TLS, PACE, and EAC:
A Cryptographic View at Modern Key Exchange Protocols

Christina Brzuska Özgür Dagdelen Marc Fischlin
Technische Universität Darmstadt
`www.cryptoplexity.de`

Abstract. To establish a secure channel between two parties common security solutions often use a key exchange protocol as a preliminary subroutine to generate a shared key. These solutions include the protocols for secure communication between a reader and an identity card or passport, called PACE and EAC, and the TLS protocol for secure web communication. In this work we survey the cryptographic status of these protocols and the recent developments in this area.

1 Introduction

A secure channel between two parties is an important cryptographic primitive. It allows to communicate securely over a public channel by “wrapping” the communication into a cryptographically protected layer. The secure channel at foremost provides two security properties: *confidentiality* and *authenticity*. The former means that no outsider can learn the payload. In contrast, authenticity ensures integrity of the data, i.e., no adversary can inject or modify data, or even change the order of transmissions.

Most secure channel protocols assume that the two parties already share a common secret. Thus, before exchanging the actual data via a secure channel, the two parties first need to establish such a shared secret. For this, they usually first run a so-called key exchange protocol (occasionally also called key agreement protocol) over the public channel to agree on a symmetric key. The security property of the key exchange protocol assures that no eavesdropper on the communication can learn the key. Subsequently, the two parties use the derived secret to implement the secure channel, usually through (symmetric) encryption for confidentiality, and message authentication for authenticity. This paradigm is widely deployed by a number of practical protocols, amongst which are the prominent TLS protocol and the PACE/EAC protocols on the new German identity card.

In TLS, the key exchange is called *handshake*, and the channel protocol is named *record layer*. For the identity card, the Password-Authenticated Connection Establishment (PACE) protocol establishes a shared key between the card and the channel protocol is called *secure messaging*. The latter is also used as the channel protocol between the card and a terminal (e.g., a service provider), after the Extended Access Control (EAC) protocol has been executed to generate the key between these two parties.

In this work we survey the recent cryptographic developments for building secure channels, especially for TLS and the German identity card, where we focus on the key exchange step. To this end we first review the traditional security notions for key exchange and discuss their shortcomings compared to recent approaches. We then discuss the identity card protocols and TLS in more detail, and also give a brief overview over the state-of-the-art concerning their respective channel protocols.

2 Building Secure Channels

We look at secure channel protocols that consist of the key exchange step in which the parties agree upon a shared secret key, and the channel implementation that protects the payload (as in TLS/SSL, PACE, and EAC). Instead of a monolithic cryptographic analysis, it is often convenient to use a modular approach and to investigate the security of each step individually. This, however, requires to recompose the individual protocols and security proofs to be able to argue about the security of the whole protocol. Towards such a modular analysis of composed two-stage channel protocols, there are two main approaches: game-based security models and simulation-based security models.

2.1 Game-Based Security Models

In game-based security models one defines security of the task in question as a game played between a challenger and an adversary. The game specifies the moves the adversary is allowed (and the way the challenger answers them), i.e., it defines the attack model. Consider, for example, the well established model by Bellare and Rogaway [BR94] for authenticated key exchange protocols. There, the adversary may for instance observe several protocol executions between honest parties, may modify messages sent between the parties or inject new messages, or corrupt parties and learn their long-term secret keys.

In addition, a game specifies when the adversary "wins", i.e., when the adversary breaks security. For example, to break a secure channel, the adversary wins if it either breaks authenticity or confidentiality. For authenticity, the adversary's goal is to modify the transmitted data between two honest parties without being detected. To break confidentiality, the adversary has to learn some non-trivial information about the transmitted data. According to such a game-based definition, a scheme is called secure, if all (efficient) adversaries have only negligible winning advantage.

Game-based Security of Key Agreement. We now provide an (informal) description of the well-established and game-based Bellare-Rogaway security model [BR94] for authenticated key agreement. As mentioned above, we therefore need to define the attack model and the adversary's winning condition.

In the attack model, the adversary is mainly a very powerful Dolev-Yao adversary [DY81], that is, the adversary fully controls the network and decides when and what messages are received by a running session of a user. This, in particular, enables the adversary to see the messages sent by honest parties, to modify them, change the order of delivery, and to inject new messages. Moreover, after two honest sessions agreed on a key, the adversary may “reveal” it, which models potential leakage of keys. The adversary can also corrupt users and mount insider attacks, i.e., run arbitrarily many (concurrent) sessions of the key exchange protocol with honest users. The number of all executed sessions as well as their scheduling is also chosen by the adversary. In other words, the adversary’s attack capabilities are very broad, such that any security proof against such adversaries provides strong guarantees.

We now turn to the adversary’s winning condition. At some point in the game, the adversary may pick an (unrevealed) terminated session between two honest users. She then needs to prove that she knows at least one bit of information about their session key. This is modeled as an indistinguishability game: the adversary is given either the session key or a random string of the same length, and the game challenger asks the adversary to distinguish between the two cases, i.e., to correctly predict the case with probability significantly greater than the pure guessing probability $\frac{1}{2}$. Note that a random guess trivially achieves the success probability $\frac{1}{2}$; we are thus interested in the adversary’s advantage beyond this.

Game-Based Security for Secure Channels. Krawczyk [Kra01] and Bellare and Namprempre [BN00] consider different security levels of the underlying building blocks, messages authentication codes (MACs) and symmetric encryption and ask when their composition yields a secure channel. As there are various ways to compose these two building blocks, the theorems do not apply directly to specific real-world protocols such as TLS, where the channel implementation is a stateful algorithm that uses padding, sequence numbers and a range of different operator modes for the cipher; moreover, error messages are usually not covered by these models. These supposedly minor aspects, however, can impact security severely, as shown in Vaudenay [Vau02] and Paterson and Watson [PW08]. Therefore, depending on the protocol under consideration, the security models sometimes need to be tailor-made. We refer to Paterson et al. [PRS11] for a recent state-of-the-art analysis of the TLS record layer.

2.2 Simulation-Based Security Models

In contrast to game-based security models, simulation-based notions do not specify the adversary’s goal explicitly. Instead, they first define an ideal version of the primitive in question such that the ideal version reflects the desired functionality. For example, the ideal version of an authenticated channel would faithfully deliver messages, which are input from the honest sender, to the intended receiver; the adversary could see the message, but not modify it or inject new messages. Authenticity is thus guaranteed by definition. An implementation through a real protocol is then called secure if it is essentially “as good as” the ideal functionality, even in the presence of an adversary.

More formally, a protocol is secure according to such a simulation-based model if any attack against the real-world protocol can also been mounted against the ideal primitive. Consequently, as the ideal primitive trivially prevents any kind of attack, there cannot be an attack against the real-world protocol either, as it is indistinguishable from the ideal functionality. Put differently, the (only trivial) attacks on the ideal functionality give an upper bound for potential attacks on the real protocol. This is usually formalized by saying that, for any adversary $\mathcal{A}$ against the real protocol, there exists a so-called ideal-model adversary (aka. the simulator) $\mathcal{S}$ which can generate the same output as $\mathcal{A}$ when attacking the real protocol.

Simulation-Based Models for Key Agreement. The ideal functionality for a key agreement protocol is a trusted third party that gives a random string to both honest parties. The local output of the parties only consists in this random string. If a protocol is indistinguishable from this ideal functionality, then the outputs of the protocol look like random strings. For a formal description, see Canetti and Krawczyk [CK01] or the full version of Canetti [Can01].[1]

Simulation-Based Models for Secure Channels. The ideal version of a secure channel protocol is a functionality (or trusted third party) to which the sender can securely pass a message, that the functionality securely delivers to the intended receiver, i.e., the adversary neither gets to see the message nor to modify it. For a detailed formulation, we refer the reader to the full version of Canetti [Can01].

2.3 Comparison

There are advantages and disadvantages to each of the two approaches. Simulation-based security usually provides stronger security guarantees than game-based security: if one can break security in the game, then one can (in many cases) also make the protocol deviate from its ideal specification. Thus, whenever there is a successful adversary in the game-based setting, then there is a successful adversary against the simulation-based security. Taking the logical contra-position this means simulation-based security usually implies game-based security.

In addition, simulation-based security turns out to be useful for secure composition. Protocols shown to be secure in Canetti's (simulation-based) universal composition (UC) framework [Can01] compose well with other protocols in this framework. In particular, the compound execution of a UC-secure key agreement protocol with a UC-secure channel is also secure.

Unfortunately, simulation-based security is sometimes not achievable by real-life protocols, or sometimes even by any protocol [CF01]. As far as secure channels and key agreement are concerned, they cannot be secure against fully adaptive corruptions in the

[1]While [CK01] give a game-based model, they prove it to be equivalent to a simulation-based one.

UC model where the adversary decides during executions if and when to corrupt parties [Nie02][2]. Moreover, security in the UC framework necessitates the use of global session identifiers, and real-life protocols such as TLS or PACE/EAC usually do not meet these syntactical stipulations. We note that a recent work by Kuesters and Tuengerthal [KT11] somewhat alleviates the latter point, but their model still does not overcome the other problems with adaptive corruptions and strong simulation requirements.

In contrast, game-based security only asks for a specific security property (such as key indistinguishability) instead of general "ideal world" indistinguishability. As such, game-based security appears to be sufficient for the task at hand and is also often easier to achieve. The downside of achievable game-based security is that games are either not known, or provably lack, composition properties. This implies that when analyzing key exchange protocols and channel protocols separately in games, then their composition is often not known to be secure.

In conclusion, it is desirable to combine the advantages of both approaches: feasibility of game-based security and composability of simulation-based security. In the following section, we cover the recent progress on composability of game-based security in the area of key agreement.

2.4 Other Approaches

Simulation-based models and game-based models both consider arbitrary computationally bounded adversaries, usually modeled through Turing machines. In contrast, in the setting of symbolic methods, adversaries can derive symbolic expressions via a restricted set of so-called derivation rules. In a key exchange protocol, they can inject any message, which can be derived via these rules. A session key is called secure if the adversary is unable to derive it via the rules. As this is a weaker requirement than in the computational setting, one often enhances —if possible— a symbolic security analysis by a so-called soundness result to obtain the same security guarantees as in the computational setting.

The weaker security guarantees come at the advantage of a simpler (or even automated) analysis such that, in particular, a granular analysis of protocols is feasible, as done for TLS by Fournet et al. [FKS11] and for Kerberos by Backes et al. [BCJ$^+$11]. Note, however, that composition of symbolic analysis is as challenging as in the computational setting, as the derivation rules for the attack model of one protocol might affect the security of another protocol which was shown to be secure against a different set of derivation rules. For advances in this regard we refer the reader to the recent work by Cortier et al. [CW11] who prove first composition results for certain classes of derivation systems.

[2]Canetti and Krawczyk [CK01] claim to achieve full security—towards this purpose, they assume that the parties securely erase their state, which makes state corruption trivial and is, essentially, equivalent to not allowing state corruption.

3 Compositional Game-Based Models

The idea of looking into game-based security notions which also provide composition guarantees appears to be a promising venue to overcome the disadvantages of game-based models while keeping their benefits.

3.1 Composition of Bellare-Rogaway Protocols

For key agreement protocols, a recent result by Brzuska et al. [BFWW11] proves that BR-secure key agreement protocols are generally composable with arbitrary game-based protocols, if the key agreement protocol satisfies a notion called "session matching property". This property says that a key exchange protocol allows a third party to always identify from the public communication which sessions of which users agreed on the same key. While BR-secure protocols were always somewhat believed to be composable, the result in [BFWW11] is the first to show this property formally and to identify a sufficient condition for this.

Theorem 3.1 (from [BFWW11], informally) *Let* ke *be a key agreement protocol and let* π *be an arbitrary symmetric-key protocol that is secure according to some game-based model, then the following holds: if the key agreement protocol is correct, BR-secure, and satisfies public session matching, then the composed protocol* (ke, π) *is secure.*

We note that the public session matching holds for all common protocols but one can easily devise examples for which this is not true. Moreover, Brzuska et al. [BFWW11] show that the public session matching algorithm is not merely an artefact of their theorem but instead a necessary condition for key exchange protocols to be composable.

Theorem 3.2 (from [BFWW11], informally) *If a key agreement protocol* ke *is composable with arbitrary symmetric-key protocols, then* ke *satisfies the public session matching requirement.*

One can now conclude that any BR-secure protocol, composed with a secure channel protocol, automatically provides a secure channel. This holds if each of the components has been proven secure in its respective game-based model, and if the mild assumption about public session matching holds.

3.2 Compositional Results for TLS/SSL, PACE, and EAC

To protocols such as TLS and PACE/EAC, the composition result for BR-protocols, however, does not immediately apply. The reason is that in the key exchange stage, all these protocols use a so-called *key confirmation* step. In this step both participants authenticate the protocol transcript already with the established session key, with the intuition that it provides an a-posteriori authentication of the otherwise unauthenticated transmissions. But this means that within the key agreement protocol, in the final protocol messages, the session key is already used. This, however, violates the security in the BR sense as the established key cannot be considered fresh anymore (and is easily distinguishable from random). Consequently, we cannot apply the general composition result for BR-secure protocols, even though the protocols allow for public session matching.

In the sequel, we present several approaches to deal with the problem. Firstly, as done in the PACE/EAC proof [BFK09, DF10a], one can add an additional *key refresh* step to the protocol and argue that if the protocol satisfies an additional property (namely a different message structure in the key confirmation message than in the actual payload transmission), then the original protocol is secure whenever the modified protocol is secure. This approach is valid whenever the key agreement protocol is used with secure channels, and the authenticated message in the key confirmation step is distinct from the messages sent over the secure channel.

For TLS, Jager et al. [JKSS11] deal with this problem by mounting a monolithic analysis, i.e., they forgo splitting the TLS protocol into separate phases, and instead analyze the whole protocol. In another approach, Küsters and Tuengerthal [KT11] cope with the key confirmation message via considering ideal functionalities that implement several primitives simultaneously. Their model is again simulation-based and thus provides high modularity but also suffers from the aforementioned disadvantages (see Section 2.3).

A recent solution, which even goes beyond the application of secure channels, is a general game-based composition framework by Brzuska et al. [BFS$^+$11]. They consider composition of key agreement with arbitrary game-based protocols, without requiring the key agreement to satisfy BR-security. Instead, they use a form of key usability, a notion first considered by Datta et al. [DDMW06] and formalized in the simulation-based setting in [KT11]. The idea is roughly to invest a bit more work for analyzing the key agreement and to show that it is good for some primitives like symmetric encryption and message authentication. By this, one can automatically conclude security of the composition of the key agreement protocols with any protocol that relies on these primitives, like secure channels built out of symmetric encryption and message authentication. Using their framework, they give a modular analysis of TLS and provide foundations for such proofs in a general when not only composition with secure channels, but also with other protocols is desirable.

4 TLS: Overview over the Cryptographic Status

The TLS protocol consists of a key agreement step, the *handshake* protocol and a secure channel, called the *record layer*. Both protocols have been scrutinized with different goals and varying strengths of the results. As explained in the previous section, the TLS Handshake protocol in its due form (unlike the modified versions considered by Morrissey et al. [MSW10] and by Gajek et al. [GMP$^+$08]) cannot be analyzed in security models that require key indistinguishability. For TLS, only this year several solutions to this problem have been considered and, noteworthy, appeared concurrently.

For one, Jager et al. [JKSS11] presented a monolithic analysis of the TLS protocol for *one* choice of the TLS Handshake (among the choices negotiated by the parties during the handshake). While monolithic approaches usually suffer from a lack of modularity (entailing poor robustness of the analysis in light of future protocol modifications), their analysis, interestingly, does not. Their key ingredient towards modularity is somewhat a special case of the composition result by Brzuska et al. [BFWW11]. In particular, they show that a truncated version of the TLS Handshake protocol is BR-secure and identify a sufficient condition for the TLS Record Layer to be securely composable with the (truncated) TLS Handshake. Hence, their analysis can be composed with any formal security proof for the TLS Record Layer, provided the achieved security level matches the one considered by Jager et al. [JKSS11]. In light of [BFWW11], we can even deduce that their result composes in general. Note that this approach is TLS-specific; usually, truncated versions of protocols, now without key confirmation, become insecure.

Simultaneously, two new frameworks were introduced, as explained in the previous section, one by Küsters and Tuengerthal [KT11] and one by Brzuska et al. [BFS$^+$11]. Both allow to analyze the TLS Handshake protocol[3]. As the model by Küsters and Tuengerthal is simulation-based, it suffers from the aforementioned restrictions on the adversary's corruption capacities, while, noteworthy, nicely circumventing nuisances of prior simulation-based approaches. The game-based composition framework by Brzuska et al. [BFS$^+$11] fully applies to TLS. They carry out a security proof that, combined with a security proof for the TLS Record Layer, yields a security analysis of (an abstracted version of) the entire TLS protocol.

Theorem 4.1 (from [BFS$^+$11], informally) *The TLS Handshake protocol can be securely composed with a secure channel that uses (specific stateful implementations of) MACs and encryption.*

The TLS Record Layer is similarly issue of intense research. In [Kra01], Krawczyk analyzes two abstract versions of the TLS record layer. One of them is a general scheme called MAC-then-ENC whereas the other is more specific. While he proves that MAC-then-ENC does not always yield a secure channel, Krawczyk proves it to be a secure channel when the encryption is instantiated via a secure cipher in CBC mode. The latter result was interpreted as a security proof for the TLS Record Layer and the protocol description complied, indeed, with the accepted standard for cryptographic abstraction of protocols.

However, Vaudenay [Vau02] later discovered that some of the dispatched details leave leverages for attacks. In particular, the process of message padding enables additional security breaches. In subsequent works, e.g., Paterson and Watson [PW08] as well as Maurer and Tackmann [MT10], the analysis progressively reflected the actual real-world implementation, culminating in the recent work by Paterson et al. [PRS11] who analyze the actual RFC [DA99, DA06]. On the way several attacks were discovered, and the RFC was modified to prevent those attacks.

We can therefore consider the cryptographic analysis of the TLS protocol as having achieved an acceptable level of rigorousness, showing that the protocol is secure.

[3]Küsters and Tuengerthal [KT11] actually do not carry out this analysis formally, but convincingly show its feasability in their model.

5 EAC & PACE: Overview over the Cryptographic Status

The Password Authenticated Connection Establishment (PACE), and the Extended Access Control (EAC) protocol, are deployed in the current electronic ID card in Germany. They each enable the card and the (local) reader resp. the card and the (remote) terminal to share a secret session key which is used subsequently in the secure messaging protocol.

5.1 Extended Access Control (EAC)

EAC is deployed in two different variants. Here, we focus on the latest version which is implemented in the German ID card. In EAC, a chip and terminal agree upon a secret session key if both parties mutually authenticate first. This is accomplished by two sub-protocols, called Terminal Authentication (TA) and Chip Authentication (CA). First, in the TA phase, the terminal signs in a challenge-response step the card's challenge under a certified key. Subsequently, in the CA step, the chip authenticates itself also via a challenge-response step, but using already the derived session key in a message authentication code instead of a signature, as this ensures deniability for the card. The session key itself, which is also used for this chip authentication, is derived from a DH key computed between the static (certified) key of the card and an ephemeral key of the terminal.

The security analysis of EAC, as a key exchange protocol, in [DF10a] is done in the Bellare-Rogaway model, with the hindsight that the key is already used in the key exchange step for a dedicated message which does not appear in the channel protocol. The formal analysis thus assumes a key refresh step for which the composition result for BR-key exchange protocols in [BFWW11] applies, noting that one can publicly match session. Alternatively, for the version using the session key already, the result [DF10b] about compositional properties of key exchange protocols with controlled usage of the key applies.

The security analysis of the channel protocol, called secure messaging, is done in [DF10b]. The secure messaging follows an encrypt-then-authenticate method with a counter value for each message, with all data like the counter value, the auxiliary information, and the message carefully aligned to block length. As such it provides a secure channel *for randomly chosen keys* (instead of keys generates through the key exchange protocol).

The entire secure channel, consisting of the EAC key exchange composed with the secure messaging, is analyzed in [DF10b] in a monolithic way, saying that security follows from the security of the key exchange protocols and the security of the primitives:

Theorem 5.1 (from [DF10b], informally) *The security of the composed protocol, consisting of the EAC key exchange protocol with secure messaging, follows from the security of the EAC key exchange protocol, the security of the encryption scheme, and the unforgeability of the message authentication code.*

5.2 Password-Authenticated Connection Establishment (PACE)

The PACE protocol is deployed between the chip and a reader. In contrast to EAC it does not rely on certified public keys, but is password based. That is, the chip and the reader share a low-entropy secret at the outset of the execution. In practice, the card holder types in this password at the reader, or in case of machine readable travel documents the password is built out of the data in the machine readable zone and obtained by the reader through a physical read-out (say, via swiping the passport at the reader).

PACE, as a key exchange protocol, has been analyzed in a version of the BR model, suitable for the case of password-based protocols. The model is due to Bellare-Pointcheval-Rogaway (BPR model) [BPR00] and takes into account that an adversary could potentially guess the low-entropy password in an execution with an honest party. But the model says that the adversary should not be able to do more than that. In particular, the adversary should only be able to test one password in any active execution (online testing) but must not be able to identify a password after eavesdropping a communication (offline testing).

Formally, the model is as the BR model, except that we measure the adversary's success probability against the trivial prediction strategy of guessing the right password with probability $1/N$ (if passwords are chosen uniformly from a set of size N, say, $N = 10^6$ for 6-digit PINs). This bound is for a single execution and grows to q/N when the adversary can probe q executions. To be precise, the adversary in q executions should then not be able to be significantly better in distinguishing the genuine key from a random string, than with probability q/N. Such a password-based protocol is considered to be optimal.

While the PACE protocol as a key exchange protocol has been shown in [BFK09] to be optimal, the channel protocol following the PACE step has, to best of our knowledge, not been analyzed formally (when composed with PACE). Luckily, since the channel is also implemented by secure messaging, and by the similarity of the BPR model to the BR model, the monolithic analysis in [DF10b] for EAC and Secure Messaging immediately carries over. Only the security bound for the overall protocol now also contains the weaker bound q/N (plus a negligible term). This, of course, is inevitable for password-based protocols:

Theorem 5.2 (adopted from [DF10b], informally) *The security of the composed protocol, consisting of the PACE password-based key exchange protocol with secure messaging, follows from the security of the PACE key exchange protocol, the security of the encryption scheme, and the unforgeability of the message authentication code.*

6 Conclusion

As discussed, research on key exchange protocols is crucial for a wide range of practical protocols such as TLS, EAC, and PACE. The latter, however, are often non-trivial to analyze. Fortunately, several new approaches have been developed such that TLS, EAC, and PACE are now shown to be cryptographically sound protocols.

References

[BCJ+11] Michael Backes, Iliano Cervesato, Aaron D. Jaggard, Andre Scedrov, and Joe-Kai Tsay. Cryptographically sound security proofs for basic and public-key Kerberos. *Int. J. Inf. Secur.*, 10:107–134, 2011.

[BFK09] Jens Bender, Marc Fischlin, and Dennis Kügler. Security Analysis of the PACE Key-Agreement Protocol. In *ISC 2009*, volume 5735 of *LNCS*, pages 33–48. Springer, 2009.

[BFS+11] Christina Brzuska, Marc Fischlin, Nigel P. Smart, Bogdan Warinschi, and Stephen C. Williams. Less is More: Relaxed yet Composable Security Notions for Key Exchange. *In submission*, 2011.

[BFWW11] Christina Brzuska, Marc Fischlin, Bogdan Warinschi, and Stephen C. Williams. Composability of bellare-rogaway key exchange protocols. In *CCS 2011*, pages 51–62. ACM, 2011.

[BN00] Mihir Bellare and Chanathip Namprempre. Authenticated Encryption: Relations among notions and analysis of the generic composition paradigm. In *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 531–545. Springer, 2000.

[BPR00] Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated Key Exchange Secure against Dictionary Attacks. In *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 139–155. Springer, 2000.

[BR94] Mihir Bellare and Phillip Rogaway. Entity Authentication and Key Distribution. In *CRYPTO'93*, volume 773 of *LNCS*, pages 232–249. Springer, 1994.

[Can01] Ran Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society, 2001.

[CF01] Ran Canetti and Marc Fischlin. Universally Composable Commitments. In *CRYPTO 2001*, volume 2139 of *LNCS*, pages 19–40. Springer, 2001.

[CK01] Ran Canetti and Hugo Krawczyk. Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 453–474. Springer, 2001.

[CW11] Véronique Cortier and Bogdan Warinschi. A composable computational soundness notion. In *CCS 2011*, pages 63–74. ACM, 2011.

[DA99] T. Dierks and C. Allen. The TLS Protocol Version 1.0. In *RFC 2246*, 1999.

[DA06] T. Dierks and C. Allen. The TLS Protocol Version 1.2. In *RFC 4346*, 2006.

[DDMW06] Anupam Datta, Ante Derek, John Mitchell, and Bogdan Warinschi. Computationally Sound Compositional Logic for Key Exchange Protocols. In *CSFW 2006*, pages 321–334. IEEE Computer Society, 2006.

[DF10a] Özgür Dagdelen and Marc Fischlin. Security Analysis of the Extended Access Control Protocol for Machine Readable Travel Documents. In *ISC 2010*, volume 6531 of *LNCS*, pages 54–68. Springer, 2010.

[DF10b] Özgür Dagdelen and Marc Fischlin. Sicherheitsanalyse des EAC-Protokolls. Technical report, Project 826, 2010. `http://www.personalausweisportal.de/SharedDocs/Downloads/DE/Studie_Kryptographie_Volltext.pdf?__blob=publicationFile`.

[DY81] D. Dolev and A. C. Yao. On the security of public key protocols. In *SFCS '81*, pages 350–357. IEEE Computer Society, 1981.

[FKS11] Cédric Fournet, Markulf Kohlweiss, and Pierre-Yves Strub. Modular code-based cryptographic verification. In *CCS 2011*, pages 341–350. ACM, 2011.

[GMP$^+$08] Sebastian Gajek, Mark Manulis, Olivier Pereira, Ahmad-Reza Sadeghi, and Jörg Schwenk. Universally Composable Security Analysis of TLS. In *ProvSec 2008*, volume 5324 of *LNCS*, pages 313–327. Springer, 2008.

[JKSS11] Tibor Jager, Florian Kohlar, Sven Schäge, and Jörg Schwenk. A Standard-Model Security Analysis of TLS-DHE. *IACR Cryptology ePrint Archive*, 2011:219, 2011.

[Kra01] Hugo Krawczyk. The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?). In *CRYPTO 2001*, volume 2139 of *LNCS*, pages 310–331. Springer, 2001.

[KT11] Ralf Küsters and Max Tuengerthal. Composition theorems without pre-established session identifiers. In *CCS 2011*, pages 41–50. ACM, 2011.

[MSW10] Paul Morrissey, Nigel P. Smart, and Bogdan Warinschi. The TLS Handshake Protocol: A Modular Analysis. *Journal of Cryptology*, 23(2):187–223, 2010.

[MT10] Ueli Maurer and Björn Tackmann. On the soundness of authenticate-then-encrypt: formalizing the malleability of symmetric encryption. In *CCS 2010*, pages 505–515. ACM, 2010.

[Nie02] Jesper Buus Nielsen. Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case. In *CRYPTO 2002*, volume 2442 of *LNCS*, pages 111–126. Springer, 2002.

[PRS11] Kenneth G. Paterson, Thomas Ristenpart, and Thomas Shrimpton. Tag Size Does Matter: Attacks and Proofs for the TLS Record Protocol. In *ASIACRYPT* 2011, to appear.

[PW08] Kenneth G. Paterson and Gaven J. Watson. Immunising CBC Mode Against Padding Oracle Attacks: A Formal Security Treatment. In *SCN 08*, volume 5229 of *LNCS*, pages 340–357. Springer, 2008.

[Vau02] Serge Vaudenay. Security Flaws Induced by CBC Padding - Applications to SSL, IPSEC, WTLS ... In *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 534–546. Springer, 2002.