マルチメディア通信と分散処理　106－9
コンピュータセキュリティ　16－9
（２００２．２．１４）

階層分散CAシステムによるネットワークの効率的管理

山渕 深司[†]　　近山 隆[‡]
[†‡] 東京大学新領域創成科学研究科基盤情報学専攻

CA システムにおいて，攻撃や故障に対しよりロバストにすることを目的として，複数のサーバに CA 機能を分散させる「分散 CA」が提案されているが，この方式はネットワーク負荷などの点で問題がある．これを改善するため，通信においてローカリティが生かされるようにサーバのグルーピングによって CA 機能を階層化した「階層分散 CA」を新たに提案する．この階層分散 CA の実装として DSA 署名に基いたシステムを設計し，通信効率の改善を定量的に確認するとともに，従来の分散 CA と同等のセキュリティを確保するためにはパラメタをどのように設定すべきかについて考察する．

Efficient Network Management with Hierarchical Distributed CA

Shinji YAMABUCHI[†]　　Takashi CHIKAYAMA[‡]
[†‡]Dept. of Frontier Informatics, the University of Tokyo

To improve robustness of CA systems, “distributed CA” has been proposed in which several servers share the certification function. However, this scheme is inefficient in communication between servers. “Hierarchical distributed CA” we propose can improve this problem by grouping servers and hierarchical certification to exploit locality. We design a hierarchical distributed CA system based on DSA, and quantitatively show improvements on communication efficiency. Parameter setting required to ensure the same robustness level as non-hierarchical distributed CA is also given.

1　序論

ネットワーク上で公開鍵暗号方式を用いる場合，なりすましを防止するため一般的にはCA(Certification Authority) に公開鍵証明書を発行してもらう[1]．ただし，CA は完全に信頼でき，またいつでも利用可能であるという前提が必要となる．なぜなら，もし CA がクラックされ CA 秘密鍵が漏洩してしまうと，その CA が発行した全証明書が無意味になってしまい，また CA の故障・電源断で証明書が発行できなくなると，ネットワークに新規に参加した端末が証明書を持てず，セキュア通信を行うことができないためである．このように，ネットワークでの安全な通信のためには，CA 端末の安定運用および秘密鍵の保管に細心の注意を払わなければならない．

[1]実際の CA では，公開鍵とその所有者の組み合わせが正しいかどうかの審査も行うが，本稿における「CA」の意味は，与えられた証明書本文に対し CA 秘密鍵によって電子署名を生成・発行するという機能のみに限定したものである．

これに対し，1 台の計算機ではなく複数の計算機で分散的に処理を行い，それらの協調により 1 台の CA と同等の機能を実現する方式が分散 CA である．この方式だと，一部の計算機に上述のような事態が発生しても，残りの計算機によって CA 機能自体は安全かつ継続的に運用でき，非常にロバストなシステムとなる．また，モバイルアドホックネットワーク等の Pure Peer to Peer ネットワークでは，信頼できる中央サーバが存在せず，またどのピアもネットワークから離脱しうることも考慮すると，単独の CA 端末ではなく，複数のピアが分散協調する分散 CA 方式をとる必要がある [9].

しかし，従来の分散 CA 方式は通信効率の点などで問題を抱えているため，これらを改善する新しいシステム「階層分散 CA」を提案した [1]．本稿では，DSA 署名に基いた階層分散 CA システムを実際に設計し，効率およびセキュリティについて，さらに詳細な検討を加える．

2 分散 CA

2.1 分散 CA の構成

本節では，従来から提案されている分散 CA の構成および通信プロトコルについて解説する．

分散 CA において CA 機能を分担する端末をサーバと呼ぶ．サーバはネットワーク内に n 台あり，そのうち t 台以上が協調すればはじめて CA 相当の機能を実現できる．つまり，$t-1$ 台以下がクラックされても，また $n-t$ 台以下が故障・電源断・ネットワーク離脱などによりサーバとして機能できなくなっても，全体としての CA 機能の信頼性・可用性には影響を及ぼさない．

CA 秘密鍵はどこにも物理的に存在せず，誰も知り得ない．各サーバが持つ値は，CA 秘密鍵を (t,n) 閾値秘密分散 [2] により分散させた秘密シェア[2]の 1 つであり，各サーバはこの秘密シェアにより「部分署名」を生成し，それらを t 以上集めることで，CA 署名と同等の「最終署名」を生成できる．一方，CA 公開鍵はネットワーク内の全端末に公開されるため，（最終）署名の検証は各端末が単独で行える．

具体的なプロトコルはシステムの暗号体系により異なってくる．以下では，[3] による変形 DSA 署名の場合について説明する（RSA 署名でも可能だが，DSA より複雑な手続きが必要となる [4][5]）．

2.2 分散鍵生成

各サーバが CA 秘密鍵のシェアを得る方法としては，信頼できるディーラーが CA 秘密鍵を生成し，閾値秘密分散法で分散させた値をそれぞれサーバに配る方法と，サーバの分散協調により生成する方法がある．前者は簡単だが，システムを構築する環境によっては実現不可能な場合がある．後者は以下のようなプロトコルとなる．なお，各サーバはあらかじめ他のサーバの公開鍵を知っており，サーバ間のセキュア通信路が常に確保されていること，また各サーバが相異なる ID を持っていることを仮定する．

step1 ある 1 台のサーバが p,q,g を生成し，他サーバにブロードキャストする．受け取ったサーバではそれらが正当なものか検証する．

step2 サーバ i は乱数 $r_i(0<r_i<q)$ を生成し，それを (t,n) 閾値分散させたサブシェア $s_{i,j}(1\leq j\leq n)$ を対応サーバ j にセキュアに送信する．

step3 サーバ j は集まったサブシェアの総和 $(\bmod q)$ をとり，それを CA 秘密鍵 x の秘密シェア x_j とする．また $y_j=g^{x_j} \bmod p$ を公開シェアとして全サーバにブロードキャストする．

step4 各サーバは集まった公開シェアの「積合成」[3]を行い，CA 公開鍵 y を計算する．

なお，サーバが送信したサブシェアが確かにそのサーバが生成した乱数の分散であること，公開シェアが確かに g^{x_j} であることは，VSS(Verifiable Secret Sharing)[6] を用いれば証明できる．

2.3 分散署名

署名の分散生成は以下のような手順となる．

step1 秘密鍵の分散生成の step2 以降と全く同様の手続きを行う．各サーバの秘密シェアを署名用の乱数 k_i，公開シェアの積合成の q による剰余を 1 つ目の最終署名 a とする．

step2 各サーバが部分署名を

$$b_i = x_i \cdot a + k_i \cdot h(M) \bmod q$$

として生成し，他のサーバにブロードキャストする．ただし $h(M)$ はハッシュ関数によるメッセージのダイジェストである．

step3 集まった部分署名を和合成することで 2 つ目の最終署名 b が得られる．a と b を CA 署名として発行する．

step4 署名の検証は

$$g^{b/h(M)} y^{-a/h(M)} \bmod p \bmod q = a$$

を判定することで行う．

[2] (t,n) 閾値秘密分散に基く x の秘密シェア $s_i(1\leq i\leq n)$ は $s_i = x + \sum_{j=1}^{t-1} f_j \cdot \mathrm{ID}_i{}^j \bmod q$ となる（f_j は乱数）．
秘密シェアが t 個以上集まれば，それらを「和合成」することで，x を $x = \sum_{i=1}^{t} w_i \cdot s_i \bmod q$ （ただし $w_i = \prod_{j=1,j\neq i}^{t} \frac{\mathrm{ID}_j}{\mathrm{ID}_j - \mathrm{ID}_i} \bmod q$ ）と計算できる．

[3] 計算式は $y = \prod_{i=1}^{t} {y_i}^{w_i} \bmod p$ である．w_i は脚注 1 参照．

2.4 シェア・リフレッシュ

(t, n) 閾値秘密分散のもとでは，t 台以上のサーバで秘密シェアが漏洩すると，CA 機能を乗っ取られることになる．攻撃者は長い時間をかけることで t 個以上の秘密シェアを得ることができるかもしれない．そこで，一定時間ごとにシェアを更新する (ただし，CA 秘密鍵は変更しない) ことにより，安全性を高める手法が提案されている [7].

これは，各サーバが値「0」を (t, n) 分散させたサブシェアを対応サーバに送り，受信側で自分の秘密シェアにそれらのサブシェアの総和を加えたものを新しい秘密シェアとすることで実現される．

3 階層分散 CA

3.1 フラット分散 CA の問題

本節では，従来の分散 CA の問題点を改善するため筆者らが新たに提案した階層分散 CA の構成および通信プロトコルについて解説する．

FDCA（従来の分散 CA を便宜上フラット分散 CA と呼び，こう略す）の最大の問題点は，多くの場面で全サーバ対全サーバの通信が発生することである．もし全てのサーバがトポロジ的に近い場所にあれば影響はあまりないが，ネットワーク中に分散しているような構成の場合はネットワーク全体の通信帯域を圧迫することになる．システムの安全性を上げるには，サーバ数を増やすかリフレッシュ間隔を短くするかのどちらかであるが，どちらの場合でも全サーバ対全サーバの通信量を増大させることになる．

この問題は，FDCA でローカリティが全く考慮されていないために起こる．そこで，ネットワークトポロジ的に近いサーバ同士でグループを構成し，多くの通信がグループ内で行われ，グループ間通信ができるだけ少なくなるよう試みたのが筆者らの提案した階層分散 CA(HDCA) システムである．

3.2 階層分散 CA の構成

HDCA は簡単に言えば，一般的な階層 CA における各 CA の機能を分散させたものである．階層 CA 同様，各端末はあるグループ CA にその端末の公開鍵証明書を発行してもらい．各グループ CA はルート CA にそのグループ CA の公開鍵証明書を発行してもらう．ここで，グループ CA の機能はそのグループ内のサーバによって，ルート CA の機能は全サーバによって分担される（図 1）．ルート CA 秘密鍵はまず各グループに分散され，それらが各グループ内のサーバにさらに分散されている構造になっている．

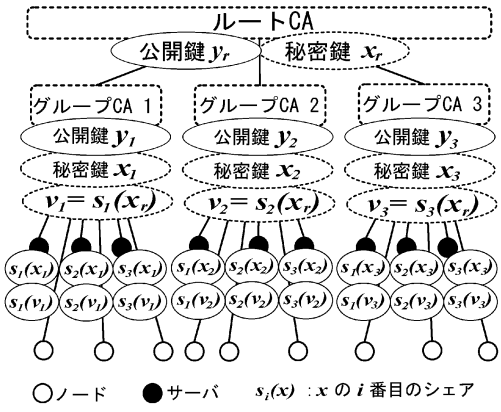

図 1: 階層分散 CA の秘密鍵分散

3.3 各種プロトコル

以下，DSA でのルート CA 関連各種プロトコルを述べる（グループ CA については FDCA と同様）．ルート CA の各グループへの分散スキームを (t_1, n_1)，各グループ内の分散スキームが全て (t_2, n_2) であると仮定する．まず鍵生成は次のようになる．

step1 p, q, g の生成・告知・検証.

step2 グループ i のサーバ j は乱数 $r_{i,j} (\mathrm{mod} q)$ を生成し，それを (t_1, n_1) 閾値分散させたサブシェア $s_{i,j,k} (1 \leq k \leq n_1)$ を作る．この各々をさらにグループ k の分散スキームで分散させたサブシェア $s'_{i,j,k,l} (1 \leq l \leq n_2)$ を作る.

step3 $s'_{i,j,k,l}$ をグループ内で交換する．この時，j と l 間にランダムに 1 対 1 関係を定め，サーバ j には $s'_{i,*,*,l}$ のみが送られることとする．受信した $s'_{i,j,k,l}$ を j について総和 $(\mathrm{mod} q)$ を取った $s''_{i,k,l}$ をグループ k のサーバ l にセキュアに送信する.

step4 グループ k のサーバ l は受信した $s''_{i,k,l}$ の総和 $(\mathrm{mod} q)$ をルート CA 秘密鍵の秘密シェア $x_{k,l}$ とする．さらに $g^{x_{k,l}} \bmod p$ を公開シェアとしてグループ内にマルチキャストする．

step5 グループ内の各サーバの公開シェアを積合成し，それをグループ CA の公開シェアとして他グループにブロードキャストする．

step6 各グループ CA の公開シェアを積合成したものをルート CA 公開鍵 y とする．

ルート CA による署名を生成する手順は，署名用乱数および 1 つ目の最終署名 a については，やはり鍵生成と同じ手順を踏み，最終的な積合成値の q による剰余を a とする．2 つ目の最終署名 b は，各サーバが自分の持つ CA 秘密鍵の秘密シェアおよび署名用乱数のシェアにより 2 次部分署名を生成してグループ内にマルチキャストし，和合成することでグループごとの 1 次部分署名が得られる．それらをブロードキャストし，和合成すれば b が得られる．

シェア・リフレッシュの手順についても，FDCA から HDCA への拡張は，鍵生成法と同様にできる．

各グループ内のサーバ数が異なる場合については，グループ間のサーバの対応関係が 1 対 1 でなくなり，1 台のサーバが複数のサーバに送るサブシェアを持っている，あるいは逆に複数のサーバから 1 台のサーバにサブシェアが送られてくる場合が出てくる．この時もプロトコル自体が破綻することはないが，セキュリティ上の問題が出てくる（後述）．

4 階層分散 CA の分析と評価

4.1 通信効率

全サーバ数を n とした時，(t, n) 閾値分散に基いた FDCA と，$(t^{\frac{1}{2}}, n^{\frac{1}{2}})$ 閾値分散に基いた 2 階層 HDCA を，通信効率の面から定量的に比較分析する．簡単のため以下の仮定を置く．

- HDCA では，各グループ内のサーバは全て $n^{\frac{1}{2}}$ 台，また閾値も全て $t^{\frac{1}{2}}$ とする．
- 全ての CA で，p および q のビット数はそれぞれ l_p, l_q と等しい．
- 不正者の攻撃がない（検証失敗によるリトライ等はない）環境である．
- k 台へのマルチキャストにおけるセッション数は k と数える．

表 1: 分散 CA と階層分散 CA の通信効率

	FDCA	HDCA(2 階層)
セッション総数	$\mathcal{O}(n^3)$	$\mathcal{O}(n^2)$
通信総ビット数	$\mathcal{O}(tn^3 l_p)$	$\mathcal{O}(t^{\frac{1}{2}} n^{\frac{5}{2}} l_p)$
内 グループ間		$\mathcal{O}(n^2 l_q + t^{\frac{1}{2}} n^{\frac{3}{2}} l_p)$

表 1 の上段は，n 台の端末および全グループ CA がそれぞれ上位 CA に証明書を発行してもらうまでにネットワーク内で張られる通信セッションの総数である．FDCA に比べ HDCA ではセッション数が大幅に削減されることが分かるが，これは端末 1 台に対し，証明書発行に直接かかわるサーバが n 台から $n^{\frac{1}{2}}$ 台になることが主要因である．全サーバが協調するのは，ルート CA の鍵生成時および各グループ CA に対する署名生成時のみである．

下段は，各セッションで何ビットの情報が転送されるかを見て，全体としての通信量を算出した値である．セッション当たりの平均ビット数では FDCA が $\mathcal{O}(tl_p)$，HDCA が $\mathcal{O}(t^{\frac{1}{2}} n^{\frac{1}{2}} l_p)$ と HDCA の方が大きくなる．これは階層化に伴い VSS の検証用情報がより多く必要になるためである．しかしセッション数の減少効果が勝るため，全体の通信量としては HDCA の方が少なくなる．

また HDCA でのグループ間通信をグループ内通信と比較すると，通信セッション数では同一オーダーであるにもかかわらず，通信量ではずっと少なくなる．これはグループ間でサブシェアの交換をする場合の通信方法を工夫しているためである．具体的には，グループ間で公開情報をマルチキャストする際，グループをまたぐセッションは 1 回しか張らず，残りはグループ内でのマルチキャストとする．この情報量はグループ間でやりとりする秘密サブシェアの情報量よりずっと多いため，グループ間通信の平均情報量削減に大きく貢献する．

グループ間通信を少なくすることによる効果についても見積もってみる．一辺 $d = n^{\frac{1}{2}}$ の正方形のエリアに n 台のサーバが一様に分布しているモデルを考える．このうち任意の 2 台の間のマンハッタン距

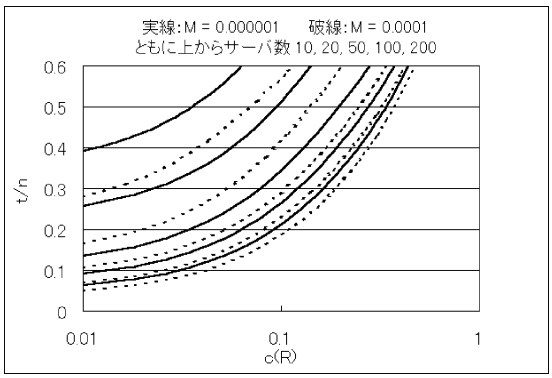

図 2: 閾値法のパラメタ設定

離の平均は $\frac{2}{3}d$ なので，グループ間通信の平均距離は $\mathcal{O}(n^{\frac{1}{2}})$ となる．一方，1 グループを，元のエリアを $n^{\frac{1}{2}}$ 分割した一辺 $d^{\frac{1}{2}}$ の正方形のエリアとすれば，グループ内通信の平均距離は $\mathcal{O}(n^{\frac{1}{4}})$ となる．これらを情報量に乗じ，ネットワーク帯域占有の度合いを求めても，グループ内通信が支配的であることは変わらないことから，ローカリティが生かされていることが実際に確認できる．

4.2　セキュリティ

HDCA は FDCA に比べ，より少ない台数でのシェア漏洩でシステムが乗っ取られうる．例えば閾値が一律「半分」の時，バランスした 2 階層分散 CA においても最悪の場合半数のグループのそれぞれ半数のサーバ，つまり全サーバの 1/4 で秘密シェアが漏洩するとルート CA 機能が乗っ取られてしまう．この脆弱性[4]に対処するため，階層ごとにシェア・リフレッシュ間隔および閾値を調整する必要が生じる．以下，これらのパラメタをどのような基準で調整すればよいかについて検討する．

個々のサーバにおいて秘密シェアが漏洩する確率は全て等しく，かつ独立であると仮定すると，次のような問題として定式化できる.「各々のサーバでシェアが漏洩する確率は，リフレッシュされてからの時間 T に対する単調増加関数 $c(T)$ で表される．ここで上位 CA 機能が乗っ取られる確率をある十分小さな値 M 以下に抑えるための，閾値 t とリフレッシュ間隔 R が満たすべき条件を求めよ.」なお，HDCA の場合でも,「サーバのシェア漏洩確率」を「グループが乗っ取られる確率」に置き換えれば全く同様の議論ができる．

R 以内に上位 CA 機能が乗っ取られるのは，n 台中 t 台以上でシェアが漏洩した場合であり，その確率は累積二項分布関数で

$$\sum_{m=t}^{n} \binom{n}{m} (c(R))^m (1-c(R))^{n-m}$$

と表される．これを M 以下にするような (t, R) を求めればよい．

例えば，$M = 0.000001, M = 0.0001$ の 2 つの場合について，$c(R)$ と t/n の関係をグラフに表すと図 2 のようになる．漏洩確率を M 以下に抑えるためには，各曲線より左上方向（左:リフレッシュ間隔を短縮，上:閾値法の閾値を上げる）の座標相当のパラメタを設定する必要がある．一般に関数 $c(T)$ は，T に対し非常にゆっくりと増加する関数だと考えられるため，方針としては，リフレッシュ間隔を十分な余裕がある程度に短くとり，閾値はそれほど上げなくてもよいであろう．

HDCA では，グループ内でのシェア・リフレッシュ間隔を短く，また閾値をやや高めに設定してグループが乗っ取られる確率を小さくする．こうすれば，グループ間通信を減らすためにグループ間のシェア・リフレッシュ間隔を長く，またグループ数をやや少なめに取っても，ルート CA が乗っ取られる危険性は高まらず，効率と安全性を両立するような HDCA システムを構成できる．

以上は各グループ内のサーバ数が等しい場合の議論であるが，各グループ内のサーバ数が異なる場合，前述の通りグループ間のサーバの対応関係に不平等性が発生する．このためサーバ数が大きく食い違っている場合，より少ないシェア漏洩でもルート CA が乗っ取られてしまう危険性が大きくなる．安全性を保つには，できるだけ各グループのサーバ数に差異が生じないよう調整する必要がある．

[4]なお，ルート CA 機能が乗っ取られると，FDCA では安全な通信が一切できなくなるが，HDCA では，乗っ取られていない同じグループ CA が署名した証明書を持つ端末同士では，ルート CA ではなくこのグループ CA を信頼の基点とするため，依然セキュアな通信が保証される．このように，階層化でセキュリティ的に有利になる面もある．

4.3 その他の利点

その他，階層化による利点をいくつか挙げる．まず，サーバ構成の局所的な変化に対し効率的に対処できる．分散 CA システムの構成サーバが変化した場合，FDCA ではシェアの再配分 [8] をするために全サーバ対全サーバの通信を行わなければならないが，HDCA では変化が局所的な場合，再配分はグループ内のみで行えばよく，ネットワーク全体から見ると非常に効率が良い．

また，HDCA では仮想 CA が複数あるため，ネットワークの合体・分裂時にも効率的に対処できる．これは普通のネットワークでは通常ないが，アドホックネットワークのような環境では起こりうる．

他にも，モバイル端末に対してでは，ルート CA からそのノードまでの一連の公開鍵証明書をどのグループに所属しているかという位置情報として利用でき，ルーティング等の効率化にも寄与しうる．

5 まとめ

分散 CA システムは，そのロバスト性により通常の CA システムをリプレースする用途にも使われるが，Peer to Peer ネットワークのように，分散化せざるを得ない環境であるために用いられる場合もある．

階層分散 CA も，通常の階層 CA システムのリプレースとして使えるが，主たる適用分野としてはむしろ大規模な Peer to Peer ネットワークのような環境であろう．だが，このようなダイナミックな環境においては，システムの構成法についてまだ課題が残る．例えば，できるだけ平等なグルーピングをいかに素早くかつ自律的に行うかといった点である．

また，現在のグループ間でのサブシェア交換プロトコルがまだスケーラビリティに欠けるのも問題であり，より効率的なアルゴリズム[5]についてさらなる研究が必要であろう．

[5]DSA を使う場合，a, b それぞれの分散値をグループ内の各サーバが持っている時に，各分散値および a, b が分からないようにしながら $a \cdot (g^b \bmod p) \bmod q$ を求める効率的なマルチパーティプロトコルを構成できれば，通信だけでなくメモリ量の点からもスケーラビリティの高いプロトコルを構成できることが分かっている．

参考文献

[1] 山渕 深司, 近山 隆. アドホックネットワークにおける階層分散 CA システムの提案. 日本ソフトウェア科学会第 18 回大会, 3C-3, 2001.

[2] A. Shamir. How to Share a Secret. Communications of the ACM, Vol. 22, No. 11, pp. 612–613, 1979.

[3] C. Park and K. Kurosawa. New ElGamal Type Threshold Digital Signature Scheme. IEICE Trans. Fundamentals, Vol. E79-A, No. 1, pp. 86–93, 1996.

[4] Y. Frankel, P. Gemmell, P. D. MacKenzie and M. Yung. Optimal-resilience proactive public key cryptosystems. 38th Annual Symposium on Foundations of Computer Science, pp. 384–393, 1997.

[5] Y. Frankel, P. D. MacKenzie and M. Yung. Robust Efficient Distributed RSA-Key generation. 30th ATM Symposium on the Theory of Computing, pp. 663–672, 1998.

[6] M. Stadler. Publicly Verifiable Secret Sharing. *Advances in Cryptology – Eurocrypt '96 (Lecture Notes in Computer Science No. 1070)*, pp. 190–199, 1996.

[7] A. Herzberg, S. Jarecki, H. Krawczyk and M. Yung. Proactive Secret Sharing or: How To Cope With Perpetual Leakage. *Advances in Cryptology – Crypto '95 (Lecture Notes in Computer Science No. 963)*, pp. 457–469, 1995.

[8] Y. Desmedt and S. Jajodia. Redistributing Secret Shares to New Access Structures and its Applications. Technical Report ISSE TR-97-01, 1997.

[9] L. Zhou and Z. J. Haas. Securing Ad Hoc Networks. IEEE Network Magazine, Vol. 13, No. 6, pp. 24–30, 1999.