Expanding ACCE: Robust Secure Channel Establishment

Britta Hale

NTNU, Norwegian University of Science and Technology
britta.hale@ntnu.no

Abstract

Analysis of the Transport Layer Security (TLS) protocol has motivated the development of the Authenticated and Confidential Channel Establishment (ACCE) model, due to the impossibility of modeling key indistinguishability in TLS. As proposed, the intended channel established by ACCE demands perfect message delivery. While these goals are realistic for TLS, they are not for DTLS, for which key indistinguishability is also impossible. In this work we discuss modeling issues for a variety of channel constraints and extend the ACCE model for channels with more robust channel goals, such as DTLS.

Keywords: Authenticated and secure channel establishment (ACCE), secure channels, AEAD, TLS, DTLS, QUIC

1 Introduction

Key exchange protocol analysis is historically premised on the indistinguishability of the derived key from a random value. However, some protocols, such as the Transport Layer Security[1] (TLS) protocol [DR08], cannot achieve this goal because of its key confirmation messages. Although such a failure would typically allow for the protocol to be dismissed as insecure, there is no known attack on TLS that exploits this "weakness". Indeed, TLS (in various versions) has

This paper was presented at the NIK-2017 conference; see http://www.nik.no/.

[1]When referring to TLS, we refer to TLS 1.2 [DR08] throughout, although previous versions also suffer from a lack of key indistinguishability. TLS 1.3 (in draft [Res17]) aims to correct this issue; however, it is unlikely that all prior protocol versions will be phased out of existence in the near or even distant future.

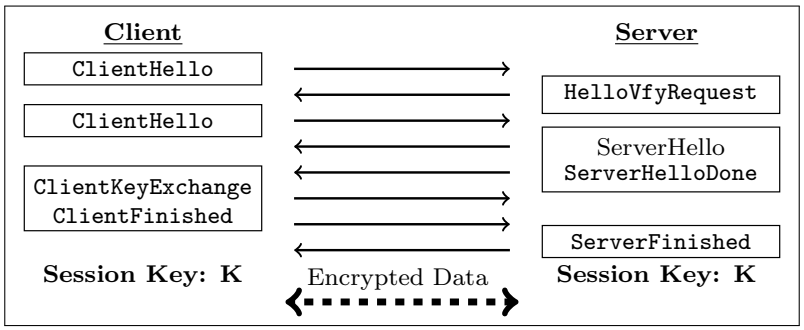

Figure 1: Datagram TLS (DTLS) handshake.

been in existence since 1999 and has emerged as the foremost protocol in internet security. This conundrum led to the development of the Authenticated and Confidential Channel Establishment (ACCE) model for better analysis of TLS by Jager et al. [JKSS12], which uses a monolithic security analysis of a composed key exchange and channel. ACCE does not demand key indistinguishability, instead requiring that the derived key, when used, results in a secure channel. Protocols which have been analyzed in the ACCE model include TLS and SSH [JKSS12, KPW13, BDK+14], and variants of the model have been applied in analyses of pre-shared key ciphersuites of TLS, QUIC, and EMV [LSY+14, LJBN15, BSWW13].

Saliently, ACCE requires perfect packet delivery in a secure channel, with replay detection, no packet dropping, and a strict delivery of packets in increasing order. These assumptions are not always realistic or desired. ACCE was motivated by TLS, for which such demands are appropriate; however, other real-world protocols exist which allow for packet dropping and re-ordering, and even packet replays. Previous research by Boyd *et al.*[BHMS16] provides a hierarchy of authenticated encryption with associated data (AEAD) guarantees – namely, basic authenticated encryption with associated data (AEAD); AEAD plus replay detection, and AEAD ensuring strictly increasing order of receipt and replay detection , and "perfect delivery". [BHMS16] also provides examples of corresponding real-world protocols, including Datagram TLS (DTLS) [RM12], DTLS with replay detection, and 802.11 [IEE12].

Like TLS, DTLS does not achieve key indistinguishability, as the session key is used to encrypt the handshake `Finished` messages (see Figure 1) and an adversary can therefore distinguish between a random and real key (from the key exchange experiment) by trial decryption. Thus analyses of DTLS variants (with/without replay protection) require ACCE-like models. In order to address this issue,

we present ACCE-q: a hierarchy of ACCE-like oracles corresponding to various channel security goals. Building the hierarchy, we start at basic AEAD goals (ACCE-1) and sequentially add replay protection (ACCE-2), strict increasing order demands (ACCE-3), and packet dropping restrictions (ACCE-4). In this sequence, the strictest (ACCE-4) corresponds to the original ACCE oracles. ACCE-1 and ACCE-2 would be appropriate for analysis of DTLS and DTLS with optional replay protection, respectively. Channel goals in ACCE-3 align with those of 802.11, making ACCE-3 ideal for application to any similar protocol which cannot achieve key indistinguishability.

Contributions We provide variations of the ACCE channel experiment oracles under the goals of basic AEAD, AEAD plus replay detection, and AEAD ensuring strict increasing order of receipt and replay detection. We compare our oracles to those of the original ACCE model, enabling the analysis of secure channel establishment. Our models are the essential first step in the formal analysis of DTLS and other real-world protocols which are required to be more robust to errors than TLS. We specifically address encryption and decryption oracle queries under varying environments; these oracles can be combined with various partnering definitions, or other pre-accept phase ACCE model variants.

Outline Section 2 describes modeling and analysis issues for real-world protocols, Section 3 introduces AEAD and the ACCE model, Section 4 introduces the ACCE-q hierarchy, and application of ACCE-q to real-world protocols is discussed in Section 5.

2 Real-World Protocols

Often the network environment plays a central role in what demands can be made on real-world protocols. For example, running over TCP, TLS can make strong demands on packet delivery while DTLS over UDP cannot. Such demand variation can be desirable: it could be appropriate for a financial transaction to fail if a message packet is dropped in a TLS connection while it would be annoying for a video call to fail every time a frame is missing in a DTLS connection. Here we consider a sampling of protocols with increasingly weaker channel demands.

TLS *(Fully reliable channel.)* TLS stands on the precipice of change: the current standard (TLS 1.2 [DR08]) uses key confirmation

messages which prevent key indistinguishability while the new standard draft (TLS 1.3 [Res16]) enables key indistinguishability. However, even after standardization of TLS 1.3, past versions are likely to persist in use for a long time to come, due to the lifespan of devices using those versions and backwards compatibility. As noted above, the ACCE model plays a central role in various analyses of TLS 1.2 [JKSS12, KPW13].

802.11 *(Packet drops allowed.)* As the basis of wireless communication, 802.11 is designed to accommodate a variety of demands and consequently does not require perfect packet delivery [IEE12]. However, it does cover replay protection and re-ordering. While 802.11 does not suffer from the lack of key indistinguishability that usually propels protocols into an ACCE model analysis, it is possible to use the ACCE to analyze it.

QUIC *(Packet drops and re-ordering allowed.)* Google's Quick Internet UDP Connections (QUIC) protocol was proposed in answer to the delay incurred by the round-trips (RTT) of using a full TLS handshake. Analyses of QUIC have used differing approaches. [FG14] employed an idealized version of the key derivation, thus avoiding the issue of key indistinguishability. Another work [LJBN15] employed an ACCE definition to address channel security establishment. Due to the multiple session keys in QUIC, the authors adapted the ACCE definition, defining QACCE (Quick ACCE). However, the formulation of encryption and decryption queries obscures the connection to the original ACCE experiment. Our definitions fit within the adapted framework of Lychev et al. while maintaining a clear link to ACCE as we adapt the channel oracle queries while leaving room for variation in adversarial capabilities and partnering definition variations.

DTLS *(Packet drops, re-ordering, and possible replays allowed.)* Despite the extensive attention on TLS, very little attention has been given to DTLS (currently at version DTLS 1.2) [RM12]. DTLS is a TLS variant, designed for use in environments that cannot achieve the strict requirements of TLS. Namely, DTLS running UDP must be able to adapt to packet dropping and re-ordering. As a result, while DTLS mirrors TLS, results on the security of the latter cannot simply be extended to it. Note that there are two variants of DTLS: with replay protection and without. Essentially, this results in two distinct protocols with different channel guarantees.

3 ACCE

ACCE runs in two phases: a *pre-accept* phase and a *post-accept* phase, differentiated by the status of key acceptance. Before key acceptance, the *pre-accept* phase handles session matching, where an adversary wins if it gets a session oracle to accept maliciously. After key acceptance, channel security under the session key is handled in the *post-accept* phase, where correctly answering an AEAD encryption challenge implies an adversarial win. Thus, a break in ACCE security can occur in either or both phases. In this section we present the ACCE post-accept phase Encrypt, and Decrypt oracles from [JKSS12] and later revisions [JKSS11], and include small revisions from other sources [KPW13, BSWW13]. For more details relating to the ACCE model; general definitions for oracles Send, Reveal, and Corrupt; and experiments, see [JKSS12, JKSS11]. We use the standard stateful AEAD primitive, omitted here due to space restrictions.

Let $\{P_1, \ldots, P_l\}$ be a set of identities, for $l \in \mathbb{N}$, where each P_i possesses a long-term key-pair (sk_i, pk_i) and is modeled by a collection of session oracles $\pi_i^1, \ldots, \pi_i^d$. Each π_i^s has access to (sk_i, pk_i), as well as $pk_1, \ldots, pk_l$. Let b_i^s, a bit selected at random as $b_i^s \xleftarrow{\$} \{0,1\}$ at the start of the game.

$\mathsf{Encrypt}(\pi_i^s, l, \mathtt{ad}, m_0, m_1)$:

1: $u \leftarrow u + 1$
2: $(c^{(0)}, st_{\mathrm{E}}^{(0)}) \xleftarrow{\$} \mathrm{st.Enc}(K, l, \mathtt{ad}, m_0, st_{\mathrm{E}})$
3: $(c^{(1)}, st_{\mathrm{E}}^{(1)}) \xleftarrow{\$} \mathrm{st.Enc}(K, l, \mathtt{ad}, m_1, st_{\mathrm{E}})$
4: **if** $c^{(0)} = \perp$ or $c^{(1)} = \perp$ **then**
5: **return** $\perp$
6: $(ad_u, c_u, st_{\mathrm{E}}) \leftarrow (\mathtt{ad}, c^{(b_s^P)}, st_{\mathrm{E}}^{(b_s^P)})$
7: **return** c_u

$\mathsf{Decrypt}(\pi_i^s, \mathtt{ad}, c)$:

1: **if** $b_s^P = 0$ **then**
2: **return** $\perp$
3: $v \leftarrow v + 1$
4: $(m, st_{\mathrm{D}}) \leftarrow \mathrm{st.Dec}(K, \mathtt{ad}, c, st_{\mathrm{D}})$
5: **if** $(u < v) \vee (c \neq c_v) \vee (\mathtt{ad} \neq \mathtt{ad}_v)$ **then**
6: $\mathtt{phase} \leftarrow 1$
7: **if** $\mathtt{phase} = 1$ **then**
8: **return** m
9: **return** $\perp$

Figure 2: ACCE security game oracles, for principal P and session π_s^P, with session key K.

In the ACCE experiment for a protocol Π, $\mathsf{Exp}_{\Pi}^{\mathsf{ACCE}}$, a challenger $\mathcal{C}$ takes as input a security parameter λ and plays a game with an adversary $\mathcal{A}$: $\mathcal{C}$ generates l pairs (sk_i, pk_i), for $i \in [l]$ and implements π_i^s for $i \in [l]$ and $s \in [d]$ and gives $pk_1, \ldots, pk_l$ to $\mathcal{A}$ as input. $\mathcal{A}$ may make Send, Reveal, Corrupt, Encrypt, and Decrypt queries, until it outputs a triple (i, s, b') and terminates. $\mathcal{A}$ wins if $b' = b_i^s$ (i.e.

$\mathsf{Exp}^{\mathsf{ACCE}}_{\Pi} = 1$). Thus the advantage of $\mathcal{A}$ in $\mathsf{Exp}^{\mathsf{ACCE}}_{\Pi}$ is defined as: $\mathbf{Adv}^{\mathsf{ACCE}}_{\Pi}(\mathcal{A}) = |2 \cdot \Pr[\mathsf{Exp}^{\mathsf{ACCE}}_{\Pi,\mathcal{A}}(\lambda) = 1] - 1|$.

Following [KPW13], we include a check on the received associated data in step 5 of Decrypt in Fig. 2. While this check was omitted in [JKSS12, JKSS11], it is necessary for allowing an adversarial win in the case of a collision. If an adversary submits a valid $\mathtt{ad}$ to the decryption oracle along with c, from which c was not created via a call to Encrypt, then the adversary has made a successful forgery.

Remark. *The condensed nature of the oracles presented in Figure 2, it could appear that successful trivial attacks could be mounted by $\mathcal{A}$. We comment on one such "false" attack here and how it is inhibited: Suppose that an adversary simply modifies* $\mathtt{ad}$*, without attempting to break the AEAD, and subsequently submits valid ciphertexts to the* Decrypt *oracle. In this scenario, $\mathcal{A}$ may hope that* $\mathtt{phase} \leftarrow 1$ *since* $\mathtt{ad} \neq \mathtt{ad}_v$*, and that subsequent decryption queries on valid ciphertexts will therefore result in m being returned. However, this scenario is covered by the security experiment:*
If $\mathcal{A}$ modifies $\mathtt{ad}$*, then the decryption* st.Dec *should fail, with the result that $m = \perp$ at step 5. This would result in* $\mathtt{ad} \neq sent.ad_v$*, leading to* $\mathtt{phase} \leftarrow 1$ *and m being returned. As $m = \perp$, the adversary gains no advantage. Subsequent, even correct decryption oracle queries depend on the* st.Dec *algorithm in question: these should result in $m = \perp$ as well, based on the AEAD goals, but could alternatively result in a valid adversarial win under correct decryption as* $\mathtt{phase} = 1$.

Due to space constraints, the following definition omits key acceptance, identity corruption, and key reveal conditions, as well as ACCE details linked to pre-acceptance. This matches our focus on channel oracles.

Definition 3.1 (ACCE – Secure Channel)**.** *A protocol Π is a (t, ϵ)-secure channel ACCE protocol if, for all adversaries $\mathcal{A}$ which run in time t, when $\mathcal{A}$ terminates and outputs (i, s, b') such that*

$$\mathbf{Adv}^{\mathit{ACCE}}_{\Pi}(\mathcal{A}) = |\Pr[b' = b^s_i] - 1/2| < \epsilon \ .$$

4 Extended ACCE

We now extend the ACCE Encrypt and Decrypt oracles for other secure channel variants with weaker channel demands. Following the hierarchy of AEAD demands of [BHMS16], we have the following ACCE-q hierarchy based on the post-accept channel experiment:

- $q = 1$: AEAD
- $q = 2$: AEAD with replay detection
- $q = 3$: AEAD with replay detection and ensured strictly increasing packet ordering
- $q = 4$: AEAD with replay detection, strictly increasing packet ordering, and no drops (original ACCE)

Each q-th level of ACCE-q corresponds to a test condition ($\mathsf{condition}_q$) in the oracle queries of Fig. 3.

Building on the ACCE oracles, we require each π_i^s to also maintain a set of sent ciphertexts $sent.C_i^s$ and received ciphertexts $rcvd.C_i^s$, where the u-th entry in $sent.C_i^s$ is written $sent.c_u$ and the v-th entry in $rcvd.C_i^s$ is written $rcvd.c_v$. This move mirrors the original ACCE definition [JKSS12] where a list of ciphertexts C_i^s was maintained by π_i^s, and the u-th entry was referred to as $C_i^s[u]$ (in comparison to c_u, see line 7 in Fig. 2). As maintaining such a list was unnecessary to the channel authentication goals of ACCE, later works used the simplified model presented in Fig. 2 (e.g. [KPW13]). We reintroduce the sent ciphertext list in a simplified form and add a received ciphertext list due to the complexity of modeling weaker channel demands, especially at $\mathsf{condition}_3$ (see Fig. 3). Likewise, our list element notation is aimed for enhancing comprehensibility of $\mathsf{condition}_q$.

Remark. Encrypt *and* Decrypt *oracles for ACCE-4 correspond exactly to the original ACCE experiment oracles. The notational changes in ACCE-4 from the original ACCE (i.e.* $sent.c_u$ *vs* c_u*,* $rcvd$ *notation, etc.), and the addition of line 4 in* Decrypt *(for variable assignment) are not necessary. They are included in ACCE-4 for comparison and uniformity to the oracles of ACCE-1,2,3. In the case of ACCE-4, lines 6 and 7 of* Decrypt *in Fig. 3 match line 5 of* Decrypt *in Fig. 2, under notational changes.*

Remark. *The addition of* $m \neq \perp$ *at line 8 is necessitated by standard protocol behavior at lower levels: 802.11, DTLS, and IPsec [Ken05] all require the silent dropping of a message under authentication failure. Generally, such protocols are required to be robust to authentication failures, as they are based on an unreliable transport (such as UDP for DTLS). In comparison, Level 4 protocols, such as TLS, do not need to allow this luxury.*
There are optional alternatives: while DTLS [RM12, pg. 12] mandates that implementations should *silently discard messages*

$\underline{\mathsf{Encrypt}(\pi_i^s, l, \mathtt{ad}, m_0, m_1)}$:

1: $u \leftarrow u + 1$
2: $(c^{(0)}, st_{\mathrm{E}}^{(0)})$
$\xleftarrow{\$} \mathrm{st.Enc}(K, l, \mathtt{ad}, m_0, st_{\mathrm{E}})$
3: $(c^{(1)}, st_{\mathrm{E}}^{(1)})$
$\xleftarrow{\$} \mathrm{st.Enc}(K, l, \mathtt{ad}, m_1, st_{\mathrm{E}})$
4: **if** $c^{(0)} = \bot$ or $c^{(1)} = \bot$ **then**
5: **return** $\bot$
6: $(sent.ad_u, sent.c_u, st_{\mathrm{E}})$
$\leftarrow (\mathtt{ad}, c^{(b_s^P)}, st_{\mathrm{E}}^{(b_s^P)})$
7: **return** $sent.c_u$

$\underline{\mathsf{Decrypt}(\pi_i^s, \mathtt{ad}, c)}$:

1: **if** $b_s^P = 0$ **then**
2: **return** $\bot$
3: $v \leftarrow v + 1$
4: $(rcvd.ad_v, rcvd.c_v) \leftarrow (\mathtt{ad}, c)$
5: (m, st_{D})
$\leftarrow \mathrm{st.Dec}(K, \mathtt{ad}, c, st_{\mathrm{D}})$
6: **if** $(q = 4) \wedge \mathsf{condition}_4$ **then**
7: $\mathtt{phase} \leftarrow 1$
8: **else if** $(m \neq \bot) \wedge \mathsf{condition}_q$ **then**
9: $\mathtt{phase} \leftarrow 1$
10: **if** $\mathtt{phase} = 1$ **then**
11: **return** m
12: **return** $\bot$

1. **AEAD:**
$\mathsf{condition}_1 = (\nexists w : (c = sent.c_w) \wedge (\mathtt{ad} = sent.ad_w))$
2. **AEAD, no replays:**
$\mathsf{condition}_2 = (\nexists w : (c = sent.c_w) \wedge (\mathtt{ad} = sent.ad_w)) \vee (\exists w < v : c = rcvd.c_w)$
3. **AEAD, no replays, strict incr. order:**
$\mathsf{condition}_3 = (\nexists w : (c = sent.c_w) \wedge (\mathtt{ad} = sent.ad_w)) \vee (\exists w, x, y : (w < v) \wedge (sent.c_x = rcvd.c_w) \wedge (sent.c_y = rcvd.c_v) \wedge (x \geq y))$
4. **AEAD, no replays, strict incr. order, no drops:**
$\mathsf{condition}_4 = (u < v) \vee (c \neq sent.c_v) \vee (\mathtt{ad} \neq sent.ad_v)$

Figure 3: Encrypt and Decrypt oracles for the ACCE-q security game corresponding to secure channel condition q, with stateful AEAD scheme $\mathrm{stAEAD} = (\mathrm{Gen}, \mathrm{Init}, \mathrm{st.Enc}, \mathrm{st.Dec})$.

under authentication failure, they are allowed to output a fatal alert, closing the connection (similarly to TLS). Adjusting the Decrypt *oracle to accommodate analysis of such special implementations is simple: $m \neq \bot$ on line 7 is removed, making the test conditions at all levels the same, namely* $\mathsf{condition}_q$*. However, as such variation is not recommended by protocols in general, and we leave such special cases as non-standard.*

In Fig. 3, $\mathsf{condition}_1$ demands that the received ciphertext/ad pair has been sent at some time. Correspondingly, $\mathsf{condition}_2$ makes the same requirements as $\mathsf{condition}_1$ but also allows $\mathcal{A}$ to win if the ciphertext has previously been received. Formulating the constraint of $\mathsf{condition}_3$ requires finesse: the desired condition must test replay

protection and strict increasing order, but still allow packet dropping. The first half of the disjunction handles authentication; the second half checks for the existence of any two packets for which the sending and receiving order has been switched, and replays are accounted for if $x = y$, i.e. the same packet was received twice ($w < v$). Finally $\mathsf{condition}_4$ checks that the received packet is exactly as sent.

Experiments for the various AEAD levels are linked in [BHMS16]. We inherit these theorems for ACCE-q: Theorem 4.1 demonstrates how security at one ACCE level implies that of lower levels.

Theorem 4.1 (ACCE-(q+1) implies ACCE-q)**.** *Let* Π *be a stateful AEAD scheme with encryption and decryption algorithms* st.Enc *and* st.Dec*, and let* $q \in \{1, 2, 3\}$*.*

For any adversary $\mathcal{A}$*,*

$$\mathbf{Adv}_{\Pi}^{\mathsf{ACCE}\text{-}q}(\mathcal{A}) \leq |\mathbf{Adv}_{\Pi}^{\mathsf{ACCE}\text{-}(q+1)}(\mathcal{A})| \ .$$

The proof follows according that of [BHMS16].

5 Application

The hierarchy of ACCE-q channel oracles is applicable to the analysis of many protocols, including DTLS. Note that such an analysis is non-trivial: despite DTLS being designed on TLS, the initial handshake as well as the channel protocol must account for both packet dropping and re-ordering as well as (potentially) replays. These considerations lead to slight but extremely important changes to the protocol, requiring special attention during analysis and prohibiting a simple extension of TLS results to DTLS.

Table 1 poses possible alignments of ACCE-q models to the real-world protocols discussed in Section 2, as well as the current analysis status of those protocols. This is not an exhaustive list, but demonstrates how the model hierarchy can be used to associate and compare analyses, and analysis goals, of protocols.

We align the QUIC protocol to ACCE-1, and note the ACCE variant used by Lychev et al. [LJBN15]. While the goals of QACCE align to those of ACCE-1, they differ by customization to a multi-stage protocol and clarity of alignment to the original ACCE model. QACCE is customized for handling the two key stages of QUIC (i.e. the early data key stage and data exchange key stage). Such customization could also be done with ACCE-1, or indeed any ACCE-q, by separating the encryptions and decryptions in Encrypt and Decrypt into stages; if the protocol was in stage 1, for example, st.Enc

Level	Protocol	Analysis Status
ACCE-1	DTLS 1.2	Unanalyzed
	QUIC	Analyzed (variant) [LJBN15]
ACCE-2	DTLS 1.2+	Unanalyzed
ACCE-3	802.11	ACCE model optional
ACCE-4	TLS 1.2	Analyzed [JKSS12, KPW13]

Table 1: ACCE-q in analyses. DTLS 1.2+ refers to DTLS inclusive of optional replay protection.

and st.Dec would have a first input of K_1, while in stage 2 they would each take as input K_2.

Comparison to other ACCE models is also a salient issue. Due to slight changes in the channel encryption and decryption oracles for QACCE, the clarity of comparison to ACCE is obscured. Namely, the authors use deterministic nonce-based AEAD instead of probabilistic stateful AEAD (as used in ACCE), and strictly require that plaintexts m_0 and m_1 are of equal length, in comparison to ACCE which more broadly allows optional length-hiding for the ciphertext. While the change to deterministic nonce-based encryption for AEAD could allow for slightly stronger security analysis using IND$-CPA as discussed in [Rog02], the authors instead use IND-CPA[2]. Thus the potential advantage of this modeling choice is precluded.

In comparison, use of our ACCE-q variants would elucidate the connections between analyses instead of obscuring them, while still allowing flexibility for all parts of the ACCE model external to the Encrypt and Decrypt oracles. This provides flexibility, for example allowing for QACCE variations of Reveal queries, new Connect and Resume queries, etc., or partnering definitions.

Ultimately, Theorem 4.1 links analyses of protocols at various levels and allows for a clear comparison of the secure channels established by such protocols. For example, if DTLS was proven to be secure using ACCE-1, or 802.11 was proven to be secure using ACCE-3, then Theorem 4.1 would directly relate the security of those protocols.

[2]Length-hiding IND-CPA combined with INT-CTXT yields length-hiding AE, while stateful length-hiding IND-CPA combined with INT-sfCTXT is equivalent to stateful length-hiding AE [PRS11]. The latter combination is the basis for ACCE (Fig. 2).

6 Conclusion

In the context of modern cryptography, secure channels play an integral role – secure messaging, standard internet protocols, and the extended world of IoT all rely upon the existence of these constructs. These ACCE variants not only enable modeling of protocols under various demands, but builds clear links among the modeling differences of such demands and therefore among the protocols themselves. We view these ACCE variants as fundamental to security analyses of DTLS (both with and without replay protection), which is as yet an open issue, and for clarifying the link between such analyses and those of TLS.

References

[BDK+14] Florian Bergsma, Benjamin Dowling, Florian Kohlar, Jörg Schwenk, and Douglas Stebila. Multi-Ciphersuite Security of the Secure Shell (SSH) Protocol. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 369–381. ACM, 2014.

[BHMS16] Colin Boyd, Britta Hale, Stig Frode Mjølsnes, and Douglas Stebila. From stateless to stateful: Generic authentication and authenticated encryption constructions with application to TLS. In Kazue Sako, editor, *CT-RSA 2016*, volume 9610 of *LNCS*, pages 55–71. Springer, Heidelberg, February / March 2016.

[BSWW13] Christina Brzuska, Nigel P. Smart, Bogdan Warinschi, and Gaven J. Watson. An analysis of the EMV channel establishment protocol. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 13*, pages 373–386. ACM Press, November 2013.

[DR08] T. Dierks and E. Rescorla. *The Transport Layer Security (TLS) Protocol Version 1.2*, 2008. RFC 5426.

[FG14] Marc Fischlin and Felix Günther. Multi-stage key exchange and the case of Google's QUIC protocol. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *ACM CCS 14*, pages 1193–1204. ACM Press, November 2014.

[IEE12] IEEE 802.11. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 2012.

[JKSS11] Tibor Jager, Florian Kohlar, Sven Schäge, and Jörg Schwenk. On the security of TLS-DHE in the standard model.

Cryptology ePrint Archive, Report 2011/219, 2011. http://eprint.iacr.org/2011/219.

[JKSS12] Tibor Jager, Florian Kohlar, Sven Schäge, and Jörg Schwenk. On the security of TLS-DHE in the standard model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 273–293. Springer, Heidelberg, August 2012.

[Ken05] S. Kent. *IP Encapsulating Security Payload (ESP)*, 2005. RFC 4303.

[KPW13] Hugo Krawczyk, Kenneth G. Paterson, and Hoeteck Wee. On the security of the TLS protocol: A systematic analysis. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 429–448. Springer, Heidelberg, August 2013.

[LJBN15] Robert Lychev, Samuel Jero, Alexandra Boldyreva, and Cristina Nita-Rotaru. How secure and quick is QUIC? Provable security and performance analyses. In *2015 IEEE Symposium on Security and Privacy*, pages 214–231. IEEE Computer Society Press, May 2015.

[LSY+14] Yong Li, Sven Schäge, Zheng Yang, Florian Kohlar, and Jörg Schwenk. On the security of the pre-shared key ciphersuites of TLS. In Hugo Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 669–684. Springer, Heidelberg, March 2014.

[PRS11] Kenneth G. Paterson, Thomas Ristenpart, and Thomas Shrimpton. Tag size does matter: Attacks and proofs for the TLS record protocol. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 372–389. Springer, Heidelberg, December 2011.

[Res16] E. Rescorla. *The Transport Layer Security (TLS) Protocol Version 1.3*, 2016. draft-ietf-tls-tls13-14.

[Res17] E. Rescorla. *The Transport Layer Security (TLS) Protocol Version 1.3*, 2017. draft-ietf-tls-tls13-21.

[RM12] E. Rescorla and N. Modadugu. *Datagram Transport Layer Security Version 1.2*, 2012. RFC 6347.

[Rog02] Phillip Rogaway. Authenticated-encryption with associated-data. In Vijayalakshmi Atluri, editor, *ACM CCS 02*, pages 98–107. ACM Press, November 2002.