Vol.2015-IOT-31 No.6
Vol.2015-SPT-15 No.6
2015/9/25

# 多数決に基づく公開鍵決定プロトコルによる中間者攻撃への対応とその評価

猿渡 翔一郎[1]　山森 一人[1]　相川 勝[1]

**概要**：SNS や電子商取引などの，インターネットを介したサービスでは他人に秘匿すべき個人情報を扱うことが多い．これらの個人情報は第三者による傍受や改ざんを防ぐために暗号化を行うことが勧められている．暗号化の手法として，秘密鍵と公開鍵を用いる公開鍵暗号が普及しているが，公開鍵暗号方式では公開鍵が途中ですり替えられる中間者攻撃という問題が指摘されている．現在は認証局が公開鍵の正当性を保証しているが，認証局には信頼性と維持コストに問題がある．そこで，本稿では中間者攻撃対策として多数決に基づく公開鍵決定プロトコルを提案する．提案するプロトコルは認証局を用いず，ネットワークに参加しているユーザが互いに持ち合っている公開鍵を使用する．ある公開鍵が必要になったとき，参加しているユーザから必要とする公開鍵を集めて多数決をとることにより公開鍵の決定を行う．実際に提案するプロトコルのプロトタイプを仮想ネットワーク上で実装し，シミュレーションによりその効果を検証する．

# A Countermeasure against Man-in-the-middle Attack by Distributed Publickeys

## 1. はじめに

近年，SNS や電子商取引 (Electronic Commerce:EC) の普及により他人に秘匿すべき個人情報を扱う機会が多くなった．SNS や EC では，個人的な写真や文章の他，氏名，住所，電話番号などの第三者に知られるべきでない個人情報も扱っている．インターネットの仕組み上，通信路を通るこれらの情報はパケットキャプチャなどで第三者が傍受可能である．そのため，SSL(Secure Socket Layer)[1] や TLS(Transport Layer Security)[2] といったプロトコルを用いて暗号化通信を行い，他人に見られないよう個人情報を保護することが勧められている．その根底にある技術として公開鍵暗号方式が存在する．公開鍵暗号方式は公開鍵と秘密鍵のペアを用いて通信内容を暗号化し，当事者同士にしか情報を理解できないようにしている．現在，公開鍵の正当性を認証局が保証しているが，認証局自体の信頼性を誰が保証するのかという点と，認証に必要な署名の維持コストに問題がある [3].

実際，暗号化通信に関するセキュリティインシデントも報告されている．例えば，SSL の特定のバージョンで通信内容が傍受される脆弱性 (Padding Oracle On Downgraded Legacy Encryption:POODLE)[4]，一部の Lenovo 社のノートパソコンに搭載されていた自己署名証明書による脆弱性 (Superfish)[5] が挙げられる．これらのインシデントは中間者攻撃により起こるため，中間者攻撃の対策は喫緊の課題であるといえる．

本稿では中間者攻撃対策として多数決に基づく公開鍵決定プロトコルを提案する．提案するプロトコルは認証局を用いず，ネットワークに参加しているユーザが互いに持ち合っている公開鍵を使用して鍵の正当性を確認する．具体的には，ある公開鍵が必要になったとき，参加しているユーザから必要とする公開鍵を集めて多数決をとることにより公開鍵の決定を行う．実際に提案するプロトコルのプロトタイプを仮想ネットワーク上で実装し，シミュレーションによりその効果を検証する．

## 2. ICMP と暗号化通信

### 2.1 ICMP

提案するプロトコルは ICMP をベースとしているので，本節では ICMP について簡単に説明する．

[1] 宮崎大学
University of Miyazaki

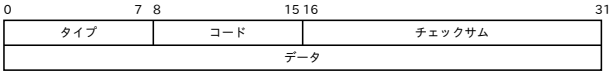


図 1　ICMP のフォーマット.

表 1　ICMP フィールド.

| タイプ | コード |
|---|---|
| 0:Echo リプライメッセージ | 0 |
| 3:宛先到達不可メッセージ | 0: ネット到達不可 |
| | 1: ホスト到達不可 |
| | 2: プロトコル到達不可 |
| | 3: ポート到達不可 |
| | 4: フラグメントが必要で DF をセットしている |
| | 5: ソースルート失敗 |
| 8:Echo メッセージ | 0 |

表 2　RSA 暗号のパラメータの説明.

| 数値 | 説明 |
|---|---|
| $p$ | 大きな素数 |
| $q$ | $p$ とは異なる大きな素数 |
| $n$ | $n = pq$ |
| $\lambda$ | $\lambda = \gcd((p-1),(q-1))$ |
| $e$ | $\lambda$ と互いに素となる数 |
| $d$ | $d = e^{-1} (\mathrm{mod}\ \lambda)$ |

ICMP(Internet Control Message Protocol)[6] は図 1 のようなフォーマットの制御メッセージ用プロトコルで，よく知られている利用方法としてネットワークの疎通確認に用いられる．図 1 中のタイプとコードの一部について，表 1 にその意味を示す．

### 2.2　公開鍵暗号

公開鍵暗号は，公開鍵と秘密鍵の 2 つの鍵を 1 組とする暗号方式である．公開鍵は通信したい相手に渡す必要があり，誰から見られてもよい．一方，秘密鍵は公開せずに他人に知られてはならない．

公開鍵暗号の一手法である RSA 暗号 [7] [8] について簡単に説明する．RSA 暗号は 1977 年に Rivest, Shamir, Adleman によって発明され，それぞれの頭文字をとり RSA 暗号と名付けられている．RSA 暗号の安全性は大きな数の素因数分解を速く解くアルゴリズムが見つかっておらず，秘密鍵を割り出すことが困難なことで担保している．そのため，現在鍵のビット長は 1024，2048 ビットが使用されることが多い [8]．RSA 暗号に必要なパラメータとして，$p, q, n, \lambda, e, d$ があり，それぞれのパラメータの説明を表 2 に示す．これらのパラメータの中で公開鍵は $e, n$ で，秘密鍵は $d, p, q$ となる．あるメッセージ $m$ に対し暗号化を行う場合は式 (1) の計算を行い，復号については式 (2) の計算を行う．

$$c = m^e (\mathrm{mod}\ n). \tag{1}$$

$$m = c^d (\mathrm{mod}\ n). \tag{2}$$

公開鍵暗号に用いられるアルゴリズムには上記で示した RSA 暗号以外に，ElGamal 方式，楕円曲線暗号 (Eliptic Curve Cryptosystems:ECC) などがある [3].

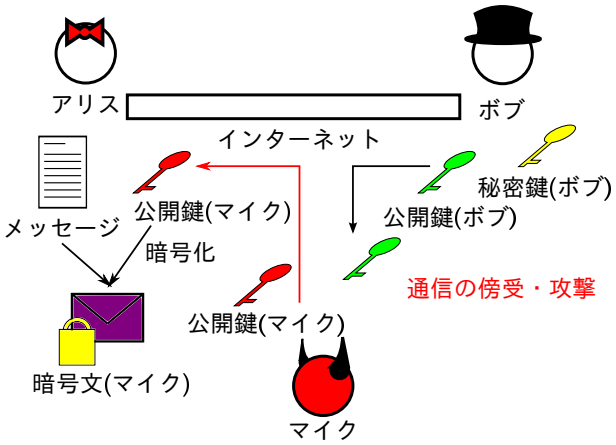


図 2　鍵入手時における中間者攻撃.

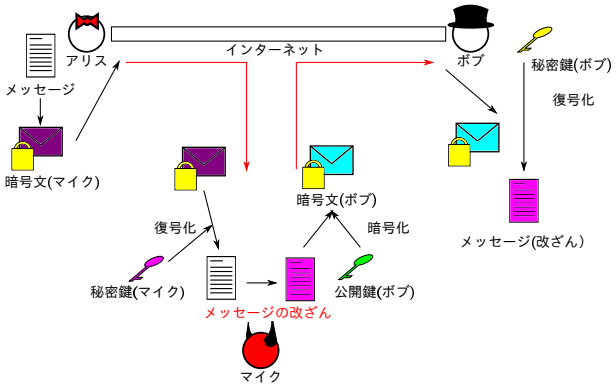


図 3　メッセージ送信時における中間者攻撃.

### 2.3　中間者攻撃

公開鍵暗号には公開鍵が誰のものは分からないという問題が存在する．この問題を悪用した攻撃として中間者攻撃 (Man-in-the-middle Attack:MITM) がある．例としてユーザ・アリスとユーザ・ボブの通信に対して，能動的攻撃者マイクが MITM を行う場合を図 2 と図 3 を用いて説明する．

前提として，マイクはアリスとボブのすべての通信を傍受することができる．アリスがマイクの公開鍵をボブの鍵と偽って入手させられるときの様子を図 2 に示し，以下にその手順を説明する．

(1) アリスはボブに公開鍵 (ボブ) を要求する．
(2) ボブはアリスに公開鍵 (ボブ) を渡そうとする．
(3) マイクはボブが送った公開鍵 (ボブ) を奪いとる．
(4) マイクはアリスに公開鍵 (マイク) を渡す．
(5) アリスは入手した公開鍵 (マイク) を使用して秘密にしたいメッセージを暗号化する．

次に，図 3 を使って，アリスがボブに暗号文を送信するときの手順を説明する．

(1) アリスは暗号文 (マイク) をボブに送信しようとする．
(2) マイクはアリスが送信した暗号文を奪いとる．

(3) マイクは秘密鍵 (マイク) を使用して暗号文を復号する.
(4) マイクはメッセージの改ざんを行う.
(5) マイクは公開鍵 (ボブ) を使用して改ざんしたメッセージを暗号化する.
(6) マイクはボブに暗号文 (ボブ) を送信する.
(7) ボブは受信した暗号文を秘密鍵 (ボブ) を使用して復号する.

以上のマイクの MITM によって，アリスのメッセージは改ざんされうる.

### 2.4 認証局

MITM は，公開鍵の正当性を 1 人では判断できないことが原因で起こる．この問題を解決するために現在用いられている手法は，認証局による公開鍵の認証である．認証局は公開鍵に対する署名を作成し，証明書を作成する．証明書の署名を確認することにより公開鍵の所有者を確認することができる．しかし，認証局には自己署名証明書の存在や認証局自身の信頼性，証明書の維持コストなど問題は残る [3].

## 3. OpenFlow

### 3.1 OpenFlow とは

提案手法の実装には OpenFlow を用いるため OpenFlow について説明する.

OpenFlow は，Software-Defined Networking(SDN) における，最初の標準化されたプロトコルである [9]．従来，ネットワーク機器はデータ転送とネットワーク制御を 1 台の機器で行っている．一方，OpenFlow ではデータ転送とネットワーク制御をそれぞれ OpenFlow スイッチ，OpenFlow コントローラに分離している．これにより，複数台の OpenFlow スイッチを 1 台の OpenFlow コントローラで制御できるなど，ネットワーク制御を柔軟に行えるようになり，ネットワークの環境変化に簡単に対応できるようになった.

### 3.2 OpenFlow の動作

OpenFlow の動作について図 4 を用いて説明する.

まず，OpenFlow コントローラは，OpenFlow スイッチと接続時に OpenFlow プロトコルのバージョンの整合性や必要なフローエントリの情報のやりとりを行う．OpenFlow スイッチは，OpenFlow コントローラから受け取ったフローエントリをフローテーブルに書き込む.

OpenFlow スイッチはパケットが到着したとき，パケットのヘッダ情報から適切な通信先へとパケットを送信する.

## 4. 提案手法と評価

### 4.1 多数決プロトコル

提案する多数決プロトコルについて説明する．多数決プロトコルのプロトコルフォーマットを図 5 に示す.

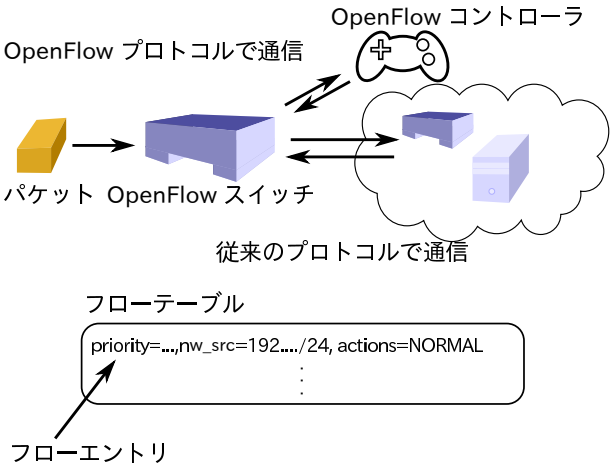


図 4 OpenFlow によるパケット処理.

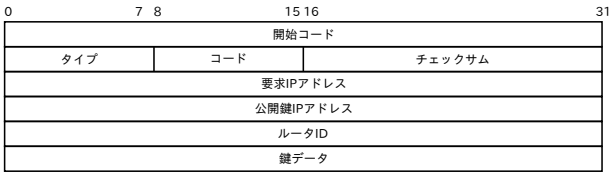


図 5 多数決プロトコルのフォーマット.

表 3 多数決プロトコルのタイプとコード.

| タイプ | コード |
|---|---|
| 0:鍵登録 | 0:登録 |
| | 1:登録済 |
| 1:プロトコル参加 | 0:参加要求 |
| 2:鍵要求 | 0:鍵要求 |
| | 1:鍵応答 |

**開始コード** 目印として先頭に 0 を 32 個並べたデータが格納されている.
**タイプ** パケットの分類を示すもので，詳細を表 3 に示す.
**コード** タイプに対するオプションで，詳細を表 3 に示す.
**チェックサム** タイプとコードから算出されるチェックサムである.
**要求 IP アドレス** アクションを起こす IP アドレスである.
**公開鍵 IP アドレス** 公開鍵を要求するときに指定する IP アドレスである.
**ルータ ID** ルータを一意に定める値である.
**鍵データ** 公開鍵 IP アドレスの公開鍵のデータを保存する．この値は 32 ビットの整数倍でなくてもよい.

以降の実験では送信元ポート番号は 41020, 宛先ポート番号は 41022 を用いる.

### 4.2 多数決プロトコルへの参加

多数決プロトコルに参加するには参加要求パケットを送信する．参加要求パケットを送信することにより，ネットワーク上のコントローラが持つ参加者リストに当該ユーザが登録される．参加者リストは，参加要求パケットを送信

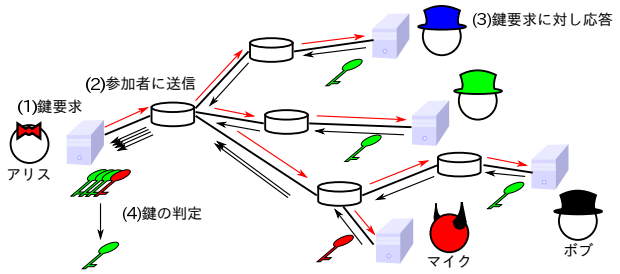


図 6　公開鍵の要求.

表 4　実験環境.

| | |
|---|---|
| OS(Ubuntu) | Version 14.04.1 |
| Open vSwitch | Version 2.0.2 |
| Ryu | Version 3.15 |
| セグメント数 | 6 |
| ルータ数 | 30 |
| ルータ 1 台あたりのホスト数 | 2 |

した IP アドレスのリストである.

### 4.3　公開鍵の登録

公開鍵を登録するには，鍵登録パケットを送信する．鍵登録を参加要求と別にしている理由は，参加要求時に IP アドレスのなりすましにより，偽装した鍵を登録されにくくするためである．そのため，登録時には，自身の IP アドレスだけでなくルータ ID も使用する．

### 4.4　公開鍵の要求

ある公開鍵を取得したい場合，鍵要求パケットを送信する．そのときの動作を図 6 に示す．ユーザ・アリスがユーザ・ボブの公開鍵を入手する場合を例に説明する．

**鍵要求**　アリスはボブの公開鍵を入手するために，ボブの IP アドレスを指定した鍵要求パケットを送信する．

**参加者に送信**　ルータは鍵要求パケットを受信すると，受信ポート以外のポートに鍵要求パケットを送信する．

**鍵要求に対し応答**　ネットワーク上の多数決プロトコル参加者は，鍵要求パケットを受信したとき，自身の持つ公開鍵リストからボブの公開鍵を検索する．見つかった場合は要求 IP アドレスに対し鍵データを付加して送信する．

**鍵の判定**　アリスは一定時間後，返答のあった鍵を種類毎に数え，多数決によりボブの公開鍵を決定する．

### 4.5　実験環境

実験環境を表 4 に示す．プロトコルの実装には OpenFlow を用いる．参加者リストを保存するコントローラは，OpenFlow コントローラ上に実装する．実験に使用するコンピュータが 1 台であることから，1 台の物理コンピュータ上に複数の仮想ネットワークを構築可能な mininet[10] を使用する．世界中で使用されることを想定し，図 7 に示す通り，ネットワークには A 国から F 国の計 6 カ国の仮想的な国を設定する．

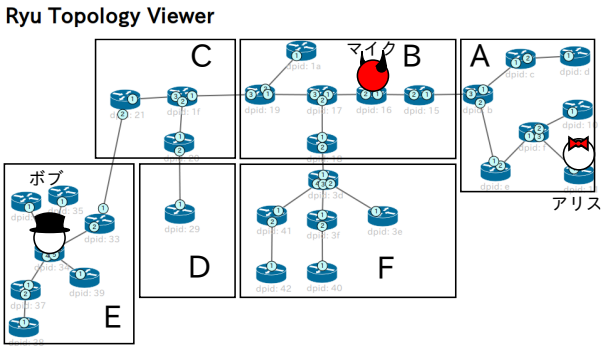


図 7　多数決プロトコルのネットワーク図.

表 5　実験環境の IP アドレス.

| 仮想の国 | サブネットマスク |
|---|---|
| A | 172.16.0.0/16 |
| B | 172.17.0.0/16 |
| C | 172.18.0.0/16 |
| D | 172.19.0.0/16 |
| E | 172.20.0.0/16 |
| F | 172.21.0.0/16 |

また，それぞれの国の IP アドレスの振り分けは表 5 の通りとし，各ルータの IP アドレスは第 3 オクテットに 1 から 7，第 4 オクテットに 1，サブネットマスクを 24 として，同一仮想国内でもセグメントを分ける．

### 4.6　評価と考察

提案手法で正しい公開鍵を選択できるか評価を行う．

図 7 のネットワークは表 4 の通り 30 台のルータが接続されており，ルータ 1 台に対し 2 台のコンピュータが接続されているので，合計で 60 台のコンピュータから構成されている．このとき，A 国にいるユーザ・アリスが E 国にいるユーザ・ボブの公開鍵を入手するため公開鍵要求パケットを送信する．アリスの鍵要求に対し参加者が送信する鍵応答パケットがアリスの元に集まるため，集まった公開鍵について以下の 2 種類の観点で実験を行う．

(1) A 国に比較的近い B 国，F 国の公開鍵がすべて偽物 (24/60，全体の 40%) の場合正しく多数決をとることができるか.

(2) アリスとボブ間の B 国，C 国，E 国の公開鍵がすべて偽物 (32/60, 全体の 50%以上) の場合，正しい鍵と偽物の鍵の割合が時間的にどのように変化するか.

この実験をそれぞれ 5 回行い，アリスが最初の鍵を受けとった時間から最後の鍵を受け取るまでの時間が 1 番短いものと 1 番長いものを除き，平均的と考えられる中間の 3 つの結果について考察する．

実験 (1) の場合である B 国，F 国の公開鍵がすべて偽物の場合の鍵の集まりかたを図 8，図 9，図 10 にそれぞれ示

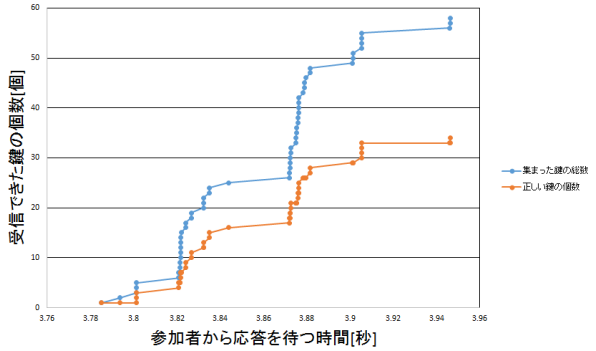


**図 8** B 国，F 国の鍵がすべて偽物のときの鍵要求パケットに対する返答と収集できた鍵数の関係（その 1）．

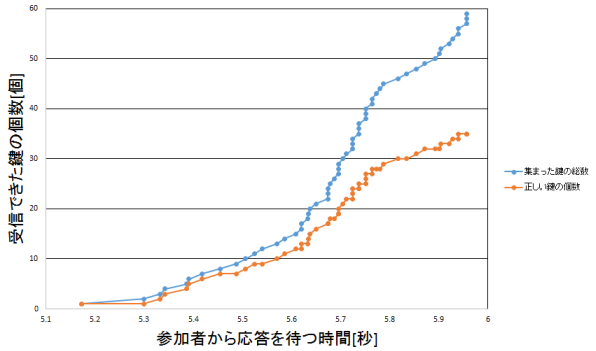


**図 9** B 国，F 国の鍵がすべて偽物のときの鍵要求パケットに対する返答と収集できた鍵数の関係（その 2）．

す．図 8 から図 10 のいずれについても，集まった鍵が 10 個以下のときは正しい鍵の割合が 33%から 100%と幅広く安定していないが，たとえば，図 8 の 3.82 秒以降のように収集できた鍵の総数が 11 個以上のとき，正しい鍵の数が 50%を下回ることはなく 60%前後で安定することが分かる．このことから，偽物の鍵が全体の 50%を下回っている場合は，公開鍵の収集時間を長く取って多数決を取ることにより正しい鍵を決定できることが分かる．実験の 3 つの場合のいずれにおいても，最初の鍵を受け取ってから遅くとも 1.18 秒目までに到着した鍵の数は 30 個程度であり，そのうち正しい鍵の数は少なくとも 18 個となっていることから，18/30 は 0.6 を超えており A 国に近い B 国，F 国の影響が小さくなったためと考えられる．

一方，ネットワークに占めるコンピュータの割合が 50%以上となる B 国，C 国，E 国の公開鍵が偽物の場合の鍵の集まりかたを図 11，図 12，図 13 にそれぞれ示す．この場合では，鍵が集まれば集まるほど偽物の鍵が増えるため収集できた鍵の総数が 20 個以上のときの正しい鍵の割合は 45%前後となった．しかし，図 11 を除く図 12 と図 13 の場合，収集できた鍵の総数が 11 個から 16 個のときに，正しい鍵の割合が 50%から 55%となっている．これは，鍵が集まり始めたときは，偽物の鍵が配布されていない A 国や F 国から正しい公開鍵を集めることができるため正しい鍵の割合が 50%を超えたと考えられる．これは多数決プロトコルの特徴で，通信相手との経路外からも鍵を集めることができるためである．

以上より，正しい鍵が過半数を超えている場合，全体の 1/6 以上の鍵を集めることができれば多数決により正しい公開鍵を決定することができると考えられる．また，偽物の鍵が全体の 50%以上を占める場合でも，鍵要求を行うユーザの周りに正しい鍵を持っている参加者が多ければ，鍵収集を比較的早いタイミングで打ち切って多数決を取ることにより正しい公開鍵を決定できる可能性がある．

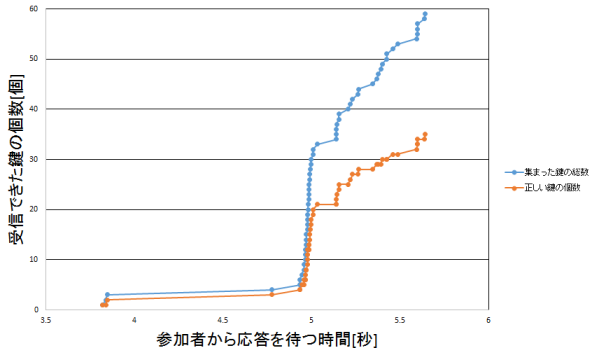


**図 10** B 国，F 国の鍵がすべて偽物のときの鍵要求パケットに対する返答と収集できた鍵数の関係（その 3）．

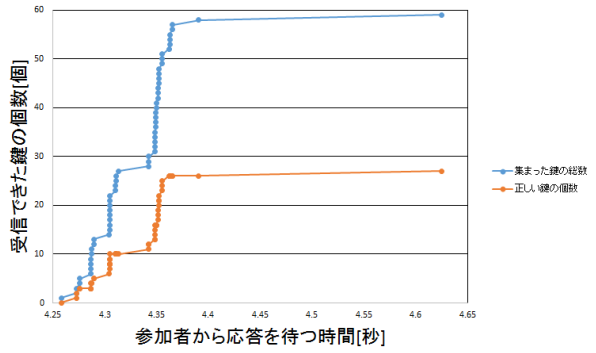


**図 11** B 国，C 国，E 国の鍵がすべて偽物のときの鍵要求パケットに対する返答と収集できた鍵数の関係（その 1）．

## 5. おわりに

本稿では，多数決に基づく公開鍵決定プロトコルを提案した．公開鍵決定プロトコルは認証局を用いず，ネットワークに参加しているユーザが互いに持ち合う公開鍵により公開鍵を決定する．実際にプロトコルのプロトタイプを仮想ネットワーク上で実装し，偽物の鍵が全体の 40%の場合は，全体の 1/6 以上の鍵を集めることで正しい公開鍵を決定できることを示した．また，偽物の鍵が全体の 50%を超える場合でも，鍵要求を行うユーザの周りに正しい鍵を持つ参加者が多ければ正しい公開鍵を決定できる可能性が

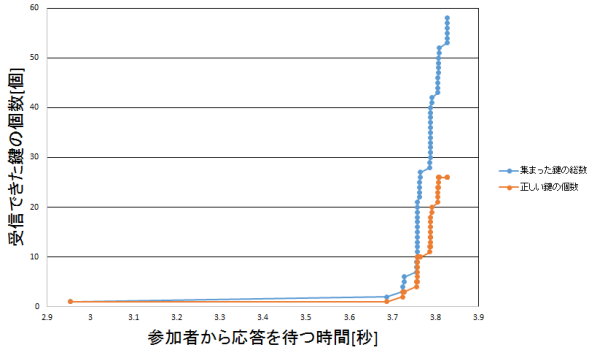


**図 12** B 国，C 国，E 国の鍵がすべて偽物のときの鍵要求パケットに対する返答と収集できた鍵数の関係（その 2）．

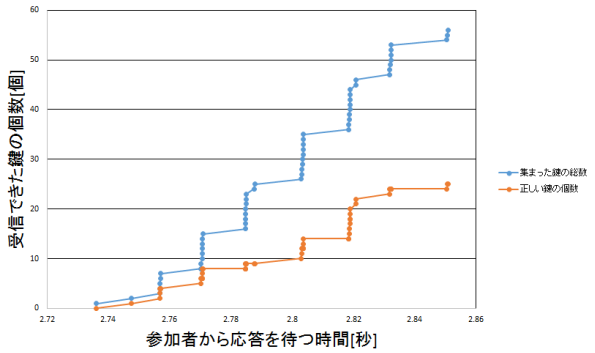


**図 13** B 国，C 国，E 国の鍵がすべて偽物のときの鍵要求パケットに対する返答と収集できた鍵数の関係（その 3）．

あることを示した．逆に言えば，鍵要求を行うユーザの周りの鍵の多くが偽物に置き換わっていた場合は，鍵収集時間をより長く取る必要があることを意味する．

今後の課題は全体から鍵を集めるのでなく，何らかの基準を用いて集める鍵の個数を減らし，偽物の鍵の影響をより少なくするような改善を行うことが挙げられる．

## 参考文献

[1] Freier, A. O., Karlton, P. and Kocher, P. C.: The Secure Sockets Layer (SSL) Protocol Version 3.0, Netscape Communications (online), available from ⟨https://www.ietf.org/rfc/rfc6101.txt⟩ (accessed 2015-12).

[2] Dierks, T.，Rescorla, E.，宮川寧夫訳：TLS Protocol Version 1.2, IPA (online), available from ⟨https://www.ipa.go.jp/security/rfc/RFC5246-00JA.html⟩ (accessed 2015-12).

[3] 結城 浩：暗号技術入門 - 秘密の国のアリス，SB クリエイティブ株式会社，新版 edition (2014).

[4] Red Hat, I.: POODLE: SSLv3.0 脆弱性 (CVE-2014-3566) - Red Hat Customer Portal, Red Hat, Inc.（オンライン），入手先 ⟨https://access.redhat.com/ja/node/1232403⟩（参照 2014-12）.

[5] Lenovo.: Superfish の脆弱性, Lenovo.（オンライン），入手先 ⟨https://support.lenovo.com/jp/ja/product_security/superfish⟩（参照 2015-08）.

[6] Postel, J.，srgia 訳：RFC792(INTERNET CONTROL MESASAGE PROTOCOL), (online), available from ⟨http://srgia.com/docs/rfc792j.html⟩ (accessed 2014-12).

[7] 佐々木良一，吉浦 裕，手塚 悟，三島久典：インターネット時代の情報セキュリティ-暗号と電子透かし-，共立出版株式会社 (2000).

[8] STINSON, D. R.，櫻井幸一監訳：暗号理論の基礎，共立出版株式会社 (1998).

[9] Foundation, O. N.: ONF Overview - Open Networking Foundation, Open Networking Foundation (online), available from ⟨https://www.opennetworking.org/ja/about-ja/onf-overview-ja⟩ (accessed 2015-01).

[10] Team, M.: Mininet: An Instant Virtual Network on your Laptop (or other PC) - Mininet, Mininet Team (online), available from ⟨http://mininet.org⟩ (accessed 2015-01).