5　各技術実証詳細

5　各技術実証詳細

e!ケアタウン創造の基盤となる IPv6 技術およびその周辺技術に関する検証を行った。この実験はすべてのモニター宅で常時稼動している機器に着目して行った。つまり、実験室内の作られた環境ではなく、実際に稼動している機器を対象としているところに大きな特徴があると言える。

これは全ての市民に対して、IT を活用した質の高い介護・看護サービスを提供するという本プロジェクトが掲げている目的に意図したもので、5 年後、10 年後の社会における「Quality of Life」を提供するために必要不可欠であると考える。

本章では、今後のインターネット環境で基盤となる IPv6 技術およびその周辺技術について、下記の 4 項目について実証を行った。

- IPv6 マイクロノード技術実証
- IPv6 ワイヤレス通信実証
- IPv6 セキュリティ通信実証
- IPv6 QoS 通信実証

5.1　IPv6 マイクロノード技術実証

5.1.1　技術内容

マイクロノードは目的を特化したインターネット接続機器の総称である。汎用型のパーソナルコンピュータ(パソコン)などの一般的なコンピュータ機器とは異なり、マイクロノードはある目的に対する特有の機能だけを提供することから、一般的に小型化が容易に可能である。このような特徴から、様々な機器をインターネットに接続するための繋ぎ込み口の役割を果たすことができる。この特徴を利用することで、パソコンなどのコンピュータ機器を一切介さずにさまざまな機器をインターネットに接続することができる。このためパソコンなどのコンピュータ機器の設置が困難な場所や用途に対するインターネット接続はマイクロノードの最も顕著な応用例と考えられる。この特徴を活かすためには、通常の家電製品と同じように電源プラグを挿すだけで利用可能な状態になることが望ましい。すなわち、一般的に面倒なネットワーク接続のための機器設定や利用者によるメンテナンスの手間を極力排除し、ネットワークサービスを受けられることが必要である。IPv6 技術におけるプラグアンドプレイ技術は、

自動的にインターネットに接続するために必要な情報を取得し自律的に利用可能な状態を作り出す技術である。したがってマイクロノードには IPv6 技術におけるプラグアンドプレイ技術が重要である。

一方、マイクロノードは特有の目的に特化している点からも、情報を提供する側、すなわちインターネットコミュニケーションにおけるサーバとしての役割を担う場合が多い。たとえば、温度センサから得られるアナログ情報を元にそのマイクロノードが設置されている場所の温度情報をリクエストに応じて提供する、と言った使い方である。しかし、既存のインターネットにおいては、グローバルアドレスを個々の機材に割り当てるのではなく、電話システムおける代表番号と内線番号のような関係を作り出す NAT(Network Address Translation)あるいは IP マスカレードと言った技術が使われている。そのため、一般的に家庭内に設置される機器へのインターネット側からのアクセスは難しい。しかし、マイクロノードの場合、原則としてインターネット上のどこからでもそのマイクロノードにアクセスできる環境がなければ、上記のような個々の機器やセンサが持つ情報を取得することはできない。そこで、NAT や IP マスカレードを一切用いず、全ての運動装置に IPv6 グローバルアドレスを割り当て、コミュニケーションの対称性のあるピア・トゥ・ピアなコミュニケーションを実現することは、このようなマイクロノード技術の利用において必要不可欠な技術である。

5.1.2 検証目標

本実証においては、以下の二つの項目について検証と評価を行う。

- プラグアンドプレイ機能
- ピア・トゥ・ピア通信機能

評価方法および評価結果は、以下の通りである。

5.1.3 プラグアンドプレイ機能

5.1.3.1. 評価方法

対象となるマイクロノードのリンクローカルアドレスおよび IPv6 ルータよりアナウンスされるプレフィクス情報から推測されるグローバルアドレスに対して、接続前より繰り返しアクセスを行い、接続した時点より最長 600 秒以内(IPv6 プロトコル仕様による最大値)にアクセスが成功できたかどうかによってプラグアンドプレイ機能が実現されているかどうか評価できる

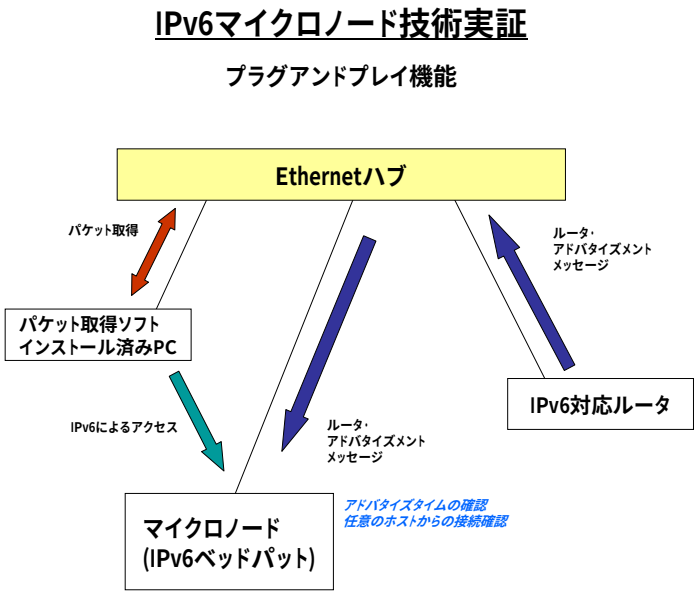

図 5.1.1 IPv6 マイクロノード技術実証（プラグアンドプレイ機能）

5.1.3.2 評価結果

実験の結果から、本プロジェクトで利用したマイクロノードが正しく動作していることを証明できた。以下では、その実験方法と実験時に得たパケットキャプチャソフトの画面を交えてどのように評価したかについて述べる。

① Ethernet ハブに IPv6 対応ルータを接続し単一セグメントのネットワークを構築した。ハブの任意のポートにパケットキャプチャソフトである Ethereal をインストールした PC を接続し、全てのパケットを取得した。

② IPv6 対応ルータで、任意の IPv6 プレフィクス情報オプションを付随したルータ・アドバタイズメントメッセージを 198 秒から 600 秒の間の任意のタイミングで送出するよう設定し、パケットキャプチャソフトウェアをインストールした PC にて IPv6 対応ルータからルータ・アドバタイズメントメッセージを送出していることを確認した。

③ IPv6 対応ルータで、ルータ・アドバタイズメントメッセージの送出を停止した後、Ethernet ハブにマイクロノードを接続した。その後、パケットキャプチャソフトウェアをインストールした PC からマイクロノードにログインし、マイクロノード

にグローバルアドレスが割り当てられていないことを確認した。図 5.1.2 の上側で囲まれた部分より、IPv6 アドレスが割り当てられていないことが分かる。

④ IPv6 対応ルータで、ルータ・アドバタイズメントメッセージの送出を開始し、同時にパケットキャプチャソフトウェアを実行する。

⑤ パケットキャプチャソフトウェアをインストールした PC にて、IPv6 対応ルータがルータ・アドバタイズメントメッセージを送信したことを確認する。図 5.1.3 より、IPv6 プレフィックスがアナウンスされていることが分かる。

⑥ 受信したマイクロノードで、IPv6 対応ルータからアナウンスされたルータ・アドバタイズメントメッセージに基づいたグローバルアドレスが割り当てられていることを確認する。図 5.1.2 の下側の囲みより、グローバルアドレスが割り当てられていることが確認できた。

⑦ パケット取得ソフトウェアをインストールした PC から、マイクロノードに割り当てられた IPv6 グローバルアドレスを指定してアクセスできることを確認する。図 5.1.4 より、ICMPv6 による通信ができていることが確認できた。

以上、①から⑦までの一連の操作の中で、図 5.1.3 の「Time」フィールド(パケットキャプチャ開始からの経過時間(秒))が 432 秒であり 600 秒以下であること、その結果から、試験を行ったマイクロノードは正しく動作していると証明された。

図 5.1.2　アドレス自動設定の前後

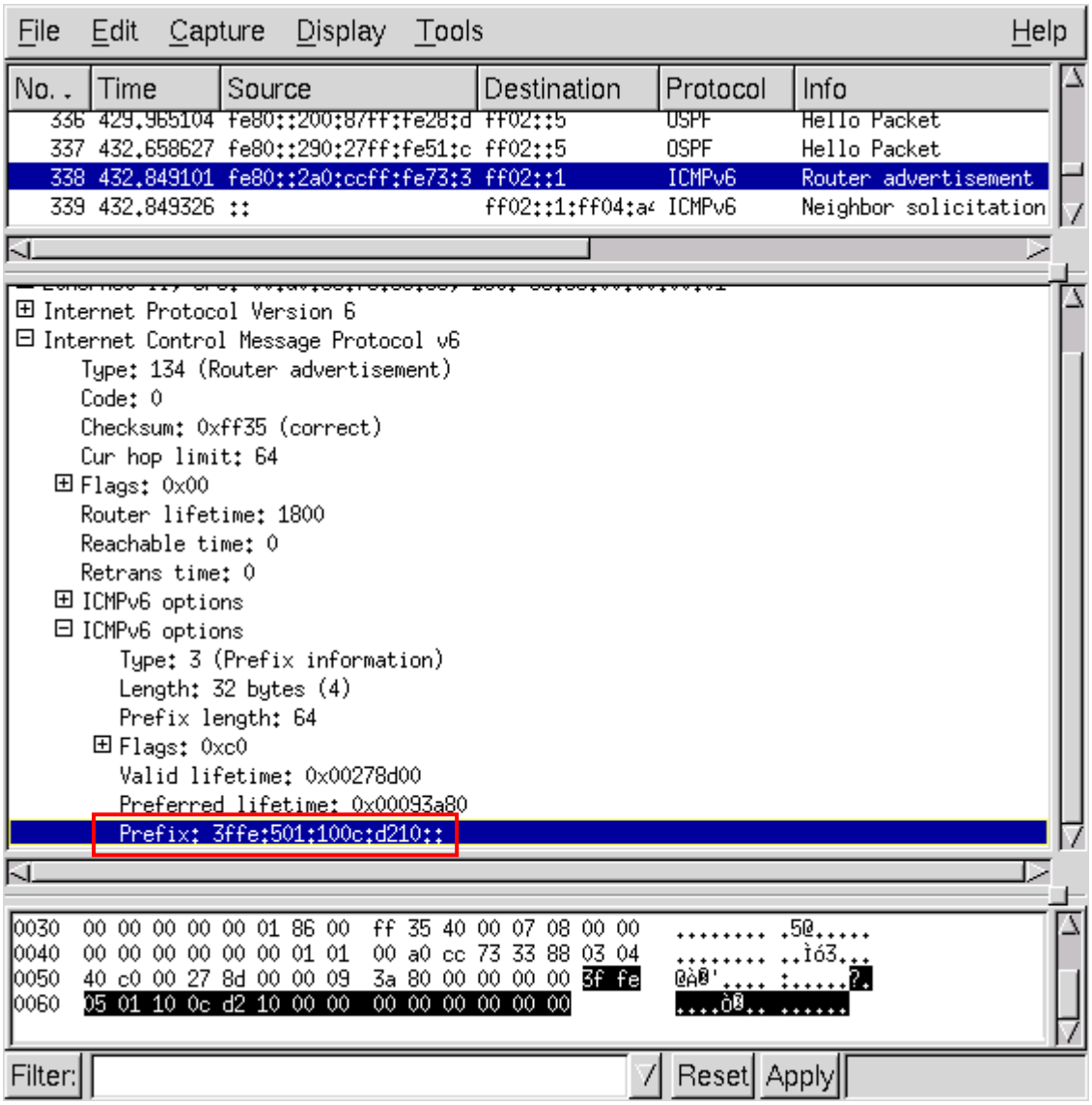

図 5.1.3　Ethereal によるパケットキャプチャ

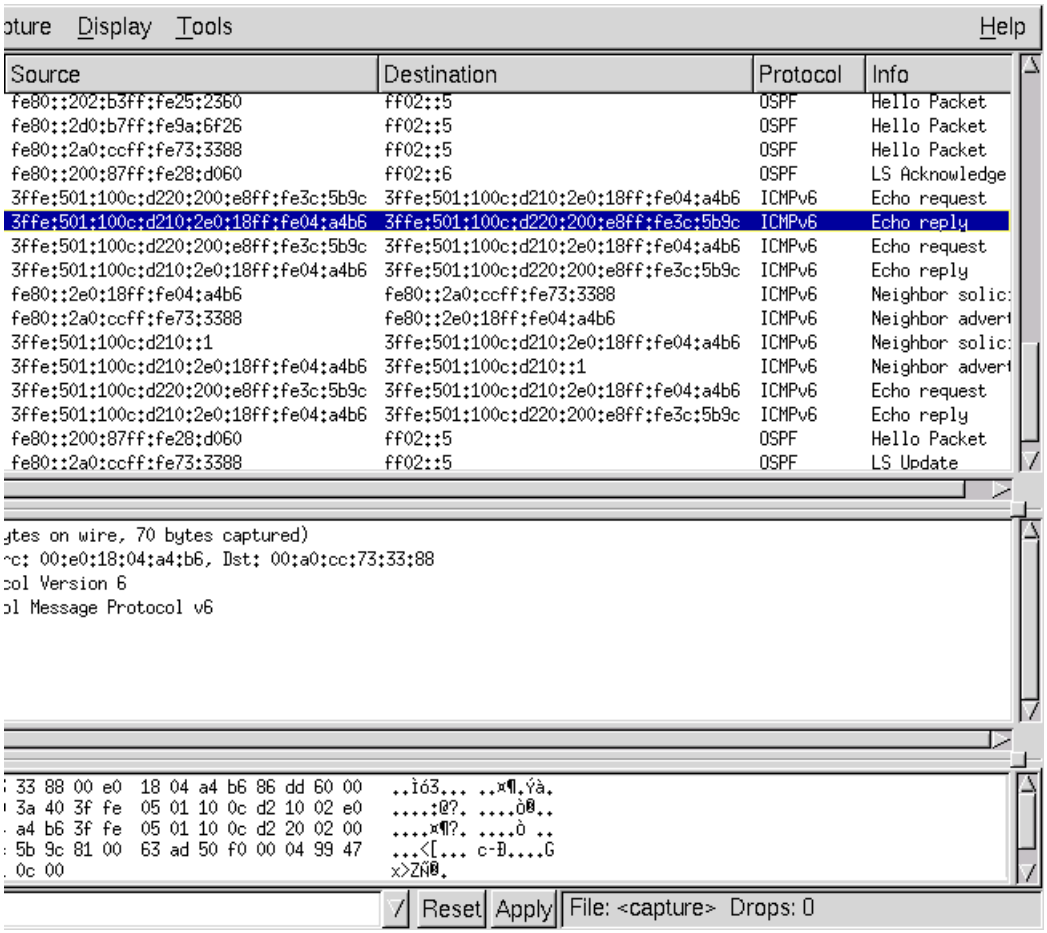

図 5.1.4　ICMPv6 メッセージが到達

5.1.4　ピア・トゥ・ピア通信機能

5.1.4.1　評価方法

インターネットに接続されたマイクロノードに対し、IPv6 グローバルアドレスを持つ任意のホストからアクセスし、正常な通信が行えたかどうかによって、ピア・トゥ・ピア通信機能を実現しているかが評価できる。

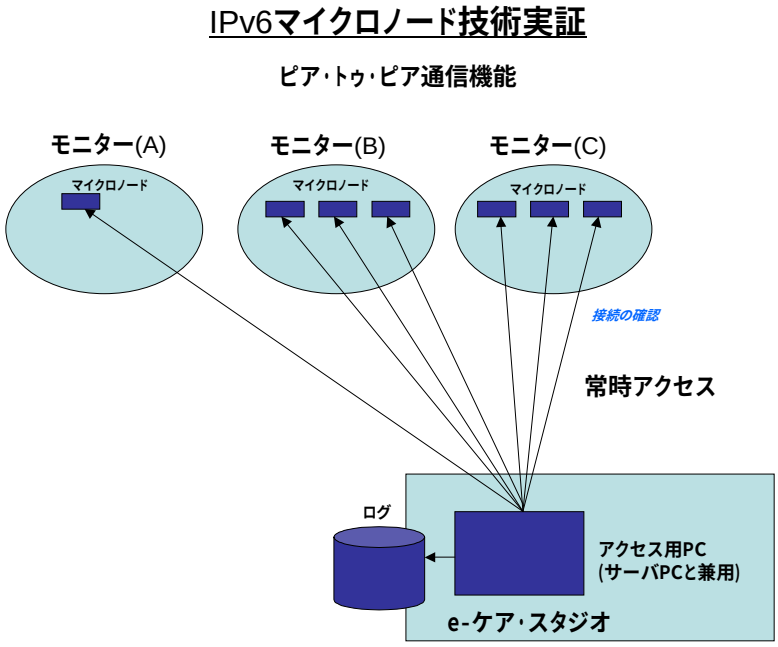

図 5.1.5　IPv6 マイクロノード技術実証（ピア・トゥ・ピア通信機能）

この時、モニター（A）で用いる IPv6 エアロバイク、モニター（B）およびモニター（C）の IPv6 ベッドパッド、IPv6 照度計のグローバルアドレスをリストアップする。いわゆる IPv6 インターネットに接続している e-ケア・スタジオおよび本プロジェクトに参加する組織よりリストアップしたマイクロノードに対してアクセスし、結果を時刻と共に記録する。実証期間中、モニター宅の任意のマイクロノードに対して継続的にアクセスを実施した。

5.1.4.2　評価結果

実証実験に参加しているモニターから無作為に選んだ IPv6 マイクロノードに対して、ping6 コマンドを利用し ICMPv6 echo request パケットを送信し、その返答として ICMPv6 echo reply パケットを受信できるかどうか実験を行った。

モニター(A)、(B)および(C)を合わせて、55 台のマイクロノードが実証実験に使われている。だが、IAF やエアロバイクのように電源の ON/OFF ができるものは任意の時刻ではつながらない可能性がある。そこで、常に稼動している IPv6 照度計にターゲットを絞り、次のような csh スクリプトを作成し実行した。

```
#!/bin/csh

foreach i ( b1 b2 b3 b4 b5 b6 b7 b8 b9 b10 c1 c2 c3 c4 c5 )
      ping6 -w -c ${I}-bio

end
```

その結果、15 台の IPv6 照度計のうち 7 台と通信ができた。その後、届かなかったノードについて電話でモニターに確認したところ、電源を抜くなどしていたことが分かった。つまり、電源の入った任意のホストには 100%通信できたことになる。

下記が ping6 コマンドの結果である。

```
[ecare@iaf-server] # ./ping6.sh
PING6(72=40+8+24 bytes) 2001:200:163:a0000::5 -->
2001:200:163:a001:130b:260:b3ff:fe69:ae68
31 bytes from 2001:200:163:a001:130b:260:b3ff:fe69:ae68: mn100064

--- b1-bio ping6 statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
PING6(72=40+8+24 bytes) 2001:200:163:a0000::5 -->
2001:200:163:a001:130c:260:b3ff:fe69:ae5f
31 bytes from 2001:200:163:a001:130c:260:b3ff:fe69:ae5f: mn10006a

--- b2-bio ping6 statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
PING6(72=40+8+24 bytes) 2001:200:163:a0000::5 -->
2001:200:163:a001:130d:260:b3ff:fe69:ae63
31 bytes from 2001:200:163:a001:130d:260:b3ff:fe69:ae63: mn100068

--- b3-bio ping6 statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
ping6: No address associated with hostname
ping6: No address associated with hostname
ping6: No address associated with hostname
PING6(72=40+8+24 bytes) 2001:200:163:a0000::5 -->
2001:200:163:a001:1311:260:b3ff:fe69:ae59
```

```
--- b7-bio ping6 statistics ---
1 packets transmitted, 0 packets received, 100% packet loss
PING6(72=40+8+24 bytes) 2001:200:163:a0000::5 -->
2001:200:163:a001:1312:260:b3ff:fe69:ae61

--- b8-bio ping6 statistics ---
1 packets transmitted, 0 packets received, 100% packet loss
PING6(72=40+8+24 bytes) 2001:200:163:a0000::5 -->
2001:200:163:a001:1313:260:b3ff:fe69:ae64

--- b9-bio ping6 statistics ---
1 packets transmitted, 0 packets received, 100% packet loss
PING6(72=40+8+24 bytes) 2001:200:163:a0000::5 -->
2001:200:163:a001:1314:260:b3ff:fe69:af01
31 bytes from 2001:200:163:a001:1314:260:b3ff:fe69:af01: mn10006b

--- b10-bio ping6 statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
PING6(72=40+8+24 bytes) 2001:200:163:a0000::5 -->
2001:200:163:a001:140b:260:b3ff:fe69:ae51

--- c1-bio ping6 statistics ---
1 packets transmitted, 0 packets received, 100% packet loss
PING6(72=40+8+24 bytes) 2001:200:163:a0000::5 -->
2001:200:163:a001:140c:260:b3ff:fe69:ae56
31 bytes from 2001:200:163:a001:140c:260:b3ff:fe69:ae56: mn100065

--- c2-bio ping6 statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
PING6(72=40+8+24 bytes) 2001:200:163:a0000::5 -->
2001:200:163:a001:140d:260:b3ff:fe69:ae8b

--- c3-bio ping6 statistics ---
1 packets transmitted, 0 packets received, 100% packet loss
PING6(72=40+8+24 bytes) 2001:200:163:a0000::5 -->
2001:200:163:a001:140e:260:b3ff:fe69:ae53
31 bytes from 2001:200:163:a001:140e:260:b3ff:fe69:ae53: mn100069
```

```
--- c4-bio ping6 statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
PING6(72=40+8+24 bytes) 2001:200:163:a0000::5 -->
2001:200:163:a001:140f:260:b3ff:fe69:ae5a
29 bytes from 2001:200:163:a001:140f:260:b3ff:fe69:ae5a: c5-bio

--- c5-bio ping6 statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
```

5.2　IPv6 ワイヤレス通信実証

5.2.1　技術内容

本実証実験で対象とするワイヤレス通信とは、無線 LAN のような狭範囲かつ広帯域な電波を用いた通信技術である。この技術は人間の移動環境をサポートするために必要不可欠である。

携帯電話とは異なり、ワイヤレス通信技術自身がパケット交換による通信を前提としていることから、インターネット技術との親和性は高いと考えられている。つまり、ワイヤレス通信技術を用いた IPv6 通信は人間の移動に非常に重要である。

しかし、一方で物理的な接続が切り替わることにより接続していたネットワークが他のネットワークに切り替わることがある。すなわち、論理的なネットワークと物理的な通信メディアが必ずしも一致しないことがインターネット上ではしばしば見られる。

したがって、物理的な通信メディアが切り替わった際に、論理的なネットワークの切り替えが行われ、加えて、その場合にいち早くそのネットワークに接続変更できなければならない。このことから、ネットワークの切り替えに対してスムーズに対応できる仕組みがインターネット技術に必要とされている。IPv6 におけるアドレス更新機能は、ネットワークの切り替えなどを IPv6 対応ルータから流れるルータ・アドバタイズメントメッセージを元に検出し、以前のアドレスではなく新たに取得できたアドレスを優先的に利用するため、IPv6 ワイヤレス通信において非常に重要である。

5.2.2　技術目標

本実証においては以下の二つの項目について検証と評価を行う。

- ワイヤレス通信技術を用いた IPv6 通信
- アドレス更新機能

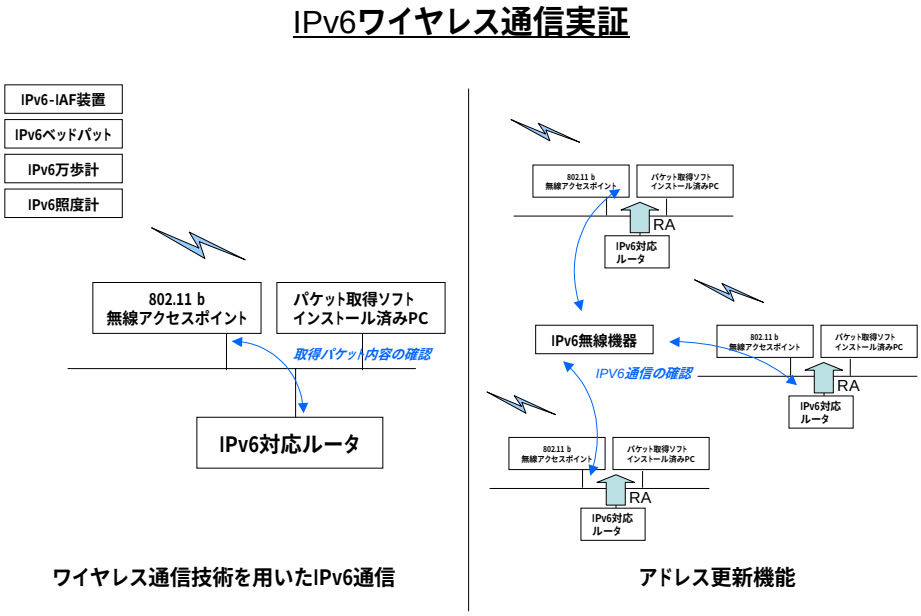

図 5.2.1 IPv6 ワイヤレス通信実証

5.2.3　ワイヤレス通信技術を用いた IPv6 通信

5.2.3.1　評価方法

対象となる IPv6 無線通信装置に対して、インターネットに接続している無線 LAN アクセスポイントを用意する。無線 LAN アクセスポイントの有線セグメントに存在する任意のルータからプレフィックス情報オプションを有効にしたルータ広告メッセージを使い、IPv6 無線通信装置にグローバルアドレスを割り当てる。しばらく時間を置いてから、同一セグメント上の任意のホストからリンクローカルで、異なるセグメント上の任意のホストからグローバルアドレスに対してアクセス可能かどうかを確認することによって、ワイヤレス通信技術を用いた IPv6 通信が実現されているかどうか評価できる。

5.2.3.2 評価結果

e-ケア・スタジオ内に Ethernet ハブを設置し、無線 LAN 基地局(アクセスポイント)を接続した。同じハブにパケットキャプチャソフトウェアをインストールした PC を接

続し、それを使って無線 LAN 上を流れる全てのパケットをキャプチャし始めた後、IPv6 無線通信機器により当該アクセスポイントに対して接続を試みる。ルータ広告メッセージのインターバルを越える十分な時間を経た後、パケットキャプチャソフトウェアを停止し、取得内容を確認したところ、IPv6 無線通信機器によって送出されたパケット見つけられた。これにより、ルータよりプレフィックス情報オプションを取得した IPv6 無線通信機器が、ワイヤレス通信技術によって有線ネットワーク同様に IPv6 通信できていることが確認できた(図 5.2.2)。

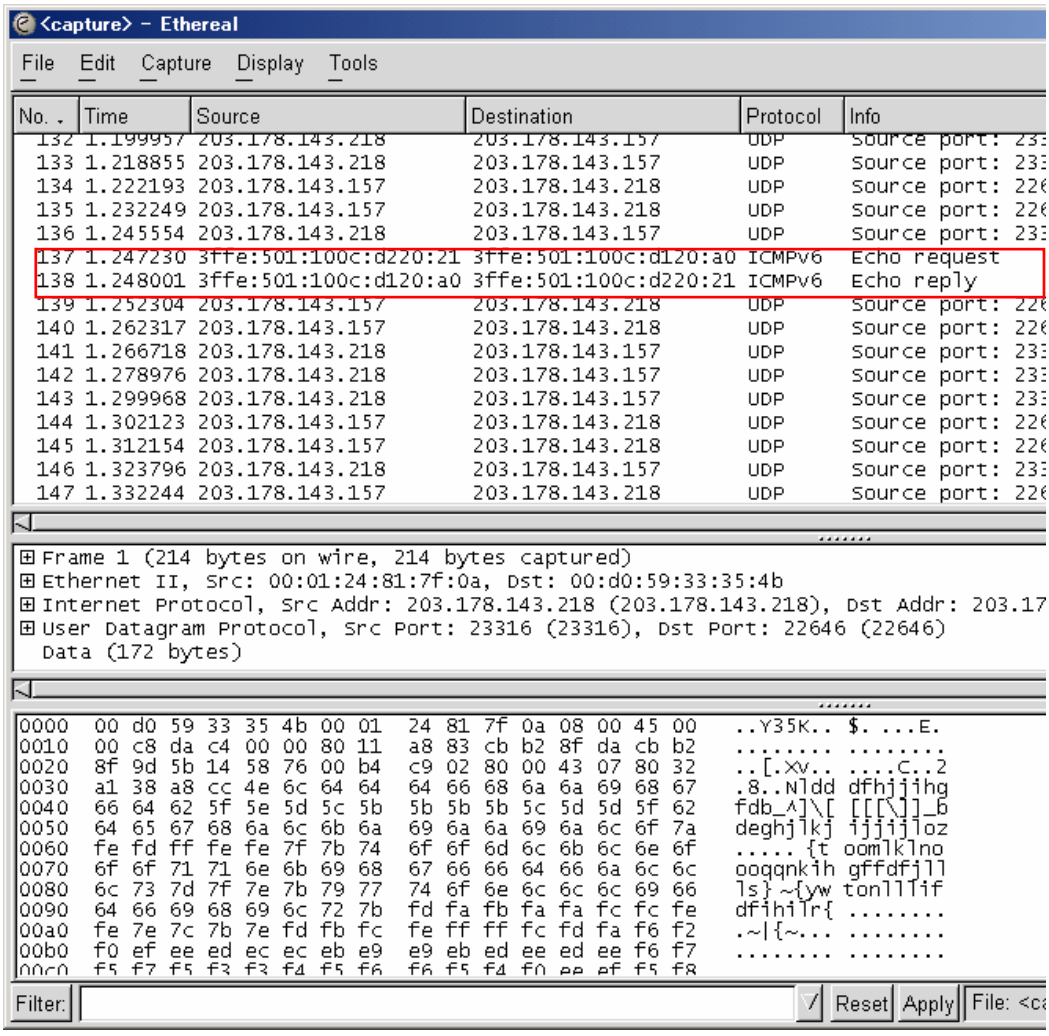

図 5.2.2 ワイヤレス通信技術を用いた IPv6 通信

5.2.4 アドレス更新機能

5.2.4.1 評価方法

二つ以上の無線 LAN に対してアクセス可能な状況において、一方の電波を遮蔽するなどして、1 つの無線 LAN でしかアクセスできない状況と、複数に対してアクセスできる状況をランダムに切り替え、IPv6 ワイヤレス通信機能を持つ機器より、外部の IPv6 ホストにアクセスできることによって、アドレス更新機能が実現されているかどうかを評価できる。

5.2.4.2 評価結果

3 台の Ethernet ハブを用意し、無線 LAN 基地局と IPv6 対応 PC をそれぞれ対になるように接続し、3つのセグメントを構築した。ぞれぞれのセグメントに IPv6 対応ルータを接続し、プレフィクス情報を含むルータ広告メッセージを送出させ、しばらく時間をおいた。

IPv6 無線通信機器を3つのアクセスポイントに十分に近いところで起動したところ、複数のグローバルアドレスを取得し、インタフェースに割り当てられていることが確認できた。

下記が ifconfig コマンドの結果である。

```
fxp0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        inet 203.178.140.171 netmask 0xfffffe0 broadcast 203.178.140.191
        inet6 fe80::202:b3ff:fe87:cba2%fxp0 prefixlen 64 scopeid 0x1
        inet6 3ffe:501:100c:d120:a00:20ff:fe80:41bf prefixlen 64
        inet6 2001:200:0:1010:a00:20ff:fe80:41bf prefixlen 64
        inet6 3ffe:517:1111:2222:1a00:20ff:fe80:41bf prefixlen 64
        ether 08:00:20:80:41:bf
        media: Ethernet autoselect (100baseTX <full-duplex>)
        status: active
```

しばらくしてから、3つのうち、2つのアクセスポイントの電源を抜くことで動作を停止させ、さらにしばらく時間をおいてから、IPv6 無線通信機器から無線 LAN で接続しているセグメントの IPv6 対応 PC に対してアクセスしたところ、通常通りのアクセスができた。

このことから、取得したアドレスの動的な更新機能が有効に働いていることが評価できた。

下記が ifconfig コマンドの結果である。

```
fxp0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        inet 203.178.140.171 netmask 0xfffffff0 broadcast 203.178.140.191
        inet6 fe80::202:b3ff:fe87:cba2%fxp0 prefixlen 64 scopeid 0x1
        inet6 3ffe:501:100c:d120:a00:20ff:fe80:41bf prefixlen 64
        inet6 2001:200:0:1010:a00:20ff:fe80:41bf prefixlen 64 deprecated
        inet6 3ffe:517:1111:2222:1a00:20ff:fe80:41bf prefixlen 64 deprecated
        ether 08:00:20:80:41:bf
        media: Ethernet autoselect (100baseTX <full-duplex>)
        status: active
```

下記が ping6 をダンプした結果である。

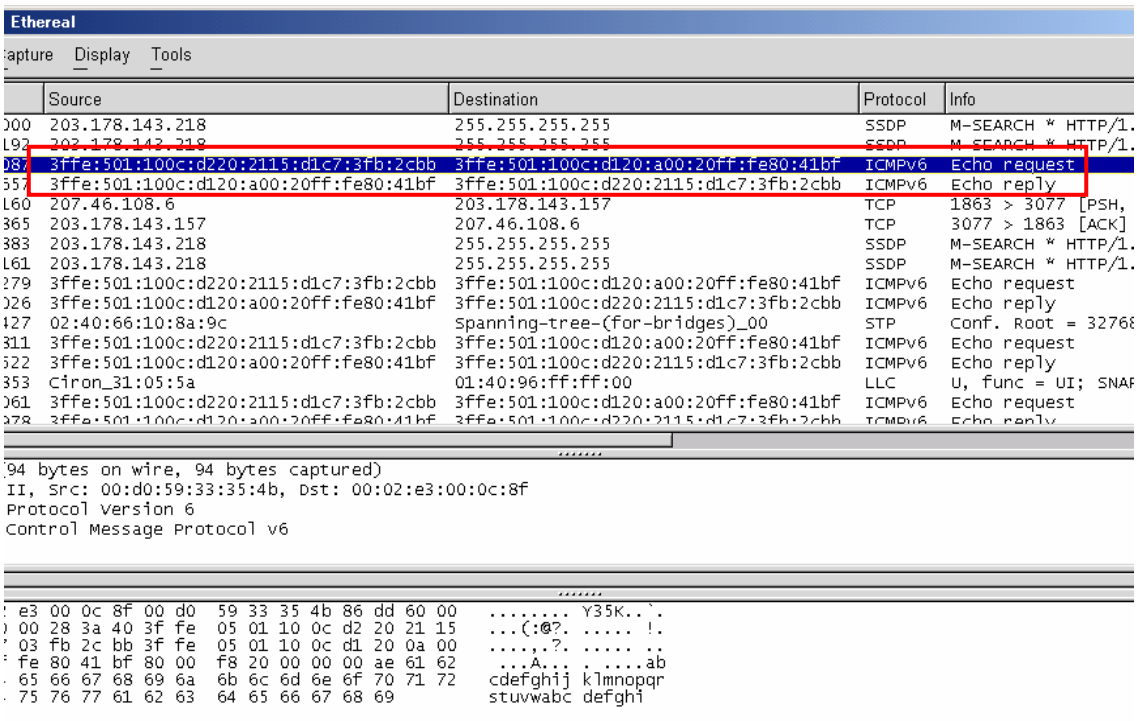

図 5.2.3 アドレス更新機能

5.3　IPv6 セキュリティ通信実証

5.3.1　実証概要

インターネット上の通信は、その内容を盗聴・改竄・なりすましができるなど安全面において問題があると言われている。IPsec は通信路の暗号化およびパケット毎の認証を行うことができ、盗聴・改竄・なりすましを防ぐことができる。しかし、一方で相手を特定することが前提となる IPsec においては、利用者の特定が最も重要な課題となる。

つまり、IPv6 セキュリティ通信には、IPsec による通信路の安全性の確保と同時に、利用者の確実な認証が必要不可欠である。

5.3.2　検証目標

以下の二つの項目について検証と評価を行う。

- IPsec 機能
- 利用者認証機能

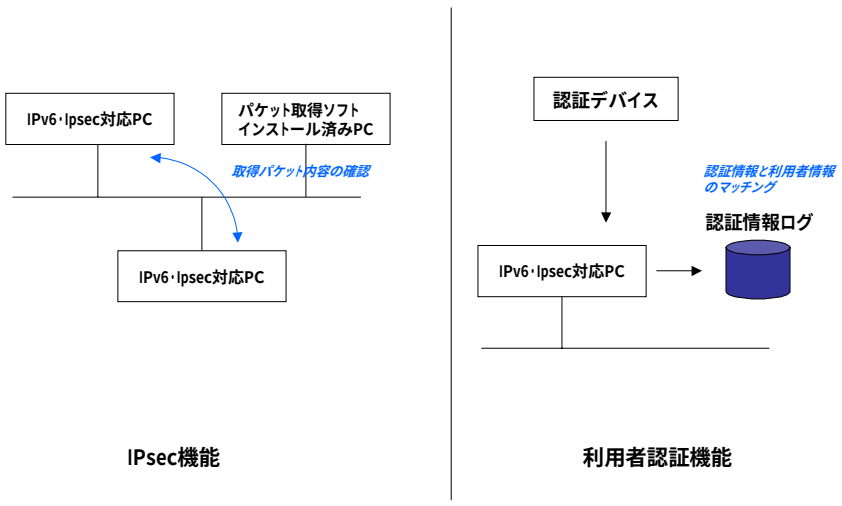

図 5.3.1 IPv6 セキュリティ通信実証

5.3.3 IPsec 機能

5.3.3.1 評価方法

2 点間で IPsec を用いた通信を行っているネットワークにおいて、通信路の途中でパケットの内容を取得し、その内容からデータの中身を再現できないことによって、IPsec 機能が実現されたことを評価できる。

5.3.3.2 評価結果

Ethernet ハブに 2 台の IPv6 および IPsec 対応 PC とパケットキャプチャソフトウェアをインストールした PC を接続した。IPsec 対応 PC は予め共有してある鍵を使い、セキュリティアソシエーションを確立させ、IPsec 通信が可能な状態にしておく。

```
# setkey -D
2001:200:0:1010:1:4a00:0:3 2001:200:0:1010:1:4a00:0:1
        esp mode=any spi=257(0x00000101) reqid=0(0x00000000)
        E: des-cbc    45535020 53412121
        seq=0x0000000c replay=0 flags=0x00000040 state=mature
        created: Mar 19 23:15:23 2003     current: Mar 19 23:31:41 2003
        diff: 978(s)      hard: 0(s)          soft: 0(s)
        last: Mar 19 23:16:19 2003          hard: 0(s)          soft: 0(s)
        current: 1368(bytes)      hard: 0(bytes)    soft: 0(bytes)
        allocated: 12     hard: 0 soft: 0
        sadb_seq=1 pid=20689 refcnt=2
2001:200:0:1010:1:4a00:0:1 2001:200:0:1010:1:4a00:0:3
        esp mode=any spi=256(0x00000100) reqid=0(0x00000000)
        E: des-cbc    45535020 53412121
        seq=0x00000000 replay=0 flags=0x00000040 state=mature
        created: Mar 19 23:15:23 2003     current: Mar 19 23:31:41 2003
        diff: 978(s)      hard: 0(s)          soft: 0(s)
        last: Mar 19 23:16:19 2003          hard: 0(s)          soft: 0(s)
        current: 1159(bytes)      hard: 0(bytes)    soft: 0(bytes)
        allocated: 13     hard: 0 soft: 0
        sadb_seq=0 pid=20689 refcnt=1
```

この時のセキュリティポリシーとして、IPsec ではない通信を拒否するよう設定しておく。そのあと、2 台の IPv6・IPsec 対応 PC 間で telnet コマンドなどを利用して、適当な文字列をネットワークを介して送る。この時パケットキャプチャソフトウェアで通信の内容を全て記録する。telnet コマンドを終了した後、パケットキャプチャソフトウェアを停止させ、取得した通信の内容を確認したところ、telnet コマンドに入力した文字列とは一致しなかった。このことから IPsec 機能により安全な通信路が確立されたことを確認できた。

次に IPsec を利用した場合と、利用しない場合のパケットキャプチャの結果を図示する。

[IPsec 利用前]

```
2001:200:0:1010:1:4a00:0:3.23 > 2001:200:0:1010:1:4a00:0:1.4004: P [tcp sum ok] 15
7:221(64) ack 216 win 58548 <nop,nop,timestamp 88327215 70144524> [flowlabel 0x678
ce] (len 96, hlim 64)
0x0000    6006 78ce 0060 0640 2001 0200 0000 1010          `.x..`.@........
0x0010    0001 4a00 0000 0003 2001 0200 0000 1010          ..J.............
0x0020    0001 4a00 0000 0001 0017 0fa4 6653 ae5f          ..J.........fS._
0x0030    f9cd 29f7 8018 e4b4 573e 0000 0101 080a          ..).....W>......
0x0040    0543 c42f 042e 520c 0d00 0d0a 4672 6565          .C./..R......Net
0x0050    4253 442f 6933 3836 2028 7370 6561 7273          BSD/i386.(test02
0x0060    2e69 6e74 6572 6e65 7469 7473 2e6f 7267          .e-care-project.
0x0070    2920 2874 7479 7031 290d 000d 0a0d 000d          org).(ttyp1)....
0x0080    0a6c 6f67 696e 3a20                              .login:.
```

[IPsec 利用後]

```
2001:200:0:1010:1:4a00:0:3 > 2001:200:0:1010:1:4a00:0:1: ESP(spi=0x00000101,seq=0x
a) [flowlabel 0x678d0] (len 120, hlim 64)
0x0000    6006 78d0 0078 3240 2001 0200 0000 1010          `.x..x2@........
0x0010    0001 4a00 0000 0003 2001 0200 0000 1010          ..J.............
0x0020    0001 4a00 0000 0001 0000 0101 0000 000a          ..J.............
0x0030    2f49 3f35 f8c7 dd4d 6845 eb29 507c 1105          /I?5...MhE.)P|..
0x0040    0829 d50a 50d3 fbb1 8c56 267d 2b3f 0932          .)..P....V&}+?.2
0x0050    e85f 7ca8 f849 debb 0782 5591 8497 250c          ._|..I....U...%.
0x0060    f30d 5aff d825 e107 e2af ceda 2d6e c630          ..Z..%......-n.0
0x0070    a1ef 05a4 8697 0e74 b079 b4cb eff9 386b          .......t.y....8k
0x0080    3af5 3d7c 6ed0 8dbb f2a5 f173 4059 cc00          :.=|n......s@Y..
0x0090    58db 01c9 14d5 f6d3 5ccc afec 74da 3ec7          X.......¥...t.>.
```

5.3.4　利用者認証機能

5.3.4.1　評価方法

指紋や虹彩などのバイオメトリクスを用いた認証デバイスが、誤って本人以外の者を正当な利用者だと認識しないことを確認するため、任意のタイミングで異なる利用者に認証を行ってもらう。これを実証実験期間中に繰り返し実施、その結果として一度も誤認識しないことで、利用者認証機能が実現されたと評価できる。

5.3.4.2　評価結果

ケア情報セキュリティプログラムに関連するモニターを対象に、普段利用している認証デバイスを用いる前に、他の人間に認証を行ってもらった。その結果、実証実験期間中、のべ 100 回を超える認証を行う中、一度も誤認識することはなかった。このことから、バイオメトリクスを用いた認証方式では、非常に誤りの少ない認証を行うことが可能であると評価できた（デバイス面の汚れ等が原因で否認された場合を除く）。

5.4. IPv6QoS 通信実証

5.4.1. 技術内容

近年 IP ネットワークにおいても、音声や映像を伝送する為の QoS(Quality Of Service)が求められており，ルータ内でのパケットのキューイングに関してさまざまな研究がなされている。トラヒック量の変化によるジッタやワンダに対して敏感なトラヒックを伝送するためには、各パケットに対し時刻情報を付与することによって、受信側でその情報を基に制御することによってリアルタイムトラヒックを伝送するアプローチとして RTP がある。しかし通常ルータで扱える IP 層と異なる層に時刻情報が付与されていること、さらに対象フローを識別するためには RTP で用いられる UDP のポート番号が不定なため、ステートの管理が必要であることなどから、ルータで RTP の情報を扱うことは困難である。

他方、パケットの優先度制御という観点からは, IntServ や DiffServ 等が挙げられる。IntServ は帯域と遅延を保証するように RSVP によるリソースを確保することを前提としている。しかし、エンドトゥエンドの帯域確保は運営ポリシの違いなどによって困難である場合が多い。一方、DiffServ は優先度の種類が 64 種類と限定されており、多種多様のトラヒックが流れるネットワークにおいてはきめ細やかな優先度制御が行いにくい。

こうした課題に対する解決を目指し、我々NTT 未来ねっと研究所では送達希望時刻指定転送(DESART: queuing system based on DEStination ARrival Time)を研究している。これは、ルータでハンドリングが容易な IP パケットのオプションヘッダ部分に時刻情報をスタンプし、帯域確保を行うことなく各ルータが自立的にその時刻情報を用いて多種多様なトラヒックに対してもきめ細やかな優先度制御を行うことを可能とする技術である。

DESART ではパケットの送達希望時刻を絶対時刻で各パケットにスタンプしているため，輻輳時等にパケットがルータ内に滞留することによって残時間が減少した場合、そこから導き出される優先度が自動的に上昇し優先的に送信される。また経路上の各ルータで目的地までの空間的な隔たりと送達希望時刻までの残時間を基にして優先度を自律的に計算しているため、同一の送達希望時刻が付けられていても遠方の目的地に向かうパケットが優先される。このため受信したパケットが目的地へ指定された時刻までに届くように優先度が計算され、最大限その条件を満たすように転送が行われる。この特徴を活かして送達希望時刻を適切に設定することにより、高負荷時においてもパケットの到着間隔の増大を最小限に押さえることが可能で、クライアントのバッファアンダーラン回避に有効であり、クライアントのバッファリングを最小限に押さえられる。また従来手法とは異なりエンドトゥエンドの帯域を固定的に確保することなく、対象フローが要求する帯域を他フローとの相対的な優先度によって動的に確保する特徴を持つため、ネットワーク資源を有効に利用できる。

5.4.2. 技術目標

DESART においては、前節で述べた課題を解決するために、以下の 3 点を技術目標とする。

1. ネットワーク上のルータで時刻情報を扱い優先度制御を行うことによる、リアルタイムトラヒックの伝送に適したネットワークの構築。
2. 時刻情報と目的地の情報を基に算出した優先度を用いた、きめ細やかな制御。
3. ユーザ端末での受信時、バッファアンダーランを回避し、リアルタイムトラヒックを安定に伝送するための、輻輳時におけるパケット到着間隔の増大の抑制。

5.4.3. 評価方法

実験ネットワークを図 5.4.1 に示す。

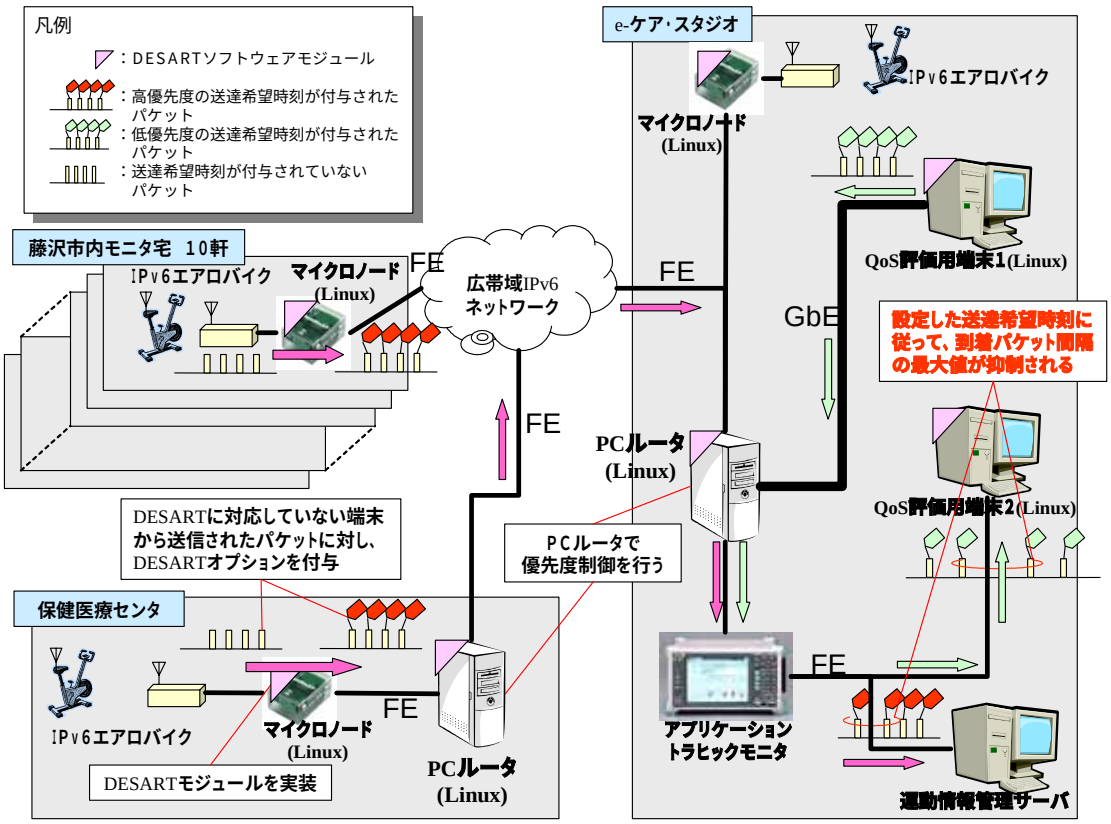

図 5.4.1　IPv6QoS 試験ネットワーク構成図

図中、QoS 評価端末 1 から QoS 評価端末 2 に向かってバックグラウンドトラヒックを流し、過負荷状態を作り出す。QoS 評価用端末 1 から PC ルータまでは 1000Base-T であり、その他はすべて 100Base-T である。QoS 評価用端末 1 及び PC ルータを図 5.4.2、e-ケア・スタジオ内マイクロノードを図 5.4.3、e-ケア・スタジオ内エアロバイクを図

5.4.4、アプリケーショントラヒックモニタ及び QoS 評価用端末 2 を図 5.4.5 にそれぞれ示す。各モニタ宅、及び保健医療センタに設置の PC ルータ、エアロバイク、マイクロノードはここで示した機器と同様のものを用いている。（保健医療センタのエアロバイクのみ、他の物より大きく仕様が若干異なる。）

図 5.4.2　ＰＣルータ（下段）及びＱｏＳ評価用端末１（上段）

図 5.4.3　マイクロノード

図 5.4.4　e-ケア・スタジオ内エアロバイク

図 5.4.5　ＰＣルータ・ＱｏＳ評価端末１用コンソール（左）ＱｏＳ評価端末２（中）トラヒックモニタ（右）

以下(1)～(4)の事項について下記(a)～(d)の観点から評価を行う。(1)、(2)では tcpdump による sniffing だけでなく、より詳細にトラヒックの様子を観測するために、NTT 未来ねっと研究所が開発し、市販測定器上に実装したアプリケーショントラヒックモニタを利用する。これは 1ms の分解能を持つリアルタイムトラヒックモニタであり、今回の実証実験のような精密な測定には非常に有効である。

評価事項

(1). DiffServ との比較検証

Linux 上での Software 実装[tc(iproute2)]を用い、DESART を用いた場合との比較を行う。

(2). 非 QoS 時との比較検証

throughput / jitter 等の評価を行う。

(3). パケットに付与された送達希望時刻の達成度

IPv6 hop-by-hop option header に付与した送達希望時刻と、受信時の現在時刻とを比較し残時間を算出する。この残時間が正の値を保持していれば、送達希望時刻は守られたことになる。

(4). パケット受信間隔のばらつきの最大値の抑制を検証

経路上のルータ前後にて tcpdump にて sniffing し、前パケットとの受信時

刻の差分を取り、評価する。

評価観点

(a). 制御からの一方向性のトラヒックについての制御性
(b). 双方向コミュニケーションにおける制御性
(c). マイクロノード上の IPv6 パケット処理能力
(d). ソフトウェアによる実現における IPv6 ルーティング制御処理能力

5.4.4. 評価結果

5.4.4.1. 制御からの一方向性のトラヒックについての制御性

一方向性のトラヒックの評価として、各マイクロノードより iperf を用いてトラヒックを UDP にて送信し、該当トラヒックに対して DESART を適用した場合、DiffServ を適用した場合、best effort の場合と 3 種について比較を行った。

また、DESART と DiffServ に関してはさらに細かく比較を行うため、測定対象トラヒックとバックグラウンドトラヒックが同一の物理キューに収まった場合について比較を行った。これは本来クラスが別のトラヒックが、悪意のあるユーザや管理者の設定ミスによって、同一キューに収められた場合を想定した検証実験である。

なお、これらすべての実験において送達希望時刻内の転送が行われ、送達希望時刻は守られていた。

```
#!/bin/sh
TC=/usr/local/v6/bin/tc

#for test 1-x and 2-x
$TC qdisc add dev eth1 handle 1:0 root dsmark indices 64 set_tc_index
$TC filter add dev eth1 parent 1:0 protocol ipv6 prio 1 tcindex mask 0xfc shift 2
$TC qdisc add dev eth1 parent 1:0 handle 2:0 cbq bandwidth 100Mbit allot 1514 cell 8
avpkt 1000 mpu 64

$TC class add dev eth1 parent 2:0 classid 2:1 cbq bandwidth 100Mbit rate 30Mbit avpkt
1000 prio 1 bounded isolated allot 1514 weight 1 maxburst 100 defmap 1
$TC qdisc add dev eth1 parent 2:1 pfifo limit 5
$TC filter add dev eth1 parent 2:0 protocol ipv6 prio 1 handle 0x2e tcindex classid
2:1 pass_on

$TC class add dev eth1 parent 2:0 classid 2:2 cbq bandwidth 100Mbit rate 10Mbit avpkt
1000 prio 2 allot 1514 weight 1 maxburst 100 borrow
$TC qdisc add dev eth1 parent 2:2 pfifo limit 5
$TC filter add dev eth1 parent 2:0 protocol ipv6 prio 2 handle 0 tcindex mask 0 classid
2:2 pass_on
```

PC ルータにおける DiffServ の設定は上記の通りであり、2 本の CBQ を設定した。DSCP が 0x2e (traffic class は 2bit 左シフトし、0xb8)のトラヒックを高優先度トラヒックとし、30Mbps の帯域が確保された優先度１のキューに入れる。このクラスのトラヒックは borrow の設定がなされていない為 30Mbps を超えることはない。2 本目のキューはその他のトラヒックが収まり、帯域は 10Mbps に設定されているが、borrow であるので、100-30=70Mbps まで最大帯域を得ることが出来る。

iperf で出力したトラヒックは、各モニタ宅から 1.5Mbps × 10 = 15Mbps。保健医療センタ及び e-ケア・スタジオより 7.5Mbps × 2 = 15Mbps の計 30Mbps である。DiffServ を用いた場合、これらのトラヒックが高優先度に設定されて送信され、優先度１のキューに収まる。マイクロノードからの送信の際 DSCP を付与するには、Linux kernel 内の DiffServ marker を用い、tc を用いて以下のように設定した。なお、destination port で該当フローを識別しているのは、設定に用いた USAGI-3.1 stable に収められている iproute2 の tc では、IPv6 の destination address での match がうまく動作しなかった為である。

```
#!/bin/sh
TC=/usr/local/v6/bin/tc
IFDEV=eth0

$TC qdisc add dev $IFDEV handle 1:0 root dsmark indices 64
$TC class change dev $IFDEV classid 1:1 dsmark mask 0x3 value 0xb8

$TC filter add dev $IFDEV parent 1:0 protocol ipv6 prio 4 u32 divisor 1
$TC filter add dev $IFDEV parent 1:0 protocol ipv6 prio 4 u32 match ip6 dport 5003 0xffff
flowid 1:1
```

図 5.4.6 に上記の例で設定した Traffic class を示す。

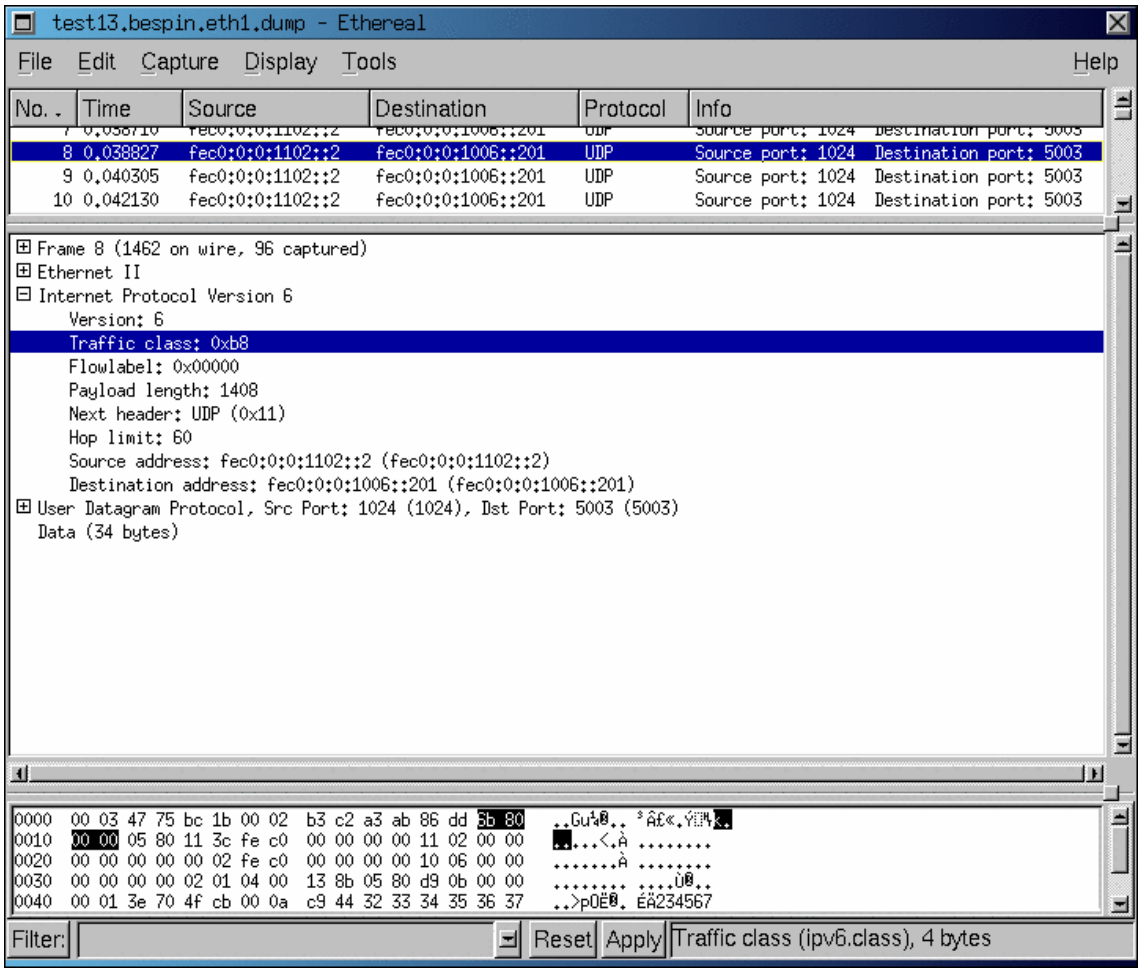

図 5.4.6　DiffServ の DSCP 設定例 (Traffic class : 0xb8 = DSCP : 0x2e)

また、DESART で設定した送達希望時刻を表 5.4.1 に示す。

表 5.4.1　設定した送達希望時刻

設置場所	モニタ A	モニタ B	モニタ C	モニタ D	モニタ E	モニタ F	モニタ G	モニタ H	モニタ I	モニタ J	スタジオ	医療センタ
時刻 [ms]	50	100	150	200	250	300	350	400	450	500	30	80

このようにして設定した送達希望時刻が付与されたヘッダの一例を図 5.4.7 に示す。

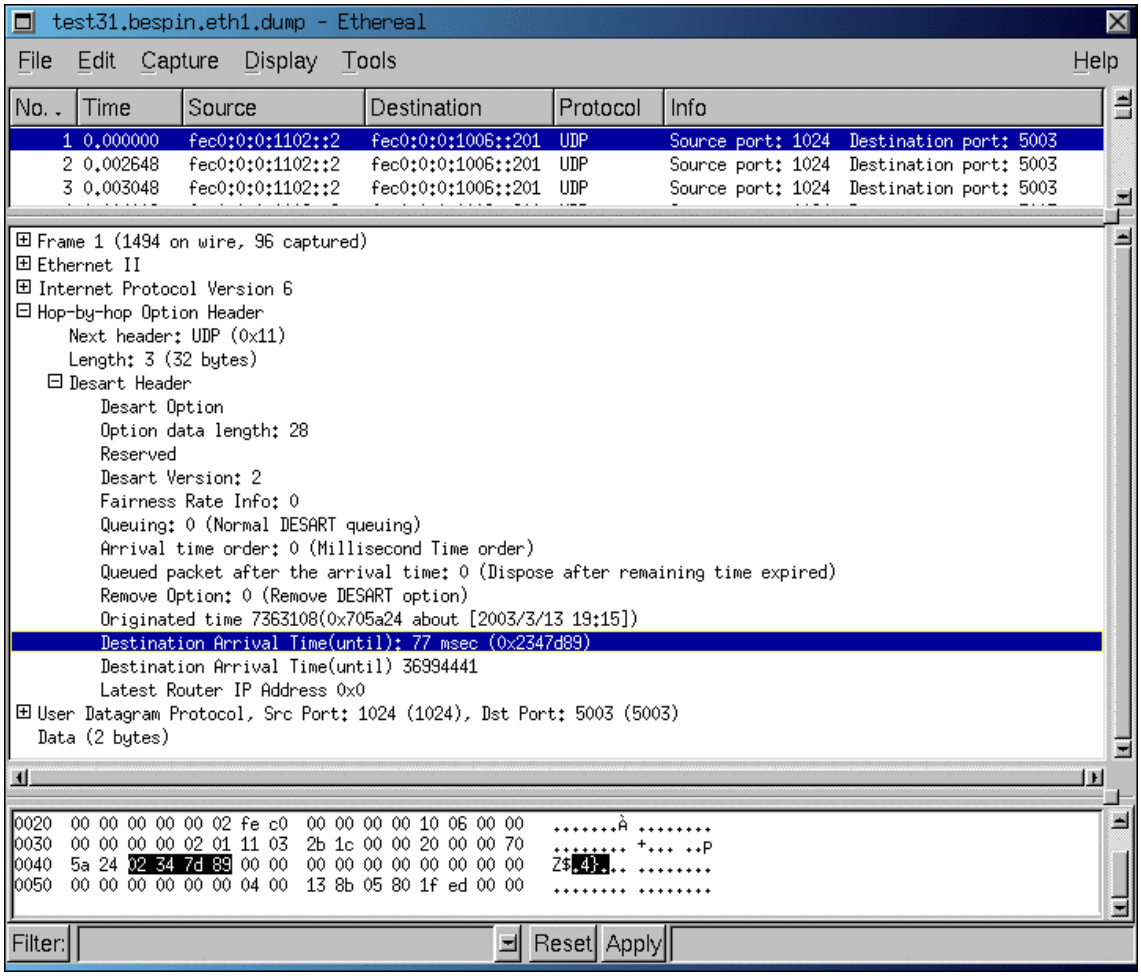

図 5.4.7　DESART オプションヘッダ付きのパケット

5.4.4.1.1.　DESART/DiffServ/Best Effort の比較

以下の例のうち DESART の場合において、バックグラウンドトラヒックとして用いるトラヒックに付与する送達希望時刻は 2000[ms]である。高優先度のキューと中優先度のキューとの切り替えは送達希望時刻の残時間が 1000[ms]を切っているか否かによって行われるため、この場合のバックグラウンドトラヒックは中優先度のキューに収まる。

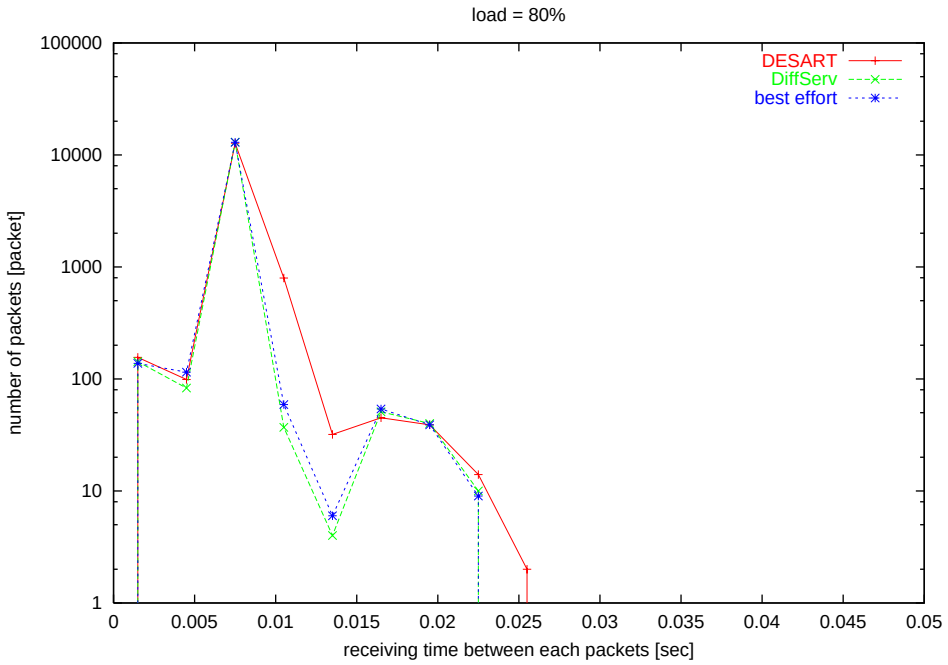

図 5.4.8　負荷 80%時のＰＣルータへの入力（モニタ宅マイクロノード）

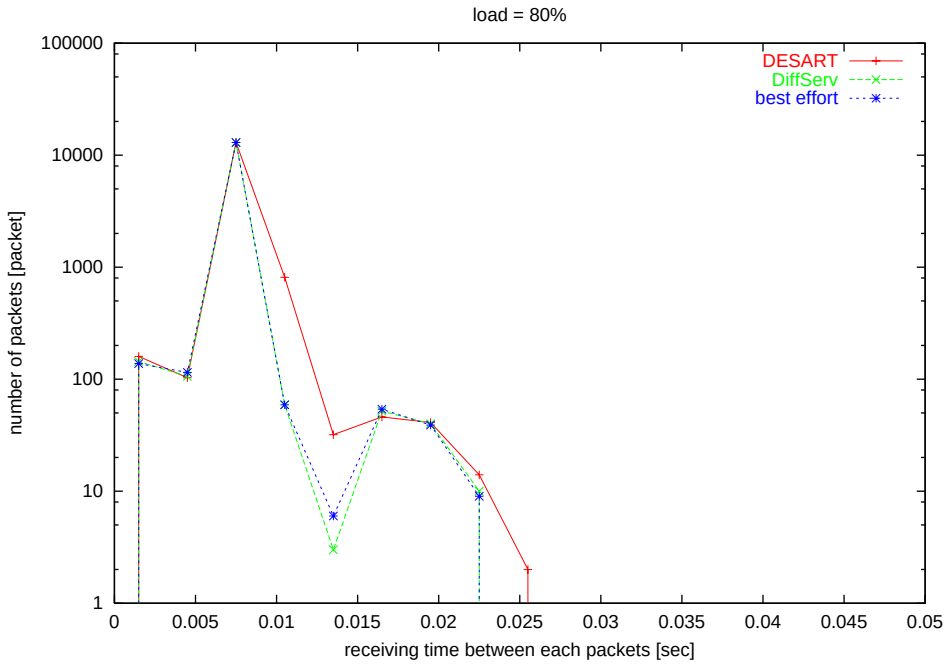

図 5.4.9　負荷 80%時の PC ルータからの出力（モニタ宅マイクロノード）

図 5.4.8 図 5.4.9 に負荷が 80%の際の PC ルータへの入力と PC ルータからの出力をモニタ宅からのパケットについてそれぞれ示す。高負荷ではないため、DiffServ と best effort での差異、さらには入力と出力の差は見られない。DESART を用いた場合に若干受信間隔が大きなパケットの分布が増大しているが、これはマイクロノードからの送信時のオプションヘッダ処理によるものと考えられる。

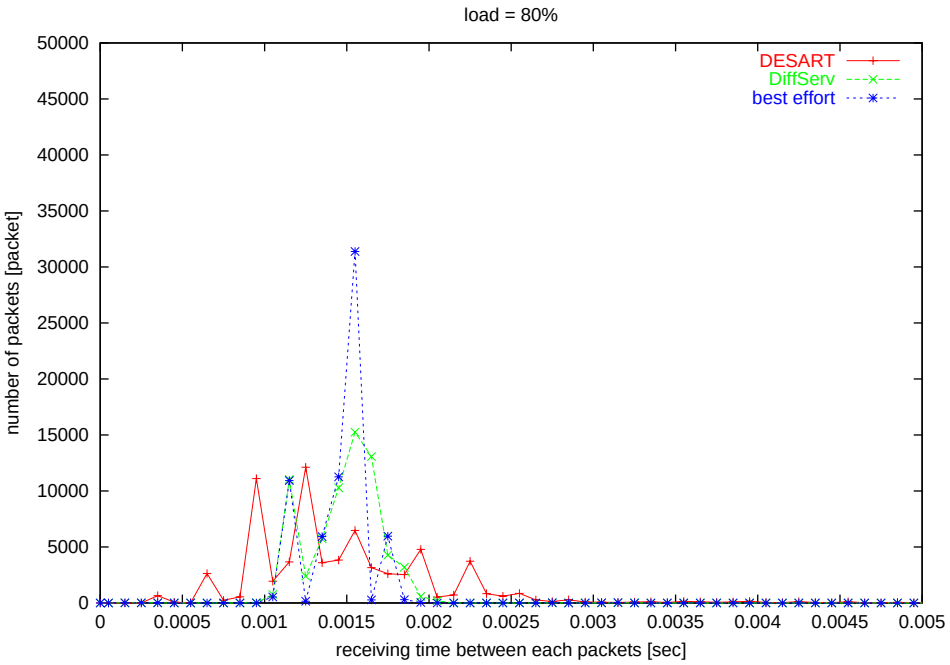

図 5.4.10　負荷 80%時の PC ルータへの入力（保健医療センタ内マイクロノード）

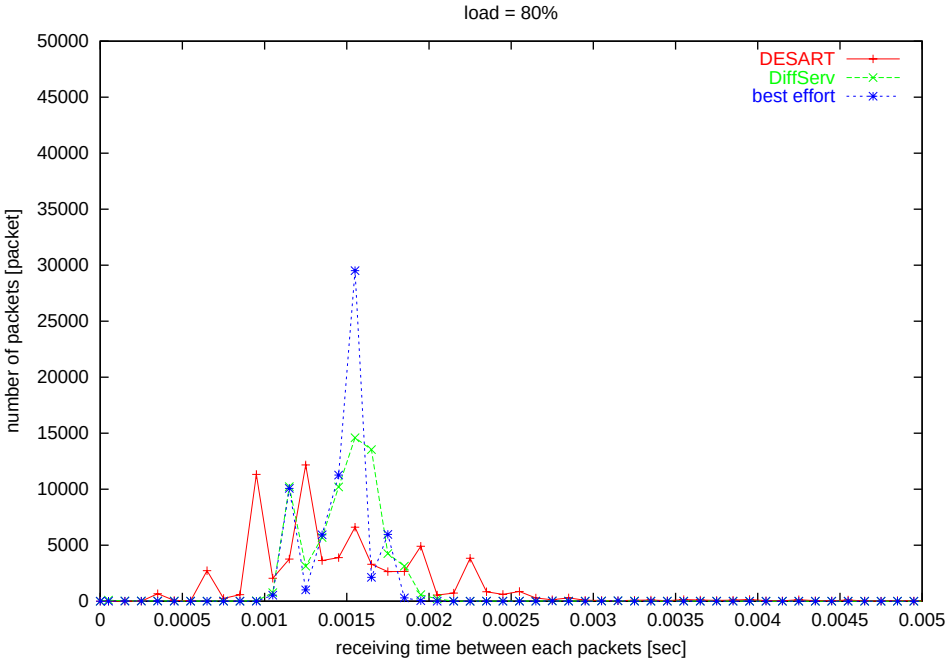

図 5.4.11　負荷 80%時の PC ルータからの出力（保健医療センタ内マイクロノード）

図 5.4.10 図 5.4.11 に同様のグラフを保健医療センタ内マイクロノードからのパケットについてそれぞれ示す。これも入力と出力に差異は見られない。ピークが移動しているのは保健医療センタ内 PC ルータを経由したためと考えられる。DESART 時にピークが広く分散して分布しているのは、マイクロノードからのパケット送信時の揺らぎに、保健医療センタ内 PC ルータの揺らぎが加わったものと考えられるが、同一の機器を用い

ているモニタ宅からのトラヒックでは発生していない事象であるため、ネットワーク構成の差も含め、今後の調査が必要である。

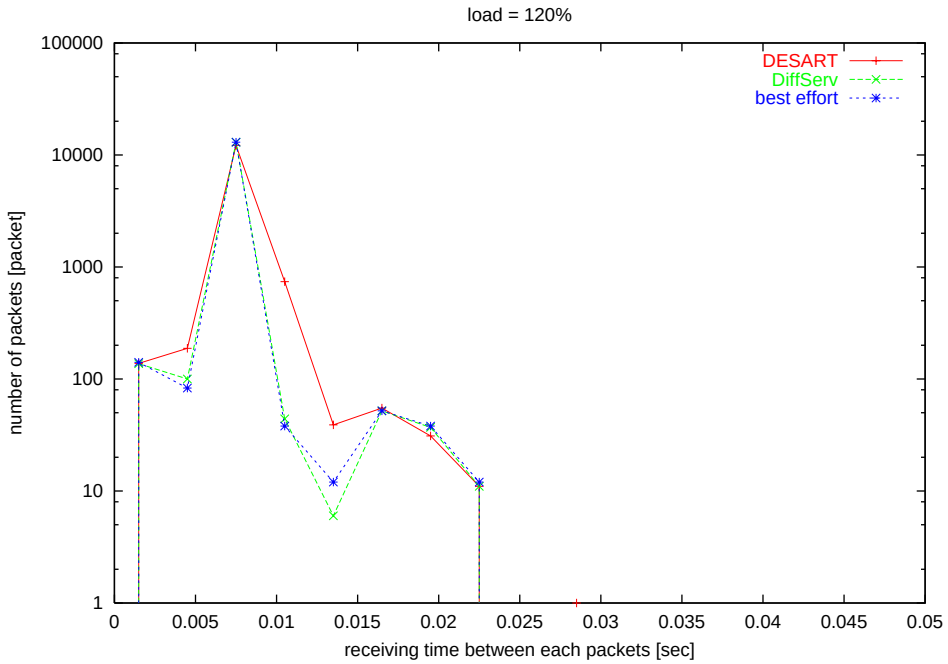

図 5.4.12　負荷 120%時の PC ルータへの入力（モニタ宅マイクロノード）

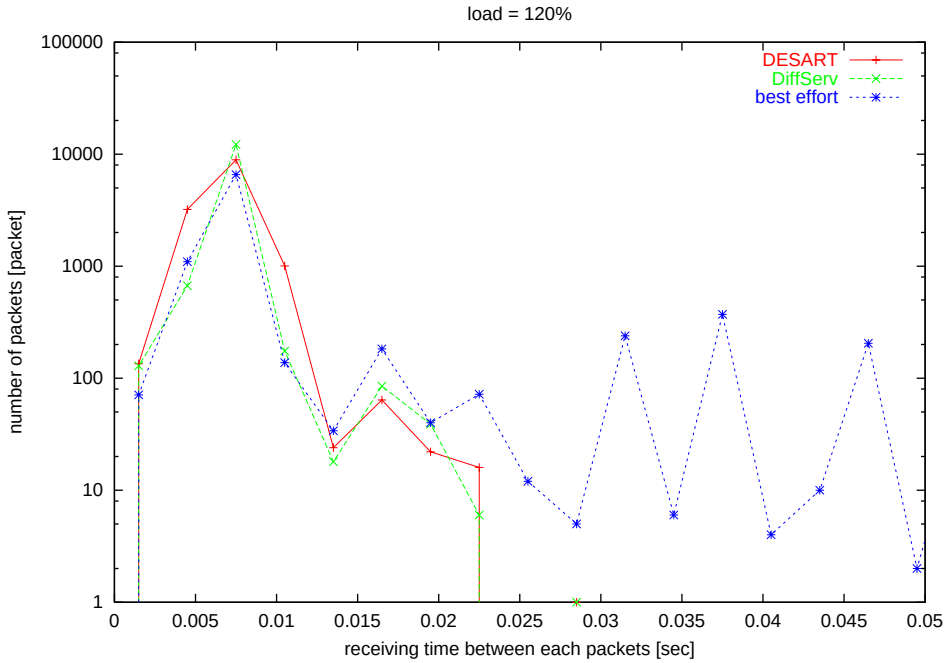

図 5.4.13　負荷 120%時の PC ルータからの出力（モニタ宅マイクロノード）

図 5.4.12 図 5.4.13 に、負荷 120%時のモニタ宅マイクロノードから送出されたトラヒックに関する PC ルータにおける入出力特性を同様に示す。best effort に比べ、DESART と DiffServ は高付加時でもパケットの受信間隔の増大が抑えられており、良好な結果が

得られている。これは、DESART と DiffServ が共に高優先度トラヒックと低優先度トラヒックのキューを分離し、高優先度トラヒックを優先度の高いキューに入れたことによる結果である。

図 5.4.14 図 5.4.15 図 5.4.16 に図 5.4.13 で示した負荷 120% 時の DESART/DiffServ/best effort の 3 方式による制御結果をアプリケーショントラヒックモニタで測定したうち、途中 1 秒間のトラヒックの状態を示したものである。

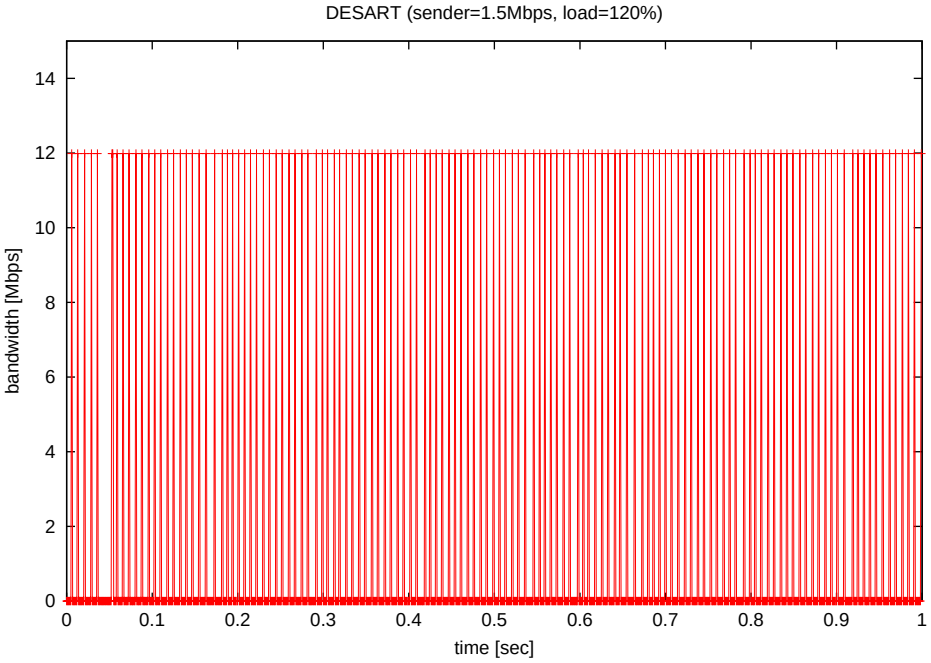

図 5.4.14 トラヒックモニタによる PC ルータからの出力測定結果(DESART)

DESART/DiffServ を用いた場合は、ほぼ等間隔できれいにトラヒックが出力されており、良好な制御結果が得られているのがわかる。平均帯域は 1.5Mbps であるが、1ms のサンプリングタイムの間に瞬時値として 12Mbps 相当のトラヒックが流れていることが確認できた。

best effort の場合は、パケット間隔がまばらになっていることがわかる。これはパケット落ちとキュー長増加に伴うレイテンシ増によってと考えられるが、等間隔に分布しているわけではなく、バースト的に送信されている箇所が存在し、その際のパケット間隔は DESART/DiffServ によって制御されている場合と同等である。これは、図 5.4.13 において best effort であっても他の方式時と同様に 0.008[sec]あたりにピークが立っていることからも確認できる。

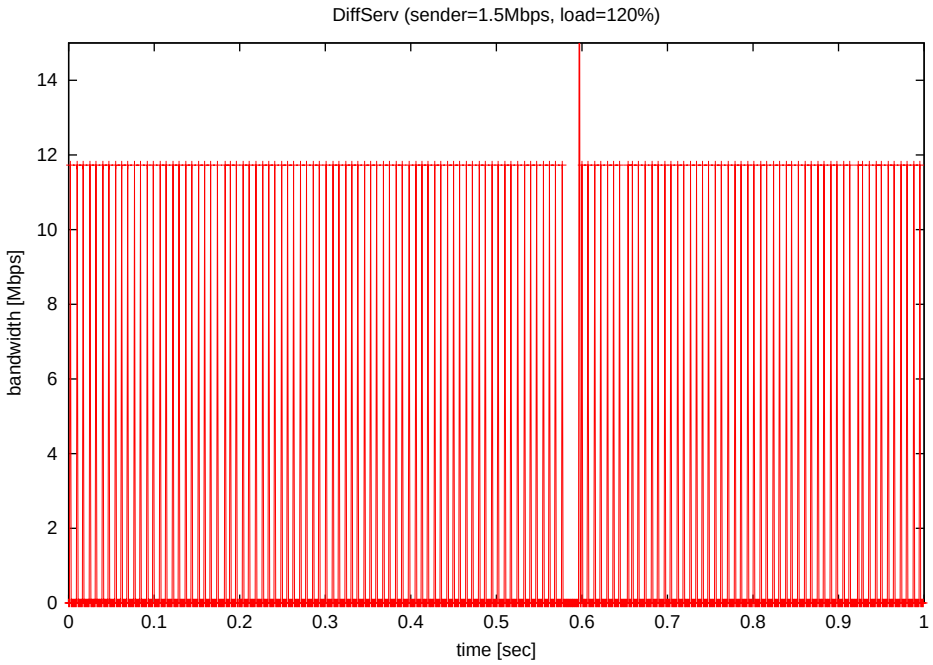

図 5.4.15　トラヒックモニタによる PC ルータからの出力測定結果(DiffServ)

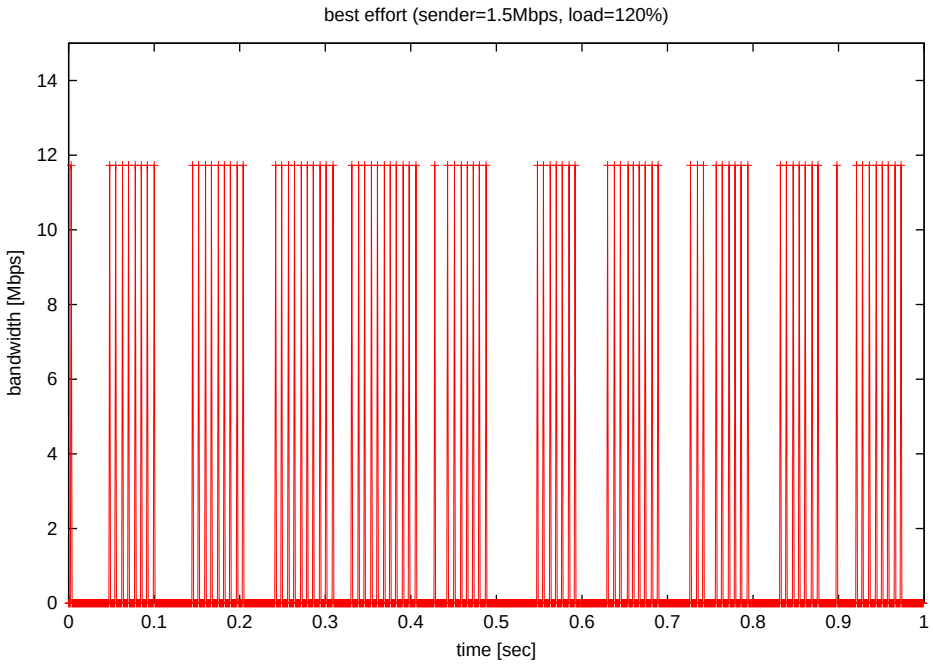

図 5.4.16　トラヒックモニタによる PC ルータからの出力測定結果

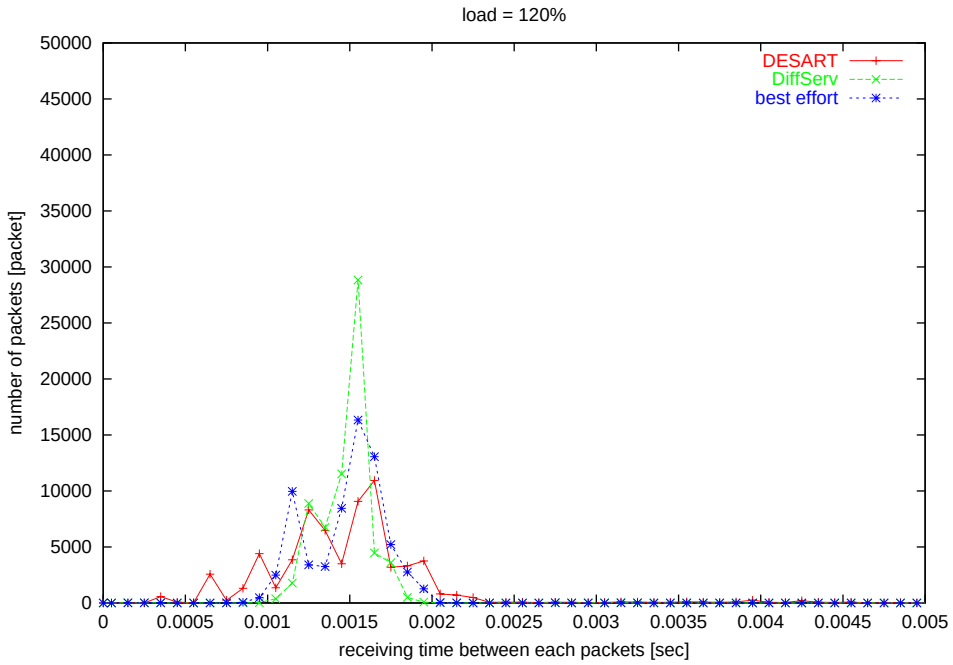

図 5.4.17　負荷 120%時の PC ルータへの入力（保健医療センタ内マイクロノード）

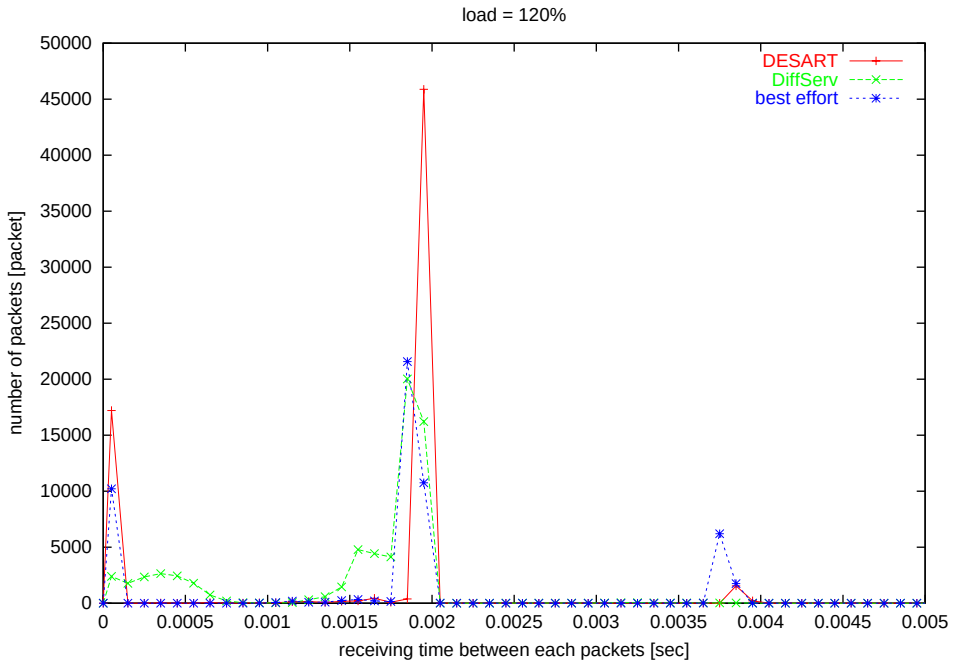

図 5.4.18　負荷 120%時の PC ルータからの出力（保健医療センタ内マイクロノード）

図 5.4.17 図 5.4.18 にモニタ宅からのものと同様のグラフを、保健医療センタ内マイクロノードから送出されたパケットについて示す。DESART は 0.0019[sec]あたりにピークが立っており、高負荷時において良好な制御性を示した。best effort は 0.0038[sec]あたりに 5000 パケットを超える分布が見られ、また DiffServ においては、0.0005[sec]付近と 0.0015[sec]付近にそれぞれ 2500 パケット、5000 パケット近い分布が見られ、ジッ

タ特性は DESART に比べ劣る。これはクラス分けによる優先度制御のみではジッタの制御性には限界があり、時刻情報を用いたジッタ制御が有効であることを意味している。また、best effort の場合が図 5.4.13 に比べ良好なのは、こちらの送信帯域は 7.5Mbps であり、よりバースト的に送信しているため、結果として通り易くなったためと考えられる。すなわち一方向性の UDP のトラヒックの場合は、より多く出した者が利を得てしまうため、公平性を考慮したキューイングが必要である。

5.4.4.1.2. DESART/DiffServ の比較

ここでは前述の通り、DESART/DiffServ それぞれについて、高優先度トラヒックと低優先度トラヒックが同一のキューに収められた場合を想定し、評価を行う。

DESART においては、バックグラウンドトラヒックに付与する送達希望時刻を 900[ms]とし、同一の測定対象の高優先度トラヒックと同一のキューに収まるよう設定した。また、DiffServ においては、ルータでの設定はそのままに、端末からの送信時において測定対象の高優先度トラヒックに対して DSCP を付与しない設定(Traffic class=DSCP=0x00)に変更した。

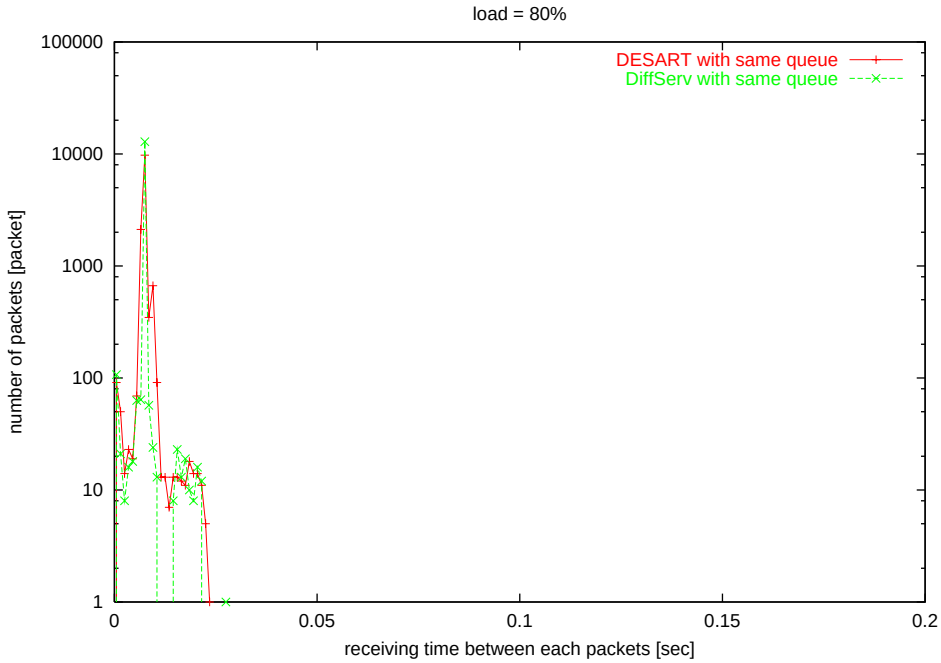

図 5.4.19　負荷 80%時の PC ルータへの入力（モニタ宅マイクロノード）

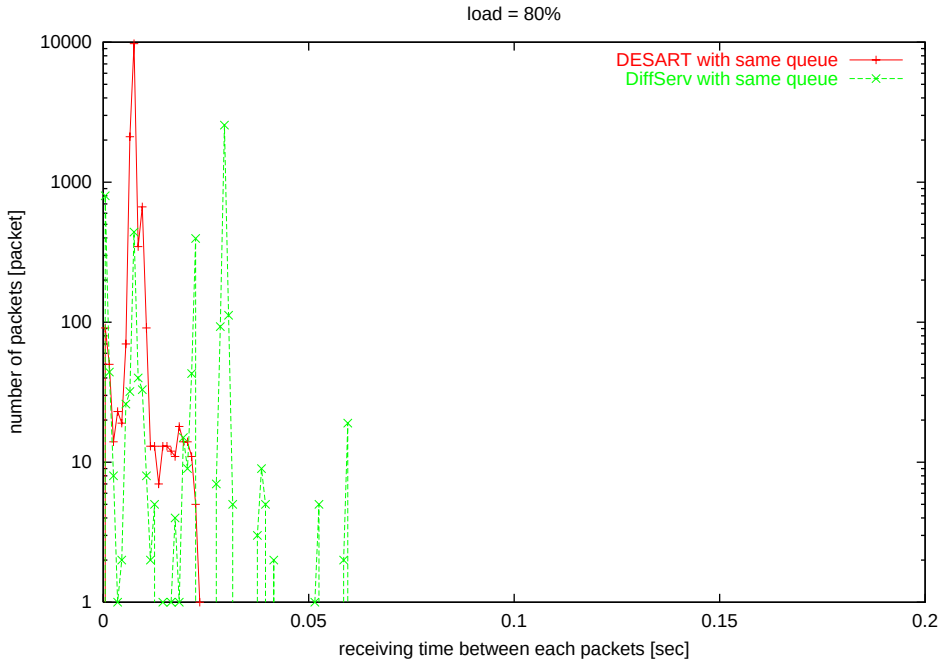

図 5.4.20　負荷 80%時の PC ルータからの出力（モニタ宅マイクロノード）

図 5.4.19 図 5.4.20 に、負荷 80%時の PC ルータの入出力関係を、モニタ宅マイクロノードからのパケットについて示す。帯域が制御されているため、DiffServ はこの状態においてもパケットの受信間隔が増大している。

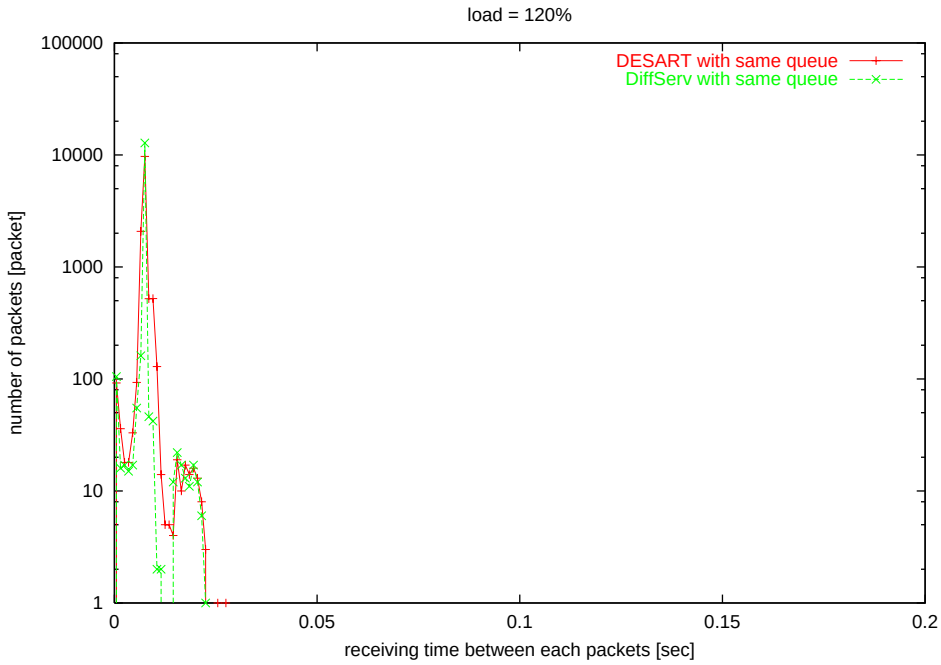

図 5.4.21　負荷 120%時の PC ルータへの入力（モニタ宅マイクロノード）

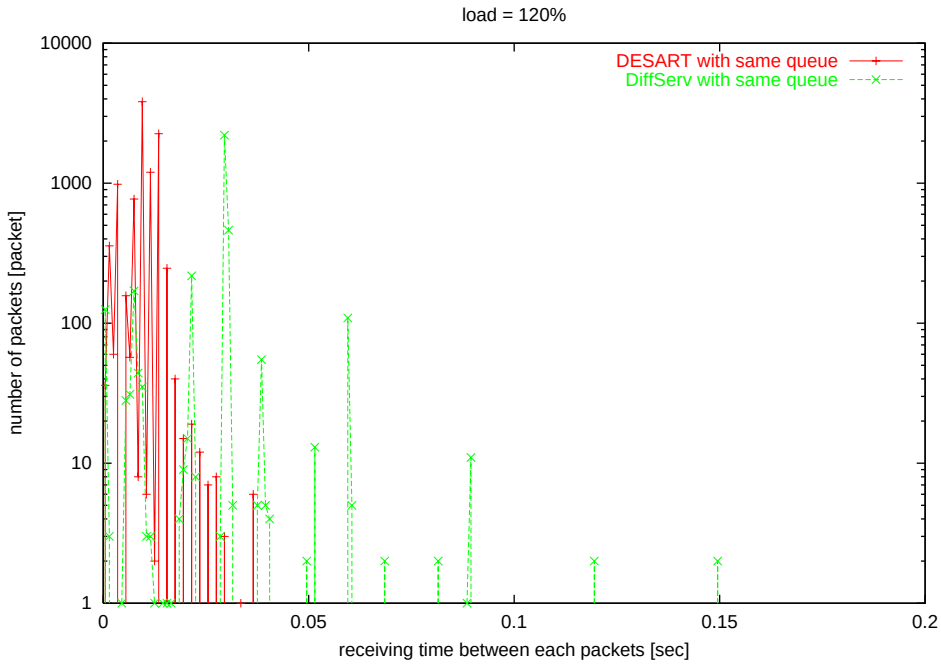

図 5.4.22　負荷 120%時の PC ルータからの出力（モニタ宅マイクロノード）

図 5.4.21 図 5.4.22 に同様のグラフを負荷 120%の場合について示す。高負荷時においても、DESART を用いた場合は同一のキュー内で優先度順に並び替えが行われ、優先度順に送信されるため、パケットの受信間隔の増大が最小限に抑えられており、良好な制御結果が得られている。

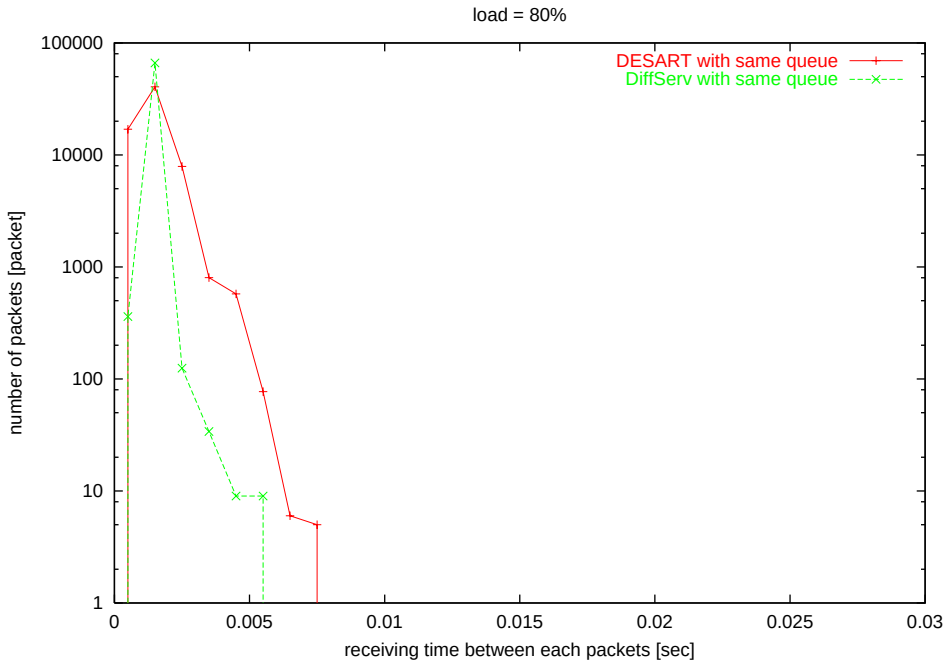

図 5.4.23　負荷 80%時の PC ルータへの入力（保健医療センタ内マイクロノード）

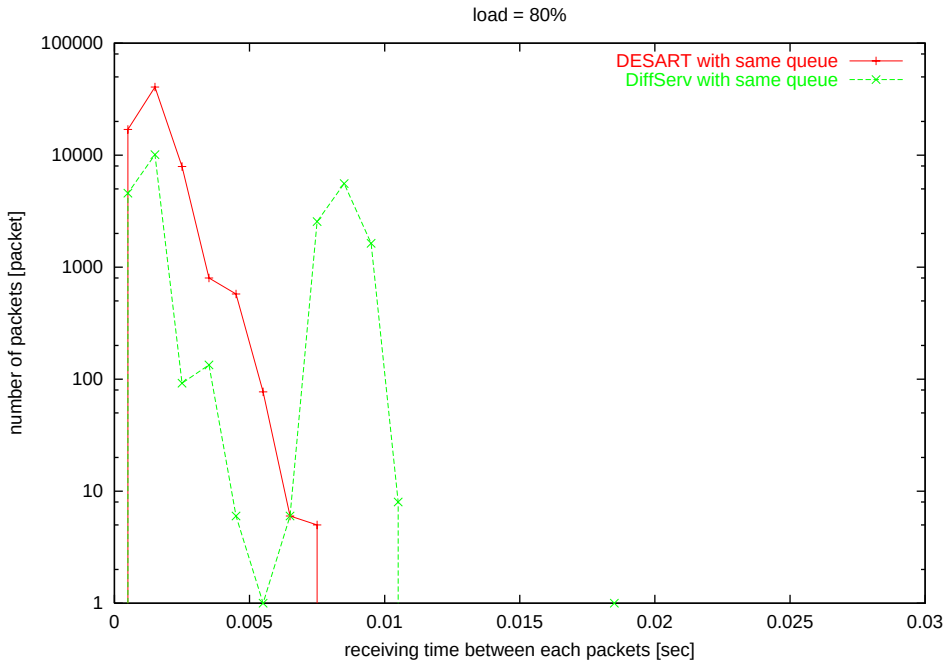

図 5.4.24　負荷 80%時の PC ルータからの出力（保健医療センタ内マイクロノード）

図 5.4.23 図 5.4.24 に負荷 80%時の PC ルータの入出力関係を、保健医療センタ内マイクロノードからのパケットについて示す。帯域制限のため、同様に DiffServ ではパケットの受信間隔が増大している。

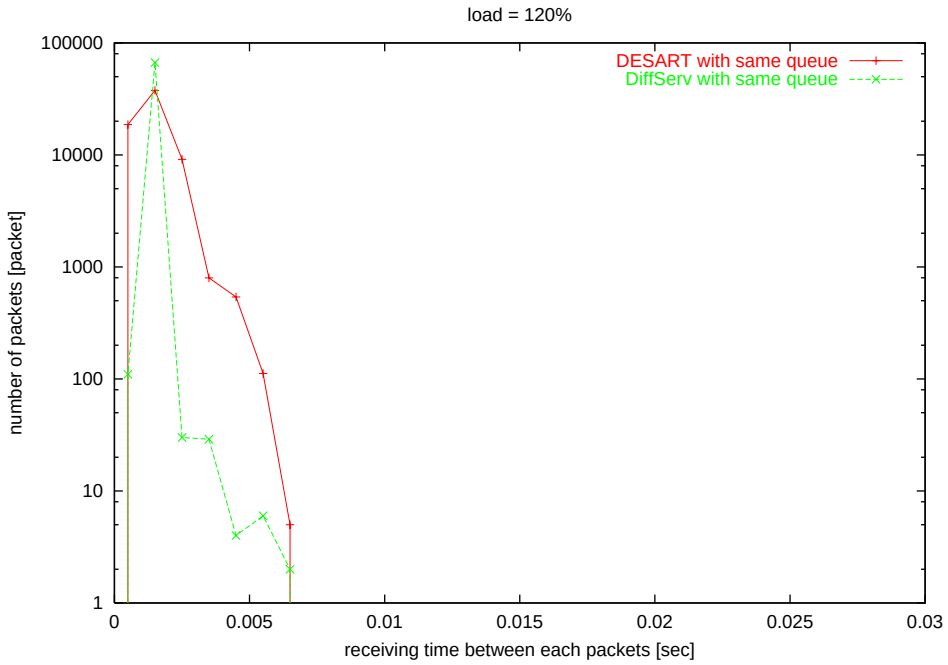

図 5.4.25　負荷 120%時の PC ルータへの入力（保健医療センタ内マイクロノード）

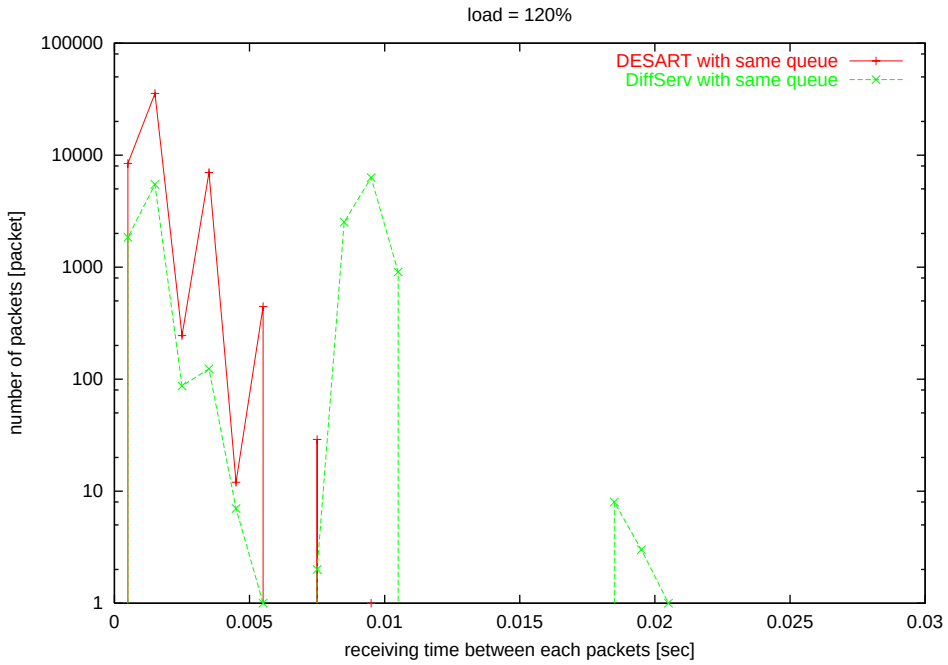

図 5.4.26　負荷 120%時の PC ルータからの出力（保健医療センタ内マイクロノード）

図 5.4.25 図 5.4.26 に、負荷が 120%の場合について同様のグラフを示す。高負荷時においても、DESART を使用することによってパケット受信間隔の増大は最小限に抑えられており、良好な制御結果が得られている。

5.4.4.2. 双方向コミュニケーションにおける制御性

双方向トラヒックの評価として、エアロバイクからのトラヒックに対して DESART を適用し、best effort の場合と比較して評価を行った。エアロバイクのトラヒックは TCP を用いており、およそ 1 秒置きに運動情報をペイロードに保持したパケットを１つ送信する。サーバは無事受信に成功すると Ack を返し、エアロバイクもその Ack を待ってから次のパケットを送信する。すなわち、高負荷時にパケットロスが生じた場合はエアロバイクからのトラヒックもそれに応じてスループットが低下する。このため、5.4.4.1 で行った実験に比べ、高負荷時の場合には PC ルータへの入力時においても best effort ではパケット数が減少している。

なお、これらすべての実験において送達希望時刻内の転送が行われ、送達希望時刻は守られていた。

エアロバイクは出力として厳密に 1 秒に一回出力されるわけではなく、以下に記すグラフの通り、1 秒を中心にして、0.96[sec]付近と 0.14[sec]付近にも分布している。

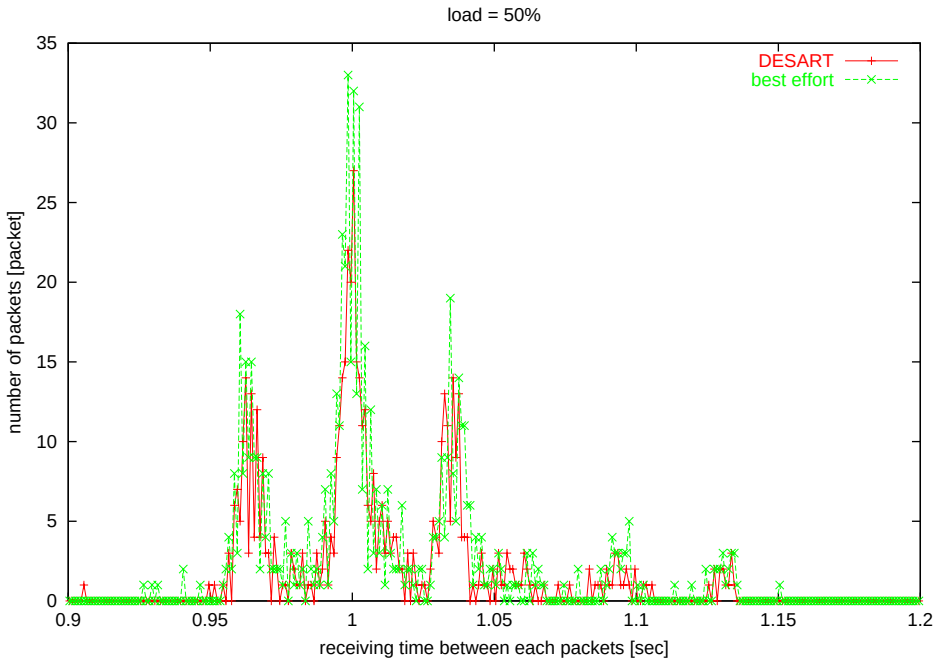

図 5.4.27　負荷 50%時のＰＣルータへの入力（モニタ宅エアロバイク）

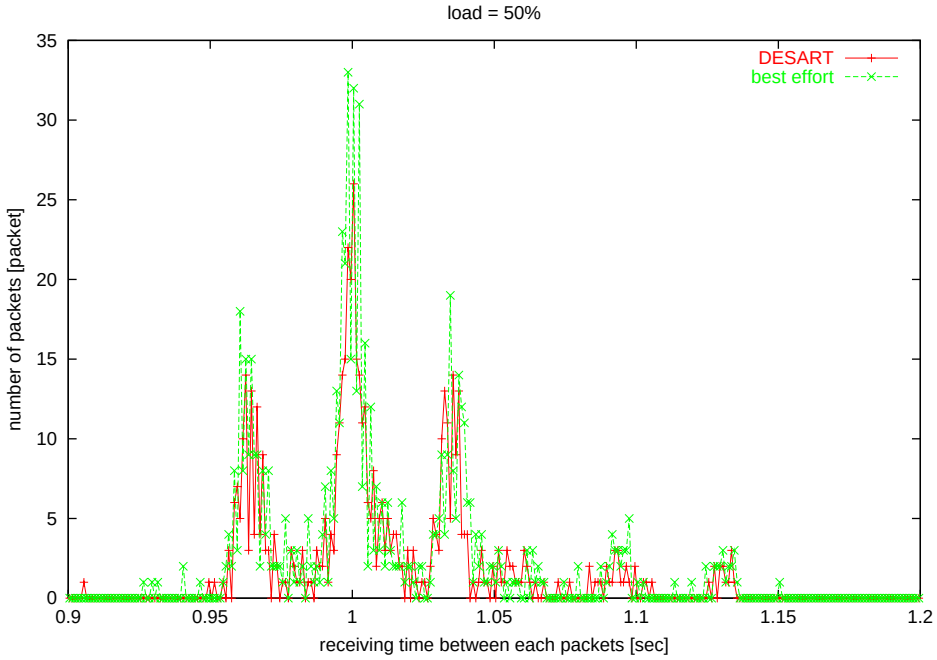

図 5.4.28　負荷 50%時のＰＣルータからの出力（モニタ宅エアロバイク）

図 5.4.28 図 5.4.29 に負荷 50%時の PC ルータの入出力関係を示す。低付加時であるので入出力関係に差異はなく、ルータへの入力パケット間隔がそのまま出力パケット間隔となっている。

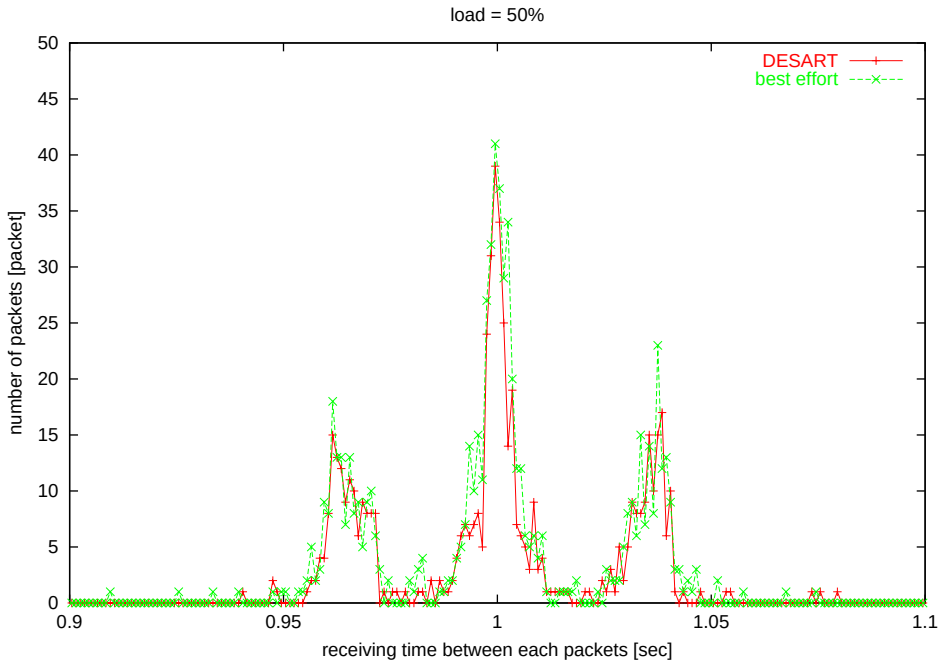

図 5.4.29　負荷 50%時のＰＣルータへの入力（e-ケア・スタジオ内エアロバイク）

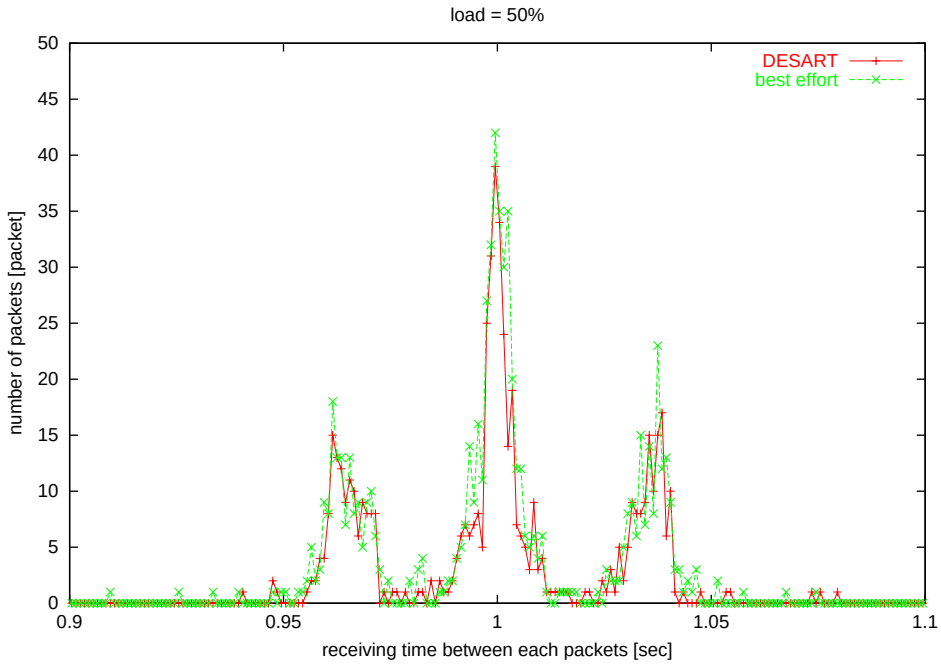

図 5.4.30　負荷 50%時のＰＣルータからの出力（e-ケア・スタジオ内エアロバイク）

図 5.4.30 図 5.4.31 に同じく負荷が 50%時について、e-ケア・スタジオ内エアロバイクからのパケットについて示す。同様に低負荷であるため入出力関係に差異は見られないが、PC ルータも e-ケア・スタジオ内に設置されているため、ネットワーク上での距離が短く 1.05[sec]を越えたあたりの分布がモニタ宅からのトラヒックに比べ少ない。

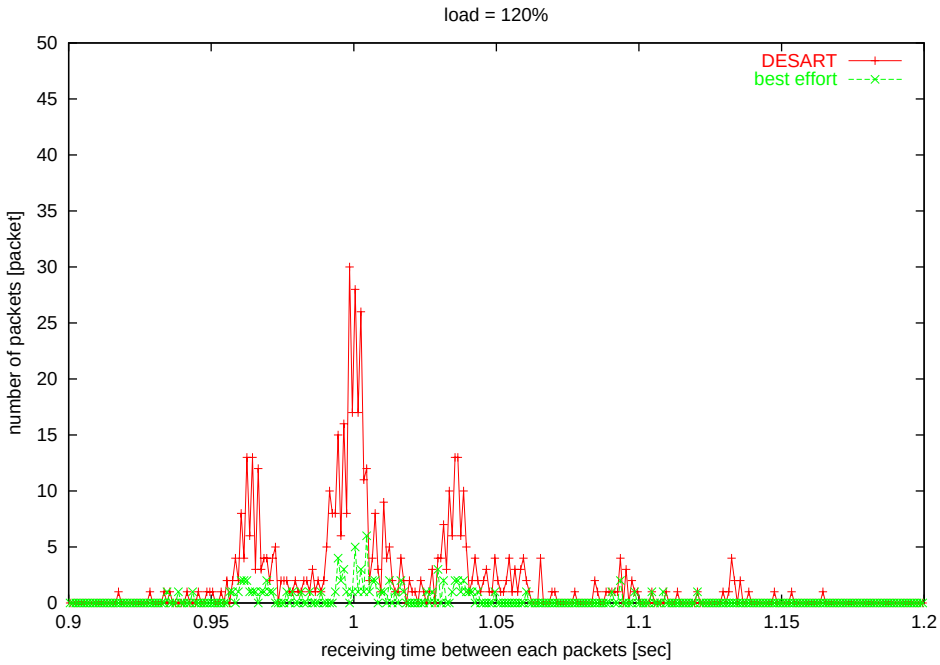

図 5.4.31　負荷 120%時のＰＣルータへの入力（モニタ宅エアロバイク）

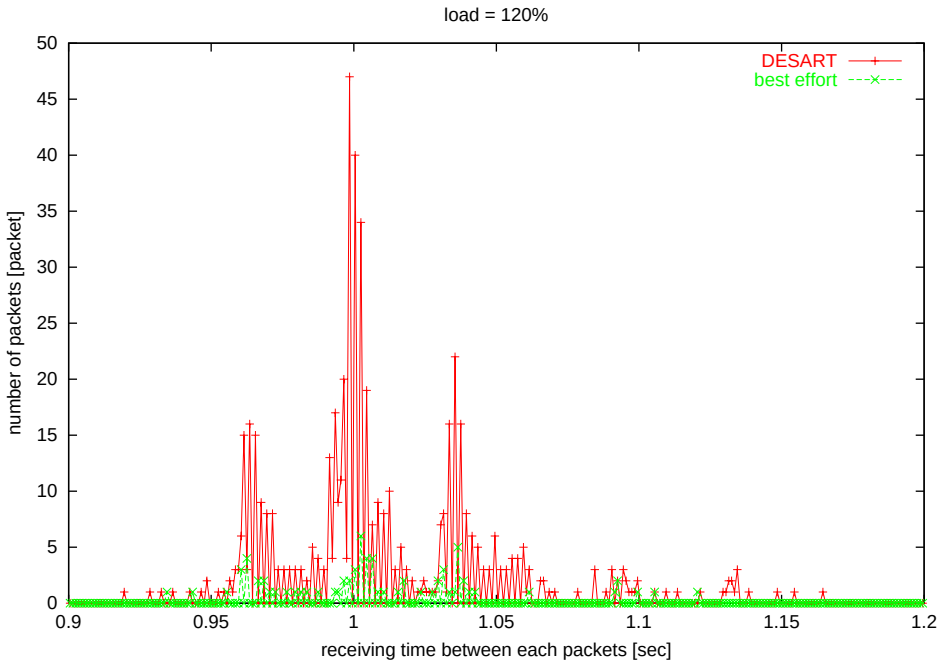

図 5.4.32　負荷 120%時のＰＣルータからの出力（モニタ宅エアロバイク）

図 5.4.31 図 5.4.32 に負荷 120%時の PC ルータの入出力関係をモニタ宅エアロバイクに関して示す。高負荷時において、送信時のトラヒックパターンに添う形で良好な制御特性が得られている。best effort に関しては、パケットロスとキュー長増加に伴うレイテンシの変動で、パケットの受信間隔が増大したものと考えられる。

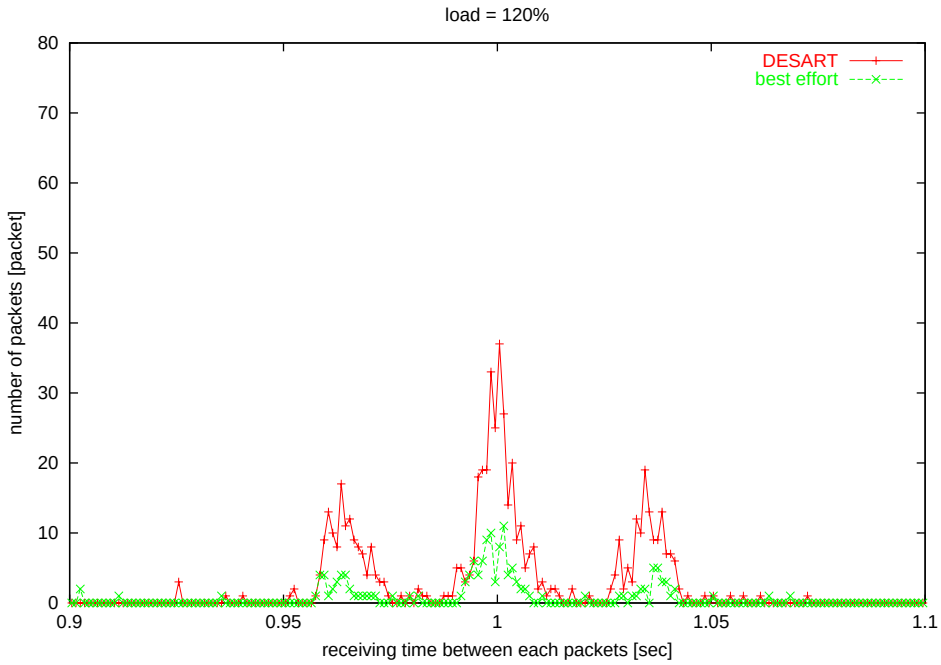

図 5.4.33　負荷 120%時のＰＣルータへの入力（e-ケア・スタジオ内エアロバイク）

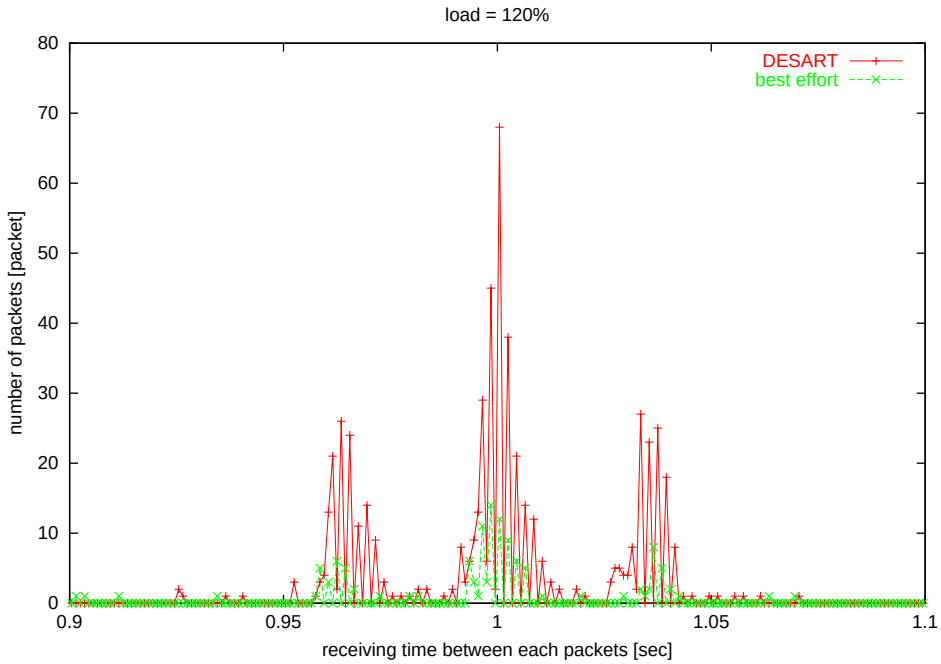

図 5.4.34　負荷 120%時のＰＣルータからの出力（e-ケア・スタジオ内エアロバイク）

図 5.4.33 図 5.4.34 に同じく負荷が 120%時について、e-ケア・スタジオ内エアロバイクからのパケットに関して示す。高負荷時において DESART を利用した場合は、送信時のトラヒックパターンに添う形で良好な制御特性が得られている。入力時に比べ出力時が離散的に分布しているのは、キュー内パケットの優先度再計算とキュー内パケットの並べ替え時にはキューに関する処理を止めて行うため、その閉塞時間により離散的にな

っていると考えられる。

5.4.4.3. マイクロノードの IPv6 パケット処理能力

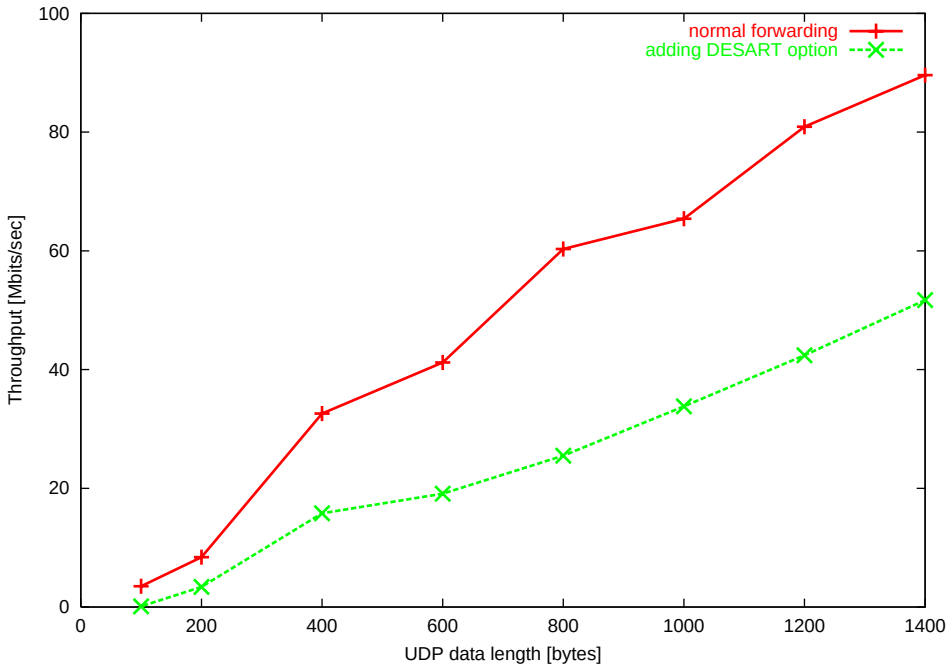

図 5.4.35 マイクロノードの IPv6 パケット処理能力

図 5.4.35 にマイクロノードを用いて IPv6 パケットをルーティングさせた場合のパフォーマンスを、通常のパケットをルーティングさせた場合と、該当フローに対し、DESART のオプションヘッダを付与した場合について示す。

マイクロノードのパケット処理能力に関しては、データ長が 1400bytes の時に 90Mbps 程度の特性が得られているので良好な結果が得られている。ソフトウェアの実現による DESART ヘッダ付与に関しては、通常のルーティング時から比較するとスループットはおよそ半分となっているが、オプションヘッダの生成を全てソフトウェアで行っていることを鑑みた場合健闘しており、マイクロノードへの実装で想定した「簡便かつ低廉な市販品に実装することによって、一般家庭への普及を容易にする」目標を達成するには、一般家庭からのアップリンクを考えると十分と言える。

5.4.5. 課題・今後の展望

時刻情報を用いた優先度制御(DESART)を行うことによって、ジッタ特性に関して送信時のトラヒックパターンに沿った形で転送が行われると共に、送信時のパケット間隔のばらつきを修正し、到着間隔の分布をグラフ化した際にピーク値を示すパケットの数が増加する特性が得られた。すなわち、時刻情報をもとに優先度制御を行うことによって、ネットワーク上でトラヒックの特性を制御しうる可能性がある。

今回の検証実験ではエアロバイクのトラヒックと、マイクロノードより送信された iperf のトラヒックを用いて行ったが、今後上記トラヒック特性の制御を目指して、ストリーミングトラヒックなどさまざまなアプリケーションによるトラヒックを制御対象とし、制御特性を検証していきたいと考える。

その際、トラヒックの振る舞いを事細かに、かつリアルタイムに把握するにはアプリケーショントラヒックモニタが有効であり、その分解能を活かせばリアルタイムストリーミングの特性把握が容易となる。

DESART 機能に関する課題として挙げられるのは、低負荷時のオプションヘッダ付与のオーバヘッドによる到着間隔のばらつき、スループットの低下である。今後ヘッダ付加時の skbuff オペレーションにおける、パフォーマンスチューニングを行い、さらには登録されたフローを参照する際の検索の高速化等でオーバヘッドそのものを抑制することと、種類や条件の異なるトラヒックが入力された際に、オーバヘッドのばらつきを最小限にするために、処理にかかる時間の平均化を進める。

DESART 機能を用いることによって、従来アプリケーションレイヤで主に行われてきた時刻制御がネットワークレイヤでも可能となり、リアルタイムトラヒックの伝送に適したネットワークの構築が可能となる。このことによって、介護・福祉分野における e-ラーニングでのリアルタイムストリーミングコンテンツや、遠隔地を結ぶ IP 上のテレビ電話などをスムーズに運用することが可能となる。