Vol.2010-UBI-26 No. 9
2010/5/28

PPv2: ナイーブな分散スマート環境間接続のための透明化アーキテクチャ

島谷宙伸[†1]　榎堀優[†2]
新井イスマイル[†3]　西尾信彦[†4]

部屋などに配置された多数の機器の連携で利用者に有用なサービスを提供する「スマート環境」が実現されつつあり，共同プロジェクトを行なう複数の組織間でスマート環境の連携を構築するなどの試みも行なわれている．そこで我々はスマート環境の特性を考慮し，連携の構築に最適な接続性を確保するトンネリング技術「PeerPool」を研究開発してきた．しかし従来の PeerPool は，ペイロード中で変換の必要があるアドレス情報の未変換，マルチキャスト・ブロードキャスト通信に非対応，また高負荷という 3 つの問題があった．本稿では，それら課題を解決する機構を備えた PPv2(PeerPool version 2) の開発し評価した結果，実用的なパフォーマンスで上記の問題を解決できることを確認した．

PPv2: A Transparent Network Architecture for Naive Inter-Smart Environment Communication

Michinobu Shimatani,[†4] Yu Enokibori,[†2]
ismail Arai[†3] and Nobuhiko Nishio[†4]

”Smart environment” which provide useful services for users by the cooperation of a lot of equipments arranged in a room. is being achieved. Furthermore,some challengers for collaborative systems between two or more organisations working on a joint project have been appeared. We have been developing PeerPool, a network tunneling technology optimised for smart spaces. However, PeerPool doesn’t support broadcast and multicast communications,auto translation of payloads which have important data of communication , and its high CPU loads.To solve these problems,we designed and implemented PPv2 and confirmed it solved all problems with the realistic performances.

1. はじめに

ユビキタスコンピューティングの研究の進展により，スマート環境と呼ばれる IP ネットワーク化された空間コンピューティング環境が家庭やオフィスに普及することが期待される．スマート環境は，空間内に遍在したセンサやデバイスがネットワークを介して連携し，ユーザに有益なサービスを提供する．例えば，自宅のメディアサーバの動画を TV に出力するなどが考えられる．一般に，スマート環境は，セキュリティの考慮や潤沢な IP アドレスの使用などを目的としてプライベートネットワーク上に構築され，NAT やファイアウォールで外部からの通信を制限するものと考えられる．

一方，一定の信頼の置けるスマート環境同士の連携も試みられており，大学間の交換講義において照明・音響サービスを協調動作させた遠隔講義，共同プロジェクトを行なう複数の組織間でユーザが相手に対し公開を許可した機器やサービスを連動させた作業の効率化や資源共有，遠隔会議などの実現が考えられている．これらの連携は，スマート環境を構築するミドルウェアが提供するネットワークサービスや各種デバイスを抽象化したネットワークサービスが，異なるスマート環境間で連携することによって実現される．しかしスマート環境の特性として，モバイルデバイスなどの頻繁なノードの参加・離脱，セキュリティの要求，ソフトウェアの更新が困難な組み込み機器の存在などがある．それら特性からスマート環境間を連携させるには，動的な接続性の制御，任意のノードのみに通信が可能，拡張が想定されていないノードにも利用が可能という条件を満たしたトンネリングが必要となった．

そこで我々は PeerPool[1)] を開発してきた．PeerPool は DNS クエリによる操作インタフェースを持つことで，拡張が想定されていないノードからも利用することができ，ノード単位の接続性を動的に制御することができる．さらに接続性が与えられた他の環境のノードを仮想的に自身の環境のローカルノードとして扱える．しかし従来の PeerPool は以下に挙げる 3 つの問題によりネットワークの透過性が十分ではなかった．

†1 立命館大学大学院 理工学研究科
Graduate School of Science and Engineering, Ritsumeikan University
†2 立命館大学 総合理工学院 情報理工学部
Institute of Science and Engineering, Ritsumeikan University
†3 立命館大学 総合理工学研究機構
The Research Organization of Science and Engineering, Ritsumeikan University
†4 立命館大学 情報理工学部
Department of Computer Science, Ritsumeikan University

Vol.2010-UBI-26 No. 9
2010/5/28

まず PeerPool は IP ヘッダの書き換えによって NAT 間の通信を実現するが，ペイロードは加工しないため，ペイロード中に通信に関わるプライベートドメイン名が含まれる場合，ローカルの IP アドレスが含まれているプロトコルを用いる場合は，通信に支障がでる．さらにマルチキャスト・ブロードキャストに対応していないため，UPnP[2] や Bonjour[3] の利用が困難となる．最後に一般的な通信の制御に，従来の PeerPool が利用する CPU リソースの大きさは問題にはならなかったが，先に述べた問題を解決する機構の導入など，新たに機能拡張する場合それが影響し通信に悪影響を及ぼす場合がある．このプライベートドメイン名の問題は，接続先環境内のプライベートドメイン名に対する名前解決を，自身の環境内に仮想的に生成されたローカルノードの別名として返すことで対応した．[4]

本研究では，残された課題である転送パケットのペイロード変換およびブロードキャスト通信とマルチキャスト通信の中継の 2 つの機能の追加実装を行なうことで，分散スマート環境においてネットワークの構成や各々のノードのソフトウェアへの変更を抑えつつ，ペイロードに加工が必要な通信に対応し，UPnP や Bonjour を含む広範のアプリケーションを利用できるようにする．機能の一部をカーネルモジュール化することで，さらに PeerPool の CPU 利用率を低く抑えることでシステム全体の性能を向上させた．

本稿では 2 章で PeerPool の概要と課題について述べ，3 章で PeerPool の改良案 PPv2 を述べる．さらに 4 章で PPv2 の設計・実装，5 章で評価を述べ，最後にまとめについて述べる．

2. PeerPool の概要と課題

2.1 PeerPool の概要

PeerPool は，スマート環境に PoolGW という DNS と NAT 機能を同時に持つゲートウェイを設置し，その PoolGW 間でオーバーレイネットワークを構築し，DNS クエリを用いた要求に従って各スマート環境のノード間の接続性を制御するシステムである．

PoolGW は DNS クエリ型インタフェースを持ち，特定の規則に従った DNS クエリに従ってスマート環境間の接続性を制御する．多くのネットワーク機器は DNS クエリの発行が容易なため，拡張が容易でないソフトウェアや組み込み機器からも動的に接続性を制御できる．また，PoolGW は DNS クエリにより公開要求のあったノードにのみ接続性を与えるため，ユーザは他のスマート環境へ公開するノードを任意に必要最小限に抑えることが可能である．公開ノード宛の通信は，PoolGW がローカルネットワーク内の空きアドレスを用いて作成 (仮想ローカルノード化) した仮想的なローカルノードに対する通信として行なう (**図 1**)．仮想ローカルノード化されたノードと他のノードとの通信は，PoolGW による透過的なアドレス解決により，あたかも同一ネットワークのものと認識される．そのため通信を行なうノード同士に VPN の設定等の特別な設定は必要ない．PeerPool は，仮想ローカルノードなどのドメイン名を `.pool` という特別な接尾辞を持つ Pool ノード名として管理する．この Pool ノード名は，接続された環境間で単一のものであり，プライベートドメイン名といった他のドメイン名とは独立したものである．

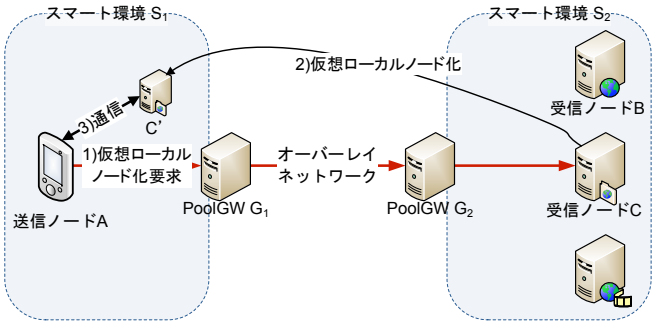

図 1 PeerPool の動作概要 (ノード C の仮想ローカルノード化)

また PeerPool は，仮想ローカルノードと環境内のノードの通信のみをオーバレイネットワーク上で取り持つため，他のノードの通信に関与することはなく，既存のルータの下に設置することが可能である．さらに PoolGW が DNS サーバになることも可能なため，末端ノードの問い合わせを既存の DNS に代理問い合わせすることで，既存の DNS の変更も不要である．

このように PeerPool は，分散スマート環境を構築する上で最適な接続性を確保しており，比較的低い層の接続性であることから，既存のネットワークやアプリケーションの変更を最小限に出来る分散スマート環境接続技術である．

2.2 PeerPool の課題

前章で述べたように，プライベートドメイン名に対する名前解決については解決したが，既存の PeerPool(以下 PPv1) には，パケット加工を要する通信，ブロードキャスト通信およびマルチキャスト通信 (以下 B/Mcast 通信) に未対応であり，また CPU リソースを多く使用するという問題が残されている．以下にその概要を述べる．

PPv1 は特定のプロトコルを利用する際に不具合が生じる．PoolGW がパケットを転送する際に IP ヘッダに書かれたアドレスを書き換えるため，送信者と受信者で IP アドレスに対する認識に齟齬が生じる．このような不具合から，ペイロード中に IP アドレスを含む Web サービスやサービスの詳細が記述されている UPnP や Bonjour のようなサービス発見プロトコル，また H.323 といった接続先の IP アドレスが記述されている通信プロトコルは利用することができない．

さらに PPv1 はブロードキャスト通信およびマルチキャスト通信の転送を適切に行なえなかった．これは仮想ローカルノード化されたノードにのみ接続性を与えるという特性を PeerPool が持つため，単純に B/Mcast パケットを遠隔の環境に転送できなかったためである．この制約より，UPnP や Bonjour のようにブロードキャスト通信を用いるプロトコルにおいて問題が生じる．通信ノードから見て仮想ローカルノードは同一ネットワーク上に存在するため，これらの通信が行えることが望ましい．

最後に PPv1 の機能を使用する上では問題にはならなかったが，PeerPool に新機能を追加し，さらに PoolGW の処理を増大させたとき，スループットの低下やパケットロスを発生させてしまうほど CPU リソースを使用してしまう可能性がある．よって PoolGW の CPU 利用を抑える必要がある．

3. PPv2(PeerPool version 2) の提案

3.1 転送パケットのペイロード変換

図 2 にペイロード変換処理の一例を示す．まずスマート環境 S_2 内のノード C がスマート環境 S_1 内のノード A へ自身が提供するサービスを広告するとする．このときノード C が送信するパケットの中には自身の IP アドレスが含まれている．通常このパケットをノード A が受け取ったとしても通信を行なうことは出来ない．そこで，PoolGW G_1 がパケットを転送する際，パケット内に含まれている IP アドレスを適切な IP アドレスへと書き換えてやることで，以降の通信を正常に行なえるようにする．

このペイロード変換については，iptables[5] のコネクション追跡モジュールにより対応することも可能だが，モジュールを新たに追加するには iptables の再コンパイルが必要となる場合があり，導入コストが高い．

そこでペイロードに IP アドレスを含むプロトコルに関しては，プロトコルに専用のペイロード変換モジュールを実装し，PoolGW においてアプリケーションレベルゲートウェイを構成することによって対応する．ユーザ空間において変換モジュールを動作させることにより，再コンパイルなしに変換機能の on/off を容易に行なうことができ，さらに新規プロトコルに対する要求があった場合においても，PoolGW が NAT 変換前後の IP アドレスを管理していることから追加実装が容易である．

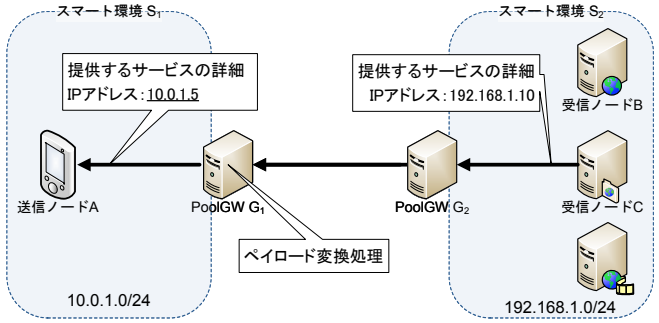

図 2 ペイロード変換の一例

3.2 B/Mcast 通信の中継

2.2 節で述べたように，パケットが到達する範囲を適切に制限する必要がある (**図 3**)．

パケットの到達範囲は，B/Mcast パケットの送信者と物理的および仮想的にリンクローカルであるノードとするのが自然である．任意のノード A,B に対し，A と B が仮想的にリンクローカルなのは，最低どちらか一方が他方のノードを仮想ローカルノード化している場合である．図 3 では，ノード A がノード B,C を仮想ローカルノード化している．この場合，ノード A からの B/Mcast は，スマート環境 S_2 内のノード B,C にのみパケットが到達する．このときノード B あるいは C が B/Mcast を行なった場合，スマート環境 S_1 では，ノード A にのみパケットが到達する．また，ノード D が B/Mcast を行なった場合，S_2 内のノード B,C にはパケットは到達しない．ノード D がノード B,C と通信を行ないたい場合は，ノード D 自身がノード B,C を仮想ローカルノード化する必要がある．

VPLS(Virtual Private Lan Service) では，MAC アドレスに基づいてデータを転送するため，任意のノードにのみ B/Mcast を送信することができる．しかし，VPLS を構築するためには，VPLS に対応したルータを新たに設置する必要があり，導入の敷居が高い．PeerPool は 2.1 節で述べたように，特別な機器の導入が不要であるため，導入の敷居が低く，利便性が高いと言える．

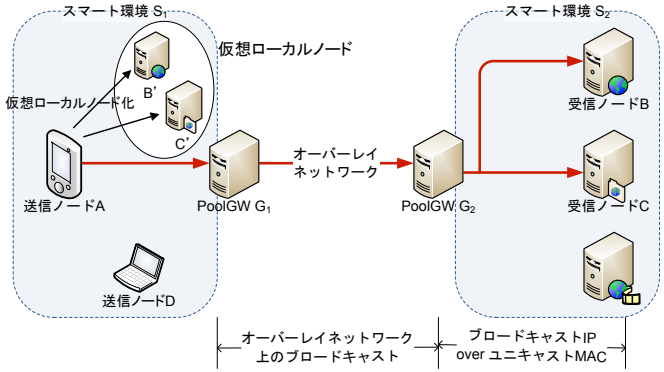

図 3 B/Mcast 通信の中継

表 1 PoolGW の CPU 使用率内訳 (仮想ローカルノード通信時)

Java	OpenVPN	その他	未使用
30%	15%	22%	33%

3.3 パケットキャプチャリングのカーネル空間への移行

表 1 は，PPv1 における仮想ローカルノード通信時の PoolGW の平均 CPU 使用率を表す．最も多く CPU リソースを利用しているのは Java プロセスだとわかる．これは Jpcap[6] により到着したパケットを無差別的に吸い上げていたことが原因である．本来 NAT において，転送するパケットはカーネル空間で処理するが，従来の PeerPool はヘッダの書き換えなど多くの処理をユーザ空間プロセスで行なっていた．これらの処理をカーネル空間において行なうことで，高速化を実現し，さらに引き上げるパケットの減少により負荷の軽減が図れると思われる．さらに 3.1 節で述べたペイロード変換機構へ送るパケットも，高速な選定が期待できる．

これらを考慮し PPv2 では，負荷の大きい Jpcap を排除し，iptables を用いてカーネル空間でパケットの選定を行ない，必要なパケットのみをユーザ空間へ送ることにより負荷の軽減を図る．

4. PPv2 の実装

4.1 ペイロード変換機構

ペイロード変換モジュールは，ユーザの利用実態に応じて任意のプロトコルに対応できるようにするため，プラグインの形態をとる．プラグインはアプリケーション毎に作成する．これによって，プラグインの最大同時稼働数は利用するトランスポートプロトコルの最大ポート数の和 (例：TCP のみの場合 65536 個) になる．読み込まれたプラグインは，当該プラグインが扱うプロトコルに対応するポート番号を iptables のルールに追加する．1 つのプラグインにつき追加されるルール数は多くの場合，オーバーレイネットワーク側とローカルネットワーク側のインタフェースからパケットを吸い上げるための 2 つであるが，アプリケーションのプロトコルによりその数は変動する．また，各プラグインは，与えられたパケットのペイロードをアドレス変換による矛盾がなくなるように書き換えるペイロード変換ルーチンを実装する．

PoolGW のメインプログラムは，4.3 節で述べるパケットキャプチャ機構から吸い上げられたパケットを該当するペイロード変換ルーチンに渡す．また，ペイロード中でプラグインが書き換えるのは原則として IP アドレスが埋め込まれた部分であるから，PoolGW においてアドレス変換が適用される前後のアドレスを調べるための API をプラグインに提供する．

4.2 B/Mcast 中継機構

PeerPool における分散スマート環境間のユニキャスト通信は，PoolGW がルーティングする際にオーバーレイネットワークのアドレス空間を利用して, iptables を用いたアドレス変換を適用することによって実現されている（**図 4**.(a)）．図中 Addr-A と Addr-W はそれぞれ通信ノード A と共有ノード W の実アドレス，Addr-A' と Addr-W' はそれぞれオーバーレイネットワークのアドレス空間内の Addr-A と Addr-W の写像，Addr-A'' と Addr-W'' はそれぞれ A と W の仮想ローカルノードのアドレスである．

B/Mcast 通信も同様に PoolGW によるアドレス変換で実現する（図 4.(b)）．送信者アドレスはユニキャストとまったく同様に変換する．宛先アドレスの変換は，ブロードキャストの場合とマルチキャストの場合で異なる．ブロードキャストの場合は，送信者側の PoolGW（PoolGW G_1）が自スマート環境のブロードキャストアドレス（Addr-S_1）宛のパケットをオーバーレイネットワークのブロードキャストアドレス（Addr-B_x）宛に変換する．さらに，受信者側の PoolGW（PoolGW G_2）がオーバーレイネットワークのブロードキャストアドレス宛のパケットを自スマート環境のブロードキャストアドレス宛に変換する．マルチキャ

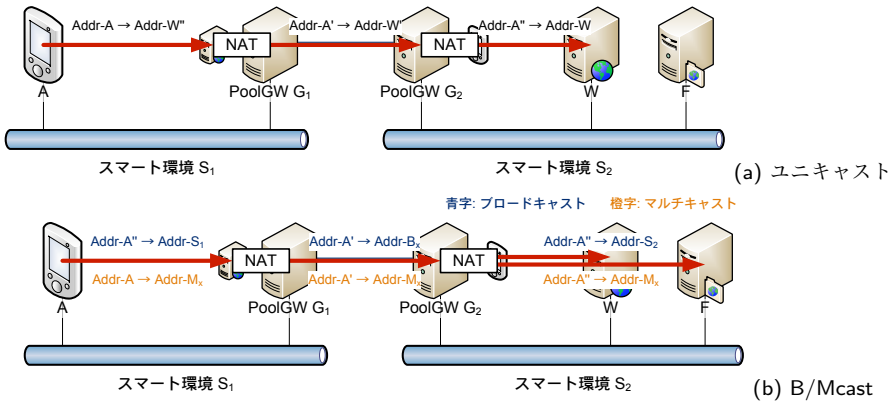

図 4 B/Mcast 通信のルーティングとアドレス変換

ストの場合は，宛先アドレス（Addr-M_x）を変換しない．

受信側の PoolGW から受信者への B/Mcast パケットは，データリンク層でユニキャストを送信する．すなわち，送信者と仮想的にリンクローカルなノードの数だけ B/Mcast IP パケットを複製し，ユニキャストの MAC フレームに納めてそれぞれの宛先に転送する．

4.3 ULOG によるパケットキャプチャ機構

本システムではカーネル空間でパケットの選定を行なうのに，iptables を利用することにした．iptables ではルールを指定することができ，このルールにマッチするパケットのみを引き上げるものとする．またカーネル層に位置する iptables から，ユーザ空間の PeerPool にパケットを引き上げるために，今回 iptables モジュールである ULOG モジュール[7] を利用した．ULOG は、ルールにマッチしたパケットについて、ユーザ空間でのロギングを提供してくれる。パケットがルールにマッチした時、ULOG ターゲットがセットされていると、パケットの情報がパケットそのものと一緒にネットリンクソケット (カーネルモジュールとユーザ空間プロセス間で情報のやりとりを行なうためのインタフェース) を通じてマルチキャストされる．したがって，ユーザ空間プロセスである PeerPool を、いずれかのマルチキャストグループに参加させておけば、それらのパケットを受け取ることができる．

5. 評　　価

5.1 評 価 環 境

評価環境を**図 5** に示す．各ネットワークは 100BASE-TX で構成し，PoolGW G_1-G_2 間

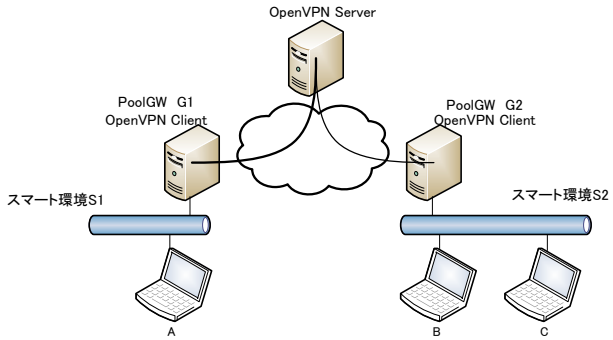

図 5 評価環境

のオーバレイネットワークは OpenVPN の L2VPN(UDP 通信，暗号化有り，圧縮有り) として構成した．環境を構築するマシンは全て同一のもので，OS は Ubuntu 8.04，CPU は Intel Atom 260 1.60GHz，RAM は 1GB のものである．

5.2 ペイロード変換機構の負荷テスト

まずペイロード変換機構に対し，Bonjour のペイロード変換プラグインを利用しブレークダウンテストを行なうことで，本機構が実用に耐え得るかを評価する．図 5 において，ノード A から Bonjour パケットを一定のビットレートで送信し，パケットを G_1,G_2 を経由させノード B に転送する．その際 G_2 においてペイロード変換処理を行い，ペイロード変換処理中の G_2 の CPU 使用率を測定する．ビットレートは段階的に増加させ，それぞれの値を測定した．さらにノード B に到達したパケット数を計算することでパケット損失率を測定する．そして G_2 において Bonjour パケットのペイロード変換を行わなかった場合のパケット損失率と比較を行なう．また同時に netperf により TCP 通信を行い，AB 間のスループットを測定することでペイロード変換が他の通信に及ぼす影響を示す．**図 6** に測定結果を示す．

ペイロード変換を行わなかった場合に対し，変換を行った場合は 128kbps を超えたあたりからパケットロスが発生している．しかし，あるスマート環境中おいて 50 台のノードが Bonjour プロトコルに対応しているとし，それらノードが同時に自身のサービスを広告したとしても瞬間的に発生する Bonjour のビットレートは 50Kbps 程度である．よってこの値は実用上に問題はないと考えられる．またスループットは 64Kbps を超えると急激に低下しているが，同様の理由から許容範囲といえる．パケットロスの原因は，ULOG が持つ

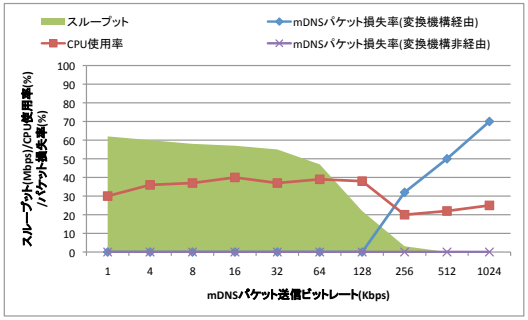

図 6 mDNS のペイロード変換におけるパケット損失率，スループット，CPU 使用率

表 2 1 パケット引き上げるのに要する時間

平均 (s)	標準偏差	遅延時間 (平均＊2)
0.00084	0.00028	0.00168

バッファサイズを上回る速さでパケットが送信され，バッファからあふれてしまうことにあった．またこれはパケットの単位時間あたりの送信数にも比例しており，パケットのヘッダ数が増えればパケットロス率も増加する．

一方，CPU 使用率は，128Kbps を超えてから低下している．これは先ほど述べたようにULOG でバッファオーバーフローが発生し，PoolGW までパケットが吸い上げられず処理が行なわれていないことが原因だと考えられる．

5.3 ペイロード変換機構の処理性能

ペイロード変換機構は，カーネルから一度ユーザ空間に，ルールに該当するパケットを引き上げるため，全てをカーネルレベルで処理する iptables モジュールよりも大きい遅延が発生すると考えられる．iptables のルールにマッチした時間とユーザ空間のプロセスである PoolGW にパケットが引き上げられた時間をそれぞれ測定し，その差を計算することで，1 パケットあたり引き上げるのに発生する遅延を測定した．結果を**表 2** に示す．測定の結果，同一ネットワークのノードにパケットを届ける程度の時間で遅延しているが，リアルタイム性を厳密に要求しない Bonjour のようなサービス発見のためには十分だと考えられる．

また実装した H.323 ペイロード変換プラグインと，iptables で提供されている H.323 モジュールとの処理速度を比較することで，ペイロード変換機構がどれほどカーネル空間モジュールより遅延するかを計測する．測定は通信開始からコネクションが確立されるまでの時間を計測する．測定結果を**表 3** に示す．結果としては，やはりカーネル空間で全ての処理を行なうカーネルモジュールの方がやや速い結果となった．しかし H.323 プラグインが動作をするのは，始めのコネクション確立までであり確立後はペイロードの処理などは行なわないため，以後の通信ではこのような遅延は発生しない．H.323 プロトコルを利用したビデオ・オーディオ会議機能を持つ NetMeeting で動作テストをしたところ，プラグイン形式でも正常に通信できており，ユーザにとってこの程度の遅延ならば問題ないと考えられる．またこの遅延の原因の多くは，先に述べたパケットを引き上げる際のものだと考えられる．

表 3 H.323 プラグインの処理速度

	平均処理時間 (s)
プラグイン形式	1.63114
カーネルモジュール	1.39661
遅延時間	0.23453

またペイロード変換機構では，導入するプラグインの数が増加することに比例し，iptables に記述されるルールも増加していく．それにより通信速度に悪影響を与えることが懸念される．そこで導入するプラグインの数を変化させながらスループットを測定する．

図 5 の AC 間での TCP・UDP 通信のスループットを測定した．その際 PoolGW G_1 において，AC 間の通信とは関係のないプラグインを導入し，ルール数を 0, 10, 30, 50 と増やして実験を行なった．4.1 節でも述べたとおり，多くの場合 1 プラグインあたり 2 ルールのため，0〜25 の範囲でプラグイン数を変更したことになる．また今回は測定するプラグイン数の最大値を 25 としたが，これは実際に導入可能なプラグイン数の上限値ではなく，導入できるプラグイン数の上限は iptables のルール数の上限に比例する．測定されたスループットは，TCP・UDP 共に大きな変化は無かった．ペイロード変換を必要とするプロトコルは FTP, TFTP, Amanda や UPnP, Bonjour などが代表として挙げられる．一般的にそれらを同時に利用するのは，多く見積もって 15 個程度だと考えられるため，本機構は実用上問題ないと考えられる．

5.4 B/Mcast 中継機構のスケーラビリティ

本実験で想定する環境は，上位 24 ビットをサブネットマスクとする最大 254 台の環境である．物理的に受信ノードを増加させる代わりに，ノード C が Proxy ARP によってすべてのパケットの複製を収集し，現実的な受信ノード数で B/Mcast の中継が機能するか否かを確認した．

表 4 iptables のルール増加による性能への影響

	TCP(Mbps)	UDP(Mbps)
ルール数 0	61.18	95.04
ルール数 10	61.08	95.05
ルール数 30	60.30	95.31
ルール数 50	60.40	95.15

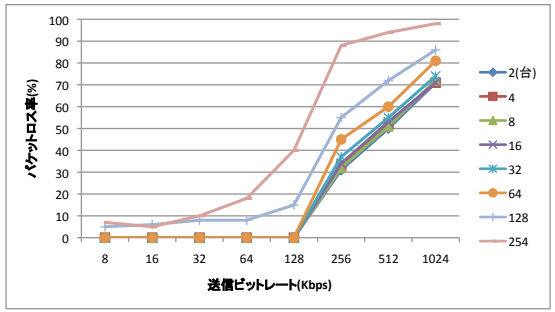

図 7 B/Mcast 中継機構におけるパケット損失率

図 5 に示したノード A からのブロードキャストパケットを中継し，スマート環境 S_2 内の受信ノード数と送信ビットレートを変化させてパケット損失率を測定した．なお，ノード C が Proxy ARP によってすべてのパケットの複製を収集するため，パケット損失率が 0 ならば，C が受信するパケット数は，A が送信したパケット数 $\times$ G_2 のパケット複製数である．したがって，パケット損失率は次式で算出した．

$$1 - \frac{C \text{ が受信したパケット数}}{A \text{ が送信したパケット数} \times G_2 \text{ のパケット複製数}} \quad (1)$$

測定結果を図 7 に示す．

測定結果より，受信ノード数が 64 台以下で送信ビットレートが 128Kbps 以下ならば，ほとんどパケットを損失しない．受信ノード数または送信ビットレートのいずれかがこの水準を超えるとパケット損失率が増大する．スループットは，映像のストリーミング配信には不足だが，UPnP や Bonjour などのサービス発見のためには十分である．

5.5 従来 PeerPool との性能比較

PPv1 と性能を比較するため，図 5 におけるノード A からノード B へ netperf による TCP 通信を行ない，G_1,G_2 の CPU 使用率およびスループットを測定する．測定結果を**表 5** に示す．これによれば全体の CPU 使用率に変化は見られないが，OpenVPN に割り当てられる CPU リソースが PPv1 に比べ PPv2 の方が飛躍的に上昇していることがわかる．これは PPv1 においてのボトルネックであった Java プログラムによるパケットの書き換えが排除されたため，その分 OpenVPN に割り当てられるリソースが増えたと考えられる．それに伴いスループットもおよそ 2 倍増加している．以上のことから PPv2 は PPv1 に比べ性能が向上したと言える．

表 5 PPv1 と PPv2 の CPU 使用率とスループット比較

	G_1/内 OpenVPN	G_2/内 OpenVPN	スループット
PPv1	67%/15%	66%/15%	32Mbps
PPv2	65%/40%	67%/41%	63Mbps

6. ま と め

本稿では，スマート環境を構築する上で最適な接続性を確保しており，IP 層の接続性から多様な用途で利用ができる通信技術 PeerPool の問題を解決するために，ペイロード変換機構・B/Mcast 中継機構の 2 つの機構および性能向上について，設計・実装を述べ，このシステムを PPv2(PeerPool version 2) とした．評価の結果，ペイロード変換機構が想定される利用に耐えられること，B/Mcast 中継機構のスケーラビリティが十分に確保されていることを確認した．さらに提案した PPv2 が既存の PeerPool に比べ大きく性能が向上したことを確認した．

参 考 文 献

1) 榎堀　優，中野悦史，新井イスマイル，西尾信彦：PeerPool: DNS クエリを操作に用いた分散スマート環境間接続技術，情報処理学会論文誌コンピューティングシステム (ACS) 第 28 号, 2010 年 2 月発行 (2009).
2) : UPnP Forum, Online (2009). http://upnp.org/.
3) : Bonjour, Online (2009). http://developer.apple.com/networking/bonjour/.
4) 榎堀　優，新井イスマイル，西尾信彦：サービス利用に必要なノードを自動的に追加する動的 VPN 管理機構，情報処理学会研究報告 (2008-UBI-20),pp.9–14 (2008 年 11 月).
5) : IPTABLES (2009). http://www.linux.or.jp/JM/html/iptables/man8/iptables.8.html.
6) Fujii, K.: Jpcap- a Java library for capturing and sending network packets, Online (2007). http://netresearch.ics.uci.edu/kfujii/jpcap/doc/.

Vol.2010-UBI-26 No. 9
2010/5/28

7) : The Netfilter.org "ulogd" project, Online (2009). http://ftp.roedu.net/web-mirrors/netfilter/projects/ulogd/index.html.