マイクロサーバを利用した学内ネットワーク監視システムの構築

大垣内 多徳 † 山下 芳範 † 高岡 宏光 ‡
福井医科大学 医療情報部 †
株式会社 九南 ‡

概 要

マイクロサーバを利用した，学内ネットワーク監視システムの構築を行った．マイクロサーバをネットワーク機器の下流に設置することにより，SNMP, RMON 等によるネットワーク機器監視システムでは不可能な end-to-end の通信状態を複数のプロトコル別に監視する事が可能となる．本システムでは，マイクロサーバに Unix like な OS を採用したことにより，リモートから監視対象へアクセスすることができ，幹線側からだけでなく支線側からの調査も可能である．従って，障害発生時に調査する際の手段を拡げられ，障害解析に貢献することが明らかとなった．

An implementation of LAN monitoring system with microservers

Tatoku OGAITO†, Yoshinori YAMASHITA† and Hiromitsu TAKAOKA‡
Fukui Medical University†
Kyunan Corporation‡

Abstract

We have developed LAN monitoring system with microservers. Using a microserver at the downstream position of the each branch switch, we can detect errors of the specific end-to-end connection, which cannot be detected by SNMP nor RMON base monitoring system, for any supported protocols. Furthermore, we can study the network status from the terminal of end-user since we adopt Unix like operating system for client machines. This new system is very helpful for us to analyse the network errors in detail.

1 はじめに

福井医科大学 (以下，本学と略す) は職員数 1000 名，学生 840 名の，単科医科大学である．研究系，業務系のネットワークは分離されておらず，病院を含めた学内全体が一つのネットワークを構成している (以下，学内ネットワークと略す)．この学内ネットワークに接続している計算機は，病院での利用を含めて約 3800 台である．

学内ネットワークは情報処理センターに設置したセンタースイッチと各建物に配置した支線スイッチの間を光ケーブルを用いて接続しており，各支線スイッチ位置で IPv4, AppleTalk, IPX については routing を行っている．その他のプロトコルについては bridge 運用である．

利用者には，利用する部屋に設置されている情報コンセント番号と利用プロトコル，利用端末に関する情報等を申請させ，情報処理センターでホスト名，IPv4 アドレス等の管理を行っている．利用形態としては，業務系では Windows 端末が多く利用されているが，研究系では Apple Macintosh の利用も非常に多いのが現状である．また，病院業務で利用されている機器の中には，AppleTalk や LAT 等が必要なものも多数存在する．

このような状況において，ネットワーク機器の不調が引金となり病院を含む学内ネットワークが利用できない事態が発生した．この障害を調査している際に，ネットワーク機器へのアクセスの可否とその先の端末に対する通信の可否が独立である (一致していない) 事が判明した．この事実は，従来利用されていた SNMP および RMON 等のネットワーク機器に対する監視では発見できないネットワークエラーがあることを意味する．特にネットワーク機器は正常に動作しているにも関わらず，その先の端末に対して通信ができないタイプのエラーは，病院等のネットワークに依存しているユーザがいる場合，早急に対処する必要があるにも関わらず，管理者側では発見しづらいのは大きな問題である．

本学では，上記の問題に対応するため，マイクロサーバを利用して端末間の通信を監視するシステムを構築したので，ネットワーク監視の一手法としてここに報告する．

2 さまざまな監視方法とその問題点

本研究で取り上げる監視システムを構築するまでに，いくつかの手法による監視を行ったので，それぞれの方法とその問題点についてまとめる．

2.1 先史時代

問題が発生した当初は，エラー発生の頻度や個所の特定を最優先課題としたため，既存の端末，コマンドを用いて監視を行った．具体的には情報処理センター設置の Microsoft Windows 上で複数の DOS 窓を開き，各支線配下もしくは支線スイッチに対して，ping コマンドを発行していた．

この方法では，監視状態は画面上を流れていくだけであり，監視するための人員が必要である．また，人員を確保できたとしても，不注意による見逃し等が多発したとともに，エラーが発生した事実が認識できたとしても頻度等の統計処理に多大なコストがかかり，統計処理まで手が回らなかった．

さらに，本センターで利用している端末では IPv4 のみの監視を行っていたが IPv4 通信が不能であるにも関わらず，IPv6 通信には異常が生じてない事実が発見され，IPv4 以外のプロトコルに対する監視が必要であることが明らかとなった．

2.2 監視プログラムの利用

監視プログラムとして，協力して障害解決にあたった業者の機材にインストールされていた市販監視プログラムも利用した．

利用したプログラムは，SNMP に基づく監視機能に加えて，いくつかの用意されたフォームを用いた監視機能を持っている．また 1 週間平均などのあらかじめ用意された簡単なフォームに基づき統計処理も行うことが可能である．

しかしながら，これらのプログラムを用いた場合でも，IPv4 以外の監視ができないこと，また障害発生を速やかに把握できないことは問題であった．また，本学はライセンス取得を行っていないため，常時利用するためにはライセンス取得のための費用が発生することになるのも問題であった．

2.3 積極的なコンピュータの利用

前述の，問題点に対応するために，未配置であった病院用端末の PC を NetBSD[1] で運用し，さらに netatalk[2] を導入して重要な拠点を中心に配備した．NetBSD は標準状態で IPv4, IPv6 をサポートしている．これらを監視対象として，既存コマンド (ping, ping6, aecho) による監視を行った．

監視方法としては，各コマンドの出力を log としてファイルに記録し，その log file を解析するプログラムを cron から定期的に起動する方法を取った．さらに解析プログラムが，停止等の問題が発見した場合には，関係者の携帯電話アドレスも含む mailing list あてに警報メールを送出するようにし，監視人員を必要としなくなった．

当初は，各探査コマンドを実行し続け，シーケンス番号と開始時刻から時刻情報を算出していたが，ネットワークエラーの結果，通信不能状態が生じた場合，time out を待つため，時刻情報が大きくずれ，他の監視ログと時刻情報を比較できなくなることが見いだされた．そのため，cron から毎分 1 回起動し，時刻のずれを小さくするように運用を変更した．

このように既存コマンドを利用することにより監視自体は可能となったが，次のような問題を生じることが判明した．

- ログファイルのサイズ:
  既存コマンドの出力は，各パケットに含まれる情報とともに固定文字列も含む．また，パケットに含まれる情報も正常なものに関しては，記録する必要がない．これら不要な情報によりログファイルのサイズが大きくなり，ディスク容量を圧迫する．

- ログファイルの後処理:
  既存コマンドの出力は，シーケンス番号情報は含まれるが，ネットワーク機器の log と比較するために必要な時刻情報は記録されていないため，ログファイルから時刻情報を算出する後処理が必要となる．
- プロセス起動処理:
  Cron(8) を用いてほぼ同時に多数のコマンドを起動することにより，プロセス生成，File IO 等の kernel への負荷が特定の時刻に集中し，その結果 Round Trip Time(RTT) の定量的な評価に問題が生じた．
- 電源，配置場所の確保:
  いわゆる PC を利用したことにより，配置済み無停電電源の容量によっては電源の確保が難しい場合や，設置スペースが確保できない場合があった．

以上の考察により，この方法では，数日から数週間程度の監視程度なら可能であるが，常時監視し続けるためには，上記の問題点について対策を行う必要があることが判明した．

3 実装

以上の経験から，監視システムに必要な機能は次のようにまとめられる．

1. 監視プログラムを実行する親機と監視対象となる子機からなること
2. 子機はリモートから利用/管理ができること
3. 複数のプロトコルの監視が可能であること
4. 正確な時刻情報が記録されること
5. 必要十分な情報が記録されること
6. 管理者が異常発生時に速やかにその事実を知ることができること

(7. 統計処理ができること)

上記要請を満たすように次のように設計実装を行った．

3.1 監視装置

3.1.1 子機

監視対象となる，子機は次の条件を満たす必要がある．

1. 複数のプロトコルに対応していること
2. 無停止で運用できること
3. 小型，小電力であり，配置が容易であること
4. 2 種以上のプロトコルでリモートからアクセスでき，ネットワーク障害について調査が可能であること

本実装では，これらの要求を満たすよう，PlatHome 社製の OpenBlockSS(図 1) を用いた．初期状態では IPv4 のみ対応であったため，SSD/Linux[3] に入れ替え，さらに netatalk を導入し，各支線スイッチ配下に設置した．

子機として利用中のマイクロサーバの主な仕様を表 1 に示す．

3.1.2 親機

監視起点として用いる親機は子機と異なり，配置場所，電源の確保が容易である．また，監視ログを保存するためのディスクが必要でもあることから，マイクロサーバは用いていない．現在は，学内で利用されなくなり，センターに寄付された Power Macintosh 7500 を用いている．親機の仕様は表 (2) に示す．

3.2 監視プログラムの設計

監視プログラムの設計においては次の点に注意して実装を行った．

1. 正確な時刻情報の取得:
   障害発生時に複数の機器にまたがるログ情報の突き合せが必要であるため， 時刻を記録した後に，各プロトコルのパケットを送出する事．時刻同期を取るためには NTP を利用する．

図 1: OpenBlockSS

CPU	PowerPC 405GP 200MHz
Memory	64MB SDRAM
Flash ROM	8MB
Compact Flash	128MB
LAN	100BaseTX 2Ports
消費電力	最大 5W
外形寸法 (mm)	118(W)×84(D)×53(H)
kernel	Linux 2.4.18
IPv4	NET4.0
IPv6	usagi-stable-release 3.1
AppleTalk	netatalk 1.5.3.1

表 1: 子機仕様

CPU	PowerPC G3 450MHz
Memory	32MB
HDD	8GB
LAN	10BaseT
kernel	NetBSD 1.6
IPv6	KAME (標準)
AppleTalk	netatalk 1.5.3.1

表 2: 親機仕様

2. プロトコル依存 な routine の分離:
   容易に新たなプロトコルの監視の追加が行えるよう，プログラム内でプロトコル依存なルーティンを分離し，保守運用が容易となるような設計である事.

3. 負荷の軽減/分散:
   特定の時刻に負荷が集中するのを避けるため，各監視対象へのプログラムを時間的に分散して起動する事．また，FILE IO を削減するため，単一プログラムですべてのプロトコル監視を行い，監視対象につき一つのログファイルを用いる事.

4. 障害情報の管理者への連絡:
   学内ネットワークには，病院等の重要度の高い部分と，夜間人の出入りが少ない比較的重要度の低い部分があることを考慮し，重要度に応じて異なるメンバーへ連絡できるよう，複数の警報送付先をサポートする事.

以上の条件を考慮し，実際に監視を行う探査コマンドと，それらを管理し，障害発生時に警報を選択/送付する管理コマンドにわけて実装を行った．監視対象および警報送付先などのすべての設定は，単一ファイルに記述され，探査コマンドは適切な時間をおいて管理コマンドから起動される．探査コマンドは各時刻における探査パケットの RTT を記録するとともに指定した間隔でパケットロス数や RTT の統計情報を出力するように実装した.

4 評価と議論

4.1 親機の負荷

親機の性能が低いこともあり，ping 等の既存プログラムを利用した場合には常に load average が 7 程度だったものが，新たに作成した監視プログラムに変更した後は，0.5 程度まで減少した．これは，連続的に単一プログラムで監視することによりプロセス生成コストおよびログファイルの作成コストがなくなったことによるものと考えられる．

ただし，OS のスケジューリング機能が適切であるためか，障害発生時の親機での作業のレスポンスは，それほど変化は生じなかった．

4.2 安全性

現状では，監視プログラムは IPsec には対応していない．しかしながら子機へのアクセスは IPv4, IPv6 による ssh のみ提供しているため，リモートから子機を起点に調査する際であっても，現地起点の調査と同等の安全性は確保できているものと考える．

4.3 可用性

現状では，IPv6 routing が可能な機器は高価なため，IPv6 に関しては多くの場合 bridge 運用が行われているものと考えられる．本学での運用も IPv6 に関しては bridge 運用であり，これは ネットワーク機器の Layer2 スイッチング機能が正常に動作している限り、到達性が保証される事を意味している．したがって，routing 等の Layer 3 に異常が生じた場合でも、監視拠点から移動することなく各支線における状況を調べることができ，問題の把握，解決にいたる時間が短縮できると考えられる．

4.4 ネットワークへの負荷

ネットワークに対する負荷としては，もっとも負荷が大きいと考えられる親機が所属するネットワークセグメントで考えた場合，

監視対象数 × 監視プロトコル数 × packet 長

となるが，本学のネットワークの場合，主要部分の帯域は 1Gbps となっているため，応答性が悪化することもなく，また mrtg による監視からも顕著な変化は見られなかった．

4.5 他の監視装置との比較

他の監視システムと比較して，本監視システムには次のような特徴を持つ．

- ライセンス問題:
  ベースとなる OS に対して，自由に利用できる NetBSD や Linux を採用したことによりライセンス費用等が発生せず，安価に構築することが可能である．
- 双方向監視が可能であること:
  既存の監視システムの多くは一つの監視拠点から全体を見る「センター監視型」であるのに対し，本システムは「分散監視型」と位置づけることができる．通常は監視拠点からの監視を行い，必要時には子機からの調査を行うことで経路情報などを双方向から確認することが可能である．
- 複数のプロトコルによる監視:
  既存の監視システムの多くは，IPv4 等の限られたプロトコルのみの監視しか行えず，本学のような複数のプロトコルを利用している環境では不十分な調査しか行えない．本監視システムでは，利用する OS を目的のプロトコルに対応させることにより，容易に監視対象プロトコルを増やすことが可能である．具体的には，今後，IPX や LAT 等への対応を考えている．

  また，bridging を行っているプロトコルを含めることにより，routing におけるバグ等の影響を避けることが可能である．
- end-to-end の監視:
  SNMP 等の監視においては router 位置までの監視しか行えない．本システムでは，

router の下流側に子機を設置したことにより実際にユーザが利用するのと同じ環境の監視を行うことが可能である．この事により，本学学内ネットワークで発生した router 位置までは異常が見られず end node 間の通信に異常が生じるタイプの障害の検出が可能となった．

- ネットワーク性能の測定:
  本来の目的とは異なるが packet の RTT を測定しているのでネットワーク性能の変化もとらえることができ，障害予知にも役立てることが可能である．

4.6 今後の課題

2002 年 8 月より本監視システムにより学内ネットワークの監視を行っている．監視については安定して行えているが，実際に長期間運用することにより幾つかの問題点も明らかとなってきている．

1. ネットワーク情報の変更への対応:
   AppleTalk においては通信相手の特定が名前ベースで行われるため，seed router や子機のリブートに際し node number が変化することがある．現在の実装では，この変更に自動的に追随しない[1]ため，関連する機器のリブートがあった場合，名前解決を行うようプログラムに signal を送る必要がある．

2. ログファイルの保存方法:
   必要十分な情報のみを記録するように設計を行ったが，監視を長期間にわたって行った結果，ログファイルが多数生成される事となった．今後は mrtg[4] 等に見られるような，経過時間に応じて情報を圧縮するような機能を実装することを考える必要がある．

3. ユーザインタフェースの開発:
   現在の実装は，各時刻におけるすべてのプロトコルの RTT を記録するとともに指定した時間幅での統計情報を出力している．これらのデータを適切に用いればネットワークの状況を把握はできるのであるが，Unix like な OS に詳しくないセンター職員の場合，その処理を行う事ができない．管理者の負担を減らすためには，適切なユーザインタフェースを作成し，一般職員であっても状況把握ができるようにすることが必要である．

5 まとめ

ネットワーク機器位置より下流にマイクロサーバを設置した監視システムを従来の SNMP ベースの監視システムと相補的に利用することにより，ネットワーク障害解析に有用であることが明らかとなった．特に，ネットワーク機器に起因する障害の解析は，SNMP ベースの監視システムからでは発見しづらく，本監視システムは有用である．

今後は，本監視システムで配置した子機を利用し，モバイルエージェント技術等を用いた監視についても考えていきたい．

参考文献

[1] http://www.netbsd.org/

[2] http://sourceforge.net/projects/netatalk/

[3] http://openlab.plathome.co.jp/linux/ssdlinux.html

[4] http://ee-staff.ethz.ch/%7Eoetiker/webtools/mrtg/mrtg.html

[1] 当初の実装では，一定の間隔で名前解決を行っていたが，これにかかるコストが大きかったため行わないように変更した．