

大規模災害時のインターネット非接続時における XMPP に 基づく情報共有システムの検討

于 卉¹ 大和田 泰伯² 小口 正人¹

概要：人々の生活において、また、経済の発展においても、ネットワークはより便利で効率的な手段を提供し、重要な役割を担っている。よって、連絡や情報の取得手段を選択する際に、人々は電話網だけではなく、電子メール、SNS などのインターネットを介する方法も優先的に考えている。そのため、地震等の自然災害により、ネットワークインフラが断片的に壊れることによる通信不能や通信障害が発生する際に、物資や避難場所などの必要な情報及び安否の確認情報が取得できなければ、人々に不便と困惑を招く可能性がある。また、支援者や避難所においては、被災程度の把握及び災害の関連情報を配信するため、被災者人数、避難場所などの避難者情報も必要である。対応する取得手段も必要である。本研究では災害時の烈悪なネット状況に対応するため、断片的なネットワークや XMPP に基づくサーバを利用し、災害時の大規模配信ができることを目的とし、被災者と支援者が繋がることのできるアプリケーションを作る。また、各避難所間の情報交換ができるようするために、災害時の不安定で不安全な接続状況に対応できる再接続方法を提案する。

An Initial Study of Local Communication Based on XMPP Without Depending on the Internet at A Large-Scale Disaster

HUI YU¹ YASUNORI OWADA² MASATO OGUCHI¹

1. はじめに

現代社会はネットワーク化がますます進んでいる。インターネットは、欠かせない社会の一部になっており、重要な情報基盤として、広く普及している。人々の生活においても、経済の発展においても、ネットワークはより便利で効率的な手段を提供し、重要な役割を担っている。よって、連絡や情報の取得手段を選択する際に、人々は電話網だけではなく、電子メール、SNS などのインターネットを介する方法を優先的に考えている。

そのため、地震等の自然災害により、ネットワークインフラが断片的に壊れることによる通信不能や通信障害が発生する際に、物資や避難場所などの必要な情報及び安否の確認情報が取得できなければ、人々に不便と困惑を招く可

能性がある。また、支援者や避難所においては、被災程度の把握及び災害の関連情報を配信するため、被災者人数、避難場所などの避難者情報も必要である。そのために、対応する取得手段も必要である。

一般的に、図 1 のように、SNS サービスを提供する会社は同じデータベースを使用する複数のサーバを相互接続され、全体が 1 つのシステムとしてサービスを提供するというクラスタの方法で利用者にサービスを提供している。一方、ロードバランサを使い、リクエストを分散させることで、すべてのサーバは処理を分担することができる。また、もしあるサーバに故障が起きれば、障害の発生したサーバをシステムから切り離し、最大限にサービスの品質が保証できる。ユーザにおいては、どちらのサーバに登録するかは関心なく、ネットワークがサポートされていれば、どこでも利用できる。

しかし、外部のアクセスネットワークに障害が起きるなどでインターネットにつながらない場合には、これらが対応不能になる可能性がある。さらに、各会社はシステムセ

¹ お茶の水女子大学
〒 112-8610 東京都文京区大塚 2-1-1

² 情報通信研究機構
〒 980-0812 宮城県仙台市青葉区片平 2-1-3

セキュリティやユーザの個人情報を守るため、異なる会社のアプリケーション間の情報交換が制約されている。

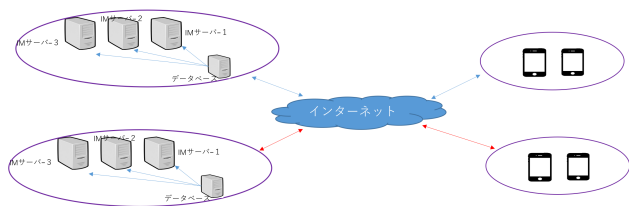


図 1 平常時の通信システム

つまり、災害時には情報の大規模配信の必要があるが、対応手段が少ない。また、インターネット全体からの通信は遮断されたが、一部分が通信可能なローカル環境が存在している可能性がある。

そのため、本研究は災害時の大規模情報配信ができることを目的とし、被災者と支援者が繋がることができるアプリケーションについて構築を行っていく。また、各避難所間の情報交換ができるようにするため、不安定で不安全なネット状況に対応できる再接続方法について提案と検討を行っていく。

2. 関連技術

2.1 XMPP

XMPPとは、現在広く使用されているインスタントメッセージ (Instant message) システムの核心プロトコルの一つである。ユーザはJIDというアカウントによって、ユーザ間、ユーザ・サーバ間で通信する。

JIDとは、「任意の名前@登録したサーバのドメイン名」という唯一のアカウントである。特に、名前が同じであっても、異なるサーバに登録する際に、異なるユーザと推定される。つまり、JIDを利用すれば、ユーザ名の重複は問題にはならず、異なるサーバ間での通信が可能になる。

XMPPでメッセージを伝達するため、全てのメッセージが以下のようにXML stanza を標準にして、パッケージされる。

- (1) Message stanza によって、ユーザ間、ユーザ・サーバ間で、メッセージを発信する。図2に示す。
- (2) Presence stanza によって、複数の受け手に状態を知らせる。(マルチキャスト的な一方向の通知) 図3に示す。
- (3) IQ stanza によって、ユーザ間、ユーザ・サーバ間で、要求と応答をする。idは要求と応答の番号を取る。図4に示す。

```
<message
  to="test@192.168.111.174" id="tJhQ8-143"
  type="chat"
  from="admin@192.168.111.174">
  <thread>tERD40</thread>
  <paused xmlns="http://jabber.org/protocol/chatstates"/>
</message>
```

図 2 Message stanza

```
<presence
  to='a@192.168.111.174'
  id='3NIsU-11'
  type='subscribed'>
</presence>
```

図 3 Presence stanza

```
<iq
  type="get"
  id="368-6"
  from="192.168.111.175"
  to="test@192.168.111.175/AndroidClient">
  <ping xmlns="urn:xmpp:ping"/>
</iq>
```

図 4 IQ stanza

2.2 Openfire サーバ

Openfire とは、XMPP に基づく小範囲通信に向いている軽量級のサーバの一つである。使用方法が簡単で、基本的な通信システムの機能が付いており、プラグインがサポートするため、拡張性が高い。

図5のように、Apache MINA を使用することにより、メッセージの受け取り、及びタイプの判断と処理が別々に行われる。開発者がソケット (socket) と NIO (Java new I/O) の複雑な操作を考えず、処理の機能やロジックに集中できる。

2.2.1 Openfire のフェデレーション

各避難所間の情報交換ができるためには、各避難所に置くサーバ同士での接続が必要である。災害時に、存在している避難所以外の新たな避難所が生じる可能性がある。またさらに、偽物の避難所が生じる可能性もある。もし、分散されたサーバが認証せず、誰でも接続できる状態になれば、相互接続が簡単になるが、接続セキュリティが全く守られないようになる。これによる問題が起きる可能性がある。そのため、災害時であっても、一定程度のセキュリティ対策が必要である。

Openfire のフェデレーションとは、別々のデータベースを利用した独立のサーバ同士間での認証機能。これを図6に示す。

Openfire は信頼関係を締結したいサーバ同士の IP アドレスを手入力するという方法で、認証を行う。信頼できるサーバ同士で、発信と受信ができる。また、サーバに登録したユーザ同士でも、情報のやり取りができる。しかし、

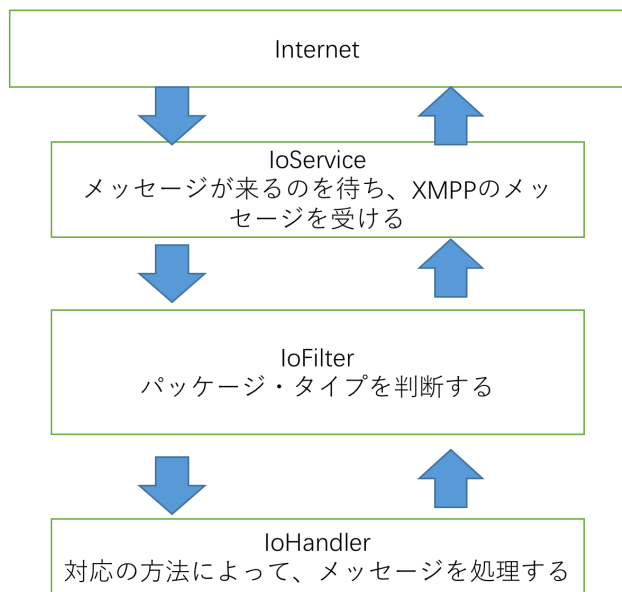


図 5 openfire の動き流れ

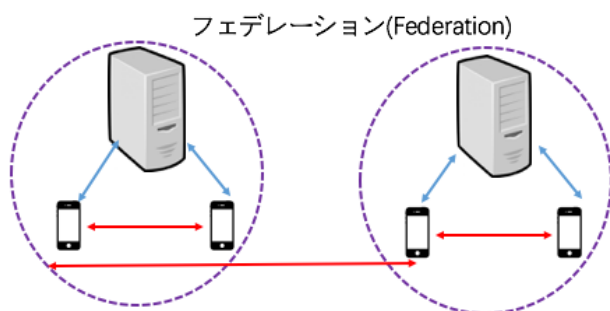


図 6 フェデレーション

これらはサーバ間の接続セキュリティが守られるが、柔軟性が少ないと思われる。

3. 提案システム

以上の考察に基づき、図7のようなシステムを提案した。

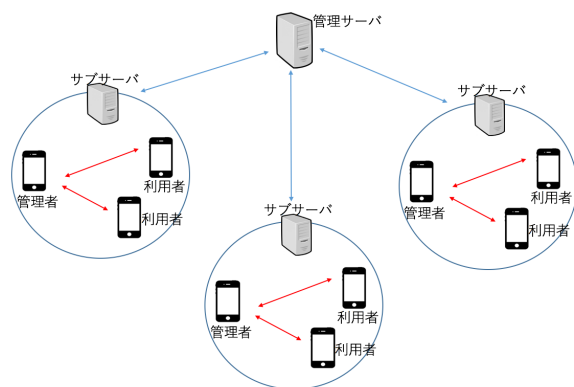


図 7 提案システム

サーバは管理サーバとサブサーバの2種類を想定し、

Openfire やデータベースを搭載し、それぞれ避難本部と各避難所に設置する。全てのサブサーバは管理サーバと繋がっていて、情報の同期を行っていく。

ユーザがアプリケーションを利用し、情報のやり取りができる。そのための対応可能な仕組みが必要である。これらはチャットルームと類似の仕組みをベースにして情報に対応する説明を付けるものを作っていく。

ユーザも管理者と利用者の2種類を想定し、それぞれ支援者と被災者に対応する。管理者とは、信頼できる発信者であり、物資や避難所に関連する情報を配信する役割に担う。利用者とは、避難者である。

3.1 システム概要

続いて提案するシステムについて、詳しく説明する。これには、通信ベース、再接続対策と情報のやり取りの3種類があるため、それぞれについて説明する。

3.1.1 通信ベース

(1) 管理サーバ

管理サーバは各サブサーバの情報を集めたり、同期したりするなどの管理を行う。

各サブサーバは通常の状態では、管理サーバと直接接続している。信頼関係を締結しているため、管理サーバの方は全ての存在しているサブサーバの情報を集め、全てのサブサーバに送る。言い換えると、「server-list」というサーバ情報リストの同期を行う。

さらに、各サブサーバは登録した管理者の情報を管理サーバへ送ったり、他のサブサーバの管理者情報を受け取ったりする。言い換えると、管理サーバは「user-list」という管理者リストを作り、同期を行う。

もう一つの機能は、管理サーバもサブサーバ間の認証の管理を行うことである。

単純な同期を行えば、全てのサブサーバと管理サーバだけに繋がっていても良い。しかし、「user-list」による特定の二つのサブサーバ間での連続的な通信を行う場合に、全てのメッセージが管理サーバに通されれば、管理サーバの負担が大きくなり、システムも性能が低下する可能性がある。そのため、図8のようなエンドツーエンドに類似したサーバ間通信を提案する。

まず、各サブサーバは管理サーバだけを信頼している。①のようにサーバ間での連続した情報交換を行う際に、②と③のように、管理サーバは両方へ相手の情報と信頼許可を送り、信頼関係の締結を助ける。これで、④のように一時的な認証を確立させる。この認証により、サブサーバ間の通信効率が高くなり、管理サーバの負担も下がる。そして、連続した通信が終了すれば、この一時的な認証は取り消す。

(2) サブサーバ

サブサーバとは、ユーザを直接管理しているサーバである。情報のやり取り機能を利用して、配信を行う。

利用者は自分自身の被災情報、避難場所などをサブサーバに送り、管理者に被災状況を把握してもらう。需要があれば、各サブサーバは各被災範囲内の被災状況を管理サーバへ報告することもできる。

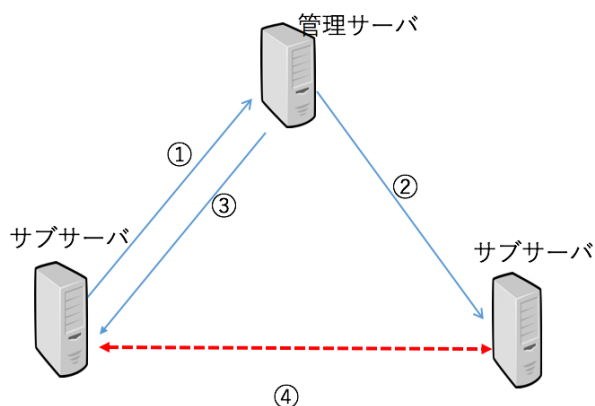


図 8 サーバツサーバ通信

3.2 情報のやり取り

災害時は平常時に比べ、情報の取得手段が少なくなり、本物と偽物が混在する可能性があって、人々は不安と困惑状態である。よって、情報の真実性を守るため、一定程度の認証が必要である。そこで、暗証番号が合えば、管理権が取得できるという方法を提案する。また、個人により、「有用な情報」の意味が違うので、内容についての説明を並べし、利用者に周知する必要があると考える。

全ての情報のやり取りはアプリケーションの方で行っていくため、今回はチャットルームと類似の仕組みをベースにして、情報内容に対応する説明が並ぶものを作っていく。

3.2.1 再接続対策

災害時、ネットワーク状況が不安定になり、及びサーバが壊れる可能性があるため、対応手段が必要である。そして、災害時に、分散されたサーバ間での接続セキュリティと情報共有可能範囲の拡張のバランスを取るため、サブサーバと管理サーバが中断された両方の場合を考え、再接続策を提案する。

以下でこの2つの提案手法について詳しく説明する。

(1) 管理サーバが遮断された場合

管理サーバが遮断された場合、各サブサーバは前に受け取った「server-list」で、全てのサブサーバに直接接続を試行する。

管理サーバが遮断された際に、サブサーバは自分自身以外の全てのサーバが全部遮断された状態になる。

しかし、この情報は管理サーバから得られた情報であるので、正しくない可能性がある。よって、サブサーバは、このリストで、他のサブサーバに再接続を試行していく。この対策により、接続セキュリティが低くなるが、最大限に情報共有の可能な範囲が広がる。

(2) サブサーバが遮断された場合

あるサブサーバが遮断された場合、管理サーバは使用可能である。まず、このサブサーバは全てのサブサーバに直接接続を試行していく。

同時に管理サーバがこの状況を見つけることを待つ。管理サーバはこのサブサーバが中断されたことを見つけると、他のサブサーバに「このサブサーバと接続してください」というメッセージを送り、他のサブサーバはこのサブサーバと再接続を試行していく。この対策により、再接続する際にも、一定程度の接続セキュリティが守られる上に、最大限に情報共有の可能な範囲が広がる。

4. 実験環境

本研究では、ローカル環境をネット環境と Virtualbox を利用し、管理者と利用者の端末機2台、管理サーバ1台、サブサーバ9台を仮想した。これを図9に示す。これにより、実験用の環境を構築した。

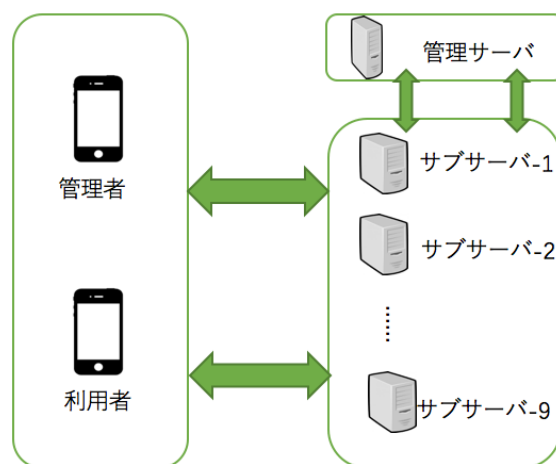


図 9 実験環境

アンドロイドに基づく端末機は OS が mac の PC で仮想的に構築した。全てのサーバは PowerEdgeR430 で仮想的に構築した。

5. 実験結果

5.1 情報のやり取り

図9のように、まずは管理者と利用者を同じサブサーバ

PowerEdgeR430	CPU	Intel Xeon 2.2GHz
	OS	Centos 7.4
	Memory	16GB
	DHH	1TB
Android	Fireware version	6.0.0

表 1 実験環境

1 に登録し、情報のやり取りをする実験を行った。

(1) 管理権の登録や個人情報の報告

図 10 のように、ユーザは管理者に登録する際に、暗証番号や配信情報の内容についての説明をサブサーバへ送信した。暗証番号が合ったため、サブサーバは管理権を取得した。

また、図 10 のように、管理者であっても、利用者であっても、自分自身の状況をサブサーバへ報告することができるようになった。



図 10 管理権の登録と個人情報の選択

(2) 発信結果

図 11 のように、お知らせリストに、管理コースと受け取りコースの二つのリストがある。管理されているグループは管理コースに表示される。これにより、発信ができるようになった。



図 11 発信結果

(3) 受信結果

図 12 のように、利用者は登録した配信グループを選択し、情報を受け取った。

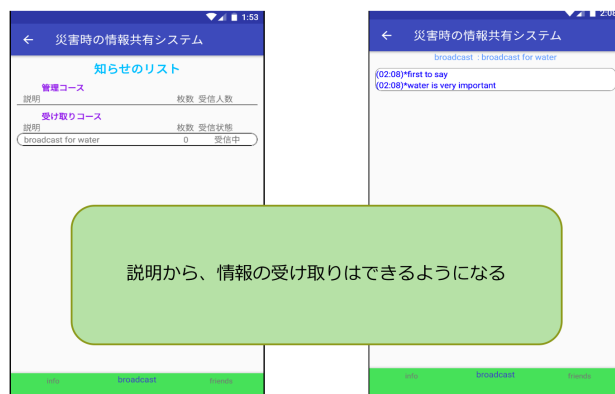


図 12 受信結果

5.2 「user-list」による通信

続いて「user-list」により、同じサブサーバに登録したユーザ間及び異なるサブサーバに登録したユーザ間の通信実験を行った。

同期した「user-list」を図 13 のように示す。赤線枠は本サブサーバの管理者で、青線枠は他のサブサーバの管理者である。図 14 のように、同じサブサーバに登録したユーザ間であっても、異なるサブサーバに登録したユーザ間であっても、通信が可能になった。



図 13 user-list

5.3 「server-list」による再接続

続いてサブサーバ同士の再接続対策についての実験を行った結果を示す。

(1) 管理サーバが遮断された場合

図 15 のように、管理サーバが遮断された際に、サブサーバは自分自身以外の全てのサーバが全部遮断さ

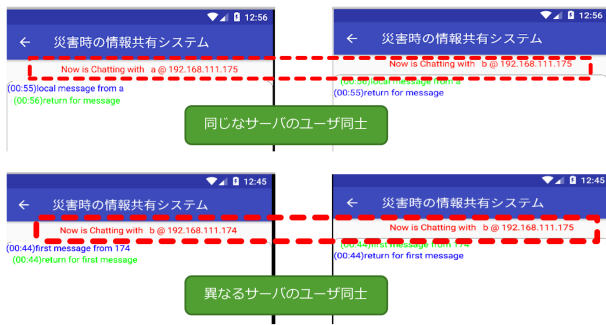


図 14 「user-list」による通信

れた状態になった。全てのサブサーバは再接続を行った。最後に、全てのサブサーバは管理サーバ以外の他のサブサーバとの接続を回復した。

平常時	ServerId : 192.168.111.284 TAG : local user-number : 1 status : Alive
	ServerId : 192.168.111.282 TAG : manager user-number : 1 status : Alive
	ServerId : 192.168.111.217 TAG : remote user-number : 1 status : Alive
	ServerId : 192.168.111.216 TAG : remote user-number : 1 status : Alive
	ServerId : 192.168.111.215 TAG : remote user-number : 1 status : Alive
	ServerId : 192.168.111.285 TAG : remote user-number : 1 status : Alive
	ServerId : 192.168.111.283 TAG : remote user-number : 1 status : Alive
管理サーバを中断	ServerId : 192.168.111.284 TAG : local user-number : 1 status : Alive
	ServerId : 192.168.111.282 TAG : manager user-number : 1 status : Dead
	ServerId : 192.168.111.217 TAG : remote user-number : 1 status : Dead
	ServerId : 192.168.111.216 TAG : remote user-number : 1 status : Dead
	ServerId : 192.168.111.215 TAG : remote user-number : 1 status : Dead
	ServerId : 192.168.111.285 TAG : remote user-number : 1 status : Dead
	ServerId : 192.168.111.283 TAG : remote user-number : 1 status : Dead
再接続	ServerId : 192.168.111.284 TAG : local user-number : 1 status : Alive
	ServerId : 192.168.111.282 TAG : manager user-number : 1 status : Dead
	ServerId : 192.168.111.217 TAG : remote user-number : 1 status : Alive
	ServerId : 192.168.111.216 TAG : remote user-number : 1 status : Alive
	ServerId : 192.168.111.215 TAG : remote user-number : 1 status : Alive
	ServerId : 192.168.111.285 TAG : remote user-number : 1 status : Alive
	ServerId : 192.168.111.283 TAG : remote user-number : 1 status : Alive

図 15 「server-list」による再接続 1

(2) あるサブサーバが遮断された場合

図 16 のように、遮断されたサブサーバは管理サーバが遮断された時と同じ方法で他のサブサーバに再接続を行った。管理サーバはこのサブサーバが遮断されたため、遮断されたサブサーバの情報と接続許可を他のサブサーバに送った。他のサブサーバは再接続を行った。最後に、遮断されたサブサーバは管理サーバ以外の他のサブサーバとの接続を回復した。

あるサブサーバが 中断された	ServerId : 192.168.111.284 TAG : manager user-number : 1 status : Alive	管理サーバは中断された サブサーバを見つける
	ServerId : 192.168.111.282 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.285 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.217 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.216 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.215 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.283 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.217 TAG : local user-number : 1 status : Alive	
	ServerId : 192.168.111.282 TAG : manager user-number : 1 status : Dead	中断されたサブサーバ
	ServerId : 192.168.111.285 TAG : remote user-number : 1 status : Dead	
	ServerId : 192.168.111.217 TAG : remote user-number : 1 status : Dead	
	ServerId : 192.168.111.216 TAG : remote user-number : 1 status : Dead	
	ServerId : 192.168.111.215 TAG : remote user-number : 1 status : Dead	
	ServerId : 192.168.111.283 TAG : remote user-number : 1 status : Dead	
	ServerId : 192.168.111.217 TAG : local user-number : 1 status : Alive	
	ServerId : 192.168.111.282 TAG : manager user-number : 1 status : Dead	中断されたサブサーバ
	ServerId : 192.168.111.285 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.217 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.216 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.215 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.283 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.217 TAG : local user-number : 1 status : Alive	
	ServerId : 192.168.111.282 TAG : manager user-number : 1 status : Dead	消息不明のサーバ
	ServerId : 192.168.111.285 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.217 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.216 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.215 TAG : remote user-number : 1 status : Alive	
	ServerId : 192.168.111.283 TAG : remote user-number : 1 status : Alive	

図 16 「server-list」による再接続 2

6. まとめと今後の課題

本研究では、災害時の大規模配信需要に対応するため、被災者と支援者が繋がることのできるアプリケーションを作成した。このアプリケーションによって避難や物資などの災害に関連する情報のやり取りができた。また、各避難所の情報交換ができるようにため、災害時の烈悪なネットワーク環境に対応できる接続方法を提案した。また、本研究は災害時に、分散されたサーバ間での接続セキュリティと情報共有可能範囲の拡張のバランスをとるため、サブサーバと管理サーバ両方を考え、再接続策を提案した。

今後は、通常時の通信や非常時の通信の評価を行っていく。また、利用者の被災状況の報告と同期機能を改善する必要がある。今後の課題としたい。

7. 謝辞

本研究で UCLA の高井峰生先生から有用なアドバイスをいただきました。また、本研究は一部、JST CREST JPMJCR1503 の支援を受けたものです。ここに感謝の意を表します。

参考文献

- [1] XMPP 入手先 <http://<https://xmpp.org/>> (参照 2017-10).
- [2] android-Developers 入手先 <<https://www.igniterealtime.org/projects/openfire/>>, (参照 2017-10).
- [3] VirtualBox に CentOS6.5 をインストール ~ 初期設定 入手先 <<http://damepg.hatenablog.com/entry/2014/08/06/090217>>, (参照 2017-10).
- [4] 仮想マシン IP 構成の概要 入手先 <<http://blog.5lcto.com/igbugs/1679539>>, (参照 2017-10).
- [5] Xmpp presentation 入手先 <<https://www.slideshare.net/javaranger123/xmpp-presentation>>, (参照 2017-10).
- [6] 総務省, 大規模災害時におけるインターネットの有効活用事例解説集 pp10-15, (参照 2017-10).
- [7] cross-domain communication using an XMPP chat guard, Raymond Haakseth, Nils Agne Nordbotten, Oddvar Brnstad, yvind Jonsson, Bengt Kristiansen, (参照 2017-10).
- [8] A Lightweight XMPP Publish/Subscribe Scheme for Resource-Constrained IoT Devices, HENG WANG, (Member, IEEE), DALJIN XIONG, PING WANG, AND YUQIANG LIU, (参照 2017-11).
- [9] Android-消息推送刻篇 入手先 <<http://www.imoooc.com/learn/358>>, (参照 2017-11).
- [10] Android 消息推送刻 入手先 <<http://www.imoooc.com/learn/223>>, 第二章 Socket 的使用和

- Mina 框架解, (参照 2017-11).
- [11] Connect to Genymotion Android emulator remotely
入手先 <http://www.sarpex.co.uk/index.php/2016/10/02/connect-genymotion-emulator-remotely/>
, (参照 2017-11).
 - [12] Linux 下 iptables 禁止端口和放端口
入手先 <http://blog.csdn.net/zht666/article/details/17505789>, (参照 2017-11).
 - [13] クラスタサービスとは?
入手先 <https://enterprisezine.jp/iti/detail/12> (参照 2017-12).
 - [14] StanzaHandler
入手先 <https://github.com/igniterealtime/Openfire/blob/master/src/java/org/jivesoftware/openfire/net/StanzaHandler.java> (参照 2017-12).
 - [15] XMPPService の一例
入手先 <https://www.javatips.net/api/woc-master-WoChat-master/xmpp/src/main/java/com/wuxiaolong/xmpp/XMPPService.java>, (参照 2017-12).
 - [16] 入手先 <https://enterprisezine.jp/iti/detail/12>, (参照 2018-02).
 - [17] 入手先 <http://itdoc.hitachi.co.jp/manuals/3020/30203R3231/PCOP0228.HTM>, (参照 2018-02).
 - [18] 入手先 <https://www.hpe.com/jp/ja/japan/icewall/federation/overview.html>, (参照 2018-02).
 - [19] 入手先 <http://www.infraexpert.com/study/networking1.html>), (参照 2018-03).