=== ch04.docx

[chap]4章　Wiresharkでのパケットキャプチャのテクニック

前章でWiresharkを学習したので、実際にパケットをキャプチャし、解析する準備が整いました。本章ではキャプチャしたファイル、パケット、そして時間の表示フォーマットの扱い方を学びます。またパケットをキャプチャするさらに高度なオプションを学び、フィルタについても学習します。

キャプチャファイルの扱い方

パケット解析は、パケットキャプチャと同時にできるわけではありません。通常、パケットをキャプチャして、保存して、それから解析を開始します。そのため、Wiresharkにはキャプチャしたパケットをキャプチャファイルとして保存する機能が付いています。また複数のキャプチャファイルをマージすることもできます。

キャプチャファイルの保存とエクスポート

キャプチャしたパケットを保存するには、[File]から[Save As]を選択します。すると図4-1のように[Save File As]ダイアログが表示されます。キャプチャしたパケットを保存する場所とファイル形式を選択します。何も指定しない場合は、.pcapファイル形式で保存します。

図4-1　[Save File As]ダイアログからキャプチャファイルを保存

[Save File As]ダイアログ機能を使うと特定のパケットのみを保存できるため、膨れ上がったキャプチャファイルのサイズを小さくするのに非常に便利です。ある範囲のパケット番号を持つパケット、マーキングされたパケット、ディスプレイフィルタによって表示されたパケット（マーキングとフィルタについては本章でのちほど扱います）など、特定のパケットのみを保存することができます。

Wiresharkでは、テキスト、ポストスクリプト、CSV、XMLなど、他のパケット解析ツールのキャプチャファイルの保存形式にキャプチャデータをエクスポートすることができます。エクスポートするには、[File]から[Export]を選び、エクスポートするファイルの保存形式を選択してください。[Save As]から保存するときにも、[ファイルの種類]から保存形式を選択することができます。

キャプチャファイルのマージ

パケット解析をしていると、複数のキャプチャファイルをマージしたくなることがあります。2つのデータストリームを比較したり、別々にキャプチャした同じトラフィックのストリームを組み合わせたりするときなどに、よく行われます。

キャプチャファイルをマージするには、マージしたいキャプチャファイルを開き、[File]から[Merge]を選択し、図4-2のように[Merge with Capture File]ダイアログを開きます。マージしたファイルを選択し、どのようにマージするかを選択します。マージする方法には、[Prepend packets to existing file]（現在表示されているパケットの前にマージするキャプチャファイルのパケットを追加する）、[Append packets to existing file]（現在表示されているパケットの後にマージするキャプチャファイルのパケットを追加する）、[Merge packet chronologically]（タイムスタンプに沿って時系列に追加する）の3つがあります。

図4-2　[Merge with Capture File]から2つのファイルをマージする

パケットを扱う

パケット解析を始めると、膨大な量のパケットに遭遇することになります。数万、数百万とパケットの数が膨れ上がってくると、よほど効率的に解析しないと対応しきれなくなるでしょう。このためWiresharkでは、ある法則に従い、パケットをマーキングすることができるようになっています。またパケットをプリントし、参照することもできます。

パケットの検索

特定のパケットを見つけるには、図4-3のように、Ctrl-Fで[Find Packet]ダイアログを開きます。

図4-3　Wiresharkで特定のパケットを見つける

パケットの検索には3つのオプションがあります。

･[Display filter]オプションでは、検索するフィルタの項目を入力します。

･[Hex value]オプションでは、検索するパケットを16進数（バイトをコロンで区切った）で指定します。

･[String]オプションでは、検索するパケットを文字列で指定します。

表4-1にそれぞれの例があります。

表4-1　パケット検索の例

検索のタイプ　例

Display filter　not ip ip address==192.168.0.2 arp

Hex value 00:ff ff:ff 00:AB:B1:f0

String 　　　ワークステーション1　　ユーザーB　　　ドメイン名

他のオプションとして、検索したいペインの指定、使用するキャラクタセットの設定、検索する方向の設定があります。

オプションを設定し、テキストボックスに検索する文字列を入力して、[Find]をクリックすれば、条件に一致するパケットが表示されます。次候補を検索する場合にはCtrl-Nを、前候補を検索する場合にはCtrl-Bを押してください。

パケットのマーキング

条件に一致するパケットを見つけたら、マーキングしておくことができます。たとえばパケットを分けて保存しておきたい場合や、色を付けてあとから簡単に見つけられるようにしておきたい場合など、マーキングしたいと考えるでしょう。マーキングされたパケットは、図4-4のように黒地に白文字になり目立つようになります。キャプチャされたパケットをファイルに保存するときに、マーキングしたパケットのみを保存することも可能です。

パケットをマーキングするには、パケット一覧のペインでパケットを右クリックし、ポップアップメニューから[Mark Packet]を選んでください。また、パケットをクリックし、Ctrl-Mを押すことでもマーキングできます。Ctrl-Mをもう一度押せばマーキングを解除することができます。パケットは好きなだけマーキングすることが可能です。複数のパケットをマーキングした場合、Shift-Ctrl-NまたはShift-Ctrl-Bでマーキングされたパケットにジャンプすることができます。

図4-4　マーキングされたパケットがハイライト表示されている。この例では1番目のパケットがマーキングされ、暗い色になっている

パケットの印刷

実際の解析は画面上で行われることがほとんどですが、データを印刷する必要があるかもしれません。筆者はパケットを印刷して机の上に貼っておき、他の解析を行いながらその内容をすぐに参照できるようにしています。特に報告書を作成する場合など、パケットをPDF形式で印刷できるのも非常に便利です。

キャプチャしたパケットを印刷するには、図4-5のようにメインメニューの[File]から[Print]を選択します。

図4-5　[Print]ダイアログからパケットの印刷ができる

[Print]ダイアログから、選択したデータをテキストまたはポストスクリプトとして印刷するか、ファイルとして出力することができます。[Save File As]ダイアログと同じように、ある範囲のパケット番号を持つパケット、マーキングされたパケット、ディスプレイフィルタによって表示されたパケットなど、特定のパケットのみを印刷することができます。また、3つのペインのうちどれを印刷するかを選択することも可能です。オプションを選択したら[Print]をクリックしてください。

時間の表示フォーマットと相対時間表示

パケット解析において、時間は重要な要素です。解析する際には、通信にかかる時間とその傾向を調べる必要があります。Wiresharkでは時間の重要性を認識して、いくつかのオプションが提供されています。ここでは、時間の表示フォーマットと相対時間表示を見ていきましょう。

時間の表示フォーマット

Wiresharkでは、各パケットにシステム時刻をもとにタイムスタンプが記録されています。パケットがキャプチャされたまさにその瞬間のシステム時刻や、最初にキャプチャされたパケットからの相対的な時間を表示することもできます。

時間の表示に関するオプションは、図4-6のように、メインメニューの[View]の[Time Display Format]から設定できます。時間の表示フォーマットのほか、時間の精度についても選択が可能です。秒、ミリ秒、マイクロ秒などを指定できます。本書では頻繁にこれらのオプションを変更するので、今のうちに慣れておいてください。

相対時間表示

Wiresharkでは、あるパケットがキャプチャされた時間からの相対的な時間を表示することができます。この機能を使えば、ファイルのキャプチャ開始時点以外のどこかでトリガされた一連のイベントを調べることができます。

相対的な時間を表示するには、パケット一覧のペインから基準となるパケットを選び、メインメニューの[Edit]から[Set Time Reference]を選択します。表示しないようにするには、パケットを選び、[Edit]から[Set Time Reference]設定を切り替えます。

相対的な時間を表示するよう設定すると、図4-7のように、時間の基準となったパケットの時間の部分に\*REF\*と表示されます。

相対的な時間の表示は、時間の表示フォーマットを、キャプチャを開始してからの経過時間を表示するようにしておかないと意味がありません。ほかのフォーマットは混乱を招くだけでしょう。

図4-6　さまざまな時間の表示フォーマット

図4-7　時間の基準になったパケット

キャプチャオプションの設定

3章ではパケットキャプチャの基本中の基本について説明しました。Wiresharkでは図4-8に示したように、[Capture Options]ダイアログからかなりの数のオプションを選ぶことができます。ダイアログを開くには、[Capture]から[Interfaces]を選択、キャプチャしたいパケットのインターフェースの[Options]ボタンをクリックします。

[Capture Options]ダイアログにはあれこれとおまけがついていますが、いずれもパケットキャプチャを便利にするために設計されたものです。[Capture]、[Capture Files]、[Stop Capture]、[Display Options]、[Name Resolution]に分かれたこれらオプションを、ひとつずつ見ていきましょう。

Capture設定

[Capture]の[Interface]ドロップダウンメニューから、ネットワークインターフェースを選択します。左側でインターフェースがローカルかリモートかを指定でき、右側にはキャプチャに使えるインターフェースの一覧が表示されます。ドロップダウンメニューの下に、選択したインターフェースのIPアドレスが表示されます。

図4-8　[Capture Options]のダイアログ

左側の3つのチェックボックスで、プロミスキャスモードの有効無効を切り替えたり（デフォルトでは有効になっています）、現時点では実験段階にあるpcap-ngフォーマットでパケットをキャプチャしたり、各キャプチャパケットのサイズをバイトで制限したりすることができます。

右側のボタンでは、ワイヤレスやリモートでの設定が可能です（使用可能な状況の場合）。その下にあるバッファサイズのオプションは、Windows搭載システムでのみ利用できます。ディスクに書き込む前に、カーネルバッファに保存するキャプチャしたパケットデータの量を指定することができます。大量のパケットをドロップオフしていることに気付かない限り、通常この値は変更しません。[Capture Filter]オプションでキャプチャフィルタを設定します。

Capture File設定

[Captre File]では、パケットをまずキャプチャしてからファイルに保存するのではなく、キャプチャしたパケットを自動的にファイルに保存できます。そのためパケットの保存がかなり楽になります。ひとつのファイル、あるいはファイルセットとして保存することも、またはリングバッファを使って作成したファイルの数を管理することも可能です。このオプションを使うには、[File]テキストボックスに完全なファイルパスとファイル名を入力します。

大量のパケットをキャプチャする場合や、長期にわたってキャプチャを行う場合には、ファイルセットが便利です。ファイルセットとは、特定の条件によって分類した複数のファイルをグループにまとめたものです。ファイルセットとして保存するには、[Use Multiple Files]オプションを選択します。

Wiresharkではさまざまなトリガを使って、ファイルサイズや時間などの条件をもとにファイルセットを管理できます。これらのオプションを有効にするには、[Next File Every]オプション（フルサイズのトリガの上、時間ベースのトリガの下）を選択し、トリガする値と単位を指定します。たとえばキャプチャしたパケットの容量が1MBに達するごとに、あるいは図4-9のようにキャプチャ中に1分経過するごとに新たなファイルを作成するトリガを作ることができます。

図4-9　1分ごとに作成されたファイルセット

オプションを組み合わせて使用することも可能です。たとえばさきほどのトリガを両方指定すると、データが1MBキャプチャされるか、あるいは1分経過するか、どちらかの条件が満たされると新たなファイルが作成されます。

[Ring Buffer With]オプションでは、ファイルセット作成の際にリングバッファを使います。これはいわゆるFIFO（最初に入ってきたものを最初に処理し、次に入ってきたものは最初の処理が終わるまで待たせる）方式です。情報技術では「リングバッファ」には複数の意味がありますが、Wiresharkでは最後のファイルの書き込みが終了した時点でさらにデータ保存の必要がある場合、最初のファイルを上書きするよう指定することを指します。このオプションを選択すると、新規作成するファイルの最大数が指定できます。仮に1時間おきに新しいファイルを作成するよう設定し、リングバッファを「6」に設定するとしましょう。6番目のファイルが作成されるとリングバッファがサイクルバックし？？、7番目のファイルを作成せずに、最初のファイルを上書きします。この設定をすると、新しいデータは書き込まれますが、ハードディスク内のデータファイルの数は6個以上に増えなくなります。

[Stop Capture After]オプションを使うと、一定数のファイルが作成されると、キャプチャを停止するよう設定できます。

Stop Capture設定

[Stop Capture]では、一定のトリガに達するとキャプチャを停止するよう設定できます。複数のファイルセットを扱う場合、ファイルサイズや時間の間隔、またパケットの数によってトリガできます。これらのオプションは先ほど説明したファイルオプションとの併用が可能です。

Display Options

[Display Options]では、キャプチャしたパケットをどのように表示するかを管理できます。[Update List of Packets in Real Time]は見ての通りのオプションで、[Automatic Scrolling in Live Capture]オプションと組み合わせて使用することができます。両方を有効にすると、最後にキャプチャしたパケットがまず表示され、キャプチャしたすべてのパケットが表示されます。

注意　[Update List of Packets in Real Time]と[Automatic Scrolling in Live Capture]を一緒に使用すると、それほど大容量でないデータをキャプチャする場合でも、プロセッサに相当の負担がかかります。リアルタイムでパケットをチェックしなければならない理由がない限り、両方のオプションを使わないほうがいいでしょう。

[Hide Capture Info Dialog]オプションを使うと、プロトコル別にキャプチャしたパケットの数とパーセントを示す小さなウィンドウが表示できます。

Name Resolution設定

[Name Resolution]オプションでは、MAC（レイヤ2）、ネットワーク（レイヤ3）、トランスポート（レイヤ4）の名前解決を自動化することができます。ドローバック？？を含むWiresharkの名前解決については、5章で詳しく説明します。

フィルタを使う

フィルタを使うと、どのパケットを解析するかを正確に指定できます。つまりフィルタは、パケット含むか除外するかの条件を決める構文なのです。表示したくないパケットがあれば、フィルタを作成して取り除けます。特によく見たいパケットがあるなら、それだけを表示するフィルタを作成すればいいのです。

Wiresharkには、2種類のフィルタがあります。

・キャプチャフィルタはパケットをキャプチャしている最中に使用されるもので、所定の構文に含めるまたは除外すると指定したパケットのみがキャプチャされます。

・ディスプレイフィルタはすでにキャプチャしたパケットに適用され、特定の構文に基づいて不要なパケットを隠し、必要なパケットのみを表示します。

まずキャプチャフィルタを見てみましょう。

キャプチャフィルタ

キャプチャフィルタはパケットをキャプチャしている最中に使用されます。キャプチャフィルタを使う主な理由はそのパフォーマンスにあります。解析の必要がないトラフィックがわかっている場合、キャプチャフィルタでフィルタをかければ、これらパケットのキャプチャに使う分の処理能力がセーブできます。

大量のデータを扱う場合、カスタムのキャプチャフィルタを作成できる機能が役立ちます。関係のあるパケットのみを確実に表示することで、解析プロセスが高速化できるからです。

複数のサービスを提供しているサーバに関するトラフィックを解析したいときなどが、キャプチャフィルタを使いたい例のひとつといえます。たとえば、262番ポートを使用するサービスを提供しているサーバのトラブルシューティングを考えてみましょう。サーバがさまざまなポートでサービスを提供しているのならば、262番ポートのトラフィックのみを見るのは大変ですが、フィルタを使えばそれが可能です。この章の最初に説明したように、[Capture Options]ダイアログを使って、フィルタが作成できます。

1．[Capture]ダイアログを開き、[Interfaces]を選択したら、パケットのキャプチャに使いたいインターフェースの横にある[Options]ボタンをクリックし、[Capture Options]ダイアログを開きます。

2．パケットのキャプチャに使いたいインターフェースを選び、キャプチャフィルタを選択します。

3．[Capture Filter]ボタンの横に構文を入力し、キャプチャフィルタを適用します。今回は262番ポートを通るトラフィックだけをキャプチャしたいので、図4-10のように「port 262」と入力します（構文については次の項で詳しく説明します）。

4．フィルタを作成したら、[Start]ボタンをクリックしてキャプチャを始めます。

図4-10　Capture Optionsダイアログでキャプチャフィルタを作成する

適切なサンプルを集めたら、262番ポートを通るトラフィックのみが見えるはずです。これで特定データだけを効率よく解析できます。

キャプチャ/BPF構文

キャプチャフィルタはWinPcapによって適用でき、Berkeley Packet Filter（BPF）構文が使えます。複数のパケット解析ツールでもこの構文が使われているのは、ほとんどのパケット解析ツールがlibpcap/WinPcapライブラリを使っており、このライブラリでBPFが使用できるからです。ネットワークをパケットレベルまで深く掘り下げていくには、BPF構文の知識が不可欠です。

BPF構文を使って作成したフィルタを「expression（式）」と呼び、各式は1つまたは複数の「primitive（プリミティブ）」で構成されています。プリミティブは1つまたは複数の「qualifier（限定子）」で構成され（表4-2のように）、そのあとに図4-11で示したように、ID名または数字が続きます。

表4-2　BPF限定子？？？？？

限定子　説明　例

タイプ　Dir　Proto

ID名または数字による識別子　　host、net、port

Dir　ID名または数字との転送方向を指定する　　src、dst

Proto　特定のプロトコルへのマッチを限定する　　ether、ip、tcp、udp、http、ftp

図4-11　キャプチャフィルタの一例

プリミティブ　演算子　プリミティブ

限定子

ある式の要素では、srcの限定子と192.168.0.10のIDが一緒になって、プリミティブを形成します。このプリミティブのみだと、192.168.0.10のIPアドレスからのトラフィックのみがキャプチャされます。

論理演算子を使ってプリミティブを組み合わせ、より高度な式を作成することが可能です。使える論理演算子は3つあります。

・連結演算子　AND（&&）

・交代演算子　OR（||）

・否定演算子　NOT（!）

たとえば次の式は、192.168.0.10のIPアドレスで、80番ポートを通過するトラフィックのみをキャプチャします

src 192.168.0.10 && port 80

ホスト名とフィルタの指定

フィルタは通常、特定のネットワーク機器か、グループ化された機器を中心とします。状況にもよりますが、その機器のMACアドレス、IPv4アドレス、またはDNSホスト名などをもとにフィルタするのです。

たとえばネットワーク上のあるサーバとやり取りしている特定ホストのトラフィックが気になるとしましょう。そのサーバでhost限定子を使い、ホストのIPv4アドレスに関連するすべてのトラフィックをキャプチャするフィルタを作成します。

host 172.16.16.149

IPv6ネットワーク上にいる場合、次に示すようにhost限定子を使って、IPv6アドレスを元にフィルタをかけます。

host 2001:db8:85a3::8a2e:370:7334

ホスト名でフィルタをかけることもできます。

host testserver2

ホストがIPアドレスを変更している可能性が気になるなら、etherプロトコル限定子を追加して、MACアドレスでフィルタをかけることも可能です。

ether host 00-1a-a0-52-e2-a0

先の例のように、転送方向の限定子？もフィルタと組み合わせ、ホストを通過するかどうかを基準にトラフィックをキャプチャする場合に使えます。たとえば、特定ホストから来るトラフィックのみをキャプチャするなら、src限定子を追加します。

src host 172.16.16.149

サーバ172.16.16.149から疑わしいホストに向かうデータのみをキャプチャするなら、dst限定子を使います。

dst host 172.16.16.149

タイプ限定子（host、net、port）をプリミティブと一緒に使用しない場合は、host限定子だとみなされます。したがって前の例ではhost限定子が削除されます。

dst 172.16.16.149

ポートおよびプロトコルフィルタ

ホストでのフィルタだけでなく、ポートでもフィルタをかけることができます。ポートフィルタは、既知のサービスポートを使っているサービスまたはアプリケーションでの使用が可能です。8080番ポートを通過するトラフィックのみをキャプチャする簡単なフィルタの例を示します。

port 8080

8080番ポートを通過する以外のすべてのトラフィックをキャプチャする場合は次のようになります。

!port 8080

ポートフィルタと転送方向限定子を組み合わることが可能です。たとえば、標準的なHTTP80番ポートで待ち受けするWebサーバへ向かうトラフィックのみをキャプチャする場合、dst限定子を使います。

dst port 80

プロトコルフィルタ

プロトコルフィルタを使うと、特定のプロトコルでフィルタをかけることができます。特定ポートでは定義できない、非アプリケーションレイヤプロトコル？と合わせる場合に使います。つまりICMPトラフィックだけをチェックしたい場合などにこのフィルタが便利です。

icmp

IPv6トラフィック以外のすべてのトラフィックを見たい場合は、これが使えます。

!ip6

プロトコルフィールドフィルタ

BPF構文の真の力のひとつが、プロトコルヘッダのすべてのバイトを調べて、そのデータをベースにした特別なフィルタを作成できる機能です。ここで説明するこの高度なフィルタを使うと、パケットの特定の位置から、特定のバイトだけを検索することができます。

たとえば、ICMPヘッダのタイプフィールドでフィルタをかけたいとしましょう。タイプフィールドはパケットの一番先頭にあり、オフセットが0になっています。パケット内で調べたい位置を識別するには、プロトコル限定子の横にあるスクエアブラケットで、バイトをオフセットにします。この例ではicmp[0] となります。すると1バイトの整数値が戻されるので、比較することができます。たとえば、タイプ3到達不能（Destination Unreachable）メッセージを持つICMPパケットのみを取得するなら、次のようにフィルタ式で等値演算子を使います。

icmp[0] == 3

エコー要求（タイプ8）またはエコー応答（タイプ0）のICMPパケットのみを調べるには、2つのプリミティブとOR演算子を使います。

icmp[0] == 8 || icmp[0] == 0

これらのフィルタはうまく機能しますが、パケットヘッダの情報の1バイトしかフィルタしません。ただ幸いなのは、スクエアブラケット内のオフセットの数字のあとに、コロンで区切ってバイト長をフィルタ式に付け加えれば、戻ってくるデータの長さも指定できることです。

たとえば、タイプ3、コード1として識別される、すべてのICMP到達不能、ホスト到達不能パケットをキャプチャするフィルタを作成したいとします。これらはパケットヘッダのオフセット0で、互いに隣り合う1バイトのフィールドです。パケットヘッダのオフセット0で始まるデータの2バイトをチェックし、16進値0301（タイプ3、コード1）と比較するフィルタを作成します。

icmp[0:2] == 0x0301

TCPパケットだけをキャプチャする場合によく使われるのが、RSTフラグセットです。このTCPについては6章で詳しく説明します。ここではTCPパケットのフラグがオフセット13にあるということだけ理解してください。これはフラグフィールドとしては1バイトなのですが、それぞれのフラグがこのバイト内でシングルビットとして識別されるという、面白いフィールドです？？TCPパケットでは複数のフラグを同時に設定できるので、単一のtcp[13]値だけは効果的にフィルタをかけることができません。複数のものがRSTビットを表している可能性があるからです？？？したがって調べたいバイト内の位置を指定する必要があり、そのためにはその位置を現在のプリミティブに、ひとつのアンパサンド（&）で付け加えます。RSTフラグはこのバイト内で4を表しているので、このビットを4に設定すればフラグを設定したことになります？？？。フィルタはこのようになります。

tcp[13] & 4 == 4

数字の8で表すビット位置で識別される？？PSHフラグセット内のすべてのパケットを見る場合は、代わりにその位置を使います。

tcp[13] & 8 == 8

キャプチャフィルタ式のサンプル

現在の状況に合ったフィルタを作成できるかどうかで、解析の成功の是非が決まります。表4-3は筆者がよく使うキャプチャフィルタの一例です。

表4-3　一般的なキャプチャフィルタ

フィルタ　説明

tcp[13] & 32 ==32　URGフラグセットのTCPパケット

tcp[13] & 16 ==16　ACKフラグセットのTCPパケット

tcp[13] & 8 == 8　PSHフラグセットのTCPパケット

tcp[13] & 4 == 4　RSTフラグセットのTCPパケット

tcp[13] & 2 == 2　SYNフラグセットのTCPパケット

tcp[13] & 1 == 1　FINフラグセットのTCPパケット

tcp[13] ==18　TCP SYN-ACKパケット

ether host00:00:00:00:00:00（MACを入れる）　MACアドレスを通過するトラフィック

!ether host 00:00:00:00:00:00（MACを入れる）MACアドレスを通過しないトラフィック

broadcast　ブロードキャストトラフィックのみ

icmp　ICMPトラフィック

icmp[0:2] == 0x0301　ICMP到達不能、ホスト到達不能

ip　IPv4トラフィックのみ

ip6　IPv6トラフィックのみ

udp　UDPトラフィックのみ

ディスプレイフィルタ

ディスプレイフィルタは、作成されたキャプチャファイルに適用されるフィルタで、フィルタに一致するパケットのみを表示させます。パケット一覧のペインの上部にある[Filter]テキストボックスにフィルタを記述します。

ディスプレイフィルタはキャプチャフィルタより使う機会が多いでしょう。実際のキャプチャファイルのデータを変更することなく、特定のパケットだけを表示することができるからです。もともとのキャプチャファイルのパケットを表示させたいなら、テキストボックスに記述したフィルタを消せばよいのです。

フィルタは、キャプチャファイルから解析に無関係のパケット（ARPブロードキャストパケットなど）を一時的に消去する役にも立ちます。しかしながら、ARPブロードキャストパケットは後で解析に必要になる場合があるので、キャプチャフィルタを使うよりも、ディスプレイフィルタで一時的に表示させないようにするほうが便利なのです。

ARPパケットを表示させないようにするには、パケット一覧のペインの上部にある、[Filter]テキストボックスに移動します。!arpと入力し、図4-12のように、パケット一覧のペインからすべてのARPパケットを削除します。フィルタを削除するには、[Clear]ボタンをクリックします。

図4-12　[Filter]テキストボックスでディスプレイフィルタを作成する

[Filter Expression]ダイアログ（簡単な方法）

図4-13の[Filter Expression]ダイアログは、Wireshark初心者がキャプチャフィルタやディスプレイフィルタを作成する支援をしてくれる機能です。ダイアログを表示するには、[Capture Options]で[Capture Filter]ボタンをクリックし、次に[Expression]ボタンをクリックします。

ダイアログの左側には、使用可能なプロトコルの一覧が表示されており、各プロトコルで使用可能なフィルタ要素を指定できます。フィルタを作成するには、以下の手順に従ってください。

1．プロトコル名の横にある[+]をクリックすれば、各プロトコルで利用可能なフィルタ要素を見ることができます。利用したいフィルタ要素をクリックしてください。

2．選択したフィルタ要素と、その評価値の評価方法を指定してください。評価方法は、イコール（=）、大なり（>）、小なり（<）などの演算子です。

3．評価値を指定して、フィルタを作成します。Wiresharkが提供する評価値を選択するか、自身で値を選択してください。

4．フィルタを作成し終わったら、[OK]ボタンをクリックしてください。作成したフィルタがテキストで表示されます。

[Filter Expression]ダイアログは初心者には非常に便利な機能ですが、フィルタの使用方法が理解できれば、手動でフィルタを作成するほうが効率がよいでしょう。ディスプレイフィルタは非常に強力ですが、構文は簡単です。

図4-13　[Filter Expression]ダイアログを使えば、より簡単にフィルタを作成できる

Filter Expression構文構造（難しい方法）

キャプチャフィルタやディスプレイフィルタは、特定のプロトコルを選別するときに使うことが多いでしょう。たとえばTCPのトラブルシューティングの場合は、TCPのトラフィック以外は必要ないので、TCP以外のものをフィルタリングしてしまいましょう。

この課題を別の側面から見てみましょう。トラブルシューティングのためにpingを多用して、ICMPのトラフィックが大量に発生したとします。!icmpというフィルタを使えば、ICMPのトラフィックを削除することができます。

比較演算子を使えば、値を比較することができます。たとえばTCP/IPネットワークのトラブルシューティングの場合、特定のIPアドレスを参照するすべてのパケットを見る必要があるでしょう。比較演算子「==」を使えば、192.168.0.1というIPアドレスを含むパケットのみを表示するフィルタが作成できます。

ip.addr==192.168.0.1

今度は長さが128バイト以下のパケットのみを表示するとしましょう。この場合は「<=」という比較演算子を、フィルタの式で使用すればよいのです。

frame.len　<=128

Wiresharkで使用可能な比較演算子は表4-4のとおりです。

表4-4　Wiresharkのフィルタとして使用できる比較演算子

演算子　説明

等しい

等しくない

大なり

小なり

以上

以下

論理演算子を使えば、複数のフィルタを1つの表現として使用することができます。論理演算子を使いこなすことができれば、使用できるフィルタが飛躍的に増えます。たとえば、2つのIPアドレスを含むパケットのみを表示したいとしましょう。この場合「or」演算子を使って、次のようにどちらかのIPアドレスを含むパケットを表示する式を作ればいいのです。

ip.addr==192.168.0.1 or ip.addr==192.168.0.2

Wiresharkで使用可能な論理演算子は表4-5のとおりです。

表4-5　Wiresharkのフィルタとして使用できる論理演算子

演算子　概要

論理積

論理和

排他的論理和

否定

ディスプレイフィルタのサンプル

フィルタの概念は単純ですが、実際にフィルタを作成するときには、どんなキーワードや演算子を使ったらよいか悩むところでしょう。表4-6は筆者がもっとも頻繁に使うディスプレイフィルタの一部です。すべてのリストを見るには、<http://www.wireshark.org/docs/dfref/>を参照してください。

表4-6　一般に使用されるディスプレイフィルタ

RDPトラフィックをクリアにする

SYNフラグセットのTCPパケット

RSTフラグセットのTCPパケット

ARPトラフィックをクリアにする

すべてのHTPトラフィック

クリアテキスト管理トラフィック？？（TelnetまたはFTP）

クリアテキストemailトラフィック？？（SMTP、POP、IMAP）

フィルタの保存

フィルタを使用していると、特定のフィルタを頻繁に使うことがあります。同じフィルタを何度も作成する必要はありません。Wiresharkには、フィルタを保存する機能がついているのです。フィルタを保存するには、以下の手順に従ってください。

1．[Capture]から[Capture Filters]を選択し、[Capture Filter]ダイアログを開いてください。

2．ダイアログの左側にある[New]ボタンをクリックし、新たなフィルタを作成します。

3．[Filter Name]ボックスにフィルタの名前を入力します。

4．[Filter String]ボックスに実際のフィルタの式を入力します。

5．フィルタ式を入力したら、[Save]ボタンをクリックして保存します。

カスタムのディスプレイフィルタを保存するには、以下の手順に従ってください。

1．[Analyze]から[Display Filters]を選択、またはパケット一覧のペインの上部にある[Filter]　ボタンをクリックして、図4-14のように[Display Filter]ダイアログを開いてください。

図4-14　[Display Filter]ダイアログからフィルタを保存できる

2．ダイアログの左側にある[New]　ボタンをクリックし、新たなフィルタを作成します。

3．[Filter Name]ボックスにフィルタの名前を入力します。

4．[Filter String]ボックスに実際のフィルタの式を入力します。

5．フィルタ式を入力したら、[Save]ボタンをクリックして保存します。

Wiresharkには、ビルトインのフィルタがいくつかありますが、これらはフィルタがどのようなものかを示す一例です。独自のフィルタを作成する際に、Wiresharkのヘルプページと合わせて利用できます。フィルタの例は本書を通じて使用します。