Study on cryptographic protocols

November, 2014

European Union Agency for Network and Information Security www.enisa.europa.eu

About ENISA

Authors

This work was commissioned by ENISA under contract ENISA D-COD-14-TO9 (under F-COD-13-C23) to the consortium formed by K.U.Leuven (BE) and University of Bristol (UK).

- Contributors: Nigel P. Smart (University of Bristol), Vincent Rijmen (KU Leuven), Martijn Stam (University of Bristol), Bogdan Warinschi (University of Bristol), Gaven Watson (University of Bristol).
- Editor: Nigel P. Smart (University of Bristol).
- ENISA Project Manager: Rodica Tirtea.

Agreements of Acknowledgements

We would like to extend our gratitude to:

- External Reviewers: Michel Abdalla (ENS Paris), Kenneth G. Paterson (Royal Holloway, University of London), Ahmad-Reza Sadeghi (T.U. Darmstadt), Michael Ward (Mastercard) for their comments suggestions and feedback.
- We also thank a number of people for providing anonymous input and Cas Cremers (Oxford University) and Hugo Krawczyk (IBM) for detailed comments on various aspects.

Contact

For contacting the authors please use sta@enisa.europa.eu.

For media enquires about this paper, please use press@enisa.europa.eu.

Catalogue number TP-06-14-085-EN-N   ISBN 978-92-9204-103-8   DOI 10.2824/3739

Executive summary

Cryptographic algorithms, when used in networks, are used within a cryptographic protocol. In the ENISA algorithms report of 2013 [113], several protocols were discussed. In this document (which is the sister document of the 2014 report [115]) we extend the work in the 2013 report to cover more categories of protocols.

The focus of this report is on decision makers in corporations and governments making decisions as to what protocols to use so as to protect online communications which contain personal data which needs to be kept private. Even if the cryptographic primitives and schemes (discussed in [115]) are deemed secure, their use within a protocol can result in a vulnerability which exposes the supposedly secured data.

Another focus of the report is to point out the paucity of work, which demonstrates that standard protocols meet well defined security goals. Thus the report, hopefully, will act as a stimulus to researchers and funders in this area to focus efforts on the important area of cryptographic protocols. Whilst the security of basic cryptographic building blocks, such as primitives and protocols, is well studied and understood, the same cannot be said of cryptographic protocols. The scientific study of such protocols can be said to be still not mature enough.

As an example of this infancy we point to the discussion in this document on TLS (Transport Layer Security) security, Section 3.1. The TLS protocol is the protocol used to secure traffic from web-sites to browsers; despite a lot of effort on understanding this protocol in the last few years, basic protocol errors are still being found (e.g. Lucky13 [17]), as well as implementation errors (e.g. HeartBleed [87]). In this report we focus on the former type of problems as opposed to the latter type of problems.

Guidelines

The following guidelines are addressed to researchers in the field:

1. Many cryptographic and security protocols have in the past been designed by networking and protocols experts, and not by cryptographic protocol experts. The design of cryptographic protocols is particularly challenging, and the associated security proofs to guarantee correctness are vastly more complicated than those for cryptographic schemes. Researchers need to work on simplifying the analysis, and enabling automated tools to provide strong computational guarantees using computational soundness results.
2. More attention needs to be paid to automated verification that an implementation of a protocol actually meets a given security goal. This is distinct from showing that an abstract protocol meets a given security goal described above. This problem is particularly challenging, compared to normal automatic code verification, due to the interactive nature of protocols. Thus researchers need to examine how such automated tools can guarantee correct implementation of a protocol design.
3. Designers and implementers should refrain from “optimising” well studied protocols to achieve some specific application need; unless they are prepared to revisit and re-evaluate the above security proofs. Small insignificant changes in protocols can result in invalidating the guarantees of such proofs.

The key problem with protocols today is that many result from cryptographic design many years (even decades) ago. Thus cryptographic protocols suffer more from legacy issues than the underlying cryptographic components. The goal should be to work towards a better cryptographic protocol infrastructure which does not exhibit such problems. Thus we provide the following guidelines to organisations which are developing new protocols:

1. Future protocols should be designed using solid and well-established engineering principles, but also with ease of formal security analysis in mind, and ideally in conjunction with the development of formal security proofs. They should also be designed in the light of the state of the art in cryptanalysis of their constituent primitives. The many issues that have arisen in real world protocols in recent years highlight the inadequacies of the ad hoc design approach, albeit it supported by well-established "rules of thumb"
2. Future protocols should not be any more complex than they need to be (eliminate extraneous options which only invite attacks - e.g. renegotiation attacks, and Triple Handshake attacks on TLS; the high number of key exchange options in IPSec).
3. Carefully designed protocols will exhibit algorithm agility and secure version negotiation to support future development (but not at the expense of simplicity). Thus, lock in of cryptographic protocols and schemes for many years should be avoided. It should be relatively easy to upgrade cryptographic components, by designing protocols in a modular manner. Enabling components which are no longer deemed secure to be swapped out.
4. More work needs to be performed on verifying Application Programming Interfaces (APIs) for application protocols. Both in terms of understanding their security properties and verifying that the API meets such properties. For example a secure channel API should be secure as long as the underlying cryptographic components are secure; no security weakness should be introduced by the API itself.

Acronyms

3DES Triple DES
3GPP 3rd Generation Partnership Project (mobile phone system)
A5/X Stream ciphers used in mobile phone protocols
ABE Attribute Based Encryption
AE Authenticated Encryption
AEAD Authenticated Encryption with Auxiliary Data
AES Advanced Encryption Standard
AESKW A Key Wrap Scheme
AH Authentication Header
AKA Authentication and Key Agreement
AKW1 A Key Wrap Scheme
AKW2 A Key Wrap Scheme
AMAC ANSI Retail MAC
ANSI American National Standards Institute
BB Boneh–Boyen (ID based encryption)
BF Boneh–Franklin (ID based encryption)
BPP Binary Packet Protocol
BPR Bellare, Pointcheval and Rogaway
BMP Boyko, MacKenzie and Patel
CBC Cipher Block Chaining (mode)
CCA Chosen Ciphertext Attack
CCM Counter with CBC-MAC (mode)
CFB Cipher Feedback
CMA Chosen Message Attack
CMAC Cipher-based MAC
CPA Chosen Plaintext Attack
CTR Counter (mode)
CVA Ciphertext Validity Attack
CWC Carter–Wegman + Counter
DAA Direct Anonymous Attestation
DEM Data Encapsulation Mechanism
DES Data Encryption Standard
DLP Discrete Logarithm Problem
DSA Digital Signature Algorithm
E0 A stream cipher used in Bluetooth
EAX Actually stands for nothing (mode)
EC2 Elastic Computing Cloud
ECB Electronic Code Book (mode)
ECC Elliptic Curve Cryptography
ECDLP Elliptic Curve Discrete Logarithm Problem
ECIES Elliptic Curve Integrated Encryption Scheme
EEA EPS Encryption Algorithm
EIA EPS Integrity Algorithm EKE Encrypted Key Exchange
EMAC Encrypted CBC-MAC
EME ECB-mask-ECB (mode)
EMV Europay–Mastercard–Visa (chip-and-pin system)
ENISA European Union Agency for Network and Information Security
ESP Encapsulating Security Payload
FDH Full Domain Hash
FHE Fully Homomorphic Encryption
GCM Galois Counter Mode
GDSA German Digital Signature Algorithm
GMAC The MAC part of the GCM block cipher mode
GSM Groupe Special Mobile (mobile phone system)
HMAC A hash based MAC algorithm
IAPM Integrity Aware Parallelizable Mode
IEC International Electrotechnical Commission
IEEE Institute of Electrical and Electronics Engineers
IETF Internet Engineering Task Force
IKE Internet Key Exchange
IND Indistinguishability of Encryptions
INT-CTXT Integrity of Ciphertexts
IPsec Internet Protocol Security
ISO International Standards Organization
IV Initialisation Vector (or Value)
J-PAKE Password Authenticated Key Exchange by Juggling

KDSA	Korean Digital Signature Algorithm
KDF	Key Derivation Function
KEM	Key Encapsulation Mechanism
KW	AES Key Wrap
KWP	AES Key Wrap with Padding
LAMP	Linux, Apache, MySQL, PHP
LTE	Long Term Evolution (mobile phone system)
LWE	Learning With Errors
MAC	Message Authentication Code
MOV	Menezes–Okamoto–Vanstone (attack)
MPC	Multi Party Computation
MQV	Menezes–Qu–Vanstone (protocol)
MS	Mobile Station (i.e. a phone in UMTS/LTE)
NIST	National Institute of Standards and Technology (US)
NMAC	Nested MAC
NTRU	A Post Quantum Encryption Algorithm
OAEP	Optimal Asymmetric Encryption Padding
OCB	Offset Code Book (mode)
OFB	Output Feedback (mode)
OPE	Order Preserving Encryption
ORAM	Oblivious Random Access Memory
PACE	Password Authenticated Connection Establishment
PAK	Password Authenticated Key (PAK) Exchange
PAKE	Password Authenticated Key Exchange
PEKS	Public Key Encryption Keyword Search
PKCS	Public Key Cryptography Standards
PKE	Public Key Encryption
POR	Proofs Of Retrievability
PRF	Pseudo Random Function
PRNG	Pseudo Random Number Generator
PRP	Pseudo Random Permutation
PSEC	Provable Secure Elliptic Curve (encryption)
PSS	Probabilistic Signature Scheme
PV	Pointcheval–Vaudenay (signatures)
RC4	Ron’s Cipher Four (a stream cipher)
RDSA	Russian Digital Signature Algorithm
RFC	Request For Comments
RAM	Random Access Memory
ROM	Random Oracle Model
RSA	Rivest–Shamir–Adleman
SA	Security Association
SHA	Secure Hash Algorithm
SIMD	Single Instruction Multiple Data
SIV	Synthetic Initialization Vector
SK	Sakai–Kasahara (ID-based encryption)
SN	Serving Network (i.e. a provider in UMTS/LTE)
SPAKE	Single-Party Public-Key Authenticated Key Exchange
SPD	Security Policy Database
SQL	Structured Query Language
SSE	Symmetric Searchable Encryption
SSH	Secure Shell
SSL	Secure Sockets Layer
TKW	Triple-DEA Key Wrap
TDKW	A Key Wrap Scheme
TRNG	True Random Number Generator
UEA	UMTS Encryption Algorithm
UF	Universally Unforgeable
UIA	UMTS Integrity Algorithm
UMAC	Universal hashing based MAC
UMTS	Universal Mobile Telecommunications System
VPN	Virtual Private Network
WEP	Wired Equivalent Privacy
WPA	Wi-Fi Protected Access
XTS	XEX Tweakable Block Cipher with Ciphertext Stealing

Table of Contents

1 Introduction

A cryptographic protocol is a procedure carried out between two parties which is used to perform some security task. Typically cryptographic protocols make use of one, or more, cryptographic primitives and/or schemes. An example might be the transmission of a credit card number from Bob to an e-commerce web site Alice. Such a protocol might involve a digital signature scheme (so Bob knows he is talking to Alice), and a form of encryption (to ensure Bob's credit card details are not intercepted in transit). Examples of deployed protocols which perform such operations are TLS or IPSec.

Whilst the theory and design of cryptographic primitives and schemes (discussed in the sister report [115]) is relatively well established, and the feed through into deployed systems and products is progressing well, the same cannot be said for protocols. Due to the inherent complexity of the situations for which they are intended and the many subtleties that arise, cryptographic protocols are notoriously difficult to design, and even more difficult to analyse.

In this document we outline the current knowledge on various classes of protocols. Our choice of protocols to cover is mainly dictated by what we feel is of most interest to the type of reader specified in the Executive Summary. Thus the protocols are either deployed in products, or are defined in standards and are hence planned to be deployed. Much of what we discuss in this report is likely to change as the research community shifts its focus to analysing protocols over the coming decade. The reader will hopefully also see by our analysis that most of the deployed protocols which can be used by a naive user, are in fact either incredibly complex to configure in a manner which we would deem secure, or are in fact insecure by modern cryptographic standards.

How did such a poor state-of-affairs for the analysis of protocols arise? There are multiple answers to this question. Firstly, although academic researchers have put a large amount of effort in analysing protocols, the results are in settings that although informed and motivated by practice, do not immediately impact real uses of protocols. Some research, mainly that on formal methods (e.g. [88,275,298]) or the UC framework (e.g. [75]), aims to develop general frameworks for protocol analysis. However, concrete protocol analysis in such frameworks is confined to a small class of protocols (witness the numerous analyses of the Needham-Schroeder protocol [236]) or to generic results (for example general multi-party computation [138, 310]). Some academic results concern more specific classes of protocols such as key agreement [38]. However, this analysis has often been performed after the standards have been produced, and usually considers versions of the implemented protocols that are too simplified.

A second reason is that practitioners usually adopt an ad-hoc design mentality for protocols. In this approach a protocol is designed according to some well-established principles and then one tries to find attacks; if attacks succeed, then a fix is suggested and new attacks are searched. This was acceptable in the early days of the subject, but now that we have formal theories to capture security definitions for protocols this approach does not seem adequate anymore.

Thirdly, there is a certain (perhaps understandable) inertia that prevents practice from incorporating academic research. It is often the case that issues identified within the design of protocols already in place are dismissed as being only of “theoretical” interest, until the issues are exploited by a real attacker. Thus there is an inbuilt conservatism to not change anything, despite evidence that a given protocol may have a problem. This resistance to change can result in security vulnerabilities.

1.1 Designing and Verifying Protocols: Ideal World

After having discussed what can go wrong, we now explain how things could be improved. Firstly, we note there are two schools as to how to verify a protocol; we refer to these loosely as the computational and the symbolic schools.

• **Computational School**: In this school, often called the provable security school, a protocol is analysed according to some well-defined model. A complexity theoretic reduction is given which turns an adversary against the protocol into an adversary against a component. Thus the protocol is secure if all of its components are secure. To determine whether the component is secure (which is often a cryptographic scheme) a further reduction is provided to a cryptographic primitive. These reductions can be complex, with proofs being many pages long, and needing (currently) to be produced and verified by hand.

• **Symbolic School**: In this school, often called the formal methods school, a protocol is analysed assuming perfect cryptography. The protocol is then abstractly described in, for example, a process algebra, with the attacker's goals being described by equations and the attacker's capabilities defined by equational theories. After this stage an automated checking process is applied to ensure that the attacker can never reach the stated goals. This type of analysis is often said to use the "Dolev-Yao" model [105].

Each school has its advantages and disadvantages. The first school has the disadvantage that it requires hand generated and checked proofs; on the other hand it models the cryptography and adversary as close to the real world as possible. The second school has the disadvantage that it assumes perfect cryptography (which we know is not possible in real systems), but it allows for tools that automate devising proofs, or at the very least checking them [54, 91, 218, 258].

In recent years there has been some progress towards using the best of both worlds. For *some* protocols the theory of *computational soundness* will imply that a protocol shown to be secure in the symbolic method will also be secure in the computational model [6, 89]. However, this method only applies to a restricted set of protocols. In the other direction, some researchers are applying the techniques from the symbolic school to try to automate the proof generation methods in the computational school [27, 28, 55]. However, both of these techniques are still in their infancy.

A major problem is that whatever method is used an actual protocol may not follow the abstract specification which is analysed. The process, known as *idealisation*, of turning a protocol specification as found in a standard into one which is analysable by any of the above techniques is notoriously error prone.

Thus the "correct" way for a protocol to be designed is for a specification to be given, which allows for it to be easily translated into an abstract specification. Then the protocol should be analysed using the techniques of either of the above schools. The designer should feedback lessons learned from the analysis into the protocol design and the process should be repeated.

At the end, if analysis has been performed with symbolic methods, and computational soundness results do not apply, then we can conclude that the protocol has no structural weaknesses and no obvious attacks. If computational soundness results do apply, or the protocol has been proved secure in the computational school, then we can conclude that the protocol is secure *with respect to the given computational security model*. This latter point means that any insecurity will either be one of implementation or uses an attack vector which was not captured by the security model.

1.2 Designing and Verifying Protocols: Real World

In the real world protocols are almost always designed, deployed, patched and extended before any formal analysis of their properties happens. This is either for historic reasons (the protocols date to a time before verification techniques were understood), or for business reasons (for example a protocol is defined “in house” and only then is made public for public analysis, after a product has been produced). Thus formal verification using either symbolic or computational means is often applied post-hoc.

Often this post-hoc analysis finds faults in the standardised/deployed protocols (see for example our discussion of ISO 9798 in Section 2.2) or it can only be applied to small subsets of the actual protocol (see for example our discussion of TLS in Section 3.1). In the latter case one can obtain verification of the small part, but not of the whole protocol.

A major problem with protocols is the question of composition. A protocol which is secure when run in isolation may not be secure if run in composition (either sequentially or in parallel). Since most protocols are run in an online environment, with multiple executions happening at any one time, the issue of parallel composition is vital to ensure. Theoretically, techniques exist to enable composition of protocols to be designed in from the start, for example the UC Framework [75]. However, the analysis of most existing protocols has been carried out in the standalone setting, which may result in attacks in practical situations.

1.3 How to Read This Document

We divide our discussion on protocols into four sub-themes:
• **General Protocols**: These are protocols which are designed to meet some general security requirement and which can be used in a multitude of applications. They are generally simple stand-alone protocols with no “reference” implementation or deployment; examples include: General key agreement and general identification.
• **Specific Protocols**: These are protocols which often implement some well specified security requirement, but for which there are well known instantiations. These protocols are ones which can be used to provide various application services depending on the specific environment in which they are deployed; examples include: TLS and Kerberos.
• **Application Specific Protocols**: These are protocols which are designed for a tightly defined specific application; examples include: UMTS/LTE, WEP/WPA and ZigBee.
• **Application Areas**: These are areas to which various protocols can be applied, and for which ENISA specifically requested a discussion. In this document this section focuses on the area of cloud computing.

To understand the distinction between the latter three cases consider the following: Application developers often choose to use TLS as a means to securing a channel between applications, and the various parameters can then be selected by the developer; irrespective of whether the application is providing a web service or not. On the other hand, one is forced to use the mobile phone protocols UMTS/LTE if one wishes to use a standard mobile phone, with provider mandated parameters and algorithms. In addition the use of UMTS/LTE outside this application is very limited. Clearly this division is not complete, as in some sense one is also forced to use TLS when accessing secure web sites, but the usage of TLS is much wider than that. In particular TLS can be used in the context of cloud computing to secure connections from a client to a cloud, or between cloud installations.

Protocol choices also depend on the specific environment being considered. In this version of the report, at the request of ENISA, we include a section detailing issues related to the environment which has been called “Cloud Computing”.

2 General Protocols

In this section we detail some protocol classes; which on their own have been specified in standards but for which we know of no concrete implementation. Thus we call them general protocols as they essentially specify classes of protocols; some of which are then instantiated in Section 3 on Specific Protocols.

For each protocol class we specify a general description, standardization efforts, and limitations and then we give technical details.

2.1 Key Agreement

General Description: A key agreement protocol allows two parties to agree on a shared secret key. There are various properties which may hold in a protocol; for example, one or both parties may end up being authenticated to the other, the protocol may guarantee a random key is output even if one party is compromised, and so on. Authentication of parties is generally obtained by parties holding a *static* public/private key pair, which are used in multiple sessions. Any key pairs used in a specific session are called *ephemeral* keys. An important distinction is between *key transport* where one party generates a key and sends it to the other, and *key agreement* where neither party has complete control over the key generation process.

We first discuss key agreement, since this is the one area in which there has been a rigorous analysis of protocols; with concrete security definitions being given. Despite this, the situation in relation to how these security definitions map onto real world protocols and their usages is still in a state of flux.

Standardization Efforts: The NIST standard [240] (resp. [241]) and the ANSI standards [20, 22] (resp. [21]) define methods for general key agreement using discrete logarithm based systems (resp. factoring based systems). The standard [240] introduces a nice taxonomy for such schemes with the notation *C(a, b)*, where $a, b \in \{0, 1, 2\}$. The number *a* refers to how many of the two parties contribute ephemeral keys to the calculation and the number *b* refers to how many of the two parties are authenticated by long term public/private key pairs. For example, traditional non- authenticated Diffie–Hellman is of type C(2,0), whereas traditional MQV [206] is of type C(2,2). The standards also provide various mechanisms for key confirmation.

There are a sequence of standards in the ISO/IEC 11770 series, which mirror the ISO/IEC 9798 series discussed below. The main set (ISO/IEC 11770-2,-3, and -4) detail mechanisms involving symmetric encipherment techniques [163], public key techniques [164], and weak secrets (such as passwords) [165].

Limitations: The general standards on key agreement mentioned above are very general. The reader should refer to the TLS, IPSec, etc discussions in Section 3 for examples of such protocols in more detail.

Technical Details: The security of key agreement schemes is somewhat complicated. The traditional security models of Bellare, Rogaway, et al base security on indistinguishablity of keys [37, 38, 53, 79]. This property is often not satisfied by real world protocols, and in particular by protocols using key confirmation. This issue has started to be treated in a number of works focusing on the TLS protocol (see below). Also discussion of the notion of one-sided authentication in key agreement protocols has only recently been considered in the academic literature [74,199]. Thus many of the options in these standards cannot be said to have fully elaborated proofs of security which are applicable in general situations.

The precise choice of which key agreement scheme to use is therefore highly dictated by the underlying application. However, the sister report to this one [115] can be used to determine key sizes (for the underlying factoring and discrete logarithm based primitives) as well as the key derivation functions, MAC functions and any other basic cryptographic components used.

Finally, a crucial requirement which is becoming more important in the real world is that of *forward secrecy*. A key agreement scheme is said to be forward secure if the compromise of the long term static private key of a party does not compromise the confidentiality of the agreed key for sessions which occurred prior to the compromise of the key. Thus we are ensured that the key agreed now will be secure against any future compromise of the static keys.

2.2 Identification and Authentication Protocols

General Description: Identification protocols are protocols which enable a party to establish in an online protocol that they are both "live" and the claimed person at the other end of the communication. As such, parties have a *static* secret or private key whose possession is being verified. This static key may either be associated to the known static public key, or may be a shared secret held between the person verifying their identity and the person proving their identity.

In this work we make no distinction between identification and user authentication; however in some applications (for example those based on biometrics) the distinction is important. In general, in an authentication protocol we are aiming to verify the person against a known *claimed* identity, in an identification protocol the verifier is not given the claimed identity[1] and needs to also output this value. Thus, *identification* means a way of determining who someone is from a given population, whereas *authentication* means confirming a claimed identity.

Identification and user authentication are closely related to the topic of key agreement. Indeed in the academic key agreement literature these are often treated at the same time, see for example [79]. The reason for this linkage is that key agreement without user identification achieves relatively little. In addition, often keys are agreed for the primary purpose of authenticating a subsequent communication without the need to perform expensive public key operations on each communication. However, in terms of applications the usage of identification and authentication techniques have wider impact; for example in access to resources and/or physical settings.

Standardization Efforts: The main standards in this area are the ISO/IEC 9798 series [168–173]. The main set (ISO/IEC 9798-2,-3, and -4) detail mechanisms involving symmetric encipherment (i.e. encrypting a challenge under a secret key as a means of authentication) [169], those involving a cryptographic check function (i.e. applying a MAC function to a challenge as a means of authentication) [170], and those involving a digital signature (i.e. signing a given challenge as a means of authentication) [171].

Limitations: The standards in the ISO/IEC 9798 series are mainly "folklore" and as such their analysis has only recently been performed in the academic literature. Work in the provable security tradition was first performed as early as 2001 [34], despite this the original standards contained numerous problems which were identified in 2011 [29, 30] using the symbolic tradition. See below for an extensive discussion on this point.

Technical Details: The problems identified in [29, 30] were identified using a tool called SCYTHER, which found the weaknesses and determined whether proposed fixes were correct. A related tool called SCYTHER-PROOF was used to produce proof scripts which were then machine-checked using the Isabelle/HOL theorem prover. Various problems were identified including role-mixup attacks, type flaws, and reflection attacks; most of the flaws resulted from poor specification of message formats or crucial missing fields. Thus data intended for one person could be routed to another, or data

[1] They know however, for example, that the identifying party claims to have an identity in some given database.

elements could be interpreted in different ways. The standards have since been revised to take into account the problems identified, but the analysis is a lesson in the importance of applying modern scientific techniques to protocol design as opposed to relying on folklore. As mentioned the analysis in [29, 30] is purely in the symbolic tradition. Thus we obtain guarantees of correctness and the identification of logical weaknesses. To our knowledge, no systematic analysis has been done in the computational tradition; nor has an analysis been conducted as to whether computational soundness results can be applied to the existing symbolic analysis. Clearly some of the protocols in ISO/IEC 9798-2, -3, and -4 have been analysed in the academic literature in a computational manner but this is not documented well, and there is always the issue of problems related to idealisation between the definition in the standard and the definition used in the academic literature.

The standard ISO/IEC 9798-5 [172] details protocols based on zero-knowledge techniques. Due to the difficulty of dealing with these using symbolic methods, these are not analysed in [29, 30]. However, all of the protocols in this standard have appeared in the academic literature with computational proofs of security. All of the basic techniques are based on the ideas behind the Fiat–Shamir identification scheme [117], and the closely related Guillou–Quisquater scheme [144]. In these protocols a user is identified by showing knowledge of some secret value which has been committed in their identifying information (e.g. by showing knowledge of a private key associated to a public key). As well as these "classic" methods the standard also contains the schemes of Girault, Poupard and Stern [133, 265], Girault and Pailles [134], Brandt et al [69] and Mitchell and Yuan [227]. The technique of Girault, Poupard and Stern [133, 265] also appears in ISO/IEC 29192-4 [166], which focuses on lightweight cryptographic techniques.

Despite being based on provably secure protocols there has been (to our knowledge) no analysis as to whether the idealisation process between the specification and the prior analysed protocols is correct, nor whether the protocols as specified are subject to type attacks (since the standardised protocols introduce many fields for various application specific reasons).

Standard ISO 9798-6 [173] examines mechanisms which utilise the need of a human operator to manually transfer a short data string from one device to another, or to manually verify that two short data strings are identical. The standard makes no reference to academic literature, although there is an extensive literature in this space, [25, 136, 181, 203–205,222,223,234,235,250,251,271, 292, 303].

2.3 Password Authenticated Key Exchange Protocols

General Description: Just like in key-agreement, password-based key exchange protocols (PAKEs) allow two parties to share a key. The difference is that the authentication of the entities involved in the exchange relies on passwords shared between clients and servers (thus reducing the dependence on a PKI). The challenge is to design protocols that are secure against off-line dictionary attacks – attacks where adversaries infer information about the password only from the transcripts of protocol executions. The guarantee one wants is that an adversary cannot impersonate a user except if he successfully guesses a password.

Standardization Efforts: There has been some standardization of PAKE protocols. But these are usually relatively limited in terms of application areas, being tied to a specific application, or have limited (if any) take up.

Limitations: Despite their intuitive usefulness there has been little take up in the real world of PAKE protocols. One reason, which is often cited for this, is the existence of a general patent on the EKE protocol. It may be useful to note that the patent on the EKE protocol expired in 2011

Technical Details: Syntactically, PAKE protocols fall in two classes, balanced and augmented. In balanced PAKEs the server and the users share passwords, whereas in augmented PAKEs (or verifier-based PAKEs) the server has only a one-way function of the passwords (e.g. a hash of the passwords). The latter are preferable as they offer some degree of security even in face of a complete server breach.

Two types of models are used in the security analysis of these protocols. The model proposed by Bellare, Pointcheval, and Rogaway (henceforth the BPR model) [37] is an indistinguishablity based model that builds on the ideas in [38]. There are two types of simulation based models, one due to Boyko, MacKenzie and Patel (henceforth the BMP model) [65], which in turn build on those of Bellare, Canetti, and Krawczyk [33] and Shoup [286], and one due to Canetti et al. (henceforth the CHKLM model) [78], which builds on the Universal Composability framework [75]. We start with an overview of existent efficient protocols in the Random Oracle Model, with Figure 3.1 summarising the discussion.

The seminal protocol in this area is the Encrypted Key Exchange (EKE) protocol of Bellovin and Merritt [41], followed by an augmented version [42]. The security of the protocol had been first analysed by Bellare et al. [37] but the analysis relies on the strong ideal cipher model. Slight variations that aim to preserve the efficiency of the protocol but reduce the assumptions needed in the proof have later been provided [70, 71]. The most efficient protocol that resulted from this line of work is the SPAKE protocol due to Abdalla and Pointcheval [9]; the protocol has a security analysis in the random oracle model.

The SPEKE B-SPEKE protocols proposed by David Jablon [174, 175] were two of the first proposed protocols following the publication of EKE. MacKenzie provides an analysis of SPEKE in a restricted variant of the BPR model [219] (the mBPR model). In a similar vein Boyko, MacKenzie and Patel [65] proposed PAK and prove it secure in the BMP model assuming the random oracle. They also provide an augmented version called PAK-X.

In contrast to the above protocols, the PACE protocol is one which was designed for a specific application. It was proposed by the German Federal Office for Information Security, and is intended for deployment in machine readable travel documents, and protocol is fully specified in standards, e.g. in [198]. The protocol has a security proof with respect to (a variant) of the BPR model. The proof assumes both random oracles and ideal ciphers [44].

Protocol	Augmented /Balanced	Security Model/Proof	References
EKE	balanced	BPR/ICM	[37, 41]
SPEKE	balanced	mBMP/ROM	[175]
B-SPEKE	augmented	none	[174]
PAK	balanced	BMP/ROM	[65]
SPAKE	balanced	BPR/ROM	[9]
PACE	balanced	mBPR/ROM	[44, 201]
J-PAKE	both	See text	[146, 147]
Dragonfly	balanced	none	[148]

Figure 2.1: Summary of PAKE in the Random Oracle Model (ROM) and Ideal Cipher Model (ICM)

Some other protocols that have gained some traction recently (mainly as IP free alternatives) are J-PAKE [147] and Dragonfly [149]. Claims of security for these protocols are however not supported by fully worked-out proofs.

Just like for any other primitive/protocol PAKE protocols secure in the standard model, i.e. ones not using random oracles, are not very efficient; to the point where they are not really practical. A series of works starting with the protocol of Katz, Ostrovsky and Yung [185] and the more general framework of Goldreich and Lindell [137] propose PAKEs that are secure in the standard model. These protocols which include those in [8, 10, 26] but are significantly less efficient than those discussed above.

3 Specific Protocols

In this section we detail general purpose protocols for accomplishing various tasks, which can be used, and are, used in multiple application scenarios. Whilst the protocols here were designed for *specific* tasks (for example TLS was designed to secure communication between a browser and a web site) they can, and are, used for other applications.

3.1 TLS

General Description: The TLS protocol (the current version being v1.2) is primarily aimed at securing traffic between an unauthenticated web browser and an authenticated web site, although the protocol is now often used in other applications due in part to the availability (and ease of use) of a variety of libraries implementing TLS. The TLS protocol suite aims to provide a confidential channel rather than simply a key agreement protocol as discussed before in Section 2.1. The protocol is broken up into two phases: A handshake (or key agreement) phase and a record layer encryption phase.

Standardization Efforts: The protocol has been standardised by the IETF in various standards, of which we list just some [52,101–103,224,279]. The genesis of the protocol dates back to SSL v1.0, in 1993, and its current complex state is a symptom of both issues related to backward compatibility and mission creep. The protocol is currently undergoing a major revision so as to produce TLS v1.3.

A complete list of ciphersuites for TLS is listed at the website http://www.iana.org/assignments/ tls-parameters/tls-parameters.xml. If following the recommendations of this document, the restrictions on the ciphersuites to conform to our guidelines for future systems means this large list becomes relatively small. We provide guidelines on specific ciphersuites, for both the hand- shake and record layer transport phases, below.

Limitations: Due to the non-systematic development process, the protocol is hard to analyse and easily prone to implementation weaknesses. Below we summarise the latest knowledge in this regard. Care must be taken in long term key generation as a number of TLS implementations have been shown to be weak due to poor random number generation, see [151] and [115][Section 6.2].

Technical Details: The handshake/key agreement phase has now been fairly thoroughly analysed in a variety of works [73, 176, 177, 199, 231]. A major issue in these analyses is the use of the derived key during key confirmation via the FINISHED messages.

The handshake/key agreement phase runs in one of essentially two main modes: either RSA- based key transport or Diffie–Hellman key exchange (an option also exists for pre-shared keys). The RSA key transport methodology uses RSA-PKCS#1 v1.5, which is discussed in [115][Section 4.6.1] is not considered secure in a modern sense. However, the use of this key transport methodology has been specifically patched in TLS to avoid the attack described in [56], and a formal security analysis supporting this approach in the TLS context can be found in [199]. This latter analysis shows that key transport in TLS can be made secure (but not forward secure) under a sufficiently strong number theoretic assumption and in the Random Oracle Model. The Diffie–Hellman based key agreement mode is considered much more secure, and offers the benefit of perfect forward secrecy of the agreed key. In both modes the output of the key agreement phase is a so-called pre-master secret.

For the handshake part of the protocol the principle issue is that the RSA signing algorithm in TLS 1.2 is RSA-PKCS# 1 v1.5. Since most certificates issued are certificates on RSA keys, this means that RSA-PKCS# 1 v1.5 is the default signing algorithm for use in TLS. As explained in [115][Section 4.8.1] we do not advise the use of this signature scheme in future systems.

Considering the discrete logarithm or elliptic curve signature variants, one finds that the situation is a little better. The required signature algorithm here is (EC)DSA, which also has no proof of security, bar in the generic group model for the elliptic curve variant. See [115][Section 4.8.5] for more details. Thus for the key negotiation phase one is left to rely on cryptographic schemes which we only suggest for legacy use.

Given these caveats, we advise the following key exchange methods for legacy use in TLS as they provide forward secrecy

• TLS_DHE_DSS_WITH_⋆,

• TLS_DHE_RSA_WITH_⋆,

• TLS_ECDHE_ECDSA_WITH_⋆,

• TLS_ECDHE_RSA_WITH_⋆,

Where the ⋆ suffix denotes an underlying record layer encryption method. The only thing which stops us recommending any key exchange methods for future use is the lack of a provably secure public key signature algorithm within the available choices. Of the four choices TLS_ECDHE_ECDSA_WITH_⋆ is probably to be preferred as ECDSA signatures are more likely to be secure in the long run than the RSA method.

In TLS 1.3 it is proposed to remove the key transport (i.e. RSA variant) and only have forward secure key agreement phases. In particular this would mean that long term public keys are only used to provide key authentication and are not used to provide key confidentiality. This change, as well as being good security practice, has been accelerated since summer 2013 due to the Snowden revelations.

During the handshake phase the key to use in the transport layer is derived from the agreed pre-master secret. This derivation occurs in one of two ways, depending on whether TLS 1.2 [103] is used or whether an earlier standard is used (TLS 1.0 [102] and TLS 1.1 [102]). As discussed in [115][Section 4.4.5], the use of TLS-v1.1-KDF should only be used for legacy applications, with the TLS-v1.2-KDF variant being considered suitable for future applications.

The record layer, i.e. the layer in which actual encrypted messages are sent and received, has received extensive analysis. In TLS 1.0 and TLS 1.1 the two choices are either MAC-then-Encode- then-Encrypt using a block cipher in CBC mode or the use of MAC-then-Encrypt using the RC4 stream cipher. Both these forms of the record layer have been shown to be problematic [16,17,81, 255,302]. The main problems here are that the MAC-then-Encode-then-Encrypt construction used in TLS is difficult to implement securely (and hard to provide positive security results about), and that RC4 is, by modern standards, a weak stream cipher. These issues are partially corrected in TLS 1.2 [103] by adding support for Authenticated Encryption, and with GCM mode and CCM mode for TLS being specified in [279] and [224], respectively. Other recent attacks include those by Duong and Rizzo, known as BEAST [107] and CRIME [108]. BEAST exploits the use of chained IVs in CBC mode in TLS 1.0, and CRIME takes advantage of information leakage from the optional use of data compression in TLS. In TLS 1.3 it is proposed that only AEAD methods are used to secure the record layer.

Looking at the record layer protocol (i.e. the algorithms to encrypt the actual data), we see that only the use of Camellia and AES, within a mode such as GCM or CCM, are compatible with the guidelines in [115], This means at the time of writing we would only advise the following cipher suites, for the record layer for future (and legacy) use within TLS

• ⋆_WITH_Camellia_128_GCM_SHA256,

• ⋆_WITH_AES_128_GCM_SHA256,

• ⋆_WITH_Camellia_256_GCM_SHA384,

- ⋆_WITH_AES_256_GCM_SHA384,
- ⋆_WITH_AES_128_CCM,
- ⋆_WITH_AES_128_CCM_8,
- ⋆_WITH_AES_256_CCM,
- ⋆_WITH_AES_256_CCM_8.

where the ⋆ prefix denotes an underlying key exchange method.

Given the above discussion it is hard to recommend that TLS 1.0 and TLS 1.1 be used in any new application, and phasing out their use in legacy applications is advised. It would appear that TLS 1.2 is sufficient for future applications. However, there are currently very few implementations of clients, servers, or libraries which support TLS 1.2, although this is now increasing rapidly.

3.2 SSH

General Description: Secure Shell (SSH) was originally designed as a replacement for insecure remote shell protocols such as telnet. It has now become a more general purpose tool that is used to provide a secure channel between two networked computers for applications such as secure file transfer. In general the host one is connecting to is authenticated, whereas the client is not (although some corporations do insist on client side authentication for SSH usage).

Standardization Efforts: SSHv2 was standardised in a collection of RFCs [312–314] in 2006, other relevant standards are [32, 48, 93, 162, 215, 215]. The original version, SSHv1 has several design flaws and should no longer be used. OpenSSH [242] is the most widely used implementation of the protocol. In 2008 it accounted for more than 80% of all implementations. The transport layer of SSH [314] is responsible for the initial key-exchange, server authentication and, confidentiality and integrity of messages sent on the channel.

Limitations: The main issue with SSH, much like TLS above, is that most of the standard encryption algorithms for the transport layer are not sufficient to ensure complete security. They possess a number of cryptographic weaknesses, which would not exist if the protocol choices had been made more recently. See the following section for a technical discussion on these matters, as well as guidelines for going forward.

Technical Details: The key-exchange protocol is based on Diffie–Hellman and host authentication is provided by combining this with a signature. Client authentication is also possible but defined in a separate RFC [312]. Methods for authenticating the client are either using a password, public- key cryptography (DSA, RSA, X.509), an “interactive-keyboard” challenge-response method [93] or the GSSAPI [215] which allows the use of external mechanisms such as Kerberos. Support for the key-exchange methods, `diffie-hellman-group1-sha1` and `diffie-hellman-group14-sha1` is mandated by the RFC [314]. These methods use the Oakley Group 1 (1024-bit prime field) and Oakley Group 14 (2048-bit prime field) [195]. RFC4419 [123] describes a key-exchange method for SSH that allows the server to propose new groups on which to perform the Diffie–Hellman key exchange with the client. RFC4432 [150] specifies a key-transport method for SSH based on 1024- bit and 2048-bit RSA. RFC5656 [293] defines introduces support for Elliptic-Curve Cryptography; detailing support for ECDH and ECMQV.

Williams [309] has performed an analysis of the key-exchange methods in SSH. It has been shown the six application keys (two IV keys, two encryption keys and two integrity keys) generated by the protocol and passed to the next stage of the SSH protocol are indistinguishable from random. The analysis assumes the server’s public key is validated through a certificate from some secure public-key infrastructure. The author of [309] notes that if no such certificate is used, then the protocol is vulnerable to attack, unless the client has some other method of verifying the authenticity of a server's public key.

Once keys are established all message are then sent encrypted over the channel using the Binary-Packet Protocol (BPP) [314, Section 6]. This specifies an encryption scheme based on an Encode- then-Encrypt-and-MAC construction using a block cipher in CBC mode or the stream cipher RC4. The encode function specifies two length fields which must be prepended to messages prior to encryption and a padding scheme (for the case of CBC mode). The first length field specifies the total length of the packet and the second gives the total length of padding used. The specification recommends using CBC mode with chained IVs (the last block of the previous ciphertext becomes the IV of the following ciphertext). This has been shown to be insecure by Dai [94] and Rogaway [273]. Albrecht et al. [15] were able to perform plaintext-recovery attacks against SSH (when using CBC mode) by exploiting the use of encrypted length fields. As a result of these attacks we state that CBC mode should not be used. We note that OpenSSH Version 6.2 [242] supports a non- standard version of the BPP for use with CBC mode in an Encrypt-then-MAC construction where length fields are not encrypted but still authenticated. This style of construction would be secure against the Albrecht et al. attack.

A first formal security analysis of the SSH-BPP was performed by Bellare et al. [36]. As a result of the Albrecht et al. attacks this security analysis was proved to be incomplete and a further security analysis, which more closely matched actual implementations of SSH, was performed by Paterson and Watson [256]. They proved that the Encode-then-Encrypt-and-MAC construction utilising counter mode encryption is secure against a large class of attacks including those of Albrecht et al. We advise counter mode as the best choice of available cipher in the Encode-then-Encrypt- and-MAC construction when combined with a secure MAC algorithm. The original choice of MAC algorithms specified in RFC4253 was limited to HMAC with either SHA-1 or MD5. We advise neither of these hash functions for current use. RFC6668 [48] details the use of SHA-2 for HMAC. In addition to the Encode-then-Encrypt-and-MAC construction confidentiality and integrity in SSH may also be provided by GCM encryption as specified in RFC5647 [162].

A complete list of ciphersuites for SSH is listed at the website http://www.iana.org/assignments/ssh-parameters/ssh-parameters.xml.

Based on the guidelines of this document we would only advise the following encryption and MAC algorithms, for future use within SSH:

• aes128-ctr with hmac-sha2-256 or hmac-sha2-512

• aes192-ctr with hmac-sha2-256 or hmac-sha2-512

• aes256-ctr with hmac-sha2-256 or hmac-sha2-512

• AEAD_AES_128_GCM

• AEAD_AES_256_GCM

3.3 IPSec

General Description: IPSec is designed to provide security at the IP network layer of the TCP/IP protocol stack. This differs from protocols such as TLS and SSH, above, which provide security at higher layers such as the application layer. This is advantageous since no re-engineering of the applications is required to benefit from the security IPSec provides. The main use of IPSec has been to create virtual private networks (VPNs) which facilitates secure communication over an untrusted network such as the Internet.

The IPSec protocols can be deployed in two basic modes: tunnel and transport. In tunnel mode cryptographic protection is provided for entire IP packets. In essence, a whole packet (plus security fields) is treated as the new payload of an outer IP packet, with its own header, called the outer header. The original, or inner, IP packet is said to be encapsulated within the outer IP packet. In tunnel mode, IPSec processing is typically performed at security gateways (e.g. perimeter firewalls or routers) on behalf of endpoint hosts. By contrast, in transport mode, the header of the original packet itself is preserved, some security fields are inserted, and the payload together with some header fields undergo cryptographic processing. Transport mode is typically used when end-to- end security services are needed, and provides protection mostly for the packet payload. In either mode, one can think of the IPSec implementation as intercepting normal IP packets and performing processing on them before passing them on (to the network interface layer in the case of outbound processing, or to the upper layers in the case of inbound processing).

Standardization Efforts: The protocol was originally standardised in a collection of RFCs in 1995 and their third incarnation can be found in RFCs 4301–4309 [110,153,155,187,190–193,280]. For a more complete description of the cryptography in the IPSec standards we refer the reader to the survey by Paterson [252].

Limitations: The key agreement phase of IPSec, called IKE, is well studied and well defined. As for TLS and SSH the payload encryption algorithms have had a number of issues over the years, related to poor acceptance of the need for AEAD/IND-CCA encryption algorithms. More details are provided in the technical section below.

Technical Details: Each IPSec implementation contains a Security Policy Database (SPD), each entry of which defines processing rules for certain types of traffic. Each entry in the SPD points to one or more Security Associations (SAs) (or the need to establish new SAs). The SAs contain (amongst other information) cryptographic keys, initialisation vectors and anti-replay counters, all of which must be initialised and shared between appropriate parties securely. This can be solved manually, and such an approach works well for small-scale deployments for testing purposes.

However, for larger scale and more robust use of IPSec, an automated method is needed. The Internet Key Exchange (IKE) Protocol provides the preferred method for SA negotiation and associated cryptographic parameter establishment. The latest version of IKE, named IKEv2 [188], provides a flexible set of methods for authentication and establishment of keys and other parameters, supporting both asymmetric and symmetric cryptographic methods. There were initially two Diffie–Hellman Groups defined for use in IKEv2 [188, Appendix B], one with a 768-bit modulus the other with 1024-bit modulus. Further DH groups are defined in RFC3526 [195] of sizes 1536, 2048, 3072, 4096, 6144 and 8192 bits. Elliptic Curve groups are defined in RFC 5903 [124] with sizes of 256, 384 and 521 bits. RFC5114 [208] defines an additional 8 groups. Based on [115][Section 3.5] we advise for future use a group size of at least 3072 bits, and 256 bits in the case of elliptic curve groups. For key derivation, as discussed in [115][Section 4.4.4], the use of IKE-v1-KDF should only be used for legacy applications, with the IKE-v2-KDF variant being considered suitable for future applications.

There are two main IPSec protocols which specify the actual cryptographic processing applied to packets. These are called Authentication Header (AH) and Encapsulating Security Payload (ESP).

AH provides integrity protection, data origin authentication and anti-replay services for packets through the application of MAC algorithms and the inclusion of sequence numbers in packets. There are a number of MAC algorithms defined for use with IPSec. These include HMAC (with MD5 [220], SHA-1 [221] or SHA-2 [189]), GMAC [225] and XCBC (a CBC-MAC variant) [121]. Based on earlier sections we only advise HMAC with SHA-2 for future use.

ESP provides similar services to AH (though the coverage of its optional integrity protection feature is more limited) and in addition provides confidentiality and traffic flow confidentiality services through symmetric key encryption and variable length padding of packets. ESP allows both encryption-only and authenticated encryption modes. The attacks we shall mention in the following paragraph demonstrate the encryption-only modes should not be used. ESP must therefore always be configured with some form of integrity protection. The encryption algorithms on offer are CBC mode (with either 3DES [259], AES [120] or Camellia [184]), CTR mode (with either AES [154] or Camellia [184]). Of these algorithms we would only advise CTR mode and stress it must be combined with a MAC algorithm. Further options for authentication encryption are provided by the combined algorithms CCM (with either AES [155] or Camellia [184]) and GCM with AES [155].

An initial analysis of the IPSec standards was performed by Ferguson and Schneier [116]. This was followed by Bellovin [40] who found a number of attacks against encryption-only ESP. Practical attacks were demonstrated by Paterson and Yau [257] against the Linux implementation of IPSec where encryption-only ESP was operating in tunnel mode. By adapting the padding oracle attack of Vaudenay [302], Degabriele and Paterson were then able to break standards-compliant implementations of IPSec [99] with practical complexities. These attacks were against encryption-only ESP using CBC mode and operating in either tunnel or transport mode. From these attacks, the need to use some form of integrity protection in IPSec is evident. It is therefore recommended that encryption-only ESP not be used. A further set of attacks by Degabriele and Paterson [100] breaks IPSec when it is implemented in any MAC-then-Encrypt configuration (for example, if AH in transport mode is used prior to encryption-only ESP in tunnel mode). On the other hand, no attacks are known if ESP is followed by AH, or if ESP's innate integrity protection feature is used. To conclude, we reiterate that ESP should always be used with some form of integrity protection, and that care is needed to ensure an appropriate form of integrity protection is provided.

A close to complete list of ciphersuites for IPSec is listed at the website http://www.iana.org/assignments/isakmp-registry/isakmp-registry.xml.

Based on the guidelines in [115] we would only advise the following algorithms for future use within IPSec:

- If only authentication is required then either AH or ESP may be used with one of the following MAC algorithms as defined in RFC4868 [189].
  - HMAC-SHA2-256,
  - HMAC-SHA2-384,
  - HMAC-SHA2-512,
- If confidentiality is required then ESP should be used by combining one of the following encryption algorithms with one of the MAC algorithms described above.
  - AES-CTR,
  - CAMELLIA-CTR,
- Alternatively one of the following combined authenticated encryption modes may be used:
  - AES-CCM_⋆,
  - CAMELLIA-CCM_⋆,
  - AES-GCM_⋆,

Here, the ⋆ denotes the size (in bytes) of the integrity check value (ICV) and we advise choosing either 12 or 16.

3.4 Kerberos

General Description: Kerberos is an authentication service which allows a client to authenticate his or herself to multiple services e.g. a file server or a printer. The system uses a trusted authentication server which will grant tickets to participating parties allowing them to prove their identity to each other. It is primarily based on symmetric-key primitives; the specific construction being derived from the Needham-Schroeder Protocol [236]. Public-key primitives, namely RSA signatures, may also be used during the initial authentication phase [319].

Standardization Efforts: Kerberos was designed as part of project Athena at MIT during the 1980s [226]; the first three versions were not released publicly; Version 4 can therefore be viewed as the “original” release. The current version, Version 5 [237], fixed a number of security deficiencies present in its predecessor [39]. Version 4 required the use of DES; Version 5 expanded the possible ciphers and AES is now supported [269].

Limitations: Again there are issues related to the usage of strong encryption schemes (i.e. AEAD/IND-CCA schemes) due to the age of the documents defining the protocol.

Technical Details: Version 4 made use of a non-standard version of CBC mode called PCBC which has been shown to be insecure [196]. The encryption scheme used by Version 5 has been formally analysed by Boldyreva and Kumar [61], who first show that the Encode-then-Checksum- then-Encrypt construction defined in RFC3961 [270] does not meet the INT-CTXT definition of security. If a secure MAC algorithm is used for the checksum then this construction will be secure. Additionally, Boldyreva and Kumar analyse the Encode-then-Encrypt-and-MAC construction given in RFC3962 [269] and show this to be secure assuming the underlying primitives meet standard definitions of security. The encryption scheme specified for use in Version 5 is CBC mode with ciphertext stealing using either DES, 3DES [270], AES [269] or Camellia [157] as the underlying blockcipher.

A complete list of ciphersuites for Kerberos is listed at the website http://www.iana.org/assignments/kerberos-parameters/kerberos-parameters.xml. At the time of writing we advise the following ciphersuites for future use within Kerberos:

- aes128-cts-hmac-sha1-96
- aes256-cts-hmac-sha1-96
- camellia128-cts-cmac
- camellia256-cts-cmac

4 Application Specific Protocols

In this section we present a quick overview of protocols which are used in relatively restricted application areas; for example wireless, mobile communications or banking.

4.1 WEP/WPA

General Description: The WEP/WPA protocols are used to protect communication in wireless networks; for example in securing the communication between a laptop and the wireless router (a.k.a access point) to which it connects. The key design requirement is to ensure that an eavesdropper is unable to break the confidentiality of the messages being sent. We discuss their use in the setting where the device and the access point to which it connects have a shared key.

Standardization Efforts: WEP (Wired Equivalent Privacy) is specified in the IEEE 802.11 standard [158]. The protocol is intended to offer confidential and authenticated communication. The protocol is symmetric key based (it uses either 40 bit, 104 bit, or 232 bit keys) and employs RC4 for confidentiality and CRC32 for authentication. WPA (Wi-Fi Protected Access) is a successor of WEP. It employs the Temporal Key Integrity Protocol (TKIP) a stronger set of encryption and authentication algorithms; but TKIP has been deprecated by the IEEE. The WPA2 is the latest version of the protocol suite which is described in [159].

Limitations: Practical key-recovery attacks against the WEP protocols have been devised [51, 119,297] and the protocol is considered completely broken. The use of this protocol should be avoided. WEP has been deprecated by the IEEE. The TKIP protocol was intended as a temporary replacement for WPA, and is capable of running on legacy hardware. The protocol fixes some of the design problems in WEP, but some attacks against TKIP have been found [145,228,230,281, 296,299]. A recent attack, [254], based on prior analysis of RC4 [16] in TLS, breaks the basic WPA protocol, and thus users should move to WPA2 as a matter of urgency.

Technical Details: The protocol WPA2 uses stronger primitives. It employs the Counter Cipher mode with Message Authentication Code Protocol (CCMP), an encryption scheme that uses AES in CCM mode (see [115][Section 4.3.3]) and offers both message confidentiality and message authentication. While some weaknesses in settings where WPA2 is used exist, no serious attacks are known against the protocol itself.

4.2 UMTS/LTE

General Description: The GSM, UMTS and LTE protocols are designed to secure communications between a mobile phone and the operators' base station. The goal being to provide confidentiality services for the user and authentication services for the mobile phone operator. The protocols also define what happens when a user "roams" to another service provider's network, by for example travelling to another country. In addition the protocols provide a limited form of anonymisation of the user, by preventing a passive eavesdropper from linking one communication with another from the same phone.

Standardization Efforts: The Universal Mobile Telecommunication System (UMTS) and its latest version called Long-Term Evolution (LTE) are standards for wireless communication in mobile phones and data terminals. The standard is developed by the 3rd Generation Partnership Project (3GPP) and is now at version 10. The protocol is intended as a replacement for GSM. All technical specification documents referenced in this section are available at www.3gpp.org.

Limitations: There are known minor weaknesses in the cryptographic components used in LTE, in particular Kasumi [49] and SNOW 3G [50,194], but these do not seem to translate into attacks against the secure channel that they implement.

Technical Details: Very roughly, the protocol works in two phases, a key-establishment and authentication phase, and a data transmission phase. Unlike the TLS and IPSec protocols discussed earlier the key establishment and authentication are obtained via symmetric as opposed to public key techniques.

UMTS/LTE replaces the one-way authentication protocol used in GSM (which authenticates the mobile but not the network) with a stronger protocol called Authentication and Key Agreement (AKA). This is a three party protocol that involves a mobile station (MS) a serving network (SN) and the home environment (HE). Upon a successful execution of the protocol MS and SN have confirmed that they communicate with valid partners and establish a shared key. An additional design goal for the protocol is to protect the identity of the mobile station: an eavesdropper should not be able to determine whether the same mobile station was involved in two different runs of the protocol.

The key shared between MS and SN is used to implement a bi-directional secure channel between the two parties. Integrity and confidentiality are implemented (respectively) via algorithms UIA1 and UEA1 (in UMTS) [1] and UIA2 and UEA2 (in LTE) [2]. The algorithms have the same structure; the difference is determined by the underlying primitive: the Kasumi block cipher [4] in UMTS and SNOW 3G stream cipher [3] in LTE.

There are no provable security guarantees for the protocol. The few published analyses for the protocol are mainly concerned with the anonymity guarantees [23,197] and indicate that the protocol is susceptible to a number of attacks against mobile station confidentiality. Security of the channel established via UMTS/LTE had not been thoroughly analysed.

4.3 Bluetooth

General Description: Bluetooth is technology for exchanging data, securely, over short-distances between up to seven devices. It is often used to connect devices on a body (for example a mobile phone and a headset) or within a vehicle (for example a mobile phone and the vehicles audio system).

Standardization Efforts: The protocol stack for Bluetooth was originally standardised as IEEE802.15.1 standard, which is no longer maintained. The current development is over seen by the Bluetooth Special Interest Group. We discuss the cryptographic features of Bluetooth 2.1; the later versions (the latest is Bluetooth 4.0) are mainly concerned with improved bandwidth and power efficiency with little changes to the underlying cryptography.

Limitations: See below for issues related to the pairing of devices.

Technical Details: Operating takes place in two stages. In the “pairing” stage, two Bluetooth devices agree on a pair of keys, an initialisation key used for mutual authentication via a challenge response protocol based on HMAC-SHA-256; after authentication succeeds, the devices also agree on a link key for encrypting the traffic. Since Bluetooth 2.1 this stage is implemented with Elliptic Curve Diffie-Hellman (ECDH); depending on the capabilities of the devices involved, several mechanisms for providing protection against man-in-the-middle can be used. Data is encrypted in Bluetooth using streamcipher E0. Each packet is XORed with a keystream obtained by running the E0 algorithm on several inputs, one of which is the key link and another is a unique identifier.

The main weakness of Bluetooth is the pairing phase. Although stronger than in Bluetooth 1.0-2.0, pairing is still open to Man-In-The-Middle attacks for devices without user input/output capabilities or other out-of-band communication means, or in configurations where a predefined PIN is used. As far as confidentiality of the communication goes, the few known theoretical attacks against E0 [118,152] do not seem to impact confidentiality of messages. Message integrity protection is implemented with a cyclic redundancy code and is therefore minimal.

4.4 ZigBee

General Description: ZigBee is a radio communication standard which can be considered to operate mainly at lower power and ranges than Bluetooth. The key idea is to provide extended ranges by utilising mesh networks of ZigBee connected devices.

Standardization Efforts: The ZigBee protocol suite is defined by the ZigBee Alliance http://www.zigbee.org/.

Limitations: There are no known issues with the Zigbee protocols, although we know of no formal analysis of the protocols.

Technical Details: Bulk data encryption and authentication is based on the symmetric key mechanisms of IEEE 802.15.4 [160], and key management is implemented either by active key management with ZigBee-specific uses of ECDSA/ECDH or by predistribution of symmetric keys.

The main confidentiality algorithms are AES in CTR mode, an AES based CBC-MAC algorithm outputting either a 32-bit, 64-bit or 128-bit MAC value, or for combined authenticated encryption the use of AES in CCM mode, or a variant of CCM mode called CCM∗. TLS support is provided with two mandatory cipher suites

TLS_PSK_WITH_AES_128_CCM_8 and TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8,

these derive keying material either via symmetric preshared keys or via a elliptic curve Diffie–Hellman key exchange authenticated with ECDSA respectively. An optional suite of

TLS_DHE_RSA_WITH_AES_128_CCM_8

prepares the shared keying material via a finite field Diffie–Hellman exchange authenticated with RSA signatures.

4.5 EMV

General Description: The EMV system defines a “platform” for enabling a chip card to be used in banking applications. Generally this platform is used to implement the chip-and-pin system deployed across much of the world. It is also used in some countries to provide online banking services via the use of a cheap chip-card reader. The EMV system is slated to be rolled out world-wide with the adoption in the United States in the next couple of years.

Standardization Efforts: The chip-and-pin bank/credit card system follows a specification defined by EMVCo. Since this report is mainly focused on cryptographic aspects, we will restrict our discussion to the cryptographic components only; which are defined in “EMV Book 2” [111]. A new system is currently in the process of being standardised.

Limitations: There are a number of systems security level issues observed by the Cambridge security group [11,12,62,106,232].

Technical Details: Much of the existing EMV specification dates from before the advent of provable security; thus many of the mechanisms would not be considered cryptographically suitable for a new system. For example, the RSA based digital signature is DS1 from the standard ISO 9796-2 [167]; in a message recovery mode. As already explained in [115][Section 4.8.4], this scheme suffers from a number of weaknesses, although none have been exploited to any significant effect in the EMV system.

As a second example, the RSA encryption method (used to encrypt PIN blocks in some countries) is bespoke and offers no security guarantees. The only known analysis of this algorithm is in [289], which presents a Bleichenbacher-style attack against this specific usage.

Another issue is that the card is allowed to use the same RSA private key for signing and encryption. This is exploited in [98] via another Bleichenbacher-style attack which converts the decryption oracle provided by the Bleichenbacher-style attack into a signing oracle; in turn, this can be used to forge EMV transaction certificates. It should be stated that none of the above attacks has been shown to be exploitable “in the wild”. Rather, they highlight potential problems with the current algorithm choices.

The symmetric key encryption schemes used in EMV are also slightly old fashioned. Two block ciphers are supported Triple DES and AES, with the underlying encryption method being CBC mode. The standard supports two MAC functions, AMAC for use with single DES and CMAC for use with AES.

EMVCo is currently engaged in the process of renewing their cryptographic specifications to bring them up to date. There has been a lot of work on defining elliptic curve based schemes for use in EMV. Some work has been done on analysing the specific protocols and schemes being considered for use in the new specifications. For example [74] presents a detailed security analysis of the key agreement and secure channel protocol which is proposed to be used to secure the communication between the chip card and the merchants’ terminal.

5 Application Areas

In this section we focus on specific application areas; and discuss the application of existing protocols to these areas. In this year's report we focus on the area of Cloud Computing only; the idea being that additional areas will be covered in future reports.

5.1 Cloud Computing

The notion of cloud computing is essentially a catch all buzzword for a number of differing ways of working, which in the past would have been called a client-server or mainframe model of computing. The key differentiator between cloud computing and prior models is that the server, or mainframe, is now a set of many connected computers. With individual computers being made to look like multiple physical computers via the use of virtualization.

The main reason for the take up of the model of cloud computing is the reduced cost of utilising cloud infrastructures. Large data centres can be centralised, thus reducing costs of real estate and power (indeed some data centres are located next to their own power plants). In addition the use of one central infrastructure used by multiple clients (a concept called multi-tenancy) enables the client to expand and contract their usage on a needs basis (notion of on demand scalability).

In general cloud computing offers a relatively neutral effect on system security. On one hand the centralisation and management of the security infrastructure enables the cloud's customers to enjoy a greater level of security than they would have if they ran the system "in house". On the other hand, the aggregation of customers into one homogeneous infrastructure provides an attacker with a single high value target. However, outsourcing produces problems related to ensuring the privacy of any outsourced data; a subject which we will come back to below.

5.1.1 Cloud Models

However, despite these commonalities cloud computing is not a single problem instance from the point of view of security. There are basically a number of differing cloud business models[2] and applications; each of which pose their own security challenges. The three core areas are denoted IaaS, PaaS, and SaaS;

- **Infrastructure as a Service (IaaS)**: Here the provider provides a set of virtual machines and storage capability to the users. The most well-known of these installations is perhaps Amazon's Elastic Computing Cloud (EC2) and Microsoft's Azure. From the point of view of the user the cloud provider presents them with a physical machine running a specific operating system which they remote log into. However this physical machine could be shared between multiple clients via virtualization.
- **Platform as a Service (PaaS)**: The next layer up is where the infrastructure provider presents a complete platform to the client, for example a LAMP stack of web-server, database and PHP engine, or a web-front end equivalent. In general the user has no knowledge (or need for knowledge) of the underlying hardware. The platform provides development building blocks for the user, and not an end user application. Charging is usually done on a usage basis (transaction or storage volume). The client then uses this platform to build their own applications. Some providers (e.g. Google) automatically scales the users storage and computing requirements with the users needs.

[2] In this section the focus is beyond the cloud infrastructure and the deployment model; the perspective used cover also what we want to achieve using the cloud infrastructure.

- **Software as a Service (SaaS)**: Here the provider provides a number of applications to the user on their cloud infrastructure. Generally these are a set of web applications, and clients do not need to install anything on the cloud infrastructure; clients connect to the service via a standard web browser. The software is usually accessed via a web front end, and users pay on a per-use or monthly basis (or have the inconvenience of adverts). Examples of SaaS include Saleforce's applications, Microsoft's 365-suite, Google's applications such as Calendar, Email etc, and some online games.

Each of the above application areas creates its own security challenges, some of which are standard and do not deviate from normal computing operations. Others, however, produce unique security problems due to the nature of the use of remote clouds. See the ENISA report [112] for a full discussion.

Things become more complicated as the above only provides one division of the cloud space into application areas. One can also divide the different types of cloud infrastructure into subsets depending on the interaction and ownership of the data centre itself. The usual division in this regard is:

- **Private Cloud**: This is a cloud infrastructure which is devoted to a single organisation. However, it need not be hosted by the organisation, or even located on the organisations premises. Thus the cloud infrastructure could reside anywhere.
- **Public Cloud**: A public cloud on the other hand is one whose capability is open for general use by anyone. This is probably the type of cloud most people are aware of, whether via Amazon EC2 or Microsoft Azure.
- **Community Cloud**: This is a half-way between private and public cloud, where a community of users come together to provide a common cloud infrastructure for their communities use. Examples could include those in the area of health care where hospitals pool their needs with others.
- **Hybrid Cloud**: Whereas community clouds bind end users together into a single infrastructure provider, a hybrid clouds binds different cloud infrastructures together. An example could be a corporate web site hosting, where the hosting is performed on a public cloud, but the security sensitive data processing is performed on the corporates private cloud.

5.2 Cloud Computing Security

A good overview of general security in the area of cloud computing is given in the survey article [233]. In this report we focus primarily on the *cryptographic* aspects of cloud security, but the reader is warned that this is only the tip of the iceberg. However, the fact that cloud providers (in any model) are merging user requirements into one infrastructure can enable dedicated security expertise to be applied to a wider range of end user applications. Thus the scale issue of cloud computing can provide a potential security benefit as well as realising potential security problems, see the ENISA reports [112,114].

In many cloud application areas basic cryptographic security mechanisms can be deployed to ensure operations are performed securely. For example in the IaaS model one should log into the remote service using SSH, when interacting with services in any model one should utilise connections over TLS/SSL, passwords should be strong, etc. Since at its heart cloud computing is nothing but a variant on the client-server model of computing, with some extra web front ends added for various applications, standard security as for any client-server or web application can be provided by basic cryptographic protocols.

The key aspect of cloud computing is the use of virtualized machines, where various users (called tenants) access the same physical hardware. This allows various so-called cross VM-attacks; where attackers who are co-resident on the same physical hardware extract data from one VM via the effects of aspects of the same hardware being used (usually cache based side-channels). Of course to mount such attacks an adversary needs to be co-resident on the same physical machine as the intended victim. See [272,317] for a discussion of such attacks in the cloud environment.

The use-case of cloud computing brings up its own set of problems. These are mainly due to the utility computing nature of the resource: Services are provided by third parties and not in house, and users have no real guarantee that the service provider is doing what they claim to be doing. In the next paragraphs will detail some of these issues. The cryptographic solutions to solve these issues are somewhat in their infancy, or not deployed to large extent.

Guarantee of Image: In the IaaS model a user selects a given image to install on the virtual machine. For example this could be a specified version of Ubunto Linux with certain packages already installed. This image is then loaded and the user can access the virtual machine as if it were a stand-alone machine. The problem is that the user has almost no guarantee that the provider has loaded the desired image on, and with no modifications. In addition the entire software stack up to and including the image needs to be trusted; this includes the hypervisor which is used to share resources between the different virtual machines on the same physical hardware.

Technologies exist which can address this issue, for example *Direct Anonymous Attestation (DAA)* allows one machine to attest to its configuration state to another [46, 72]. This is done by utilising zero-knowledge protocols and some special functionality designed into the TPM chip which sits on most computer motherboards. However, take up of this technology has been limited.

Proofs of Data Storage: If a cloud infrastructure is used for long term storage, say for example many years, then a client may want to know whether the data they uploaded is still there. After all a rogue provider could still charge for storage which they have deleted on the hope that the client never asks for the data back. However, the data may be so large that the client simply asking for it to be downloaded every so often for checking may not be feasible. This leads to the topic of *Proofs of Storage*. These are cryptographic protocols which enable a storage provider to prove to a client that certain files are still being stored, without the client needing to keep copies of all that has been stored.

Juels and Kaliski address the problem of how to provide data owners with formal assurance that their data still exists (and that it can be recovered in full) by introducing Proofs of Retrievability (POR) [179]. Juels and Kaliski's POR system works by first encoding a file F under some key K. The user retains the key K and the encoded file F' is stored at the server. To obtain a POR a user can query the server storing the file with a series of challenges and verify the server's responses using the key K. A similar type of protocol known as Proofs of Data Possession was proposed by Ateniese et al. [24].

Further research in this area has resulted in more efficient schemes such as the "Compact Proofs of Retrievability" by Shacham and Waters [282]. The most recent development has been in the area is Dynamic PORs [83,285]. In traditional POR schemes data is static and cannot be updated. Dynamic POR schemes provide assurance about data Retrievability whilst still allowing the data owner to perform arbitrary changes to data.

Proofs of Data Location: For many types of sensitive data, e.g. health records, it is extremely important that it remains within the appropriate legal jurisdiction. For example, Canadian law requires that health care data and other sensitive personal information is restricted to machines which are physically located in Canada. As a result, when outsourcing data to the cloud data owners may request that their data only be stored within a specific geographic location. Watson et al. [305] provide a construction for such a scheme under the assumption that data centres must declare all copies of files to their owners. Their construction is based on Proofs of Retrievability [282] and Time-Based Geolocation schemes [186]. Similar work has been carried out by Albeshri et al. [14].

5.3 Data Security in the Cloud: Future Technology

The main perceived security problem with cloud computing is that data loaded into the cloud is no longer in the control of the data creator, or the data controller in the case of data collected by an organisation and then passed to the cloud. In many cloud applications the reason for moving to the cloud is to enable computation on the said data using fast and cheap cloud infrastructure.

There is a conflict between the need to secure data and the need to process it. Simply encrypting the data and then passing it to the cloud for storage, solves only half the problem. When the data has to be retrieved, processed or searched, use of standard encryption is not useful. One either has to allow the cloud to decrypt the data, or to download the entire data set to the user's machine, thereby mitigating the benefits of moving to the cloud in the first place.

The article [182] provides a good overview of the cryptographic mechanisms which can support cloud applications focused on securing data storage (as opposed to data processing). But even if processing is not required, one needs some additional cryptographic functionality to be able to search and retrieve data which has been stored in the cloud.

There are a number of emerging technologies which aim to solve this clash of requirements. We outline just a few of either the most well-known, or the most promising.

- Fully Homomorphic Encryption (FHE)
- Multi-Party Computation (MPC)
- Searchable Encryption
- Order Preserving Encryption
- Attribute Based Encryption
- Delegated Computation

We now deal with each of these technologies in turn. Most are experimental technologies in that they are not mature enough yet for deployment, or they are only suitable for a relatively limited number of applications.

Fully Homomorphic Encryption (FHE): Probably the biggest encryption advance to attract widespread attention in the last few years has been the invention of Fully Homomorphic Encryption (FHE) by Gentry in [127,128]. The basic idea is that an arithmetic circuit can be applied to a set of ciphertexts, the result being the encryption of the output of the arithmetic circuit as if it had been evaluated on the underlying plaintexts. Since all FHE schemes are semantically secure, the ciphertext reveals no information about the plaintext (without possession of the decryption key). FHE schemes allow computations of arbitrary functions on the underlying plaintexts. In general an FHE scheme can evaluate any circuit, whereas a Somewhat Homomorphic Encryption (SHE) scheme can evaluate only circuits of bounded (multiplicative) depth, namely functions of bounded size.

In the cloud scenario the use case for FHE would be for the data owner to encrypt their data *D*, to obtain *enc(D)* and send it to the cloud provider. Then suppose the data owner wanted to perform some operation *f* on the data *D* to obtain *f(D)*. For example *D* could be a data base and *f* could be a search for all items corresponding to the person "John Doe". With FHE/SHE the cloud infrastructure can compute *enc(f(D))*, which is then returned to the data owner. The data owner then decrypts this ciphertext to obtain *f(D)*.

The construction of SHE schemes is now at a point where relatively simple circuits can be evaluated efficiently. There are a number of such schemes, of which the most efficient are those based on Ring-

LWE. For example the BGV scheme [66, 67] and the NTRU based scheme [64,217]. All the ring based SHE schemes support the concept of SIMD evaluation [66, 130, 290], which enables more efficient processing of bulk data. The other family of efficient schemes are those based on integer approximate-GCD problems [300]; this scheme has been extended to cope with SIMD processing in [85].

To turn an SHE scheme into an FHE scheme one needs a technique to bootstrap. The current state of the art in bootstrapping is far from what could be deemed truly practical. But at the time of writing the following are the key best-in-class results [18, 19, 85, 129, 243]. The standard benchmark for evaluating SHE/FHE schemes is to present run times for evaluating the AES functionality in a secure manner, see [85,131].

Multi-Party Computation (MPC): Multi-Party Computation (MPC) allows the equivalent of FHE, but with the use of multiple servers as opposed to a single one. Thus data can (theoretically) be securely outsourced to multiple cloud providers who then can compute on this data, such that one can tolerate a set of colluding adversaries (up to some bound on the number of adversaries).

The cloud scenario where this could be applied would be as follows. A data owner 'splits' their data *D* into shares *D1, … , Dn* via a secret sharing scheme. The shares are then distributed to n distinct cloud providers. The cloud providers are now able to compute any function *f(D)*, but they obtain an answer which is shared. The shares are then returned to the data owner for combining into the final solution *f(D)*. The key security concept being that as long as the data owner trusts $n - t$ out of the n servers, their data is still kept confidentially. Here t is the maximum number of adversaries which can be tolerated.

The concept of MPC has been around for a long time, with much of the foundational work dating back to the 1980's [31, 43, 84, 310, 311]. Security is usually defined to be one of passive or active security. In a passively secure protocol the adversaries are assumed to follow the protocol, and the protocol protects against privacy violations. In an actively secure protocol the adversaries can arbitrarily deviate from the protocol and privacy (and in some cases accuracy) is still preserved.

There are two families of MPC protocols; one for an arbitrary number of players based on secret sharing (which originate in [43, 84]); whilst the other are based on the concept of garbled circuits (which originate in [31,310,311]) and are focused on two parties.

In recent years there have been major advances in the capability of MPC systems. Some passively secure systems have been deployed to solve real world problems, e.g. [57, 58]. Indeed, at least three small companies Dyadic Security (Israel), Partisia (Denmark) and Cybernetica (Estonia) have products based on MPC. In addition SAP (Germany) has had a long standing research programme on using MPC for supply chain management decisions.

In terms of actively secure secret sharing based protocols, recent years have seen staggering efficiency gains in this area, see for example [45, 95, 97, 238]. We can now compute relatively complex functions f via MPC techniques, this has resulted in the interest of companies mentioned above. The protocol in [97] is notable as it uses SHE as a way of enhancing performance; which demonstrates that SHE can be used for some practical purposes already. A similar improvement has been witnessed in practical results for garbled circuit based constructions, see for example [68,156, 212–214,229] for protocol improvements and [122,200,262,283] for implementation improvements.

Searchable Encryption: Searchable Encryption aims to solve the following problem. A user outsources a database to a server, and then wants to query the server to obtain data. The first key question is what do we mean by security in this context? Clearly the database contents should remain private to the client, but should the access patterns also be private? The answer to this question is a design decision for an application developer, but it has an impact on the precise cryptographic solution one is after. To obtain privacy of the access patterns a more complex cryptographic solution is required, which we will come to below after we have discussed the basic scenario.

Searchable encryption comes in two variants; a public key and a symmetric key variant. In both variants the encrypted database is augmented with a set of tokens. Each token is associated with a keyword. The database itself is encrypted with any encryption algorithm, the searchable encryption scheme is solely used to perform the construction, and searching, of the tokens. Thus searchable encryption schemes are focused purely on searching for keywords.

The public key variants (called PEKS in the literature) are probably less useful in the cloud scenario (due to the presence of a selected keyword attack). But see [7, 63] for more details on public key variants.

Symmetric Searchable Encryption (SSE) on the other hand has seen great advances in the last few years. With improvements in both the security model and in terms of efficient constructions, see [82,92,135,183,291,301]. The paper [82] presents examples of encrypted search on a databases with over 13 million documents, equivalent to 0.4 Terrabytes of data.

To obtain privacy of the access patterns the SSE scheme needs to be augmented with something akin to Oblivious RAM (ORAM), a concept introduced by Goldreich and Ostrovsky in 1996 [139]. Recent years has seen a great advance in protocols to implement ORAM, most notably by Elaine Shi and her co-authors. See for example [96,140,141,202,216,245,247,261,284,294,295,306–308].

Order Preserving Encryption (OPE): Order Preserving Encryption (OPE) solves a variant of the searchable encryption problem in another way. It enables ciphertexts to be ordered in a way which respects the ordering of the underlying plaintexts. Thus binary search can be conducted on the ciphertexts. OPE had been originally introduced in a naive way in the database community [13]. A more scientific treatment is presented in [59, 60]. The key problem with OPE is that the security definitions are incredibly weak. It has however been used in systems such as CryptDB, see below.

Attribute Based Encryption (ABE): Attribute Based Encryption (ABE) is an extension of Identity Based Encryption where by decryption of data is allowed on the basis of whether the decryptor satisfies a set of policies associated with given secret keys, which are themselves associated to attributes. The key concept applicable to cloud scenarios is that complex access control policies for encrypted outsourced data can be embedded within the ciphertext itself. This means that a data owner does not need to trust the cloud provider to implement a policy they require.

The idea was first formally introduced by Sahai and Waters in [278], but the basic idea of using IBE to perform policy based decryption goes back a bit further, e.g. to [288]. Since its invention the literature on ABE has been enormous, with a significant body of work being attributed to Lewko, Sahai, and Waters, see for example [47,142,209–211,246,263] amongst many other papers.

ABE comes in two variants so called *Ciphertext Policy* and *Key Policy* ABE schemes, or CP- ABE and KP-ABE schemes. In CP-ABE schemes the encryptor has the list of attributes and he creates a ciphertext which can be decrypted under a policy based on these attributes. In KP-ABE schemes the policies are associated to the keys, as opposed to the ciphertexts. The decryptor can decrypt if he has a key whose underlying policy is satisfied by the given attributes.

Delegated Computation: A whole sub-area of delegated computation has arisen; mainly consisting of theoretical proof-of-concept results. The key aspects of delegated computation are the following

- A resource constrained client can delegate a computation to a powerful cloud service provider
- The client can check whether the provider has performed the correct computation.

Delegated computation can run in either a non-private manner (in which the cloud server learns the clients data), or a private manner (in which case the input data, and perhaps the function, are kept private). As remarked, almost all of the work on cryptographically secure delegated computation is of a theoretical nature at present. The key literature on this problem can be found in [80, 86, 125, 126, 249]; all of which utilise ideas from FHE, MPC and ABE in some way. A notable exception to the theoretical work is the Pinochio system of Parno et al [248] which presents a semi-practical system for verification of non-private delegated computation.

De-Duplication: Another issue which arises in secure storage on the cloud is one of secure de-duplication. Services such as Dropbox enable a user to store large amounts of data. However, to keep the costs down a cloud provider does not store multiple copies of the same file. This is important for cloud based back-up services; since many files which need to be backed up are common across many users. Thus, assuming files are not encrypted, de-duplication enables cloud providers to provide such services at a lower cost.

However, as soon as a user wants to encrypt their data the cloud provider is unable to perform de-duplication. This is because all secure encryption algorithms are probabilistic in nature. This has led to the introduction of encryption algorithms which are deterministic and data dependent, called message locked encryption [5,35]. The basic idea in this line of work is to encrypt the data under a symmetric key which is obtained as a function of the message. Thus identical messages will lead to identical ciphertexts, and the data owner needs only store the key (which is short) along with its file handle so as to be able to retrieve it (via the handle) and then decrypt it (via the key).

Summary: We summarise the above technologies in Table 6.1 to indicate whether they are ready for application in limited environments and application domains, whether more research is needed but they should be ready in the short to medium term for major deployment (and maybe limited deployment in the meantime) or whether they require a great deal more research before they are ready for practical deployment. Note, a technology can have multiple ticks to it on this classification.

Technology	Ready for Deployment	Short Term Research Needed	Longer Term Research Needed
Fully Homomorphic Encryption	×	×	√
Multi-Party Computation	√	√	×
Searchable Encryption	√	√	×
Order Preserving Encryption	×	×	√
Attribute Based Encryption	×	√	×
Delegated Computation	×	×	√
Message Locked Encryption	√	√	×

Table 5.1: Summary of technology readiness of advanced cryptographic mechanisms

5.3.1. CryptDB

In [264] Popa et al present a realisation of a database which can be queried using a restricted set of SQL queries, but for which the database entries are encrypted. The resulting system, called CryptDB, is relatively efficient and enables a large enough set of standard SQL queries to be executed with a reasonable delay compared to computing with an unencrypted database. The scenario is that a data holder is aiming to outsource their database to a third party cloud provider; whilst wanting to maintain database security as well as maintain a reasonable amount of standard SQL functionality.

The basic idea behind CryptDB is to take a sensitive column in the database and then to encrypt it using successively stronger forms of encryption; such as standard encryption, SSE and OPE. The system is efficient, but suffers from inherent leakage of information. This means that CryptDB essentially only provides a layer of weak security to an outsourced database application, as opposed to providing truly strong security.

A similar idea to CryptDB is the SEEED system of SAP [143], which extends the SQL syntax of CryptDB to support more elaborate queries.

Bibliography

[1] Technical Specificaiton Group Services 3rd Generation Parternship Project and 3G Security System Aspects. Confidentiality and integrity algorithms UEA1 & UIA1. Document 1: UEA1 and UIA1 specifications.

[2] Technical Specificaiton Group Services 3rd Generation Parternship Project and 3G Security System Aspects. Confidentiality and integrity algorithms UEA2 & UIA2. Document 1: UEA2 and UIA2 specifications.

[3] Technical Specificaiton Group Services 3rd Generation Parternship Project and 3G Security System Aspects. Confidentiality and integrity algorithms UEA2 & UIA2. Document 2: SNOW 3G specification.

[4] Technical Specificaiton Group Services 3rd Generation Parternship Project and 3G Security System Aspects. Specification of the 3GPP confidentiality and integrity algorithms; Docu- ment 2: KASUMI specification, v.3.1.1.

[5] Mart́ ın Abadi, Dan Boneh, Ilya Mironov, Ananth Raghunathan, and Gil Segev. Message-locked encryption for lock-dependent messages. In Canetti and Garay [76], pages 374–391.

[6] Mart́ ın Abadi and Phillip Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). In Proc. 1st IFIP International Conference on Theoretical Computer Science (IFIP–TCS’00), volume 1872 of Lecture Notes in Computer Science, pages 3–22, 2000.

[7] Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. J. Cryptology, 21(3):350–391, 2008.

[8] Michel Abdalla, Olivier Chevassut, and David Pointcheval. One-time verifier-based encrypted key exchange. In Serge Vaudenay, editor, Public Key Cryptography, volume 3386 of Lecture Notes in Computer Science, pages 47–64. Springer, 2005.

[9] Michel Abdalla and David Pointcheval. Simple password-based encrypted key exchange proto- cols. In Alfred Menezes, editor, CT-RSA, volume 3376 of Lecture Notes in Computer Science, pages 191–208. Springer, 2005.

[10] Michel Abdalla and David Pointcheval. A scalable password-based group key exchange protocol in the standard model. In Xuejia Lai and Kefei Chen, editors, ASIACRYPT, volume 4284 of Lecture Notes in Computer Science, pages 332–347. Springer, 2006.

[11] Ben Adida, Mike Bond, Jolyon Clulow, Amerson Lin, Ross J. Anderson, and Ronald L. Rivest. On the security of the EMV secure messaging API (extended abstract). In Bruce Christianson, Bruno Crispo, James A. Malcolm, and Michael Roe, editors, Security Protocols Workshop, volume 5964 of Lecture Notes in Computer Science, pages 147–149. Springer, 2007.

[12] Ben Adida, Mike Bond, Jolyon Clulow, Amerson Lin, Steven J. Murdoch, Ross J. Anderson, and Ronald L. Rivest. Phish and chips. In Bruce Christianson, Bruno Crispo, James A. Malcolm, and Michael Roe, editors, Security Protocols Workshop, volume 5087 of Lecture Notes in Computer Science, pages 40–48. Springer, 2006.

[13] Rakesh Agrawal, Jerry Kiernan, Ramakrishnan Srikant, and Yirong Xu. Order-preserving encryption for numeric data. In Gerhard Weikum, Arnd Christian K ̈onig, and Stefan Deßloch, editors, SIGMOD Conference, pages 563–574. ACM, 2004.

[14] Aiiad Albeshri, Colin Boyd, and Juan Manuel Gonź alez Nieto. Enhanced geoproof: improved geographic assurance for data in the cloud. Int. J. Inf. Sec., 13(2):191–198, 2014.

[15] Martin R. Albrecht, Kenneth G. Paterson, and Gaven J. Watson. Plaintext recovery attacks against SSH. In IEEE Symposium on Security and Privacy, pages 16–26. IEEE Computer Society, 2009.

[16] Nadhem J. AlFardan, Daniel J. Bernstein, Kenneth G. Paterson, Bertram Poettering, and Jacob C. N. Schuldt. On the security of RC4 in TLS. In Samuel T. King, editor, USENIX Security, pages 305–320. USENIX Association, 2013.

[17] Nadhem J. AlFardan and Kenneth G. Paterson. Lucky Thirteen: Breaking the TLS and DTLS Record Protocols. In IEEE Symposium on Security and Privacy. IEEE Computer Society, 2013.

[18] Jacob Alperin-Sheriff and Chris Peikert. Practical bootstrapping in quasilinear time. In Canetti and Garay [76], pages 1–20.

[19] Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In Juan A. Garay and Rosario Gennaro, editors, CRYPTO (1), volume 8616 of Lecture Notes in Computer Science, pages 297–314. Springer, 2014.

[20] ANSI X9.42. Agreement of symmetric keys using discrete logarithm cryptography. American National Standard Institute, 2005.

[21] ANSI X9.42. Key agreement and key transport using factoring-based cryptography. American National Standard Institute, 2005.

[22] ANSI X9.63. Public key cryptography for the financial services industry – Key agreement and key transport using elliptic curve cryptography. American National Standard Institute, 2011.

[23] Myrto Arapinis, Loretta Ilaria Mancini, Eike Ritter, and Mark Ryan. Formal analysis of UMTS privacy. CoRR, abs/1109.2066, 2011.

[24] Giuseppe Ateniese, Randal C. Burns, Reza Curtmola, Joseph Herring, Osama Khan, Lea Kissner, Zachary N. J. Peterson, and Dawn Song. Remote data checking using provable data possession. ACM Trans. Inf. Syst. Secur., 14(1):12, 2011.

[25] Dirk Balfanz, Diana K. Smetters, Paul Stewart, and H. Chi Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In NDSS. The Internet Society, 2002.

[26] Boaz Barak, Ran Canetti, Yehuda Lindell, Rafael Pass, and Tal Rabin. Secure computation without authentication. J. Cryptology, 24(4):720–760, 2011.

[27] Gilles Barthe, Benjamin Gŕ egoire, Sylvain Heraud, and Santiago Zanella B́ eguelin. Computer- aided security proofs for the working cryptographer. In Rogaway [274], pages 71–90.

[28] Gilles Barthe, Benjamin Gŕ egoire, and Santiago Zanella-B́ eguelin. Formal certification of code-based cryptographic proofs. In 36th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2009, pages 90–101. ACM, 2009.

[29] David A. Basin, Cas Cremers, and Simon Meier. Provably repairing the iso/iec 9798 standard for entity authentication. Journal of Computer Security, 21(6):817–846, 2013.

[30] David A. Basin, Cas J. F. Cremers, and Simon Meier. Provably repairing the iso/iec 9798 standard for entity authentication. In Pierpaolo Degano and Joshua D. Guttman, editors, POST, volume 7215 of Lecture Notes in Computer Science, pages 129–148. Springer, 2012.

[31] Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). In Ortiz [244], pages 503–513.

[32] M. Bellare, T. Kohno, and C. Namprempre. The Secure Shell (SSH) Transport Layer En- cryption Modes. RFC 4344 (Proposed Standard), January 2006.

[33] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols (extended abstract). In Jeffrey Scott Vitter, editor, STOC, pages 419–428. ACM, 1998.

[34] Mihir Bellare, Marc Fischlin, Shafi Goldwasser, and Silvio Micali. Identification protocols secure against reset attacks. In Pfitzmann [260], pages 495–511.

[35] Mihir Bellare, Sriram Keelveedhi, and Thomas Ristenpart. Message-locked encryption and secure deduplication. In Johansson and Nguyen [178], pages 296–312.

[36] Mihir Bellare, Tadayoshi Kohno, and Chanathip Namprempre. Breaking and provably re- pairing the SSH authenticated encryption scheme: A case study of the Encode-then-Encrypt- and-MAC paradigm. ACM Trans. Inf. Syst. Secur., 7(2):206–241, 2004.

[37] Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated key exchange secure against dictionary attacks. In Preneel [266], pages 139–155.

[38] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In Douglas R. Stinson, editor, CRYPTO, volume 773 of Lecture Notes in Computer Science, pages 232–249. Springer, 1993.

[39] S. M. Bellovin and M. Merritt. Limitations of the Kerberos authentication system. SIGCOMM Comput. Commun. Rev., 20(5):119–132, October 1990.

[40] Steven M. Bellovin. Problem areas for the IP security protocols. In Proceedings of the Sixth Usenix Unix Security Symposium, pages 1–16, 1996.

[41] Steven M. Bellovin and Michael Merritt. Encrypted key exchange: Password-based protocols secureagainst dictionary attacks. In Proceedings of the 1992 IEEE Symposium on Security and Privacy, SP ’92, pages 72–, Washington, DC, USA, 1992. IEEE Computer Society.

[42] Steven M. Bellovin and Michael Merritt. Augmented encrypted key exchange: A password- based protocol secure against dictionary attacks and password file compromise. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, ACM Conference on Computer and Communications Security, pages 244–250. ACM, 1993.

[43] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non- cryptographic fault-tolerant distributed computation (extended abstract). In Simon [287], pages 1–10.

[44] Jens Bender, Marc Fischlin, and Dennis Ku¨gler. Security analysis of the pace key-agreement protocol. In Pierangela Samarati, Moti Yung, Fabio Martinelli, and Claudio Agostino Ardagna, editors, ISC, volume 5735 of Lecture Notes in Computer Science, pages 33–48. Springer, 2009.

[45] Rikke Bendlin, Ivan Damg˚ard, Claudio Orlandi, and Sarah Zakarias. Semi-homomorphic encryption and multiparty computation. In Paterson [253], pages 169–188.

[46] David Bernhard, Georg Fuchsbauer, Essam Ghadafi, Nigel P. Smart, and Bogdan Warinschi. Anonymous attestation with user-controlled linkability. Int. J. Inf. Sec., 12(3):219–249, 2013.

[47] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In IEEE Symposium on Security and Privacy [161], pages 321–334.

[48] D. Bider and M. Baushke. SHA-2 Data Integrity Verification for the Secure Shell (SSH) Transport Layer Protocol. RFC 6668 (Proposed Standard), July 2012.

[49] Eli Biham, Orr Dunkelman, and Nathan Keller. A related-key rectangle attack on the full KASUMI. In Bimal K. Roy, editor, ASIACRYPT, volume 3788 of Lecture Notes in Computer Science, pages 443–461. Springer, 2005.

[50] Alex Biryukov, Deike Priemuth-Schmid, and Bin Zhang. Multiset collision attacks on reduced- round SNOW 3G and SNOW 3G (+) . In Zhou and Yung [318], pages 139–153.

[51] Andrea Bittau, Mark Handley, and Joshua Lackey. The final nail in WEP’s coffin. In IEEE Symposium on Security and Privacy, pages 386–400. IEEE Computer Society, 2006.

[52] S. Blake-Wilson, N. Bolyard, V. Gupta, C. Hawk, and B. Moeller. Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS). RFC 4492 (Informational), May 2006. Updated by RFC 5246.

[53] Simon Blake-Wilson, Don Johnson, and Alfred Menezes. Key agreement protocols and their security analysis. In Michael Darnell, editor, IMA Int. Conf., volume 1355 of Lecture Notes in Computer Science, pages 30–45. Springer, 1997.

[54] Bruno Blanchet. An efficient cryptographic protocol verifier based on prolog rules. In CSFW, pages 82–96. IEEE Computer Society, 2001.

[55] Bruno Blanchet and David Pointcheval. Automated security proofs with sequences of games. In Dwork [109], pages 537–554.

[56] Daniel Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In Hugo Krawczyk, editor, CRYPTO, volume 1462 of Lecture Notes in Computer Science, pages 1–12. Springer, 1998.

[57] Dan Bogdanov, Sven Laur, and Jan Willemson. Sharemind: A framework for fast privacy- preserving computations. In Sushil Jajodia and Javier L´opez, editors, ESORICS, volume 5283 of Lecture Notes in Computer Science, pages 192–206. Springer, 2008.

[58] Peter Bogetoft, Dan Lund Christensen, Ivan Damg˚ard, Martin Geisler, Thomas P. Jakobsen, Mikkel Krøigaard, Janus Dam Nielsen, Jesper Buus Nielsen, Kurt Nielsen, Jakob Pagter, Michael I. Schwartzbach, and Tomas Toft. Secure multiparty computation goes live. In Dingledine and Golle [104], pages 325–343.

[59] Alexandra Boldyreva, Nathan Chenette, Younho Lee, and Adam O'Neill. Order-preserving symmetric encryption. In Antoine Joux, editor, EUROCRYPT, volume 5479 of Lecture Notes in Computer Science, pages 224–241. Springer, 2009.

[60] Alexandra Boldyreva, Nathan Chenette, and Adam O'Neill. Order-preserving encryption revisited: Improved security analysis and alternative solutions. In Rogaway [274], pages 578–595.

[61] Alexandra Boldyreva and Virendra Kumar. Provable-security analysis of authenticated encryption in Kerberos. IET Information Security, 5(4):207–219, 2011.

[62] Mike Bond, Omar Choudary, Steven J. Murdoch, Sergei P. Skorobogatov, and Ross J. Anderson. Chip and skim: Cloning EMV cards with the pre-play attack. CoRR, abs/1209.2531, 2012.

[63] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In Christian Cachin and Jan Camenisch, editors, EURO- CRYPT, volume 3027 of Lecture Notes in Computer Science, pages 506–522. Springer, 2004.

[64] Joppe W. Bos, Kristin Lauter, Jake Loftus, and Michael Naehrig. Improved security for a ring-based fully homomorphic encryption scheme. In Martijn Stam, editor, IMA Int. Conf., volume 8308 of Lecture Notes in Computer Science, pages 45–64. Springer, 2013.

[65] Victor Boyko, Philip D. MacKenzie, and Sarvar Patel. Provably secure password- authenticated key exchange using diffie-hellman. In Preneel [266], pages 156–171.

[66] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, ITCS, pages 309–325. ACM, 2012.

[67] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In Rogaway [274], pages 505–524.

[68] Lu´ıs T. A. N. Brand˜ao. Secure two-party computation with reusable bit-commitments, via a cut-and-choose with forge-and-lose technique - (extended abstract). In Kazue Sako and Palash Sarkar, editors, ASIACRYPT (2), volume 8270 of Lecture Notes in Computer Science, pages 441–463. Springer, 2013.

[69] Jørgen Brandt, Ivan Damg˚ard, Peter Landrock, and Torben P. Pedersen. Zero-knowledge authentication scheme with secret key exchange (extended abstract). In Shafi Goldwasser, editor, CRYPTO, volume 403 of Lecture Notes in Computer Science, pages 583–588. Springer, 1988.

[70] Emmanuel Bresson, Olivier Chevassut, and David Pointcheval. Security proofs for an efficient password-based key exchange. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, ACM Conference on Computer and Communications Security, pages 241–250. ACM, 2003.

[71] Emmanuel Bresson, Olivier Chevassut, and David Pointcheval. New security results on encrypted key exchange. In Feng Bao, Robert H. Deng, and Jianying Zhou, editors, Public Key Cryptography, volume 2947 of Lecture Notes in Computer Science, pages 145–158. Springer, 2004.

[72] Ernest F. Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In Vi- jayalakshmi Atluri, Birgit Pfitzmann, and Patrick Drew McDaniel, editors, ACM Conference on Computer and Communications Security, pages 132–145. ACM, 2004.

[73] Christina Brzuska, Marc Fischlin, Nigel P. Smart, Bogdan Warinschi, and Stephen C. Williams. Less is more: relaxed yet composable security notions for key exchange. Int. J. Inf. Sec., 12(4):267–297, 2013.

[74] Christina Brzuska, Nigel P. Smart, Bogdan Warinschi, and Gaven J. Watson. An analysis of the emv channel establishment protocol. In Sadeghi et al. [276], pages 373–386.

[75] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In FOCS, pages 136–145. IEEE Computer Society, 2001.

[76] Ran Canetti and Juan A. Garay, editors. Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I, volume 8042 of Lecture Notes in Computer Science. Springer, 2013.

[77] Ran Canetti and Juan A. Garay, editors. Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II, volume 8043 of Lecture Notes in Computer Science. Springer, 2013.

[78] Ran Canetti, Shai Halevi, Jonathan Katz, Yehuda Lindell, and Philip D. MacKenzie. Universally composable password-based key exchange. In Cramer [90], pages 404–421.

[79] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In Pfitzmann [260], pages 453–474.

[80] Ran Canetti, Ben Riva, and Guy N. Rothblum. Practical delegation of computation using multiple servers. In Yan Chen, George Danezis, and Vitaly Shmatikov, editors, ACM Conference on Computer and Communications Security, pages 445–454. ACM, 2011.

[81] Brice Canvel, Alain P. Hiltgen, Serge Vaudenay, and Martin Vuagnoux. Password interception in a SSL/TLS channel. In Dan Boneh, editor, CRYPTO, volume 2729 of Lecture Notes in Computer Science, pages 583–599. Springer, 2003.

[82] David Cash, Stanislaw Jarecki, Charanjit S. Jutla, Hugo Krawczyk, Marcel-Catalin Rosu, and Michael Steiner. Highly-scalable searchable symmetric encryption with support for boolean queries. In Canetti and Garay [76], pages 353–373.

[83] David Cash, Alptekin Ku¨p¸cu¨, and Daniel Wichs. Dynamic proofs of retrievability via oblivious ram. In Johansson and Nguyen [178], pages 279–295.

[84] David Chaum, Claude Cŕepeau, and Ivan Damg̊ard. Multiparty unconditionally secure protocols (extended abstract). In Simon [287], pages 11–19.

[85] Jung Hee Cheon, Jean-Ś ebastien Coron, Jinsu Kim, Moon Sung Lee, Tancr`ede Lepoint, Mehdi Tibouchi, and Aaram Yun. Batch fully homomorphic encryption over the integers. In Johansson and Nguyen [178], pages 315–335.

[86] Kai-Min Chung, Yael Tauman Kalai, and Salil P. Vadhan. Improved delegation of computation using fully homomorphic encryption. In Rabin [268], pages 483–501.

[87] Codenomicon. The Heartbleed bug. http://heartbleed.com/, 2014.

[88] Hubert Comon-Lundh and Vitaly Shmatikov. Intruder deductions, constraint solving and insecurity decision in presence of exclusive or. In LICS, pages 271–. IEEE Computer Society, 2003.

[89] V́eronique Cortier, Steve Kremer, and Bogdan Warinschi. A survey of symbolic methods in computational analysis of cryptographic systems. J. Autom. Reasoning, 46(3-4):225–259, 2011.

[90] Ronald Cramer, editor. Advances in Cryptology - EUROCRYPT 2005, 24th Annual Inter- national Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings, volume 3494 of Lecture Notes in Computer Science. Springer, 2005.

[91] Cas J. F. Cremers. The scyther tool: Verification, falsification, and analysis of security protocols. In Aarti Gupta and Sharad Malik, editors, CAV, volume 5123 of Lecture Notes in Computer Science, pages 414–418. Springer, 2008.

[92] Reza Curtmola, Juan A. Garay, Seny Kamara, and Rafail Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. In Juels et al. [180], pages 79–88.

[93] F. Cusack and M. Forssen. Generic Message Exchange Authentication for the Secure Shell Protocol (SSH). RFC 4256 (Proposed Standard), January 2006.

[94] W. Dai. An attack against SSH2 protocol. E-mail to the SECSH Working Group available from ftp://ftp.ietf.org/ietf-mail-archive/secsh/2002-02.mail, 6th Feb. 2002.

[95] Ivan Damg˚ard, Martin Geisler, Mikkel Krøigaard, and Jesper Buus Nielsen. Asynchronous multiparty computation: Theory and implementation. In Stanislaw Jarecki and Gene Tsudik, editors, Public Key Cryptography, volume 5443 of Lecture Notes in Computer Science, pages 160–179. Springer, 2009.

[96] Ivan Damg˚ard, Sigurd Meldgaard, and Jesper Buus Nielsen. Perfectly secure oblivious ram without random oracles. In Yuval Ishai, editor, TCC, volume 6597 of Lecture Notes in Computer Science, pages 144–163. Springer, 2011.

[97] Ivan Damg˚ard, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In Safavi-Naini and Canetti [277], pages 643–662.

[98] Jean Paul Degabriele, Anja Lehmann, Kenneth G. Paterson, Nigel P. Smart, and Mario Strefler. On the joint security of encryption and signature in EMV. In Orr Dunkelman, editor, CT-RSA, volume 7178 of Lecture Notes in Computer Science, pages 116–135. Springer, 2012.

[99] Jean Paul Degabriele and Kenneth G. Paterson. Attacking the IPsec standards in encryption- only configurations. In IEEE Symposium on Security and Privacy [161], pages 335–349.

[100] Jean Paul Degabriele and Kenneth G. Paterson. On the (in)security of IPsec in MAC-then- encrypt configurations. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, ACM Conference on Computer and Communications Security, pages 493–504. ACM, 2010.

[101] T. Dierks and C. Allen. The TLS Protocol Version 1.0. RFC 2246 (Proposed Standard), January 1999. Obsoleted by RFC 4346, updated by RFCs 3546, 5746, 6176.

[102] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.1. RFC 4346 (Proposed Standard), April 2006. Obsoleted by RFC 5246, updated by RFCs 4366, 4680, 4681, 5746, 6176.

[103] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), August 2008. Updated by RFCs 5746, 5878, 6176.

[104] Roger Dingledine and Philippe Golle, editors. Financial Cryptography and Data Security, 13th International Conference, FC 2009, Accra Beach, Barbados, February 23-26, 2009. Revised Selected Papers, volume 5628 of Lecture Notes in Computer Science. Springer, 2009.

[105] Danny Dolev and Andrew Chi-Chih Yao. On the security of public key protocols (extended abstract). In FOCS, pages 350–357. IEEE Computer Society, 1981.

[106] Saar Drimer, Steven J. Murdoch, and Ross J. Anderson. Optimised to fail: Card readers for online banking. In Dingledine and Golle [104], pages 184–200.

[107] T. Duong and J. Rizzo. Here come the $\oplus$ ninjas. Unpublished manuscript, 2011.

[108] Thai Duong and Juliano Rizzo. The CRIME attack. Presentation at ekoparty Security Conference, 2012.

[109] Cynthia Dwork, editor. Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006, Proceedings, volume 4117 of Lecture Notes in Computer Science. Springer, 2006.

[110] D. Eastlake. Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH). RFC 4305 (Proposed Standard), December 2005. Obsoleted by RFC 4835.

[111] EMV Co. Book 2 – Security and key management. EMV 4.3, 2011.

[112] ENISA. Cloud Computing Risk Assessment. https://www.enisa.europa.eu/activities/risk-management/files/deliverables/cloud-computing-risk-assessment, 2009.

[113] ENISA. Algorithms, key size and parameters report – 2013 recommendations, http://www.enisa.europa.eu/activities/identity-and-trust/library/deliverables/algorithms-key-sizes-and-parameters-report, 2013.

[114] ENISA. Critical Cloud Computing – A CIIP perspective on cloud computing services. https://www.enisa.europa.eu/activities/Resilience-and-CIIP/cloud-computing/ critical-cloud-computing, 2013.

[115] ENISA. Algorithms, key size and parameters report – 2014, ISBN 978-92-9204-102-1, DOI 10.2824/36822, available at: https://www.enisa.europa.eu/activities/identity-and-trust/library/deliverables/algorithms-key-size-and-parameters-report-2014/, 2014.

[116] Niels Ferguson and Bruce Schneier. A cryptographic evaluation of IPsec. Unpublished manuscript available from http://www.schneier.com/paper-ipsec.html, February 1999.

[117] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, CRYPTO, volume 263 of Lecture Notes in Computer Science, pages 186–194. Springer, 1986.

[118] Scott R. Fluhrer and Stefan Lucks. Analysis of the E0 encryption system. In Vaudenay and Youssef [304], pages 38–48.

[119] Scott R. Fluhrer, Itsik Mantin, and Adi Shamir. Weaknesses in the key scheduling algorithm of RC4. In Vaudenay and Youssef [304], pages 1–24.

[120] S. Frankel, R. Glenn, and S. Kelly. The AES-CBC Cipher Algorithm and Its Use with IPsec. RFC 3602 (Proposed Standard), September 2003.

[121] S. Frankel and H. Herbert. The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec. RFC 3566 (Proposed Standard), September 2003.

[122] Tore Kasper Frederiksen and Jesper Buus Nielsen. Fast and maliciously secure two-party computation using the gpu. In Michael J. Jacobson Jr., Michael E. Locasto, Payman Mohas- sel, and Reihaneh Safavi-Naini, editors, ACNS, volume 7954 of Lecture Notes in Computer Science, pages 339–356. Springer, 2013.

[123] M. Friedl, N. Provos, and W. Simpson. Diffie-Hellman Group Exchange for the Secure Shell (SSH) Transport Layer Protocol. RFC 4419 (Proposed Standard), March 2006.

[124] D. Fu and J. Solinas. Elliptic Curve Groups modulo a Prime (ECP Groups) for IKE and IKEv2. RFC 5903 (Informational), June 2010.

[125] Rosario Gennaro, Craig Gentry, and Bryan Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In Rabin [268], pages 465–482.

[126] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span pro- grams and succinct nizks without pcps. In Johansson and Nguyen [178], pages 626–645.

[127] Craig Gentry. A fully homomorphic encryption scheme. PhD Thesis, Stanford, 2009.

[128] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher,

editor, STOC, pages 169–178. ACM, 2009.

[129] Craig Gentry, Shai Halevi, and Nigel P. Smart. Better bootstrapping in fully homomorphic encryption. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, Public Key Cryptography, volume 7293 of Lecture Notes in Computer Science, pages 1–16. Springer, 2012.

[130] Craig Gentry, Shai Halevi, and Nigel P. Smart. Fully homomorphic encryption with polylog overhead. In David Pointcheval and Thomas Johansson, editors, EUROCRYPT, volume 7237 of Lecture Notes in Computer Science, pages 465–482. Springer, 2012.

[131] Craig Gentry, Shai Halevi, and Nigel P. Smart. Homomorphic evaluation of the aes circuit. In Safavi-Naini and Canetti [277], pages 850–867.

[132] Henri Gilbert, editor. Advances in Cryptology - EUROCRYPT 2010, 29th Annual Inter- national Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings, volume 6110 of Lecture Notes in Computer Science. Springer, 2010.

[133] Marc Girault. Self-certified public keys. In Donald W. Davies, editor, EUROCRYPT, volume 547 of Lecture Notes in Computer Science, pages 490–497. Springer, 1991.

[134] Marc Girault and Jean-Claude Pailĺ es. On-line/off-line RSA-like. In Proceedings of WCC 2003, pages 173–184, 2003.

[135] Eu-Jin Goh. Secure indexes. IACR Cryptology ePrint Archive, 2003:216, 2003.

[136] Ian Goldberg, Atefeh Mashatan, and Douglas R. Stinson. On message recognition protocols: recoverability and explicit confirmation. IJACT, 2(2):100–120, 2010.

[137] Oded Goldreich and Yehuda Lindell. Session-key generation using human passwords only. J. of Cryptology, 19(3):241–340, 2006.

[138] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In Alfred V. Aho, editor, STOC, pages 218–229. ACM, 1987.

[139] Oded Goldreich and Rafail Ostrovsky. Software protection and simulation on oblivious rams. J. ACM, 43(3):431–473, 1996.

[140] Michael T. Goodrich and Michael Mitzenmacher. Privacy-preserving access of outsourced data via oblivious ram simulation. In Luca Aceto, Monika Henzinger, and Jiri Sgall, editors, ICALP (2), volume 6756 of Lecture Notes in Computer Science, pages 576–587. Springer, 2011.

[141] Michael T. Goodrich, Michael Mitzenmacher, Olga Ohrimenko, and Roberto Tamassia. Privacy-preserving group data access via stateless oblivious ram simulation. In Rabani [267], pages 157–167.

[142] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Juels et al. [180], pages 89–98.

[143] Patrick Grofig, Martin H˙arterich, Isabelle Hang, Florian Kerschbaum, Mathias Kohler, Andreas Schaad, Axel Schr¨opfer, and Walter Tighzert. Experiences and observations on the industrial implementation of a system to search over outsourced encrypted data. In Stefan Katzenbeisser, Volkmar Lotz, and Edgar R. Weippl, editors, Sicherheit, volume 228 of LNI, pages 115–125. GI, 2014.

[144] Louis C. Guillou and Jean-Jacques Quisquater. A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In Christoph G. Gu˙nther, editor, EUROCRYPT, volume 330 of Lecture Notes in Computer Science, pages 123–128. Springer, 1988.

[145] Finn Michael Halvorsen, Olav Haugen, Martin Eian, and Stig Fr. Mjølsnes. An improved attack on TKIP. In Audun Jøsang, Torleiv Maseng, and Svein J. Knapskog, editors, NordSec, volume 5838 of Lecture Notes in Computer Science, pages 120–132. Springer, 2009.

[146] Feng Hao and Peter Ryan. J-pake: Authenticated key exchange without pki. Transactions on Computational Science, 11:192–206, 2010.

[147] Feng Hao and Peter Y. A. Ryan. Password authenticated key exchange by juggling. In Bruce Christianson, James A. Malcolm, Vashek Matyas, and Michael Roe, editors, Security Protocols Workshop, volume 6615 of Lecture Notes in Computer Science, pages 159–171. Springer, 2008.

[148] D. Harkins and G.Zorn. Extensible authentication protocol (eap) authentication using only a password. RFC 5931 (Informational), 2010.

[149] Dan Harkins. Simultaneous authentication of equals: A secure, password-based key exchange for mesh networks. In Proceedings of the 2008 Second International Conference on Sensor Technologies and Applications, SENSORCOMM ’08, pages 839–844, Washington, DC, USA, 2008. IEEE Computer Society.

[150] B. Harris. RSA Key Exchange for the Secure Shell (SSH) Transport Layer Protocol. RFC 4432 (Proposed Standard), March 2006.

[151] Nadia Heninger, Zakir Durumeric, Eric Wustrow, and J.Alex Halderman. Mining your Ps and Qs: Detection of widespread weak keys in network devices. In USENIX Security Symposium – 2012, pages 205–220, 2012.

[152] Miia Hermelin and Kaisa Nyberg. Correlation properties of the Bluetooth combiner generator. In JooSeok Song, editor, ICISC, volume 1787 of Lecture Notes in Computer Science, pages 17–29. Springer, 1999.

[153] P. Hoffman. Cryptographic Suites for IPsec. RFC 4308 (Proposed Standard), December 2005.

[154] R. Housley. Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP). RFC 3686 (Proposed Standard), January 2004.

[155] R. Housley. Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP). RFC 4309 (Proposed Standard), December 2005.

[156] Yan Huang, Jonathan Katz, and David Evans. Efficient secure two-party computation using symmetric cut-and-choose. In Canetti and Garay [77], pages 18–35.

[157] G. Hudson. Camellia Encryption for Kerberos 5. RFC 6803 (Informational), November 2012.

[158] IEEE 802.11. Wireless LAN medium access control MAC and physical layer PHY specifications. Institute of Electrical and Electronics Engineers Standard, 1999.

[159] IEEE 802.11-2012 (Revision of IEEE 802.11-2007). Wireless LAN medium access control MAC and physical layer PHY specifications. Institute of Electrical and Electronics Engineers Standard, 2012.

[160] IEEE 802.15.4 . Law rate WPAN. Institute of Electrical and Electronics Engineers Standard, 2012.

[161] IEEE Computer Society. 2007 IEEE Symposium on Security and Privacy (S&P 2007), 20-23 May 2007, Oakland, California, USA, 2007.

[162] K. Igoe and J. Solinas. AES Galois Counter Mode for the Secure Shell Transport Layer Protocol. RFC 5647 (Informational), August 2009.

[163] ISO/IEC 11770-2. Information technology – Security techniques – Key management – Part 2: Mechanisms using symmetric techniques. International Organization for Standardization, 2008.

[164] ISO/IEC 11770-3. Information technology – Security techniques – Key management – Part 3: Mechanisms using asymmetric techniques. International Organization for Standardization, 2008.

[165] ISO/IEC 11770-4. Information technology – Security techniques – Key management – Part 3: Mechanisms based on weak secrets. International Organization for Standardization, 2006.

[166] ISO/IEC 29192-4. Information technology – Security techniques – Lightweight cryptography – Part 4: Mechanisms using asymmetric techniques. International Organization for Standardization, 2013.

[167] ISO/IEC 9796-2. Information technology – Security techniques – Digital signatures giving message recovery – Part 2: Integer factorization based schemes. International Organization for Standardization, 2010.

[168] ISO/IEC 9798-1. Information technology – Security techniques – Entity authentication – Part 1: General. International Organization for Standardization, 2010.

[169] ISO/IEC 9798-2. Information technology – Security techniques – Entity authentication – Part 2: Mechanisms using symmetric encipherment techniques. International Organization for Standardization, 2008.

[170] ISO/IEC 9798-3. Information technology – Security techniques – Entity authentication – Part 3: Mechanisms using digital signature techniques. International Organization for Standardization, 1998.

[171] ISO/IEC 9798-4. Information technology – Security techniques – Entity authentication – Part 4: Mechanisms using a cryptographic check function. International Organization for Standardization, 1999.

[172] ISO/IEC 9798-5. Information technology – Security techniques – Entity authentication – Part 5: Mechanisms using zero-knowledge techniques. International Organization for Standardization, 2009.

[173] ISO/IEC 9798-6. Information technology – Security techniques – Entity authentication – Part 6: Mechanisms using manual data transfer. International Organization for Standardization, 2010.

[174] David P. Jablon. Extended password key exchange protocols immune to dictionary attacks. In WETICE, pages 248–255. IEEE Computer Society, 1997.

[175] David P. Jablon and Westboro Ma. Strong password-only authenticated key exchange. ACM Computer Communications Review, 26:5–26, 1996.

[176] Tibor Jager, Florian Kohlar, Sven Sch¨age, and J˙org Schwenk. On the security of TLS-DHE in the standard model. In Safavi-Naini and Canetti [277], pages 273–293.

[177] Tibor Jager, Florian Kohlar, Sven Sch¨age, and J˙org Schwenk. On the security of tls-dhe in the standard model. In Safavi-Naini and Canetti [277], pages 273–293.

[178] Thomas Johansson and Phong Q. Nguyen, editors. Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings, volume 7881 of Lecture Notes in Computer Science. Springer, 2013.

[179] Ari Juels and Burton S. Kaliski Jr. Pors: proofs of retrievability for large files. In Ning et al. [239], pages 584–597.

[180] Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors. Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, Ioctober 30 - November 3, 2006. ACM, 2006.

[181] Ronald Kainda, Ivan Flechais, and A. W. Roscoe. Usability and security of out-of-band channels in secure device pairing protocols. In Lorrie Faith Cranor, editor, SOUPS, ACM International Conference Proceeding Series. ACM, 2009.

[182] Seny Kamara and Kristin Lauter. Cryptographic cloud storage. In Radu Sion, Reza Curtmola, Sven Dietrich, Aggelos Kiayias, Josep M. Miret, Kazue Sako, and Francesc Seb´e, editors, Financial Cryptography Workshops, volume 6054 of Lecture Notes in Computer Science, pages 136–149. Springer, 2010.

[183] Seny Kamara, Charalampos Papamanthou, and Tom Roeder. Dynamic searchable symmetric encryption. In Yu et al. [315], pages 965–976.

[184] A. Kato, M. Kanda, and S. Kanno. Modes of Operation for Camellia for Use with IPsec. RFC 5529 (Proposed Standard), April 2009.

[185] Jonathan Katz, Rafail Ostrovsky, and Moti Yung. Efficient and secure authenticated key exchange using weak passwords. J. ACM, 57(1), 2009.

[186] Ethan Katz-Bassett, John P. John, Arvind Krishnamurthy, David Wetherall, Thomas E. Anderson, and Yatin Chawathe. Towards ip geolocation using delay and topology measurements. In Jussara M. Almeida, Virg´ılio A. F. Almeida, and Paul Barford, editors, Internet Measurement Conference, pages 71–84. ACM, 2006.

[187] C. Kaufman. Internet Key Exchange (IKEv2) Protocol. RFC 4306 (Proposed Standard), December 2005. Obsoleted by RFC 5996, updated by RFC 5282.

[188] C. Kaufman, P. Hoffman, Y. Nir, and P. Eronen. Internet Key Exchange Protocol Version 2 (IKEv2). RFC 5996 (Proposed Standard), September 2010. Updated by RFC 5998.

[189] S. Kelly and S. Frankel. Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec. RFC 4868 (Proposed Standard), May 2007.

[190] S. Kent. Extended Sequence Number (ESN) Addendum to IPsec Domain of Interpretation (DOI) for Internet Security Association and Key Management Protocol (ISAKMP). RFC 4304 (Proposed Standard), December 2005.

[191] S. Kent. IP Authentication Header. RFC 4302 (Proposed Standard), December 2005.

[192] S. Kent. IP Encapsulating Security Payload (ESP). RFC 4303 (Proposed Standard), December 2005.

[193] S. Kent and K. Seo. Security Architecture for the Internet Protocol. RFC 4301 (Proposed Standard), December 2005. Updated by RFC 6040.

[194] Aleksandar Kircanski and Amr M. Youssef. On the sliding property of SNOW 3G and SNOW 2.0. IET Information Security, 5(4):199–206, 2011.

[195] T. Kivinen and M. Kojo. More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE). RFC 3526 (Proposed Standard), May 2003.

[196] John T. Kohl. The use of encryption in Kerberos for network authentication. In Gilles Brassard, editor, CRYPTO, volume 435 of Lecture Notes in Computer Science, pages 35–43. Springer, 1989.

[197] Geir M. Køien and Vladimir A. Oleshchuk. Location privacy for cellular systems; analysis and solution. In George Danezis and David Martin, editors, Privacy Enhancing Technologies, volume 3856 of Lecture Notes in Computer Science, pages 40–58. Springer, 2005.

[198] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-Hashing for Message Authentication. RFC 2104 (Best Current Practice), February 1997.

[199] Hugo Krawczyk, Kenneth G. Paterson, and Hoeteck Wee. On the security of the tls protocol: A systematic analysis. In Canetti and Garay [76], pages 429–448.

[200] Benjamin Kreuter, Abhi Shelat, and Chih-Hao Shen. Billion-gate secure computation with malicious adversaries. In Tadayoshi Kohno, editor, USENIX Security Symposium, pages 285–300. USENIX Association, 2012.

[201] D. Kuegler and Y. Sheffer. Password authenticated connection establishment with the internet key exchange protocol version 2 (ikev2). RFC 2104 (Best Current Practice), 2012.

[202] Eyal Kushilevitz, Steve Lu, and Rafail Ostrovsky. On the (in)security of hash-based oblivious ram and a new balancing scheme. In Rabani [267], pages 143–156.

[203] Sven Laur and Kaisa Nyberg. Efficient mutual data authentication using manually authenticated strings. In David Pointcheval, Yi Mu, and Kefei Chen, editors, CANS, volume 4301 of Lecture Notes in Computer Science, pages 90–107. Springer, 2006.

[204] Sven Laur and Sylvain Pasini. Sas-based group authentication and key agreement protocols. In Ronald Cramer, editor, Public Key Cryptography, volume 4939 of Lecture Notes in Computer Science, pages 197–213. Springer, 2008.

[205] Sven Laur and Sylvain Pasini. User-aided data authentication. IJSN, 4(1/2):69–86, 2009.

[206] Laurie Law, Alfred Menezes, Minghua Qu, Jerome A. Solinas, and Scott A. Vanstone. An efficient protocol for authenticated key agreement. Des. Codes Cryptography, 28(2):119–134, 2003.

[207] Dong Hoon Lee and Xiaoyun Wang, editors. Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings, volume 7073 of Lecture Notes in Computer Science. Springer, 2011.

[208] M. Lepinski and S. Kent. Additional Diffie-Hellman Groups for Use with IETF Standards. RFC 5114 (Informational), January 2008.

[209] Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In Gilbert [132], pages 62–91.

[210] Allison B. Lewko and Brent Waters. Decentralizing attribute-based encryption. In Paterson [253], pages 568–588.

[211] Allison B. Lewko and Brent Waters. Unbounded hibe and attribute-based encryption. In Paterson [253], pages 547–567.

[212] Yehuda Lindell. Fast cut-and-choose based protocols for malicious and covert adversaries. In Canetti and Garay [77], pages 1–17.

[213] Yehuda Lindell and Benny Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In Moni Naor, editor, EUROCRYPT, volume 4515 of Lecture Notes in Computer Science, pages 52–78. Springer, 2007.

[214] Yehuda Lindell and Benny Pinkas. Secure two-party computation via cut-and-choose oblivious transfer. J. Cryptology, 25(4):680–722, 2012.

[215] J. Linn. Generic Security Service Application Program Interface Version 2, Update 1. RFC 2743 (Proposed Standard), January 2000. Updated by RFC 5554.

[216] Chang Liu, Michael Hicks, and Elaine Shi. Memory trace oblivious program execution. In CSF, pages 51–65. IEEE, 2013.

[217] Adriana L´opez-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty com- putation on the cloud via multikey fully homomorphic encryption. In Howard J. Karloff and Toniann Pitassi, editors, STOC, pages 1219–1234. ACM, 2012.

[218] Gavin Lowe. Casper: A compiler for the analysis of security protocols. Journal of Computer Security, 6(1-2):53–84, 1998.

[219] Philip MacKenzie. On the security of the SPEKE password-authenticated key exchange protocol. IACR Cryptology ePrint Archive, 2001:19, 2001.

[220] C. Madson and R. Glenn. The Use of HMAC-MD5-96 within ESP and AH. RFC 2403 (Proposed Standard), November 1998.

[221] C. Madson and R. Glenn. The Use of HMAC-SHA-1-96 within ESP and AH. RFC 2404 (Proposed Standard), November 1998.

[222] Atefeh Mashatan and Douglas R. Stinson. Practical unconditionally secure two-channel message authentication. Des. Codes Cryptography, 55(2-3):169–188, 2010.

[223] Atefeh Mashatan and Serge Vaudenay. A message recognition protocol based on standard assumptions. In Zhou and Yung [318], pages 384–401.

[224] D. McGrew and D. Bailey. AES-CCM Cipher Suites for Transport Layer Security (TLS). RFC 6655 (Best Current Practice), July 2012.

[225] D. McGrew and J. Viega. The Use of Galois Message Authentication Code (GMAC) in IPsec ESP and AH. RFC 4543 (Proposed Standard), May 2006.

[226] S. P. Miller, B. C. Neuman, J. I. Schiller, and J. H. Saltzer. Kerberos authentication and authorization system. In In Project Athena Technical Plan, 1987.

[227] Chris J. Mitchell and Chan Yeob Yeun. Fixing a problem in the helsinki protocol. Operating Systems Review, 32(4):21–24, 1998.

[228] Vebjørn Moen, H˚avard Raddum, and Kjell Jørgen Hole. Weaknesses in the temporal key hash of WPA. Mobile Computing and Communications Review, 8(2):76–83, 2004.

[229] Payman Mohassel and Matthew K. Franklin. Efficiency tradeoffs for malicious two-party computation. In Yung et al. [316], pages 458–473.

[230] Masakatu Morii and Yosuke Todo. Cryptanalysis for RC4 and breaking WEP/WPA-TKIP. IEICE Transactions, 94-D(11):2087–2094, 2011.

[231] Paul Morrissey, Nigel P. Smart, and Bogdan Warinschi. The TLS handshake protocol: A modular analysis. J. Cryptology, 23(2):187–223, 2010.

[232] Steven J. Murdoch, Saar Drimer, Ross J. Anderson, and Mike Bond. Chip and pin is broken. In IEEE Symposium on Security and Privacy, pages 433–446. IEEE Computer Society, 2010.

[233] Mihir Nanavati, Patrick Colp, Bill Aiello, and Andrew Warfield. Cloud security: a gathering storm. Commun. ACM, 57(5):70–79, 2014.

[234] Moni Naor, Gil Segev, and Adam Smith. Tight bounds for unconditional authentication protocols in the manual channel and shared key models. In Dwork [109], pages 214–231.

[235] Moni Naor, Gil Segev, and Adam Smith. Tight bounds for unconditional authentication protocols in the manual channel and shared key models. IEEE Transactions on Information Theory, 54(6):2408–2425, 2008.

[236] Roger M. Needham and Michael D. Schroeder. Using encryption for authentication in large networks of computers. Commun. ACM, 21(12):993–999, 1978.

[237] C. Neuman, T. Yu, S. Hartman, and K. Raeburn. The Kerberos Network Authentication Service (V5). RFC 4120 (Proposed Standard), July 2005. Updated by RFCs 4537, 5021, 5896, 6111, 6112, 6113, 6649, 6806.

[238] Jesper Buus Nielsen, Peter Sebastian Nordholt, Claudio Orlandi, and Sai Sheshank Burra. A new approach to practical active-secure two-party computation. In Safavi-Naini and Canetti [277], pages 681–700.

[239] Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors. Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007. ACM, 2007.

[240] NIST Special Publication 800-56A. Recommendation for pair-wise key establishment schemes using discrete logarithm cryptography. National Institute of Standards and Technology, 2007.

[241] NIST Special Publication 800-56B. Recommendation for pair-wise key establishment schemes using integer factorization cryptography. National Institute of Standards and Technology, 2009.

[242] Open SSH Project. OpenSSH project. http://www.openssh.org/.

[243] Emmanuela Orsini, Joop van de Pol, and Nigel P. Smart. Bootstrapping BGV ciphertexts with a wider choice of p and q. IACR Cryptology ePrint Archive, 2014:408, 2014.

[244] Harriet Ortiz, editor. Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA. ACM, 1990.

[245] Rafail Ostrovsky. Efficient computation on oblivious rams. In Ortiz [244], pages 514–523.

[246] Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non- monotonic access structures. In Ning et al. [239], pages 195–203.

[247] Rafail Ostrovsky and Victor Shoup. Private information storage (extended abstract). In Frank Thomson Leighton and Peter W. Shor, editors, STOC, pages 294–303. ACM, 1997.

[248] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In IEEE Symposium on Security and Privacy, pages 238–252. IEEE Computer Society, 2013.

[249] Bryan Parno, Mariana Raykova, and Vinod Vaikuntanathan. How to delegate and verify in public: Verifiable computation from attribute-based encryption. In Ronald Cramer, editor, TCC, volume 7194 of Lecture Notes in Computer Science, pages 422–439. Springer, 2012.

[250] Sylvain Pasini and Serge Vaudenay. An optimal non-interactive message authentication protocol. In David Pointcheval, editor, CT-RSA, volume 3860 of Lecture Notes in Computer Science, pages 280–294. Springer, 2006.

[251] Sylvain Pasini and Serge Vaudenay. Sas-based authenticated key agreement. In Yung et al. [316], pages 395–409.

[252] Kenneth G. Paterson. A cryptographic tour of the IPsec standards. Cryptology ePrint Archive, Report 2006/097, 2006. http://eprint.iacr.org/.

[253] Kenneth G. Paterson, editor. Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings, volume 6632 of Lecture Notes in Computer Science. Springer, 2011.

[254] Kenneth G. Paterson, Bertram Poettering, and Jacob C. N. Schuldt. Plaintext recovery attacks against wpa/tkip. IACR Cryptology ePrint Archive, 2013:748, 2013.

[255] Kenneth G. Paterson, Thomas Ristenpart, and Thomas Shrimpton. Tag size does matter: Attacks and proofs for the tls record protocol. In Lee and Wang [207], pages 372–389.

[256] Kenneth G. Paterson and Gaven J. Watson. Plaintext-dependent decryption: A formal security treatment of SSH-CTR. In Gilbert [132], pages 345–361.

[257] Kenneth G. Paterson and Arnold K. L. Yau. Cryptography in theory and practice: The case of encryption in IPsec. In Serge Vaudenay, editor, EUROCRYPT, volume 4004 of Lecture Notes in Computer Science, pages 12–29. Springer, 2006.

[258] Lawrence C. Paulson. A fixedpoint approach to (co)inductive and (co)datatype definitions. In Gordon D. Plotkin, Colin Stirling, and Mads Tofte, editors, Proof, Language, and Interaction, pages 187–212. The MIT Press, 2000.

[259] R. Pereira and R. Adams. The ESP CBC-Mode Cipher Algorithms. RFC 2451 (Proposed Standard), November 1998.

[260] Birgit Pfitzmann, editor. Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding, volume 2045 of Lecture Notes in Computer Science. Springer, 2001.

[261] Benny Pinkas and Tzachy Reinman. Oblivious ram revisited. In Rabin [268], pages 502–519.

[262] Benny Pinkas, Thomas Schneider, Nigel P. Smart, and Stephen C. Williams. Secure two-party computation is practical. In Mitsuru Matsui, editor, ASIACRYPT, volume 5912 of Lecture Notes in Computer Science, pages 250–267. Springer, 2009.

[263] Matthew Pirretti, Patrick Traynor, Patrick McDaniel, and Brent Waters. Secure attribute- based systems. In Juels et al. [180], pages 99–112.

[264] Raluca A. Popa, Catherine M. S. Redfield, Nickolai Zeldovich, and Hari Balakrishnan. Cryptdb: processing queries on an encrypted database. Commun. ACM, 55(9):103–111, 2012.

[265] Guillaume Poupard and Jacques Stern. Security analysis of a practical ”on the fly” authentication and signature generation. In Kaisa Nyberg, editor, EUROCRYPT, volume 1403 of Lecture Notes in Computer Science, pages 422–436. Springer, 1998.

[266] Bart Preneel, editor. Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding, volume 1807 of Lecture Notes in Computer Science. Springer, 2000.

[267] Yuval Rabani, editor. Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012. SIAM, 2012.

[268] Tal Rabin, editor. Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings, volume 6223 of Lecture Notes in Computer Science. Springer, 2010.

[269] K. Raeburn. Advanced Encryption Standard (AES) Encryption for Kerberos 5. RFC 3962 (Proposed Standard), February 2005.

[270] K. Raeburn. Encryption and Checksum Specifications for Kerberos 5. RFC 3961 (Proposed Standard), February 2005.

[271] Mohammad Reza Reyhanitabar, Shuhong Wang, and Reihaneh Safavi-Naini. Non-interactive manual channel message authentication based on etcr hash functions. In Josef Pieprzyk, Hossein Ghodosi, and Ed Dawson, editors, ACISP, volume 4586 of Lecture Notes in Computer Science, pages 385–399. Springer, 2007.

[272] Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In Ehab Al- Shaer, Somesh Jha, and Angelos D. Keromytis, editors, ACM Conference on Computer and Communications Security, pages 199–212. ACM, 2009.

[273] P. Rogaway. Problems with proposed IP cryptography. Available at http://www.cs. ucdavis.edu/~rogaway/papers/draft-rogaway-ipsec-comments-00.txt, 61995.

[274] Phillip Rogaway, editor. Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings, volume 6841 of Lecture Notes in Computer Science. Springer, 2011.

[275] Michael Rusinowitch and Mathieu Turuani. Protocol insecurity with finite number of sessions is NP-complete. In Proc. of the 14th Computer Security Foundations Workshop (CSFW'01), pages 174–190, Cape Breton, Nova Scotia, Canada, 2001. IEEE Computer Society Press.

[276] Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors. 2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013. ACM, 2013.

[277] Reihaneh Safavi-Naini and Ran Canetti, editors. Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings, volume 7417 of Lecture Notes in Computer Science. Springer, 2012.

[278] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In Cramer [90], pages 457– 473.

[279] J. Salowey, A. Choudhury, and D. McGrew. AES Galois Counter Mode (GCM) Cipher Suites for TLS. RFC 5288 (Proposed Standard), August 2008.

[280] J. Schiller. Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2). RFC 4307 (Proposed Standard), December 2005.

[281] Pouyan Sepehrdad, Serge Vaudenay, and Martin Vuagnoux. Statistical attack on rc4 - distinguishing wpa. In Paterson [253], pages 343–363.

[282] Hovav Shacham and Brent Waters. Compact proofs of retrievability. J. Cryptology, 26(3):442– 483, 2013.

[283] Abhi Shelat and Chih-Hao Shen. Fast two-party secure computation with minimal assumptions. In Sadeghi et al. [276], pages 523–534.

[284] Elaine Shi, T.-H. Hubert Chan, Emil Stefanov, and Mingfei Li. Oblivious ram with o((logn)3) worst-case cost. In Lee and Wang [207], pages 197–214.

[285] Elaine Shi, Emil Stefanov, and Charalampos Papamanthou. Practical dynamic proofs of retrievability. In Sadeghi et al. [276], pages 325–336.

[286] Victor Shoup. A proposal for an ISO standard for public key encryption. Cryptology ePrint Archive, Report 2001/112, 2001. http://eprint.iacr.org/.

[287] Janos Simon, editor. Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA. ACM, 1988.

[288] Nigel P. Smart. Access control using pairing based cryptography. In Marc Joye, editor, CT-RSA, volume 2612 of Lecture Notes in Computer Science, pages 111–121. Springer, 2003.

[289] Nigel P. Smart. Errors matter: Breaking rsa-based pin encryption with thirty ciphertext validity queries. In Josef Pieprzyk, editor, CT-RSA, volume 5985 of Lecture Notes in Computer Science, pages 15–25. Springer, 2010.

[290] Nigel P. Smart and Frederik Vercauteren. Fully homomorphic simd operations. Des. Codes Cryptography, 71(1):57–81, 2014.

[291] Dawn Xiaodong Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. In IEEE Symposium on Security and Privacy, pages 44–55. IEEE Computer Society, 2000.

[292] Frank Stajano and Ross J. Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In Bruce Christianson, Bruno Crispo, James A. Malcolm, and Michael Roe, editors, Security Protocols Workshop, volume 1796 of Lecture Notes in Computer Science, pages 172–194. Springer, 1999.

[293] D. Stebila and J. Green. Elliptic Curve Algorithm Integration in the Secure Shell Transport Layer. RFC 5656 (Proposed Standard), December 2009.

[294] Emil Stefanov, Elaine Shi, and Dawn Xiaodong Song. Towards practical oblivious ram. In NDSS. The Internet Society, 2012.

[295] Emil Stefanov, Marten van Dijk, Elaine Shi, Christopher W. Fletcher, Ling Ren, Xiangyao Yu, and Srinivas Devadas. Path oram: an extremely simple oblivious ram protocol. In Sadeghi et al. [276], pages 299–310.

[296] Erik Tews and Martin Beck. Practical attacks against wep and wpa. In David A. Basin, Srdjan Capkun, and Wenke Lee, editors, WISEC, pages 79–86. ACM, 2009.

[297] Erik Tews, Ralf-Philipp Weinmann, and Andrei Pyshkin. Breaking 104 bit WEP in less than 60 seconds. In Sehun Kim, Moti Yung, and Hyung-Woo Lee, editors, WISA, volume 4867 of Lecture Notes in Computer Science, pages 188–202. Springer, 2007.

[298] F. Javier Thayer, Jonathan C. Herzog, and Joshua D. Guttman. Strand spaces: Why is a security protocol correct? In IEEE Symposium on Security and Privacy, pages 160–171. IEEE Computer Society, 1998.

[299] Yosuke Todo, Yuki Ozawa, Toshihiro Ohigashi, and Masakatu Morii. Falsification attacks against WPA-TKIP in a realistic environment. IEICE Transactions, 95-D(2):588–595, 2012.

[300] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In Gilbert [132], pages 24–43.

[301] Peter van Liesdonk, Saeed Sedghi, Jeroen Doumen, Pieter H. Hartel, and Willem Jonker. Computationally efficient searchable symmetric encryption. In Willem Jonker and Milan Petkovic, editors, Secure Data Management, volume 6358 of Lecture Notes in Computer Science, pages 87–100. Springer, 2010.

[302] Serge Vaudenay. Security flaws induced by CBC padding - Applications to SSL, IPSEC, WTLS ... In Lars R. Knudsen, editor, EUROCRYPT, volume 2332 of Lecture Notes in Computer Science, pages 534–546. Springer, 2002.

[303] Serge Vaudenay. Secure communications over insecure channels based on short authenticated strings. In Victor Shoup, editor, CRYPTO, volume 3621 of Lecture Notes in Computer Science, pages 309–326. Springer, 2005.

[304] Serge Vaudenay and Amr M. Youssef, editors. Selected Areas in Cryptography, 8th Annual International Workshop, SAC 2001 Toronto, Ontario, Canada, August 16-17, 2001, Revised Papers, volume 2259 of Lecture Notes in Computer Science. Springer, 2001.

[305] Gaven J. Watson, Reihaneh Safavi-Naini, Mohsen Alimomeni, Michael E. Locasto, and Shiv- aramakrishnan Narayan. Lost: location based storage. In Ting Yu, Srdjan Capkun, and Seny Kamara, editors, CCSW, pages 59–70. ACM, 2012.

[306] Peter Williams and Radu Sion. Single round access privacy on outsourced storage. In Yu et al. [315], pages 293–304.

[307] Peter Williams, Radu Sion, and Bogdan Carbunar. Building castles out of mud: practical access pattern privacy and correctness on untrusted storage. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, ACM Conference on Computer and Communications Security, pages 139–148. ACM, 2008.

[308] Peter Williams, Radu Sion, and Alin Tomescu. Privatefs: a parallel oblivious file system. In Yu et al. [315], pages 977–988.

[309] Stephen C. Williams. Analysis of the SSH key exchange protocol. In Liqun Chen, editor, IMA Int. Conf., volume 7089 of Lecture Notes in Computer Science, pages 356–374. Springer, 2011.

[310] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In FOCS, pages 160–164. IEEE Computer Society, 1982.

[311] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In FOCS, pages 162–167. IEEE Computer Society, 1986.

[312] T. Ylonen and C. Lonvick. The Secure Shell (SSH) Authentication Protocol. RFC 4252 (Proposed Standard), January 2006.

[313] T. Ylonen and C. Lonvick. The Secure Shell (SSH) Protocol Architecture. RFC 4251 (Proposed Standard), January 2006.

[314] T. Ylonen and C. Lonvick. The Secure Shell (SSH) Transport Layer Protocol. RFC 4253 (Proposed Standard), January 2006. Updated by RFC 6668.

[315] Ting Yu, George Danezis, and Virgil D. Gligor, editors. the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012. ACM, 2012.

[316] Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors. Public Key Cryptography - PKC 2006, 9th International Conference on Theory and Practice of Public-Key Cryptography, New York, NY, USA, April 24-26, 2006, Proceedings, volume 3958 of Lecture Notes in Computer Science. Springer, 2006.

[317] Yinqian Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart. Cross-vm side channels and their use to extract private keys. In Yu et al. [315], pages 305–316.

[318] Jianying Zhou and Moti Yung, editors. Applied Cryptography and Network Security, 8th International Conference, ACNS 2010, Beijing, China, June 22-25, 2010. Proceedings, volume 6123 of Lecture Notes in Computer Science, 2010.

[319] L. Zhu and B. Tung. Public Key Cryptography for Initial Authentication in Kerberos (PKINIT). RFC 4556 (Proposed Standard), June 2006. Updated by RFC 6112.

Catalogue number TP-06-14-085-EN-N ISBN 978-92-9204-103-8 DOI 10.2824/3739

ENISA
European Union Agency for Network and Information Security
Science and Technology Park of Crete (ITE)
Vassilika Vouton, 700 13, Heraklion, Greece

Athens Office
1 Vassilis Sofias, Marousi 151 24, Athens, Greece

Catalogue number TP-06-14-085-EN-N
ISBN 978-92-9204-103-8
DOI 10.2824/3739

PO Box 1309, 710 01 Heraklion, Greece
Tel: +30 28 14 40 9710
info@enisa.europa.eu
www.enisa.europa.eu