社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－DPS－130 (41)
2007－CSEC－ 36 (41)
2007／3／2

P2Pネットワークを用いた仮想IPネットワーク構築

高野 祐輝[†] 井上 朋哉[†] 篠田 陽一[‡§]

北陸先端科学技術大学院大学 情報科学研究科[†] / 情報科学センター[‡]
独立行政法人 情報通信研究機構[§]

概 要

インターネットは，中央に依存しない，完全に分散化されたネットワークの構築を目標として構築されており，この基本理念は今でも変わっていない．分散化されたネットワークは，一部に障害が起きても大勢には影響を与えないという点で，非常に耐障害性の強いものとなる．しかしながら，従来手法でIPネットワークを仮想化した場合，中央集権的になり，インターネットのように完全に分散化されたネットワークにはならない．一方，P2Pネットワークと呼ばれるネットワークは，完全に分散化されたネットワークとなっており，インターネットと類似している点も多くある．我々は，このP2Pネットワークを用いて，分散化された耐障害性に優れた仮想IPネットワークを構築する手法を提案する．

Virtual IP Network Constructed by P2P Network

Yuuki Takano[†] *Tomoya Inoue*[†] *Yoichi Shinoda*[‡§]

SCHOOL OF INFORMATION SCIENCE[†] / CENTER FOR INFORMATION SCIENCE[‡],
JAPAN ADVANCED INSTITUTE OF SCIENCE AND TECHNOLOGY
NATIONAL INSTITUTE OF INFORMATION AND COMMUNICATIONS TECHNOLOGY[§]

Abstract

To build completely decentralized network is the principle of the Internet. This fundamental principle has not changed since the beginning of the Internet. A decentralized network is high fault-resilient network, because it keeps on working even if a part of its network fails. However, a virtual IP network constructed by existing methods depends on a center server and is not decentralized unlike the Internet. In contrast, P2P networks which are a virtual network are completely decentralized like the Internet. In this paper, we will propose the method of constructing completely decentralized virtual IP network by using P2P network.

1 はじめに

P2Pネットワークを利用すると，インターネット上に仮想的なネットワークを構築することができる．一方，インターネットは，中央に依存しない完全に分散化されたネットワークの構築を目標として作られたものであり，その基本理念は今でも変わっていない．しかしながら，従来の仮想IPネットワークの構築手法は，インターネットのような分散化された耐障害性に強いネットワークを実現しているとは言いがたい．

本論分では，この構造化P2Pネットワークを基盤として，従来手法よりも耐規模性，耐障害性に優れた，完全に分散化された仮想的なIPネットワークを構築する手法を提案し，実証アプリケーションであるP2P@iについて説明する．

2 準備と概説

本節では本論文で用いる表記の説明を行った後，我々の提案する手法の概要を述べる．

2.1 表記

本論文では以下の表記を用いる．

$a||b$： a と b の連結
$a \oplus b$： a と b の排他的論理和
$a[i]$： a の下位 i バイト目
$\{0,1\}^+$： 0か1の1回以上の繰り返し

2.2 概説

我々の提案する機構は，構造化P2Pネットワークを利用して，仮想的なIPネットワークを構築する．

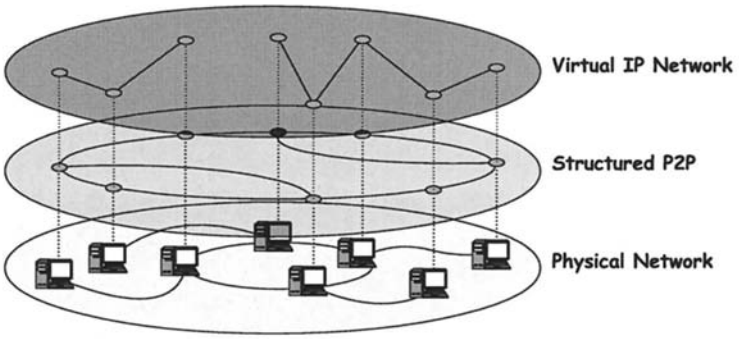

図 1: 概要図

その概観は図 1 となる．図 1 では，最下位のレイヤに実際のインターネットが描かれている．中間のレイヤが，インターネット上に構築された構造化 P2P ネットワークとなっている．我々は，構造化 P2P ネットワークを用いて，さらに仮想的な IP ネットワークを構築する手法を提案する．

P2P ネットワークは分散化されたネットワークである．そのため，我々の提案手法を用いると，従来までのネットワーク仮想化技術では得られなかった耐障害性が得られると期待できる．

3 P2P ネットワークについて

本節では，P2P ネットワークとその分類について説明を行った後，構造化 P2P ネットワークについての説明と定義を行う．また，構造化 P2P ネットワークを用いて実現される機能である，分散ハッシュテーブルと，オーバレイマルチキャストについても説明する．

3.1 P2P ネットワークの分類

P2P ネットワークは大きく分けて，非構造化 P2P ネットワークと構造化 P2P ネットワークという分類がなされる．非構造化 P2P ネットワークを用いたソフトウェアとして代表的なものは，Gnutella [1] 等が挙げられる．これらの非構造化 P2P ネットワークはアドレス構造を持たず，メッセージの検索や転送をフラッディングベースで行っている．一方の構造化 P2P ネットワークはアドレス構造を持つネットワークとなっており，アドレスをベースとして効率的に転送を行うことが出来る．構造化 P2P ネットワークを実現するアルゴリズムには，Symphony [3] や，Chord [5] 等が提案されている．

3.2 構造化 P2P ネットワークとその定義

構造化 P2P ネットワークを実現するアルゴリズムは多数提案されているが，実現出来ることはほぼ同じである．しかしながら，構造化 P2P ネットワークを定義する統一的な概念は示されていない．そこで我々は，構造化 P2P ネットワークを用いた仮想 IP ネットワーク構築法の説明を行う前に，構造化 P2P ネットワークについて定義を行う．

まず構造化 P2P ネットワークの定義を行う前に，アドレスについて定義を行う．

定義 1. 構造化 *P2P* ネットワークはアドレス構造を持ち，そのアドレスの集合を A と表す．

次に，ノードについて定義を行う．

定義 2. 構造化 *P2P* ネットワークに参加している n 個のノードを $\nu_0, \nu_1, \cdots, \nu_{n-1}$ とする．ただし $\{^{\forall} i \in \{0, \cdots, n-1\}; ^{\exists} \nu_i \in A\}$ かつ，$\{i, j \in \{0, 1, \cdots, n-1\}, i \neq j; \nu_i \neq \nu_j\}$ となる．

ここで，n 個のノード全体の集合を N_n とすると，

$$N_n = \{\nu_0, \nu_1, \cdots, \nu_{n-1}\} \tag{1}$$

となる．すなわち，これは $N_n \subset A$ である．

また，あるノードは，あるアドレスからあるノードへの写像を持っている．これは，次のように定義出来る．

定義 3. あるノード $\nu_i \in N_n, i \in \{0, \cdots, n-1\}$ は以下のような写像を持つ．

$$f_{\nu_i}(a) \rightarrow \nu,\ a \in A, \nu \in N_n \tag{2}$$

さらに，構造化 P2P ネットワークは，アドレス対ノードへの写像を持つ．これは，次のように定義出来る．

定義 4. 構造化 *P2P* ネットワークでは系全体として，

$$g(a) \to \nu,\ a \in A, \nu \in N_n \tag{3}$$

となる写像を持っている．ただし，$g(a)$ は定義 *3* で示された写像の集合うち任意の *1* つを起点として，再帰的に計算するような写像である．

定義 4 は，ある任意のノードの持つ写像 $f_\nu(a)$ を起点として，再帰的に求めた場合，あるノードに収束する事を示している．つまりこれは，任意の $\nu_k \in N_n$ で

$$\begin{aligned} f_{\nu_k}(a) &= \nu_{k+1} \\ f_{\nu_{k+1}}(a) &= \nu_{k+2} \\ &\cdots \\ f_{\nu_{k+m-1}}(a) &= \nu_{k+m} \\ f_{\nu_{k+m}}(a) &= \nu_{k+m} \end{aligned} \tag{4}$$

となる事を意味する．ただしここで，$\nu_{k+1}, \cdots, \nu_{k+m} \in N_n$ である．

最後に，構造化 P2P ネットワークの定義を行う．

定義 5. 構造化 *P2P* ネットワークとは，定義 *1, 2, 3, 4* を満たす構造を持つ論理ネットワークである．

3.3 分散ハッシュテーブル

構造化 P2P ネットワークを利用すると，分散ハッシュテーブルと呼ばれるデータ構造を実現することが出来る．ハッシュテーブルでは，Key-Value ペアを Bucket と呼ばれるエントリポイントに格納する．一方，分散ハッシュテーブルでは，構造化 P2P ネットワークに参加しているノードをハッシュテーブルで言う Bucket と見なし，各ノードに分散して Key-Value ペアを格納する．

分散ハッシュテーブルは，ハッシュテーブルと同じように，あるデータ列からハッシュ値を求めるためのハッシュ関数が用いられる．このハッシュ関数 $\mathcal{H}(x)$ は，次のように表す事が出来る．

$$\mathcal{H}(x) \to a,\ x = \{0,1\}^+, a \in A \tag{5}$$

ハッシュテーブルでは基本操作に put, get, remove があり，それぞれ，Key-Value ペアの保存，取得，削除の操作を意味する．構造化 P2P ネットワークを用いた分散ハッシュテーブルの put は，次のアルゴリズムで実現可能である．

アルゴリズム 1.

1. $v = \{0,1\}^+$ を *Value*，$w = \{0,1\}^+$ を *Key* の *seed* とする
2. $k = \mathcal{H}(w)$ を計算して *Key* を得る
3. 手順 *1, 2* より *Key-Value* ペア，(k, v) を得る
4. 定義 *4* より，$g(k) = \nu_m, m \in \{0, \cdots, n-1\}$ となる，ノード ν_m を得る
5. 得られたノード ν_m に，*Key-Value* ペア *(k, v)* を保存

また，アルゴリズム 1 と同様な操作を行うことで，分散ハッシュテーブルの get，remove も実現できる．

3.4 オーバレイマルチキャスト

構造化 P2P ネットワークを用いると，マルチキャスト通信も行うことが出来る．P2P ネットワークなどのオーバレイネットワークを用いたマルチキャストは，オーバレイマルチキャストと呼ばれ，一般的な IP マルチキャスト等とは動作するレイヤが違う．

Scribe [2] 等では，構造化 P2P ネットワークを利用したマルチキャスト通信の手法を提案している．これらは基本的に，ランデブーポイントを中心とした共有木（マルチキャストツリー）を構成することで，マルチキャスト配信を実現している．

ここで，簡単にマルチキャストツリーの構成法を説明する．構成するマルチキャストのグループ名を $p \in \{0,1\}^+$ とする．すると，定義 4 と式 (5) から，ランデブーポイントとなるノード ν_r を得るための式は，次のようになる．

$$\nu_r = g(\mathcal{H}(p)) \tag{6}$$

今，グループ p に参加している s 個のノードの集合を $Q_s = \{\upsilon_0, \cdots, \upsilon_{s-1}\} \in N_n$ とする．すると，あるノード $\upsilon_i \in Q_s, i \in \{0, \cdots, s-1\}$ からランデブーポイント ν_r までの経路は，式 (4) と同様に求めることが出来る．このノード υ_i から，ノード ν_r までの経路を R_i とすると，グループ p のマルチキャストツリー T_p は以下のようになる．

$$T_p = \bigcup_{i=0}^{s-1} R_i \quad (7)$$

3.5 実装と改良点

我々は，これら構造化P2Pネットワーク，分散ハッシュテーブル，オーバレイマルチキャストを実現するライブラリをC++を用いて開発を行った．構造化P2Pネットワークのアルゴリズムには，Small Worldネットワーク理論[4]を適用したSymphony[3]を採用している．Symphonyを採用した理由は，他のアルゴリズムと比較して比較的シンプルであり，また，安全性の面でも有利[6]であるためである．

構造化P2Pネットワークでは，トポロジ維持のために，大量のメッセージを送信する必要がある．このため，ある時間において，制御パケットがバースト的に送信されることがある．そこで，我々は，制御パケットの送信間隔を，素数倍に設定することで，バースト的なトラフィックを防いでいる．具体的には，我々の実装では，トポロジ維持のため6種類の制御パケットを数十秒間隔で送信しているが，この間隔を，11, 13, 17, 23, 29, 89秒等としている．

また，P2Pネットワークはその特性上，ノードの離脱や参加が頻繁に起きることになる．そうすると，分散ハッシュテーブルでは，Bucketの位置が変わる等の問題が起きる．これを回避するため，put時には複数のBucketにデータを保存したり，定期的にputを行う等している．しかしながら，これらを過剰に行うと，全体のトラフィック量が莫大にふくれあがってしまう．そこで我々は，定期的にputする間隔を指数関数的に増大させることで，トラフィック量の増大を抑えている．

4 仮想IPネットワークの構築

本節では，P2Pネットワークを用いた仮想IPネットワーク構築の実証ソフトウェアP2P@iについての説明を行った後，仮想IPネットワークの構築方法と，ユニキャスト，ブロードキャスト，マルチキャストで送信を行うための手法について説明する．

4.1 ソフトウェアアーキテクチャ

我々は，構造化P2Pネットワークを実現するライブラリlibsymphonyと，これを用いて仮想IPネットワークを構築するコンポーネントP2P@i Coreの開発を行っている．P2P@iは，libsymphonyとP2P@i Coreを含むアプリーケション名である．このP2P@iのアーキテクチャは図2となる．

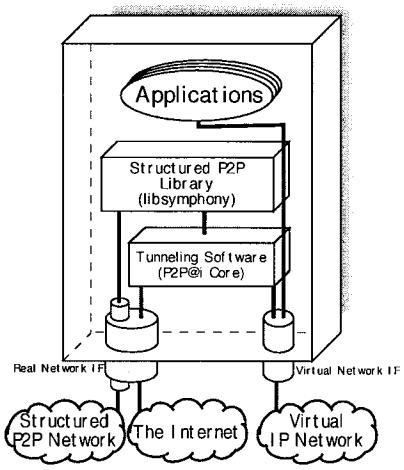

図2: アーキテクチャ

図2では，libysymphonyが構造化P2Pネットワークと接続し，P2P@i Coreが仮想IPネットワークと接続するための，仮想インターフェースを提供しているのが分かる．また，仮想ネットワークを構築するために，libsymphonyとP2P@i Coreがお互いに連携していることが分かる．

4.2 仮想IPネットワーク構築の概要

我々は，仮想IPネットワークを構築するため，チャネルと言う概念を導入する．このチャネルは，以下のような属性を持つ，仮想IPネットワークの識別子である．

- チャネル名
- パスワード
- ネットワークアドレス
- ネットワークマスク

すなわち，あるノードは仮想IPネットワークを構築する際，あるチャネル名，パスワード，ネットワークアドレス，ネットワークマスクの属性を持つチャネルを選択することになる．P2P@iでは，こうして選択されたチャネルのIPアドレスに基づいて，アドレス付け，ユニキャスト，マルチキャスト，ブロードキャストの送信を行う．

4.3 IPアドレスの設定

インターネットでは，各ノードにユニークなIPアドレスが割り振られている．仮想IPネットワークを構築する際も，ノードにユニークなIPアドレスを割り振る必要がある．そこで我々は，分散ハッシュテー

ブルを用いてIPアドレスを自動設定する手法を提案する．

提案するIPアドレス自動設定のアルゴリズムは次のようになる．

アルゴリズム **2.**

1. 参加する仮想*IP*ネットワークのチャンネル名をch，パスワードを$pass$，ネットワークアドレスを$nwaddr$．ネットマスクを$mask$とする
2. $nwaddr$に対応するホストアドレス$haddr$をランダムに生成
3. $key = \mathcal{H}(ch||pass||nwaddr||mask||haddr)$を計算する
4. 計算した*Key*をもとに，分散ハッシュテーブルから*get*
5. *get*出来たら手順*2*にもどる，出来なかったら手順*6*へすすむ
6. 自ノードのグローバル*IP*アドレスを$gaddr$，通信のために用いるポート番号を$port$とする
7. $value = gaddr||port$を計算する
8. *Key-Value*ペア$(key, value)$を分散ハッシュテーブル上に*put*
9. 分散ハッシュテーブルから，keyをキーにして取得．このとき得られた値を$value2$とする
10. $value = value2$でないなら手順*2*へもどる，$value = value2$ならば手順*11*へ進む
11. 仮想インターフェースに，アドレスを設定

同一仮想IPネットワーク内では，重複したIPアドレスを利用することが出来ない．そのため，IPアドレスを設定する際に重複チェックを行う必要がある．アルゴリズム2では，はじめに，ランダムにIPアドレスを生成している．その後，生成したIPアドレスをもとに生成したkeyを用いて，分散ハッシュテーブルからgetを行い，重複チェックを行う．

以上は自動IP設定の説明であったが，手動で設定を行う際にも，ほぼ同様に行うことが出来る．

4.4 ユニキャスト通信

本ソフトウェアでは，tapデバイスを用いて仮想インターフェースを提供している．そのため，ユニキャストで通信を開始する際，OSから仮想インターフェースへ向けてARPパケットが送信される．ARP要求に対して，ARP応答を行わないと実際のIPパケットが送信されないので，我々のソフトウェアでは，代理ARPを用いてアドレス解決を行う．

代理ARPを用いてアドレス解決を行った後，IPアドレスの宛先をもとに，分散ハッシュテーブルを用いて，実際のアドレス解決をおこなう．アドレス解決のアルゴリズムは以下のようになる．

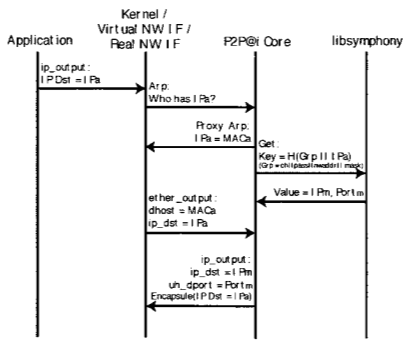

図3: ユニキャストでの転送

アルゴリズム **3.**

1. 仮想*IP*ネットワークのチャンネル名をch，パスワードを$pass$，ネットワークアドレスを$nwaddr$．ネットマスクを$mask$とする
2. パケットの宛先アドレスを$daddr$とする
3. $key = \mathcal{H}(ch||pass||nwaddr||mask||daddr)$を計算
4. 計算した*Key*をもとに，分散ハッシュテーブルから*get*
5. 手順*4*で*get*が出来たら手順*6*へ，出来なかったら*ICMP Host Unreach*を送信して終了
6. 手順*4*で得られた値より，宛先のグローバル*IP*アドレス$gaddr$と，ポート番号$port$を得る

アルゴリズム3を用いて宛先のグローバルIPアドレスとポート番号を得られたら，パケットをカプセル化して送信を行う．なお，ここで得られるグローバルIPアドレスとポート番号は，仮想IPネットワーク参加時に，宛先ノードがputしたものである（アルゴリズム2）．

図3は，実際のユニキャスト送信時に行われるパケットのやりとりを示している．図3では，まず始めにARPを受信したP2P@i Coreが代理ARPを用いてアドレスの解決を行っている．その後，P2P@i Coreはlibsymphonyを通じて実際の送信先アドレスとポート番号を得ている．最後に，P2P@i Coreはカプセル化したパケットを実際の宛先に送信する．

4.5 ブロードキャストとマルチキャスト

P2P@iは仮想的なIPネットワークを構築し，その仮想ネットワークは単一のブロードキャストドメインとなる．IPの場合，ブロードキャストとマルチ

キャストは，同一ブロードキャストドメインに存在する全てのノードに送信されることになる．従って，P2P@i では，これらブロードキャストとマルチキャストパケットの送受信を行うために，オーバレイマルチキャストを利用する．

3.4 節で説明したように，マルチキャストツリーを構築するためには，グループ名を用いる必要がある．P2P@i では，仮想 IP ネットワークを構築するためのグループ名 Grp は次のように計算する．

$$Grp = ch||pass||nwaddr||mask \quad (8)$$

5 暗号化通信

P2P ネットワークでは全てのノードがルータともなるため，メッセージを転送する際に，容易にデータの改ざん盗聴を行うことができる．そのため，P2P@i では事前鍵共有方式による暗号化通信を行う．

暗号化アルゴリズムの説明の前に，アルゴリズムで用いる表記を以下に示す．

r_1 :	256bit 乱数
r_2 :	128bit 乱数
K :	256bit 事前共有鍵
seq :	シーケンシャル番号
$AES128_k(a)$:	128bit AES を使って a を鍵 k により暗号化
$AES256_k(a)$:	256bit AES を使って a を鍵 k により暗号化
$HSHA1_k(a)$:	HMAC SHA1 を使って a のメッセージダイジェストを鍵 k により計算
pkt :	送信するパケット

暗号化のアルゴリズムは次のようになる．

アルゴリズム **4.**

まずはじめに，以下を計算する．

1. r_1 を生成
2. $k_1 = r_1 \oplus K$ を計算
3. $seq = 0$ とする

実際のパケットの暗号化は以下のように行う．

1. $seq > 2^{127}$ ならば，もう一度 r_1 を選びなおして k_1 を計算し，$seq = 0$ とする．
2. r_2 を生成
3. $h_1 = AES256_{k_1}(r2||seq)$ を計算
4. $h_2 = AES128_{r_2}(pkt)$ を計算
5. $h = r_1||h_1||h_2$ を計算
6. $k_2[i] = K[i] \oplus K[i+20]$ として k_2 を計算
7. $md = HSHA1_{k_2}(h)$ を計算
8. seq をインクリメント
9. $ct = md||r_1||h_1||h_2$ を得る

アルゴリズム 4 では，パケットを送信時にシーケンシャル番号を同時に送信している．受信する側では，このシーケンシャル番号をチェックすることで，リプライ攻撃を防ぐことが可能である．また，一定数以上パケットを送信したら，暗号化の鍵として利用する乱数 r_1 を選びなおしている．このため，大量にパケットを送信する場合でも，鍵の安全性が低下することはない．さらに，HMAC SHA1 を用いてメッセージダイジェストを計算しているため，完全性のチェックも高速に行うことが出来る．

6 結論と今後の展望

本論分では，構造化 P2P ネットワークを用いて仮想 IP ネットワークを構築する手法を提案し，その実証ソフトウェアである P2P@i について説明を行った．P2P@i は，大規模かつ，耐障害性につよい仮想 IP ネットワークを用意に構築することを目的として作成されている．今後，大規模実験を行い，耐規模性，耐障害性について検証していく予定である．

また，IPv6 や名前解決機構についても対応していく予定である．IPv6 の場合，アドレス体系が複雑なため名前解決機構が重要になってくる．従来の名前解決機構は，中央のサーバに頼るため，P2P@i のように仮想 IP ネットワークを自由に構築するようなネットワークには不向きである．そこで，P2P@i に適した名前解決機構についても提案，実装していく．

参考文献

[1] The gnutella protocol specification v0.4. http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf.

[2] Jon Crowcroft and Markus Hofmann, editors. *Networked Group Communication, Third International COST264 Workshop, NGC 2001, London, UK, November 7-9, 2001, Proceedings*, volume 2233 of *Lecture Notes in Computer Science*. Springer, 2001.

[3] Gurmeet Singh Manku, Mayank Bawa, and Prabhakar Raghavan. Symphony: Distributed hashing in a small world. In *USENIX Symposium on Internet Technologies and Systems*, 2003.

[4] S. Milgram. The small world problem. In *Psychology Today*, page 67(1), 1967.

[5] Ion Stoica, Robert Morris, David R. Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *SIGCOMM*, pages 149–160, 2001.

[6] Yuuki Takano, Naoki Isozaki, and Yoichi Shinoda. Multipath key exchange on p2p networks. In *ARES*, pages 748–755, 2006.