基礎から学ぶWindowsネットワーク

第15回　信頼性のある通信を実現するTCPプロトコル（2）

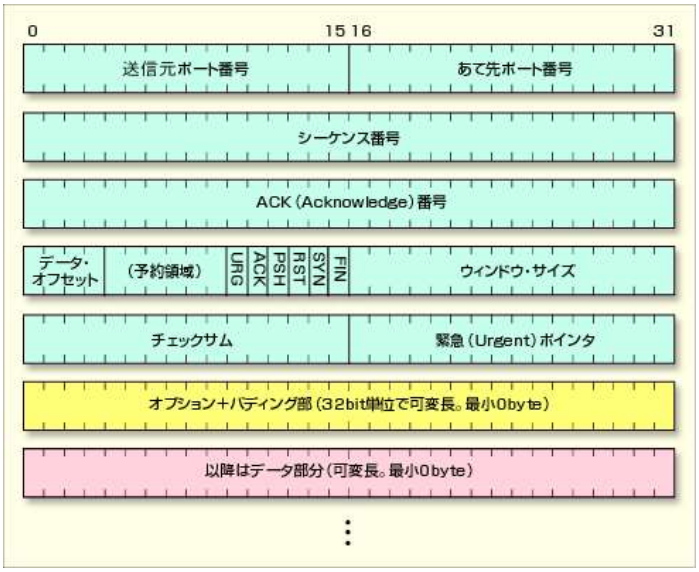
（2/3 ページ）

2004年01月29日 00時00分 公開

[デジタルアドバンテージ，著]

前のページへ 1 2 3 次のページへ

TCPプロトコルの詳細やパケットの構造などは、RFC793（STD0007）で定義されている。以下にTCPのヘッダ部分の詳細構造を示しておく。



TCPヘッダの構造

TCPでは信頼性の高い通信を実現するために、受信確認やスライディング・ウィンドウ制御、そしてさまざまな付加機能などを用意している。そのためUDPよりも複雑なヘッダ情報を持っている。「チェックサム」はIPヘッダなどと同様に、1の補数で計算する。

連載第7回「データグラム通信を実現するUDPプロトコル―2．UDPパケットの構造」で示したUDPパケットの構造と比べると、非常に複雑になっている。UDPでは、通信に先立ってコネクションを確立する必要のないデータグラム型通信モデルを使用しているため、送信される各UDPパケットは完全に独立していた。そのため、UDPパケットごとにあて先のポート番号（送信元を区別するための送信元ポート番号）さえあれば、相手にパケットを届けることができる。

だがTCPでは、通信に先立ってコネクションを開設し、さらに通信中にも、前回解説したシーケンス番号に基づいた送受信の確認やウィンドウ制御なども行っている。そのため、通信のたびにシーケンス番号やACK番号、ウィンドウ・サイズなどを渡す必要があるので、TCPヘッダの内容も複雑になっている。以下、各フィールドについて順番に見ていこう。

「送信元ポート番号」フィールド：16bit幅

これは、TCPパケットの送信元のアプリケーションを識別するための番号である。UDPの場合と同様に、送信元とあて先のポート番号、および送信元とあて先のIPアドレス（これはIPパケット中から抽出する）の4つの番号の組によって、TCPのコネクションを識別する。パケットの返信時には、送信元とあて先のポート番号（およびIPアドレス・フィールド）を入れ替えて、送信する。

UDPの場合と違って、送信元ポート番号を0にすることはできない。TCPでは必ず双方向に通信する必要があるため、逆方向のTCPパケットにも、あて先となるポート番号が必ず必要となる。

「あて先ポート番号」フィールド：16bit幅

UDPの場合と同様に、あて先となるアプリケーションが待ち受けしているポートの番号を表す。16bit幅なので1～65535まで利用できるが（0は予約済み）、目的別に利用可能な範囲が決められている。UDPの場合と同様であるが、以下に範囲別の用途を再掲しておく。

範囲	意味
1～1023	Well Known Port（WKS、ウェル・ノウン・ポート）。特権ユーザーや管理者モードで動作するサービスが利用するポート。直訳して「よく知られたポート」と呼ばれることもある
1024～49151	Registered Port（登録済みポート）。登録されたサービスが利用するポート
49152～65535	Dynamic Port／Private Port。動的なアプリケーションなどで利用するポート

UDP／TCPにおけるポート番号
UDP（およびTCP）では、16bit幅のポート番号が利用できるが、用途に応じて利用可能な範囲が決められている。OS標準のサービスはWKSのポートを利用し、ユーザー・アプリケーションはそれ以外のポートを利用することが望ましいとされている。

1023番以下のポート番号は特権ポートであり、特権ユーザーや管理者モードで動作するサービスが利用するポートとされている。簡単にいうと、システムが標準的に提供するような、公共性／有用性が高いサービスが利用し、ユーザーが作成したプログラムなどでは1024以上のポート番号を利用することになっている。代表的なところでは、HTTPは80番、POPメール・サーバでは110番、SMTPメール・サーバでは25番などを使っている。

「シーケンス番号」フィールド：32bit幅

送信するデータ（バイト・データ）に対して、順序付けを行うための「シーケンス番号」を指定するフィールド。送信するデータ1byteごとに、シーケンス番号を1つずつ昇順に割り当て、どこまでデータを送信したかを指定する。このフィールドは32bit幅しかないので、2の32乗＝4Gbytes分送信すると、また同じ番号に戻ってくる（ラップアラウンドする）ことになる。初期値は0とは限らず、ランダムな値がセットされる。このフィールドは常に有効である。

シーケンス番号は、TCPデータの送信側で管理されており、データを送信するたびに、送信したデータのバイト数分だけシーケンス番号が加算され、TCPパケットに入れて送信される。データ部分の最初のバイト位置が、このシーケンス番号に一致する。例えばTCPデータとして10bytesのデータが格納されているとすると、データの先頭バイトがシーケンス番号「n」の位置に相当し、10bytes目はシーケンス番号「n+9」の位置に相当する。

「ACK番号」フィールド：32bit幅

これは受信したデータに対して、どこのバイト位置までを受信したかを表すフィールドである。ACK（Acknowledge）は「承認」という意味。シーケンス番号と同様に32bit幅なので、4Gbytesごとにラップアラウンドすることになる。ACK番号は、データを受信した側が、どこまで受信したかを示すために、応答TCPパケットにセットして送

信する。後述するACKフラグがオンの場合にのみ、このACK番号フィールドが有効となる。

このフィールドは、受信したデータのシーケンス番号に対応しており、受信が完了したデータ位置のシーケンス番号＋1を返すことになっている。例えば受信したTCPパケット中にデータが10bytes含まれているとすると、受信したパケットのシーケンス番号nに10を加えた「n＋10」がACK番号として返される。これは、シーケンス番号「n」から「n+9」まではすべて受信が完了したという意味である。

「データ・オフセット」フィールド：4bit幅

TCPデータが始まる位置を表すフィールド。「データ・オフセット」という名称になっているが、TCPヘッダの直後にデータ部が続いているため、TCPヘッダのサイズを表していると考えてもよい。このフィールドは4bitしかないため、0〜15しか表すことができないが、IPヘッダの場合と同様に、1ワード＝32bit（4bytes）単位で数えることになっている。よって、TCPヘッダ・サイズは最大で15×4＝60bytesまでとなる。この図からも分かるように、TCPヘッダの最小サイズは最低でも20bytesなので、このフィールドの最小値は5（2進数でいうと0101）となる（5×4＝20bytes）。

「URG（urgent）フラグ」フィールド：1bit幅

このURGからFINまでは、それぞれが1bitのフラグ・フィールドである。デフォルトではすべて初期値が0であるが、1になるとそれぞれのフラグの意味が有効（オン）になる。

「URG（urgent、緊急）」フラグは、このTCPパケット中に「緊急データ」が含まれていることを表す。ただし実際に緊急データを使っているアプリケーションはほとんどない。

「ACKフラグ」フィールド：1bit幅

「ACK（acknowledge）」フラグがオンならば、TCPヘッダ中に有効なACK番号が含まれていることを表す。実際には、TCP接続確立時の一番最初に送信されるTCPパケットを除き、すべてのTCPパケットにおいてこのACKフラグがセットされている。つまり、最初のパケットを除き、すべてのパケットでACK番号フィールドが有効である（最初のパケットの場合はACK番号フィールドの内容は意味を持たないので、ACKフラグはオフになっている）。

「PSHフラグ」フィールド：1bit幅

「PSH（push）」フラグは、受信したデータをすみやかに上位アプリケーションに引き渡すように要求するためのフラグである。

TCP通信で送信されたデータは、まずは受信側の受信バッファに格納され、適当なタイミングで受信側の上位アプリケーションに渡される。受信したデータをすぐに上位アプリケーションに渡すのではなく、できるだけまとめてから受け渡した方が、受け渡しなどのオーバーヘッドが少なくなり、結果的に処理が効率よく行えるからである（と、TCP/IP規格の制定当初は考えられていた）。だがこのバッファリングを行うと、その代償としてアプリケーションの応答性が損なわれる可能性がある。例えば文字をインタラクティブに入出力させたいのに、バッファリングしてしまうと、応答が少し遅れたような感じになるかもしれない。

だがデータを送信する場合にPSHフラグもセットしておくと、受信したデータ（および受信バッファにたまっていたデータ）は直ちに上位アプリケーションへと引き渡され、応答性が向上する可能性がある。例えばTelnetでは、ユーザーの入力した文字を送信する場合にこのPSHフラグをセットしており、ユーザーの入力に素早く応答するようにしている。

ただしPSHフラグの設定に応じて実際に処理方法を変えるかどうかは実装依存である。現在の一般的なTCP/IP実装では、受信したデータはすみやかに上位アプリケーションに渡されるようになっており、PSHフラグのオン／オフには影響を受けないことが多い。

「RSTフラグ」フィールド：1bit幅

「RST（reset）」は、TCP接続を中断または拒否したい場合にセットされる。受信した側では、接続要求が拒否されたとみなし、現在のTCP接続を破棄または強制終了をしなければならない。

TCP接続がエラーなどで長く中断して、シーケンス番号とACK番号の整合性が取れなくなった場合、ACKを返す代わりにこのRSTフラグをセットしたTCPパケットを送信すると、現在のTCP接続を強制終了することができる。またTCP接続要求に対して、ACKではなくRSTを返すと、接続を拒否していることを表す。例えばサーバ側のリソースが不足していてオープン要求に応えられない場合や、許可されていないIPアドレスからの接続要求にに対して拒否したい場合に、このRSTパケットが返されることがある。

「SYNフラグ」フィールド：1bit幅

TCP接続を要求する場合は、この「SYN（synchronize）フラグ」をセットしたパケットを送信する。これによって、TCP接続のオープン処理が開始される。TCPは双方向通信路なので、双方から送信されるそれぞれの最初の接続要求パケットにはこのSYNフラグがセットされている。だが2番目以降のパケットにはセットされていない。

Synchronizeとは「同期する」という意味であるが、TCP接続の確立に伴い、双方のシーケンス番号とACK番号を同期させるということからこう呼ばれている。SYNフラグをがセットされたパケットを受信した場合、自身のACK番号を受信したシーケンス番号に同期させる。これにより、以後の通信ための準備が整う。

「FINフラグ」フィールド：1bit幅

「FIN（finis、終了）」フラグは、TCP接続を終了させるために利用される。FINフラグがセットされたパケットは、もうこれ以上データの受信が必要ないことを意味し、受信した側では終了処理を開始する。双方からFINが送られるとTCP接続が終了し、TCP接続のために用意されていた内部バッファなどのリソースが解放される。

「ウィンドウ・サイズ」フィールド：16bit幅

このフィールドは、受信側のウィンドウ・サイズを相手に伝えるために利用される。TCPの送信側では、相手から通知されたウィンドウ・サイズを見て、送信可能な最大のデータ量を判断している。値0は、データを受信することができないという意味であり、送信側に対してデータの送信を一時的に停止してほしいという意味になる。

ウィンドウ・サイズ・フィールドは16bit幅なので、最大では65,535bytesまでのウィンドウ・サイズを設定することができる。またRFC1323で定義されているTCPの拡張プロトコルを使うと、より大きなウィンドウ・サイズを利用することもできる（もちろん通信する双方がこの仕様を実装している必要がある）。詳細については次回解説する。

「チェックサム」フィールド：16bit幅

これはTCPパケットの整合性を検査するための検査用データを格納するフィールドである。計算方法は、UDPヘッダ中のチェックサムと同様に、「1の補数演算」を利用して計算する。ただし、チェックサム計算の対象となるデータは、「TCP擬似ヘッダ（12bytes）」と「TCPヘッダ（8bytes）」「TCPペイロード」の3つの部分からなる。

「TCP擬似ヘッダ（pseudo header）」とは、チェックサムの計算時だけに使われる仮想的なヘッダ・データであり、実際のTCPパケット中には含まれていない。具体的に

は、以下のような擬似ヘッダがTCPパケットの先頭に存在するものとして、これら全体を対象としてチェックサムが計算される。

オフセット	長さ	データ
0	4bytes	送信元IPアドレス
4	4bytes	あて先IPアドレス
8	1byte	0（ダミー・データ。未使用）
9	1byte	6（「6」は、IPヘッダ中において、TCPプロトコルを表すためのプロトコル番号）
11	2byte	パケット長（TCPヘッダも含めた長さ）

チェックサム計算のためのTCP擬似ヘッダ
TCPのチェックサムを計算する場合は、先頭にこの擬似的なヘッダが存在するものとして、TCPヘッダ、TCPペイロードとともに計算する。IPアドレスの情報はIPヘッダ中から抜き出してくる。ペイロード長が奇数の場合は、最後に1byteの「0」を補って計算する（この追加する1byteのデータは、パケット長には含めない）。

「緊急ポインタ」フィールド：16bit幅

TCPパケットに中に緊急データが含まれる場合、URGフラグをセットするとともに、この「緊急ポインタ」フィールドに、緊急データの場所（サイズ）を表す数値を指定する。ただしこのフィールドの値の解釈については、初期に発行されたRFC793の記述は間違いであり、RFC1122の記述が正しいとされているが、実際には両方の仕様に基づいたプロトコル・スタックの実装が混在している。具体的には、例えば緊急データのサイズが10bytesならば、RFC1122方式なら「9」を、RFC793方式なら「10」をそれぞれセットする。これは緊急データの終わりのバイト位置を指すか、それとも（緊急データではない）通常のデータ領域の先頭位置を指すかの違いである。

Windows OSの場合はデフォルトではRFC798仕様になっているが、この動作を変更するには、サポート技術情報の「Windows XP での TCP/IP と NBT の構成パラメータ」や「Microsoft Windows 2000 TCP/IP 実装詳細」などを参考にして、レジストリTcpUseRFC1122UrgentPointer の値を変更する必要がある。

「オプション」フィールド：32bit単位で可変長

このフィールドは、TCP接続における各種の特性を設定するために利用される。例えば次回解説するMSSやウィンドウ・サイズのスケーリング・オプションなどを、個々のTCP接続ごとに設定することができる。オプションはバイト単位で可変長であり、同時に複数のオプションを設定することができるが、最終的には32bitの倍数になるように、必要ならば最後にバイト・データの0が埋められる。

「データ」フィールド：可変長

TCPヘッダの直後には、（存在するなら）緊急データと、通常のTCPデータ部が続く。UDPの場合と違って、データを含まない、単なるTCPヘッダだけのパケットも多く使われる。単にウィンドウ・サイズを通知したり（フロー制御を行う）、キープ・アライブを通知したりするためである。キープ・アライブ（keep-alive）とは、データ通信が何も行われない場合でも、一定時間間隔で空のTCPパケットを送受信することにより、TCP接続がアクティブであることをお互いに通知、確認するための通信機能である。何も通信を行わないでいると、無通信で回線が切断されてしまったり、TCP接続がタイムアウトして切断されてしまったりするので、キープ・アライブでこれを防ぐことができる。

TCPのオープンとクローズ処理

前のページへ 1 | 2 | 3 次のページへ

基礎から学ぶWindowsネットワーク 連載一覧

全 23 回

新しい連載記事が 6 件あります

過去の連載記事が 12 件あります

この記事に関連する製品／サービスを比較（キーマンズネット）

既存のネットワーク構成とマッチする？『WAN高速化』製品の選び方
まずネットワークの性質を十分に見極めよう！『ネットワーク管理』製品比較
信頼性や可用性に対する取り組みは？『ネットワークスイッチ』製品比較
L4負荷分散とL7負荷分散どちらを重視？『ADC／ロードバランサ』製品一覧
構築したいネットワーク要件で大きく変わる『ルーター』の選び方