# Blinded Diffie-Hellman

## Preventing Eavesdroppers from Tracking Payments

Duncan Garrett and Michael Ward

EMVCo Security Working Group
www.emvco.com

**Abstract.** In this paper we present a novel form of ECC Diffie-Hellman key agreement that provides privacy and anti-tracking for contactless payments. The payer's device can be authenticated by a payment terminal using a static public key with associated certificates belonging to the payer's device; however, a passive eavesdropper is unable to determine the static data and keys that might otherwise be used to identify and track the payer. The new protocol has better performance than alternative protocols; it avoids the payer's device having to support signature algorithms with dedicated hashes and it has a security proof given in [3]. The new protocol does not appear in any standards known to the authors.

**Keywords:** Elliptic curve cryptography, Diffie-Hellman, Key agreement, Privacy, Payments, Standards.

## 1 Introduction

The purpose of this paper is to describe a new protocol, called 'Blinded Diffie-Hellman', for establishing and using a secure channel between a payment card and a merchant terminal.

EMVCo (www.emvco.com) has stated [7] that it needs such a protocol as the basis of future payment technology which is intended to use elliptic curve cryptography (ECC) rather than RSA. In the absence of suitable standardised techniques, EMVCo has developed new key agreement technology. Two candidate mechanisms were published in November 2012 [7], and subsequently a security proof for the more efficient mechanism, the Blinded Diffie-Hellman protocol, was published by Brzuska, Smart, Warinschi and Watson [3]. This paper provides a detailed description of this protocol and of its security and performance benefits.

The rest of the paper is organized as follows. We first describe the EMV security requirements for the EMV secure channel and then we review the landscape of currently available standards and mechanisms. In Section 3 we describe the only protocol in the literature (Station-to-Station protocol) that meets the EMV security objectives, and then describe the Blinded Diffie-Hellman protocol that also meets the EMV security objectives but is more efficient. In Section 4 we compare the performance of the new protocol with the Station-to-Station protocol and briefly describe its security

L. Chen and C. Mitchell (Eds.): SSR 2014, LNCS 8893, pp. 79–92, 2014.

properties and associated proof of security (more details of which are provided in [3]). Section 5 contains concluding remarks.

## 2 EMV Security Requirements and the Standards Landscape

### 2.1 EMV Security Requirements

The EMV specifications form the basis for debit and credit card transactions in many parts of the world. The current specifications enable the card to locally authenticate itself to the merchant's terminal using an RSA based PKI and also to remotely authenticate itself (and the payment details) to the card-issuing bank using symmetric cryptography. Local card authentication is achieved by the card creating a digital signature in response to a random challenge from the terminal and the terminal verifying this signature using public key certificates provided by the card and a root public key installed in the terminal. However a consequence of choosing this method of local card authentication is that a passive bugging device located near the terminal might eavesdrop on a contactless transaction and discover the card account number and the public key of the card. Even if the card account number could be protected, the public key of the card is still on its own sufficient to track card use at this terminal and at other terminals whose transactions are similarly eavesdropped.

It would clearly be desirable if, when making the changes to introduce elliptic curve cryptography, the part of the EMV protocol that provides local card authentication could be enhanced to provide protection against such passive eavesdropping. In [7] EMVCo states the security objectives for a new secure channel protocol as follows:

The protocol takes place between a payment card and a merchant terminal and establishes a secure channel between the card and terminal.

The security objectives of the secure channel are

- to provide authentication of the card to the terminal,
- to detect modifications to the communications, and
- to protect against eavesdropping of transactions and card tracking.

It is given that the card contains

- a private key unique to the card,
- the corresponding card public key certified by the card issuer, and
- the corresponding issuer public key certified by a Payment System,

and that the terminal contains the corresponding Payment System public key.

The card and terminal achieve a secure channel by using a key agreement protocol based on Diffie-Hellman, a session key derivation function and a block-cipher based authenticated encryption algorithm (see [29]). The payments may take place in situations that require fast contactless transactions and where the contactless card is only briefly in range of the terminal's contactless field, so good performance of the key agreement protocol is critical.

### 2.2 Standards Landscape

Key management standards within ISO/IEC [22], ANSI [21], IETF [14,15], NIST [18], IEEE [13] and SECG [16] do not appear to have methods for hiding the authenticated party's public key. For example ISO/IEC 11770-3 [23] defines 12 key agreement mechanisms, classifying them in terms of number of passes, implicit key authentication, key confirmation, entity authentication, public key operations, forward secrecy and key freshness. However none of the mechanisms offer secrecy of the public key and only two of them (mechanisms 2 and 11) support one-sided key agreement where only one party is authenticated. The Handbook of Applied Cryptography [12] discusses variants of the Diffie-Hellman algorithm that provide anonymity and this includes the Station-to-Station protocol which does afford anonymity for the participants; however it is described as a scheme where both parties have public keys which can be authenticated and this is not the situation for the EMV system.

One might also consider key transport mechanisms using public key encryption methods such as El Gamal and ECIES as specified in ISO/IEC 18033-2 [27]. However, although such key transport mechanisms provide confidentiality for the payload/message, they do not in themselves protect the confidentiality of the public key used nor offer joint key control.

Finally, there do exist standardised methods for achieving privacy and anonymity objectives (e.g. ISO/IEC 20008 Anonymous Digital Signatures [30], ISO/IEC 20009 Anonymous Authentication [31], ISO/IEC 18370 Blind Signatures [28]); however the privacy issues that these methods aim to solve are different to those considered in this paper (e.g. they provide anonymity via group signatures).

The only 'standard' method in the literature that might meet the EMV privacy objectives would appear to be a one-sided version of the Station-to-Station protocol (see [12]) where both card and terminal conduct a Diffie-Hellman key agreement using ephemeral ECC keys to establish a secure channel and then the card digitally signs its ephemeral key using its static and certified ECC keys. However, the performance of this protocol is not very good (see Section 4) and this therefore led to the design of a new alternative protocol [7] referred to as "Blinded Diffie-Hellman" that is described in detail in the next section.

## 3 Description of Protocols

In this section we describe the Station-to-Station protocol using elliptic curve cryptography. We also describe a variant that meets the one-sided constraints of EMV described in the previous section and the new Blinded Diffie-Hellman protocol that also meets the one-sided constraints of EMV.

In the descriptions below, the following conventions are used:

- Conversion between integers and byte strings and between elliptic curve points and byte strings is not considered. Rules for this are defined in ISO/IEC 15946 [26].

- For simplicity the descriptions do not include important aspects such as bounds checking, parsing and checking that points are on curves.
- Optimisations such as point compression are not considered.
- The derivation of a single symmetric key for authenticated encryption is shown, whereas in practice it is expected that two uni-directional keys would be derived according to standard methods (see Section 4.3).

### 3.1 Station-to-Station Protocol

The original Station-to-Station protocol described in [12] is an extension of the Diffie-Hellman protocol that allows the establishment of a shared secret key between two parties A and B with mutual entity authentication and mutual explicit key authentication. The Station-to-Station protocol also facilitates anonymity whereby the identities of A and B may be protected from eavesdroppers.

The Station-to-Station protocol also employs digital signatures. In our description we assume the use of elliptic curve cryptography for the Diffie-Hellman aspects, so the signature mechanism could, for example, be ECDSA or a variant thereof (see [25]). In the description below we introduce a key derivation function KDF which is not mentioned in [12].

Summary:

- Parties A and B exchange 3 messages

Result:

- key agreement, mutual entity authentication, explicit key authentication, anonymity.

Notation:

- $G$ is a generator of a group of points on an elliptic curve whose order is $n$, and $d \cdot G$ is scalar multiplication of $G$ by an integer $d$ (i.e. the point on the curve resulting from adding $G$ to itself $d$ times).
- $E(k)[m]$ denotes encryption of $m$ under key $k$ using an encryption algorithm E.
- $S_A(m)$ denotes A's signature (e.g. using ECDSA) on $m$.
- Cert(A) denotes A's public key and associated public key certificates.
- KDF is a key derivation function that generates, from a point on the elliptic curve, a symmetric key suitable for the encryption algorithm E.

Protocol steps:

1. A generates an ephemeral private key $d_a$ and generates A's ephemeral public key as $Q_a = d_a \cdot G$
2. $A \rightarrow B : Q_a = d_a \cdot G$

3. B generates an ephemeral private key $d_b$ and generates B's ephemeral public key $Q_b = d_b \cdot G$
4. B computes key $K_b = \mathrm{KDF}(d_b \cdot Q_a)$ and signs $Q_b \| Q_a$
5. $B \rightarrow A : Q_b \| E(K_b)[\ Cert(B) \| S_B(Q_b \| Q_a)]$
6. A computes key $K_a = \mathrm{KDF}(d_a \cdot Q_b)$. A authenticates B's public key using B's certificates and verifies B's signature on $Q_b \| Q_a$. A only accepts the validity of key $K_a$ if the signature verifies successfully. A signs $Q_a \| Q_b$
7. $A \rightarrow B : E(K_a)[\ Cert(A) \| S_A(Q_a \| Q_b)]$
8. B authenticates A's public key using A's certificates and verifies A's signature on $Q_a \| Q_b$. B only accepts the validity of key $K_b$ if the signature verifies successfully.

### 3.2 One-Sided Station-to-Station Protocol

We now describe the variant of the Station-to-Station protocol, which we refer to as 'one-sided Station-to-Station'. This variant would suit the requirements of EMV, namely where only one party, the card, has a static certified public key. Note that according to [7] the EMV secure channel will use authenticated encryption instead of plain encryption, so this is included in the description below.

In our description below we denote the participants as C and T (rather than A and B) to better reflect that the participants are a payment card and a payment terminal, respectively.

Summary:

- Parties C and T exchange 3 messages

In the description below, party C is the card and party T is the terminal. Subscripts *c* and *t* are used for their associated data.

Result:

- key agreement, entity authentication of party C to party T, explicit key authentication to party T, anonymity.

Notation:

- $G$ is a generator of a group of points on an elliptic curve whose order is $n$, and $d \cdot G$ is scalar multiplication of $G$ by an integer $d$ (i.e. the point on the curve resulting from adding $G$ to itself $d$ times).

- $Æ(k)[m]$ denotes encryption of $m$ under key $k$ using an authenticated encryption algorithm Æ.
- $S_C(m)$ denotes C's signature (e.g. using ECDSA) on $m$.
- Cert(C) denotes C's public key and associated public key certificates.
- KDF is a key derivation function that generates, from a point on the elliptic curve, a symmetric key suitable for the encryption algorithm Æ.

Protocol steps:

1. C generates an ephemeral private key $d_C$ and generates C's ephemeral public key as $Q_C = d_C \cdot G$
2. $C \rightarrow T : Q_C = d_C \cdot G$
3. T generates an ephemeral private key $d_T$ and generates T's ephemeral public key $Q_T = d_T \cdot G$
4. T computes key $K_T = KDF(d_T \cdot Q_C)$
5. $T \rightarrow C : Q_T \| Æ(K_T)[ Q_T \| Q_C]$
6. C uses $Q_T$ to compute key $K_C = KDF(d_C \cdot Q_T)$ and uses $K_C$ to authenticate and decrypt the payload $Q_T \| Q_C$. (With correct operation of the protocol $K_C = K_T$; if authentication fails then as soon as this happens the protocol is terminated). C signs $Q_C \| Q_T$
7. $C \rightarrow T : Æ(K_C)[ Cert(C) \| S_C(Q_C \| Q_T)]$
8. T uses $K_T$ to authenticate and decrypt the payload. (With correct operation of the protocol $K_C = K_T$; if authentication fails then as soon as this happens the protocol is terminated) and authenticates C's public key using the certificates and verifies C's signature on $Q_C \| Q_T$. T only accepts the validity of key $K_T$ if the signature verifies successfully.

### 3.3 Blinded Diffie-Hellman Protocol

We now describe the Blinded Diffie-Hellman protocol, a protocol that also suits the requirements of EMV, namely where only party C, the card, has a static certified public key, but which also has advantages compared to the one-sided Station-to-Station protocol (see Section 4.1).

Summary:

- Parties C and T exchange 3 messages

In the description below, party C is the card and party T is the terminal. Subscripts $c$ and $t$ are used for their associated data.

Note that in this description C has a static private key $d_C$ and corresponding public key $Q_C$ with associated certificates Cert($Q_C$).

Result:

- key agreement, entity authentication of party C to party T, explicit key authentication to party T, anonymity.

Notation:

- $G$ is a generator of a group of points on an elliptic curve whose order is $n$, and $d \cdot G$ is scalar multiplication of $G$ by an integer $d$ (i.e. the point on the curve resulting from adding $G$ to itself $d$ times).
- Æ($k$)[$m$] denotes encryption of $m$ under key $k$ using an authenticated encryption algorithm Æ.
- Cert($Q_C$) denotes C's public key and associated public key certificates.
- KDF is a key derivation function that generates, from a point on the elliptic curve, a symmetric key suitable for the encryption algorithm Æ.

Protocol steps:

1. C generates a random integer $r$ ($1 \leq r < n$) and generates C's blinded public key $R = r \cdot Q_C$ .
2. C → T : $R$
3. T generates an ephemeral private key $d_t$ and generates T's ephemeral public key $Q_t = d_t \cdot G$
4. T computes key $K_t$ = KDF($d_t \cdot R$)
5. T → C : $Q_t \parallel$ Æ($K_t$)[ $D_1$], where $D_1$ can be an optional application-level message
6. C uses $Q_t$ to compute key $K_C$ = KDF($rd_C \cdot Q_t$) and optionally uses $K_C$ to authenticate and decrypt the payload $D_1$. (With correct operation of the protocol $K_C = K_t$; if authentication fails then as soon as this happens the protocol is terminated.)

7. C → T : Æ($K_C$)[ $r$ ‖ Cert($Q_C$) ‖ $D_2$ ] where $D_2$ is optional data that the card may wish to send protected to the terminal.
8. T uses $K_t$ to authenticate and decrypt the payload. (With correct operation of the protocol $K_C = K_t$; if authentication fails then as soon as this happens the protocol is terminated) and authenticates C's public key using the certificates and verifies that $R$ received in Step 2 is equal to $r \cdot Q_C$.

The diagram below illustrates the Blinded Diffie-Hellman key agreement using the notation defined above and where the dotted arrow represents a secure channel created using the derived AES keys.

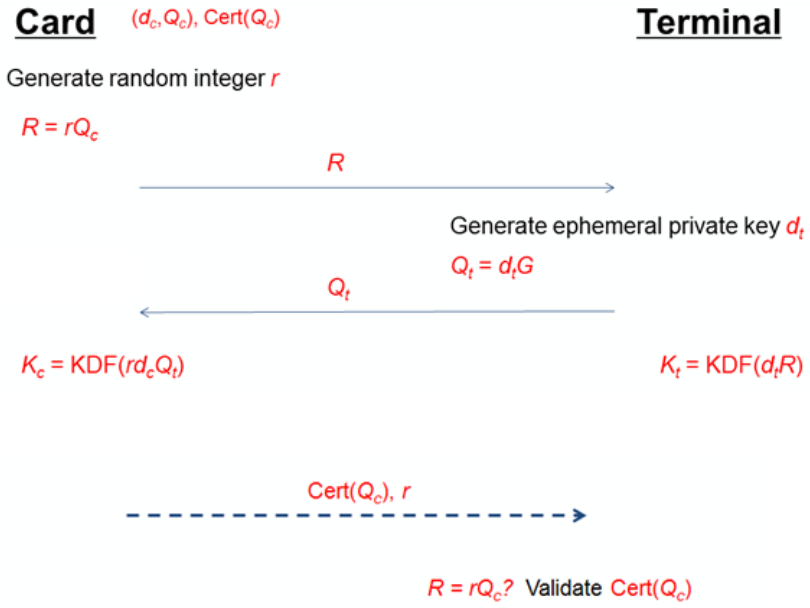


**Fig. 1.** The Blind-Diffie Hellman protocol

## 4 Performance and Security

This section addresses the performance and security of the Blinded Diffie-Hellman protocol.

### 4.1 Performance Comparison with Alternative Protocol

This section makes a performance comparison between the one-sided Station-to-Station protocol and the Blinded Diffie-Hellman protocol. As can be seen from the extract below, the message flows for the two protocols are similar.

| Step | Station-to-Station | Blinded Diffie-Hellman |
|---|---|---|
| 2. C → T | $Q_C = d_C \cdot G$ | $R = r \cdot Q_C$ |
| 5. T → C | $Q_t \parallel Æ(K_t)[\ Q_t \parallel Q_C]$ | $Q_t \parallel Æ(K_t)[\ D_1]$ |
| 7. C → T | $Æ(K_C)[\ \text{Cert(C)} \parallel S_C(Q_C \parallel Q_t\ )]$ | $Æ(K_C)[\ r \parallel \text{Cert}(Q_C) \parallel D_2\ ]$ |

Although the authenticated encryption payload is significantly larger for Station-to-Station compared to Blinded Diffie-Hellman, the authenticated encryption is likely to consume significantly less time than the elliptic curve computations. However, for completeness, we outline the possible impact.

Suppose, for example, that the protocols use an elliptic curve defined over a 256-bit prime field, the public keys (represented as (x,y)-coordinates on the curve) and signatures occupy 64 bytes and the certificates occupy over 256 bytes and the random blinding factor r is 32 bytes. Note that here the certificates are assumed to include at least the public key of the card, a signature of the issuing bank on that public key, the public key of the issuing bank, and a signature of the payment system on that public key.

Then, for the authenticated encryption payloads in steps 5 and 7, the size of the payload for the Station-to-Station is roughly 384 bytes and for Blinded Diffie-Hellman is roughly 288 bytes. If processing of the authenticated encryption by the card would take about 50ms for 100 bytes then this in itself might make Station-to-Station as much as 50ms slower than Blinded Diffie-Hellman.

However the main performance penalty introduced by Station-to-Station is the cost of the ECC operations. With contactless card transactions performance can be critical and the main performance concern is the time taken for ECC operations on the card-side, especially the ECC scalar multiplications. We see that the Blinded Diffie-Hellman protocol requires two scalar multiplications:

- Blinding the card public key in Step 1, and
- Calculating $rd_C \cdot Q_t$ in Step 6

whereas the one-sided Station-to-Station protocol requires two scalar multiplications:

- Generating the ephemeral public key in Step 1, and
- Calculating $d_C \cdot Q_t$ in Step 6

and a signature generation in Step 6.

Thus if we allocate 100ms for a card scalar multiplication or ECDSA calculation (although in reality a signature calculation involves more than a simple scalar

multiplication depending on the algorithm) then Blinded Diffie-Hellman would require 200ms (plus time for the authenticated encryption and communications processing); whereas the one-sided Station-to-Station would require 300ms (plus time for the authenticated encryption and communications processing). This 100ms difference between the two protocols is very significant when one considers requirements for special environments such as transit ticketing.

Note that these performance considerations focus on the card computations required while the card is in the contactless field and although pre-computations may be possible it is likely that both protocols could benefit equally from this (e.g. step 1 for both protocols might be pre-computed).

So clearly Blinded Diffie-Hellman is more efficient and has the added benefit that the card does not need to implement ECC signature algorithms (e.g. ECDSA) and any associated hash function (e.g. SHA256).

The Blinded Diffie-Hellman mechanism would also provide improved performance if a smaller than full-size blinding factor would be used. However in this case the security proof no longer holds (see next section).

### 4.2 Security of Blinded Diffie-Hellman

As stated in Section 2.1, the EMV security objectives for the secure channel using the Blinded Diffie-Hellman protocol are

- to provide authentication of the card to the terminal (entity authentication),
- to detect modifications to the communications (message authentication), and
- to protect against eavesdropping of transactions and card tracking (privacy and unlinkability).

The first and second of these together ensure bilaterally that messages transmitted over the resulting secure channel are guaranteed to come from the other party engaged in the protocol and that for the terminal this other party is guaranteed to be the card identified by the card public key certificate that was verified. This does not, and because the card does not authenticate the terminal it clearly cannot, provide assurance to the card that it is corresponding with a legitimate terminal.

Similarly for the third objective, if the card is engaged in a legitimate transaction with a legitimate terminal then privacy and unlinkability are achieved even in the presence of active adversaries capable of intercepting, modifying, or injecting messages. However this does not rule out the possibility that a rogue device could perform the whole protocol with a card. Thus, in particular, communications confidentiality is assured for the corresponding parties against eavesdroppers (passive and active), but active adversaries can still engage with a card by pretending to be a terminal for the whole protocol run.

Brzuska, Smart, Warinschi, and Watson [3] have proved the security of the Blinded Diffie-Hellman key agreement protocol on the basis that the secure channel uses secure authenticated encryption and the PKI ensures secure authentication of the card public key. Their proof uses reductions in the Random Oracle Model and assumes the intractability of hard problems on the elliptic curve: the Gap Diffie-Hellman problem,

the Diffie-Hellman problem and the Discrete Logarithm problem. The proof uses a modular security technique wherein the reduction takes a component of a hard problem and embeds this into the card's blinded public key. See [9] and [3] for more details of this technique.

[3] provides full details of the security proof for the Blinded Diffie-Hellman protocol (an earlier version can also be found at `http://www.iacr.org/2013/031`).

Although performance would certainly be improved by using a smaller blinding factor (e.g. 128 bits), the security proof from [3] would no longer hold. Indeed this possibility has also been considered in [4] where it is shown that bounded height discrete logarithm attacks may become feasible when using shorter blinding factors.

Note that because the card initiates the protocol rather than the terminal, it is possible that a terminal may be able to partially control the value of the uni-directional keys. As described in Section 8 of ISO/IEC 11770-3 [23], the terminal could control the value of $s$ bits in the established key at the cost of generating $2^s$ candidate values for their ephemeral key in the time interval between discovering the card's blinded public key and choosing their ephemeral key. However with no terminal authentication and with the requirement for very fast transactions, this type of attack is not so relevant. If it would be considered relevant then the terminal could instead initiate the protocol (and it is understood that the security proof of [3] would still hold in this case) and/or the terminal could be required to make an initial commitment of its ephemeral key.

The Blinded Diffie-Hellman key agreement protocol requires the use of a secured communications channel after the key has been agreed and derived. This secure channel is constructed using the derived keys and enables the card public key certificates and the blinding factor to be sent protected to the terminal so that the terminal can authenticate the card public key and validate the blinded public key received previously. The secure channel should use a standard authenticated encryption algorithm (e.g. mechanism from ISO/IEC 19772 [29]) and a standard technique for deriving keys (see next section).

### 4.3 Uni-directional Keys and Key Derivation Function KDF()

According to the security proof in [3], the terminal and card should use message counters to ensure statefulness and should use uni-directional keys. Such uni-directional keys can be easily derived as $(K_1, K_2) = \mathrm{KDF}(Q)$, where

- for the terminal $Q = d_t \cdot R$
- for the card $Q = r\, d_c \cdot Q_t$

where $Q$ is a point on the elliptic curve and is the same for both terminal and card (assuming the protocol is operating correctly).

The key derivation function KDF() can be any standard key derivation function (see for example ISO/IEC 11770-6 [24] and NIST SP-800-56A [18], NIST SP-800-56C [19], NIST SP-800-108 [20]) so long as it is secure and known to both card and terminal.

For completeness this section concludes by providing an example where the key derivation function KDF() generates two AES keys $K_1$ and $K_2$ by encrypting two fixed strings using a key derivation key *KDK* that has been generated by MACing the x-coordinate of $Q$ with an all-zero key (per CMAC in ISO/IEC 9797-1 [22] and NIST SP800-38B [17]). As with the performance example, the example below assumes that the elliptic curve is defined over a 256-bit prime field.

1. Key Extraction

Convert the x-coordinate of $Q$ to be the 256-bit string $Z = Z_0 || Z_1$ where $Z_i$ is a 128-bit string for i=0,1.

2. Randomness Extraction

Compute a 128-bit *KDK* as an AES128 CBC MAC of Z using the all zero key (no padding)

- $KDK$ = AES128[0]( AES128[0]($Z_0$) xor $Z_1$)

3. Key Expansion

Derive uni-directional keys as

- $K_1$ = AES128[$KDK$] ( 0x00 || $S$)
- $K_2$ = AES128[$KDK$] ( 0x01 || $S$)

where the value $S$ could be any 15 byte field and could be chosen to represent a KDF version number, card/terminal identifiers, authenticated encryption algorithm identifier, an extra card nonce, and/or the bit-length of the total keying material produced.

Note that one could alternatively take Z as a function of the x and y coordinates, however, the use of the x-coordinate only is consistent with NIST SP-800-56A [18].

## 5 Conclusions

This paper has described a new key agreement protocol that has important privacy and performance properties that are not available in existing standards. The protocol has a security proof and a potential application in card payments.

**Acknowledgements.** We would like to thank the anonymous referees and Chris Mitchell for their helpful comments.

## References


1. Blake-Wilson, S., Menezes, A.: Authenticated Diffie-Hellman key agreement protocols. In: Tavares, S., Meijer, H. (eds.) SAC 1998. LNCS, vol. 1556, pp. 339–361. Springer, Heidelberg (1999)
2. Blake-Wilson, S., Johnson, D., Menezes, A.: Key agreement protocols and their security analysis. In: Darnell, M. (ed.) Cryptography and Coding 1997. LNCS, vol. 1355, pp. 30–45. Springer, Heidelberg (1997)
3. Brzuska, C., Smart, N.P., Warinschi, B., Watson, G.: An Analysis of the EMV Channel Establishment Protocol. In: ACM CCS 2013, pp. 373–386. ACM (2013)
4. Blackburn, S., Scott, S.: The discrete logarithm problem for exponents of bounded height. J. Computation and Mathematics 17(Special Issue A), 148–156 (2014)
5. Canetti, R., Krawczyk, H.: Analysis of key-exchange protocols and their use for building secure channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001)
6. Dagdelen, O., Fischlin, M., Gagliardoni, T., Marson, G.A., Mittelbach, A., Onete, C.: A Cryptographic Analysis of OPACITY (2013), `http://www.iacr.org/2013/234`
7. EMVCo: EMV ECC Key Establishment Protocols. Draft, 1st edn. (2012), `http://www.emvco.com/specifications.aspx?id=243`
8. Goldberg, G., Stebila, S., Ustaoglu, B.: Anonymity and one-way authentication. In: Key Exchange Protocols. Designs, Codes and Cryptography, vol. 67(2), pp. 245–269 (May 2013)
9. Kudla, C., Paterson, K.G.: Modular security proofs for key agreement protocols. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 549–565. Springer, Heidelberg (2005)
10. Hankerson, D., Menezes, A., Vanstone, S.: Guide to Elliptic Curve Cryptography. Springer (2004) ISBN 0-387-95273-X
11. Law, L., Menezes, A., Qu, M., Solinas, J., Vanstone, S.: An efficient protocol for authenticated key agreement, Dept. C & Q, Univ. of Waterloo, CORR 98-05 (1998)
12. Menezes, A., Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography. CRC Press (1997)
13. IEEE P1363: A standard for RSA, Diffie-Hellman, and Elliptic-Curve cryptography (1999)
14. IETF RFC 2631, Diffie-Hellman Key Agreement Method (June 1999)
15. IETF RFC 4492, Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) (2006)
16. Certicom Research, Standards for Efficient Cryptography (2000)
17. NIST Special Publication 800-38B, Recommendation for Block Cipher Modes of Operation: the CMAC Mode for Authentication (May 2005)
18. NIST Special Publication 800-56A, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised) (March 2007)
19. NIST Special Publication 800-56C, Recommendation for Key Derivation through Extraction-then-Expansion (November 2011)
20. NIST Special Publication 800-108, Recommendation for Key Derivation using Pseudorandom Functions (Revised) (October 2009)
21. ANSI X9.63, Public Key Cryptography for the Financial Services Industry Key Agreement and Key Transport Using Elliptic Curve Cryptography (2011)
22. ISO/IEC 9797-1: Information technology — Security techniques — Message authentication codes — Part 1: Mechanisms using a block cipher (2011)
23. ISO/IEC 11770-3: Information technology — Security techniques — Key management — Part 3: Mechanisms using asymmetric techniques (2008)

24. ISO/IEC CD 11770-6. Information technology — Security techniques — Key management — Part 6: Key derivation
25. ISO/IEC 14888-3: Information technology — Security techniques — Digital signatures with appendix — Part 3: Discrete logarithm based mechanisms (2006)
26. ISO/IEC 15946-1: Information technology — Security techniques — Cryptographic techniques based on elliptic curves (2008)
27. ISO/IEC 18033-2: Information technology — Security techniques — Encryption algorithms — Part 2: Asymmetric ciphers (2006)
28. ISO/IEC CD 18370-1. Information technology — Security techniques — Blind digital signatures — Part 1: General
29. ISO/IEC 19772: Information technology — Security techniques — Authenticated Encryption (2009)
30. ISO/IEC 20008-1: Information technology — Security techniques — Anonymous digital signatures — Part 1: General (2013)
31. ISO/IEC 20009-1: Information technology — Security techniques — Anonymous entity authentication — Part 1: General (2013)