ON SECURITY AND PRIVACY IN CLOUD COMPUTING

Daniel Slamanig[1] and Stefan Rass[2]
[1]*Department of Medical Information Technology, Healthcare IT & Information Security Group*
Carinthia University of Applied Sciences, 9020 Klagenfurt, Austria
[2]*Institute of Applied Informatics, System Security Group, Klagenfurt University, 9020 Klagenfurt, Austria*

Keywords: Cloud computing, Cloud storage and computation services, Security, Privacy, Non-standard cryptography.

Abstract: Cloud computing is an evolving paradigm that is believed to play a key-role in future information processing. It is reasonable to expect a cloud computing environment equipped with security systems, but anything not covered by standard measures such as firewalls or encrypted channels is subject to mere trust in the cloud provider. The acceptance of cloud computing might be higher if less trust in the infrastructure is demanded, thanks to a more comprehensive employment of cryptography for security and privacy. Despite a vast amount of cryptographic primitives available today, their full power still remains to be exploited for numerous aspects in cloud computing. The goal of this paper is drawing attention to various primitives in cryptography that might become or actually are already considered to be useful in a cloud computing environment, but have not received as much attention as they deserve from experts in this area.

1 INTRODUCTION

Cloud computing is a paradigm in which information processing and storage is outsourced to an external service provider, hosting either hardware or software resources that can be utilized on a “pay-as-you-go” basis. The customer’s benefit is sparing the need to buy expensive hardware or software for a perhaps short-term usage and capable to cope with peak loads or to avoid burdening the own system with resource-intensive computations. At the same time, the customer needs to be assured that his information is not lost, leaked or otherwise misused by hackers, other customers or the cloud provider himself. However, achieving this kind of assurance is highly nontrivial for various reasons. If the cloud provider claims that his infrastructure is secure, how can this claim be substantiated for the customer? If the customer distrusts the cloud provider, what measures can he take in order to have his information processable yet inaccessible for unauthorized parties?

Cryptographic research has considered a vast amount of problems that are applicable to those encountered when building a secure cloud computing environment. However, since cloud computing is a very recent field, their applications to cloud computing have not been considered adequately. Nevertheless, for many issues arising in a cloud computing environment, there might be a perfect but perhaps little-known solution available in cryptography. The goal of this work is to shed light on ”cloud cryptography”.

2 STAKEHOLDERS AND ASSETS

We consider an abstract model of cloud computing whereas the cloud provider (CP) is assumed to be a monolithic entity. Besides, we have a set of cloud users (CUs) who store data items in the cloud for later access and processing. Each user (i.e. data owner) might have his own favored policy, defining access control permissions for other CUs. This policy may specify such permissions in terms of required attributes, roles, or rights that either a CU or the CP must possess.

Obviously, security in this setting is mutual: CP as well as CUs need protection of their assets from each other’s negative influence. The CP is interested in the fair, authorized and observant use of his facilities, and the protection of his own infrastructure against damage. On the other hand, the CU is interested in the secrecy and integrity of his information, computations over his data, its correct execution and his privacy.

For many applications, it is useful to classify an adversary in terms of his capabilities. In a cloud computing environment, such a classification could

Slamanig D. and Rass S..
ON SECURITY AND PRIVACY IN CLOUD COMPUTING.
DOI: 10.5220/0003382106040609
In *Proceedings of the 1st International Conference on Cloud Computing and Services Science* (CLOSER-2011), pages 604-609
ISBN: 978-989-8425-52-2

be made up of two attributes, i.e. an adversary can be *active* or *passive*, as well as *internal* or *external*. In either case, we will assume a (computationally) bounded attacker. In cryptography, the distinction between active and passive adversaries is very important, since a passive adversary is restricted to obey the protocols and merely attempts to extract information, as opposed to an active adversary who is not bound to hold on to any protocol specification. Consequently, he may insert, delete, modify, delay or block information to his own advantage. Building a defense against an external adversary amounts to protection of links and nodes in the (cloud's) network. Standard encryption and access control primitives can do that. Defending against an internal (CP related) adversary is more sophisticated, and one important countermeasure is storing solely encrypted data in the cloud. However, there are additional issues discussed below.

2.1 Requirements

Subsequently, we discuss security and privacy requirements that are considered important from our point of view: **Anonymity** is a demand of the CU, since leaving footprints everywhere in the cloud supports profiling the user's behavior and might yield unwanted implications for the CU. Closely related to anonymity is **unlinkability**, which means that any two actions observed (by the CP) may not effectively be relatable to a CU, nor to each other. Both terms appear frequently in the cryptographic literature and have seen sophisticated ways of achieving them.

Authorization is a multi-layered term with different semantics depending on the point of view. From the CP's perspective, it is necessary to assure that resources are used on a fair base and that resource limits, e.g. storage space or CPU time, are strictly obeyed. Furthermore, a user may neither gain access to past resources (*backward secrecy*) nor retain access to resources after expiration of the contract (*forward secrecy*). The problem of a CU subletting his resources is most challenging, and from the field of *broadcast encryption*, various techniques for *traitor tracing* are known (Boneh and Naor, 2008; Jin and Lotspiech, 2009), as well as forward and backward security are comprehensively treated in this area.

From the customer's point of view, authorization means retaining access rights and the ability to grant or revoke access to data items or resources stored in the cloud. This problem can be addressed on several levels: making it part of a CU's contract is most straightforward but reduces security to pure trust. Keeping information encrypted in the cloud with the owner keeping the key outside the cloud provider's scope (cf. sections 3.2.1 and 3.2.2) will ensure inaccessibility, especially by insiders.

Availability. It can be assumed that the cloud infrastructure provides high availability. Nevertheless, the access of services in the cloud may be endangered due to denial of service (DoS) attacks. Actually, (distributed) DoS attacks are assumed to be a major security issue in cloud computing (Jensen et al., 2009).

Confidentiality. If the mere CP's assurance regarding data confidentiality is not enough, then a user can keep confidentiality under his own control by relying on *data encryption* in general or *fully homomorphic encryption* (cf. sections 3.2.1 and 3.2.2) when computations on these data should be performed in the cloud in a confidential way. In a nutshell, the latter cryptosystems permit performing calculations on an encrypted piece of data without ever decrypting it.

Integrity. When CUs store their data in the cloud, they are interested in detecting whether their data was tampered with and whether the CP indeed is able to deliver all the stored data of a CU if required (cf. section 3.2.6). Note, that besides external or internal adversaries who intentionally modify data, CPs may also try to reduce costs by discarding CUs data that is not accessed or rarely accessed (and hoping that it won't be accessed anymore).

Regarding computations, integrity would mean the assurance of the correct algorithm being executed correctly. Fault injections are attacks targeting the execution of a (cryptographic) protocol with the goal of having secret information extractable from one or more faulty outputs. Fault injections are a well-studied problem and related countermeasures could do well in cloud computing environments either.

Private Computations. Besides data, an algorithm can equally well be a business asset. In that case, fully homomorphic encryption may be adapted to achieve this. Another solution are *garbled circuits*, to be expanded in sections 3.2.1 and 3.2.3.

Provenance. Audit trails, which are deemed to be an important issue in cloud computing (Lu et al., 2010), are an application of the more general concept of provenance. Basically, provenance is metadata that describes the history of an object, be it a document, a process, a transaction, etc. Because data can be shared widely and anonymously, data consumers may have no means to verify its authenticity and integrity. Therefore, (Muniswamy-Reddy et al., 2010) argue that provenance should be incorporated as a core cloud feature. Thereby, it would be desirable that audit trails do not reveal any information to unauthorized users (preserve the privacy of all involved parties), are still unforgeable, i.e. any unauthorized user cannot forge a valid audit trail (entry), and the provenance works correctly (Lu et al., 2010).

Provisioning. Quality of service is closely related to authorization, as fair service use policies are a widely accepted standard. Secure provisioning means retaining the quality of service even in the presence of adversaries. Game-theoretic tools like those outlined in (Rass and Schartner, 2010) have been designed for optimized service provisioning in various aspects of security (including confidentiality and availability). As a by-product, these yield optimal service provisioning strategies, which may be adopted in cloud computing.

Verifiability. Regardless of whether the cloud is used for computations or storage, verifiability of outputs is a highly nontrivial issue. In case of cloud storage, message authentication codes or digital signatures can do the job. Doing computations in a verifiable manner is far less trivial. Solutions to verifiability of computations are given in section 3.2.3.

3 CLOUD CRYPTOGRAPHY

In this section we present a classification of resources in context of cloud computing and discuss various (non-standard) cryptographic approaches which can be used to realize properties discussed in the previous section.

3.1 Classification of Resources

In general, we can abstractly classify the use of cloud computing into two classes of resources which we briefly discuss below.

Cloud Storage. Here CUs store data in the cloud and may want to selectively share their data, i.e. provide other CUs permissions such that they are able to read or modify the data. As already mentioned, *encrypted* storage of data seems to be sine qua non in order to prevent unauthorized access from external and internal (CP related) adversaries. But when encrypted data is stored in the cloud, selectively sharing these data and realizing searches on these encrypted data items in an efficient way are challenging tasks. Nevertheless, encryption is needed to preserve confidentiality. As we discuss below, *attribute-based* and *searchable encryption* are promising building blocks to realize these tasks. Furthermore, if CUs want to hide their identity and/or which data they are accessing from the CP, while the CP can be sure that authorization is still given, one can employ suitable *privacy-enhancing cryptographic protocols* or *private information retrieval* and *oblivious transfer*, respectively. Another issue is provable data possession, which allows CUs to check whether their data is still retrievable from a CP. We refer the interested reader also to (Kamara and Lauter, 2010) for several proposals for architectures to realize cryptographic cloud storage.

Cloud Computation Services. Here CUs want to outsource computations on data which may already be stored in the cloud. As discussed above, usually it will be desirable that data are hidden from the CP and thus computations are only performed on *encrypted* data. If the CP performs the computations, we have the following scenarios:

CP knows f. The CP does not know the actual data x, but is aware of what function f he is computing.

CP doesn't know f. The CP neither knows the actual data x nor what the function f does. In this scenario, CP will evaluate a function f' obtaining a result $y' = f'(x)$, whereas the CU is able to derive y from y' such that $y = f(x)$.

Main tools in this context are *fully homomorphic encryption* and *garbled circuits* (discussed below). We refer the reader also to (van Dijk and Juels, 2010) for a discussion of private multi-party computations, i.e. inputs for CP's computations are from multiple CUs, and an impossibility result in this setting.

3.2 How Cryptography Helps

Below, we discuss the above mentioned techniques.

3.2.1 Fully Homomorphic Encryption

Fully homomorphic encryption was a longstanding open problem and recently realized by Gentry (Gentry, 2009). This solution represents a semantically secure public-key encryption scheme that allows the computation of an arbitrary function f on *encrypted* data by using the respective public-key PK. Consequently, if CU stores some encrypted data $c = E_{PK}(x)$ in the cloud, CP can compute $c' = Evaluate_{PK}(f, c)$ on the encrypted data, which results in a value c' such that $f(x) = D_{SK}(c')$ holds. Thus, CP performs the correct computation, but does not learn neither the input nor the result in plaintext. In (Gennaro et al., 2010), fully homomorphic encryption is combined with garbled circuits for verifiability, i.e. CU's can check whether CP has performed the computation correctly without locally re-executing the entire computation by himself. Furthermore, Gentry notes that it is easy to tweak the fully homomorphic scheme to provide unconditional circuit privacy, i.e. even the functionality f is hidden from CP (garbled circuits discussed below achieve this too). A recent working implementation of Gentry's scheme (Gentry and Halevi, 2010) incorporating some recent experiences and improvements (Stehlé and Steinfeld, 2010) shows that these schemes are, unfortunately, still far too impractical and it will take some time until we will see them in practice.

3.2.2 Attribute-based Encryption

In a *ciphertext-policy attribute-based encryption* scheme (CP-ABE), a user's private-key is associated with a set of attributes and a ciphertext specifies an access policy over a universe of attributes. A user will be able to decrypt a ciphertext, if and only if his attributes satisfy the policy of the respective ciphertext. Bethencourt et al. (Bethencourt et al., 2007) proposed the first CP-ABE, where a message can be encrypted with respect to policies defined over attributes using conjunctions, disjunctions and (k,n)-threshold gates. Note, that besides achieving confidentiality due to storing encrypted data in the cloud, this concept allows to realize implicit authorization, i.e. authorization is included into the encrypted data and only people whose attributes satisfy the associated policy can decrypt data. This has recently been considered as an interesting concept in context of cloud based electronic health records (Li et al., 2010; Akinyele et al., 2010), where stored data is very sensitive, high confidentiality guarantees and fine-grained authorization are required.

CP-ABE can be considered as a special case of functional encryption (Lewko et al., 2010). This means that a party encrypting a message embedds a ciphertext descriptor d_c into the ciphertext and private-keys have associated key descriptors d_k. A decryption can only be successfull if a certain relation R between d_c and d_k holds.

3.2.3 Garbled Circuits

The concept of *garbled circuits* (GCs) was introduced by Yao in (Yao, 1986) as a means to realize secure computation of arbitrary functions. GCs can be used in the cloud setting in the following way: The CU can take a function f (as a boolean circuit) which should take as input some data x. Then he produces a garbled circuit f' and garbled data x', which do not provide any information on f and x, and lets the CP evaluate $y' = f'(x')$. If CU is given the output values y', he is able to reconstruct the original output $y = f(x)$. Recently, Sadeghi et al. (Sadeghi et al., 2010) have proposed a combination of secure hardware and GCs to efficiently realize arbitrary computation while achieving confidentiality, integrity and verifiability. We refer the reader also to (Sadeghi et al., 2010) for a comparison with other potential approaches, e.g. fully homomorphic encryption discussed above.

3.2.4 Private Information Retrieval and Oblivious Transfer

Private information retrieval (PIR) (Chor et al., 1995) enables a CU to query data from the CP, whilst CP learns nothing about what particular data the user has queried. A trivial solution is handing over the entire data to CU, letting CU take out whatever he needs, which is clearly impractical. Assuming that only one copy of the data is available at the CP, one can prove that no better solution exists in the information-theoretic security model. However, when the CP is computationally bounded more efficient so called single-database PIRs can be constructed (Ostrovsky and Skeith, 2007). Alternatively, if one assumes that there are n non-communicating replicas of the data (e.g. stored at different CPs), there also exist solutions which are more efficient than the trivial one (even in the strong information-theoretic sense). We note that the latter assumption seems to be somewhat exotic for practical applications.

Oblivious transfer (OT) (Rabin, 1981) is a stronger version of PIR which in addition guarantees that the CU does not learn any additional information with exception of the exact data item the CU has queried from the CP. In this context, we want to mention a recent work by Camenisch et al. (Camenisch et al., 2009), who present a protocol for anonymous access to a database where different records have different access control permissions based on OT.

3.2.5 Privacy-enhancing Cryptography

This class of protocols contains cryptographic protocols which protect the CU's privacy (anonymity). Regarding the communication channel external adversaries can be locked out by using anonymous communication channels like Tor (Dingledine et al., 2004), which provide anonymity and unlinkability of messages sent from a CU to a CP and vice versa.

Privacy preserving authentication can be achieved by means of several techniques for anonymous authentication (Ateniese et al., 2000; Rivest et al., 2001; Slamanig et al., 2009) or anonymous credentials (Brands, 2000; Camenisch and Lysyanskaya, 2001) which allows a CU to prove that he is a member of a group of CUs and is authorized without revealing the exact identity to the CP respectively. So far, we have not discussed the policies associated with data. In an anonymous setting this can for instance be realized by means of anonymous credentials (Backes et al., 2005) or by using oblivious transfer (Camenisch et al., 2009). In the aforementioned setting we have explicit policies defined for data objects.

Another interesting issue is discussed in (Lu et al., 2010), where CUs can anonymously access resources, but it provides provenance to data ownership and process history of data. This means that a trusted party is able to reconstruct which data was accessed or modified by whom, but the CP is not able to do so. An approach that achieves similar goals was also proposed in (Slamanig and Rass, 2010).

3.2.6 Provable Data Possession

A trivial method to realize a verification whether the entire data of CUs are still stored at the CP, is that CUs compute a message authentication code (MAC) or a digital signature for each data item they store (and store them locally), then retrieve *all* data stored at the CP and recompute as well as verify the MACs or signatures. Obviously, this approach is terribly inefficient. In order to largely reduce the computational as well as bandwidth overhead for the CUs, Juels and Kaliski (Juels and Jr., 2007) and independently Ateniese et al. (Ateniese et al., 2007) have introduced the concept of *proofs of retrievability* (PORs) and *provable data possession* (PDT) respectively. The former approach employs error-correcting codes and is stronger in the sense that there is a guarantee that CUs can retrieve their data, whereas the latter employs so called homomorphic verifiable tags (essentially homomorphic MACs). We refer the reader to (Bowers et al., 2008) for a detailed discussion on PORs and related work and to (Bowers et al., 2009) for a proposal of a real-world concept in context of cloud storage.

3.2.7 Searchable Encryption

Searchable encryption pursuits the idea of enabling to build an encrypted search index (full-text or keyword index) such that the content of the index is hidden from the CP and CUs who are given an appropriate information can perform keyword searches on this index without revealing any information on the keywords to the CP. Hence, the CP can give back pointers to encrypted data that contain a keyword (or keywords) without gaining any knowledge (he may only figure out that some data contain the same but unknown keywords). The concept of searchable encryption using symmetric encryption schemes was introduced in (Song et al., 2000) and intensively studied in (Curtmola et al., 2006). Searchable encryption from public-key encryption schemes was introduced in (Boneh et al., 2004) and efficient solutions are proposed in (Bellare et al., 2007).

4 CONCLUSIONS

From a security and privacy perspective, cloud computing is a most interesting and challenging field of future research. It calls for unifying a broad spectrum of solutions into a highly complex secure system. It appears that many problems arising in cloud computing have been addressed in cryptography. In order to successfully establish cloud computing as a valuable and secure future computing paradigm, researchers from security and cloud computing must join forces to fruitfully, efficiently and effectively exploit the power of cryptography for meeting the needs of cloud providers and users. However, cryptography alone will surely not be able to provide solutions to all cloud related security and privacy issues.

Nevertheless, we hope to draw the attention to some existing cryptographic mechanisms suitable for cloud computing to make future cloud computing more secure and attractive.

REFERENCES

Akinyele, J. A., Lehmann, C. U., Green, M. D., Pagano, M. W., Peterson, Z. N. J., and Rubin, A. D. (2010). Self-Protecting Electronic Medical Records Using Attribute-Based Encryption. Cryptology ePrint Archive, Report 2010/565. http://eprint.iacr.org/.

Ateniese, G., Burns, R. C., Curtmola, R., Herring, J., Kissner, L., Peterson, Z. N. J., and Song, D. X. (2007). Provable Data Possession at Untrusted Stores. In *CCS 2007*, pages 598–609. ACM.

Ateniese, G., Camenisch, J., Joye, M., and Tsudik, G. (2000). A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. In *CRYPTO '00*, volume 1880 of *LNCS*, pages 255–270. Springer.

Backes, M., Camenisch, J., and Sommer, D. (2005). Anonymous Yet Accountable Access Control. In *WPES '05*, pages 40–46. ACM.

Bellare, M., Boldyreva, A., and O'Neill, A. (2007). Deterministic and Efficiently Searchable Encryption. In *CRYPTO 2007*, volume 4622 of *LNCS*, pages 535–552. Springer.

Bethencourt, J., Sahai, A., and Waters, B. (2007). Ciphertext-Policy Attribute-Based Encryption. In *28th IEEE Symposium on Security and Privacy*, pages 321–334. IEEE.

Boneh, D., Crescenzo, G. D., Ostrovsky, R., and Persiano, G. (2004). Public Key Encryption with Keyword Search. In *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 506–522. Springer.

Boneh, D. and Naor, M. (2008). Traitor Tracing with Constant Size Ciphertext. In *CCS 2008*, pages 501–510. ACM.

Bowers, K. D., Juels, A., and Oprea, A. (2008). Proofs of Retrievability: Theory and Implementation. Cryptology ePrint Archive, Report 2008/175. http://eprint.iacr.org/.

Bowers, K. D., Juels, A., and Oprea, A. (2009). HAIL: A High-Availability and Integrity Layer for Cloud Storage. In *CCS '09*, pages 187–198. ACM.

Brands, S. (2000). *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*. MIT Press.

Camenisch, J., Dubovitskaya, M., and Neven, G. (2009). Oblivious Transfer with Access Control. In *CCS '09*, pages 131–140. ACM.

Camenisch, J. and Lysyanskaya, A. (2001). An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation. In *EUROCRYPT '01*, volume 2045 of *LNCS*, pages 93–118. Springer.

Chor, B., Goldreich, O., Kushilevitz, E., and Sudan, M. (1995). Private Information Retrieval. In *FOCS '95*, pages 41–50. IEEE.

Curtmola, R., Garay, J. A., Kamara, S., and Ostrovsky, R. (2006). Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions. In *CCS 2006*, pages 79–88. ACM.

Dingledine, R., Mathewson, N., and Syverson, P. (2004). Tor: The Second-Generation Onion Router. In *13th USENIX Security Symposium*, pages 21–21.

Gennaro, R., Gentry, C., and Parno, B. (2010). Non-interactive Verifiable Computing: Outsourcing Computation to Untrusted Workers. In *CRYPTO 2010*, volume 6223 of *LNCS*, pages 465–482. Springer.

Gentry, C. (2009). Fully Homomorphic Encryption using Ideal Lattices. In *STOC 2009*, pages 169–178. ACM.

Gentry, C. and Halevi, S. (2010). Implementing Gentry's Fully-Homomorphic Encryption Scheme. Cryptology ePrint Archive, Report 2010/520. http://eprint.iacr.org/.

Jensen, M., Schwenk, J., Gruschka, N., and Iacono, L. L. (2009). On Technical Security Issues in Cloud Computing. In *IEEE International Conference on Cloud Computing*, pages 109–116. IEEE.

Jin, H. and Lotspiech, J. (2009). Unifying broadcast encryption and traitor tracing for content protection. In *Annual Computer Security Applications Conference, ACSAC*, pages 139 –148.

Juels, A. and Jr., B. S. K. (2007). PORs: Proofs of Retrievability for Large Files. In *CCS 2007*, pages 584–597. ACM.

Kamara, S. and Lauter, K. (2010). Cryptographic Cloud Storage. In *Financial Cryptography Workshops 2010*, volume 6054 of *LNCS*, pages 136–149. Springer.

Lewko, A. B., Okamoto, T., Sahai, A., Takashima, K., and Waters, B. (2010). Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 62–91. Springer.

Li, M., Yu, S., Ren, K., and Lou, W. (2010). Securing Personal Health Records in Cloud Computing: Patient-centric and Fine-grained Data Access Control in Multi-owner Settings. In *SecureComm 2010*, volume 50 of *LNICST*, pages 89–106. Springer.

Lu, R., Lin, X., Liang, X., and Shen, X. S. (2010). Secure Provenance: The Essential of Bread and Butter of Data Forensics in Cloud Computing. In *ASIACCS 2010*, pages 282–292. ACM.

Muniswamy-Reddy, K.-K., Macko, P., and Seltzer, M. (2010). Provenance for the Cloud. In *FAST 2010*, pages 197–210. USENIX.

Ostrovsky, R. and Skeith, W. E. (2007). A Survey of Single-Database Private Information Retrieval: Techniques and Applications. In *PKC 2007*, volume 4450 of *LNCS*, pages 393–411. Springer.

Rabin, M. (1981). How to Exchange Secrets by Oblivious Transfer. Tr-81, Aiken Comp. Lab., Harvard University.

Rass, S. and Schartner, P. (2010). A unified framework for the analysis of availability, reliability and security, with applications to quantum networks. *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews*, 40(5):107–119.

Rivest, R. L., Shamir, A., and Tauman, Y. (2001). How to Leak a Secret. In *ASIACRYPT '01*, volume 2248 of *LNCS*, pages 552–565. Springer.

Sadeghi, A.-R., Schneider, T., and Winandy, M. (2010). Token-Based Cloud Computing – Secure Outsourcing of Data and Arbitrary Computations with Lower Latency. In *Workshop on Trust in the Cloud*, volume 6101 of *LNCS*, pages 417–429. Springer.

Slamanig, D. and Rass, S. (2010). Anonymous But Authorized Transactions Supporting Selective Traceability. In *SECRYPT '10*, pages 132–141.

Slamanig, D., Schartner, P., and Stingl, C. (2009). Practical Traceable Anonymous Identification. In *SECRYPT '09*, pages 225–232.

Song, D. X., Wagner, D., and Perrig, A. (2000). Practical Techniques for Searches on Encrypted Data. In *21st IEEE Symposium on Security and Privacy*, pages 44–55. IEEE.

Stehlé, D. and Steinfeld, R. (2010). Faster Fully Homomorphic Encryption. In *ASIACRYPT 2010*, LNCS. Springer.

van Dijk, M. and Juels, A. (2010). On the Impossibility of Cryptography Alone for Privacy-Preserving Cloud Computing. In *HotSec '10*. USENIX Association.

Yao, A. C.-C. (1986). How to Generate and Exchange Secrets (Extended Abstract). In *FOCS '86*, pages 162–167. IEEE.