J ∫ Skripte Mo. 11. März

Kryptologie ICS.KRYPTO

Folien zur Präsenz 4

«Key Management für symmetrische Verfahren», FS 24, V2.2

Das Kap. 4.3 in JS Skripte ist vorgängig als HA zu repetieren (flipped classroom)!!

Tutorial zur Vorbereitung

https://hslu.zoom.us/rec/share/LjHvX0GQ0gagFVxJQFDFs_F0zV20yTk4dqH4ze872fPAD79j-tNlYquOOy9ToDCy.ne876fTxqZFjc2tG

©Josef Schuler, dipl. math., dipl. Ing. NDS ETHZ, MSc Applied IT-Security, Feldhof 25, 6300 Zug, j.schuler@bluewin.ch resp. josef.schuler@hslu.ch

Inhaltsübersicht

- Eine Einführung in die Elemente des Key Managements für symmetrische Verfahren
  - Bitte repetieren Sie vorgängig das Kap. 4.3 „Anzahl Schlüssel bei N Computern“ im JS Skript „Einführung in die Kryptologie“.
  - Wir besprechen das Kap. 9 „Key Management für symmetrische Verfahren“ im JS Skript „Einführung in die Kryptologie“. Tw. in der Nachbearbeitung zu vervollständigen!
  - Wir besprechen das Kap. 13 „Key Distribution“ im JS Skript „Einführung in die Kryptologie“. Tw. in der Nachbearbeitung zu vervollständigen!

Der Slogan zu dieser Präsenz

Es gibt keine Theorien zu Key Management, also machen wir eine!

Lernziele

- Ich kann die Elemente eines symmetrischen Key Managements aufzählen.
- Ich kann ein einfaches Key Management gemäss den vorgegebenen Eigenschaften beurteilen.
- Ich kann die Anzahl benötigter Schlüssel berechnen.
- Ich kann eine einfache Schlüsselverteilung nachvollziehen.

Verweise zur Literatur

- JS Skripte „Einführung in die Kryptologie“, Kap. 4.3 „Anzahl Schlüssel bei N Computern“ (vorgängig als HA repetiert), 9 & 13. Beide Kap. sind tw. als Nachbearbeitung zu vervollständigen.
- In [CP-D] das Kap. 13, ist ***nicht*** prüfungsrelevant!!
- Die Kapitelnummerierung in den folgenden Folien entspricht derjenigen im oben erwähnten JS Skript „Einführung in die Kryptologie“. D.h. die Details zu den Folien können im Skript nachgelesen werden. Zudem hat es im Skript weitere Übungen und Beispiele. ***Die Aufgaben- und Beispielnummerierung im JS Skript «Einführung in die Kryptologie» und in den vorliegenden Folien stimmen nicht immer überein!***
- ***Wichtig:*** Es ist unbedingt zu beachten, dass nur das Bearbeiten und Lernen der Folien nicht genügt. Das Durcharbeiten der oben erwähnten Kapitel in JS Skripte „Einführung in die Kryptologie“ sind absolut zentral zum Bestehen der Modulendprüfung.

EINFÜHRUNGSAUFGABE

Aufbau einer Maestro-Karte

- Auf dem Magnetstreifen hat es zwei Spuren:
  - Sie enthalten viele Daten, die auch auf der Karte ersichtlich sind.
  - Können mit Kartenlesegerät ohne weiteres gelesen werden.
  - Können (zumindest tw.) beschrieben werden.
- Chipkarte:
  - Enthält in einem öffentlich zugänglichen Bereich die gleichen Daten wie auf dem Magnetstreifen.
  - Diese Daten können mit Chipkartenleser problemlos gelesen werden.

Skimming Attacke: Maestro-Karte kopieren

- Mit Fake Kartenleser (Aufsatz an bestehenden Kartenleser) wird beim Einschieben der Karte, der Magnetstreifen (*) gelesen.
- Im Fake Kartenleser steckt eine beliebige Karte. Mit dem Einschieben der Kundenkarte wird die Fake Karte in den Bancomat geschoben.
- Der Prozess läuft und der Kunde gibt die PIN ein.
- Antwort: Pin falsch
- Egal, ob man nun abbricht, oder es dreimal probiert. Der Betrüger hat PIN und Kartendaten.
- Er übermittelt die Daten → ein Duplikat wird erstellt und damit betrogen so lange es funktioniert.
- (*) Angriff auf Chipkarte funktioniert im Prinzip gleich!

Skimming Attacke verhindern, aber wie?

Die unmittelbarste Lösungsidee ist schnell geboren:

Daten auf dem Magnetstreifen (resp. im File im Chip) verschlüsseln

Fragen:

1) Was würde das bedeuten?

2) Welchen Schutzmechanismus muss (und ist) mit Bestimmtheit für die Daten auf dem Magnetstreifen (resp. im File im Chip) implementiert?

Fazit: Einmal mehr Kennzahlen ist nicht die Lösung

Frage: Was ist der Zusammenhang mit Key Management?

→ siehe übernächste Folie "Bestimmen des Schwerpunkts"

Kap. 9

KEY MANAGEMENT FÜR SYMMETRISCHE VERFAHREN

Aufgaben des Key Management

- An dieser Stelle sei zunächst erwähnt, dass in den Lehrbüchern zur Kryptologie kaum je etwas über das Key Management – ausgenommen PKI – steht, obwohl das Key Management meistens mehr als 50% eines Projekts ausmacht.
- Eine Ausnahme bildet [CP-D] mit Kap. 13 → hat aber mit dem hier behandelten Inhalt nur wenige Berührungspunkte.
- Ein Key Management für symmetrische Verfahren beinhaltet folgende Aufgaben:
  - Erzeugung der Systemschlüssel = Masterkeys = Top-Level-Keys
  - Verteilung der System- und/oder Terminalschlüssel
  - Speicherung und Administration der Systemschlüssel
  - Berechnung der aktuellen Sessionkeys
  - „Transport“ der Sessionkeys.

Bemerkung: „Sessionkeys“ heissen in der dt. Literatur auch„kurzlebige“ oder „temporäre“ Schlüssel oder „Sitzungsschlüssel“. Auf englisch werden sie auch „epheremal“ Keys genannt. Wobei in gewissen Protokollen „epheremal“ Keys kurzlebige Schlüsselverschlüsselungsschlüssel (KEK = Key Encryption Key) sind.

Vorarbeit & Designkrit. eines Key Managements

Aspekte	Erklärung	Org/Tech
	Vorarbeit	
Bestimmen der Schutzziele:	Die Schutzziele müssen gemeinsam (d.h. mit der „Applikation“) diskutiert und bestimmt werden. M.a.W. der Kryptodesigner darf nicht alleine die Schutzziele definieren und bestimmen.	O.
Unique Key per Transaction (= Session):	Bei jeder Transaktion muss ein neuer Sessionkey verwendet werden (ev. mehrere neue).	T
Unique Key per Terminal:	Jedes Terminal muss individuelle Schlüssel haben.	T
Unique Key per Application (= Dienste):	Falls gleichzeitig mehrere Dienste beansprucht werden (z.B. Verschlüsseln und MAC-Berechnung, müssen verschiedene Sessionkeys verwendet werden.	T
Mehrere Generationen von Schlüsseln:	Ein Key Management muss so konzipiert sein, dass ein Wechsel von Masterschlüsseln (= Masterkeys = Top-Level-Keys) „in vernünftiger“ Zeit möglich ist.	O/T
Schlüssel dürfen nur in einem Sicherheitsmodul in klar sein:	Schlüssel dürfen nur in Sicherheitsmoduln (z.B. Chipkarte) unverschlüsselt sein. Das impliziert, dass Kryptographie in einem PC in der Regel suboptimal ist.	T
Im realen Betrieb dürfen keine Testschlüssel mehr vorhanden sein:	Die während der Testphase verwendeten – und meistens einfache Schlüssel wie 1111222233334444 – dürfen im realen Betrieb nicht mehr eingesetzt werden.	O
Trennung der Realschlüssel auf mehrere Personen:	Die Aufbewahrung der Realschlüssel muss auf drei Personen aufgeteilt werden.	O
Verfahrenskenner sind keine Schlüsselträger, resp. -kenner:	Die Designer und Kenner der Verfahren dürfen keine Realschlüssel kennen oder aufbewahren. bis 16.4.18	O

Aufgabe 1 = Aufgabe 9.4 im JS Skript

Betrachten Sie ein fiktives Zahlungssystem, das dem aktuellen Bezahlsystem mit der Maestrokarte ähnelt. Im Gegensatz zum „richtigen“ Bezahlsystem sind einige kryptographische Designkriterien im betrachteten fiktiven System nicht erfüllt. Das fiktive System solle folgende Eigenschaften haben:

a) Das Terminal besitzt kein Sicherheitsmodul
b) Jedes Terminal besitzt einen individuellen Schlüssel.
c) Das Key Management ist so ausgelegt, dass ein Schlüsselwechsel jederzeit möglich ist.
d) Bei jedem Meldungsaustausch (Zahlungstransaktion) wird ein neuer Zufallsschlüssel erzeugt. Dieser wird mit dem individuellen Terminalschlüssel verschlüsselt.
e) Die PIN-Verschlüsselung, MAC-Berechnung und die Meldungsverschlüsselung wird mit dem gleichen Zufallsschlüssel (siehe d)) gemacht.
f) Weil das System in der Testphase problemlos lief, hat man sich entschieden alle Masterkeys beizubehalten, dies im Sinne „never change a winning team“.

bis 16h30

Aufgabe 1 = Aufgabe 9.4, Tabelle

Welche Aspekte (genauer: eine Vorarbeit und 8 Designkriterien) sind erfüllt?
Bitte ***kreuzen*** Sie die Spalte **erfüllt, nicht erfüllt** oder **kann nicht beurteilt werden** an.
Und geben Sie **die Nummer der Eigenschaft** an, aus der Sie die Eigenschaft herausgelesen haben.

Paul 16h45

Aspekte	Erfüllt? Nr?	N. erfüllt? Nr?	Kann n. beurteilt werden!
Bestimmen der Schutzziele:			
Unique Key per Transaction (= Session):			
Unique Key per Terminal:			
Unique Key per Application (= Dienste):			
Mehrere Generationen von Schlüsseln:			
Schlüssel dürfen nur in einem Sicherheitsmodul in klar sein:			
Im realen Betrieb dürfen keine Testschlüssel mehr vorhanden sein:			
Trennung der Realschlüssel auf mehrere Personen:			
Verfahrenskenner sind keine Schlüsselträger, resp. -kenner:			

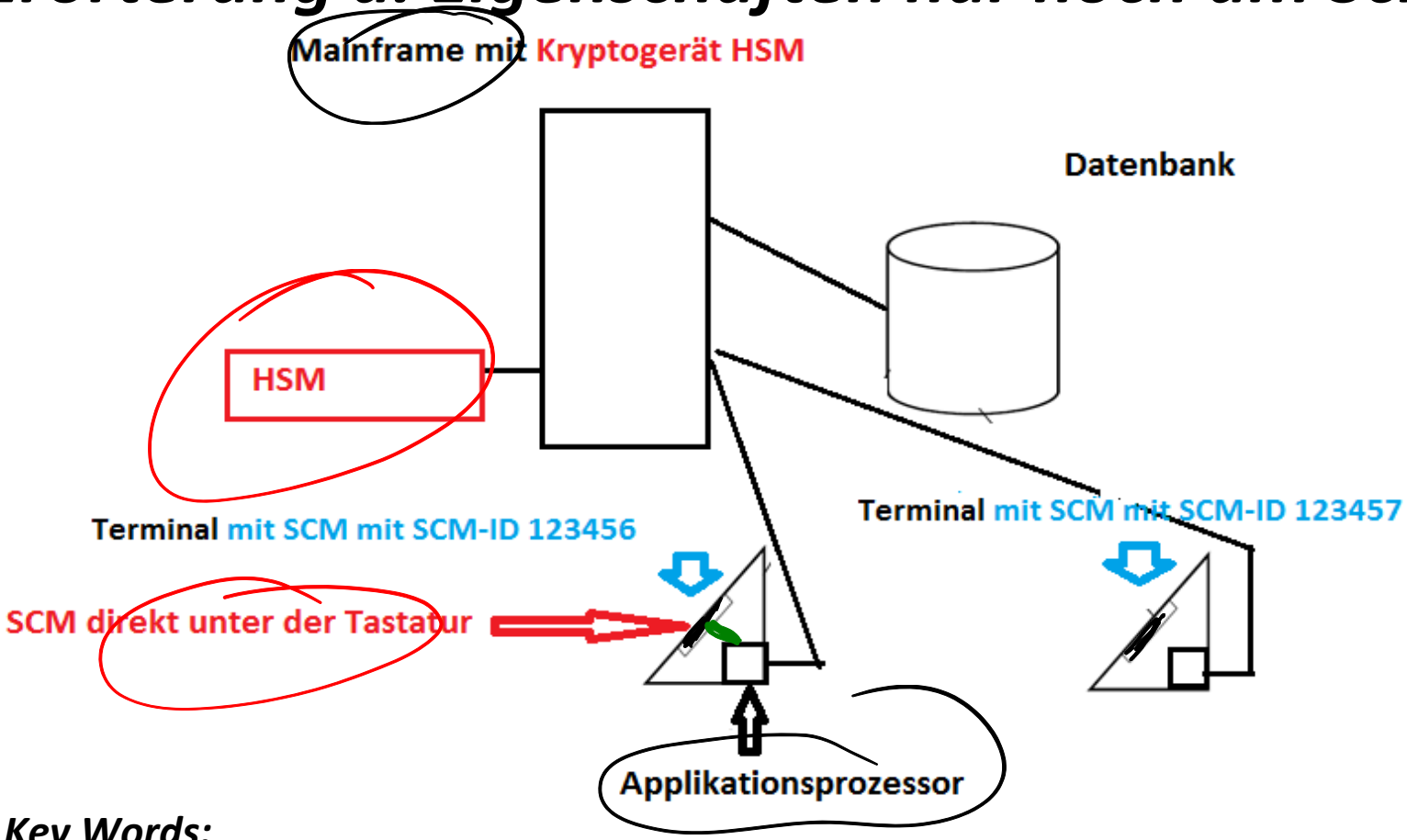
Beispiel 9.2: Bancomat und Maestro Bezahlgerät sowie Host mit Kryptogerät

HSM = High Security Module

Host (Mainframe oder Serverfarm) = Applikation

Tastatur mit integriertem Sicherheitsmodul (SCM) und eingebrannter SCM-ID. Das SCM befindet sich direkt unter Tastatur. Der ganze Block wird dann vergossen!!

Erörterung d. Eigenschaften nur noch am Schema



Key Words:

- Das SCM wird in einer sicheren Umgebung hergestellt.
- Beim Herstellungsprozess werden die SCM-ID & individuelle Keys „eingebrannt“.
- Das SCM ist direkt unter Tastatur → das Ganze ist ein Monolith.
- Es gibt eine Schnittstelle nach aussen; der Befehlssatz ist eingeschränkt, z.B. Herauslesen der Schlüssel ist nicht möglich.

Ausgangslage und die zentrale Frage

Ausgangslage für die weiteren Betrachtungen:

Wir gehen nun davon aus, dass …

- … in jedem SCM individuelle Ladeschlüssel enthalten sind. Diese werden „nur“ gebraucht, um die Terminalkeys zu laden → im Weiteren nicht mehr von Bedeutung.
- … in jedem SCM eigene (fixe) Terminalkeys enthalten sind.
- … für jeden der fünf Dienste (Verschlüsselung der PIN, Meldungsverschlüsselung & MAC-Berechnung vom Terminal zum Host und die Meldungsverschlüsselung & MAC-Berechnung vom Host zum Terminal) je eigene (z.T. je mehrere) Schlüssel verwendet werden.
- … bei jeder Transaktion werden immer neue (und zwar für alle Dienste) Schlüssel generiert und verwendet.

Folgerung:

Aufmerksame Studierende fragen sich nun, ob die PIN doppelt verschlüsselt – einmal mit der PIN-Verschlüsselung und mit der Meldungsverschlüsselung – wird oder „nur“ einmal mit der PIN-Verschlüsselung → das ist sicherheitstechnisch nicht von Belang, sondern einfach ein Systemdesign.

Wegen Unique Key per Terminal sind diese pro SCM-ID verschieden.

Die zentrale Frage: Braucht es die Datenbank um die Keys zu holen?

Antwort: Nein!

Übersicht der Prozesse und Varianten

Wir müssen nun zwei total unterschiedliche Prozesse unterscheiden:

I. Das (erstmalige) Erzeugen der Terminalkeys, die dann ins Terminal geladen werden und dort fix gespeichert sind. Und dann der jeweilige Erhalt der Terminalkeys der Hostseite im HSM. Dazu werden wir zwei Varianten betrachten:
   A) ***Mit*** Datenbankzugriff
   B) ***Ohne*** Datenbankzugriff

II. Das Erzeugen der Sessionkeys sowohl der Seite des Terminals wie auf der Seite des HSM's. Wiederum zwei Varianten.
   C) Die Sessionschlüssel werden ***verschlüsselt mitgeschickt.***
   D) Die Sessionschlüssel werden ***auf jeder Seite erzeugt, daher nicht mitgeschickt.***

Ein Resultat wird sein:

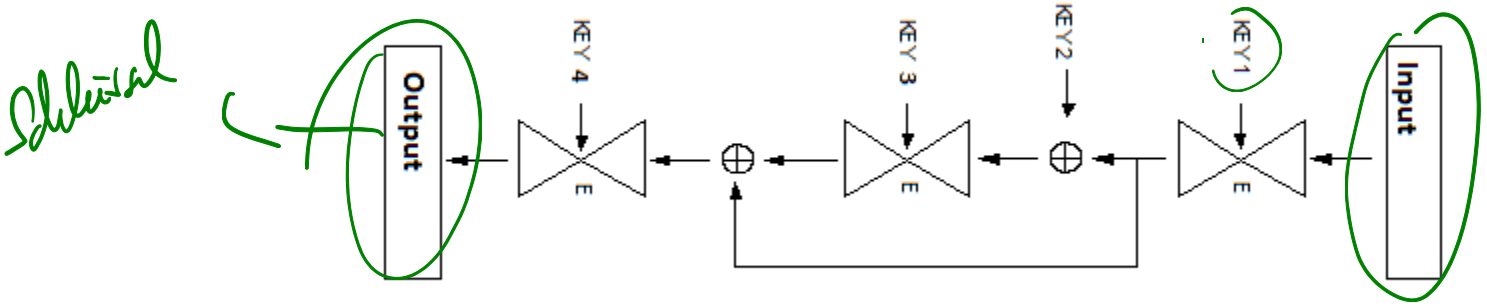
Die zwei Variantengruppen A) & B) einerseits und C) & D) andererseits sind kombinierbar, im Konkreten ist es möglich …

- A) & C)
- A) & D)
- B) & C)
- B) & D) … zu realisieren.

Variante B) ohne Datenbankzugriffe, Vorb.

Im Folgenden verwenden wir die Randomfunktion aus Kap. 8.4.1.

- Der Input und die Keys 1, …, 4 werden dann situativ definiert.
- Der Output ist dann jeweils ein erzeugter Schlüssel.
- Die „Sanduhr“ repräsentiert wiederum eine Blockchiffre, z.B. AES-256.
- In der nachfolgenden Folie werden wir dann 4 Terminalkeys erzeugen. Die Anzahl „4“ kommt aus der Tatsache, dass wir beim Erzeugen der Sessionkeys Variante D) wiederum die unten aufgeführte Randomfunktion mit 4 Keys verwenden werden.
- Eine solche Randomfunktion mit Blockchiffren ist eine Einwegfunktion (warum?) und wird oft auch KDF (Key Derivation Function, also Schlüsselableitungsfunktion) genannt.

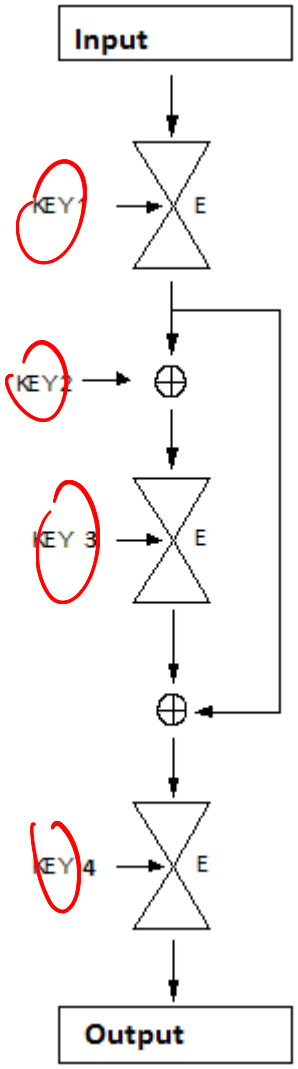


Bemerkung:

Die Variante A) ***mit*** Datenbankzugriff ist im JS Skript „Einführung in die Kryptologie“ in Kap. 9.2.2 beschrieben.

Aufgabe 2a) Bearbeiten Sie in der Nachbearbeitung die Variante A) ***mit*** Datenbankzugriff.

Var. B) Erzeugen der 4 Terminalkeys TM 1, …, 4



Für TM x (x = 1, …, 4):

Input = SCM-ID || x || Weitere Daten

- „||“ = Konkatenation oder Aneinanderlegen
- Key 1 = Masterkey 1.x
- Key 2 = Masterkey 2.x
- Key 3 = Masterkey 3.x
- Key 4 = Masterkey 4.x

1.1 … 1.4
2.1 … 2.4
4.1 … 4.4

Beispiel:

- Für TM 1 (x = 1) braucht es die Masterkeys 1.1, 2.1, 3.1 & 4.1.
- Für TM 2 (x = 2) braucht es die Masterkeys 1.2, 2.2, 3.2 & 4.2.
- Usw.

Es braucht also total 4 · 4 = 16 Masterkeys im HSM

Für TM 4 (x = 4)

Alternative: Variante A) mit Datenbankzugriff:

Die 4 Terminalmasterkey je zufällig erzeugen und in der DB verschlüsselt speichern. Sie werden bei jeder Transaktion aus der DB geholt, Nachteile:

- Datenbankzugriff
- Verwaltung
- Usw.

Erzeugen der Sessionkeys, Variante C)

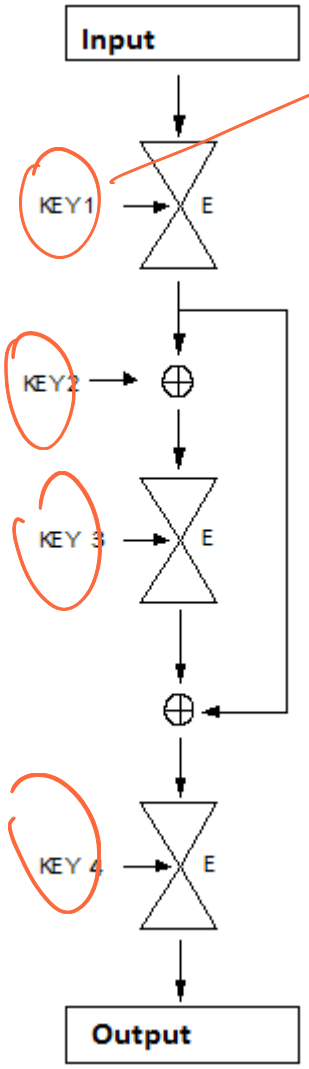
Diese Variante können wir mit den folg. Punkten zusammenfassen:

- Unique Key per Session und per Application kann in beide Richtungen durch Erzeugen von zufälligen Schlüsseln erreicht werden.
- Diese zufälligen Schlüssel müssen dann jeweils mit den Terminalkeys verschlüsselt werden.
- Selbstredend heisst das, dass das empfangende Gerät (HSM oder Terminal) jeweils diese verschlüsselten Sessionkeys entschlüsseln muss.

Aufgabe 2b)
Bearbeiten Sie in der Nachbearbeitung die Variante C) ***die zufällig erzeugten Sessionkeys werden verschlüsselt mitgeschickt***.
Lesen Sie dann bitte …

- … die zusätzlichen Ausführungen zur sog. „Unicity Distance“ von Shannon.
- … die Analyse dieser Variante C).

Erzeugen der versch. Sessionkeys, Var. D)



Key Words:

- AES-256 im ECB-Modus für die PIN-Verschlüsselung.
- AES-256 im CBC-Modus für die Meldungs-Verschlüsselung.
- MAC Variante von ISO 9797-1 mit AES-256 für die MAC-Berechnung.
- Im Wesentlichen findet im SCM wie im HSM der gleiche Prozess statt.
- Weitere Details im JS Skript.

Wir gehen von 7 (d.h. x = 1, …, 7) ev. mehr Diensten aus:

Für Sessionkey x: Input = SCM-ID || x || Seq.Nr || Weitere Daten

- x = 1 = PIN-Verschlüsselung
- x = 2 = Meldungs-Verschlüsselung Terminal – HSM
- x = 3 = K für MAC-Berechnung Terminal – HSM
- x = 4 = K' für MAC-Berechnung Terminal – HSM
- x = 5 = Meldungs-Verschlüsselung HSM–Terminal
- x = 6 = K für MAC-Berechnung HSM – Terminal
- x = 7 = K' für MAC-Berechnung HSM – Terminal
- x = 8
- x = 9 usw. Reservé, falls weitere Dienste benötigt würden

Key 1 = Terminalkey 1, Key 2 = Terminalkey 2, usw.

Haben wir an alles gedacht?

Aufgabe 3 Füllen Sie die Tabelle aus, und machen Sie ein Fazit.

Aspekte	Erfüllt? Nr?	N. erfüllt? Nr?	Kann n. beurteilt werden!
Bestimmen der Schutzziele:	(X)		(X)
Unique Key per Transaction (= Session):	X		
Unique Key per Terminal:	X		
Unique Key per Application (= Dienste):	X		
Mehrere Generationen von Schlüsseln:			X
Schlüssel dürfen nur in einem Sicherheitsmodul in klar sein:	X		
Im realen Betrieb dürfen keine Testschlüssel mehr vorhanden sein:			X
Trennung der Realschlüssel auf mehrere Personen:			X
Verfahrenskenner sind keine Schlüsselträger, resp. -kenner:			X

Fazit: Die rein techn. Aspekte sind offensichtlich berücksichtigt
Die org. Aspekte können wir nicht beurteilen.

Die fehlenden Punkte; Details cf. JS Skript

- Wie kommen die Terminalmasterkeys ins SCM?
  - → wurde mit dem Ladegerät schon kurz angesprochen.
- Wie realisiert man, einen Schlüsselwechsel?
  - System grounden und von Hand jedes SCM neu laden?
  - Ev. anders?
- Keine Testschlüssel im Realsystem?
  - Das liegt in der Verantwortung des Kryptoexperten!
- Trennung von (Real-)schlüsselträger und Verfahrenskenner?
  - Muss in der Organisation im Detail besprochen werden.
- Auftrennung der Realschlüssel auf mehrere Personen?
- Administration der 4*4 = 16 Masterkeys im HSM, resp. Wie kann bei einem Totalkollaps wieder gestartet werden?
  - Lösungsansatz für alle 3 Forderungen zusammengefasst: Masterkeys in Papierform in Banksafe (ggf. aus Redundanzgründen bei zwei versch. Banken) aufbewahren. Zugriff nur mit zwei Personen.

Beispiel 9.1
Erzeugung und Transport des Sessionkeys, Teil 1

Voraussetzungen:

- ➔ Wir haben eine Punkt zu Punkt Verbindung, also kein Netzwerk (z.B. eine Host zu Host Verbindung).
- ➔ Es wird nur eine Verschlüsselung gemacht.
- ➔ Die Sessionkeyerzeugung wird mit der Randomfunktion aus Kap. 8.4 gemacht (evtl. mit 4 Schlüsseln).
- ➔ ~~Key 1 und 2 (resp.~~ Key 1 – 4) der verwendeten Randomfunktion sind vorgängig auf sicherem Weg schon ausgetauscht worden.
- ➔ Um bei jeder Schlüsselerzeugung einen anderen Input zu haben, verwenden wird den Vorschlag aus Kap. 8.4.1: **Vorschlag:** Input = (Datum || Zeit) ⊕ Output einer „Rauschdiode“.

Erzeugung und Transport des Sessionkeys, Teil 2

Variante 1

Sender	Empfänger
1. Erzeugt (siehe Vorschlag von oben) einen Zufallsbitstring. 2. Dieser Input wird als Input für die Randomfunktion (cf. Kap. 8.4.1) gebraucht. Der Output ist der zu verwendende Schlüssel. 3. Verschlüsselt die Nachricht mit dem oben erzeugten Schlüssel. 4. Schickt die verschlüsselte Nachricht und den (unverschlüsselten) Zufallsbitstring zum Empfänger.	1. Berechnet mit dem mitgeschickten, unverschlüsselten Zufallsbitstring und der Randomfunktion aus Kap. 8.4.1 den Schlüssel. 2. Entschlüsselt die Nachricht mit dem berechneten Schlüssel.

Erzeugung und Transport des Sessionkeys, Teil 3

Bemerkungen resp. weitere Varianten:

1) Eine **Variante 2** wäre, den Sessionkey symmetrisch verschlüsselt mit einem fixen Masterkey mitzuschicken.
2) Ein solcher wie in Variante 2 erwähnte fixe Masterkey heisst „Key-Encryption-Key“ KEK oder auf Deutsch „Schlüsselverschlüsselungsschlüssel“.
3) Eine **Variante 3** wäre, den Sessionkey asymmetrisch verschlüsselt mitzuschicken, dies wäre ein sog. ***Hybrides Verfahren.***

Frage:

Was ist jetzt der Unterschied zwischen der Variante, wo der PIN-Block immer mit dem gleichen Schlüssel verschlüsselt wird und der Variante, wo der PIN-Block mit einem Zufallsschlüssel verschlüsselt wird, aber der Zufallsschlüssel immer mit dem gleichen Schlüssel verschlüsselt wird?

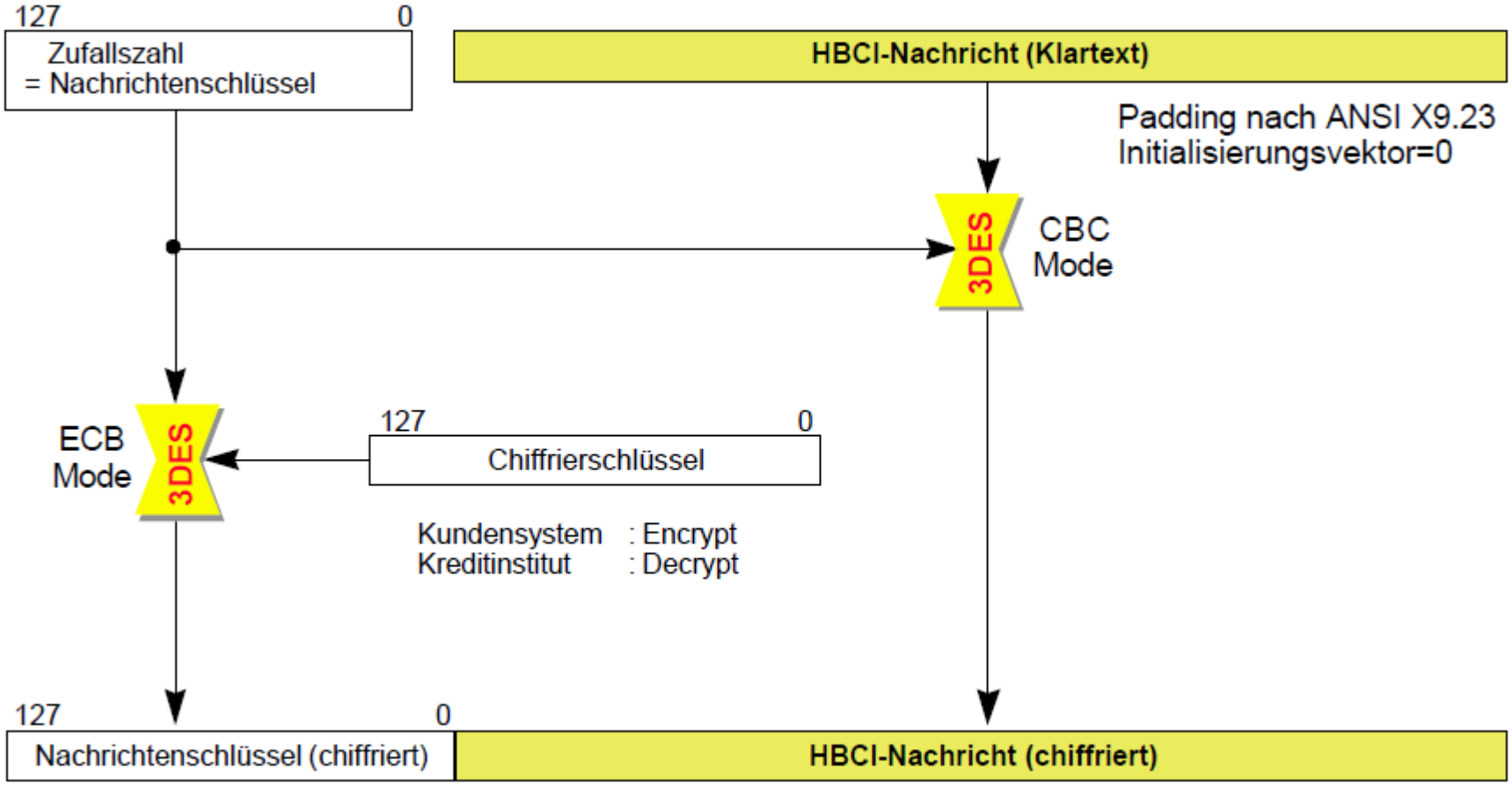
Antwort:

Ja, der Unterschied ist tatsächlich frappant. Im Thema „Kryptoanalyse“ gehen wir u.a. genauer auf diese Frage ein.

bei 17435

Aufgabe 4 = Aufgabe 9.3a) im JS Skript

a) Welche Variante wird mit diesem Diagramm gezeigt?



Aufgabe 4 = Aufgabe 9.3b) – f) im JS Skript

b) Kreuzen Sie nun alle Angriffe an, gegen die die „Nachricht“ geschützt ist.

- ☐ Verändern der Meldung (Integrity)
- ☐ Abstreiten die Meldung erhalten zu haben (Non repudiation of receipt)
- ☐ Abhören der Meldung (Confidentiality)
- ☐ Eine Meldung abfangen und später wieder einspielen (Replay)
- ☐ Löschen von Meldungen (Delete)Eine erfundene Meldung einspielen (Insertion)
- ☐ Sich für jemanden anders ausgeben (Masquerade)
- ☐ Abstreiten die Meldung geschickt zu haben (Non repudiation of origin)

c) Was müssen Sender und Empfänger notwendigerweise auf einem anderen Wege vorher ausgetauscht haben?

d) Geben Sie die Reihenfolge der Schritte an, die der Sender und dann der Empfänger durchführen müssen.

e) Erstellen Sie eine saubere Skizze, in der klar ersichtlich ist, wie der Empfänger zum korrekten Schlüssel kommt, um die Nachricht entschlüsseln zu können.

f) Gibt es noch andere Möglichkeiten des 3DES? Was müsste dann in der Zeichnung geändert werden?

Aufgabe 9.3g) ohne Musterlösung

Im Diagramm der vorherigen Folie steckt, neben der eigentlichen Aufgabe, viel Potenzial, um Themen der Kryptologie zu repetieren.

Was kann man aussagen oder welche Fragen kann man sich stellen? z.B.

- Wie wird genau verschlüsselt?
- Was muss vorher schon ausgetauscht werden?
- Beschreiben Sie den genauen Ablauf?
- Was ist ANSI X9.23?
- Usw. usw.

Macht euch eigene Gedanken, diskutiert untereinander darüber. Macht z.B. einen Eintrag im Forum.
Es gibt dazu keine Musterlösung, aber ich bin gerne bereit zugeschickte Lösungsvorschläge zu kommentieren.

„Schlusswort“ zu Key Management

- ***Mit dem Key Management kann z.B. auch Systeme trennen, aber auch gezielt und z.B. zeitlich limitiert verbinden → cf. Kap. 9.3 im JS Skript, wo ein Polizeifunkbeispiel ganz kurz beschrieben ist.***

- ***Das Key Management …***
  - … ist oft unsichtbar und wird vor allem immer unterschätzt.
  - … kann man nicht ab Stange kaufen.
  - … ist für jedes Projekt individuell.
  - … kann man nicht im Hörsaal lernen wie zu designen, das ***entsprechende Denken und die Sensibilisierung*** aber schon.

- ***Es bleibt schlussendlich nur Wenigen vorbehalten einmal ein Key Management von Grund auf zu definieren, …***
  - ***… bei einigen von Ihnen ist die Chance aber durchaus vorhanden.***

- ***Das Polizeifunkbeispiel könnte z.B. als WiPro dienen …***
  - … mit zentraler Instanz oder ohne (cf. Präsenz 12)
  - … sym. oder asym. (z.B. Kerberos oder PKI)
  - … inkl. Authentisierung und anderen Aspekten wie Forward Security usw.

Weitere Aufgaben (Nachbearbeitung)

- ***Aufgabe 5***
  - Visualisieren Sie sich im Beispiel 9.1 die verschiedenen Varianten (z.B. durch Zeichnen des Ablaufs).
  - Sie dürfen sich auch eigene Varianten überlegen, formulieren, ins Forum hochladen und/oder mit dem Dozenten besprechen.

- ***Aufgabe 6***
  - Lösen Sie die restlichen Aufgaben in „Einführung in die Kryptologie“, Kap. 9.

Kap. 13

KEY DISTRIBUTION

Kap. 13.2.1 Ein klassisches Kuriersystem

Austausch eines Anfangsgeheimnisses zw. Rechenzentrum RZ_1 und RZ_2:

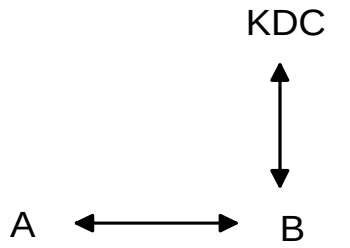
RZ_1	Kurier	RZ_2
Erzeugt 3 Schlüsselteile T_1, T_2 & T_3		
	T_1 in verschlossenem Couvert ---------->	
		Unterschreibt, das Couvert in unbeschädigtem Zustand erhalten zu haben.
	Bringt die Bestätigung zurück <----------	
Dieser Ablauf wird nun dreimal gemacht.		
Nun erzeugt RZ_1 den Verschlüsselungsschlüssel $V = T_1 \oplus T_2 \oplus T_3$. Und verschlüsselt den Masterkey mit V.		
	Der mit V verschlüsselte Masterkey wird nun übertragen. ---------->	
		RZ_2 berechnet auch den Verschlüsselungsschlüssel $V = T_1 \oplus T_2 \oplus T_3$. Und entschlüsselt mit V den verschlüsselten Masterkey.

Es gibt natürlich Spielvarianten, wo z.B. RZ_2 auch einen Teil erzeugen kann.

Aufgabe 7 Cf. JS Skript, Kap. 13.2.1. Bearbeiten Sie die dort aufgeführten Spielvarianten.

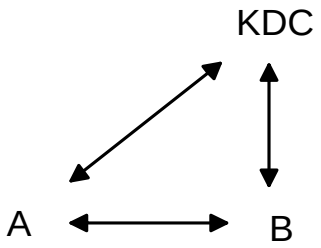
Weitere Schlüsselverteilsysteme

- ***Kap. 13.3 Diffie-Hellman Schlüsselaustausch***
  - Das DH-Schlüsselaustauschverfahren cf. asym. Kryptographie (Präsenz 7 resp. 8).
- ***Kap. 13.4 PKI***
  - Cf. Präsenz 11, Prof. A. Portmann wird PKI mit eigenen Unterlagen unterrichten.
- ***Kap. 13.5 Schlüsselverteilung mittels KDC (Key Destribution Center), (#) → im Skript nur rudimentär → nicht Prüfungsstoff → cf. [CP-D], Kap. 13.***
  - Jeder Teilnehmer teilt dem Key Distribution Center KDC (auf einem sicheren Weg, z.B. wie wir ihn vorhin beschrieben haben) einen Grundschlüssel mit.
  - Die Schlüsselverteilungsprotokolle basieren im Prinzip auf 3 Meldungspaaren, um die 3 Partner gegeneinander zu authentisieren (Dreiecks Authentisierung)
  - Session Keys werden mit Hilfe von „aufgeblähten“ 2PAP Protokollen ausgetauscht.
  - 3 Modelle: **Pull Model** **Push-Modell (Kerberos)** **Mixed Modell**



B holt für A die Keys im KDC

A holt für B die Keys beim KDC



Beide holen die Keys beim KDC

Basis-Test Präsenz 4

Aussage	Richtig oder falsch?	Begründung
Ein Key Management kann man ab Stange kaufen.	☐ richtig ☐ falsch	
Die Lehrbücher sind voll von Informationen zu Key Management.	☐ richtig ☐ falsch	
Das Key Management wird oft unterschätzt, es macht i.d.R. 50% und mehr eines Projektes aus.	☐ richtig ☐ falsch	
Die Grundaufgaben eines Key Managements sind in diesem Skript aufgeführt.	☐ richtig ☐ falsch	
Ein Key Management wird aufgrund von rein technischen Kriterien erstellt.	☐ richtig ☐ falsch	
In diesem Skript sind 8 Designkriterien zum Key Management aufgeführt.	☐ richtig ☐ falsch	
In diesem Skript sind keine konkreten Beispiele zum Key Management aufgeführt.	☐ richtig ☐ falsch	
PKI ist (u.a.) ein Schlüsselverteilsystem.	☐ richtig ☐ falsch	

bis Nr 44

LÖSUNGEN DER AUFGABEN

Aufgabe 1 = Aufgabe 9.4

Aspekte	Erfüllt? Nr?	N. erfüllt? Nr?	Kann n. beurteilt werden!
Bestimmen der Schutzziele (*):	*Ev. X e)*		*Ev. X*
Unique Key per Transaction (= Session):	*X d)*		
Unique Key per Terminal:	*X b)*		
Unique Key per Application (= Dienste):		*X e)*	
Mehrere Generationen von Schlüsseln:	*X c)*		
Schlüssel dürfen nur in einem Sicherheitsmodul in klar sein:		*X a)*	
Im realen Betrieb dürfen keine Testschlüssel mehr vorhanden sein:		*X f)*	
Trennung der Realschlüssel auf mehrere Personen:			*X*
Verfahrenskenner sind keine Schlüsselträger, resp. -kenner:		*X (**)*	

(*) Ein Hinweis, dass man sich bez. der Schutzziele Gedanken gemacht hat, ist, weil man sich offensichtlich bewusst ist, dass es mehrere Dienste gibt. Man hat dann (offensichtlich) entschieden, für alle Dienste jeweils den gleichen Schlüssel zu nehmen.
OK, die Aufgabe ist von mir so kreiert worden. Damit will ich u.a. auch zeigen, dass es auch sein kann, dass man auf Grund der Angaben nicht definitiv entscheiden kann, d.h. in der realen Arbeitswelt müsste man nachfragen.
(**) Wegen f) sind die Schlüssel dem Verfahrenskenner bekannt.

Aufgabe 2a) & b) Keine Musterlösung, da es ein Nachbearbeitungssauftrag ist.

Aufgabe 3

Aspekte	Erfüllt? Nr?	N. erfüllt? Nr?	Kann n. beurteilt werden!
Bestimmen der Schutzziele (*):	*(X)*		*(X)*
Unique Key per Transaction (= Session):	*X*		
Unique Key per Terminal:	*X*		
Unique Key per Application (= Dienste):	*X*		
Mehrere Generationen von Schlüsseln:			*X*
Schlüssel dürfen nur in einem Sicherheitsmodul in klar sein:	*X*		
Im realen Betrieb dürfen keine Testschlüssel mehr vorhanden sein:			*X*
Trennung der Realschlüssel auf mehrere Personen:			*X*
Verfahrenskenner sind keine Schlüsselträger, resp. -kenner:			*X*

Fazit:

- Bez. der reinen technischen Aspekte haben wir – offenbar – an alles gedacht.
- Bez. den organisatorischen (und ggf. dazugehörenden org.) Aspekten können wir es nicht beurteilen.
- «Bestimmen der Schutzziele», hier können wir annehmen, dass es einen gewissen Diskussionsprozess gab, aber sicher sind wir nicht; darum 2-mal (X).

Aufgabe 4 = Aufgabe 9.3) im JS Skript

a) Variante 2

b) ☑ Abhören der Meldung (Confidentiality)

c) Den Chiffrierschlüssel.

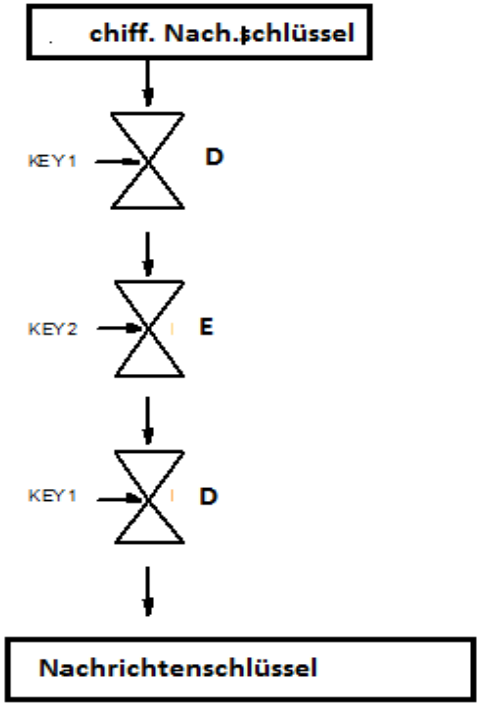
d)

i. Der Chiffrierschlüssel muss vorgängig ausgetauscht sein (siehe b)).
ii. Der Nachrichtenschlüssel muss mit Zufallsgenerator erzeugt werden.
iii. Die Nachricht wird mit diesem Nachrichtenschlüssel verschlüsselt.
iv. Der Nachrichtenschlüssel wird mit dem Chiffrierschlüssel verschlüsselt.
v. Der Empfänger entschlüsselt den chiffrierten Nachrichtenschlüssel.
vi. Mit dem Nachrichtenschlüssel entschlüsselt er die Nachricht.

Die Schritte iii) & iv) können auch vertauscht werden.

e) Rechts die Zeichnung für e)

Bemerkung: Da der Chiffrierschlüssel 128 Bit gross ist (siehe Diagramm in der Aufgabenstellung) ist es offensichtlich, dass ein 3-DES mit 2 Schlüsseln verwendet wird.



f) Ja, 3-DES mit 3 Schlüsseln, die Zufallszahl und der chiffrierte Nachrichtenschlüssel müssten von 0 bis 191 nummeriert sein.

Aufgabe 5 – 7

Keine Musterlösungen, da es Nachbearbeitungsaufträge sind.

Basis-Test Präsenz 4

Aussage	Richtig oder falsch?	Begründung
Ein Key Management kann man ab Stange kaufen.	☐ richtig ☑ falsch	
Die Lehrbücher sind voll von Informationen zu Key Management.	☐ richtig ☑ falsch	
Das Key Management wird oft unterschätzt, es macht i.d.R. 50% und mehr eines Projektes aus.	☑ richtig ☐ falsch	
Die Grundaufgaben eines Key Managements sind in diesem Skript aufgeführt.	☑ richtig ☐ falsch	
Ein Key Management wird aufgrund von rein technischen Kriterien erstellt.	☐ richtig ☑ falsch	
In diesem Skript sind 8 Designkriterien zum Key Management aufgeführt.	☑ richtig ☐ falsch	Das Bestimmen der Schutzziele ist eine (notwendige) Voraussetzung, resp. Vorarbeit und diesem Sinne kein Designkriterium, daher nur 8 Designkriterien.
In diesem Skript sind keine konkreten Beispiele zum Key Management aufgeführt.	☐ richtig ☑ falsch	
PKI ist (u.a.) ein Schlüsselverteilsystem.	☑ richtig ☐ falsch	