Cryptography in the Cloud: Advances and Challenges

Colin Boyd*, *Member, KIICE*

School of Electrical Engineering and Computer Science, Queensland University of Technology, Brisbane, Australia

Abstract

Cloud computing is a currently developing revolution in information technology that is disturbing the way that individuals and corporate entities operate while enabling new distributed services that have not existed before. At the foundation of cloud computing is the broader concept of converged infrastructure and shared services. Security is often said to be a major concern of users considering migration to cloud computing. This article examines some of these security concerns and surveys recent research efforts in cryptography to provide new technical mechanisms suitable for the new scenarios of cloud computing. We consider techniques such as homomorphic encryption, searchable encryption, proofs of storage, and proofs of location. These techniques allow cloud computing users to benefit from cloud server processing capabilities while keeping their data encrypted; and to check independently the integrity and location of their data. Overall we are interested in how users may be able to maintain and verify their own security without having to rely on the trust of the cloud provider.

Index Terms: Cloud computing, Cryptography, Information security

I. INTRODUCTION

The idea of cloud computing is today well known and the term is widely used in popular, as well as technical, media. However, a precise definition of cloud computing can be difficult to pin down. The United States National Institute of Standards and Technology (NIST) provides a helpful short document [1] summarizing its own definition of the term. According to NIST there are five essential characteristics of cloud computing: on-demand self-service, broad network access, resource pooling, elasticity, and measured service. Each of these characteristics provides opportunities for more efficient and flexible computing resources to the benefit of a wide range of consumers. At the same time each characteristic also brings some challenging security implications.

While security has often been recognized as a challenge for cloud computing, it should also be acknowledged that for some consumers it can be an added benefit of moving to

the cloud. Individuals and small businesses can shed the burden of basic good security practices such as on-time patching of operating systems and applications, backing up of critical data, and malware checking; instead they can rely on the cloud provider to deliver these services on their behalf with consequent efficiency savings due to becoming part of a large-scale operation. This scenario makes sense when the consumer trusts the cloud provider to look after the security of their data. However, sometimes users of cloud services may not wish to, or may not be able to, place full trust in their provider. In this situation users need to implement and manage the security of their own data in such a way that they can still benefit from the services that the cloud provides. This tension between the desirability of outsourcing user data to a third party, while at the same time maintaining control of the security of that data, is the source of much of the challenge of security in cloud computing. Chow et al. [2] have explored this tension in some detail.

Received 30 October 2012, Revised 18 December 2012, Accepted 10 January 2013

*Corresponding Author Colin Boyd (E-mail: c.boyd@qut.edu.au, Tel: +61-7-31389549)

School of Electrical Engineering and Computer Science, Queensland University of Technology, Brisbane, Australia

Open Access <http://dx.doi.org/10.6109/jicce.2013.11.1.017>

print ISSN:2234-5973 online ISSN:0000-0000

© This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/by-nc/3.0/>) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

Copyright © The Korea Institute of Information and Communication Engineering

Cryptography is now a mainstream technology used to maintain the confidentiality and integrity of data while being communicated or stored in electronic form. Standardized algorithms for encryption, message authentication, and digital signatures are routinely applied to ensure that data is only readable by authorized recipients, to check that data has not been altered in any way, and to ensure that originators can be held accountable for digitized statements. Such algorithms can be applied to data to be sent to the cloud but they do not necessarily provide the security properties required when the data is intended for storage and processing on a remote site, especially when the consumer does not completely trust the provider. The fundamental issue is which parties control the keys required to decrypt the data and to compute the integrity checks. If the keys are known to the cloud provider then the consumer needs to trust the cloud provider not to violate the data confidentiality and to correctly compute the integrity checks on the correct data. On the other hand, if the consumer keeps the keys from the cloud provider this returns management burden to the consumer and will, in general, prevent processing of the data by the cloud provider. We will explore ways to address this conflict later in this article. Throughout this article we will generally use the term consumer to denote the client of the cloud computing service. This may be an individual or a corporate client. We will also use the term provider to denote the cloud computing server or company which provides the service.

In the next section we review the security requirement for cloud computing. Section III reviews cryptographic measures to allow users to obtain assurances regarding how their data is stored without trusting the cloud provider. Section IV reviews cryptographic mechanisms for processing data while still encrypted. Finally, Section V discusses the implications of the current status and speculates on future developments.

II. SYSTEM SECURITY REQUIREMENTS FOR CLOUD COMPUTING

Introductory texts on information security often emphasize three fundamental concerns when it comes to information security: confidentiality, integrity and availability. These security services are intended to ensure that data is available only to the correct parties, that it is correct, and that it can be accessed when needed by authorized parties. However, stating these requirements without clarifying where the threat originates does not allow a useful implementation of security mechanisms. In the cloud computing environment new security threats arise due to the changed architecture of data processing. We now explore these threats by returning to the fundamental characteristics

of cloud computing defined by NIST [3].

On-demand self-service: This property allows consumers to utilize storage and processing on the provider's site when required. This means that user data will be stored and processed on the provider's servers and processors. If the data has commercial value (such as business plans or contracts) then the consumer needs to perform a risk assessment about the possibility that the data is compromised or altered (including being lost). If the data includes personal information such as medical records or banking details, legislation will often apply. Laws such as the European Union (EU) Data Protection Directive oblige the data owner to protect the data adequately. If the provider is not completely trusted to act properly, it will mean that the consumer will probably want the data to be protected cryptographically to prevent it being read or altered by the provider. However, data processing may not be possible if the data is encrypted unless special kinds of cryptography are used (see Section IV).

Broad network access: This property allows consumers to access cloud services from a variety of platforms including portable computers or mobile phones. Remote access requires authentication of the user to the provider, and preferably from provider to user too, but the flexible access requirements may prevent certain security mechanisms from being applied. Use of the Internet or wireless networks for such access opens the possibility of eavesdropping of the communications between the consumer and the provider. This may require the user data to be encrypted, either on the communications link or more generally at any time when it has left the consumer's premises.

Resource pooling: This property describes the sharing of provider resources by different consumers, often called a multi-tenant model. This is one important way that providers can achieve efficiency and is related to the characteristics of on-demand service and elasticity. Different users may be sharing physical and virtual resources. Even though mechanisms may be in place to ensure that consumers have access only to their authorized resources, one can compare the situation to having a rival or a hacker using a different part of your own desktop computer and see that there is significant potential for data breaches. This property again suggests the need for encryption of the user data.

Another aspect of resource pooling is that in general consumers do not control the physical location of the servers on which their data is stored or processed. This can have implications for the risk assessment of business data if there is potential for data to be stored in unfriendly states.

Furthermore, data privacy legislation often restricts moving of personal data offshore. For example, the EU Data Protection Directive restricts the movement of personal data outside EU countries. Some cloud providers do provide assurance regarding where their servers are placed and provide premium services restricting the server location to specified areas. However, this again requires trust in the provider which may not even be possible if the consumer is bound by legislation.

Elasticity: This property allows consumers to rapidly expand or contract their usage of cloud resources. Due to the scale of cloud services this can provide the appearance of infinite resources to the consumer which is an attractive reason to move to cloud computing. Moreover, consumers can rapidly scale down their requirements when the need is no longer there. This flexibility can be achieved by sharing resources, leading to the security issues associated with resource pooling discussed above.

Measured service: This property allows consumers to pay only for the cloud resources that they use. Payment can be determined in advance by contractual agreements. In case of dispute between consumer and provider, non-repudiation services may be required so that a third party can reliably judge whether resources were genuinely requested and provided.

Table 1 summarizes the security threats that we have identified above. We can see that encryption of user data is a useful way to achieve data confidentiality and this can be supplemented with integrity checks or digital signatures to prevent modification of data. However, if users are to remain independent from providers in ensuring their confidentiality and integrity they must use special techniques if they want to allow their data to be processed by the provider. These techniques are the subject of Section IV. When it comes to availability, consumers may require proofs of where and how the data is stored which can be independently verified. We look at how this can be done next.

Table 1. Threats to consumers in cloud computing

Threat	Threat agent	Mitigation steps
Eavesdropping	Provider Other cloud tenants Network hackers	Encryption
Data modification	Provider Other cloud tenants Network hackers	Integrity checks Digital signatures
Availability	Provider	Redundancy Proofs of storage
Data movement	Provider	Proofs of location

III. ASSURANCES FOR REMOTELY STORED DATA

Consumers who want to outsource their IT provisioning into the cloud will typically want to place a large amount of data onto cloud servers. This may be databases of company information, medical records of large hospitals, or an individual's lifelong photography collection. Some of this data may not be used for extended periods but its availability when needed may be critical. Whether or not the consumer wants to process this data, they may want to check from time to time that the data is still available. One way to do this is to retrieve a copy of the data from the cloud and check it by using a checksum based on a cryptographic hash function (for which it is computationally hard to find a different input with the same hash output). However, this will be very inconvenient if the data is large. Cryptographers noticed that there are ways to achieve assurance of the continuing availability of data by probabilistic checking to examine only a small fraction of the data combined with cryptographic processing.

A. Proofs of Storage and Proofs of Retrievability

The application of proofs of retrievability in cloud computing seems to have been first recognized by Juels and Kaliski [4] although they point to an extensive earlier literature on checking for integrity of files in applications such as cryptographic file systems. Their idea was to first encrypt the data and then insert blocks of data called sentinels into the encrypted data at random points - the sentinels are generated using a cryptographic function whose values cannot be distinguished from the encrypted data without knowing the function inputs. This prepared file is then sent to the cloud server for storage. The provider only sees a file of random bits and cannot tell which parts of the data are the encrypted original and which parts are the inserted randomness. The data owner who wants to check if the data is intact can ask for a random selection of the sentinels to be returned by specifying their positions and can verify whether the sentinels are correctly returned by the server. Since the provider does not know where the sentinels are, any change in the data has a chance of being detected.

Juels and Kaliski's method [4] includes a fundamental observation that was subsequently used in the last later methods. This is that the basic method only needs to be able to detect large changes in the data. By applying an error correcting code to the original data, small changes in the data can be detected and corrected. If the error correcting code can correct a certain fraction of bits, then the sentinels need to be numerous enough to detect corruptions of greater

than that fraction of bits. Juels and Kaliski [4] provide sample parameter sizes which require a 2% overhead to store the sentinels and a further 14% for the error correction overhead. With these parameters an adversary who does not store the correct file can answer 12 challenges with probability only 10^{-6} while the system remains good for up to 1,000 challenges.

Despite its innovative ideas, the scheme of Juels and Kaliski [4] has some significant drawbacks. One is that it can only be used a limited number of times since the server learns about where the sentinels are stored from the queries that are made. Another is that it requires that the user data is encrypted so that it cannot be distinguished from the sentinels. Not all applications wish to encrypt all their data. Therefore a number of alternative proof schemes have been proposed subsequently. It is worth noting that some schemes use the term proof of storage in contrast to proof of retrievability. The distinction is a technical one—the latter type of proof can in principle be used to recover the file while the former only shows that it is possible to retrieve it. In practice both types of proof use similar techniques and it is arguable whether their practical utility is any different.

Shacham and Waters [5] provided improvements to the scheme of Juels and Kaliski [4]. They replaced the sentinels with cryptographic functions based on message authentication codes (MACs) or short digital signatures. In response to a verifier challenge, the server needs to send a proof which is an aggregated MAC or signature designed to prove that all the blocks specified in the challenge have been accessed. In the case of the MAC-based version the client is able to verify the proof using its secret MAC key while the signature-based variant has the added advantage of being publicly verifiable but has greater computational cost.

A number of enhanced proofs of storage schemes have been proposed. One specific feature that may be very useful is to allow clients to update their stored data without the need to completely reset the scheme which has led to a number of so-called dynamic proof of storage schemes [6, 7].

B. Proofs of Location

We already pointed out that where a consumer's data is located may be an important security issue. Using conventional cryptographic methods it can seem an impossible task to provide assurance as to data location. By its very nature, cloud computing is intended to provide flexibility as to which physical machine is used to provide user resources, in line with the characteristics of resource pooling and elasticity. Indeed, it may even be the case that data is split across multiple servers in order to provide fault tolerance. However, when a provider guarantees that data will be

stored within some specified geographic area it may be possible for consumers to obtain some assurance about that. The basic idea for achieving such assurance is to measure how long it takes for data to be returned to the user from a claimed location. The further away that the data is stored, the longer it will take to transmit. Of course such measurements need to be conducted with a high degree of accuracy since data moves very fast in electronic communications. In fact this idea has been around for some time in different situations.

1) Distance Bounding Protocols

The notion of distance bounding as a security protocol goes back to Brands and Chaum [8] twenty years ago. Their motivation was to prevent so-called mafia fraud on authentication protocols in which an adversary relays authentication information from a distant honest user to convince a local verifier that the honest user is locally present. This may allow, for example, the (mafia) adversary to conduct an expensive purchase while the honest user is making a small value purchase elsewhere.

Distance bounding protocols always entail a timing phase in which the verifier issues a challenge and the prover responds. The prover is required to respond within a time limit dependent on the distance between the prover and the verifier. The cryptographic aspect is required to ensure that the responses did come from the intended verifier rather than from a nearby adversary masquerading as the verifier. In the original protocol of Brands and Chaum [8] the prover and verifier exchange random bits during the timing phase and afterwards this exchange is digitally signed by the verifier.

The distances involved to prevent mafia fraud can be expected to be relatively small, of the order of a few meters, and high accuracy timing is essential. In the cloud computing context the distances may be much larger. Typically location verification may only be required to the accuracy of the country, or even the continent, in which the data is located. This makes the problem potentially easier but at the same time there are a number of uncertainties that may apply that may make the measurement correspondingly harder. For example, when verifying stored data there will be some delays incurred while the server accesses the disk and sector required to read the required file blocks. In addition, the delay incurred in communicating the data on the Internet needs to be estimated.

2) Cloud Geolocation

A few authors have designed protocols for proving location of files in the cloud using distance bounding ideas [9-11]. One thing to emphasize is that there is nothing to stop a cloud provider keeping data in two places, one of which is in the agreed location and the

other in a forbidden location. In this sense the task of restricting where the provider puts that data is impossible, as pointed out by Watson et al. [9]. It is therefore necessary to make some assumptions about how a real-world cloud provider will act. The basic idea is to assume that the provider is *economically rational*. This assumption in fact is used in other types of proof too [12]. The assumption is based on the expectation that a commercial entity, such as a cloud provider, will not take actions that harm its efficient operation. An economically rational provider who has to prove that certain files are stored in a specific location will not store extra copies of the same file in other locations since that would simply result in a cost without benefit.

To minimize the uncertain delays in measurement, protocol designers have made various assumptions. Watson et al. [9] assume the existence of a trusted landmark infrastructure which consists of nodes of known location which will be used as verifiers. Benson et al. [10] suggest that it may be possible to store the verifier in a virtual machine inside the data center in which the stored data files are claimed to reside. Albeshri et al. [11] assume the existence of a trusted device located close to the server and with global positioning system capability.

C. Proofs of Redundancy

Another property which may be of interest to cloud consumers is assurance that backups have been performed properly. Again, it may at first seem that such properties are impossible to verify. However, timing techniques can again provide a way to achieve this. Bowers et al. [13] designed a protocol to show that data is stored on more than one disk, in other words that there is a backup copy. This can be done by timing how long it takes to obtain random parts of the data. This is because the physical limitations on disc storage allow faster recovery of multiple random portions of a file if there are two copies on separate disks, much in the manner of how redundant array of independent disks (RAID) storage works.

The protocol of Bowers et al. [13] only applies if the backup copy is stored in the same data center. This may be a reasonable assumption in practice and provides fault tolerance against the failure of one disk. However, it does not provide the most robust type of backup. Failures due to natural or man-made disasters are likely to impact the whole data center and in such cases both copies on separate discs are likely to be lost simultaneously. An interesting extension of distance bounding is therefore to use it to prove geographic separation of multiple copies in order to provide assurance of more robust backup.

IV. PROCESSING ENCRYPTED DATA

We now turn to the complementary issue of how cloud consumers can encrypt their data in such a way that the provider can perform computational tasks without the need to decrypt the data. These techniques can be used together with the techniques in the previous section.

A. Searchable Encryption

One of the simplest ways that the provider can help the consumer is by finding data amongst the consumer's stored data. Remember that the stored data may be huge so that returning all the data to the consumer for decryption and searching is highly undesirable. Therefore we need methods to allow searching through encrypted data.

Searchable encryption has been of interest since before cloud computing became popular because there have earlier been applications such as searching through encrypted mail stored on a mail server. However, there have been many new and improved schemes proposed since its importance for cloud security was recognized.

There are two flavors of searchable encryption. Symmetric searchable encryption [14] uses only one key so the generator of the encrypted data is the same as the user who wants to retrieve the encrypted file. In contrast public key searchable encryption [15] allows any user who knows the public key to add to the encrypted data which the owner of the corresponding private key can decrypt. When the server is asked to search for a particular keyword it must be given a token which allows it to identify any encrypted files which contain that keyword. Note that, at the minimum, this allows the server to learn which files contain identical keywords, which seems to be inevitable for any system. However, both types of searchable encryption schemes prevent the server from knowing anything more about what is in the files. Kamara and Lauter [16] provide a nice overview of recent searchable encryption techniques in the context of cloud computing.

B. Homomorphic Encryption

A *homomorphism* is a mathematical term for a mapping between two structures that preserves some operation. A homomorphic encryption scheme $E(.)$ typically preserves either addition or multiplication so that for any two messages m_1 and m_2 we have $E(m_1)+E(m_2) = E(m_1+m_2)$ or $E(m_1) \times E(m_2) = E(m_1 \times m_2)$. Thus if a user wants the server to add together a number of encrypted values, the server can simply add the ciphertexts if the encryption scheme is additively homomorphic. More generally, a scheme is said to be *fully homomorphic* if any computation can be completed without the need to decrypt the data.

Table 2. Basic RSA is homomorphic

Let $n = pq$ where p and q are large prime numbers (at least 1024 bits each)
Choose a small public exponent e and compute $d = e^{-1} \bmod (p-1)(q-1)$.
Private key: d, n
Public key: e, n
Encrypt message m : $E(m) = c = m^e \bmod n$
Decrypt ciphertext c : $D(c) = m = c^d \bmod n$
Homomorphic property: $E(m_1) \times E(m_2) = m_1^e \times m_2^e \bmod n$ $= (m_1 \times m_2)^e \bmod n$ $= E(m_1 \times m_2)$

Basic forms of homomorphic encryption have been known for many years. For example, the widely known RSA encryption scheme is multiplicatively homomorphic as illustrated in Table 2. It is worth noting that basic RSA as shown in this figure is well known not to provide strong security. Indeed the highest levels of encryption security cannot be achieved for homomorphic encryption since by definition they are *malleable*. This means that an adversary can find new valid ciphertexts from existing ciphertexts. While strong encryption schemes are required not to be malleable, this property is exactly what homomorphic encryption is designed to achieve. Thus in order to solve the problem of allowing the provider to process the consumer's encrypted data a compromise in terms of security is necessary. Another property to note about the RSA example is that it is not additively homomorphic.

Once again, the advent of cloud computing has provided a major incentive to find a fully homomorphic encryption scheme and this was achieved in 2009 by Gentry [3]. Unfortunately the original scheme of Gentry [3] was extremely inefficient by usual cryptographic standards, with encryption and decryption times taking many minutes and public key sizes measured in gigabytes. There has been a large international effort to develop more efficient fully homomorphic encryption schemes and today significantly more efficient schemes are known even though they remain far more costly than current popular encryption algorithms [17].

C. Homomorphic Authentication

For many kinds of data it is essential for consumers to be able to authenticate their data. If a cloud server is then asked to process this data the consumer will want to be able to check that the output is still authentic. This may or may not be required, independently of whether the data is encrypted. Ahn et al. [18] have developed a framework for computing with authenticated data using homomorphic digital signature schemes.

V. CONCLUSIONS

In this article we have focused mainly on consumer security requirements. Providers also have their own security needs, including the ability to keep accurate and verifiable records of resource usage by consumers.

The advent of cloud computing has led to a number of interesting security challenges. Cryptographers have risen to the challenge and developed a number of novel mechanisms which can potentially be used to provide security guarantees without the need to trust the cloud provider is acting correctly. Some of these mechanisms have already been implemented in prototype systems but it remains to be seen whether they will become mainstream services available in commercial systems. History has shown that users are often unwilling to pay for security unless strong economic or legislative motivations are in place.

It seems certain that new and improved mechanisms for cloud security are yet to be found. Fully homomorphic encryption is a hot research topic in the cryptographic research community and efforts to improve its efficiency or find more practical variants continue apace. Other challenging problems and more exotic schemes are sure to emerge, even if they may not see commercial deployment.

Consider the problem of checking whether a remotely stored file is stored in encrypted form. This seems a hard problem since the server could encrypt any file blocks in real time without any significant delay. This means that the server can respond to any challenge to display the encrypted version of the file even when the file is stored in plaintext. Recently van Dijk et al. [16] have found a solution to this problem by requiring the server to store the encrypted file in a transformed manner which takes significant time for the server to compute. Of course this does not stop the server from storing an additional copy of the file in plaintext but van Dijk et al. point out that this makes no sense for an economically rational provider. If problems like this can be founded realistic solutions, maybe other seemingly impossible requirements, such as verifying that files have been properly erased, can also be solved.

REFERENCES

- [1] P. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology, Gaithersburg, MD, Special Publication 800-145, 2011.
- [2] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina, "Controlling data in the cloud: outsourcing computation without outsourcing control," in *Proceedings of the 1st ACM Cloud Computing Security Workshop*, Chicago: IL, pp. 85-90, 2009.

- [3] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, Bethesda: MD, pp. 169-178, 2009.
- [4] A. Juels and B. S. Kaliski Jr, "PORS: proofs of retrievability for large files," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, Alexandria: VA, pp. 584-597, 2007.
- [5] H. Shacham and B. Waters, "Compact proofs of retrievability," *Advances in Cryptology - ASIACRYPT 2008, Lecture Notes in Computer Science*, vol. 5350, pp. 90-107, 2008.
- [6] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, Chicago: IL, pp. 213-222, 2009.
- [7] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," *Computer Security - ESORICS 2009, Lecture Notes in Computer Science*, vol. 5789, pp. 355-370, 2009.
- [8] S. Brands and D. Chaum, "Distance-bounding protocols," *Advances in Cryptology - EUROCRYPT 1993, Lecture Notes in Computer Science*, vol. 765, pp. 344-359, 1993.
- [9] G. J. Watson, R. Safavi-Naini, M. Alimomeni, M. E. Locasto, and S. Narayan, "LoST: location based storage," in *Proceedings of the 4th ACM Cloud Computing Security Workshop*, Raleigh: NC, pp. 59-69, 2012.
- [10] K. Benson, R. Dowsley, and H. Shacham, "Do you know where your cloud files are?," in *Proceedings of the 3rd ACM Cloud Computing Security Workshop*, Chicago: IL, pp. 73-82, 2011.
- [11] A. Albeshri, C. Boyd, and J. M. G. Nieto, "GeoProof: proofs of geographic location for cloud computing environment," in *Proceeding of the 32nd International Conference on Distributed Computing Systems Workshops*, Macau, pp. 506-514, 2012.
- [12] M. van Dijk, A. Juels, A. Oprea, R. L. Rivest, E. Stefanov, and N. Triandopoulos, "Hourglass schemes: how to prove that cloud files are encrypted," in *Proceedings of the ACM Conference on Computer and Communications Security*, Raleigh: NC, pp. 265-280, 2012.
- [13] K. D. Bowers, M. van Dijk, A. Juels, A. Oprea, and R. L. Rivest, "How to tell if your cloud files are vulnerable to drive crashes," in *Proceedings of the 18th ACM Conference on Computer and Communications Security*, Chicago: IL, pp. 501-514, 2011.
- [14] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceedings of the IEEE Symposium on Security and Privacy*, Berkeley: CA, pp. 44-55, 2000.
- [15] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," *Advances in Cryptology - EUROCRYPT 2004, Lecture Notes in Computer Science*, vol. 3027, pp. 506-522, 2004.
- [16] S. Kamara and K. Lauter, "Cryptographic cloud storage," in *Proceedings of the 14th International Conference on Financial Cryptography and Data Security*, Tenerife, Canary Islands, pp. 136-149, 2010.
- [17] V. Vaikuntanathan, "Computing blindfolded: new developments in fully homomorphic encryption," in *Proceedings of the 52nd IEEE Annual Symposium on Foundations of Computer Science*, Palm Springs: CA, pp. 5-16, 2011.
- [18] J. H. Ahn, D. Boneh, J. Camenisch, S. Hohenberger, A. Shelat, and B. Waters, "Computing on authenticated data," *Theory of Cryptography, Lecture Notes in Computer Science*, vol. 2779, pp. 1-20, 2012.

Colin Boyd

was born in the UK and studied mathematics at the University of Warwick where he was awarded the B.Sc. in 1981 and the Ph.D. in 1985. After five years working for British Telecom Research Labs he took up an academic position at University of Manchester in 1989. In 1995 he emigrated to Australia where he joined Queensland University of Technology and later became full professor in 2005. He has held positions including Director of Research in QUT's Information Security Institute and currently leads the Secure and Resilient Infrastructure theme in the Institute for Future Environments. His research interests are in cryptography and cryptographic protocols. He is the author of over 150 peer reviewed publications and his work has been cited over 6,000 times.