

通信頻度情報を用いた鍵配送の効率化

松山 徹雄† 今村 翔‡ 岡本 栄司‡

† (株) 電通国際情報サービス
108-0075 東京都港区港南 2-17-1

‡ 筑波大学 システム情報工学研究所
305-8573 茨城県つくば市天王台 1-1-1

あらまし 誰もが容易にネットワークに接続できる環境を持ち、新しい形のサービスを利用することができるようになった反面、情報の漏洩・改ざん・なりすましなどの新たな脅威が生まれている。安全な通信のためには暗号技術を用いるが、鍵の管理にはリスクが多く、鍵生成や鍵配送においても、当事者同士ではなくサーバを利用することがある。しかし、サーバを利用する場合、利用者の増加に従ってサーバの負荷も増大していくという問題がある。本研究では、鍵配送を行うサーバの負荷を軽減することを目的として、過去の利用者間の通信頻度情報を利用し、サーバの鍵配送回数を減少させる木構造の構成方法について提案する。

Efficient Key Distribution System Using Communication Probability

Tetsuo Funayama† Sho Imamura‡ Eiji Okamoto‡

† Information Services International-Dentsu, Ltd.
Kohnan 2-17-1, Minato-ku, Tokyo, 108-0075 Japan

‡ Graduate School of Systems and Information Engineering, University of Tsukuba
Tennodai 1-1-1, Tsukuba, Ibaraki, 305-8573 Japan

Abstract A large organization has various working groups and they are sometimes overlapped. Conversation in a group has to be secret usually for people outside or even for people in the organization. Encryption is a technique for secrecy, but key has to be changed every session. This paper proposes an efficient key distribution system using symmetric key encryption algorithm.

1 はじめに

1.1 背景

近年、誰もが容易にネットワークに接続できる環境を持ち、新しい形のサービスを利用することができるようになった反面、情報の漏洩・改ざん・なりすましなどの新たな脅威が生まれている。これらの脅威に対抗するために、情報セキュリティは必要不可欠な技術となっている。不特定多数が利用できるネットワークで安全に通信を行うためには、情報を暗号する必要があるが、情報の暗号化のためには鍵を共有する必要がある。しかし、通信する相手が多くなればなるほど所持する鍵の数は増大し、効率的な鍵配送や安全な鍵管理は無視できない問題となる。そこで、通信対象を構造的に配置し、鍵配送の回数や所持鍵数を減らすことも情報セキュリティにおける課題の1つとなっている。

最も基本的な鍵管理方式として Burmester らの方式がある。利用者は全員が鍵配送センターというべき第三者から配送される1つの鍵を共有し、その鍵を用いて暗号化通信を行う。しかし、利用者の内1人でも参加・脱退があれば、共有していた鍵を無効にし、新たに共有する鍵を全員に配布する必要がある。利用者にとって鍵の所持数は少なく鍵管理の手

間は小さいが、鍵配送を行うサーバにとっては、利用者の入れ替え毎に大きな負荷がかかることになる。そこで、Wong ら [2] によって木構造を応用した鍵管理方式が提案された。利用者を木構造の葉に配置しノードに共有する鍵を配置することで、利用者とサーバの負荷をより均等に分散した方式である。木構造を用いた方式であっても、利用者の参加・脱退における鍵更新は避けられない。しかし、利用者が自由にグループを組んで通信できるということを考慮すると、サーバの負担を大きく減らすことが可能である。利用者が新たなグループを組んで通信する場合、サーバは通信用の鍵を生成し各グループメンバに送信するが、木構造を用いた方式であればその構造を利用して通信用の鍵を配布することができる。

1.2 目的

本研究では、利用者がグループで通信する際のサーバの負荷をより軽減することを目的とする。その手段として、Wong らの提案をもとに過去の通信頻度情報を用いて利用者を効率的に配置した木構造の作成方法を提案する。利用者の配置と木構造のノードの取り方で、サーバの鍵の配送回数を減らすことが可能となる。Wong らの提案では木構造は n 分木に対応しているが、本研究では2分木を対象に

研究を行った。決められたグループが存在しない限り、2分木の方が自由なグループで通信するという条件下で効果を発揮すると予測できる。

2 2分木を用いた方式

2.1 2分木モデル

ワーク鍵生成・鍵配送を行うサーバは鍵管理木を作成する。ここで、鍵管理木とは利用者の公開鍵を2分木の葉に、葉以外の節にはノード鍵を配置した木構造のことをいう。サーバは、鍵管理木に配置されるノード鍵とすべての利用者の公開鍵を所持し、各利用者は自分の公開鍵の位置する葉から根までの節にあるノード鍵と秘密鍵を所持するものとする。

また、利用者は同一の鍵管理木に属する相手であれば、自由な組み合わせでグループを作り、暗号化通信を行える環境となっている。図1の場合、利

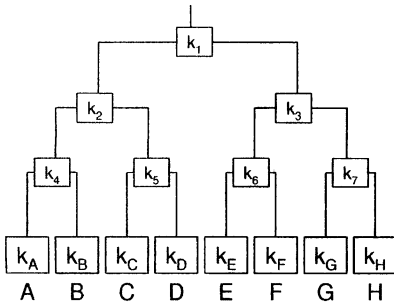


図1: 2分木モデル

用者Aは自分の秘密鍵 k_A とノード鍵 k_1, k_2, k_4 を所持していることになる。完全2分木に関わらず、利用者数 N とするとノード鍵の全個数は $N-1$ 個となり、利用者1人あたりが持つ平均ノード鍵数は $\log_2 N$ で表される。

2.2 鍵配送

鍵管理木に所属する利用者から暗号化通信を行いたいグループのリクエストがあった場合、サーバはワーク鍵を生成し、リクエストされたグループメンバに対してのみワーク鍵を配布しなくてはならない。しかし、各グループメンバに対してそれぞれの公開鍵で暗号化すると、サーバはグループメンバ数分の暗号化処理が必要となり負荷が大きくなる。そこで、リクエストされたグループの部分集合に含まれるノードを探し、そのノード鍵を用いてワーク鍵を暗号化し配布を行えば、サーバのワーク鍵の暗号化回数を減らすことが可能となる。

以下にグループワーク鍵配送の例を示す。
A, B, C で通信する場合

1. サーバはグループ(ABC)というリクエストを受け付ける。
2. ワーク鍵 K_{ABC} を作成する。
3. A, Bのみが共通に持つノード鍵 k_4 とCの公開鍵 k_C でそれぞれ暗号化し、 $E_{k_4}(K_{ABC}), E_{k_C}(K_{ABC})$ を計算する。
4. $E_{k_4}(K_{ABC})$ をA, Bに送信し、 $E_{k_C}(K_{ABC})$ をCに送信する。

他の利用者はノード鍵 k_4 も k_C も持たないので、 K_{ABC} を得ることができない。A, B, Cという3人グループで通信するが、サーバのワーク鍵暗号化回数は2回で済むことになる。

B, D, F, H で通信する場合

1. サーバはグループ(BDFH)というリクエストを受け付ける。
2. ワーク鍵 K_{BDFH} を作成する。
3. グループメンバそれぞれの公開鍵 k_B, k_D, k_F, k_H でそれぞれ暗号化し、 $E_{k_B}(K_{BDFH}), E_{k_D}(K_{BDFH}), E_{k_F}(K_{BDFH}), E_{k_H}(K_{BDFH})$ を計算する。
4. それぞれの暗号文をそれぞれのグループメンバに送信する。

B, D, F, Hの場合、グループメンバのみが持つ共通のノード鍵は存在しない。よって、グループメンバそれぞれの公開鍵で暗号化し、ワーク鍵を配布する。4人のグループで通信するが、サーバのワーク鍵暗号化回数は4回になってしまう。

つまり、鍵管理木のノードの利用が鍵配送の効率化を意味する。

2.3 効率化

図1では、鍵管理木を単純な完全2分木で表していたが、2分木モデルの更なる効率化のために新たな因子を加える。

2.3.1 通信頻度

通信頻度をグループが過去に通信した回数と定義する。通信頻度情報はグループメンバ数 ≥ 2 以上のすべての部分集合に対して与えられており、サーバが管理しているものとする。

通信頻度が多い利用者同士が同じノード以下に位置することができれば、サーバはノード鍵を利用できる割合が高くなり、効率化が期待できる。

利用者数 N の場合、すべての通信頻度情報の個数は

$$\sum_{r=2}^N {}^N C_r \quad (1)$$

で表される。これは N の冪集合から利用者単体・空集合を引いた集合であるため、通信頻度情報の個

数は $O(2^N)$ のオーダーで表される。通信頻度情報の個数は N に対し指数関数的に増加するが、実際のシステムにおいては、サーバは必ずしもすべての通信頻度情報を保持しておく必要はない。

2.3.2 2分木の構造

鍵管理木は完全2分木である必要はない。通信頻度が多い利用者ほど、鍵管理木の深い階層に位置し多くのノード鍵を所持することにより、鍵配送回数を減らすことが可能である。

また、新規利用者の参加というイベントに関しても柔軟に対応できる。完全2分木の場合、新たな利用者を追加する位置は決定しているが、完全2分木でない場合、木構造の自由な位置に新たな利用者を配置できる。通常、新規参加者は通信頻度は0であると仮定される。新規参加者を含むすべての通信頻度が0であるならば、新規参加者は鍵管理木のルートに新たに追加すればよい。また、過去の通信履歴を考慮する場合や特別なグループを構成したい場合などの特性にも対応可能である。

2.4 全数探索

利用者の2分木全パターンについての配置を計算することによって、最適な鍵管理木を求めることができる。ここでの最適な鍵管理木とは、過去の通信頻度回数だけ通信する場合に、サーバの鍵配送回数が最も少ない木構造のことをいう。

2分木の全パターン数は、

$$\prod_{r=2}^N {}_r C_2 \quad (2)$$

で表すことができる。2分木より、新たなノードは2つの葉、もしくはノードから構成される。初期状態の場合、利用者は N 人であるから、最初のノードの組み合わせ数は ${}_N C_2$ 個となる。1つのノードを組むと、ノードが1つ増え、葉が2つ減るので、新たにノードを組むことできる葉、ノードの個数は全体として $N-1$ 個となる。よって、次のノードの組み合わせ数は ${}_{N-1} C_2$ で表される。このようにしてノードを決定していくと、ルートの組み方は ${}_2 C_2$ となり、2分木の全パターン数は上式で表すことができる。確実に最適な鍵管理木を求めることができるが、計算時間が利用者数 N の指数時間となり、 N が大きくなると計算量の観点から現実的ではない。そこで、第4章では計算量を減らした提案方式について説明し、全数探索との性能比較を行う。

3 提案方式

本章では、鍵管理木の全数探索をせずに鍵配送回数をより減らすことができるような木構造作成方式の提案をする。また、提案方式について実装し、全数探索の場合との計算量・実行時間・鍵配送回数の比率について性能評価を行い、結果を示す。

3.1 概要

基本的な方針としては、作成する鍵管理木のノードが通信頻度の多いグループとなるようにすることである。しかし、ノードがグループの部分集合に入っていればそのノードを利用できるので、全体として鍵配送回数を減少させることができるノードを選択する必要がある。そのため、あるグループの通信頻度を他のグループに反映させることで、より効率のよい鍵管理木を作成する。

3.2 ヒューリスティクス

利用者数を N 、グループのメンバ数を m で表すこととし、以下に提案するヒューリスティクスについて述べる。ヒューリスティクスでは、ボトムアップで鍵管理木を作成する。はじめに、ヒューリスティクスのフローチャートを図2に示す。

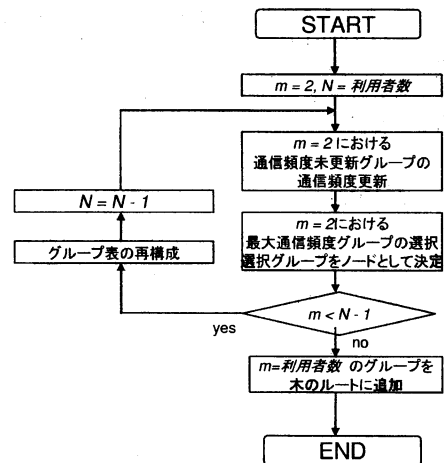


図2: ヒューリスティクスのフローチャート

フローチャートの詳細は以下の通りである。

1. $m=3$ から $m=N$ までのすべてのグループについて、 $m=2$ のグループを部分集合として含むか調べる。もし $m=2$ のグループを含む場合、調べているグループの通信頻度 $\times$ 定数 C を $m=2$ のグループに加算する。
2. $m=2$ のグループにおいて、加算後の最大通信頻度のノードを決定ノードとして選択する。
3. 2. によって選択されたノードから、以降取り得ないグループを削除する。
4. 2. によって選択されたノードを1つのメンバとみなし、 $m=3$ 以上のグループを下位に移動する。
5. 1. に戻る。

表 1: 通信頻度の例

2		3		4		5	
AB	3	ABC	0	ABCD	6	ABCDE	25
AC	10	ABD	5	ABCE	2		
AD	0	ABE	30	ABDE	5		
AE	3	ACD	0	ACDE	9		
BC	5	ACE	0	BCDE	25		
BD	20	ADE	8				
BE	4	BCD	10				
CD	2	BCE	0				
CE	5	BDE	0				
DE	0	CDE	12				

1. での定数 C とは木構造調整のための値である。あるグループを意図的に同じノード以下に配置したい場合、 C を大きい値を設定すればよい。以下、すべて $C = 1$ とする。2. で鍵管理木のノードを選択する。 $N - 2$ 回のノード選択を行い、最後にルートを決定する。鍵管理木が 2 分木であることから、新たなノードは常に 2 つのノードから選択されることになる。そこで、4. の処理を行い、実質 $m = 2$ のグループから毎回選択させている。

3.3 具体例

ヒューリスティクスの具体例を示す。利用者数 $N = 5$ とし、各グループの通信頻度はランダムに決定して表 1 に示す。

1. 通信頻度の加算

AB に加算するグループは {ABC, ABD, ABE, ABCD, ABCE, ABDE, ABCDE}

加算後: $AB' = AB + ABC + ABD + ABE + ABCD + ABCE + ABDE + ABCDE = 76$

AC に加算するグループは {ABC, ACD, ACE, ABCD, ABCE, ACDE, ABCDE}

加算後: $AC' = AC + ABC + ACD + ACE + ABCD + ABCE + ACDE + ABCDE = 52$

以下、すべての $m = 2$ のグループにおいて、同様に行くと表 2 のようになる。

表 2: 加算後

2		3		4		5	
AB'	76	ABC	0	ABCD	6	ABCDE	25
AC'	52	ABD	5	ABCE	2		
AD'	58	ABE	30	ABDE	5		
AE'	82	ACD	0	ACDE	9		
BC'	73	ACE	0	BCDE	25		
BD'	96	ADE	8				
BE'	91	BCD	10				
CD'	89	BCE	0				
CE'	78	BDE	0				
DE'	84	CDE	12				

2. $m = 2$ における最大頻度ノードの選択

表 2 より、最大頻度ノードは BD である。よって、BD を鍵管理木のノードとして決定する。

3. 表から可能性のないグループを削除する。

2. では BD を選択したので、BD をすべて含むもの、もしくは、BD を全く含まないものを残し、それ以外を表より削除する。削除後は表 3 に示す。

表 3: 削除後

2		3		4		5	
AC'	52	ABD	5	ABCD	6	ABCDE	25
AE'	82	ACE	0	ABDE	5		
		BCD	10	BCDE	25		
CE'	78	BDE	0				

4. 選択したノードを元に表を再構成する。

選択したノード: BD より、BD を含むグループを一人の利用者とみなし、下位グループに移動する。

BD を含むグループ: {ABD, BCD, BDE, ABCD, ABDE, BCDE, ABCDE}

再構成後の結果は表 4 のようになる。

表 4: 再構成後

2		3		4		5	
AC'	52	ACE	0	ABCDE	25		
AE'	82	ABCD	6				
CE'	78	ABDE	5				
ABD	5	BCDE	25				
BCD	10						
BDE	0						

5. 1. に戻る。

ただし、一度通信頻度の加算が行われたグループに対しては通信頻度の加算は行わない。

以上を繰り返し、鍵管理木が作成されるまでの表と木の状態を図 3 に示す。

4 性能評価

4.1 ヒューリスティクス考察

ヒューリスティクスについて考察する。作成した鍵管理木と最適な鍵管理木における鍵配送回数の差に精度という語句を用いる。

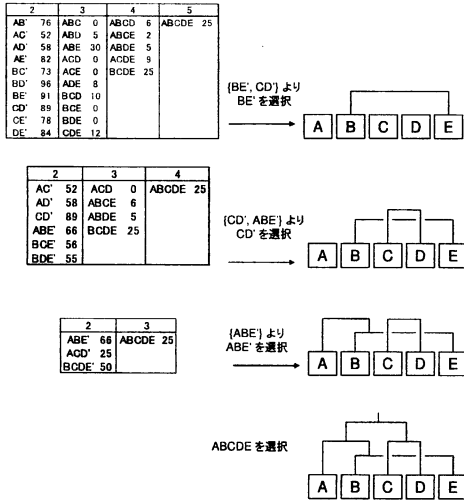


図 3: ヒューリスティクスの例

- 精度：常に $m = 2$ のグループからノードを選択するが、上位グループから通信頻度情報を加算するため、更新後はどうしても下位グループの通信頻度情報が大きくなる。そのため、メンバ数の少ないグループを選択する傾向が生じる。すると、バランスの良い木構造が作成される。詳細は後述するが、利用者入れ替えの際の負荷のかかり方に影響を与える。その傾向を嫌う場合は定数 C で調整することができる。また、他のグループと比較して非常に大きい通信頻度のグループが存在した場合でも、そのグループをノードとして選択できる。
- 計算量：ノードを選択するたびに $m = 2$ のグループの通信頻度情報を更新する。すべての $m = 2$ のグループに対し、 $m \geq 3$ のグループの通信頻度を加算する処理は利用者数に依存する。全グループの通信頻度情報の個数は利用者 N に対し、指数関数的に増加していくが、1つノードを選択した時点で、候補として対象外のグループは削除できるため、ノードを選択していくに従い更新に必要な計算を減らすことが可能である。1つノードを選択することは、利用者が1人少ない状態になることを意味する。

4.2 実行結果

ヒューリスティクスについて実装し、全数探索と比較した結果を示す。実行は4人から8人までの利用者数について、通信頻度情報に乱数を用いて100回実行する。実行環境を表5に示す。

表 5: 実行環境

CPU	Intel PentiumM 1.2GHz
RAM サイズ	768MB
OS	Windows XP Professional
開発言語 (環境)	Java(JDK 1.5.0.03)

表6には精度を比で表したものと100回の実行の中でヒューリスティクスが何度最適であったかを記載している。また、表7には鍵管理木が作成されるまでの1回当たりの実行時間を載せる。

※実行時間の単位は秒である。

※－は計測不可能だったことを表し、×はメモリ不足で実行できなかったことを意味する。

表 6: 精度

利用人数	4	5	6	7	8
提案 / 最適最小比	1.0	1.0	1.0	1.0	1.0
提案 / 最適最大比	1.0771	1.0792	1.0753	1.0752	1.0422
提案 / 最適平均比	1.0156	1.0114	1.0333	1.045	1.0316
提案 / 最悪平均比	0.7882	0.8297	0.7962	0.8334	0.791
提案手法最適回数	66	36	9	2	1

表 7: 実行時間

利用人数	4	5	6	7	8	9	10
全数探索	—	0.008	0.1	4.5	279	22654	×
提案	—	—	0.008	0.01	0.017	0.03	0.05
全数探索 / 提案比	—	—	12.5	450	16411	755133	×

精度・実行時間比較のグラフを図4に示す。

4.3 実行結果考察

実行結果について考察する。まず、表6では、利用者数の増加に対して、提案手法の性能の変化が非常に小さいということが読み取れる。利用者数4人から8人に対し、提案 / 最適平均比は5%以内に収まっており、さらに、人数の増加と精度の悪化に比例関係も見られない。よって、利用者の増加に対してもある程度の精度を保持できると予想される。

提案 / 最悪平均比が、利用者数が偶数と奇数の場合で4%ほど差が出ていることが読み取れる。利用者数が偶数の場合、奇数の場合と比べて葉を2つ合わせて組むノードが1つ増えるため、それが全体の鍵配送回数削減に影響を与えていると推測できる。つまり、ここでは通信頻度情報は乱数で与えているため、各グループの通信頻度に相関がなく、このような場合、鍵管理木をバランスよく構成した方が効率が良いと考えることができる。ここでいう相関と

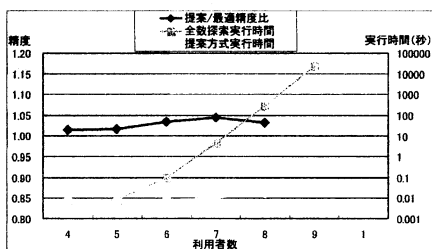


図 4: 精度・実行時間比較

は、例えば、グループ AB で多く通信が行われていれば、グループ ABC やグループ ABD においても通信が行われている可能性が高いということを意味する。

上述したように、通信頻度情報を乱数で与えたため、最適・最悪の精度に 2 割程度しか差が出ていない。しかし、実際は利用者間に相関があると考えられる。もし、利用者間にある程度の相関があれば、ノード鍵を利用できる可能性が高くなり、最適・最悪の差は大きくなることが予測される。また、ヒューリスティクスでの精度向上も期待できる。

表 7 は、全数探索と提案手法の実行時間の比較を示した表である。グループメンバー数 9 人の段階で、全数探索は 6 時間以上の実行時間となっており、現実的ではないことが分かる。全数探索では、木構造の全パターン数に比例して、指数関数的に実行時間が増大しているのに対し、提案手法では、グループメンバーに対する実行時間の増加は微小であり、違いを図 4 から読み取ることができる。現実を利用する場合、利用者数はさらに多く、全数探索では対応できないことは明らかである。以上から、提案手法において、利用者数増加に対する精度低下の傾向は見られず、現実的に利用可能であることが確認できた。

5 効率化の向上について

5.1 提案ヒューリスティクスの改良

4.2 節で提案したヒューリスティクスをヒューリスティクス 1 と呼ぶことにする。ここでは、ヒューリスティクス 2 としてヒューリスティクス 1 を改良した手法について説明する。

ヒューリスティクス 1 では、更新後の通信頻度表から最大通信頻度のグループをノードとして選択するが、ヒューリスティクス 2 では、ノードの選択の仕方を変更する。それ以外はヒューリスティクス 1 と同様である。手順を以下に示す。

1. $m = 3$ から $m = N$ までのすべてのグループについて、 $m = 2$ のグループを部分集合として含むか調べる。もし $m = 2$ のグループを含む場合、調べているグループの通信頻度 $\times$ 定数 C を $m = 2$ のグループに加算する。

2. $m = 2$ のグループにおいて、木構造として考えられるグループの集合を比較し、その中で合計通信頻度が最大となる集合を選択する。そして選択した集合の要素の内、最大通信頻度のグループをノードとして決定する。

3. 2. によって選択されたノードから、以降取り得ないグループを削除する。

4. 2. によって選択されたノードを 1 つのメンバーとみなし、 $m = 3$ 以上のグループを下位に移動する。

5. 1. に戻る。

5.2 ヒューリスティクス 2 の考察

- 精度：ヒューリスティクス 1 では、更新後の通信頻度情報から最大通信頻度のグループを選択するため、視野の狭い選択とすることができ。ヒューリスティクス 2 では、選択する際にその他の通信頻度を考慮してその中で最大通信頻度のグループを選択することになる。そのため、全体として鍵配送回数を減らせるノードを選ぶことができる可能性が高い。しかし、すべての場合においてヒューリスティクス 2 の方がヒューリスティクス 1 より精度が高いというわけではない。

- 計算量：ヒューリスティクス 1 に加えて、通信頻度の合計が最も高いグループ集合を選択する必要がある。グループ集合の要素数は $\lfloor N/2 \rfloor$ であり、同じ集合内で利用者の重複がないようにグループ集合の要素を選択する。グループ集合の要素数 K は、

$$K(2) = 1$$

$$\text{if}(N = \text{odd}) \quad K(N) = K(N-2) \times N$$

$$\text{else} \quad K(N) = K(N-1)$$

として計算することができる。これもノードを選択するごとに $N = N-1$ とすることができる。

6 おわりに

本研究では、利用者がグループで通信する際の鍵配送サーバの負荷を軽減することを目的に、過去の通信頻度情報を用いて鍵配送の効率を高める手法を提案した。提案方式の鍵管理木は全数探索と比較して、鍵配送回数を数パーセントの範囲内に抑えることができ、鍵管理木作成時間の大幅な短縮に成功した。これにより、利用者数の増加に耐える鍵配送システムを構築することが可能となる。

参考文献

- [1] 福島和英、清本晋作、田中俊昭、コンテンツ配信のためのグループ化による効率的な鍵管理方式、SCIS2005, pp.271-276, 2005
- [2] Chung Kei Wong, Mohamed G. Gouda and Simon S. Lam, Secure group communications using key graphs, ACM SIGCOMM'98 conference on Applications, technologies, architectures and protocols for computer communications, pp.68-79, 1998.