Prof. Dr. Ernst-Günter Giessmann
Department of Computer Science
Humboldt-Universität zu Berlin[1]

Berlin, 2023-04-13

Review of the equivalence of ep2 communication protocol messages to *Key Blocks* according to PCI PIN Security requirements (TR-31 and TR-34)

Summary

Under the ep2 system, symmetric keys are transmitted using asymmetric methods. If the interoperable formats and protocols defined by [X9 TR-34] are not used, it must be demonstrated that the security requirements of the Payment Card Industry (PCI) [PCI PIN Security] are met:

> **Requirement 18:** Procedures must exist to prevent or detect the unauthorized substitution (unauthorized key replacement and key misuse) of one key for another or the operation of any cryptographic device without legitimate keys.

The result of this review confirms that these requirements are met for the generation and transmission of symmetric keys using ep2's asymmetric procedures.

Security objectives of PCI PIN Security

The security objective to be met is (Control Objective 5):

> "Keys are used in a manner that prevents or detects their unauthorized usage."

The detailed PCI requirements for Key Blocks are (Requirement 18-3 [PCI PIN Security, p. 58]):

> "Encrypted symmetric keys must be managed in structures called key blocks. The key usage must be cryptographically bound to the key using accepted methods.
>
> The phased implementation dates are as follows:
>
> - Phase 1 - Implement Key Blocks for internal connections and key storage within Service Provider Environments - this would include all applications and databases connected to hardware security modules (HSM). Effective date: 1 June 2019.
> - Phase 2 - Implement Key Blocks for external connections to Associations and Networks. Effective date: 1 January 2023.
> - Phase 3 - Implement Key Block to extend to all merchant hosts, point-of-sale (POS) devices and ATMs. Effective date: 1 January 2025.
>
> Acceptable methods of implementing the integrity requirements include, but are not limited to:
>
> - A MAC computed over the concatenation of the clear-text attributes and the enciphered portion of the key block, which includes the key itself - e.g. TR 31;
> - A digital signature computed over that same data e.g., TR-34;
> - An integrity check that is an implicit part of the key-encryption process such as that which is used in the AES key-wrap process specified in ANSI X9.102."

The use of asymmetric algorithms is addressed in the FAQ:

Q 19 September 2020: HSMs are required to support key blocks using the ASC X9 TR-31 key-derivation methodology for TDES keys, and for AES keys must support either the TR-31 metho-

[1] e-mail: giessmann@informatik.hu-berlin.de
Postal address: Prof. Dr. Ernst-Günter Giessmann, 5xS Consulting, Am Tonberg 28, D-16727 Velten

dology and/or the ISO 20038 methodology. TR-31 and ISO 20038 are methods to package keys (the key blocks) for conveyance or storage, but they use symmetric mechanisms for that and for key conveyance require a symmetric key exchange key that is pre-shared for use as the key block protection key. Where a symmetric key is not previously established with a POI device for remote key distribution, and asymmetric methods will be used, is it required to support a key block methodology?

A: Yes. A method such as ASC X9 TR 34: Interoperable Method for Distribution of Symmetric Keys using Asymmetric Techniques: Part 1 - Using Factoring-Based Public Key Cryptography Unilateral Key Transport must be used. Under TR-34, similar to TR-31 and ISO 20038, the Key Block consists of three parts:

- The Key Block Header (KBH) which contains attribute information about the Key and the Key Block.
- The confidential data that is being exchanged/stored
- The Key Block Binding Method

However, TR-34 uses asymmetric methods for the Key Block Binding Method, instead of the symmetric methods used in TR-31 or ISO 20038 which require that a symmetric key was previously exchanged between the POI device and the KDH.

Nevertheless, it is explicitly allowed to use equivalent methods ([Supp. 18-3, p. 9].

Q July 2019: PIN Security Requirement 18-3 requires the implementation of key blocks. Interoperable methods include those defined in ANSI X9.143 and ISO 20038. The requirement also allows for any equivalent method whereby the equivalent method includes the cryptographic binding of the key-usage information to the key value using accepted methods. How are equivalent methods determined?

A: Equivalent methods must be subject to an independent expert review and said review is publicly available:

- The review by the independent expert must include proof that in the equivalent method the encrypted key and its attributes in the Key Block have integrity protection such that it is computationally infeasible for the key to be used if the key or its attributes have been modified. Modification includes, but is not limited to:
  - Changing or replacing any bit(s) in the attributes or encrypted key
  - Interchanging any bits of the protected Key Block with bits from another part of the block
- The PTS laboratory will validate that any device vendors implementing this methodology have done so following all guidelines of said evaluation and peer review, including any recommendations for associated key management.

In the following, the secret keys to be protected are identified on the basis of the documents provided and their generation, transmission and use are examined. The criteria of the ANSI standard X9 TR-34-2012 [X9 TR-34] were used as a basis.

Key blocks according to TR-34

The data structure of key blocks according to TR-34 [X9 TR-34X9 TR-31] uses the same elements as TR-31/X9.143 [X9.143]:

- the key block header (KBH), which contains attribute information about the overall structure and intended use of the key to be exchanged or stored,
- the confidential data to be exchanged or stored, and
- the cryptographic verification value (signature) that binds the key block header and the key data together.

← TR-31 KBH →					← TR-31 KBH →	
Header	Header (optional)	Version	ID_{KDH} (IssuerAndSerial Number	Key (k_n)	Header	Header (optional)
← Plaintext (Data) →		← Encrypted (EnvelopedData) →				
← Signed (SignedData) →						

For encryption (*EnvelopedData*) and signatures (*SignedData*), the data structures according to Cryptographic Message Syntax [X9.73] are used in TR-34 ([X9 TR-34, chap. 5, p. 23 ff.]).

Both the *Key Block Header* and the encrypted data have standardized formats down to the last detail and use fixed identifiers. Due to backwards compatibility the transition to new formats in the field requires a certain transition period, therefore equivalent formats are currently also permitted.

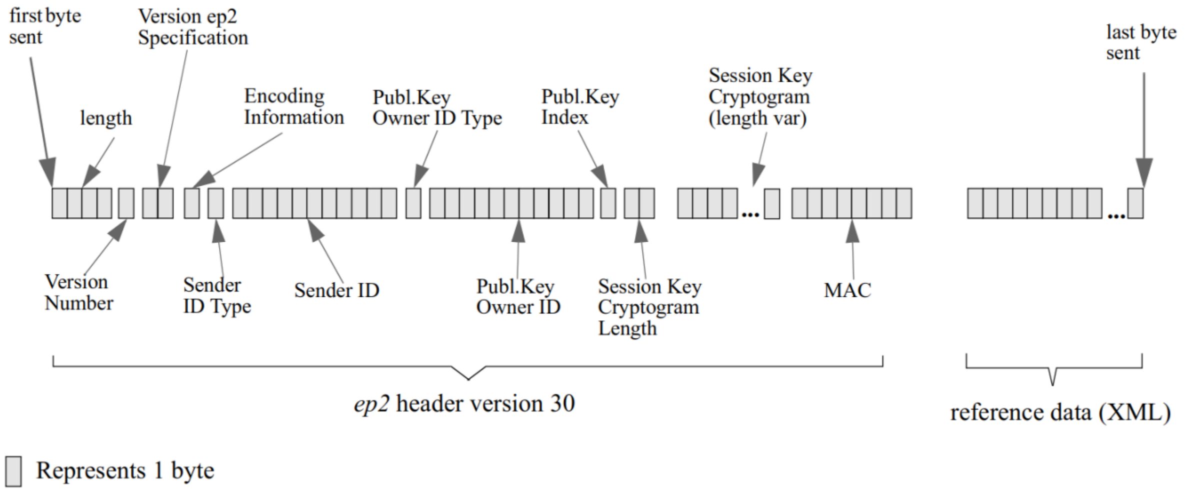
For the following, it should be noted that the *IssuerAndSerialNumber* ID_{KDH} of the (Key Distribution Host) is contained not only in the unencrypted part of the header, but also in the encrypted signature block. For the reason TR-34 states ([X9 TR-34, p. 25]):

> *Additionally, the ID_{KDH} is included in the encrypted data to prevent a signature stripping attack.*

So it is a security requirement to prevent unsigned key blocks. Anyway, this is not relevant in the following because the integrity of the encrypted blocks is not protected by signatures. Note that the SenderID is included into the encrypted block ([SecSpec, Table 5, p. 9-35]), so that sender substitution is excluded for ep2, too.

Key transmission

When symmetric keys are transmitted in the various communication channels (cf. [SecSpec, Table 2, p. 9-15f]), they are encrypted. The Session Key Cryptogram is transmitted in a block with the following structure ([InterfaceSpec, Fig. 8, p. 10-31f]).



Key derivation

The transmitted keys are not used directly, the corresponding key variants are derived from the transmitted using a salt value and an associated initial value, depending on the mode of use. The procedure is described in [SecSpec, p. 33 and 63f]. It is based on the generally accepted HKDF procedure according to RFC5869 with the hash function SHA-256.

This procedure meets the same security requirements for key derivation as the CMAC calculation proposed in X9.143.

The specification of the key type in the *Key Block Header* required by the PCI requirements is therefore not necessary, since only the corresponding derived keys are used within the framework of the ep2 protocol for the intended purpose. Each key generated by a separate "type" constant (64 bits) and a random secret salt value (256 bits) can only be used for the purpose determined by the constant. The HKDF procedure, which is based on the SHA-256 hash function, also guarantees that the initial session key and the salt value to be used cannot be determined from a derived key.

Various key attributes shall be specified in the Key Block Header according to TR-31. In addition to the type of the key [X9 TR-31, Table 6]/[X9.143, Table 2]), the associated algorithm ([X9 TR-31, Table 7]/]/[X9.143, Table 3]) and the mode of use [X9 TR-31, Table 8]/]/[X9.143, Table 4]) are also included. The TR-31 explicitly allows the key derivation as the mode of use 'X' (0x58):

> Key used to derive other key(s) (X) – The key is used only in a key derivation process that produces one or more derived keys ([X9.143, p. 20]).

Additionally in the ep2 protocol a mode of use 'G' (0x47) is included, but only used for transmission of keys to generate the surrogate PAN to the terminal in the Key Block Header according to X9.143:

> Generate only (G) – The key can be used to generate a verification value (e.g. MAC or PIN Offset), but it cannot be used to verify the value ([X9.143, p. 20]).

Since there is for the session keys only one mode of use in the ep2 protocol, such a specification is not required for the equivalence analysis.

Encryption of the session keys

The encryption of the session key in the transmitted block uses the asymmetric encryption method RSAES-OAEP [RFC8017]. The public key is used for encryption only, the secret key for decryption. The corresponding public keys are stored in the terminal.

RSAES-OAEP is provably secure with the hash function SHA-256 used as mask generation function ([RFC8017], 8.2, p. 66]). The key length of 2048 is accepted as secure for the period up to 2030 and their support is even required by TR-34 [X9 TR-34, A.2.6]. The ep2 specification also provides for key lengths of 3072 and 4096 bits. A corresponding migration concept then guarantees security beyond 2030.

Integrity protection of the session keys

The block ([InterfaceSpec, Fig. 8, p. 10-31f]) containing the encrypted session keys has no additional integrity protection such as a signature according to TR-34 or a MAC value according to TR-31. The MAC transmitted in the key block ([InterfaceSpec, Fig. 8, p. 10-31f]) secures the transmission of the XML message in the corresponding communication channels: Terminal ↔ Acquirer, POS management system/Terminal ↔ Acquirer, Service Center ↔ Acquirer, when sending or receiving.

But the integrity protection of the transmitted key data is indirectly secured in the ep2 system, and it is ensured, that the key bits remain protected against modification.

Recall that in addition to digital signatures and MAC security, other protection mechanisms are also permitted:

> Acceptable methods of implementing the integrity requirements include, but are not limited to: MAC…, digital signature …, integrity check that is an implicit part of the key-encryption process … ([PCI PIN Security, p. 58])

Because in an asymmetric encryption procedure potentially anyone is able to encrypt data with the recipient's public key, the authenticity of the data must be verified.

The message *M* (a key block of 97 bytes) to be encrypted is the concatenation of the header byte (0x7F), the sender ID, the session key, the component secret, the salt value to be used for key derivation, and the random nonce values for encryption with AES in CTR mode when sending and receiving (cf. [SecSpec,Table 5, S. 9-35]):

Field Name	Length	Description	Fmt
Data Header	1	Value '7F' (hex)	b
Sender Identifier	8	Alphanumeric number identifying the sender of the message. Following rules shall be used for the truncation • <Acquirer Identifier>, <PMS Identifier> and <Service Center Identifier> n11: • left truncation of 3 digits and port to an8<Terminal Identification> uan8: copy to an8	an
<Session Key>	16	Current session key	b
<Component Secret>	16	<Component Secret> of the sender	b
<Salt>	32	<Salt> value used during the extraction phase of the key derivation function (KDF)	b
<Nonce Variant Encryption Send>	12	Nonce used in encipherment with <Session Key Variant Encryption>	b
<Nonce Variant Encryption Receive>	12	Nonce used in encipherment with <Session Key Variant Encryption>	b

RSAES-OAEP encodes the message *M* according to the following EME-OAEP scheme ([RFC8017, Fig. 1, p. 24]).

```
                    +----------+------+--+-------+
               DB = |  lHash   |  PS  |01|   M   |
                    +----------+------+--+-------+
                                   |
         +----------+              |
         |   seed   |              |
         +----------+              |
               |                   |
               |-------> MGF ---> xor
               |                   |
      +--+     V                   |
      |00|    xor <----- MGF <-----|
      +--+     |                   |
        |      |                   |
        V      V                   V
      +--+----------+----------------------------+
EM =  |00|maskedSeed|          maskedDB          |
      +--+----------+----------------------------+
```

RSAES-OAEP's own integrity check after RSA decryption verifies the first byte of EM (0x00), the `lhash` value (32 bytes), the padding string of zero bytes (93 bytes), and the separation byte (0x01) in the unmasked data block DB. Additionally the decrypted key block is checked. The SenderID (8 bytes) and Component Secret (16 bytes) must correspond to the expected values.

These additional checks form a reliable integrity protection of the key data. The creation of a valid message for a modified key in the message *M* is prevented by the Component Secret, the EME-OAEP encoding guarantees that any change of a bit in the encrypted Key Block leads to an error during the checks of the decrypted message *M*.

MAC checking of the XML message

According to the ep2 specification, the MAC calculation is not applied to the data itself, but to the hash value of the data ([SecSpec, p. 9-36f]). With the SHA-256 hash function and the length of the exchanged messages, this does not imply any restriction on security.

The MAC calculation is again based on the HMAC function according to RFC2104, using the specially derived keys for sender and receiver.

X9.143 uses a CMAC reduced to 8 bytes for the key blocks, but allows for HMAC keys too. There is even a separate optional block defined for these keys specifying the hash function used and this definition includes SHA-256. Since HMAC and CMAC do not differ in their security properties, the use of the HMAC value reduced to 8 bytes is an equivalent procedure.

Key wrapping for 'PAN Receipt' and 'PAN Surrogate' usage

Section 8.7 of the ep2 Security Specification ([SecSpec]) uses for the transmission of symmetric keys (Key PAN Receipt and Key PAN Surrogate) key blocks according to X9.143.

The individual data fields for the block correspond exactly to the specifications of X9.143, chapter 6.3.

In order to distinguish these keys from each other, the key mode of use 'X' (Key used to derive other keys) is assigned for the PAN Receipt key and the usage purpose 'G' (Generate only) is assigned for the PAN Surrogate keys (primary and secondary). The key mode of use is specified in the key block header as required by X9.14 .

Again the HKDF method based on the hash function SHA-256 according to RFC5869 is used for the key derivation. The hash function is however not specified in an optional field, since only SHA-256 is used.

Security Review Summary

The procedures used at ep2 are supported by algorithms classified as suitable in X9 TR-34 and X9.143. These include SHA-256, RSAES-OAEP and HMAC. They guarantee that any changes of individual bits are detectable. Removal or swapping of individual bits or data groups with each other is detected for the Session Key Cryptogram after decryption by EME-OAEP encoding check and SenderID and Component Secret verification. During key wrapping, the requirements of X9 TR-31/X9.143 are met directly.

The parameters used, such as SHA-256 and RSA key lengths of 2048, 3072 and 4096 bits, are at a high security level.

The security requirements for the structure of the key blocks according to TR-34 and TR-31 are met by equivalent procedures. The ep2 protocol guarantees the binding of the derived keys to their intended use by means of a key derivation specific to the intended use. Misuse or change of the key data is ruled out by the integrity check during RSAES-OAEP decryption and EME-OAEP decoding.

Bibliography

[InterfaceSpec] eft/pos 2000 Interface Specification, Version 8.1.0, 2022-12-31

[PCI PIN-Security] Payment Card Industry (PCI) PIN Security Requirements and Testing Procedures, Version 3.1, March 2021

[PIN-FAQ] PTS PIN Security Requirements, Technical FAQs for use with Version 3, Payment Card Industry (PCI), September 2021

[HSM-FAQ] PTS HSM Security Requirements, Technical FAQs for use with Version 3, Payment Card Industry (PCI), August 2022

[RFC2104] H. Krawczyk, M. Bellare, R. Canetti:, HMAC: Keyed-Hashing for Message Authentication, IETF, Informational RFC 2104, 1997-02

[RFC5869] H. Krawczyk, P. Eronen: HMAC-based Extract-and-Expand Key Derivation Function (HKDF), IETF, Informational RFC 5689, 2010-05

[RFC8017] K. Moriarty, B. Kaliski, J. Jonsson, A. Rusch: PKCS #1: RSA Cryptography Specifications Version 2.2, IETF, Informational RFC 8017, 2016-11

[SecSpec] eft/pos 2000 Security Specification, Version 8.1.0, 2022-12-31

[X9.73] Cryptographic Message Syntax — ASN.1 and XML, Accredited Standards Committee X9, American National Standards Institute, 2017

[X9.143] Retail Financial Services Interoperable Secure Key Block Specification (ANSI X9.143-2022), Accredited Standards Committee X9, American National Standards Institute, 2022-05

[X9 TR-34] Interoperable Method for Distribution of Symmetric Keys using Asymmetric Techniques: Part 1 – Using Factoring-Based Public Key Cryptography (Unilateral Key Transport), American National Standards Institute, 2012-08

[X9 TR-31] Interoperable Secure Key Exchange Key Block Specification (ASC X9 TR 31-2018), Accredited Standards Committee X9, American National Standards Institute, 2018

[Supp. 18-3] Information Supplement: PIN Security Requirement 18-3 − Key Blocks, PIN Assessment Working Group PCI Security Standards Council, July 2022