インターネット非接続時における複数NAT越えP2P通信方式の検討

田中　有彩† 　前野　誉†† 　大和田泰伯††† 　高井　峰生†,†††† 　小口　正人†

† お茶の水女子大学　〒112-8610 東京都文京区大塚 2-1-1
†† 株式会社スペースタイムエンジニアリング　〒101-0025 東京都千代田区神田佐久間町 3-27-3
††† 情報通信研究機構　〒980-0812 宮城県仙台市青葉区片平 2-1-3
†††† UCLA Computer Science Department　3803 Boelter Hall, Los Angeles, CA 90095-1596, USA
E-mail: †{g1320523,oguchi}@is.ocha.ac.jp, ††tmaeno@spacetime-eng.com, †††yowada@nict.go.jp,
††††mineo@ieee.org

あらまし　多くのネットワークアプリケーションはクライアント・サーバ型通信を使用しており，災害時には被災域外への通信の遮断や輻輳を起こす恐れがある．そのため災害時においてインターネットにつながらない場合，もしくは衛星回線等によりインターネット接続回線が非常に細い場合においても情報共有できるアプリケーション通信環境が必要である．そこで本研究では，インターネット非接続時でもプライベートネットワーク間でNAT越えを実現することにより，P2P型通信を用いた情報共有を行うことを目指す．方法としては，シグナリングサーバとSTUNサーバ，必要に応じてTURNサーバを用意し，WebRTCを用いて，NAT配下にある端末同士のP2P型通信を可能にすることを検討する．

キーワード　NAT越え，P2P，災害，シグナリング，STUN，TURN

Studies of P2P communication system using multiple NATs traversal not on the Internet

Arisa TANAKA†, Taka MAENO††, Yasunori OWADA†††, Mineo TAKAI†,††††, and Masato OGUCHI†

† Ochanomizu University　2-1-1 Otsuka, Bunkyouku, Tokyo 112-8610 JAPAN
†† Space-Time Engineering　KandaSakumacho 3－ 27－ 3, Chiyoda-ku, Tokyo, 101-0025, JAPAN
††† National Institute of Information and Communications Technology　2-1-3, Katahira, Aoba, Sendai-city, Miyagi, 980-0812, JAPAN
†††† UCLA Computer Science Department　3803 Boelter Hall, Los Angeles, CA 90095-1596, USA
E-mail: †{g1320523,oguchi}@is.ocha.ac.jp, ††tmaeno@spacetime-eng.com, †††yowada@nict.go.jp,
††††mineo@ieee.org

1. はじめに

現在, インターネットは人々にとって欠かせない生活の一部となっており, 人類共通の重要な情報インフラとして広く普及されている. 中でも多くの人が通話や電子メール,SNSなど, 連絡を取る手段としてインターネットを利用している. そのため突然インターネットが使えなくなり, 連絡が取れなくなってしまうことは人々への不安や困惑を招く可能性がある. 特に地震大国である日本にとって, その可能性は大きいと言える. また多くのネットワークアプリケーションはクライアント・サーバ型通信を使用しているため，災害時には被災域外への通信の遮断や輻輳を起こす恐れがある．そのため災害時においてインターネットにつながらない場合，もしくは衛星回線等によりインターネット接続回線が非常に細い場合においては情報共有ができなくなってしまう．そこでインターネットにつながっていない状況下でも情報共有できるアプリケーション通信環境が必要だと考えた.

通信方法としてはクライアント・サーバ型通信ではなくP2P型通信を利用する. これは, クライアント・サーバ型通信では, 災害時にサーバが使えなくなってしまうともはや通信ができな

くなってしまうためである．これに対し P2P 型通信であれば，災害時に 1 つアクセスポイントがあれば端末同士での通信が可能となるためである．

またモバイル・アドホックネットワーク (MANET) とは基地局や固定網に依存せず, 移動端末を構成要素とする自律分散形のネットワークである [1]. 既存の施設や設備を必要とせずにネットワークを構成できるという利点にも関わらず, なかなか実用化はされていない．これは,MANET を利用した多くのものが, 端末側での事前準備を必要とする方式であったことが一因であると言える．そこで通常の MANET は端末同士間での通信としていることが多いが, 今回は NAT(Network Address Translator) ルータ同士での通信とする．つまり, この仕組みを端末自体に作るのではなく，NAT ルータ自体に作ることで，NAT ルータ同士の MANET を構築し，端末側での準備の必要なく NAT ルータを利用して P2P 型通信によりデータを共有することを最終目標とする．

本研究では初期段階として，インターネット非接続時でもプライベートネットワーク間で NAT 越えを実現することにより，P2P 型通信を用いた情報共有を行うことを目指す．方法としては，シグナリングサーバと STUN サーバ，必要に応じて TURN サーバを用意し，WebRTC 技術を用いて，NAT 配下にある端末同士の P2P 型通信を可能にすることを検討する．まずはプライベートネットワーク間での NAT 越え P2P 型通信による情報共有の実験を行った．

2. 関 連 研 究

現在，NAT の外部から内部のネットワークへ通信を開始できないという NAT 越え問題に対する研究が多くなされている．

[2] では，外部ノードとホームゲートウェイが連携することにより NAT 越え問題を解決する従来の NAT-f(NAT-free protocol) の誰でもホームネットワーク内の内部ノードにアクセスできてしまうという課題に対し，サービス単位でグルーピングすることにより，外部ノードからのアクセス制御と内部ノードが提供するサービスの制御を同時に実現できることが示されている．

[3] では，DNS サーバと NAT ルータを利用して両者を協調させることにより NAT 越えを実現する方式が提案されている．その提案方式は NTS(NAT-Traversal Support) システムと呼ばれており,DNS サーバを改造した NTS サーバ,NAT ルータを改造した NTS ルータ, 実行するプロトコルとして NTS プロトコルが使用されている．また端末の機能追加が必要な NAT-f とは異なり，一般のユーザ端末に機能を追加することなく, かつエンドエンドで NAT 越えを実現できる方式である．

[4] では，STUN では本来対応することができないシンメトリック型 NAT を STUN を拡張することで超えて，NAT の外側から TCP 通信を開始する手法の実装が行われている．本来ならば STUN は UDP 通信におけるものが多く，それと比較して TCP 通信における研究は少ないため，珍しい研究である．しかし本研究では後述する WebRTC 技術を利用するため, プロトコルは UDP を対象としている．

どの関連研究もサーバを必要とする前提での NAT 越えとなっている. そのため, 本研究の特徴とも言える NAT ルータ自体にサーバを搭載するという点は非常に有益だと考えられる．

3. P2P (Peer to Peer) 方式とクライアント・サーバ方式

P2P 方式とは，ネットワーク上で対等な関係にある端末間を相互に直接接続し，データを送受信する方式 (図 1) である．また，そのような方式を用いて通信するソフトウェアやシステムの総称でもある．データの送り手と受け手が分かれているクライアント・サーバ方式 (図 2) などと対比される用語で，利用者間を直接繋いで音声やファイルを交換するシステムが実用化されている．これにより，特定のサーバを介さずに，端末同士の通信を可能にする．

クライアント・サーバ方式とは, サービスを提供する側（サーバ）とサービスを受ける側（クライアント）をネットワークで結び, クライアントからの要求にサーバが応答する形で処理を進める方式である. 通常のネットワーク通信で利用されているのはクライアント・サーバ方式である．また MANET はモバイル環境において P2P 接続を利用し, 一時的なネットワークを構成するものである. 通常の MANET は端末間の P2P 接続を用いるが, 本研究では端末が所属するプライベートネットワーク間の接続を利用する点が特徴である．

図 1　P2P 方 式

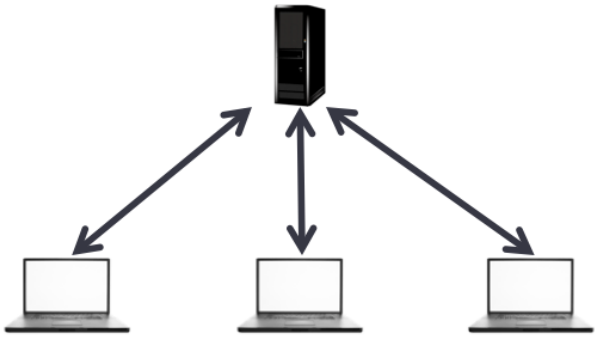

図 2　クライアント・サーバ方式

4. NAT 越え技術と WebRTC

4.1 NAT 越え

多くの端末はプライベートネットワークに所属して NAT 配下にあり，端末から外のネットワークへ通信をすることが通常である．しかし逆に，外から NAT 配下の端末へ直接通信することはできない．

この問題を解決する手法を NAT 越え（NAT Traversal）という．

4.2 STUN と TURN

STUN (Session Traversal Utilities for NATs) とは，NAT 越えの方法の一つである． 通信するホストが STUN サーバに UDP 接続を行い，NAT が割り当てたグローバル IP アドレスとポート番号を取得する．NAT の種類にはフルコーン型・制限コーン型・ポート制限コーン型・シンメトリック型と全部で 4 種類ある (表 1)．STUN が対応できる NAT は，フルコーン型・制限コーン型・ポート制限コーン型の 3 つに限られる．

表 1 から分かるように, 表が下に行くほど利用制限が厳しくなっている. 特に一番厳しいシンメトリック型は STUN で対応できない. そのため，シンメトリック型には TURN(Traversal Using Relay NAT) を使用することで，全ての NAT に対応する． しかしすべての通信を TURN サーバ経由で行うため, P2P 通信ではなくなり, またサーバにも高負荷が掛かってしまう．

表 1　NAT の種類

種類名	NAT が割り当てたポートにアクセスできるインターネット側の端末の制限	利用制限の厳しさ
フルコーン型	どの端末でもアクセス可	緩やか
制限コーン型	NAT 配下の端末がアクセスした端末からアクセス可	厳しめ
ポート制限コーン型	NAT 配下の端末がアクセスした端末とポート番号からだけアクセス可	制限コーン型より厳しい
シンメトリック型	通信元の端末と通信先の端末が 1 対 1 の場合にしか使えない	かなり厳しい

4.3 WebRTC

WebRTC(Web Real-Time Communications) とは， ブラウザでリアルタイムコミュニケーションを実現するための仕組みである． つまり，P2P 通信を利用して端末間の相互接続が可能である．プロトコルには UDP が使われているため，高速な通信ができる．WebRTC による P2P 通信をするためには，シグナリングと ICE(Interactive Connectivity Establishment) が必要である．

シグナリングとは，通信相手の IP アドレス情報やポート番号等の情報を解決する手法である．ICE は後述する NAT 越え通信のためのセッション確立のための情報交換を行うプロトコルである．

4.4 NAT 越えと P2P 通信

WebRTC を用いてプライベートネットワーク内の端末同士の P2P 通信を行うためには， NAT 越えが必要となる．この時，シグナリングと ICE という仕組みが利用される．端末同士が P2P セッションを確立するには，シグナリングにより通信相手の IP アドレスや接続ポート番号等の情報を互いに交換しなければいけないため，どちらの端末からもアクセスできるシグナリングサーバが必要となる．

ICE とは STUN や TURN などの NAT 越えの手順をまとめたものであり，通信できそうな候補を集め，シグナリングにより相手とその候補を交換し，相手との通信を試みる仕組みである．また,STUN による P2P 通信を図 3 に,TURN による通信を図 4 に示す．

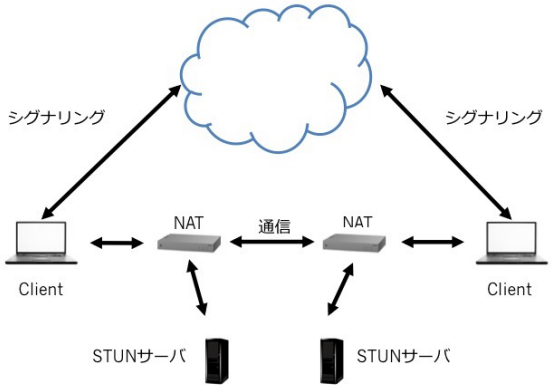

図 3　STUN による P2P 通信

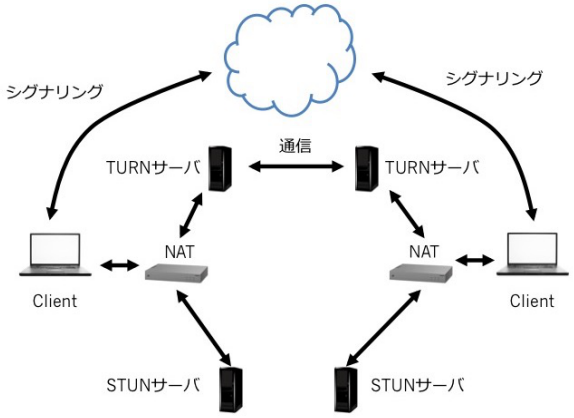

図 4　TURN によるサーバ経由通信

5. 想 定 環 境

災害時の通信環境を想定したものを図 5 に示す．この時, プライベートネットワーク同士が NAT ルータを介してアドホックに接続した際に自律的にシグナリングサーバ, STUN/TURN サーバを一意に検出し，NAT 越えの P2P 通信による情報共有が行える仕組みを備えた Wi-Fi AP 兼 NAT ルータを用いることを想定している．

災害時においてインターネットに繋がらない状況でも, この NAT ルータの Wi-Fi につなぐことで,Wi-Fi に繋がっている端末同士が情報共有を行うことが可能になる. 本研究ではこのような通信環境を想定して検討を行った．

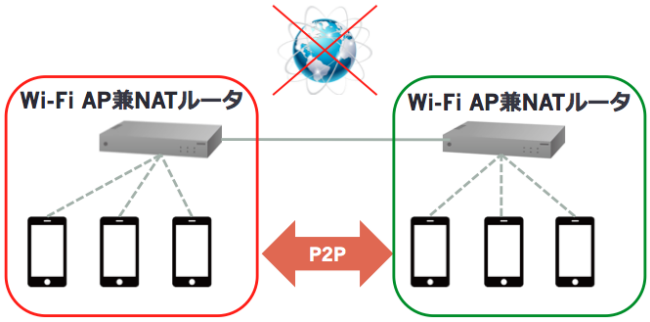

図 5　想 定 環 境

6.　ローカル環境における NAT 越え実験

6.1　動作の流れ

本研究での端末の動作の流れを図 6 に示す．これには NAT 越え技術と P2P 通信技術が使用されている．動作は以下の通りとなっている．

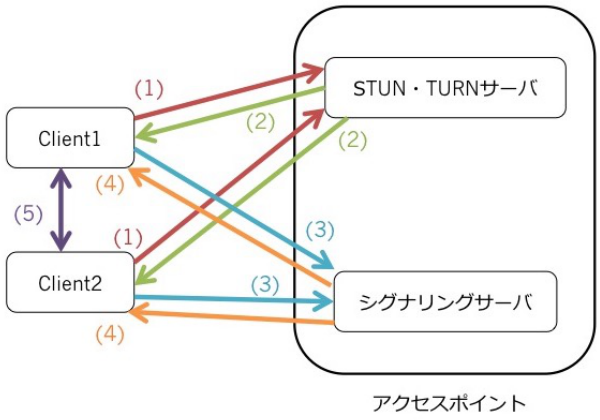

図 6　NAT 越えと P2P 通信の処理

1)　相手と通信するには相手の外から見た IP アドレスが必要であるため，STUN・TURN サーバに自分の外から見た IP アドレス情報を問い合わせる．

2)　自分の外から見た IP アドレス情報と様々な相手との経路情報を獲得．これにより，NAT 越え問題を解決．

3)　相手と通信したいことを通知．

4)　相手の情報や通信経路情報を交換．

5)　P2P 通信または TURN による通信の開始．

6.2　実 験 環 境

本研究では，WebRTC のオープンソースソフトウェアである easyRTC，STUN 兼 TURN サーバが構築できるオープンソースソフトウェアである coturn を用い，実験用のローカル環境を構築した．その環境を図 7 に示す．これにより，ローカル環境における NAT 越え情報共有の実験を行う．

また easyRTC は WebRTC のビデオ・オーディオ・データアプリケーションなどが利用でき,Node.js の WebSocket 実装である Socket.io で構築されたシグナリングサービスを使用している．

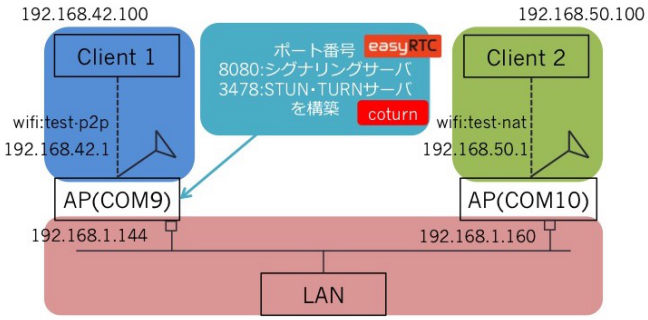

図 7　実 験 環 境

無線 LAN を搭載した Debian GNU/Linux8 ubilinux サーバ 2 台にそれぞれ hostapd をインストールし，Wi-Fi AP 兼 NAT ルータとして動作させた．このルータの仕様を表 2 に示す．ESSID は AP(COM9) が”test-p2p”，AP(COM10) が”test-nat”となっている．これにより 2 つの異なるプライベートネットワークを作成した. この時の Wi-Fi の仕様を表 3 に示す.

Client としては PC2 台をそれぞれ別の Wi-Fi AP に繋げて設置した．IP アドレスは Client1 が 192.168.42.100，Client2 が 192.168.50.100 となっている．

サーバとして，アクセスポイント (COM9) のイーサネットである IP アドレス 192.168.1.144 のポート番号 8080 にシグナリングサーバである easyRTC を，ポート番号 3478 に STUN 兼 TURN サーバである coturn を構築した．またシグナリングサーバにおける STUN/TURN サーバの設定は,IP アドレス 192.168.1.144 のポート番号 3478 とした．2 台の AP とサーバは有線 LAN でつなぎ，同じネットワーク内となっているため，ネットワークは全部で赤色と青色と緑色の 3 つとなっている．そしてプライベートネットワーク間で情報共有ができるかどうかを試す．今回は Client1 から Client2 へ test.zip という 2MB の ZIP ファイル送受信を行う．

表 2　Wi-Fi AP 兼 NAT ルータの仕様

CPU	Genuine Intel(R) CPU 4000@500 MHz
Storage	Compact Flash 6GB
OS	Debian GNU/Linux8 ubilinux
WirelessLAN	IEEE 802.11n

表 3　Wi-Fi の仕様

SSID	プロトコル	セキュリティの種類	ネットワーク帯域	ネットワークチャンネル
test-p2p	802.11n	WPA2-パーソナル	2.4GHz	7
test-nat	802.11n	WPA2-パーソナル	2.4GHz	4

6.3 実験結果

Client1 における Wireshark によるパケットキャプチャを図 8・図 9 と図 10 に，Client2 におけるものを図 11 と図 12 に示す.

図 8 から,STUN プロトコルで Client1 とサーバ間で Binding Request が互いに送られていることが分かる. また互いに Binding Success Response とあるように, 成功していることが分かる. これにより,Client1 である 192.168.42.100 とサーバである 192.168.1.144 の間で STUN による P2P 通信が可能になる. このことは次の図から見て分かる. 図 9 から，プロトコル DTLS で Client1 とサーバが Application Data のやり取りをしていることが分かる．これは先ほど述べたように,STUN による Binding Request の成功により,P2P 通信が可能となったためである．図 10 は図 9 のキャプチャの 1 つをより詳しく見たものである. これによると,Src: 192.168.42.100 Src Port: 50909 と Dst : 192.168.1.144 Dst Port: 57775 でのやり取りであることが分かる. これは図 8 の 3,4 行目の info から分かるように,Binding Success Response による IP アドレスとポート番号が使用されている.

続いて Client2 について見ていく. 図 11 から,Client2 とサーバ間で TURN による情報共有が行われていることが分かる. このときプロトコルが STUN となっているが, これは Wireshark の仕様による表記であり, 本来は TURN プロトコルが使用されている．また図 12 から,Client2 の Peer, つまり通信相手が Client1 の IP アドレス 192.168.42.100 であることが分かる. つまり,Client2 の通信相手は Client1 であることが分かる．以上のことから Client1 から Client2 への情報共有に成功していることが分かる．このことよりローカル環境における 2 つのプライベートネットワークを接続した NAT 越え情報共有ができていることが確認できた.

Source	Destination	Protocol	Info
192.168.42.100	192.168.1.144	STUN	Binding Request user: cA3u:vN5T
192.168.1.144	192.168.42.100	STUN	Binding Request user: vN5T:cA3u
192.168.42.100	192.168.1.144	STUN	Binding Success Response XOR-MAPPED-ADDRESS: 192.168.1.144:57775
192.168.1.144	192.168.42.100	STUN	Binding Success Response XOR-MAPPED-ADDRESS: 192.168.42.100:50909

図 8　Client1 におけるパケットキャプチャ 1

Source	Destination	Protocol	Info
192.168.42.100	192.168.1.144	DTLSv1.2	Application Data
192.168.42.100	192.168.1.144	DTLSv1.2	Application Data
192.168.1.144	192.168.42.100	DTLSv1.2	Application Data
192.168.42.100	192.168.1.144	DTLSv1.2	Application Data
192.168.42.100	192.168.1.144	DTLSv1.2	Application Data
192.168.42.100	192.168.1.144	DTLSv1.2	Application Data
192.168.42.100	192.168.1.144	DTLSv1.2	Application Data

図 9　Client1 におけるパケットキャプチャ 2

```
Internet Protocol Version 4, Src: 192.168.42.100, Dst: 192.168.1.144
User Datagram Protocol, Src Port: 50909 (50909), Dst Port: 57775 (57775)
```

図 10　Client1 におけるパケットキャプチャ 3

Source	Destination	Protocol	Info
192.168.1.144	192.168.50.100	STUN	ChannelData TURN Message
192.168.1.144	192.168.50.100	STUN	ChannelData TURN Message
192.168.1.144	192.168.50.100	STUN	ChannelData TURN Message
192.168.1.144	192.168.50.100	STUN	ChannelData TURN Message
192.168.1.144	192.168.50.100	STUN	ChannelData TURN Message
192.168.50.100	192.168.1.144	STUN	ChannelData TURN Message

図 11　Client2 におけるパケットキャプチャ 1

Source	Destination	Protocol	Info
192.168.50.100	192.168.1.144	STUN	Send Indication XOR-PEER-ADDRESS: 192.168.42.100:50909
192.168.1.144	192.168.50.100	STUN	Data Indication XOR-PEER-ADDRESS: 192.168.42.100:50909

図 12　Client2 におけるパケットキャプチャ 2

6.4 考　　察

これらの結果からプライベートネットワーク間での情報共有は,Client1 とサーバ間では STUN が成功し,P2P 通信がなされており,Client2 とサーバ間では TURN による中継通信となっていることが分かった．これは Client2 が多段 NAT となっているからだと考えられる. STUN はその特性から NAT の最も外側から見た（すなわちサーバ側から見た) アドレスしか知り得ないため, 多段 NAT に対応することができない．またその場合,TURN サーバがピア間のパケットを中継することによってピア間の通信を実現されている．以上のことから Client1 では P2P 通信,Client2 では多段 NAT となってしまっているため,TURN による通信になったと考えられる.

7. まとめと今後の課題

最終目標に対し, ローカル環境における STUN/TURN を用いた NAT 越えと情報共有が可能であることが確認できた.

今後は，まず STUN を用いて P2P 通信ができるようにし, プライベートネットワーク同士が NAT ルータを介してアドホックに接続した際に自律的にシグナリングサーバ, STUN/TURN サーバを一意に検出し，NAT 越えの P2P 通信による情報共有が行える手法を検討する.

謝　　辞

本研究の一部はお茶の水女子大学と情報通信研究機構との共同研究契約に基づくものである.

文　　献

[1] 間瀬憲一” モバイル・アドホックネットワーク ”， シンポジウム (47)，pp.13-26，2002 年 3 月.
[2] 鈴木秀和，渡邊晃” 通信グループに基づくサービスの制御が可能な NAT 越えシステム”，マルチメディア，分散，協調とモバイル (DICOMO2009) シンポジウム，pp.372-378，2009 年 7 月.
[3] 宮崎悠, 鈴木秀和, 渡邊晃” 端末に依存しない NAT 越えシステムの提案と実装”，マルチメディア，分散，協調とモバイル (DICOMO2008) シンポジウム，pp.587-592，2008 年 7 月.
[4] 黒田隼之輔, 中山泰一” TCP における STUN を用いた対称型 NAT 越え手法の実装と評価”，情報処理学会全国大会講演論文集，73rd,p.3.421-3.422，(2011).
[5] 竹川知孝, 佐藤未来子, 並木美太郎 ” NAT 越えの通信を実現する STUN 及び TURN の 実環境への適用可能性について”，全国大会講演論文集，74rd,p.1.13-1.14,(2012).
[6] 宮崎悠, 鈴木秀和, 渡邊晃” 端末の改造が不要な NAT 越え通信システム NTSS の提案と評価”，情報処理学会論文誌，Vol.51,

No.9，p.1873-1880，(Sep. 2010).
[7] 鈴木秀和, 渡邊晃” 通信グループに基づくサービスの制御が可能な NAT 越えシステムの提案”，情報処理学会論文誌，Vol.51，No.9，p.1881-1891，(Sep. 2010).
[8] easyRTC
https://easyrtc.com/, 2016 年 10 月参照.
[9] STUNTMAN
http://www.stunprotocol.org/, 2016 年 11 月参照.
[10] coturn
https://github.com/coturn/coturn, 2016 年 9 月参照.
[11] udonchan.”WebRTC における NAT 越えの課題へのアプローチ” Qiita.
http://qiita.com/udonchan/items/693e43b13269207f904a, 2016 年 12 月参照.
[12] がねこまさし.”そして壁の向こうへ。 NAT/Firewall を越えて通信しよう—WebRTC 入門 2016” HTML5Experts.jp.
https://html5experts.jp/mganeko/20618/, 2016 年 6 月参照.
[13] がねこまさし.”壁を越えろ！ WebRTC で NAT/Firewall を越えて通信しよう” HTML5Experts.jp.
https://html5experts.jp/mganeko/5554/, 2016 年 6 月参照.
[14] 吉川徹.”WebRTC で変わる Web の未来” QCon Tokyo.
http://www.qcontokyo.com/data_2013/ToruYoshikawa_QConTokyo2013.pdf, 2016 年 9 月参照.
[15] 望月いちろう.”WebRTC の現状 - STUN/TURN・シグナリングサーバーについて” 望月いちろうの README.md.
http://www.utali.io/entry/2016/09/07/142102, 2016 年 12 月参照.
[16] Sam Dutton.”WebRTC in the real world: STUN, TURN and signaling” HTML5ROCKS.
https://www.html5rocks.com/en/tutorials/webrtc/infrastructure/, 2017 年 2 月参照.