルゴリズムは、以下のようになります。

```
multiplier = 1
value = 0
do
     digit = 'next digit from stream'
     value += (digit AND 127) * multiplier
     multiplier *= 128
while ((digit AND 128) != 0)
```

ここで AND はビット AND 演算 (C 言語では &) です。

このアルゴリズムでは、処理が完了すると、value に残りの長さがバイト単位で挿入されます。

「残りの長さ」は、可変ヘッダーの一部としてはエンコードされません。「残りの長さ」のエンコードで使用されるバイト数は、「残りの長さ」の値とは関係ありません。可変長の「拡張バイト」は、可変ヘッダーではなく固定ヘッダーに含まれます。

2.2. 可変ヘッダー

一部のタイプの MQTT コマンド・メッセージには、可変ヘッダー・コンポーネントが含まれます。可変ヘッダー・コンポーネントは、固定ヘッダーとペイロードの間に位置します。

可変長の「残りの長さ」フィールドは、可変ヘッダーには含まれません。「残りの長さ」フィールド自身のバイト数は、「残りの長さ」の値で表されるバイト数とは関連していません。この値では可変ヘッダーとペイロードのみが考慮されます。詳細は、『固定ヘッダー』を参照してください。

可変ヘッダー・フィールドのフォーマットは、以下のセクションにヘッダーに現れる順番で説明されています。

プロトコル名

プロトコル名は MQTT CONNECT メッセージの可変ヘッダーに存在します。このフィールドは UTF エンコードされたストリングで、プロトコル名 MQIsdp となります (大文字小文字を区別します)。

プロトコル・バージョン

プロトコル・バージョンは、CONNECT メッセージの可変ヘッダーに存在します。

フィールドは 8 ビットの符号なし値で、クライアントで使用されるプロトコルの改訂レベルを表します。現行バージョンのプロトコルのプロトコル・バージョン・フィールドの値は、以下の表に示すとおり 3 (0x03) です。

ビット	7	6	5	4	3	2	1	0
	プロトコル・バージョン							
	0	0	0	0	0	0	1	1

接続フラグ

Clean session、Will、Will QoS、および Retain の各フラグは、CONNECT メッセージの可変ヘッダーに存在します。

Clean Session フラグ

位置: 接続フラグ・バイトのビット 1

(0) に設定されていない場合、サーバーはクライアントの切断後にクライアントのサブスクリプションを保存する必要があります。これには、クライアントが再接続時に送信できるように、サブスクライブされたトピックの QoS 1 および QoS 2 メッセージを引き続き保存することも含まれます。サーバーは、接続が失われた時点に送信される処理中メッセージの状況を保持する必要もあります。この情報は、クライアントに再接続するまで保持しておく必要があります。

(1) に設定されている場合、サーバーは以前に保持していたクライアントに関する情報を破棄し、接続を「クリーン」として扱う必要があります。サーバーは、クライアント切断時にすべての状態を破棄する必要もあります。

通常クライアントはいずれかのモードで動作し、変更されません。どのモードが選択されるかは、アプリケーションによって異なります。クリーン・セッション・クライアントは保存された情報を受信できず、接続するたびに再サブスクライブする必要があります。非クリーン・セッション・クライアントは、切断されている間にパブリッシュされた QoS 1 メッセージや QoS 2 メッセージを損失することはありません。QoS 0 メッセージは、ベスト・エフォート・ベースで送信されるため、保存されません。

このフラグは以前は「Clean start」と呼ばれていました。最初の接続だけでなくセッション全体に適用されることを明確にするために、名前が変更されました。

サーバーは、以降クライアントが再接続しないと判断できる場合のために、当該クライアントに関する保存情報をクリアする管理メカニズムを提供することができます。

ビット	7	6	5	4	3	2	1	0
	User Name フラグ	Password フラグ	Will Retain	Will QoS		Will フラグ	Clean Session	予約済み
	x	x	x	x	x	x		x

このバイトのビット 0 は、現行バージョンのプロトコルでは使用されません。将来の使用のために予約されています。

Will (遺言) フラグ

位置: 接続フラグ・バイトのビット 2

Will (遺言) メッセージは、クライアントとの通信中にサーバーで I/O エラーが発生したとき、またはクライアントがキープアライブ・タイマーの時間内の通信で失敗したときに、サーバーがクライアントに代わってパブリッシュするメッセージを定義します。Will メッセージは、クライアントから DISCONNECT メッセージを受け取ったサーバーからは送信されません。

Will フラグが設定されている場合、Will QoS および Will Retain フィールドが接続フラグ・バイトに存在し、Will トピックおよび Will メッセージ・フィールドがペイロードに存在している必要があります。

以下の表に、Will フラグのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
	User Name フラグ	Password フラグ	Will Retain	Will QoS		Will フラグ	Clean Session	予約済み
	x	x	x	x	x		x	x

このバイトのビット 0 は、現行バージョンのプロトコルでは使用されません。将来の使用のために予約されています。

Will (遺言) QoS

位置: 接続フラグ・バイトのビット 4 およびビット 3

接続側クライアントは、クライアントが意図せず切断したときに送信される遺言メッセージの QoS レベルを、Will QoS フィールドに指定します。遺言メッセージは、CONNECT メッセージのペイロードで定義されます。

Will フラグが設定されている場合、Will QoS フィールドは必須です。設定されていない場合は、その値は無視されます。

Will QoS の値は 0 (0x00)、1 (0x01)、または 2 (0x02) です。以下の表に、Will QoS フラグを示します。

ビット	7	6	5	4	3	2	1	0
	User Name フラグ	Password フラグ	Will Retain	Will QoS		Will フラグ	Clean Session	予約済み
	x	x	x			1	x	x

このバイトのビット 0 は、現行バージョンのプロトコルでは使用されません。将来の使用のために予約されています。

Will (遺言) Retain フラグ

位置: 接続フラグ・バイトのビット 5

Will Retain フラグは、クライアントが意図せず切断されたときにサーバーがクライアントに代わってパブリッシュした遺言メッセージを保存すべきかどうかを示します。

Will フラグが設定されている場合、Will Retain フラグは必須です。設定されていない場合は、その値は無視されます。以下の表に、Will Retain フラグのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
	User Name フラグ	Password フラグ	Will Retain	Will QoS		Will フラグ	Clean Session	予約済み
	x	x		x	x	1	x	x

このバイトのビット 0 は、現行バージョンのプロトコルでは使用されません。将来の使用のために予約されています。

User name および Password フラグ

位置: 接続フラグ・バイトのビット 6 およびビット 7

接続側クライアントは、ユーザー名とパスワードを指定できます。フラグ・ビットを設定することにより、CONNECT メッセージのペイロードにユーザー名と、オプションでパスワードが含まれることを示します。

User Name フラグが設定されている場合、ユーザー名フィールドは必須です。設定されていない場合は、その値は無視されます。Password フラグが設定されている場合、パスワード・フィールドは必須です。設定されていない場合は、その値は無視されます。ユーザー名を指定せずにパスワードを指定すると無効になります。

ビット	7	6	5	4	3	2	1	0
	User Name フラグ	Password フラグ	Will Retain	Will QoS		Will フラグ	Clean Session	予約済み
			x	x	x	x	x	x

このバイトのビット 0 は、現行バージョンのプロトコルでは使用されません。将来の使用のために予約されています。

キープアライブ・タイマー

キープアライブ・タイマーは MQTT CONNECT メッセージの可変ヘッダーに存在します。

秒単位で計測されるキープアライブ・タイマーは、クライアントから受信するメッセージの間隔の最大時間を定義します。サーバーは、クライアントに対するネットワーク接続が切断されたことを、TCP/IP タイムアウトを長い時間待つことなく検出できます。クライアントは、キープアライブ時間間隔ごとにメッセージを送信する必要があります。この時間間隔の間データ関連メッセージがない場合、クライアントが PINGREQ メッセージを送信し、サーバーが PINGRESP メッセージで確認応答を行います。

サーバーがクライアントからのメッセージを、キープアライブ時間間隔の 1.5 倍の時間以内に受信しない場合 (クライアントはその時間の半分の「猶予」があります)、サーバーは、クライアントが DISCONNECT メッセージを送った場合と同様にクライアントを切断します。このアクションは、クライアントのサブスクリプションには影響しません。詳細は、『DISCONNECT』を参照してください。

クライアントが PINGREQ を送信した後、キープアライブ時間間隔以内に PINGRESP メッセージを受信しない場合、 クライアントは TCP/IP ソケット接続を閉るべきです。

キープアライブ・タイマーは 16 ビット値で、時間間隔を秒単位で表します。実際の値はアプリケーション固有ですが、一般的な値は数分です。最大値は約 18 時間です。値 0 はクライアントが切断されないことを意味します。

以下の表に、キープアライブ・タイマーのフォーマットを示します。キープアライブ・タイマーの 2 バイトの順序は、MSB、LSB です (ビッグ・エンディアン)。

ビット	7	6	5	4	3	2	1	0
	キープアライブ MSB							
	キープアライブ LSB							

接続戻りコード

接続戻りコードは CONNACK メッセージの可変ヘッダーで送信されます。

このフィールドには、1 バイトの符号なし戻りコードが定義されます。以下の表に示す値の意味は、メッセージ・タイプによって異なります。戻りコード (0) は通常、正常終了を意味します。

列挙	16 進	意味
0	0x00	接続許可
1	0x01	接続拒否:プロトコル・バージョン許容不可
2	0x02	接続拒否:識別子拒否
3	0x03	接続拒否:サーバー使用不可
4	0x04	接続拒否:ユーザー名またはパスワードが不正
5	0x05	接続拒否:権限なし
6-255		将来の使用のため予約

ビット	7	6	5	4	3	2	1	0
	戻りコード							

トピック名

トピック名は、MQTT PUBLISH メッセージの可変ヘッダーに存在します。

トピック名は、ペイロード・データがパブリッシュされる情報チャネルを識別するキーです。サブスクライバーは、キーを使用して、パブリッシュされた情報を受け取る必要がある情報チャネルを識別します。

トピック名は、UTF エンコード・ストリングです。詳細は、『MQTT および UTF-8』のセクションを参照してください。トピック名の最大長は、32,767 文字となっています。

2.3. ペイロード

以下のタイプの MQTT コマンド・メッセージには以下のペイロードが含まれます。

CONNECT
: このペイロードには、1 つ以上の UTF-8 エンコード・ストリングが含まれます。それらは、クライアントの固有識別子、Will トピックとメッセージ、および使用するユーザー名とパスワードを指定します。1 つ目以外はすべてオプションで、それらが存在するかどうかは可変ヘッダー内のフラグに基づいて判別されます。

SUBSCRIBE
: このペイロードには、クライアントがサブスクライブできるトピック名のリストと、QoS レベルが含まれます。これらのストリングは、UTF エンコードされています。

SUBACK
: このペイロードには、付与された QoS レベルのリストが含まれます。これらは、サーバーの管理者がクライアントに特定のトピック名のサブスクライブを許可した QoS レベルです。付与された QoS レベルは、対応する SUBSCRIBE メッセージ内のトピック名と同じ順序でリストされます。

PUBLISH メッセージのペイロード部分には、アプリケーション固有のデータのみが含まれます。データの特質または内容についての前提はなく、メッセージのこの部分は BLOB として扱われます。

アプリケーションでペイロード・データに圧縮を適用させる場合は、適切なペイロード・フラグ・フィールドで圧縮の詳細を処理するようにアプリケーションで定義する必要があります。固定または可変ヘッダーにアプリケーション固有のフラグを定義することはできません。

2.4. メッセージ識別子

メッセージ識別子は、MQTT メッセージ PUBLISH、PUBACK、PUBREC、PUBREL、PUBCOMP、SUBSCRIBE、SUBACK、UNSUBSCRIBE、UNSUBACK の可変ヘッダーに存在します。

メッセージ識別子 (メッセージ ID) フィールドは、固定ヘッダー内の QoS ビットが QoS レベル 1 または 2 を示すメッセージ内にのみ存在します。詳細は、『Quality of Service のレベルおよびフロー』のセクションを参照してください。

メッセージ ID は、特定の通信方向の一連の「処理中」のメッセージ間で一意の、16 ビットの符号なし整数です。通常、メッセージ間の移動で 1 ずつ増加しますが、それは必須ではありません。

クライアントは、メッセージ ID の固有のリストを、接続されたサーバーで使用されるメッセージ ID とは別に保持します。クライアントは、メッセージ ID 1 で PUBLISH を受信するのと同時に、メッセージ ID 1 で PUBLISH を送信できます。

メッセージ識別子の 2 バイトの順序は、MSB、LSB です (ビッグ・エンディアン)。

メッセージ ID 0 は使用しないでください。無効なメッセージ ID として予約されています。

ビット	7	6	5	4	3	2	1	0
	メッセージ識別子 MSB							
	メッセージ識別子 LSB							

2.5. MQTT および UTF-8

UTF-8 は、テキストベース通信のサポートにおいて ASCII 文字のエンコード方式を最適化する、ユニコード文字ストリングの効率的なエンコード方式です。

MQTT では、以下の表に示すとおり、長さを示すための 2 バイトがストリングに接頭部として付加されます。

ビット	7	6	5	4	3	2	1	0
バイト 1	ストリング長 MSB							
バイト 2	ストリング長 LSB							
バイト 3 ...	エンコードされた文字データ							

ストリング長はエンコードされたストリングのバイト数であり、文字数ではありません。例えば、UTF-8 ではストリング“OTWP”は以下の表に示すとおりエンコードされます。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ長 MSB (0x00)							
	0	0	0	0	0	0	0	0
バイト 2	メッセージ長 LSB (0x04)							
	0	0	0	0	0	1	0	0
バイト 3	「O」(0x4F)							
	0	1	0	0	1	1	1	1
バイト 4	「T」(0x54)							
	0	1	0	1	0	1	0	0
バイト 5	「W」(0x57)							
	0	1	0	1	0	1	1	1
バイト 6	「P」(0x50)							
	0	1	0	1	0	0	0	0

Java の writeUTF() および readUTF() データ・ストリーム・メソッドはこのフォーマットを使用します。

2.6. 未使用ビット

未使用としてマークされているビットはすべて、ゼロ (0) に設定する必要があります。

3. コマンド・メッセージ

- CONNECT
- CONNACK
- PUBLISH
- PUBACK
- PUBREC
- PUBREL
- PUBCOMP
- SUBSCRIBE
- SUBACK
- UNSUBSCRIBE
- UNSUBACK
- PINGREQ
- PINGRESP
- DISCONNECT

3.1. CONNECT - クライアントからサーバーへの接続要求

クライアントからサーバーへの TCP/IP ソケット接続の確立時には、CONNECT フローを使用して、プロトコル・レベル・セッションを作成する必要があります。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (1)				DUP フラグ	QoS レベル		RETAIN
	0	0	0	1	x	x	x	x
バイト 2	残りの長さ							

DUP、QoS および RETAIN フラグは、CONNECT メッセージでは使用されません。

「残りの長さ」とは、可変ヘッダーの長さ (12 バイト) とペイロードの長さです。これはマルチバイト・フィールドにすることができます。

可変ヘッダー

以下の表に、可変ヘッダーのフォーマットの例を示します。

	説明	7	6	5	4	3	2	1	0
プロトコル名									
バイト 1	長さ MSB (0)	0	0	0	0	0	0	0	0
バイト 2	長さ LSB (6)	0	0	0	0	0	1	1	0
バイト 3	「M」	0	1	0	0	1	1	0	1
バイト 4	「Q」	0	1	0	1	0	0	0	1
バイト 5	「I」	0	1	0	0	1	0	0	1
バイト 6	「s」	0	1	1	1	0	0	1	1
バイト 7	「d」	0	1	1	0	0	1	0	0
バイト 8	「p」	0	1	1	1	0	0	0	0
プロトコル・バージョン番号									
バイト 9	バージョン (3)	0	0	0	0	0	0	1	1
接続フラグ									
バイト 10	User name フラグ (1) Password フラグ (1) Will RETAIN (0) Will QoS (01) Will フラグ (1) Clean Session (1)	1	1	0	0	1	1	1	x
キープアライブ・タイマー									
バイト 11	キープアライブ MSB (0)	0	0	0	0	0	0	0	0
バイト 12	キープアライブ LSB (10)	0	0	0	0	1	0	1	0

User name フラグ

(1) に設定します。

Password フラグ

(1) に設定します。

Clean Session フラグ

(1) に設定します。

キープアライブ・タイマー

10 秒 (0x000A) に設定します。

Will メッセージ

- Will フラグは (1) に設定します。
- Will QoS フィールドは 1 です。
- Will RETAIN フラグは (0) にクリアします。

ペイロード

CONNECT メッセージのペイロードには、可変ヘッダー内のフラグに基づいて、1 つ以上の UTF-8 エンコード・ストリングが含まれます。(存在する場合) ストリングは以下の順序とする必要があります。

クライアント識別子

最初の UTF エンコード・ストリングです。クライアント識別子 (クライアント ID) は、1 から 23 文字の長さで、サーバーに対しクライアントを一意に識別します。単一のサーバーに接続されたすべてのクライアントの間で一意とする必要があり、QoS レベル 1 および 2 のメッセージ ID メッセージを処理する際のキーとなります。クライアント ID に 23 文字を超える文字が含まれる場合、サーバーは CONNECT メッセージに対して、CONNACK 戻りコード 2 (識別子拒否) で応答します。

Will (遺言) トピック

Will フラグが設定されている場合、これは次の UTF-8 エンコード・ストリングとなります。Will メッセージは、Will トピックにパブリッシュされます。QoS レベルは Will QoS フィールドで定義され、RETAIN ステータスは可変ヘッダー内の Will RETAIN フラグで定義されます。

Will (遺言) メッセージ

Will フラグが設定されている場合、これは次の UTF-8 エンコード・ストリングとなります。Will メッセージは、クライアントが予期せずに切断された場合に Will トピックにパブリッシュされるメッセージの内容を定義します。これは、長さがゼロのメッセージである場合もあります。

Will メッセージは CONNECT メッセージに UTF-8 エンコードされていますが、Will トピックにパブリッシュされるときは、長さを表す最初の 2 バイトではなく、メッセージのバイトのみが送信されます。そのため、メッセージは 7 ビットの ASCII 文字のみで構成されている必要があります。

User Name

User Name フラグが設定されている場合、これは次の UTF エンコード・ストリングとなります。ユーザー名は、接続されているユーザーの名前を識別します。これは認証で使用することができます。ユーザー名は 12 文字以内にしておくことを推奨します (必須ではありません)。

以前の MQTT V3 の仕様との互換性を保つために、固定ヘッダーの「残りの長さ」フィールドが User Name フラグよりも優先されます。サーバー実装では、User Name フラグが設定されていてユーザー名ストリングがないという可能性

を考慮に入れる必要があります。これは有効で、接続は継続可能とするべきです。

Password

Password フラグが設定されている場合、これは次の UTF エンコード・ストリングとなります。接続されているユーザーに対応するパスワードで、認証に使用することができます。パスワードは 12 文字以内にしておくことを推奨します (必須ではありません)。

以前の MQTT V3 仕様との互換性を保つために、固定ヘッダーの「残りの長さ」フィールドが Password フラグよりも優先されます。サーバー実装では、Password フラグが設定されていてパスワード・ストリングがないという可能性を考慮に入れる必要があります。これは有効で、接続は継続可能とするべきです。

応答

サーバーは、クライアントからの CONNECT メッセージへの応答で CONNACK メッセージを送信します。

TCP/IP 接続が確立した後、サーバーが妥当な時間内に CONNECT メッセージを受信しない場合、サーバーは接続を閉じるべきです。

クライアントが妥当な時間内にサーバーから CONNACK メッセージを受信しない場合、クライアントは TCP/IP ソケット接続を閉じ、サーバーに対して新しいソケットを開いて CONNECT メッセージを発行することでセッションを再開するべきです。

これら 2 つの場合の「妥当な」の時間は、アプリケーションや通信インフラストラクチャーのタイプによって異なります。

同じクライアント ID のクライアントが既にサーバーに接続している場合、新しいクライアントの CONNECT フローを完了する前に、「古い」クライアントがサーバーによって切断されます。

クライアントが無効な CONNECT メッセージを送信した場合、サーバーは接続を閉じるべきです。このようなメッセージとしては、無効なプロトコル名やプロトコル・バージョン番号を含む CONNECT メッセージがあります。サーバーが CONNECT メッセージを十分に解析し、無効なプロトコルが要求されたことを判別できた場合、接続をドロップする前に、「接続拒否: プロトコル・バージョン許容不可」のコードを含む CONNACK の送信を試みることができます。

3.2. CONNACK - 接続要求への応答

CONNACK メッセージは、クライアントからの CONNECT 要求への応答としてサーバーから送信されるメッセージです。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (2)				DUP フラグ	QoS フラグ		RETAIN
	0	0	1	0	x	x	x	x
バイト 2	残りの長さ (2)							
	0	0	0	0	0	0	1	0

DUP、QoS および RETAIN フラグは、CONNACK メッセージでは使用されません。

可変ヘッダー

以下の表に、可変ヘッダーのフォーマットを示します。

	説明	7	6	5	4	3	2	1	0
トピック名圧縮応答									
バイト 1	予約済みの値です。使用されません。	x	x	x	x	x	x	x	x
接続戻りコード									
バイト 2	戻りコード								

以下の表に、1 バイトの符号なしの「接続戻りコード」フィールドの値を示します。

列挙	16 進	意味
0	0x00	接続許可
1	0x01	接続拒否:プロトコル・バージョン許容不可
2	0x02	接続拒否:識別子拒否
3	0x03	接続拒否:サーバー使用不可
4	0x04	接続拒否:ユーザー名またはパスワードが不正
5	0x05	接続拒否:権限なし
6-255		将来の使用のため予約

戻りコード 2 (識別子拒否) は、固有のクライアント識別子の長さが 1 から 23 文字の間以外の場合に送信されます。

ペイロード

ペイロードはありません。

3.3. PUBLISH - メッセージのパブリッシュ

PUBLISH メッセージは、関心のあるサブスクライバーに配布するために、クライアントからサーバーに送信されます。各 PUBLISH メッセージは、トピック名 (件名またはチャネルともいう) に関連付けられています。これは階層型の名前空間で、サブスクライバーが関心を登録することができる情報源の分類を定義します。特定のトピック名にパブリッシュされるメッセージは、接続されている該当トピックのサブスクライバーに送信されます。

クライアントが 1 つ以上のトピックをサブスクライブしている場合には、該当トピックにパブリッシュされたすべてのメッセージが PUBLISH メッセージとしてサーバーからクライアントに送信されます。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (3)				DUP フラグ	QoS レベル		RETAIN
	0	0	1	1	0	0	1	0
バイト 2	残りの長さ							

QoS レベル
: 1 に設定します。詳細は、『QoS』を参照してください。

DUP フラグ
: ゼロ (0) に設定します。これは、メッセージが初めて送信されていることを示します。詳細は、『DUP』を参照してください。

RETAIN フラグ
: ゼロに設定します。これは保持しないことを示します。詳細は、『Retain』を参照してください。

「残りの長さ」フィールド
: 可変ヘッダーの長さとペイロードの長さです。マルチバイト・フィールドとすることができます。

可変ヘッダー

可変ヘッダーには以下のフィールドが含まれています。

トピック名

UTF エンコード・ストリングです。

これにトピック・ワイルドカード文字を含めることはできません。

このストリングは、ワイルドカード文字を使用してサブスクライブされたクライアントで受信されると、クライアントで使用されるサブスクリプション・ストリング *ではなく*、送信側パブリッシャーが指定した絶対トピックとなります。

メッセージ ID

QoS レベル 1 および QoS レベル 2 のメッセージのためのものです。詳細は、『メッセージ識別子』を参照してください。

以下の表に、PUBLISH メッセージの可変ヘッダーの例を示します。

フィールド	値
トピック名:	「a/b」
QoS レベル	1
メッセージ ID:	10

以下の表に、この場合の可変ヘッダーのフォーマットを示します。

	説明	7	6	5	4	3	2	1	0
トピック名									
バイト 1	長さ MSB (0)	0	0	0	0	0	0	0	0
バイト 2	長さ LSB (3)	0	0	0	0	0	0	1	1
バイト 3	「a」(0x61)	0	1	1	0	0	0	0	1
バイト 4	「/」(0x2F)	0	0	1	0	1	1	1	1
バイト 5	「b」(0x62)	0	1	1	0	0	0	1	0
メッセージ識別子									
バイト 6	メッセージ ID MSB (0)	0	0	0	0	0	0	0	0
バイト 7	メッセージ ID LSB (10)	0	0	0	0	1	0	1	0

ペイロード

パブリッシュ用のデータが含まれます。このデータの内容とフォーマットはアプリケーション固有となります。固定ヘッダー内の「残りの長さ」フィールドには、可変ヘッダー長とペイロード長の両方が含まれます。つまり、PUBLISH に長さ 0 のペイロードを含むことができます。

応答

PUBLISH メッセージへの応答は、QoS レベルによって異なります。以下の表に、予想される応答を示します。

QoS レベル	予想される応答
QoS 0	なし
QoS 1	PUBACK
QoS 2	PUBREC

アクション

PUBLISH メッセージは、パブリッシャーからサーバーへ、またはサーバーからサブスクライバーへ送信できます。メッセージを受信したときの受信側のアクションは、メッセージの QoS レベルによって異なります。

QoS 0
: 関心のあるパーティーに対し、メッセージを使用可能にします。

QoS 1
: メッセージを永続ストレージに記録し、関心のあるパーティーに対して直ちに使用可能にして、送信側に PUBACK メッセージを返します。

QoS 2
: メッセージを永続ストレージに記録し、関心のあるパーティーに対して直ちには使用可能とせずに、送信側に PUBREC メッセージを返します。

サーバーがメッセージを受信した場合、関心のあるパーティーは PUBLISH メッセージのトピックへのサブスクライバーとなります。サブスクライバーがメッセージを受信した場合、関心のあるパーティーは 1 つ以上のトピックにサブスクライブし、サーバーからのメッセージを待機しているクライアント上のアプリケーションとなります。

詳細は、『Quality of Service のレベルおよびフロー』を参照してください。

サーバー実装で、クライアントによる PUBLISH の作成が許可されない場合は、そのクライアントへの通知方法はありません。そのため、サーバー実装は通常の QoS 規則に従って肯定応答を行う必要があります。クライアントはメッセージのパブリッシュが許可されなかったことを通知され*ません*。

3.4. PUBACK - パブリッシュ確認応答

PUBACK メッセージは、QoS レベル 1 の PUBLISH メッセージへの応答です。PUBACK メッセージは、パブリッシュ側クライアントからの PUBLISH メッセージへの応答ではサーバーが、サーバーからの PUBLISH メッセージへの応答ではサブスクライバーが送信します。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (4)				DUP フラグ	QoS レベル		RETAIN
	0	1	0	0	x	x	x	x
バイト 2	残りの長さ (2)							
	0	0	0	0	0	0	1	0

QoS レベル
: 使用されません。

DUP フラグ
: 使用されません。

RETAIN フラグ
: 使用されません。

「残りの長さ」フィールド
: これは、可変ヘッダーの長さです (2 バイト)。マルチバイト・フィールドとすることができます。

可変ヘッダー

応答される PUBLISH メッセージのメッセージ識別子 (メッセージ ID) が含まれます。
以下の表に、可変ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ ID MSB							
バイト 2	メッセージ ID LSB							

ペイロード

ペイロードはありません。

アクション

クライアントは、PUBACK メッセージを受信すると元のメッセージを破棄します。これは、このメッセージがサーバーでも受信 (および記録) されるためです。

3.5. PUBREC - 保証されたパブリッシュの受信 (パート 1)

PUBREC メッセージは、QoS レベル 2 の PUBLISH メッセージへの応答です。また、QoS レベル 2 のプロトコル・フローの 2 番目のメッセージです。PUBREC メッセージは、パブリッシュ側クライアントからの PUBLISH メッセージへの応答ではサーバーが、サーバーからの PUBLISH メッセージへの応答ではサブスクライバーが送信します。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (5)				DUP フラグ	QoS レベル		RETAIN
	0	1	0	1	x	x	x	x
バイト 2	残りの長さ (2)							
	0	0	0	0	0	0	1	0

QoS レベル
　　使用されません。
DUP フラグ
　　使用されません。
RETAIN フラグ
　　使用されません。
「残りの長さ」フィールド
　　可変ヘッダーの長さです (2 バイト)。マルチバイト・フィールドとすることができます。

可変ヘッダー

可変ヘッダーには、応答される PUBLISH のメッセージ ID が含まれます。以下の表に、可変ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ ID MSB							
バイト 2	メッセージ ID LSB							

ペイロード

ペイロードはありません。

アクション

受信側は、PUBREC メッセージを受信すると、PUBREC メッセージと同じメッセージ ID で送信側に PUBREL メッセージを送信します。

3.6. PUBREL - 保証されたパブリッシュのリリース (パート 2)

PUBREL メッセージは、サーバーの PUBREC メッセージに対するパブリッシャーからの応答か、またはサブスクライバーの PUBREC メッセージに対するサーバーからの応答です。QoS 2 プロトコル・フローの 3 番目のメッセージです。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (6)				DUP フラグ	QoS レベル		RETAIN
	0	1	1	0	0	0	1	x
バイト 2	残りの長さ (2)							
	0	0	0	0	0	0	1	0

QoS レベル

PUBREL メッセージでは、PUBCOMP のフォーマットでの応答が予期されるため、QoS レベル 1 を使用します。再試行は、PUBLISH メッセージと同じ方法で処理されます。

DUP フラグ

ゼロ (0) に設定します。これは、メッセージが初めて送信されていることを示します。詳細は、『DUP』を参照してください。

RETAIN フラグ

使用されません。

「残りの長さ」フィールド

可変ヘッダーの長さです (2 バイト)。マルチバイト・フィールドとすることができます。

可変ヘッダー

可変ヘッダーには、応答される PUBREC メッセージと同じメッセージ ID が含まれます。以下の表に、可変ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ ID MSB							
バイト 2	メッセージ ID LSB							

ペイロード

ペイロードはありません。

アクション

サーバーは、パブリッシャーから PUBREL メッセージを受信すると、元のメッセージを関心のあるサブスクライバーに対して使用可能にし、同じメッセージ ID で PUBCOMP メッセージをパブリッシャーに送信します。サブスクライバーは、サーバーから PUBREL メッセージを受信すると、メッセージをサブスクライブ側アプリケーションに対して使用可能にし、PUBCOMP メッセージをサーバーに送信します。

3.7. PUBCOMP - 保証されたパブリッシュの完了 (パート 3)

このメッセージは、パブリッシャーの PUBREL メッセージに対するサーバーからの応答か、またはサーバーの PUBREL メッセージに対するサブスクライバーからの応答です。QoS 2 プロトコル・フローの 4 番目および最後のメッセージです。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (7)				DUP フラグ	QoS レベル		RETAIN
	0	1	1	1	x	x	x	X
バイト 2	残りの長さ (2)							
	0	0	0	0	0	0	1	0

QoS レベル
: 使用されません。

DUP フラグ
: 使用されません。

RETAIN フラグ
: 使用されません。

「残りの長さ」フィールド
: 可変ヘッダーの長さです (2 バイト)。マルチバイト・フィールドとすることができます。

可変ヘッダー

可変ヘッダーには、応答される PUBREL メッセージと同じメッセージ ID が含まれます。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ ID MSB							
バイト 2	メッセージ ID LSB							

ペイロード

ペイロードはありません。

アクション

クライアントは、PUBCOMP メッセージを受信すると、元のメッセージを破棄します。これは、このメッセージが既に「正確に 1 回」サーバーに送信されているためです。

3.8. SUBSCRIBE - 指定トピックのサブスクライブ

SUBSCRIBE メッセージにより、クライアントは 1 つ以上のトピック名への関心をサーバーに登録できます。これらのトピックにパブリッシュされたメッセージは、サーバーからクライアントに PUBLISH メッセージとして送信されます。SUBSCRIBE メッセージには、サブスクライバーがパブリッシュされたメッセージを受信する際に希望する QoS レベルも指定されます。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (8)				DUP フラグ	QoS レベル		RETAIN
	1	0	0	0	0	0	1	x
バイト 2	残りの長さ							

QoS レベル

SUBSCRIBE メッセージでは、 複数のサブスクリプション要求に応答する場合に QoS レベル 1 が使用されます。対応する SUBACK メッセージは、メッセージ ID と突き合わせることで識別されます。再試行は、PUBLISH メッセージと同じ方法で処理されます。

DUP フラグ

ゼロ (0) に設定します。これは、メッセージが初めて送信されていることを示します。詳細は、『DUP』を参照してください。

RETAIN フラグ

使用されません。

「残りの長さ」フィールド

ペイロードの長さです。マルチバイト・フィールドとすることができます。

可変ヘッダー

SUBSCRIBE メッセージは QoS レベル 1 であるため、可変ヘッダーにメッセージ ID が含まれます。詳細は、『メッセージ識別子』を参照してください。

以下の表に、メッセージ ID が 10 の可変ヘッダーのフォーマットの例を示します。

	説明	7	6	5	4	3	2	1	0
メッセージ識別子									
バイト 1	メッセージ ID MSB (0)	0	0	0	0	0	0	0	0
バイト 2	メッセージ ID LSB (10)	0	0	0	0	1	0	1	0

ペイロード

SUBSCRIBE メッセージのペイロードには、クライアントがサブスクライブするトピック名のリストと、クライアントがメッセージの受信において希望する QoS レベルが含まれます。これらのストリングは UTF エンコードされており、QoS レベルは 1 バイトのうちの 2 ビットを占めます。トピック・ストリングには、一連のトピックを表す特別なトピック・ワイルドカード文字を含めることができます。これらのトピック/QoS のペアは、以下の表のペイロードの例で示すとおり、連続してパックされています。

トピック名	「a/b」
要求された QoS	1
トピック名	「c/d」
要求された QoS	2

SUBSCRIBE メッセージ内のトピック名は短縮されません。

以下の表に、ペイロードの例のフォーマットを示します。

	説明	7	6	5	4	3	2	1	0
トピック名									
バイト 1	長さ MSB (0)	0	0	0	0	0	0	0	0
バイト 2	長さ LSB (3)	0	0	0	0	0	0	1	1
バイト 3	「a」(0x61)	0	1	1	0	0	0	0	1
バイト 4	「/」(0x2F)	0	0	1	0	1	1	1	1
バイト 5	「b」(0x62)	0	1	1	0	0	0	1	0
要求された QoS									
バイト 6	要求された QoS (1)	x	x	x	x	x	x	0	1
トピック名									

バイト 7	長さ MSB (0)	0	0	0	0	0	0	0	0
バイト 8	長さ LSB (3)	0	0	0	0	0	0	1	1
バイト 9	「c」(0x63)	0	1	1	0	0	0	1	1
バイト 10	「/」(0x2F)	0	0	1	0	1	1	1	1
バイト 11	「d」(0x64)	0	1	1	0	0	1	0	0
要求された QoS									
バイト 12	要求された QoS (2)	x	x	x	x	x	x	1	0

要求された QoS レベルが許可されると、クライアントはパブリッシャーからのメッセージの QoS レベルに応じて、許可された QoS レベルまたはそれ以下で PUBLISH メッセージを受信します。例えば、特定のトピックに対する QoS レベル 1 のサブスクリプションがクライアントに存在する場合、そのトピックに対する QoS レベル 0 の PUBLISH メッセージは、QoS レベル 0 でクライアントに送信されます。 同じトピックに対する QoS レベル 2 の PUBLISH メッセージは、クライアントに送信するために QoS レベル 1 にダウングレードされます。

その結果、QoS レベル 2 でのトピックのサブスクライブは、「このトピックに関するメッセージはパブリッシュ時と同じ QoS で受け取る必要がある」という意味になります。

つまり、パブリッシャーはメッセージを送信できる最大 QoS を決定でき、サブスクライバーは QoS をより使用に適したものにダウングレードできます。メッセージの QoS をアップグレードすることはできません。

「要求された QoS」フィールドは、以下の表で示すとおり、UTF エンコードされた各トピック名に続くバイトにエンコードされます。

ビット	7	6	5	4	3	2	1	0
	予約済み	予約済み	予約済み	予約済み	予約済み	予約済み	QoS レベル	
	x	x	x	x	x	x		

このバイトの上位 6 ビットは、現行バージョンのプロトコルでは使用されません。それらは将来の使用のために予約されています。

両方の QoS レベル・ビット・セットが含まれる要求は無効なパケットと見なし、接続を閉じるべきです。

応答

サーバーは、クライアントから SUBSCRIBE メッセージを受信すると、SUBACK メッセージで応答します。

サーバーは、クライアントが SUBACK メッセージを受信する前に、サブスクリプション に起因する PUBLISH メッセージの送信を開始することができます。

サーバー実装でクライアントによる SUBSCRIBE 要求の実行が許可されない場合は、そのクライアントに通知する方法はありません。そのため、サーバー実装は SUBACK で肯定応答を行う必要があります。クライアントはサブスクライブが許可されなかったことを通知*されません*。

サーバーは、クライアントが要求したよりも低いレベルの QoS を許可することを選択する場合があります。これは、サーバーがより高いレベルの QoS を提供できない場合に発生します。例えば、サーバーは、信頼できるパーシスタンス機構を備えていない場合、QoS 0 でのサブスクリプションのみの許可を選択します。

3.9. SUBACK - サブスクリプション確認応答

SUBACK メッセージは、サーバーが SUBSCRIBE メッセージの受信を確認するためにクライアントに対して送信します。

SUBACK メッセージには、許可された QoS レベルのリストが含まれます。SUBACK メッセージ内の許可された QoS レベルの順序は、対応する SUBSCRIBE メッセージ内のトピック名の順序と一致します。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (9)				DUP フラグ	QoS レベル		RETAIN
	1	0	0	1	x	x	x	x
バイト 2	残りの長さ							

QoS レベル
: 使用されません。

DUP フラグ
: 使用されません。

RETAIN フラグ
: 使用されません。

「残りの長さ」フィールド
: ペイロードの長さです。マルチバイト・フィールドとすることができます。

可変ヘッダー

可変ヘッダーには、応答される SUBSCRIBE メッセージのメッセージ ID が含まれます。以下の表に、可変ヘッダーのフォーマットを示します。

	7	6	5	4	3	2	1	0
バイト 1	メッセージ ID MSB							
バイト 2	メッセージ ID LSB							

ペイロード

ペイロードには、許可された QoS レベルのベクトルが含まれます。各レベルは、対応する SUBSCRIBE メッセージ内のトピック名に対応します。SUBACK メッセージ内の QoS レベルの順序は、SUBSCRIBE メッセージ内のトピック名/要求された QoS のペアの順序と一致します。可変ヘッダー内のメッセージ ID により、SUBACK メッセージと対応する SUBSCRIBE メッセージを突き合わせできます。

以下の表に、バイトにエンコードされた「許可された QoS」フィールドを示します。

ビット	7	6	5	4	3	2	1	0
	予約済み	予約済み	予約済み	予約済み	予約済み	予約済み	QoS レベル	
	x	x	x	x	x	x		

このバイトの上位 6 ビットは、現行バージョンのプロトコルでは使用されません。それらは将来の使用のために予約されています。

以下の表に、ペイロードの例を示します。

許可された QoS	0
許可された QoS	2

以下の表に、このペイロードのフォーマットを示します。

	説明	7	6	5	4	3	2	1	0
バイト 1	許可された QoS (0)	x	x	x	x	x	x	0	0
バイト 1	許可された QoS (2)	x	x	x	x	x	x	1	0

3.10. UNSUBSCRIBE - 指定されたトピックからのアンサブスクライブ

UNSUBSCRIBE メッセージは、クライアントが指定されたトピックからのアンサブスクライブのためにサーバーに対して送信します。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットの例を示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (10)				DUP フラグ	QoS レベル		RETAIN
	1	0	1	0	0	0	1	x
バイト 2	残りの長さ							

QoS レベル

UNSUBSCRIBE メッセージでは、複数のアンサブスクライブ要求に応答する場合に QoS レベル 1 が使用されます。対応する UNSUBACK メッセージは、メッセージ ID によって識別されます。再試行は、PUBLISH メッセージと同じ方法で処理されます。

DUP フラグ

ゼロ (0) に設定します。これは、メッセージが初めて送信されていることを示します。詳細は、『DUP』を参照してください。

RETAIN フラグ

使用されません。

残りの長さ

これはペイロードの長さです。マルチバイト・フィールドとすることができます。

可変ヘッダー

UNSUBSCRIBE メッセージは QoS レベル 1 であるため、可変ヘッダーにメッセージ ID が含まれます。詳細は、『メッセージ識別子』を参照してください。

以下の表に、メッセージ ID が 10 の可変ヘッダーのフォーマットの例を示します。

	説明	7	6	5	4	3	2	1	0
メッセージ識別子									
バイト 1	メッセージ ID MSB (0)	0	0	0	0	0	0	0	0
バイト 2	メッセージ ID LSB (10)	0	0	0	0	1	0	1	0

ペイロード

クライアントは、ペイロード内で指定されたトピックのリストからアンサブスクライブします。ストリングは UTF エンコードされており、連続してパックされています。UNSUBSCRIBE メッセージ内のトピック名は短縮されません。以下の表に、ペイロードの例を示します。

トピック名	「a/b」
トピック名	「c/d」

以下の表に、このペイロードのフォーマットを示します。

	説明	7	6	5	4	3	2	1	0
トピック名									
バイト 1	長さ MSB (0)	0	0	0	0	0	0	0	0
バイト 2	長さ LSB (3)	0	0	0	0	0	0	1	1
バイト 3	「a」(0x61)	0	1	1	0	0	0	0	1
バイト 4	「/」(0x2F)	0	0	1	0	1	1	1	1
バイト 5	「b」(0x62)	0	1	1	0	0	0	1	0
トピック名									
バイト 6	長さ MSB (0)	0	0	0	0	0	0	0	0
バイト 7	長さ LSB (3)	0	0	0	0	0	0	1	1
バイト 8	「c」(0x63)	0	1	1	0	0	0	1	1
バイト 9	「/」(0x2F)	0	0	1	0	1	1	1	1
バイト 10	「d」(0x64)	0	1	1	0	0	1	0	0

応答

サーバーは、UNSUBSCRIBE メッセージへの応答でクライアントに UNSUBACK を送信します。

3.11. UNSUBACK - アンサブスクライブ確認応答

UNSUBACK メッセージは、サーバーが UNSUBSCRIBE メッセージの受信を確認するためにクライアントに対して送信します。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (11)				DUP フラグ	QoS レベル		RETAIN
	1	0	1	1	x	x	x	x
バイト 2	残りの長さ (2)							
	0	0	0	0	0	0	1	0

QoS レベル

使用されません。

DUP フラグ

使用されません。

RETAIN フラグ

使用されません。

残りの長さ

可変ヘッダーの長さです (2 バイト)。

可変ヘッダー

可変ヘッダーには、応答される UNSUBSCRIBE メッセージのメッセージ ID が含まれます。以下の表に、可変ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ ID MSB							
バイト 2	メッセージ ID LSB							

ペイロード

ペイロードはありません。

3.12. PINGREQ - PING 要求

PINGREQ メッセージは、接続されているクライアントからサーバーに送信される「死活確認」メッセージです。

詳細は、『キープアライブ・タイマー』を参照してください。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (12)				DUP フラグ	QoS レベル		RETAIN
	1	1	0	0	x	x	x	x
バイト 2	残りの長さ (0)							
	0	0	0	0	0	0	0	0

DUP、QoS および RETAIN フラグは使用されません。

可変ヘッダー

可変ヘッダーはありません。

ペイロード

ペイロードはありません。

応答

PINGREQ メッセージへの応答は PINGRESP メッセージです。

3.13. PINGRESP - PING 応答

PINGRESP メッセージは、PINGREQ メッセージに対してサーバーが送信する応答で、「活動中である」ことを示します。

詳細は、『キープアライブ・タイマー』を参照してください。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (13)				DUP フラグ	QoS レベル		RETAIN
	1	1	0	1	x	x	x	x
バイト 2	残りの長さ (0)							
	0	0	0	0	0	0	0	0

DUP、QoS および RETAIN フラグは使用されません。

ペイロード

ペイロードはありません。

可変ヘッダー

可変ヘッダーはありません。

3.14. DISCONNECT - 切断通知

DISCONNECT メッセージは、クライアントが TCP/IP 接続のクローズを示すためにサーバーに対して送信します。これにより、回線のドロップだけではなく、クリーンな切断が可能になります。

クライアントが Clean session フラグを設定して接続している場合、以前保持していたクライアントについての情報はすべて破棄されます。

DISCONNECT の受信後は、サーバーはクライアントに依存せずに TCP/IP 接続を閉じるべきです。

固定ヘッダー

以下の表に、固定ヘッダーのフォーマットを示します。

ビット	7	6	5	4	3	2	1	0
バイト 1	メッセージ・タイプ (14)				DUP フラグ	QoS レベル		RETAIN
	1	1	1	0	x	x	x	x
バイト 2	残りの長さ (0)							
	0	0	0	0	0	0	0	0

DUP、QoS および RETAIN フラグは、DISCONNECT メッセージでは使用されません。

ペイロード

ペイロードはありません。

可変ヘッダー

可変ヘッダーはありません。

4. フロー

4.1. Quality of Service レベルおよびフロー

MQTT は、Quality of Service (QoS) で定義されたレベルに従ってメッセージを送信します。レベルについて以下に説明します。

QoS レベル 0:最高 1 回 (At most once) の送信

メッセージは、基となる TCP/IP ネットワークのベスト・エフォートに従って送信されます。応答は予期されないため、このプロトコルには再試行のセマンティクスが定義されていません。メッセージは 1 回サーバーに到着するか、まったく到着しないかのどちらかです。

以下の表に、QoS レベル 0 のプロトコル・フローを示します。

クライアント	メッセージおよび方向	サーバー
QoS = 0	PUBLISH ---------->	アクション: サブスクライバーにメッセージをパブリッシュします

QoS レベル 1:最低 1 回 (At least once) の送信

サーバーはメッセージを受信すると、PUBACK メッセージで応答します。通信リンクか送信側デバイスのどちらかで障害が識別された場合、または特定時間の経過後に確認応答メッセージを受信しない場合、送信側はメッセージ・ヘッダーに DUP ビットを設定してメッセージを再送信します。このメッセージは最低 1 回サーバーに到着します。SUBSCRIBE メッセージと UNSUBSCRIBE メッセージの両方が QoS レベル 1 を使用します。

QoS レベル 1 のメッセージでは、メッセージ・ヘッダーにメッセージ ID が含まれます。

以下の表に、QoS レベル 1 のプロトコル・フローを示します。

クライアント	メッセージおよび方向	サーバー
QoS = 1 DUP = 0 メッセージ ID = x アクション: メッセージを保存します	PUBLISH ---------->	アクション ・ メッセージを保存します ・ サブスクライバーにメッセージをパブリッシュします ・ メッセージを削除します
アクション: メッセージを破棄します	PUBACK <----------	

クライアントは、PUBACK メッセージを受信しない場合 (アプリケーションで定義された時間内に受信しない場合、または障害が検出されて通信セッションがリスタートされた場合など)、DUP フラグを設定して PUBLISH メッセージを再送信できます。

重複するメッセージをクライアントから受信した場合、サーバーは、サブスクライバーにメッセージをリパブリッシュし、別の PUBACK メッセージを送信します。

QoS レベル 2:正確に 1 回 (Exactly once) の送信

QoS レベル 1 に加えてさらに追加のプロトコル・フローによって、受信側アプリケーションに重複メッセージが送信されないよう保証します。これは送信の最高レベルで、重複メッセージが許容されない場合に使用します。ネットワーク・

トラフィックは増加しますが、メッセージの内容が重要であるため、通常は許容されます。

QoS レベル 2 のメッセージでは、メッセージ・ヘッダーにメッセージ ID が含まれます。

以下の表に、QoS レベル 2 のプロトコル・フローを示します。受信側において行われるべき PUBLISH フローの処理方法には、2 つのセマンティクスがあります。それらのセマンティクスは、サブスクライバーがフロー内のどの段階でメッセージを使用できるようになるかに影響します。セマンティクスの選択は実装に固有のものですが、QoS レベル 2 のフローの保証には影響しません。

クライアント	メッセージおよび方向	サーバー
QoS = 2 DUP = 0 メッセージ ID = x **アクション:** メッセージを保存します	PUBLISH ---------->	**アクション:** メッセージを保存します *または* **アクション** ・ メッセージ ID を保存します ・ サブスクライバーにメッセージをパブリッシュします
	PUBREC <----------	メッセージ ID = x
メッセージ ID = x	PUBREL ---------->	**アクション** ・ サブスクライバーにメッセージをパブリッシュします ・ メッセージを削除します *または* **アクション:** メッセージ ID を削除します
アクション: メッセージを破棄します	PUBCOMP <----------	メッセージ ID = x

障害が検出された場合、または特定時間が経過した後は、最後の無応答のプロトコル・メッセージ (PUBLISH または PUBREL のどちらか) からプロトコル・フローが再試行されます。詳細は、『メッセージ送信の再試行』を参照してください。追加のプロトコル・フローによって、メッセージがサブスクライバーに 1 度のみ送信されることが保証されます。

QoS レベル 1 および 2 での仮定

どのようなネットワークでも、デバイスや通信リンクに障害が発生する可能性があります。障害が発生した場合、リンクの一端がもう一端で何が起こっているかを把握できない可能性があります。これを*未確定期間 (in doubt window)* といいます。このような場合、メッセージ送信に関与するデバイスやネットワークの信頼性について仮定する必要があります。

MQTT は、クライアントおよびサーバーは一般的に信頼性が高く、通信チャネルは信頼性が低い傾向があると仮定します。クライアント・デバイスに障害が発生した場合、通常それは一時的な障害ではなく、壊滅的な障害です。デバイスからデータを復旧できる可能性は低くなります。一部のデバイスは、フラッシュ ROM などの不揮発性ストレージを備えています。クライアント・デバイス上に永続性の高いストレージを組み込むことで、最も重要なデータを一部の障害モードから保護できます。

通信リンクの基本的な障害の域を越えると、障害モードのマトリックスは複雑になり、その結果 MQTT の仕様では処理しきれないケースが発生するようになります。

4.2. メッセージ送信の再試行

通常、TCP によりパケットの送信が保証されますが、MQTT メッセージが受信されない特定のケースがあります。応答 (QoS >0 PUBLISH、PUBREL、SUBSCRIBE、UNSUBSCRIBE) を予期する MQTT メッセージの場合、特定の時間内に応答が受信されないと、送信側が送信を再試行します。送信側はメッセージに DUP フラグを設定するべきです。

再試行タイムアウトは構成可能なオプションとするべきです。ただし、送信されている間にメッセージ送信がタイムアウトしないように注意を払う必要があります。例えば、低速ネットワークでの大規模メッセージの送信は、高速ネットワークでの小規模メッセージの送信よりも当然時間がかかります。タイムアウトしたメッセージを繰り返し再試行すると、問題が悪化する場合があります。そのため、複数の再試行が行われた際にタイムアウト値を大きくするなどの戦術を採用するべきです。

クライアントの再接続時にクリーン・セッションにマークが付けられていない場合、クライアントとサーバーの両方から以前の未完了メッセージがすべて再送信されます。

クライアントは、この「再接続時」の再試行動作以外は、メッセージ送信を再試行する必要はありません。ただし、ブローカーはすべての無応答メッセージを再試行するべきです。

4.3. メッセージの順序付け

メッセージの順序付けは、クライアントが未完了 PUBLISH フローをいくつ許可するか、クライアントが単一スレッドかマルチスレッドかといった多数の要因によって影響を受ける可能性があります。議論を進めるために、パケットがネットワークに対して読み取り/書き込みされる時点でクライアントは単一スレッドであると仮定します。

メッセージの順序付けに関して保証する実装の場合、メッセージ送信フローの各段階を開始された順序で完了する必要があります。例えば、QoS レベル 2 の一連のフローでは、PUBREL フローを元の PUBLISH フローと同じ順序で送信する必要があります。

クライアント	メッセージおよび方向	サーバー
	PUBLISH 1 ---------->	
	PUBLISH 2 ---------->	
	PUBLISH 3 ---------->	
	PUBREC 1 <----------	
	PUBREC 2 <----------	
	PUBREL 1 ---------->	
	PUBREC 3 <----------	
	PUBREL 2 ---------->	
	PUBCOMP 1 <----------	
	PUBREL 3 ---------->	
	PUBCOMP 2 <----------	
	PUBCOMP 3 <----------	

また、許可される未完了メッセージの数は、可能な保証のタイプに影響を及ぼします。

- 並行稼働数 1 では、各送信フローは次のフローが開始する前に完了します。これにより、サブミットされた順序でのメッセージの送信が保証されます。
- 1 より大きな並行稼働数では、QoS レベル内でのみメッセージの順序付けを保証できます。

付録 A - トピック・ワイルドカード

サブスクリプションには特殊文字を含めることができます。これにより、一度に複数のトピックへのサブスクライブが可能となります。

トピック・レベル分離文字は、トピックに構造を取り入れるために使用され、この目的のためにトピック内に指定することができます。サブスクリプションではマルチレベル・ワイルドカードと単一レベル・ワイルドカードが使用できますが、メッセージのパブリッシャーがそれらをトピック内で使用することはできません。

トピック・レベル分離文字
: スラッシュ (/) は、トピック・ツリー内の各レベルを分離するために使用され、トピック・スペースに階層構造をもたらします。2 つのワイルドカード文字がサブスクライバーによって指定されたトピック内に存在する場合、トピック・レベル分離文字の使用方法が重要となります。

マルチレベル・ワイルドカード

: 番号記号 (#) は、トピック内において任意の数のレベルに一致するワイルドカード文字です。例えば、finance/stock/ibm/# とサブスクライブする場合、以下のトピックに関するメッセージを受け取ります。

```
    finance/stock/ibm
finance/stock/ibm/closingprice
finance/stock/ibm/currentprice
```

: マルチレベル・ワイルドカードは、ゼロ以上のレベルを表すことができます。したがって、*finance/#* は *finance* のみとも一致します。この場合、*#* はゼロ・レベルを表します。この状況では、分離するレベルがないため、トピック・レベル分離文字は無意味となります。

: マルチレベル・ワイルドカードは、単体で指定するか、またはトピック・レベル分離文字の横にのみ指定できます。したがって、*#* および *finance/#* はどちらも有効ですが、*finance#* は無効です。マルチレベル・ワイルドカードは、トピック・ツリー内で使用される最後の文字とする必要があります。例えば、*finance/#* は有効ですが、*finance/#/closingprice* は無効です。

単一レベル・ワイルドカード

: 正符号 (+) は、1 つのトピック・レベルにのみ一致するワイルドカード文字です。例えば、*finance/stock/+* は、*finance/stock/ibm* および *finance/stock/xyz* と一致しますが、*finance/stock/ibm/closingprice* とは一致しません。また、単一

レベル・ワイルドカードは単一レベルにのみ一致するため、*finance/+* は *finance* とは一致しません。

単一レベル・ワイルドカードは、マルチレベル・ワイルドカードとともに、トピック・ツリー内のどのレベルでも使用できます。単体で指定される場合以外は、トピック・レベル分離文字の横で使用する必要があります。したがって、*+* および *finance/+* はどちらも有効ですが、*finance+* は無効です。単一レベル・ワイルドカードは、トピック・ツリーの最後またはトピック・ツリー内で使用できます。例えば、*finance/+* および *finance/+/ibm* はどちらも有効です。

トピックのセマンティクスおよび使用法

アプリケーションを構築する際は、トピック・ツリーの設計で、トピック名の構文とセマンティクスの以下の原則を考慮する必要があります。

- トピックは 1 文字以上の長さにする必要があります。
- トピック名では大/小文字が区別されます。例えば、*ACCOUNTS* と *Accounts* は、2 つの異なるトピックです。
- トピック名にはスペース文字を含めることができます。例えば、*Accounts payable* は有効なトピックです。
- 先頭に「/」を付けると、異なるトピックとなります。例えば、*/finance* は *finance* とは異なります。*/finance* は、「+/+」および「/+」と一致しますが、「+」とは一致しません。
- トピックにはヌル文字 (Unicode \x0000) は含めないでください。

以下の原則は、トピック・ツリーの構造および内容に適用されます。

- 長さは 64k までに制限されますが、その範囲内であれば、トピック・ツリー内のレベル数に制限はありません。
- ルート・ノードの数に制限はありません。つまり、トピック・ツリーはいくつでも存在可能です。