プロトコルエミュレーション機能を持つTCP/IPリンクモニタの設計

4N－9

大岸　智彦　　井戸上　彰　　加藤　聰彦　　鈴木　健二

国際電信電話株式会社　研究所

1．はじめに

近年、TCP/IP プロトコルに従った通信が広く行われているが、TCP ではパラメータ設定の不一致等により、スループット低下が生ずることが指摘されている。このような問題点に対しては、フォーマット解析機能を有するリンクモニタを用いて、通信シーケンスを検査するのが通常であるが、TCP では輻輳制御等の複雑な手順を実装しているため、この方法では原因の解析が困難となる。このため、単純にフォーマット解析を行うだけでなく、状態／内部変数を推定しながらプロトコル動作を追随するプロトコルエミュレーション機能[1]が必要となる。そこで筆者らは、オンライン時のフォーマット解析機能と、オフライン時の TCP のプロトコルエミュレーション機能を併せ持つリンクモニタを設計した。本稿ではその概要について述べる。

2．設計方針

(1) ネットワーク上を流れるフレームをリアルタイムに監視／解析するオンラインモードと、特定の端末間の通信に着目し、そのプロトコル動作のエミュレーションを行うオフラインモードを持つ。

(2) オンラインモードでは、IP、TCP、UDP 等のフォーマットの解析／表示を行う。また、オフラインモードで使用するため、解析結果をログに記録する。

(3) オフラインモードでは、伝送遅延やフレーム転送遅延等を考慮し、着目した端末が実際にそのフレームを送受信した時刻を推定し、モニタが取得したフレームを、端末毎に、イベント発生順序に並び替える。次に、その情報に基づいて、端末毎に、TCP の状態や内部変数の推定を含む、TCP のエミュレーションを行う。

(4) 重複(duplicate) ACK やスロースタートの発生など、エミュレーション時に得られた TCP の動作を示す情報を付加的に表示させる。

3．構成

本モニタは、図１に示すように、オンライン／オフラインモードを実現するオンライン／オフライン処理部に分かれる。フレーム取り込み部では、ネットワーク上の全てのフレームの取り込み処理を行う。オンライン解析部では、プロトコル毎にフレームのフォーマット解析とチェックサムエラーの検出のみを行う。ログ蓄積部では、IP パケットと TCP セグメントのヘッダ部分のみをログに記録する。ログ読み込み部では、オンライン処理部で作成されたログより、指定した端末(A)(B)間で通信された TCP セグメントのみを取り出し、実際の送受信時刻を推定しながら双方の端末のイベント一覧ファイルを作成する。(A)から(B)に流れたフレームは、(A)用のファイルには送信イベント、(B)用のファイルには受信イベントとして記録される。オフライン解析部では、イベント一覧ファイルをもとに、プロトコルのエミュレーションを行う。

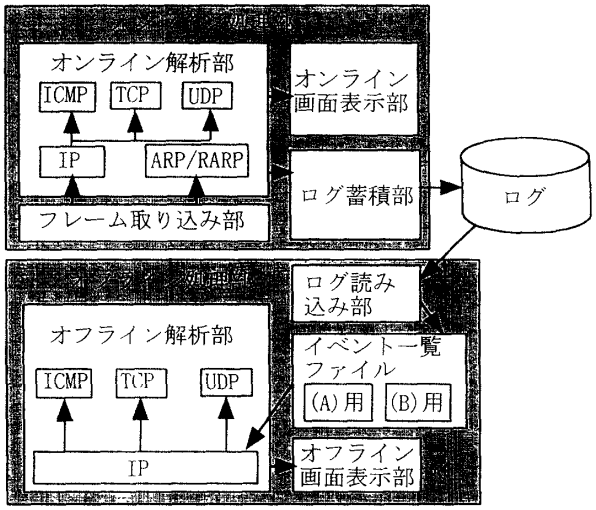


図１　TCP/IPリンクモニタの構成

4．オフライン処理の詳細

4．1．ログの形式

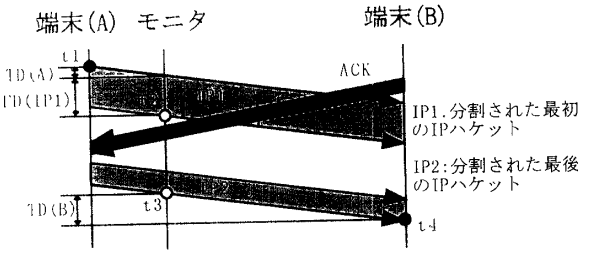
ログは、フレーム検出時刻、IP ヘッダ、IP ヘッダのチェックサムエラーの有無、TCP/UDP/ICMP ヘッダ、TCP/UDP/ICMP のパケット長、TCP/UDP のチェックサムエラーの有無の情報を持つ。

4．2．送受信時刻推定

TCP の送受信イベントの順序を正しく把握するため、着目した端末の送受信時刻を、モニタの検出時刻(フレームの最後のビットを検出した時刻)より推定する必要がある[2]。図２は、IP フラグメンテーションが行われたデータ(IP1, IP2)と ACK の送受信時刻の推定を表したものである。ここでは、伝送遅

Design of TCP/IP Link Monitor with Protocol Emulation Function
Tomohiko Ogishi, Akira Idoue, Toshihiko Kato and Kenji Suzuki
KDD R&D Laboratories

延はネットワーク構成に依存する固定値とし、フレーム転送遅延はフレームサイズに比例するものとして、イベント時刻を推定している。



(A)のデータ送信時刻(t1)=IP1検出時刻(t2)-(A)の伝送遅延(TD(A))-IP1のフレーム転送遅延(FD(IP1))
(B)のデータ受信時刻(t4)=IP2検出時刻(t3)＋(B)の伝送遅延(TD(B))

図2　送受信時刻の推定

4．3．エミュレーションの手順

エミュレーションでは、イベント一覧ファイルより、送信／受信イベントを時間順に取り出し、イベント毎に処理を行う。

エミュレーションは状態遷移表を用いて以下のように行う。送信イベントを取り出した場合、現在の状態が不明の場合は推定し、そのイベントを処理した後の状態／内部変数に遷移する。受信イベントを取り出した場合は、その受信イベントに対する送信イベントが期待されない場合は、そのイベントを処理した後の状態／内部変数に遷移し、期待される場合は、送信イベントに応じて可能な遷移を選択する。例えば、図３の状態 "SYN_RCVD" においてイベント "SYN, ACK受信" を抽出したとき、次の送信イベントが ACK なら 1)の処理を行い ESTABLISHED に遷移する。

	CLOSED	LISTEN	SYN_RCVD	SYN_SENT
SYN	RST CLOSED	1) SYN, ACK SYN_RCVD 2) LISTEN	SYN, ACK SYN_RCVD	1) SYN, ACK SYN_RCVD 2) RST SYN_SENT
SYN, ACK	RST CLOSED	RST LISTEN	1) ACK ESTABLISHED 2) RST SYN_RCVD	1) ACK ESTABLISHED 2) RST SYN_SENT
RST	CLOSED	LISTEN	1) CLOSED 2) LISTEN	CLOSED
active open	SYN SYN_SENT	SYN SYN_SENT	1) CLOSED 2) LISTEN	CLOSED

(注)横軸-状態、縦軸-受信イベント、枠内-期待される遷移

図3　状態遷移表(一部抜粋)

また、続けて取り出したセグメントの時間差が大きい場合は、その間に、再送タイマ、2MSL タイマ、パーシストタイマ等のタイムアウトの可能性があることを考慮する。

データ転送フェーズでは、主にシーケンス番号、受信ウインドウ(advertised window)、受信ウインドウの下限／上限、送信／受信バッファ等の内部変数の更新に基づいたエミュレーションを行う。また、ACK 及びデータ(ユーザデータを含む ACK)処理時に以下の情報を推定することを可能とする。

1) ACK 処理時
・対応するデータ
・重複 ACK か否か
・ウインドウ更新(window update)の ACK か否か
・パーシストタイマ、キープアライブタイマにより発生した ACK か否か
2)データ処理時
・cwnd(congestion window)の値
・タイムアウトによる再送であるか否か
・fast retransmit and fast recovery アルゴリズムによる再送であるか否か

また、過去のエミュレーション結果を蓄積することにより、特定のデータや ACK のパターンの検出により、スロースタートの発生、Nagle アルゴリズムの ON/OFF 等の推定を可能とする。

4．4．バックトラック処理

エミュレーション途中で送出できないセグメントを検出し追随不能になった場合、前の状態／内部変数に戻し、順序を入れ替えて、再度エミュレーションを実行する(バックトラック処理)。これは、次の場合に対処するためにも有効である。

●４．２．において、低速ネットワークを経由する場合や、通信経路のルータに大量のデータが蓄積されている場合などは、セグメント毎に伝送遅延が変動し、モニタで推定不能となる。このような場合、実際のデータの受信時刻が推定値よりも遅れると考えられる。

●送信セグメントと受信セグメントの推定時刻が非常に近い場合は、実際の送受信の順序が逆である可能性がある。

バックトラック処理を可能とするため、途中の状態／内部変数を逐次記録しながらエミュレーションを行う。

4．おわりに

筆者らは、プロトコルエミュレーション機能を持つ TCP/IP リンクモニタの設計を行った。本モニタは、TCP/IP の通信状況の解析や性能上の問題点の検出等を行う上で有効と考えられる。最後に日頃ご指導頂く KDD 研究所村上所長に感謝する。

参考文献

[1] 大岸他、"プロトコル誤りの自動検出機能を持つインテリジェント OSI 7層リンクモニタの設計," 信学 AI 研究会、May 1995
[2] 加藤他、"ルールベースプログラミングを用いたプロトコル相互接続試験システムの設計," 情処 DPS 研究会、Jul. 1996