IPv4 および IPv6 に対応した時刻同期プロトコルの実装と評価 *

金子 晋丈† 市川 雄一‡ 中山 雅哉†

†東京大学大学院工学系研究科
‡日本通信機 株式会社

1. はじめに

現在，インターネットの急速な普及によるアドレス不足の解消を目指して IPv6 への移行が進められつつある．これまで IPv4 インターネット上で実現されていた様々なサービスを IPv6 上でも同様に動作させることは，IPv6 環境への移行を進める上で必要不可欠である．このようなサービスの一つとして，インターネット上での時刻配信サービスが挙げられる．インターネット上での時刻配信サービスはタイムビジネスと呼ばれ，その重要性は電子商取引や電子申請などを通して，昨今，特に高まってきている．

筆者らは，インターネットにおいて時刻配信を実現するプロトコルである NTP[1] を IPv6 上でも動作させるべく実装を行った．本実装では，より正確な時間を配信するために時刻同期 PCI 基板を用いている．この PCI 基板は，JJY/GPS 電波を受信する同期用基準信号発生器に接続されており，正確な時刻情報を維持することができる．また，本実装では，IPv6 への移行期における透過性を考慮し，IPv4 , IPv6 のどちらにも対応した実装を行っている．本稿では, 本実装の詳細を述べ, 実装した時刻同期サーバの性能を検証する．

2. 時刻同期プロトコル

2.1 プロトコル概要

NTP における時刻情報の配信は階層的になされており，階層 N の時刻同期サーバを stratum N サーバと呼んでいる (N ≧ 1)．最上位の階層に位置する stratum1 サーバは外部参照クロックに時刻を同期をしているに対し，stratum N (N>1) のサーバは一つ以上の stratum N-1 サーバからの時刻情報を収集し，サーバ選択アルゴリズムにより最も信頼性の高いサーバを選択して，時刻の同期を行う．特に stratum 1 サーバにおける時刻同期プロトコルは stratum N(N>1) と異なり，サーバ選択アルゴリズムなどを持たないため，SNTP(Simple Network Time Protocol) と呼ばれている．本稿では，IPv6 用に拡張された SNTP[2] に基づいた実装および評価を行っている．

SNTP にはマルチキャストモードとユニキャストモードがある. マルチキャストモードは, マルチキャストアドレス，もしくはブロードキャストアドレスに時刻情報を定期的に送信するモードである. 一方, ユニキャストモードは，クライアントからの時刻取得要求を受け，時刻情報をクライアントに返信するモードである．

ユニキャストモードでは, クライアントにおいて, クライアントとサーバの時刻差 (以下オフセット時間) を以下のように計算している．

offset = ((T2 - T1) + (T3 - T4)) / 2

T1-T4 は，それぞれ，

T1: クライアントにおけるリクエスト送信時刻
T2: サーバにおけるリクエスト受信時刻
T3: サーバにおけるリプライ送信時刻
T4: クライアントにおけるリプライ受信時刻

である (図 1)．オフセット時間の決定に際して，クライアントとサーバ間の片道通信時間が必要になるが，これは，クライアントとサーバ間の通信遅延がリクエスト時とリプライ時で対称になるとの前提に立ち, 往復通信時間を 2 等分した値を用いている．ただし往復通信時間は,

rtt = (T4 - T1) - (T3 - T2)

によって計算される．

2.2 これまでの時刻同期プロトコルの実装

時刻同期プロトコルの実装として ntpd/ntpdate[3] が主に用いられてきた．ntpd では，デーモンプロセス内部に PLL(Phase Locked Loop)/FLL(Frequency Locked Loop) 回路をソフトウエアで実装し，精度の高い時刻情報に高速にアクセスできる機構を持って

---

* Implementation and Evaluation of NTP Server for IPv4 and IPv6

† Kunitake Kaneko, ‡Yuichi Ichikawa, and †Masaya Nakayama

† Graduate School of Engineering, The University of Tokyo

‡ Japan Communication Equipment Co.,Ltd.

e-mail: *kaneko@cnl.k.u-tokyo.ac.jp*

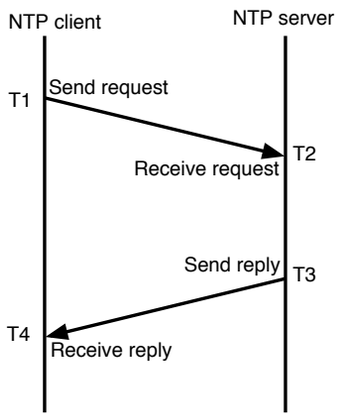

図 1 NTP で用いられるタイムスタンプ

図 2 基準信号発生器およびワークステーション

いる．また，各種の外部参照クロックを利用できる実装もあり，広範に普及している．IPv6 への対応状況は動作検証のフェーズである．

また Viagenie 社では，独自に開発した SNTP ソフトウエアを IPv6 で実運用している [4] が，ここで開発された sntpd は，サーバ端末の内部時刻をそのまま用いているだけでなく，サーバでのプロセス時間を無視した (T2=T3) 実装になっているため正確さにかける．

そこで本稿では，時刻同期 PCI 基板を用いた IPv6 に対応した SNTP サーバの実装を行う．以下では，その実装の詳細について述べる．

3. SNTP サーバの実装

3.1 ハードウエア構成

SNTP サーバを構築するためのハードウエアの性能を以下に示す．

日本通信機製 基準信号発生器

基準信号発生器は，ルビジウム発信器を用いた高精度な 10MHz 基準信号発生器である (図 2) ．ルビジウム発信器の周波数を JJY/GPS 電波を受信して再生した信号で制御し，長期的にもメンテナンスフリーの周波数安定度のよい基準信号を発生する．JJY 電波は，福島局 (40kHz) および九州局 (60kHz) に対応している．出力信号は，時刻情報と 10MHz の基準信号，1Hz50 マイクロ秒幅の 1PPS の信号を出力する．基準信号発生器の周波数確度は，$\pm 3*10^{-11}$ 未満 (100 秒) であり，周波数安定度は，$2*10^{-11}$ 未満 (1 秒) ，$2*10^{-12}$(100 秒) である．

日本通信機製 時刻同期 PCI 基板

時刻同期 PCI 基板には，基準信号発生器と接続する時刻情報用シリアルインタフェース，10MHz 入力，1PPS 入力がある．時刻同期 PCI 基板は，内部 10MHz を VCO にて発生させ 1PPS 入力信号間隔を内部 10MHz 信号で計測し 1 秒間隔との差分を計算，その差分をもとに VCO を制御し，内部 10MHz を外部 1PPS に同期させている．外部 10MHz クロックが入力されているとき，時刻同期 PCI 基板の同期精度は 10^{-11} である．

サンマイクロシステムズ製 ワークステーション

ワークステーションは，PCI インタフェースをもつ Ultra 5で，メモリは 128MB 搭載している．

3.2 ソフトウエア構成

ワークステーションでは，OS として IPv6 に対応している Soralis8 を動作させている．以下では, ワークステーション上で動作させる SNTP サーバプログラムについて述べる．

まず，SNTP サーバプログラムの設計について述べる．精度の高い SNTP サーバプログラムを作るためには，以下の 2 点に留意しなくてはならない．

(1) 正確な時刻情報をもちいること．
(2) 時刻同期プロトコルが仮定している片道通信時間が往復通信時間を単純に 2 等分した値になっていること．

SNTP サーバプログラムは，以下のように実装した．

サーバプログラムが起動されると，マルチキャストデーモン，ユニキャストデーモンを子プロセスとして起動する．マルチキャストデーモンは，以後定期的に時刻情報を配信する．ユニキャストデーモンは，サーバが保有するネットワークアドレスごとに子プロセスを生成し，それぞれクライアントからのリクエストを待つ．クライアントからのリクエストが来ると時刻同期 PCI 基板から時刻を取得 (T2) した後, リクエストのフォーマットを確認する. フォーマットの確認を受信後すぐに行っていないのは，より正確な時刻を取得するためである．フォーマットを確認後リプライメッセージを作成し，T1 および

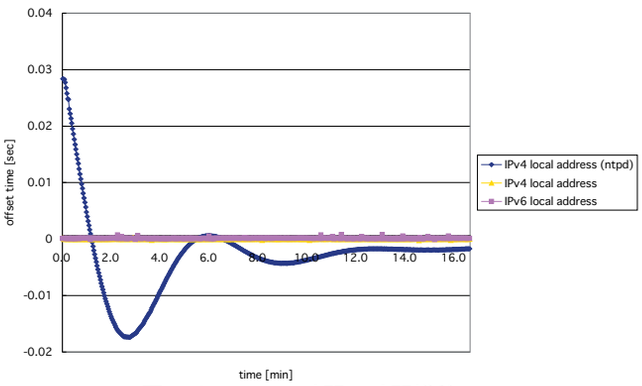

図 3 オフセット時間の時間推移

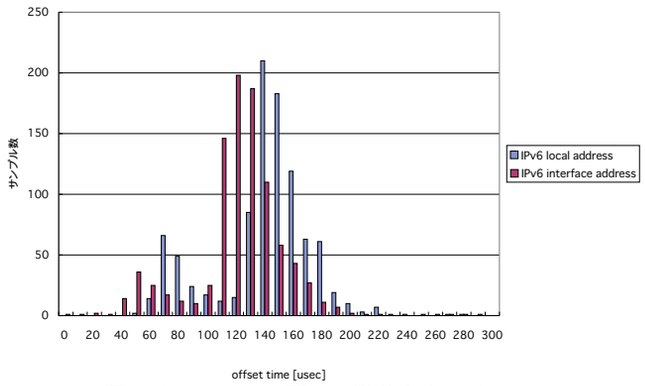

図 5 オフセット時間の度数分布 (IPv6)

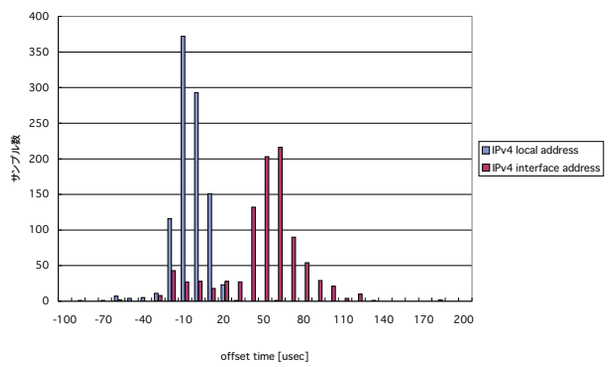

図 4 オフセット時間の度数分布 (IPv4)

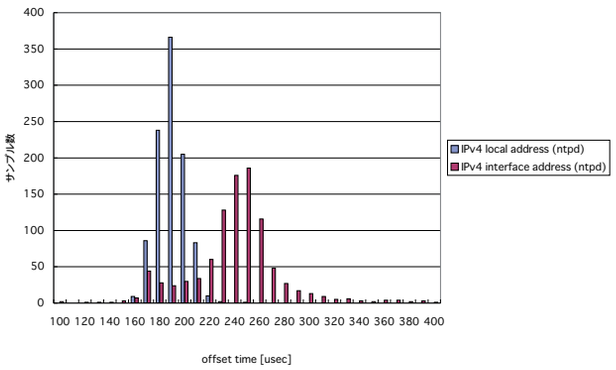

図 6 オフセット時間の度数分布 (IPv4 ntpd)

T2 を書き込む．次に，送信時刻を時刻同期 PCI 基板から取得し，リプライメッセージに T3 を書き込み，リプライを返信する．

4. SNTP サーバの動作検証

本章では，実装した SNTP サーバの性能を検証する．

4.1 時刻同期 PCI 基板を用いた SNTP サーバの検証

SNTP サーバの性能を測定するためには，正確な時刻情報を持った端末上でリクエストメッセージを作成し，SNTP サーバからのリプライメッセージに基づいてオフセット時間を計算することが必要である．そこでまず，時刻同期 PCI 基板の正確な時刻情報を用いて SNTP サーバの検証を行った．具体的には，SNTP サーバ上で，時刻同期 PCI 基板の正確な時刻情報を取得 (T1) し，これを NTP リクエストメッセージに書き込んで，ローカルホストに送信する．ローカルホストは，SNTP サーバでもあるので，メッセージを受信後，3 章で述べた手順に基づいてリプライメッセージを返信する．リプライメッセージを受信したら，時刻同期 PCI 基板から再度正確な時刻情報を取得 (T4) し，オフセット時間を計算する．このとき，理想的なオフセット時間は 0 になる．

また，本実装の検証とともに，同じワークステーション上で ntpd を SNTP サーバとして用い，比較実験を行った．

まず，SNTP サーバを起動後，2 秒間隔で時刻同期リクエストメッセージを送信し，そのオフセット時間を測定した．図 3 は，オフセット時間の時間的推移を示している．ここで，IPv4 のローカルアドレスは 127.0.0.1 を，IPv6 のローカルアドレスは ::1 を用いた．なお，現時点で ntpd は IPv6 に対応していないため，IPv4 のローカルアドレスでのみ検証を行った．図には記載していないが，IPv4 のインタフェースアドレスを用いて検証を行った際もそれぞれのローカルアドレスを用いた場合と同様の結果が得られている．

本実装では，時刻同期リクエストの度に，時刻同期 PCI 基板の時刻情報を取得するため，起動後から 0 秒に近いほぼ理想的なオフセット時間が得られている．一方，ntpd は波を打ちながら，オフセット時間が 0 秒に漸近していることが分かる．検証に用いた ntpd では，32 秒周期で時刻同期 PCI 基板の時刻情報を取得し ntpd 内部のソフトウエア PLL/FLL 回路にフィードバックをかけているため，このような現象が観測される．したがって，今回の検証と同じ条件で ntpd を動作させるときには，サーバプロ

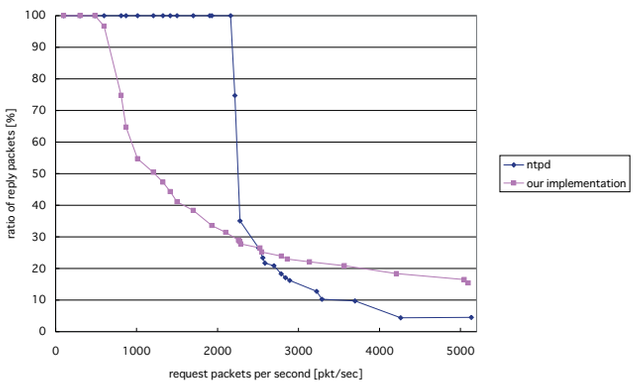

図 7 リクエスト応答率

セス起動後 15 分から 30 分経過してから利用を開始しなければ，誤った時刻が参照されることになる．

次に，ntpd のオフセット時間が安定した後，オフセット時間の分布を検証した．測定は，オフセット時間の時間的推移を検証したときと同じ方法を用いている．図 4 は本実装を用いた際の IPv4 の度数分布，図 5 は本実装を用いた際の IPv6 の度数分布，図 6 は ntpd を用いた際の IPv4 の度数分布である．それぞれ，10 マイクロ秒単位の度数分布で総サンプル数は 1000 である．

本検証では，まず 6 つのオフセット時間の平均に有意な差があるかを確認するために t- 検定を行った．その結果，これら 6 つのオフセット時間の平均は有意水準 1% で有意な差があることが分かった．いずれのオフセット時間も 0 にはならはないが，これは NTP が仮定しているリクエスト時の片道通信時間とリプライ時の片道通信時間が等しくない事を意味している．ホスト内通信においても，片道通信時間の不均衡が生じているため，ネットワークを介した時刻同期においても図 4 から図 6 が示す程度の誤差が少なくとも発生すると考えられる．またどのような IP アドレスを用いるかによっても不均衡性に影響を与えることを示している．まず IPv4 については，ローカルアドレスを用いた場合，実装を問わずインタフェースに付与されたアドレスを用いた場合に比べオフセット時間が 0 に近い．これはインタフェースアドレスを用いた際に，OS 内部のステップ数が増加したことによるものと考えられる．一方，IPv6 においては，ローカルアドレスとインタフェースアドレスの関係が逆転している現象が見受けられる．この原因は明らかではないが，IPv6 プロトコルスタックが IPv4 プロトコルスタックに後から組み込まれたことで，最適化がなされていないからであると考えられる．

4.2 スケーラビリティの検証

本実装が 1 秒間に応答できるクライアント数を検証するために，時刻同期リクエストを連続して送信した際のリプライパケットの返信数を調べた．図 7 は，クライアントが 1 秒間に生成したリクエストパケット数とリプライパケット数をまとめたものである．時刻同期リクエストの大量送信は負荷が大きいため，同一セグメント上のホストからネットワーク経由で送信した．このとき，パケットロスは検知されなかった．今回の検証において，リプライパケットの内容の確認はしていない．SNTP サーバの UDP 最大バッファサイズは，OS デフォルト値の 262144B としている．

秒間 600 パケットから 2500 パケットのリクエストが送信されたときに，ntpd を用いた場合に比べ本実装の方が返答パケットが少ない．これは本実装ではリクエストごとに時刻同期 PCI 基板にアクセスするのに対し，ntpd では時刻情報をプロセス内部に保持しているため，リプライ生成にかかる時間が短いことが理由として考えられる．リプライパケットに書き込まれているタイムスタンプをみると，プロセス時間と考えられる (T3 - T2) がおよそ 1.5msec 程度であることから，1 秒間の最大処理パケット数は，650 パケット程度となる．これにより，図 7 において 1000 パケットのリプライパケットが帰ってくるのが，600pkt/sec の頻度でリクエストパケットを出力しているときであることが裏付けられる．

5. おわりに

本稿では，時刻同期 PCI 基板を用いた IPv4/IPv6 対応の SNTP サーバの実装について述べ，特にオフセット時間の観点から評価を行った．本実装はローカルアドレスを用いた検証において，オフセット時間の平均が -10 マイクロ秒であるのに対し，ntpd では 180 マイクロ秒であり，より正確な時刻を配信できる事が確認された．また本実装では，1 秒間に 600 クライアントからのリクエストに応えることも確認された．

参考文献

[1] Mills, D., "Network Time Protocol (Version 3) specification, implementation and analysis", RFC 1305, University of Delaware, March 1992.

[2] Mills, D., "Simple Network Time Protocol (SNTP) Version 4for IPv4, IPv6 and OSI", RFC 2030, University of Delaware, October 1996.

[3] ntpd, http://www.ntp.org/

[4] sntp6, http://www.viagenie.qc.ca/en/ipv6/ntpv6/utilisation.shtml