Sujet de thèse

Le paradigme de Cryptographie Constructive appliqué à l'amélioration de preuves cryptographiques interactives

Mots-clefs :
Cryptographie constructive, codes localement décodables/modifiables, stockage dans le nuage, ORAM, preuves cryptographiques interactives.

Responsable scientifique :
F. Levy-dit-Vehel (ENSTA-IP Paris & INRIA Saclay).

Lieu :
INRIA Saclay–Ile-de-France, équipe-projet Grace (resp. Daniel Augot).

Durée : 3 ans.

Cette thèse a pour thème général la sécurisation du stockage, de l'accès, et de la maintenance des données distantes. Elle se situe dans le prolongement des travaux de thèse de J. Lavauzelle [20, 21] et ceux de post-doctorat de N. Coxon [19], tous deux réalisés au sein de l'équipe Grace d'INRIA Saclay, sous la direction de F. Levy-dit-Vehel et D. Augot.

1 La problématique de la récupérabilité et de la mise à jour d'information

L'externalisation des données sur des serveurs distants (stockage et calculs dans le "cloud") est une tendance croissante des entreprises comme des utilisateurs individuels, permettant ainsi d'alléger la charge de stockage et de maintenance locale de ces données. L'externalisation permet aussi l'accessibilité des données à partir de plusieurs endroits, l'accès à diverses fonctionnalités ("Software as a Service", SaaS) proposées par des fournisseurs de services dans le nuage ("cloud service providers"), la possibilité d'archiver, pour plusieurs années, de gros volumes de données auxquelles on accède rarement et qui n'ont donc pas de raison d'être stockées localement.

Mais le stockage dans le cloud soulève de nouvelles menaces pour les utilisateurs. Par exemple, si celui-ci accède rarement à une donnée, comment peut-il être sûr

qu'elle est effectivement stockée, qu'elle n'a pas subi d'altération ? Un serveur distant peut très bien faire face à un problème hardware conduisant à la perte de données, et ne pas juger utile de prévenir un utilisateur qui y accède rarement, voire jamais. Un serveur malicieux peut même effacer certains fichiers très peu souvent demandés pour gagner de l'espace de stockage et ainsi faire davantage de profit.

1.1 PoR

Les preuves de récupérabilité ("proof of retrievability", ou PoR) sont des preuves (protocoles) cryptographiques interactives à l'issue desquelles l'utilisateur est convaincu (si la preuve réussit) que le serveur stocke le fichier intact, et qu'il peut donc le récupérer intégralement. La sécurité d'un tel protocole peut être énoncée comme suit : si le prouveur (serveur) réussit le protocole[1], alors un algorithme appelé "extracteur", intéragissant avec le prouveur peut extraire le fichier avec forte probabilité. Cependant, le modèle rigoureux d'adversaire, et donc de sécurité, varie suivant les schémas proposés. Citons ici [33] pour l'article initiateur en 2007; [36] utilisant des authentificateurs; [35] basé sur des codes à effacements; [34] comparant les schémas précédents, et proposant une variante à vérification publique de [33].

Une fonctionalité primordiale ici est pour l'utilisateur de pouvoir mettre à jour le fichier qu'il a externalisé, sans avoir à le récupérer en totalité. On parle alors de preuves de récupérabilité *dynamiques* (DPoR). Relativement peu de schémas ont été proposés dans ce contexte : [15] en 2011, proposant en plus une fonctionnalité éliminant les clients malhonnêtes (grâce à une structure de donnée authentifiée et un mécanisme de signature incrémentale), mais reposant sur des hypothèses cryptographiques fortes (modèle de l'oracle aléatoire), et dont les performances ne sont pas explicitées; Stefanov et al. [14], dont l'audit est coûteux en complexité de communication et calculs côté serveur; [11] utilisant une construction ORAM[2], qui a donc l'avantage de cacher les accès mémoire, mais conduisant à un schéma peu efficace en pratique; [10], efficace en complexité de communication, et dont une variante permet la vérification publique. Citons enfin [13], schéma à vérification publique utilisant des codes issus du "network coding" et des codes à effacements.

1.2 ORAM

Les techniques ORAM[3] ont été introduites par Goldreich et Ostrovsky en 96 [4]. Elles permettent à un client (ressource contrainte, ex. CPU) de stocker sa mémoire

[1] En d'autres termes, s'il répond correctement aux questions de l'utilisateur.

[2] Voir ci-dessous.

[3] Accès inconscient à une mémoire à accès aléatoire.

dans un serveur distant, et y accéder en lecture/écriture de manière aléatoire et privée : le serveur n'a pas connaissance des adresses accédées par le client, ni de leur contenu. Le client lui stocke seulement une donnée de petite taille (état, clef) qu'il garde confidentielle. Dans les constructions proposées, l'accent est mis sur les structures de données permettant de réaliser les fonctionnalités requises, plutôt que sur les protocoles. Citons [7, 8, 9], mêlant tables de hachage, fonctions pseudo-aléatoires et chiffrement symétrique. Le but est d'avoir une complexité de communication et un facteur de travail client/serveur faible (ex. polylogarithmique en la taille de la mémoire stockée). En pratique, ces constructions restent coûteuses, à cause de l'uniformisation de traitement nécessaire à l''obfuscation". Recursive Path ORAM [5], utilisant des arbres binaires[4], est actuellement la plus efficace (complexité de lecture/écriture en $O(\log^2 N)$ pour une mémoire de N blocs, chacun de taille $\Omega(\log N)$), parmi les schémas à stockage faible côté client (dans ce cas : stockage polylogarithmique[5] en N).

1.3 PIR

L'accès confidentiel à un fichier d'une base de données distante est également un service que le fournisseur doit proposer : il s'agit pour l'utilisateur d'accéder à un fichier sans que le serveur sache de quel fichier il s'agit. Dans ce contexte, c'est la requête, et non nécessairement le contenu du fichier, qui doit être confidentielle. Cette problématique est la récupération confidentielle d'information ("private information retrieval", ou PIR). Lorsqu'il est impossible pour un serveur d'acquérir de l'information sur la requête, même avec une puissance de calcul illimitée, on parle de IT-PIR (PIR sûr au sens de la théorie de l'information), tandis que dans le cas d'une sécurité calculatoire, on parle de C-PIR ("computational" PIR). La solution triviale de récupérer toute la base de données dès lors que le client veut accéder à un fichier est évidemment inenvisageable en termes de complexité de communication. Chor et. al [29] ont prouvé que, lorsqu'une base de données de k-bits est stockée sur un seul serveur, un protocole de PIR ne peut être IT-sûr avec moins de $\Theta(k)$ bits de communication. Ce constat a amené à s'intéresser au stockage sur plusieurs serveurs, si la sécurité au sens IT est requise.
La littérature sur le sujet est abondante. Dans les protocoles IT-sûrs, chaque serveur doit répondre à une requête aléatoire du client en calculant une information partielle relative à la base de donnée; le client recouvrant ensuite le symbole désiré à partir des réponses des serveurs. Citons notamment [29], de complexité de communication $O(\ell \log \ell k^{1/(\log \ell)})$ pour ℓ serveurs (base de données répliquée sur chaque serveur, d'où un surcoût de stockage important). Les codes localement

[4]La construction, assez complexe, n'est pas développée ici.
[5]Stockage linéaire en N dans la version non récursive de Path ORAM.

décodables (voir ci-dessous) respectant une propriété d'uniformité conduisent naturellement à des protocoles de IT-PIR [23]. Une réciproque générique, de IT-PIR vers LDC, existe également : [30], avec un protocole de PIR en complexité de communication en $k^{O(\frac{\log\log\ell}{\ell\log\ell})}$. Une série de travaux s'attachant à distribuer la base de données plutôt qu'à la répliquer sur les serveurs s'en est suivie: notamment [31] pour trois serveurs; [21] et [17], de complexité de calcul constante côté serveurs et assurant une résistance aux serveurs byzantins[6] ; aussi [32] de complexité de communication sous-polynomiale en $O(k^{\sqrt{\log\log k/\log k}})$ pour deux serveurs, et enfin [27] pour une tolérance optimale aux serveurs byzantins.
La thèse comprendra un volet de recherche bibliographique sur le C-PIR, qui a peu été investigué dans l'équipe Grace. L'article [28] et les citations afférentes pourront néanmoins constituer un point de départ.

2 Codes localement décodables/modifiables

En 2000, Katz & Trevisan [23] ont introduit le concept de *codes localement décodables*, qui trouve ses fondements dans les "preuves vérifiables en probabilité" (PCP). L'idée est que, pour retrouver un symbole particulier avec forte probabilité, seulement quelques positions du mot encodé sont nécessaires. De tels codes possèdent une application naturelle en récupération confidentielle d'information [26].

L'enjeu principal des codes localement décodables est de construire de tels codes avec le minimum de redondance et la plus petite complexité de requêtes, ou *localité* (nombre minimal de symboles encodés nécessaires pour reconstruire un symbole particulier). Un résultat majeur dans ce domaine est la construction, par Kopparty, Saraf & Yekhanin en 2011 [24], de codes localement décodables dont le taux de transmission k/n est $> 1/2$, avec une localité sous-linéaire en k : les codes à multiplicité. Une autre approche, due à Guo, Kopparty & Sudan [22], utilise des relèvements de codes invariants sous le groupe-affine. Cette construction est motivée par le fait que tout relèvement d'un code affine-invariant de longueur q^t sur $\mathbb{F}_q$ à un code de longueur q^m, $m > t$, est localement décodable avec une localité au plus q^t. Citons également les "expander codes" construits dans [25], et possédant un taux élevé.

Afin de permettre la mise à jour locale des fichiers, N. Chandran et al. [37] ont introduit la notion de codes localement modifiables et localement décodables (LULDC : locally updatable locally decodable codes) [7]. Schématiquement, ces codes - en plus d'être décodables localement - permettent de mettre à jour un

[6] [17] ayant un sur-stockage optimal.

[7] A noter que des codes localement modifiables ont déjà été proposés dans [15], mais l'encodage se fait pour chaque bloc séparément.

symbole d'un message m (et donc de mettre à jour l'encodage de ce nouveau message) en modifiant seulement quelques symboles de son encodage $c = E(m)$.

N. Chandran et al. construisent de tels codes à partir d'une structure de données hiérarchisée introduite par Ostrovsky dans le contexte ORAM, couplée à l'utilisation de codes localement décodables. Deux variantes sont proposées, selon que l'on se place dans le modèle sûr au sens de la théore de l'information, ou le modèle calculatoirement sûr.

3 Cryptographie constructive

Le modèle de cryptographie constructive (CC) a été introduit par U. Maurer en 2011 [1]. Il s'agit de redéfinir les protocoles cryptographiques de base (chiffrement, authentification, échange de clef...) à partir de systèmes discrets de trois types : les ressources (ex. canal, clef...), les convertisseurs (ex. fonction de chiffrement, hachage), et les distingueurs (système se connectant à une ressource par ses interfaces d'entrée, et sortant un bit dans son interface de sortie).

Cette approche permet non seulement de fournir un modèle de construction effective des objets et protocoles, mais en plus d'affiner les preuves de sécurité : en effet, il est alors possible d'évaluer précisément la sécurité des constructions obtenues, en particulier les briques minimum à mettre en œuvre pour réaliser une fonctionnalité donnée (voir par exemple [2]).

Dans [3], les auteurs, partant d'un système de communication basique[8] modélisé comme une ressource avec des interfaces, y ajoutent des briques modulaires (convertisseurs) permettant de le rendre authentifié puis confidentiel puis complètement sécurisé (appelé s-SMR : *secure server memory resource*).

Une force de ce modèle est qu'il est "composable", dans le sens où si deux briques (ex. intégrité et confidentialité) ont été prouvées sûres dans celui-ci, leur composition résulte en un système sûr. Ainsi les preuves de sécurité s'en trouvent simplifiées, car une fois prouvée la sécurité d'une ressource (ex. intégrité du serveur), celle-ci se transporte lorsque l'on rajoute une fonctionnalité supplémentaire (caractère "incrémental" des preuves de sécurité).

4 Travail envisagé

Nous proposons dans cette thèse d'explorer le gain qu'il y a à appliquer le paradigme de cryptographie constructive à des protocoles cryptographiques intervenant dans la sécurisation des données dans le cloud : preuve de stockage, preuve de récupérabilité statique/dynamique, récupération confidentielle d'information, etc. Le gain peut

[8] Ex. client/serveur.

être : la relaxation de certaines hypothèses faites en pratique dans d'autres modèles, une définition plus rigoureuse du modèle de sécurité[9] et/ou une simplification des preuves de sécurité, ou encore une mise en oeuvre effective simplifiée.

4.1 PoR, DPoR et PIR à partir de CC et LULDCs

Dans un premier temps, il s'agira de prolonger l'étude initiée dans [3] concernant l'audit d'un serveur construit sur le modèle CC. D'une part - et c'est une voie encore non investiguée dans la littérature - pour l'adapter à des protocoles de PIR; d'autre part pour réaliser des preuves de récupérabilité. Un point de départ pour les PoR sera de voir comment réécrire les schémas de PoR proposés dans [16] et [18] dans le contexte CC.

Concernant le PIR, on peut remarquer que la construction d'un s-SMR permet naturellement l'écriture dans les cellules. De plus, comme les adresses mémoire sont masquées, le s-SMR semble réaliser une sorte de PIR avec possibilité d'écriture, ce qui serait une fonctionalité très utile, et mérite d'être explorée. D'un autre côté, comment relaxer le design du s-SMR pour arriver à un PIR classique, et quelles seraient les performances d'une telle construction ? A noter qu'il s'agit ici d'un PIR calculatoire (car les briques pour passer d'un serveur non sécurisé à un serveur sécurisé reposent sur un modèle de sécurité calculatoire). Il serait donc intéressant de comparer ses performances avec les constructions existantes de C-PIR.
On peut également se demander si un IT-PIR est envisageable dans le modèle CC.

Constructions de LULDCs

Concernant les PoR/DPoR, les auteurs de [3] évoquent par exemple des codes à effacements pour réaliser l'audit (sans toutefois instancier ces codes) d'un serveur. Nous envisageons d'utiliser des codes localement décodables (LDC) pour l'audit. L'intérêt ici est de ne pas avoir à lire/écrire tout le mot de code, ce qui est très coûteux dans le cas de gros fichiers.

Pour réaliser des mises à jour, il est nécessaire d'utiliser des codes localement modifiables (LULDC).
Cette utilisation des LDCs/LULDCs à la place de codes à effacements implique la remise à plat de toute la construction de [3] afin d'en évaluer l'impact sur la sécurité du système, sur sa praticabilité, ainsi que sur ses performances.
Il est notamment probable que les LULDCs qui seront construits dans notre contexte respecteront un modèle d'adversaire différent de celui considéré par N. Chandran et al.

[9] En particulier pour l'application aux PoR/DPoR.

Un volet important de la thèse sera donc de regarder comment les constructions de codes localement décodables à haut rendement évoqués précedemment ([24], [22], [25], [20]) peuvent être adaptées pour les transformer en codes localement modifiables.
Etant donnée l'applicabilité de tels codes : accès en écriture à une base de données, PoR dynamiques..., l'enjeu de construire des LULDCs avec de bons paramètres est de taille.

Les constructions trouvées impacteront directement l'efficacité et donc la praticabilité des accès aux serveurs distants utilisant ces techniques.

CC, ORAM, et LULDC

Des travaux récents de P. Grubbs et al. [6] ont montré qu'il était possible de reconstruire approximativement une base de données chiffrée dès lors que le serveur avait accès à l'historique des accès mémoire effectués par le client; cette approximation pouvant être rendue exacte si la distribution des intervalles des adresses accédées est uniforme. Ce travail met en évidence la nécessité de mettre en oeuvre des mécanismes de ORAM pour accéder aux données, en lecture comme en écriture.
La construction d'un serveur sécurisé dans le modèle CC étant étroitement liée au design de Path-ORAM d'une part, et les LULDC proposés par N. Chandran étant eux aussi basés sur un schéma ORAM, comprendre en profondeur et exploiter les liens entre ces concepts et outils permettra certainement de construire des protocoles de vérification/récupération/modification de données dans le cloud plus sûrs et plus efficaces.

Une perspective plus vaste de la thèse consistera à étendre l'application des codes localement décodables/modifiables dans le modèle CC à la sécurisation d'autres usages émanant de problématiques réelles, comme l'accès[10] anonymisé et confidentiel de clients à une base de données en présence de collusions, avec des contraintes de bande passante et/ou de complexité des calculs (voir par exemple [12]).

4.2 Mise en oeuvre informatique

Une large part du travail sera consacrée à la mise en œuvre des différentes ressources et protocoles. Création d'abord d'une "boîte à outils" de fonctions correspondant à la construction d'un SMR (serveur memory resource), d'un a-SMR (serveur authentifié), c-SMR (confidentiel), et s-SMR (serveur sécurisé). Puis mise en œuvre des protocoles construits dans la thèse.
Le but principal est ici, outre d'évaluer la facilité d'implantation des constructions,

[10] En lecture, et éventuellement aussi en écriture.

de comparer les constructions basées sur le modèle CC avec les constructions existantes réalisant des PoR, DPoR, PIR. Les points de comparaison étant essentiellement la complexité de communication, le coût du stockage côté serveur/côté client, la complexité de calcul côté serveur/côté client.

References

[1] Ueli Maurer. Constructive Cryptography - A new Paradigm for Security Definitions and Proofs. *Proceedings of TOSCA 2011, LNCS 6993, pp.33-56, 2012.*

[2] S. Coretti, U. Maurer and B. Tackmann. Constructing Confidential Channels from Authenticatd Channels - Public-key Encryption Revisited. *Advances in cryptology, Proceedings of Asiacrypt 2013, LNCS vol.8269, pp.134-153, dec. 2013.*

[3] Christian Baderstscher and Ueli Maurer. Composable and Robust Outsourced Storage. *Full version of a paper at CT-RSA 2018. See also* eprint.iacr.org/2017/133.

[4] O. Goldreich and R. Ostrovsky. Software Protection and Simulation on Oblivious RAMs. *Journal of the ACM, 1996.*

[5] E. Stefanov, M. van Dijk, E. Shi, T-H. H. Chan, C. Fletcher, L. Ren. X. Yu. S. Devadas. Path ORAM : An extremely Simple Oblivious RAM Protocol.

[6] Paul Grubbs, Marie-Sarah Lacharité, Brice Minaud and Kenneth G. Paterson Learning to Reconstruct : statistical learning theory and Encrypted Database Attacks. To appear in "Security and Privacy" S&P 2019.

[7] M.T. Goodrich and M. Mitzenmacher. Privacy-preserving access of outsorced data via oblivious RAM simulation. *Proceedings of ICALP 2011.*

[8] B. Pinkas ans T. Reinman. Oblivious RAM revisited. *Crypto 2010.*

[9] E. Shi, T-H H. Chan, E. Stefanov and M Li. Oblivious RAM with $O((\log N)^3)$ worst-case cost. *Proceedings of Asiacrypt 2011, pp.197-214, 2011.*

[10] E. Shi, E. Stefanov and C. Papamanthou. Practical Dynamic Proofs of Retrievability. *Proceedings of the ACM Conference on Computer and Communications Security, pp.325-336, 2013.*

[11] D. Cash, A. Kupçu and D. Wichs. Dynamic Proofs of Retrievability via Oblivious RAM. *Proceedings of Eurocrypt 2013.*

[12] A. Hamlin, R. Ostrovsky, M. Weiss and D. Wichs. Private Anonymous Data Access. *eprint.iacr.org/2018/363, 2018.*

[13] Z. Ren, Q. Wang and M. Xu. Dynamic Proofs of Retrevability for Coded Cloud Storage Systens. *IEEE Trans. on Services Computing, 2018.*

[14] E. Stefanov, M. van Dijk, A. Juels and A. Oprea. Iris : a scalable cloud file system with efficient integrity checks. *ACSAC 2012, pp.229-238, 2012.*

[15] Q. Zheng and S. Xu Fair and Dynamic Proofs of Retrievability. *CODASPY'11, Feb.21-23, San Antonio, USA, 2011.*

[16] Julien Lavauzelle and Françoise Levy-dit-Vehel. New Proofs of Retrievability using Locally Decodable Codes. *ISIT'2016, pp.1809–1813, DOI: 10.1109/ISIT.2016.7541611.*

[17] D. Augot, F. Levy-dit-Vehel, A. Shikfa. "A Storage-efficient and Robust Private Information Retrieval Scheme allowing few servers". *Proceedings of 13th International Conference on Cryptography and Network Security, CANS'2014, Springer vol.8813 pp.222–239*, 2014.

[18] J. Lavauzelle and F. Levy-dit-Vehel. Generic Constructions of PoRs from Codes and Instantiations. *Journal of Mathematical Cryptology, doi.org/10.1515/jmc-2018-0018, 2019.*

[19] N. Coxon. Fast systematic encoding of Multiplicity Codes. *CoRR abs/1704.07083, eprint ArXiv 2017.*

[20] J. Lavauzelle. Lifted Projective Reed-Solomon Codes. *Designs, Codes and Cryptography, to appear.*

[21] J. Lavauzelle. Private Information Retrieval from transversal designs. *IEEE Trans. on Inf Theory 65(2); pp.1189-1205, 2019.*

[22] Alan Guo, Swastik Kopparty and Madhu Sudan. New affine-invariant codes from lifting. *ECCC report no. 149, nov. 2012.*

[23] Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *STOC 2000, Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, 2000.*

[24] Swastik Kopparty, Shubangi Saraf and Sergey Yekhanin. High-Rate Codes with Sublinear-Time Decoding. In *STOC 2011, Proceedings of the Fourty-third Annual ACM Symposium on Theory of Computing, 2011.*

[25] Brett Hemenway, Rafail Ostrovsky and Mary Wooters. Local correctibility of expander codes. *ICALP 2013 - special issue on Information and Computation.*

[26] Sergey Yekhanin. Locally Decodable Codes and Private Information Retrieval Schemes. *Information Security and Cryptography, Springer 2010.*

[27] C. Devet, I. Goldberg and N, Heninger. Optimally Robust Private Information Retrieval. *21st USENIX Security symposium, 2012.*

[28] C. Devet and I. Goldberg. The best of both worlds : combining IT- and C-PIR for communication efficiency. *Privacy Enhanced Technologies (PETs) 2014.*

[29] B. Chor, O. Goldreich, E. Kushilevitz and M. Sudan. Private Information Retrieval. *J. of the ACM 45(6), pp.965-981, 1998.*

[30] A. Beimel, Y. Ishai, E. Kushilevitz and J.F. Raymond. Breaking the $O(n^{1/(2k-1)})$ barrier for IT-PIR. *43rd Symposium on Foundations of Computer Science (FOCS 2002), IEEE computer society, pp.271-270, 2002.*

[31] K. Efremenko. 3-query locally decodable codes of subexponential length. *SIAM Journal of Computing 41(6), pp.1694-1703, 2012.*

[32] Z. Dvir and S. Gopi. 2-server PIR with subpolynomial communication. *Journal of the ACM 63(4), pp 39.1-39.15, 2016.*

[33] Ari Juels and Burton S. Kaliski Jr. PoRs: Proofs of Retrievability for Large Files. *Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007.*

[34] Kevin D. Bowers, Ari Juels and Alina Oprea. Proofs of Retrievability : Theory and Implementation. *ACM Cloud Computing Security Workshop (CCSW), pp. 43-54, 2009.*

[35] Yevgeniy Dodis, Salil P. Vadhan and Daniel Wichs. Proofs of Retrievability via Hardness Amplification. *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, 2009.*

[36] Hovav Shacham and Brent Waters. Compact Proofs of Retrievability. *Advances in Cryptology - ASIACRYPT 2008, 14th International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, Australia, 2008.*

[37] N. Chandran, B. Kanukurthi and R. Ostrovsky. Locally Updatable and Locally decodable Codes. *Proceedings of TCC 2014, pp.489-514, 2014.*