X9.143 and PCI PIN Compliant Key Blocks

Richard Kisley, Chief Engineer – HSM, IBM Corporation
X9.org, ISO TC68 SC2 WG13 & WG11, PCIP

Agenda

Proper Names

- ASC X9 TR-31-*date* (2005,2010,2018)
- ANSI X9.143-*date* (2021,2022)
- ISO 20038-2017

- What are key blocks?
- Where do key blocks appear in PCI SSC documents?
- The journey: TR-31-2005 to X9.143-2022
- X9.143, TR-31, ISO 20038: security & pitfalls
- X9.143 key management features

PCI Security Standards Council®

What is a key block / key token?

Basic block / token

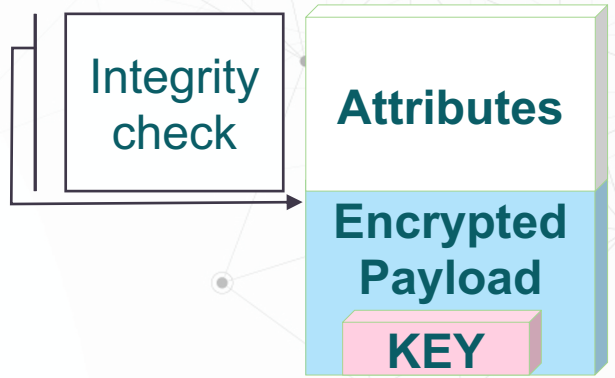
- **Attributes:**
  - usage / management controls
- **Key:**
  - the secret!
- **Integrity protection:**
  - protects the attributes
  - Options:
    - copy/hash attributes
    - XOR to key (variant),
    - MAC
- **Encryption:**
  - protects the key

Wrapping Method

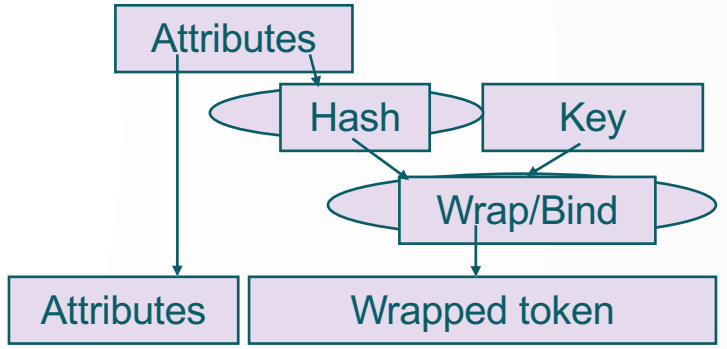
integrity protection and encryption as 1 ‘mechanism’

- **AESKW:**
  - copy or hash of attributes is placed in the payload
- **TR-31:**
  - MAC of attributes and encryption of key use ‘uniquely derived’ keys
  - A,C: encrypt-then-MAC
  - B,D: MAC-then-encrypt

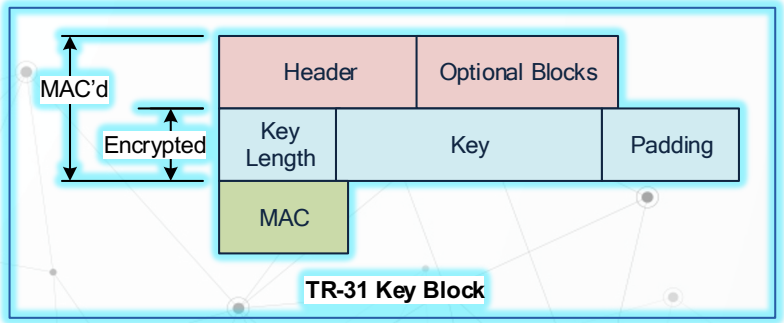
Key Block



AESKW



TR-31



PIN Security Version 3.1 March 2021

18-3 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*.

Key Blocks in PCI SSC documents:

Dec 2014: PCI PIN 2.0
Req 18-3 added (effective 1-Jan-18)

Apr 2017: split to 3 phases:
Jun 2019, Jun 2021, Jun 2023

Jun 2017: Key Block Info Supplement

Jun 2019: FAQ to Info Supplement

Sep 2020: HSM & POI FAQ: criteria for proprietary key blocks

Jan 2021: FAQ: **un-reviewed** proprietary methods sunset 1-Jan-2023

Mar 2021: Phase 3 due 1-Jan-2025

Jul 2022: FAQ2 to Info Supplement

Jan 2023: TR-31/X9.143 **or reviewed** proprietary key blocks must be used.

What are valid key blocks?

Q: Do I need to migrate from proprietary key blocks to TR-31 / X9.143?
A: NO, but you need the independent review for your auditor.

Q 17 **September 2020: PIN Security Requirement 18-3 requires the implementation of key blocks. Interoperable methods include those defined in ASC X9 TR-31 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 for peer review:*

What defines an 'Independent Expert'?

An Independent Expert possesses the following qualifications:

- *Holds one or more professional credentials applicable to the field, e.g., doctoral-level qualifications in a relevant discipline or government certification in cryptography by an authoritative body (e.g., NSA, CES, or GCHQ) and*
- *Has ten or more years of experience in the relevant subject and*
- *Subscribes to an ethical code of conduct and would be subject to an ethics compliance process if warranted and.*
  - *Has published at least two articles in peer-reviewed publications on the relevant subject or*
  - *Is recognized by his/her peers in the field (e.g., awarded the Fellow or Distinguished Fellow or similar professional recognition by an appropriate body, e.g., ACM, BCS, IEEE, IET, IACR).*

What are ‘equivalent’ methods?

Summary:

- (a) unique attributes for PIN keys – enforced by device, cryptographically bound
- (b) key length hidden
- (c) key is for one algorithm
- (d) bad are rejected before use
- …
- (h) no variants, and separately derived keys for confidentiality and authenticity
- …and the highlighted text also tells **why key blocks are important!**

Does TR-31/X9.143 have all this?
Stay tuned….

Sep 2020: PCI PTS HSM & POI FAQ

A *“Equivalency” must be demonstrated in the context of security proofs. The equivalent method must provably accomplish the functions of key integrity, restricting key usage, preventing key reuse, and the secrecy of keys. Specifically, an equivalent key block scheme must minimally offer the following properties:*

a) *It must prevent the loading of PIN, MAC, and/or Data keys - or any keys used to manage these within the key hierarchy - from being used for another purpose. IPEK, KEKs, and derivation keys must be uniquely identified where supported.*
b) *It must prevent the determination of key length for variable length keys.*
c) *It must ensure that the key can only be used for a specific algorithm (such as TDES or AES, but not both).*
d) *It must ensure a modified key or key block can be rejected prior to use, regardless of the utility of the key after modification. Modification includes changing any bits of the key, as well as the reordering or manipulation of individual single DES keys within a TDES key block.*
e) *Where different key block formats are supported, with some providing the above protections and some not, it must be humanly readable from the key block prior to loading/use which format is implemented. E.g., by looking at the commands sent to the device.*
f) *It must support all symmetric algorithms implemented by the device(s) that are to use the key blocks.*
g) *Where asymmetric algorithms are supported, the algorithm type, padding and signature formats must be identified in the key block.*
h) *It must use NIST approved modes of operation, with separate keys used for confidentially and authenticity. Any keys used must not be related in a reversible way.*

Quick History of TR-31/X9.143

TR-31 is a *teenager*!
Often helpful, but *sometimes annoying…*

- **ASC X9 TR-31-2005 *for key exchange***
  - TR = *technical report*
  - ‘A’ wrapping (*variant*)
  - DES/TDES - only one with *defined payload*
  - Usages: BDK, cipher, card/EMV, KEK, MAC, PIN ENC, PIN VER
  - 3 optional block IDs
    PITFALL-1*: no layouts, no examples*
- **ASC X9 TR-31-2010**
  - ‘B’,’C’ wrapping
  - Still DES/TDES…
  - +Usage: DUKPT Initial key
  - 3 optional block IDs
- **ISO 20038:2017**
  - ‘D’,’E’ AES wrapping
  - Real AES & HMAC
  - Usages for RSA, ECC
  - 6 optional block IDs …*but still no layouts*
- **ASC X9 TR-31-2018**
  - ‘D’ AES wrapping
  - Real AES…
  - **PITFALL-2***: different HMAC hash identifier*
  - Usages for RSA, ECC
  - RSA & ECC *payload defined!*
  - *9 optional block IDs & 3 layouts!*

➢ **ANSI X9.143-2021 (March)**
  - Standardized: ‘compliant with X9.143’ has meaning
  - ‘Local’ context
  - ‘A’,’C’ (variant) deprecated
  - 16 optional block IDs, all *with layouts + examples*!
  - Key size SHALL be hidden
  - Unknown Optional blocks SHALL be rejected

➢ **ANSI X9.143-2022 (June)**
  - Quick errata
  - Key size SHOULD be hidden
  - Unknown Optional blocks SHOULD be rejected

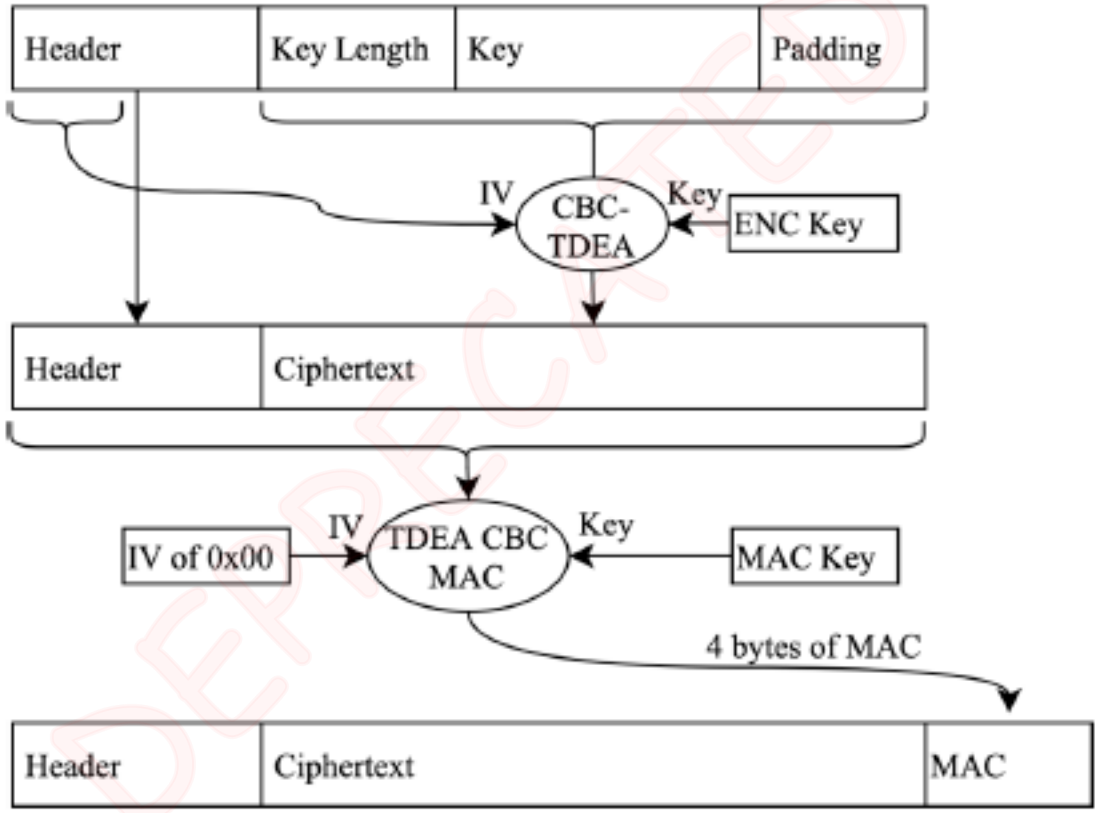
Security first…
TR-31 legacy ‘A’/’C’ wrapping methods (they’re the same)

‘Encrypt-then-MAC’

- **Variant ENC & MAC key calculation**
  - ENC & MAC keys created via XOR of wrapping key with 2 constants
- **Violates CBC mode** (encryption step):
  - NIST SP 800-38A: (6.2) "The IV need not be secret, but it must be unpredictable"
  - The MAC IV (the header) is not secret!
- **Encryption attack**: if you break TDES, you can reverse the variant and get the KEK.
- **MAC attack**: the MAC is over the ciphertext, which is available to you!

PITFALL-3: ‘A’/’C’ not really compliant: FAQ (h) is violated by variant key calculation & NIST CBC mode break

Figure 11 - Key Variant Binding Method

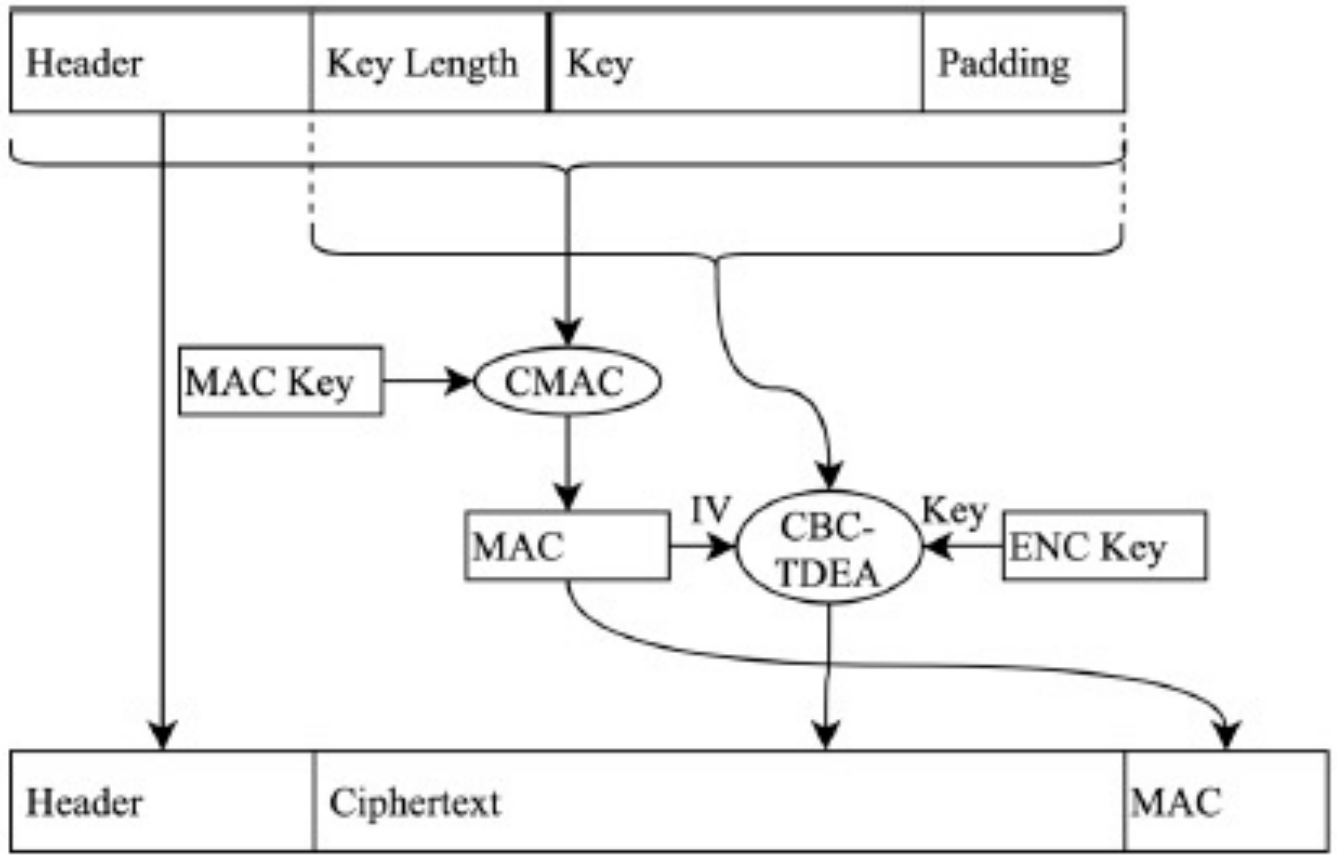


Security first…
TR-31 ‘B’/’D’ wrapping methods (TDES & AES)

‘MAC-then-encrypt’

- **One-way ENC & MAC key derivation**
  - NIST SP 800-108 CMAC based
- **Encryption CBC mode** is good
  - The IV is the MAC over the clear key, using the derived CMAC key. This is unpredictable!
- **Encryption Attack**: breaking TDES gets the derived wrapping key, *NOT* the KEK.
- **MAC attack**: the MAC is computed over plaintext, which is hidden in the encrypted key block

Figure 10 - Key Block Binding Method



TR-31/X9.143 pitfalls so far

PITFALL-1: Optional blocks had **no layouts** until 2021

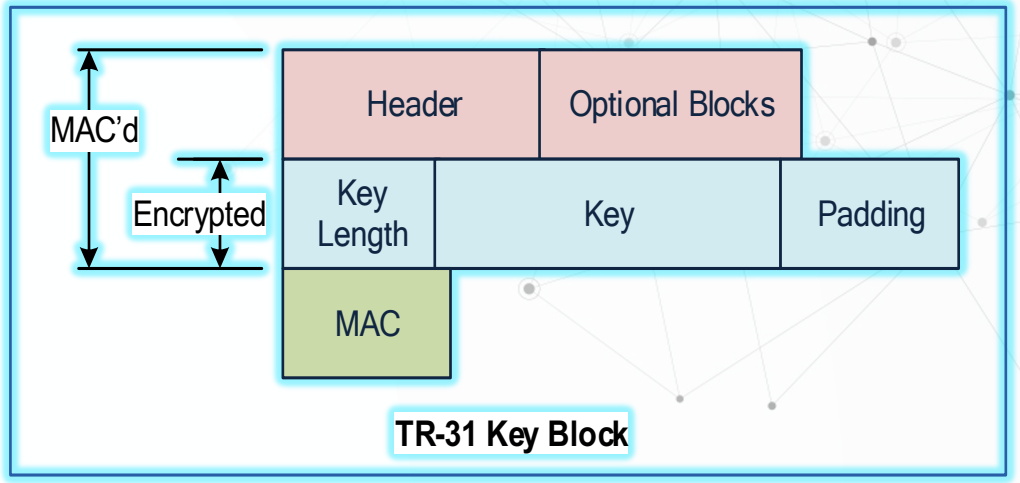
PITFALL-2: ISO 20038 uses a different HMAC hash identifier from TR-31-2018+

PITFALL-3: 'A'/'C' not compliant to 9/2020: FAQ (h) is violated by variant key calculation & NIST CBC break

PITFALL-4: 'Padding' payload NEVER required to hide key length…although it is allowed

PITFALL-5: TR-31 allowed numeric identifiers for proprietary use – even in the 1st character. *But this is your wrapping method!* Numeric 1st character = proprietary wrapping method ➔must be reviewed for equivalence

PITFALL-6: Single DES (Alg. 'D') supported. For KEKs.



ANSI X9.143 Key Block Header														
V	L	L	L	L	U	U	A	M	N	N	E	O	O	C	r
A B C D 1?	Length				Use		A D T E H R S	M O D E	Ver. Num .		E X P O R T	Opt Blk Cnt		C O N T E X T	R E S E R V E D

TR-31 biggest pitfall…is the *best reason* for X9.143

PITFALL-7: TR-31 is a *Technical Report*

- Interoperability only requires format and wrapping ‘agreement’
- Nothing in TR-31 is *enforceable*…everything is compliant! *Yay, we’re done*!
- **Questions** to ask a TR-31 provider:
  - **Real attributes**: Does the device enforce the Usage, Mode, Exportability…?
  - **’Proprietary TR-31**’: Are there numeric identifiers? If, yes, what is the security impact?
  - **Random pad**: Is the encrypted key padding actually random data?

X9.143 … *to the rescue!*

- Audit target with ‘SHALL’ statements
- Numeric header ID/wrapping methods ➔*not compliant*

TR-31 Key Block Header

V	L	L	L	L	U	U	A	M	N	N	E	O	O	C	r
A B C D 1?	Length				Use		ALGORITHM	MODE	Ver. Num.		EXPORT	Opt Blk Cnt		CONTEXT	RESERVED

X9.143 Key Management
The header…has context!

- **Byte 14** indicates ‘local’ or ‘transfer’ context ➔is the operational key under a transporter KEK or local HSM main wrapping key (MK/MFK)

- **Bytes 5&6**: Usages that separate more keys
  - **B3**: Key derivation key
  - **D2**: Data encryption key for decimalization table
  - **D3**: Encryption key for sensitive data
  - **E7**: EMV/Chip Asymmetric Key Pai
  - **F0-F6**: EMV Card Keys adde
  - **M8**: ISO 9797-1:2011 MAC Algorithm

ANSI X9.143 key block header

V	L	L	L	L	U	U	A	M	N	N	E	O	O	C	r
VERSION	Length of key block				USAGE		ALGORITHM	MODE	VERSION		EXPORT	Opt Blk Cnt		CONTEXT	RESERVED

- **X9.143 Key Management:**

  Optional Blocks become useful!

Header 0 - 15	V	L	L	L	L	U	U	A	M	N	N	E	O	O	C	r
Optional Blocks	**General Optional Blocks**															
	KC	CMAC Key Check Value of this key														
	KP	CMAC Key Check Value of the wrapping key (before derivation														
	LB	Label – your meaningful label can now be bound to the key!														
	PA	Proprietary Algorithm Identifier – instead of numeric ID														
	TC	"YYYYMMDDhhmmssZ" – time of creation for this **key**														
	TS	"YYYYMMDDhhmmssZ" - time of creation for this key **block**														
	Type-specific Optional Blocks															
	IK / KS	DUKPT Base Derivation Key identifier | Derivation identifier														
	DA	Derivation control:	U	U	A	M	E	U	U	A	M	E				
	HM	Usable hash: SHA-1, SHA-224, SHA-256, SHA-384, SHA-512														
Var.	**Encrypted Payload**															
Var.	**MAC**															

Takeaways

- **TR-31**
  - Has a long history
  - Not a standard
  - Has pitfalls and compliance challenges
- **X9.143**
  - Shiny, new!
  - Audit target with ‘SHALL’ statements
  - Local context added
  - New key types, attributes

TR-31 PITFALL REDUX

~~**PITFALL-1**: Optional blocks had **no layouts** until 2021~~…**have layouts**, may or may not match legacy TR-31!

PITFALL-2: ISO 20038 uses a different HMAC hash identifier from TR-31-2018+

~~**PITFALL-3**: ‘A’/’C’ not compliant to 9/2020: FAQ (h) is violated by variant & NIST break~~…**deprecated now**

PITFALL-4: ‘Padding’ payload NEVER required to hide key length…although it is allowed

~~**PITFALL-5:** TR-31 allowed numeric identifiers for proprietary use – even in the 1st character.~~ …**non-compliant now**

PITFALL-6: Single DES (Alg. ‘D’) supported. For KEKs.

~~**PITFALL-7**: Not a standard~~…**now it is!**

Linked in:
www.linkedin.com/in/kisley

Where I work:
https://www.ibm.com/security/cryptocards

Updates on IBM Crypto/HSMs:
https://community.ibm.com/community/user/ibmz-and-linuxone/groups/community-home/recent-community-blogs?CommunityKey=6593e27b-caf6-4f6c-a8a8-10b62a02509c