یک طرح غیرتعاملی و وارسی پذیر عمومی برای بازیابی قانونی کلید

احسان جهانگیری *، جواد مهاجری †،

چکیده

طرح‌های بازیابی قانونی کلید (Key Escrow) **اولین بار در سال 1993 توسط دولت ایالات متحدهٔ آمریکا و بمنظور تامین توأم محرمانگی کاربران و توانایی شنود مکالمات در مواقع ضروری پیشنهاد شدند.** Shamir **در سال 1995 برای جلوگیری از رمزگشایی وسیع اطلاعات کاربرانی که ارگان‌های اعمال قانون، مجاز به رمزگشایی اطلاعاتشان نیستند، طرح** Partial Key Escrow **را پیشنهاد داد. در این مقاله یک طرح جدید** Publicly Verifiable Partial Key Escrow(PVPKE) **معرفی شده است. این طرح نسبت به طرح** PVPKE، Mao **ترافیک کمتری را برای شبکه به همراه دارد. طرح پیشنهادی همچنین دارای ویژگی بازیابی متأخر است بدین معنی که حتی هنگامی که** TTP **ها بخش قابل بازیابی** (Escrow **شده) کلید را بدست آوردند هنوز** 2^{l} **مرحله برای یافتن کلید محرمانه زمان لازم است. تقلب در این طرح (بمعنی بازیابی راز برای مجموعه غیر مجاز** TTP **ها) معادل حل مسأله** Diffie-Hellman **است. این طرح علاوه بر عدم نیاز به بر خط بودن تمامی** TTP**ها هنگام** Escrow **کردن کلید (که از خصوصیات طرح های** Publicly Verifiable **است) ،می تواند بصورت غیر تبادلی پیاده‌سازی شود.**

کلمات کلیدی **Publicly Verifiable Partial Key Escrow, Publicly Verifiable Secret Sharing, Discrete Logarithm**

Non-Interactive Publicly Verifiable Partial Key Escrow

Ehsan jahangiri[1] , Javad Mohajeri[2]

[1] Department of Electrical Engineering,
Sharif University of Thechnology, Tehran, Iran
E_Jahangiri@ee.sharif.edu

[2] Faculty of Electronic Research Center,
Sharif University of Thechnology, Tehran, Iran
Mohajer@sharif.edu

Abstract

Key Escrow schemes were first suggested in 1993 by US government in order to ensure both confidentiality and interception of communications in essential cases. In 1995, Shamir proposed the Partial Key Escrow (PKE) scheme to prevent vast decryption of users' data in cases the law enforcement agencies are not permitted. In this paper, a new Publicly Verifiable Partial Key Escrow (PVPKE) scheme is proposed. This scheme has less network traffic in comparison with Mao's PVPKE scheme and offers delay recovery characteristic in the sense that there still remains 2^{l} steps to recover the secret key even when TTPs have accessed the escrowed part of the key. Wrongdoing in this scheme (in the sense of recovering the secret for an unauthorized body of TTPs) is equivalent to solving the Diffie-Hellman problem. This scheme does not require to have all TTPs online at the time of key escrowing (that is one of the characteristics of Publicly Verifiable schemes) , and it can be implemented by non-interactive manner.

Keywords

Publicly Verifiable Partial Key Escrow, Verifiable Secret Sharing, Discrete Logarithm.

* ایران- تهران- دانشگاه صنعتی شریف- دانشکده مهندسی برق E-mail: E_Jahangiri@ee.sharif.edu
† ایران- تهران- دانشگاه صنعتی شریف- پژوهشکده الکترونیک E-mail: Mohajer@sharif.edu

1- مقدمه

در طی دهۀ گذشته ارتباطات از طریق اینترنت بسیار رشد کرده است و این رشد روز افزون نیاز به تامین محرمانگی از طریق ابزارهای رمزنگاری را موجب شده است. رمز کردن اطلاعات علاوه بر اینکه محرمانگی و اصالت را برای کاربران تامین می‌کند، مشکلاتی را برای آژانس‌های اعمال قانون نیز بهمراه داشته است و از آن جمله می‌توان به کاهش توان شنود اطلاعات بمنظور مبارزه با مجرمین اشاره کرد.

در سال 1993 دولت ایالات متحده آمریکا از تولید تراشۀ رمز نگاری‌ای بنام Clipper1 خبر داد که استفاده از آن در مخابره داده-ها علاوه بر اینکه امنیت و محرمانگی کاربران را حفظ می‌کرد، خطری برای اعمال قانون و امنیت ملی بوجود نمی‌آورد. Clipper یک تراشه ضد-ضربه و ضد دستکاری2 است که همراه با دادۀ رمزشده یک حوزۀ بازیابی داده3 ارسال می‌کند و همین حوزه در بدست آوردن کلید رمزنگاری توسط نهادهای قانونی مورد استفاده قرار می‌گیرد. در سال 1994 دولت ایالات متحده این تکنولوژی را بعنوان "استاندارد رمزنگاری Escrow شده4" برای ارتباطات تلفنی غیر طبقه بندی شده شامل صوت، فکس، و داده های ارسالی (با سیم یا بی سیم) در سیستم‌های سوئیچ با نرخ ارسال مودم‌های تجاری استاندارد و یا نرخ اصلی ISDN اختیار کرد [5] و به این ترتیب برای اولین بار بمنظور جلوگیری از نقض قوانین بوسیله مجرمان طرح‌های بازیابی قانونی کلید (Key Escrow) پیشنهاد شدند.

در واقع یک طرح رمزنگاری Key Escrow طرحی است که در آن این امکان برای آژانس‌های اعمال قانون فراهم می‌آید که در مواقع ضروری و بواسطه دستور قاضی، داده‌های رمزشده را رمزگشایی کنند. این رمزگشایی علاوه بر شنود مکالمات برای جلوگیری از نقض قوانین، می‌تواند در مواقعی که کلیدهای رمزگشایی آسیب دیده‌اند و یا از بین رفته‌اند نیز مفید باشد.

در طی چند سال اخیر بحث‌های زیادی در مورد ایجاد تعادل بین نیاز به محرمانگی کاربران و توانایی اعمال قانون آژانس‌های اعمال قانون صورت گرفته است. به این ترتیب نیاز به راه‌حل‌هایی که این تعادل را به صورت مطلوبی پیاده‌سازی کنند، شدیداً احساس می‌شود.

در این مقاله در ادامه این بخش به معرفی انواعی از طرح‌های Key Escrow خواهیم پرداخت، سپس در بخش 2 به روند طی شده در مقاله‌های قبلی تا طرح پیشنهادی ما پرداخته شده است. بخش 3 به معرفی طرح جدید اختصاص دارد که چگونگی انتخاب کلیدها ،پروتکل توزیع ، پروتکل اثبات کوچک بودن توان و پروتکل بازسازی را شامل می‌شود. بخش 4 نیز به بررسی کارایی و امنیت طرح پیشنهادی اختصاص یافته است. و در نهایت بخش 5 مقاله را جمع بندی می‌کند.

1-1- (PKE) Partial Key Escrow

ایدۀ Partial Key Escrow که اولین بار توسط Shamir در سال 1995 ارائه شد ، به این منظور بود که بخش ثالث مورد اعتماد (TTP) نتواند به سرعت متن رمزشده را رمزگشایی کند. Shamir پیشنهاد کرد که تنها 8 بیت اول کلید 56 بیتی DES در اختیار TTP قرار گیرد به این ترتیب TTP مجبور خواهد بود برای یافتن کلید محرمانه جستجوی کاملی بر روی فضای 2^{48} حالتی کلید انجام دهد[3]. به سادگی می‌توان دید که بررسی کردن 2^{48} حالتِ ممکن، غیر عملی نیست ولی بهرحال یافتن حجم زیادی از کلیدها بطور سریع و همزمان نیز سخت خواهد بود.

1-2- (VPKE)Verifiable Partial Key Escrow

هنگامی که TTP بخشی قابل بازیابی (Escrow شده) کلید محرمانه را دریافت می‌کند، باید مطمئن شود که کاربر تقلب نکرده است. در واقع باید مطمئن شود که اولاً کاربر مقدار صحیحی را برای او فرستاده است و در ثانی اینکه کلید در زمان معقولی بدست خواهد آمد. این امر در طرح های VPKE تحقق می‌یابد. طرح های با قابلیت وارسی (Verifiable) اولین با توسط [9]Micali و نیز بطور مستقل توسط Bellare و Goldwasser[1] معرفی شدند. طرح‌های آنها بر پایه Diffie-Hellman و RSA بود.

1-3- Publicly Verifiable Partial Key Escrow (PVPKE)

اگر در یک طرح VPKE امکان وارسی بطور عمومی امکان پذیر باشد بگونه‌ای که هر بخش بتواند صحت سهم دریافتی هر کدام از TTPها را چک کند، طرح را PVPKE گوئیم. Mao در [4] اولین طرح PVPKE را معرفی کرده است. ایدۀ Mao برای ساخت یک طرح PVPKE بر مبنای ترکیب طرح PVSS، Stadler [13] و سیستم رمزآستانه‌ای الجمالِ قوی[5] است.

یکی از مزایایی که طرح‌های PVPKE نسبت به طرح‌های VPKE دارند این است که احتیاجی نیست، همه بخش‌هایی که قرار است قسمتی از کلید (بخشی که قرار است Escrow شود) بین آنها تسهیم شود بر خط باشند و فقط کافی است یک بخش مورد اعتماد بر خط باشد و اعتبار مقادیر ارسالی توسط کاربران را بررسی کند. (همانطور که می‌دانیم در طرح‌های VPKE هر TTP باید بر خط باشد و اعتبار مقادیر ارسالی توسط کاربران را چک کند.)[4].

2- پیش زمینه

در سال 1995 ، Micali [9] طرحی پیشنهاد کرد که بر پایه سیستم رمزنگاری Diffie-Hellman بود. در این طرح هر کاربر $x+a$ را بعنوان کلید خصوصی و g^{x+a} را به عنوان کلید عمومی اختیار می‌کند که در آن g یک مولد گروه Z_p (p یک عدد اول بزرگ) ، x یک عدد بزرگ و a یک عدد 80 بیتی(و نه بزرگتر) است (دقت شود که حاصل g^{x+a} در مد عدد اول p محاسبه می‌شود که در ادامهٔ مقاله برای سادگی عمل مدگیری ذکر نمی‌شود). سپس x از طریق یکی از طرح‌های VSS[6] ای که در [11,10,7] ارائه شده است، بین TTPها Escrow می‌شود. برای بازیابی قانونی کلید محرمانه نیز بدین صورت عمل می‌شود که: ابتدا x در پروتکل بازیابی بدست می‌آید سپس با استفاده از روش‌های حل لگاریتم گسسته، a از g^a ($g^a = \frac{P}{g^x}$) محاسبه می‌شود. بدین ترتیب کلید محرمانه بصورت $S = x + a$ بدست می‌آید.

اما لازمه طرح‌های VSS ارائه شده در [11,10,7] ، قراردادن g^x در اختیار هر TTP است. با کمی دقت متوجه می‌شویم که هرTTP از رابطه $g^a = \frac{P}{g^x}$، می‌تواند g^a را بدست آورد و به این ترتیب اگر a ، 80 بیت باشد می‌توان a را با استفاده از روش‌های حل لگاریتم گسسته مانند Shank's baby-step giant-step (این روش برای محاسبه a از g^a ، $O\left(2^{|a|/2}\right)$ زمان لازم دارد) در زمان معقولی در حدود 2^{40} مرحله بدست آورد. در چنین سیستم‌هایی به محض اینکه کلید عمومی کاربران تولید شد و بدون دانستن بخش Escrow شده کلید محرمانه، می‌توان به محاسبه بخش Escrow نشده کلید محرمانه پرداخت. پس اگر در زمانی قبل از دستور قاضی (برای بازیابی بخش Escrow شده)، بخش Escrow نشده کلید محرمانه بدست آید، هنگامی که دستور قاضی صادر شد TTP ها می‌توانند بی معطلی کلید محرمانه را بسازند. این طرح‌ها، طرح‌های با بازیابی سریع[7] نامیده می‌شوند. این طرح‌ها نسبت به طرح‌هایی که جزئی[8] نیستند مزیت بیشتری ندارند و حتی در بعضی موارد بدتر نیز هستند به این دلیل که به کاربران قول امنیتی داده شده است که تحقق نیافته است. اولین بار Bellare و Goldwasser در [1] به این مشکل اشاره کردند و راه حل آن را طرح‌های بازیابی متاخر[9] برشمردند. طرح‌های با بازیابی متأخر این اطمینان را به کاربران می‌دهند که حتی هنگامی که مقدار قابل بازیابیِ کلیدِ محرمانه ساخته شد، هنوز TTP باید 2^l مرحله برای یافتن کلید محرمانه زمان صرف کند و به هیچ طریقی نتواند این 2^l مرحله را دور بزند.

پیشنهاد Bellare و Goldwasser برای رفع این مشکل استفاده از طرحVSS ای بود که Pedersen در [6] ارائه داده بود. طرح VSS، Pedersen g^x را در فرایند تسهیم راز در اختیارTTPها قرار نمی‌داد و بدین طریق جلوی بازیابی سریع گرفته می‌شد. علاوه بر این طرح Pedersen غیرتبادلی بود. (طرح Pedersen بطور خلاصه در ضمیمه آورده شده است)

حال فرض کنید در این طرح یکی از TTP ها ادعا کند که مقدار غیر معتبری از طرف کاربر دریافت کرده است . در این حالت کاربر باید ثابت کند که جفت (s_i,t_i)ای (s_i و t_i دو نقطه بر روی دو منحنی چند جمله ای لاگرانژ هستند که در طرح Pedersen استفاده می شود; $t_i = v(i)$ ، $s_i = f(i)$) که بوسیله کلید عمومی TTP رمز کرده و برای او فرستاده است ، مقدار معتبری است و برای این کار ، کاربر مجبور است جفت (s_i,t_i) را آشکار کند و در واقع این به منزله مشخص شدن یک نقطه از منحنی چندجمله‌ای‌های بکار رفته در طرح Pedersen خواهد بود. از آنجا که در طرح‌های تسهیم راز آستانه‌ای معمولاً آستانه را $k=c+1$ (که c تعداد سهامدار[10] هایی است که احتمال می رود متقلب باشند) در نظر می‌گیرند، آشکار شدن یک نقطه از منحنی چند جمله‌ای امنیت سیستم را به خطر خواهد انداخت. و برای اجتناب از این خطر کاربر باید چند جمله‌ای جدیدی انتخاب کند و این امر خود بار عملیاتی‌ای افزون بر کاربران تحمیل خواهد کرد. بنابراین نیاز به طرحی که در آن هر TTP بتواند اعتبار مقدار دریافتی TTPهای دیگر را بررسی کند، احساس می‌شود. در صورت استفاده از چنین طرحی بار عملیاتی در هنگام بروز این مشکلات کاهش خواهد یافت. برای اینکه هر TTP بتواند اعتبار مقدار دریافتی TTP های دیگر را بررسی کند می توان از طرح‌های PVSS بجای VSS استفاده کرد. اولین بار Stadler [13] و Schoenmaker [12] سیستم‌های VSS عمومی (PVSS) را پیشنهاد دادند. اما همانطور که می‌دانیم ما برای گریز از خطر بازیابی سریع، بدنبال روشی هستیم که g^x را در اختیار TTPها قرار ندهد و از لحاظ نظری امن باشد. از لحاظ نظری امن بودن به این مفهوم است که دشمن حتی با داشتن قدرت محاسباتی نامحدود قادر به بدست آوردن کوچکترین اطلاعاتی نباشد.

در طرح های PVSS در هنگام تقلب یکی از طرفین (هنگام تسهیم راز) تبادلات بین TTPها برای خبردادن از وجود تقلب و اثبات تقلب طرف مقابل به طرز قابل ملاحظه‌ای کاهش می‌یابد. و هر TTPمستقلاً می‌تواند اثبات‌ها را بررسی کند.

3- طرح جدید

این طرح دارای زیرساختِ کلید عمومی‌ای است که $S = x + a$ را بعنوان کلید خصوصی و $P = (g')^{x+a}$ را به عنوان کلید عمومی

کاربران قرار می‌دهد. که در آن g' یک مولدِ گروهِ از مرتبهٔ اول G_q (q یک عدد اول است که رابطه $p=2q+1$ را به ازای عدد اول بزرگ p برآورده می‌کند)، x یک عدد بزرگ و a یک عدد $2l$ بیتی است(l بسته به باند محاسباتی انتخاب شده برای محاسبه لگاریتم گسسته a از $(g')^a$ اختیار می‌شود).

طرح PVPKE ای که Mao [4] پیشنهاد داده است، برای اثبات صحت هر بیت کلید خصوصی از پروتکل صفر دانائی Cramer و دیگران [8] استفاده می‌کند (بخش 3-1-3 را ببینید) و از آنجا که در پروتکل Cramer برای اثبات صحت هر بیت باید چندین پارامتر ارسال شود، حجم مخابرهٔ داده و ترافیک شبکه بالا خواهد رفت. این در حالی است که طرح پیشنهادی ما تنها برای بخش کوچکی از کلید خصوصی(a) از پروتکل Cramer بهرهٔ می-جوید (یعنی در پروتکل اثبات کوچک بودن توان a) که این خود باعث ارجح شدن طرح ما بر طرح Mao از لحاظ ترافیک شبکه می-شود.

در سال 2004 ،Tang و دیگران در [2] طرح VSS ای معرفی کردند که در گروهی از مرتبه اول کار می‌کند. به این ترتیب الگوریتم‌های حل لگاریتم گسسته‌ای مانند روش Oorschot و Wiener [14] که لگاریتم گسسته را در زمان کمتری محاسبه می‌کنند ولی در گروه های از مرتبه اول ناکارآمد هستند، تهدیدی برای خصوصیت Partiality نخواهند بود. زیرا همانطور که می‌دانیم اگر محاسبه a از g^a در زمان بسیار کمی مقدور باشد، ویژگی Partiality در خطر خواهد بود.

طرح ما از الگوریتم موجود در [2] برای تسهیم x بهره می-جوید. از آنجا که طرح [2] از لحاظ نظری امن است و $(g')^x$ را در اختیار TTPها قرار نمی‌دهد[2] طرح Key Escrow پیشنهادی از بازیابی سریع در امان خواهد بود و این در حالی است که مزایای یک طرح Publicly Verifiable را نیز دارد و بخش‌های متقلب به راحتی شناسایی خواهند شد.

برای اینکه دید کلی‌ای نسبت به این طرح پیدا شود، توضیح مختصری در ادامه می‌دهیم:

شبکه‌ای داریم متشکل از n+1 بازیگر (یک کاربر و n تا TTP)، که در آن کاربر قرار است x را بین TTPها تسهیم کند. کاربر سهم هر کدام از TTPها را با کلید عمومی آن TTP رمزکرده سپس به همراه اثبات صداقت خود برای او می‌فرستد. اثبات بگونه‌ای است که هرکدام از TTPها (و حتی هر بخش دیگری) می‌تواند بصورت غیرتبادلی اعتبار $E_i(s_i)$ ای (سهم iام TTP که با کلید عمومی آن TTP رمز می‌شود) که منتشر می‌شود را بررسی کند. به این ترتیب ادعا یک TTP متقلب در دریافت مقداری غیرمعتبر، قابل قبول نخواهد بود. و اگر هم اثبات قابل قبول نباشد ، کاربر متقلب شناخته می‌شود. در این طرح که یک طرح آستانه‌ای t از n است می‌توان راز را از یک مجموعه t عضوی از سهم‌های رمزگشایی شده با اثبات‌های معتبر بازیابی کرد. و حال پروتکل :

1-3- پروتکل:

1-1-3- Setup :

1- انتخاب عدد اول بزرگ p بگونه‌ای که $p=2q+1$ و q نیز اول باشد.

2- انتخاب G_q به عنوان گروهی از مرتبه اول q.
(اگر p یک عدد اول باشد و رابطه $p=2q+1$ نیز برقرار باشد، G_q می‌تواند مجموعه اعداد مانده مربعی از گروه Z_p^* انتخاب شود (اثبات در ضمیمه C). لازم به ذکر است در گروه‌های از مرتبه اول هر عضو غیر یک، یک مولد خواهد بود.)

3- انتخاب H, G, h, g به عنوان مولدهای مستقل از گروه G_q، بگونه‌ای که لگاریتم گسسته هیچ کدام از آنها در پایه دیگری برای هیچ کدام از طرفین معلوم نباشد.
(برای محاسبه H, G, h, g بگونه‌ای که لگاریتم گسسته آنها در پایه یکدیگر معلوم نباشد می‌توان از توابع در هم ساز امن و عمومی‌ای مانند SHA (مثلاً با خروجی 512 بیت برای $|p| = 512$) استفاده کرد بدین صورت که ابتدا g از گروه G_q به صورت اختیاری انتخاب شده و سپس دنباله $H(p,g,1), H(p,g,2), \ldots$ محاسبه می‌شود. این دنباله تا جایی ادامه می‌یابد که سه عضو از این دنباله در $G_q - \{1\}$ قرار گیرند[1]. پس از یافتن این سه عضو مجزا مقادیر H, G, h را برابر آنها قرار می‌دهیم. به عنوان مثال اگر $H(p,g,i)$، $H(p,g,j)$ و $H(p,g,L)$ در $G_q - \{1\}$ قرار داشته باشند قرار می‌دهیم :

$$\begin{aligned} h &= H(p,g,i) \ , \\ G &= H(p,g,j) \ , \\ H &= H(p,g,L) \end{aligned} \qquad (1)$$

همانطور که می‌بینیم این روش عمومی است و می توان صحت عملکرد انتخاب کننده را در انتخاب H, G, h, g بررسی کرد.)

4- انتخاب تابع درهم ساز عمومی امن $H(.)$ با $|q|$ بیت خروجی.

5- هر کاربر برای خود کلید عمومی به صورت $P = (g')^{x+a}$ انتخاب می‌کند که g' در آن یک مولد گروه G_q بوده و $S=x+a$ نیز کلید محرمانه آن کاربر است(x همان بخشی است که می-خواهیم Escrow کنیم و a آن بخش از کلید محرمانه است که باید برای بدست آوردن آن حدود 2^{40} پله زمان صرف شود). بخش x از کلید محرمانه به صورت $(s_1, s_2 \in Z_q)\ x = G^{s_1} H^{s_2}$ انتخاب

می‌شود. اینگونه انتخاب کردن x اثبات‌ها را ساده‌تر و کاراتر می‌کند[2].

6- انتخاب $h' \in G_q - \{1\}$ بگونه‌ای که لگاریتم گسسته g' و h' در پایه یکدیگر مشخص نباشد. برای این بخش هم می‌توان از روش توضیح داده شده درقسمت 3 بهره جست.

7- کاربر اعداد $y, y_0, ..., y_{2l-1} \in Z_q$ را بصورت تصادفی انتخاب کرده و $B = g'^{x} h'^{y}$ ، w بصورت:

$$w = y + y_0 2^0 + y_1 2^1 + ... + y_{2l-1} 2^{2l-1} \mod q$$

و $A_i = g'^{a_i} h'^{y_i}$ را منتشر می‌کند. که a_i در A_i بیت iام a است.

$$a = a_0 2^0 + a_1 2^1 + ... + a_{2l-1} 2^{2l-1} \ , \ a_i \in \{0,1\}$$

8- هریک از TTP ها برای خود کلید خصوصی $x_i \in Z_q^*$ را انتخاب کرده و کلید عمومی خود را $y_{i1} = G^{x_i}$ و $y_{i2} = H^{x_i}$ قرار می‌دهند.

از آنجایی که می‌توان صحتِ روش پیشنهادی برای انتخاب H,G,h را بررسی کرد، مراحل 3 و 6 می‌توانند بوسیله هر بخشی انجام شوند. انجام دیگر مراحل بغیر از مرحله 8 ، بعهدهٔ کاربر است.

حال هر کاربر، x را با استفاده از پروتکل توزیع زیر بین TTP ها تسهیم می‌کند:

2-1-3- پروتکل توزیع x:

ایدهٔ این بخش از [2] گرفته شده است. ابتدا کاربر دو چندجمله ای $f(x)$و $g(x)$ با ضرایب در Z_q و با درجه حداکثر t-1 تولید می‌کند:

$$f(x) = \sum_{j=0}^{t-1} \alpha_j x^j \quad , \quad g(x) = \sum_{j=0}^{t-1} \beta_j x^j \qquad (2)$$

که در آن:

$$\alpha_0 = s_1 \quad , \quad \beta_0 = s_2 \qquad (3)$$

و سایر α_j و β_j ها تصادفی اند.

کاربر این چند جمله‌ای‌ها را بصورت محرمانه نگهداری می‌کند ولی $commit$ ضرایب را بصورت $c_j = g^{\alpha_j} h^{\beta_j}$ برای $0 \le j < t$ منتشر می‌کند. او همچنین سهم‌های رمزشده $Y_i = y_{i1}^{f(i)} y_{i2}^{g(i)}$ برای $1 \le i \le n$ را که با استفاده از کلید عمومی TTPها تولید شده است را منتشر می‌کند. X_i نیز بصورت $X_i = \prod_{j=0}^{t-1} c_j^{i^j}$ برای همه قابل محاسبه می‌باشد. که با توجه به مقدار $c_j = g^{\alpha_j} h^{\beta_j}$ داریم:

$$X_i = g^{f(i)} h^{g(i)} \qquad (4)$$

کاربر برای اینکه ثابت کند برای i TTP مقدار غیر معتبری ارسال نکرده است باید ثابت کند که توان g و y_{i1} در X_i و Y_i با هم و توان h و y_{i2} نیز با هم برابرند. برای این منظور از امضای غیر تبادلی و صفر دانائی $Sign(X_i, Y_i, g, h, y_{i1}, y_{i2})$ بصورت زیر استفاده می‌کند:

"**1**- کاربر دو عدد تصادفی s_i', t_i' را انتخاب و عبارات $a_{1i} = g^{s_i'} h^{t_i'}$ ، $a_{2i} = y_{i1}^{s_i'} y_{i2}^{t_i'}$ ، $c = H(m \| g \| h \| G \| H \| X_i \| a_{1i} \| Y_i \| a_{2i})$ ، $r_{i1} = s_i' - c.f(i) \bmod q$ و $r_{i2} = t_i' - c.g(i) \bmod q$ را محاسبه می‌کند و سپس a_{i1} ، a_{i2} ، r_{i1} و r_{i2} را منتشر می‌کند (m متنی مشخص است).

2- وارسی‌کننده ابتدا $c = H(m \| g \| h \| G \| H \| X_i \| a_{1i} \| Y_i \| a_{2i})$ را حساب کرده و سپس صحت عبارات زیر را بررسی می‌کند:

$$a_{i1} = g^{r_{i1}} h^{r_{i2}} X_i^c \ , \ a_{i2} = y_{i1}^{r_{i1}} y_{i2}^{r_{i2}} Y_i^c \qquad (5)$$

در صورتی که این عبارات برقرار باشد، وارسی کننده از صداقت کاربر اطمینان حاصل می‌کند."

مراحل بالا برای $1 \le i \le n$ انجام می‌شود.

همانطور که دیده می‌شود بررسی صحت عبارات فوق برای هر عضو مجموعهٔ TTPها امکان پذیر است. پروتکل توزیع، یک پروتکل تسهیم راز از لحاظ نظری امن و در ضمن غیرتبادلی است[2]. از لحاظ نظری امن بودن این روش به این دلیل است که کاربر $(g')^x$ را در اختیار TTPها قرار نمی‌دهد. در ضمن مطمئن هستیم که یافتن s_2, s_1 از $g^{s_1} h^{s_2}$ عملاً مقدور نیست. این خاصیت نیز از خواص Commit ناشی می‌شود که در ضمیمه B به آن اشاره شده است.

3-1-3- اثبات کوچک بودن توان a:

برای این اثبات از پروتکل پیشنهادی Cramer و دیگران در [8] استفاده می‌کنیم. در طرح Bellare و Goldwasser [1] و همچنین طرح Mao [4] نیز برای اثبات صداقت کاربر از طرح Cramer استفاده شده است.

کاربر باید اثبات کند که a انتخابی‌اش ($a = a_0 2^0 + a_1 2^1 + ... + a_{2l-1} 2^{2l-1} \ , \ a_i \in \{0,1\}$) عددی $2l$ بیتی است و برای این کار کافی است ثابت کند A_i ، $commit$ مربوط به یک بیت (0 و یا 1) است و همچنین $commit$ همهٔ بیت‌ها نیز تولید شده است.

$$A_i = commit(a_i, y_i) = g'^{a_i} h'^{y_i} \qquad (6)$$

همانطور که در بخشSetup گفتیم $y_i \in Z_q$ بصورت تصادفی اختیار می‌شوند.

برای اثبات کوچک بودن توان a (به عنوان مثال $2l$ بیتی بودن آن) در $(g')^{x+a}$ از پروتکل صفر دانایی زیر بهره می‌جوئیم. (باید در نظر داشت که با افزایش قدرت محاسباتی به مرور زمان، تعداد بیت های a را باید افزایش داد).

داده مشترک : $h' \in G_q$ بگونه‌ای که لگاریتم گسسته g' و h' در پایه یکدیگر مشخص نباشد و $A_i \in G_q$ که باید ثابت شود یا بصورت $A_i = h'^{y_i}$ است و یا $A_i = g'h'^{y_i}$ (توان g' در A_i بیت i ام a است).

داده های در اختیار کاربر: $y_i \in Z_q$ بگونه ای که $A = h'^{y}$ یا $A_i = g'h'^{y_i}$.

ادعای کاربر: من y_i ای را می‌دانم بگونه‌ای که $A = h'^{y}$ یا $A_i = g'h'^{y_i}$.

پروتکل :

1- کاربر، R_1 و R_2 را بر اساس A_i بصورت زیر تشکیل می‌دهد:

1-1- اگر $A = h'^{y}$ باشد ، کاربر بصورت تصادفی $w_1, r_2, c_2 \in Z_q$ را انتخاب کرده و R_1 و R_2 را بصورت $R_1 = h'^{w_1}$ و $R_2 = h'^{r_2}.(A_i/g')^{-c_2}$ تشکیل می‌دهد.

1-2- اگر $A_i = g'h'^{y_i}$ باشد ، کاربر بصورت تصادفی $w_2, r_1, c_1 \in Z_q$ را انتخاب می‌کند و R_1 و R_2 را بصورت $R_1 = h'^{r_1}.A_i^{-c_1}$ و $R_2 = h'^{w_2}$ تشکیل می‌دهد.

کاربر ، R_1 و R_2 را برای وارسی‌کننده می‌فرستد.

2- وارسی‌کننده یک عدد تصادفی $c \in Z_q$ انتخاب و برای کاربر می‌فرستد. برای غیر تعاملی کردنِ اثبات می‌توان از توابع درهم ساز استفاده کرد (مانند پروتکل توزیع).

3- حال کاربر بر اساس A_i ، بصورت زیر پاسخ می‌دهد:

3-1- اگر $A = h'^{y}$ باشدکاربر ، $c_1 = c - c_2 \mod q$ و $r_1 = w_1 + c_1 y \mod q$ را محاسبه می‌کند.

3-2- اگر $A_i = g'h'^{y_i}$ باشد، کاربر ، $c_2 = c - c_1 \mod q$ و $r_2 = w_2 + c_2 y \mod q$ را محاسبه می‌کند.

حال کاربر ، r_1 ، r_2 ، c_1، و c_2 را برای وارسی‌کننده می‌فرستد.

4- وارسی‌کننده چک می‌کند که $c_1 + c_2 = c \mod q$ و $h'^{r_1} = R_1.A_i^{c_1}$ و $h'^{r_2} = R_2.(A_i/g')^{c_2}$ برقرار باشد. و در این صورت با احتمال صحت بالایی گفتهٔ کاربر را قبول می‌کند[1].

پروتکل بالا برای $0 \le i \le 2l-1$ انجام می‌شود.

برای اینکه $commit$ همهٔ بیت‌های a تولید شده باشد، باید رابطه 7 برقرار باشد که TTP می‌تواند آنرا بررسی کند:

$$P.h'^{w} = B.A_0^{2^0}.A_1^{2^1}...A_{2l-1}^{2^{2l-1}} \qquad (7)$$

P در رابطهٔ 7 کلید عمومی کاربر است[1].

در بخش بعد به چگونگی بازیابی x توسط t تا TTPای که هم سهم صحیحی از x را دریافت کرده‌اند و هم مقدار صحیحی را برای بازیابی x به اشتراک می‌گذارند، می‌پردازیم.

3-1-4- پروتکل بازیابی x :

این پروتکل شامل دو بخش است. بخش (الف) چگونگی عملکرد TTPها در رمزگشایی سهم‌ها و اثبات صداقت TTP را شامل می‌شود و در بخش (ب) چگونگی بازیابی بخش Escrow شده کلید (x) تشریح می‌گردد[2].

الف-رمزگشایی سهم‌ها و اثبات رمزگشایی صحیح:

TTP i با استفاده از کلید x_i، به محاسبه $S_i = G^{f(i)}H^{g(i)}$ از Y_i بصورت $S_i = Y_i^{x_i^{-1}}$ می‌پردازد که x_i^{-1} وارون x_i به پیمانه q است. TTP i ، S_i را به همراه اثبات اینکه S_i رمزگشایی شده صحیحی از Y_i است ، منتشر می‌کند. برای این اثبات کافی است TTP i ثابت کند که αای را می‌داند که در روابط $Y_i = S_i^{\alpha}$ و $y_i = y_{i1}y_{i2} = (GH)^{\alpha}$ صدق می‌کند. برای این کار می‌توان از $Sign(GH, y_i, S_i, Y_i)$ بصورت زیر استفاده کرد:

1- TTP i عدد تصادفی $s \in Z_q$ را انتخاب و عبارات h_1' ، h_2'، c و r را از رابطهٔ 8 حساب می‌کند:

$$h_1' = (GH)^s \ , \ h_2' = S_i^s$$
$$c = H(m \| GH \| S_i \| y_i \| Y_i \| h_1' \| h_2') \qquad (8)$$
$$r = s + c.\alpha \mod q$$

سپس h_1' ، h_2' و r را به وارسی‌کننده می‌فرستد. (m متنی مشخص است)

2- وارسی‌کننده ، $c = H(m \| GH \| S_i \| y_i \| Y_i \| h_1' \| h_2')$ را محاسبه کرده و صحت روابط 9 را بررسی می‌کند:

$$(GH)^r = y_i^c h_1' \ , \quad S_i^r = Y_i^c h_2' \qquad (9)$$

و در صورت برقرار بودن روابط 9 وارسی‌کننده از صداقت TTP i اطمینان حاصل می‌کند."

ب- بازیابی x از طریق ترکیب سهم‌ها:

با استفاده از رابطه 10 که λ_i در آن ضریب لاگرانژ و بصورت $\lambda_i = \prod_{j \neq i} \frac{j}{j-i}$ است، می‌توان راز را بدست آورد:

$$\prod_{i=1}^{t} S_i^{\lambda_i} = \prod_{i=1}^{t} \left(G^{f(i)} H^{g(i)}\right)^{\lambda_i} = G^{\sum_{i=1}^{t} f(i)\lambda_i} H^{\sum_{i=1}^{t} g(i)\lambda_i} = G^{f(0)} H^{g(0)} = G^{s_1} H^{s_2} = x \quad (10)$$

دقت کنید که TTPها احتیاجی به دانستن *f(i)* و *g(i)* ندارند و برای بازیابی راز تنها دانستن S_i ها ضروری است. همچنین TTPها ، کلید محرمانه خود x_i را آشکار نکرده‌اند ، در نتیجه می‌توانند بارها از این کلید استفاده کنند.

همانطور که قبلاً نیز ذکر شد، با بدست آمدن x می‌توان $(g')^a$ را از $(g')^a = \frac{P}{(g')^x}$ بدست آورد. برای بدست آوردن کلید محرمانه ($S = x + a$) TTP باید a را از $(g')^a$ (یک مسألهٔ لگاریتم گسسته) بدست آورد. و با بدست آمدن کلید محرمانه شنود مکالمات کاربر برای TTP مقدور خواهد بود.

4- بررسی کارایی و امنیت:

بر طبق [2] ، در این طرح رمزگشایی سهم ها بدون داشتن کلید خصوصی عملی نیست. در پروتکل بازیابی می‌توان بخشی از کلید که بوسیله کاربر Escrow شده است را یافت و اگر مجموعه TTPها مجاز نباشند هیچ اطلاعاتی(حتی جزئی) در مورد x بدست نخواهد آمد. بدست آوردن x با داشتن t-1 سهم نیز معادل حل مسأله Diffie-Hellman خواهد بود.

در این طرح تعداد توان رسانی‌ها نسبتاً کم است و در آن کاربر $5n + t + 12l$ ، عضو از گروه G_q منتشر می‌کند که 5n+t تای آن در پروتکل توزیع و $12l$ تای آن در پروتکل اثبات کوچک بودن توان صورت گرفته است. مقادیری که کاربر در پروتکل توزیع و اثبات کوچک بودن توان ارسال می‌کند در جدول 1 و 2 نشان داده شده است.

در طرح Mao، برای اثبات صحت هر بیت از کلید محرمانه از پروتکل Cramer استفاده شده است. از آنجا که پروتکل Cramer نیز باعث حجم زیاد مخابره خواهد می‌شود و حجم زیاد مخابره نیز ترافیک شبکه را در پی خواهد داشت، طرح جدید از جهت میزان بار ترافیکی شبکه، بر طرح Mao ارجح است. از آنجا که در طرح Mao برای هر بیت باید چندین عمل توان رسانی انجام شود، بار محاسباتی‌ای که طرح Mao بر کاربر و TTP ها اعمال می‌کند نسبت به طرح پیشنهادی جدید زیاد خواهد بود.

در پروتکل بازیابی x در طرح VSS، Pedersen که در [1] نیز از آن استفاده شده است، این امکان وجود ندارد که اگر یکی از TTPها مقدار اشتباهی در پروتکل بازیابی x به اشتراک بگذارد، شناخته شود و تنها می توان فهمید که در پروتکل بازسازی تقلب صورت گرفته است و یک یا بیشتر از یک TTP مقدار اشتباهی را در اختیار پروتکل بازیابی قرار داده‌اند. ولی طرح PVSS ، Tang و دیگران [2] که ما از آن برای طرح پیشنهادی جدید استفاده کرده ایم، امکان مشخص کردن دقیق TTP متقلب را فراهم می‌کند. این یکی دیگر از مزایای طرح پیشنهادی جدید است.

جدول 3 ویژگی‌های طرح جدید و طرح‌های Mao [4] و Bellare و Goldwasser [1] را بطور خلاصه نشان می‌دهد.

جدول (1): مقادیر منتشر شده توسط کاربر در پروتکل توزیع

مقادیر منتشر شده	C_j	Y_i	a_{i1}	a_{i2}	r_{i1}	r_{i2}	تعداد کل
تعداد	t	n	n	n	n	n	=5n+t

جدول (2): مقادیر منتشر شده توسط کاربر در اثبات کوچک بودن توان

مقادیر منتشر شده	R_1	R_2	r_1	r_2	c_1	c_2	تعداد کل
تعداد	$2l$	$2l$	$2l$	$2l$	$2l$	$2l$	$= 6 \times 2l$

جدول (3): مقایسه ویژگی‌های طرح جدید ، طرح Mao و طرح Bellare و Goldwasser

ویژگی / طرح	طرح جدید	طرح Mao	طرح Bellare و Goldwasser
بار محاسباتی	نسبتاً کم	زیاد	نسبتاً کم
میزان ترافیک شبکه	کم	زیاد	کم
نیاز به برخط بودن هر TTP در پروتکل توزیع	نیاز ندارد	نیاز ندارد	نیاز دارد
بازیابی متأخر	دارد	دارد	دارد

5- جمع‌بندی :

در این مقاله یک طرح جدید PVPKE ارائه شد و کارایی و امنیت آن نیز مورد بررسی قرار گرفت. همانطور که دیدیم از آنجایی طرح Mao هر دو عمل رمزنگاری سهم‌ها و اثبات کوچک بودن توان را بصورت بیت به بیت انجام می‌دهد، طرح جدید از لحاظ میزان ترافیک شبکه بر آن ارجح است. طرح جدید از بازیابی سریع نیز رنج نمی‌برد و تقلب در آن نیز معادل حل مسأله سخت Diffie-Hellman است.

ضمایم:

A- طرح VSS پیشنهادی Pedersen در [6]

روشی که Pedersen برای VSS معرفی کرده است یک روشِ از لحاظ نظری امن[11] است. " از لحاظ نظری امن بودن" بدین معنی است که دست یابی به راز تسهیم شده ، حتی با داشتن قدرت محاسباتی بینهایت نیز مقدور نیست. در روش هایی که در[11] ، [10] و [7] برای VSS ارائه شده اند ، اگر بخواهیم راز x را تسهیم کنیم ، می بایست g^x را در اختیار هر کدام از بخش ها قرار دهیم ، بدیهی است که با فرض داشتن قدرت محاسباتی بینهایت می‌توانیم x را با داشتن g و g^x محاسبه کنیم. ولی روشی که Pedersen برای VSS ارایه می‌کند ، دشمن را ، حتی با داشتن قدرت محاسباتی بینهایت ، از بدست آوردن x عاجز می‌کند. اگر بخواهیم راز x را تسهیم کنیم روش Pedersen بدین صورت عمل می کند که تسهیم کننده:

1- $f_1,...,f_t \in Z_q$ را بصورت تصادفی انتخاب می کند. $f_0 = x$ قرار می‌دهد. چند جمله ای $f(y)$ را بصورت $f(y) = f_0 + f_1 y + ... + f_t y^t$ تشکیل می‌دهد. (توجه کنید که t نباید از تعداد بخش‌هایی که احتمال می رود متقلب باشند کوچکتر باشد).

2- s_i را به صورت $s_i = f(i), i = 1,..,n$ ، محاسبه می کند. (فرض شده است n<q)

3- $v_1,...,v_t \in Z_q$ را بصورت تصادفی انتخاب می کند و F_i را بصورت $F_i = g^{f_i} * h^{v_i} = commit(f_i, v_i)$ برای $1 \le i \le t$ و $(F_0 = X, v_0 = u)$ تشکیل می‌دهد. سپس مقادیر $F_1,...,F_t$ را منتشر می‌کند. (از قبل همه F_0 را می‌دانند.)

$$\left(X = commit(x, u = random) = g^x * h^u\right) \qquad (11)$$

4- چند جمله ای $v(y)$ را بصورت $v(y) = v_0 + v_1 y + ... + v_t y^t$ تشکیل می‌دهد. t_i را بصورت $t_i = v(i)$ برای i=1,...,n محاسبه می‌کند. حال بصورت محرمانه جفت (s_i, t_i) را برای شخص i ام (i=1,..,n) می‌فرستد. حال شخص دریافت کننده در صورتی از دریافت جفت صحیحی مطمئن می‌شود ، که صحت رابطهٔ 12 برای او تایید شود:

$$g^{s_i} * h^{t_i} = \prod_{j=0}^{t} \left(F_j\right)^{i^j} \qquad (12)$$

B- خصوصیات Commit

برای Commit دو خصوصیت بسیار مهم را می توان ذکر کرد:

1-مخفی کردن:

بدین معنی که $g^a * h^b$ ، a را مخفی می‌کند. یا به بیان دیگر برای a ، b و هر $a' \neq a$ ، عبارت زیر دارای جواب است:

$$g^a * h^b = g^{a'} * h^{b'} \Leftrightarrow g^{a+b*d*\log_g^h} = g^{a'+b'*d*\log_g^h} \qquad (13)$$

می‌دانیم معادله 13 هنگامی که در یک گروه از مرتبه اول کار می‌کنیم ، به ازای a ، b و a' داده شده ، دارای جواب منحصر بفرد b' است .

2-سختی در تقلب:

$$\vartheta = (a,b) \quad , \quad \sigma = (g,h) \quad , \quad \sigma^{\vartheta} = g^a * h^b = commit(\vartheta) \qquad (14)$$

بدین معنی که انتخاب دو ϑ و ϑ' مجزا که $commit(\vartheta) = commit(\vartheta')$ برقرار باشد، کار سختی است. یا به بیان دیگر یافتن a' و b' متمایز با a و b برای معادله زیر کار سختی است.

$$g^a * h^b = g^{a'} * h^{b'} \Leftrightarrow a + b * d * \log_g^h = a' + b' * d * \log_g^h \qquad (15)$$

نشان می‌دهیم که یافتن a' و b' مجزا برای معادله 15 به معنی دانستن لگاریتم گسسته h در پایه g خواهد بود:

$$g^a * h^b = g^{a'} * h^{b'} \Rightarrow g^{a-a'} = h^{b'-b} \qquad (16)$$

از آنجا که $a' \neq a$ ، در نتیجه $b' \neq b$ ، و به این ترتیب $b'-b$ دارای معکوس (که آن را با β نشان می دهیم) خواهد بود. حال اگر طرفین معادله 16 را به توان β برسانیم داریم:

$$g^{\beta(a-a')} = h \qquad (17)$$

و این به این معنی ایست که ما مقدار لگاریتم گسسته $\log_g^h = \beta(a-a')$ را می‌دانیم.

پس همانطور که دیدیم ، تقلب کردن در این سیستم منوط به حل لگاریتم گسسته در یک گروه از مرتبه اول است که کار سختی است.

C- ساخت گروه مرتبه اول G_q

p و q دو عدد اول هستند که در شرط $p=2q+1$ صدق می‌کنند. اگر بخواهیم از گروه Z_p^* زیر گروهی (G) از مرتبه q انتخاب کنیم ، مجموعه مانده‌های مربعی $(a = w^2, w \in Z_p^*)$ a این زیر گروه را تشکیل خواهند داد ، زیرا باید رابطه $ord(G) \mid 2q$ برقرار باشد و بدین ترتیب با توجه به قضیه لاگرانژ $ord(G) = 2$ یا $ord(G) = q$ و یا $ord(G) = 2q$ خواهد بود. اگر

باشد $ord(G)=2$ به این $(a^2 \equiv w^4 \equiv 1 \bmod p)$ مفهوم است که عضوی از گروه Z_p^* از مرتبه 4 است که با $ord(w) \mid 2q,\ \forall w \in Z_p^*$ تناقض دارد. از طرفی $ord(G)=q$ بدین معنی است که $(a^q \equiv w^{2q} \equiv 1 \bmod p)$ که همیشه برقرار است ، بنابراین $ord(G)=ord(a)=q$ (این تساوی بدلیل اول بودن مرتبه برقرار شده است. در ضمن حالت $ord(G)=2q$ چون $2q > q$ است دیگر بررسی نمی‌شود).

مراجع

[1] Bellare, M., Goldwasser, S., *"Verifiable Partial Key Escrow"*, Proceedings of ACM Conference on Computer and Communications Security, 1997, pp 78-91.

[2] Tang, C., Pie, D., Liu, Z., He, Y., "Non-Interactive and Information-Theoretic Secure Publicly Verifiable Secret Sharing",
Available in: http://www.eprint.iacr.org/2004/201

[3] Shamir, A., *"Partial Key Escrow: A New Approach to Software Key Escrow"*, Private communication made at Crypto '95 Also presented at Key Escrow Conference, Washington, D.C., September 15, 1995.

[4] Mao, W., *"Publicly Verifiable Partial Key Escrow "*, 1st International Conference on Information and Communications Security (ICICS'97), Beijing, November, 1997, Lecture Notes in Computer Science, Vol 1334 Springer-Verlag, 1997 pages 409-413. http://www.hpl.hp.com/personal/Wenbo_Mao/selected papers.html

[5] Denning, D. E., Smid, M., *"Key Escrowing Today"* IEEE Communications, Vol. 32 No. 9, Sept. 1994 pp. 58-68.

[6] Pedersen, T.,*" Non-interactive and information theoretic secure verifiable secret sharing"*, Proceedings of CRYPTO, volume 576, pages 129--140. Springer-Verlag, 1991.

[7] Pedersen, T., *"Distributed provers with applications to undeniable signature"*, Advances in cryptology - EUROCRYPT91, LNCS 547, Springer-Verlag, 221-242, 1991

[8] Cramer, R., Gennaro, R., Schoenmakers, B., *"A secure an optimally efficient multi-authority election schemes"*, Advances in Cryptology, Eurocrypt '97, LNCS 1233, Springer-Verlag, pp. 103--118, 1997.

[9] Micali, S., *"Guaranteed partial key escrow"*, MIT/LCS/TM-537, Laboratory for Computer Science, Massachusetts Institute of Technology,Cambridge, MA, 1995.

[10] Micali, S., *"Fair public key cryptosystems"*, Advances in Cryptology -- CRYPTO '92, volume 740 of Lecture Notes in Computer Science, pages 113--128. International Association for Cryptologic Research , Springer-Verlag, Berlin Germany, 1993.

[11] Feldman, F., *"A Practical Scheme for Non-interactive Verifiable Secret Sharing"*, Proceedings of the 28 IEEE Symposium on Foundation of Computer Science (FOCS), IEEE, (1987)427-437

[12] Schoenmakers, B., "A Simple Publicly Verifiable Secret Sharing Scheme and its Application to Electronic Voting", Advances in Cryptology-CRYPTO'99, (1999)148-164.

[13] Stadler, M., *"Publicly Verifiable Secret Sharing"*, Advances in Cryptology EUROCRYP'96 ,(1996)190-199.

[14] Oorschot, P. C. V., Wiener, M., *"On Diffe-Hellman key agreement with short exponents"*, Advances in Cryptology-EUROCRYPT '96, volume 1070 of Lecture Notes in Computer Science, U. Maurer, ed., pp. 332 {343, Springer-Verlag, Berlin, 1996

زیرنویس‌ها

[1] در تراشه Clipper الگوریتم رمز نگاری SKIPJACK پیاده‌سازی شده است.

[2] Tamper-resistant

[3] Data Recovery Field (DRF)

[4] Escrowed Encryption Standard (EES)

[5] Robust threshold ELGamal cryptosystem (see sec 2.3 of [8])

[6] Verifiable Secret Sharing

[7] Early Recovery

[8] Partial

[9] Delay Recovery

[10] Share Holder

[11] Theoretically Secure