[redacted]

[redacted]

# Untersuchung zum Schutz der Software

## Arbeitspaket 7

[redacted]

[redacted]

| | |
|---|---|
| Version: | 1.0 |
| Datum: | 23. Dezember 2010 |

## Historie

| Version | Autor | Kommentar | Datum |
|---|---|---|---|
| 0.1 | [redacted] | Initiale Version erstellt<br>Review: [redacted] | 12.11.10 |
| 0.2 | [redacted] | – Kapitel 3 - Angriffsszenarien ausgefüllt<br>– Kapitel 4 - Angriffsbäume aus AP3 eliminiert<br>– Kapitel 4 – weiterer PBA-Schutz via Smartphone<br>– Kapitel 5 beschrieben<br>– Review [redacted] | 17.11.10 |
| 0.3 | [redacted] | – Kapitel 2.5 Smartcards eingefügt | 24.11.10 |
| 0.4 | [redacted] | – Einarbeitung BSI Review | 25.11.10 |
| 0.5 | [redacted] | – Review Kapitel 2.5<br>– Einarbeitung internes Review<br>– Neues Bild zu den unterschiedlichen Möglichkeiten beim Systemstart<br>– Finalisierung des Dokuments | 10.12.10 |
| 0.6 | [redacted] | – Einarbeitung BSI-Kommentare | 21.12.10 |
| 1.0 | [redacted] | – Einarbeitung letzter Kommentare<br>– Version 1.0 zur Abnahme | 23.12.10 |

# Inhaltsverzeichnis


1 Problembeschreibung....5
1.1 Bedrohte Werte....6
1.2 Authentisierung....7
1.2.1 Benutzer gegenüber Plattform....7
1.2.2 Plattform gegenüber Benutzer....8
2 Begriffsdefinitionen....10
2.1 Trusted Computing....10
2.1.1 Trusted Computing Group....10
2.1.2 TPM Work Group....10
2.1.3 Server Work Group....10
2.1.4 Mobile Phone Work Group....10
2.1.5 Infrastructure Work Group....10
2.1.6 Trusted Network Connect Work Group....11
2.1.7 Storage Work Group....11
2.2 Bootstrapping....12
2.2.1 Trusted Boot (Authenticated Boot)....12
2.2.2 Secure Boot....12
2.3 Trusted Platform Module....13
2.3.1 Platform Configuration Register....13
2.3.2 Roots Of Trust....14
Root Of Trust For Measurement....14
Core Root Of Trust For Measurement (CRTM / SRTM)....14
Dynamic Root Of Trust For Measurement (DRTM)....15
Root Of Trust For Reporting....15
Root Of Trust For Storage....16
2.3.3 TPM-Shielded Keys....16
Endorsement Key....17
Storage Root Key....17
2.3.4 Nicht-flüchtiger Speicher (Non-Volatile RAM)....18
2.3.5 Sealing....18
2.3.6 TPM Vertrauensbeglaubigung....19
Endorsement Credential....19
Conformance Credential....19
Platform Credential....19
2.4 Chain of Trust....20
2.5 Smartcard....22
2.5.1 Grundlagen....22
2.5.2 Leser....22
2.5.3 Kommunikation....23
2.5.4 PKCS #11 ....23
2.5.5 Einsatz....23
2.5.6 RFID....24
3 Existierende Angriffe....25
3.1 Passwort per Brute-Force angreifen....25

3.2 Passwort per Keylogger erhalten....25
3.3 Passwort aus der Auslagerungsdatei auslesen....26
3.4 Passwort aus dem Hauptspeicher auslesen....26
3.4.1 via DMA und FireWire....26
3.4.2 via Tastasturpuffer des BIOS....27
3.4.3 via Coldboot-Mechanismen....27
3.5 Passwort durch Manipulation am Bootloader protokollieren....28
3.5.1 Evil-Maid Angriff auf TrueCrypt....28
3.5.2 Reproduktion des Bootloaders (optisch) am Beispiel von Bitlocker....28
3.5.3 Stoned Bootkit....29
4 Lösungsmöglichkeiten....30
4.1 Lösungsansätze ohne Trusted Computing....35
4.1.1 M1.1 - Booten von externem Medium....35
4.1.2 M1.2 - Booten vom Netzwerk....37
4.1.3 M1.3 - Messung des Speichers mittels DMA....39
4.2 Online-Lösungen mit Trusted Computing....41
4.2.1 M2.1 - Remote Attestation mit Schlüsselversand....41
4.3 Offline-Lösungen mit Trusted Computing....44
4.3.1 M3.1 - Sealing des Volume-Key....44
4.3.2 M3.2 - Ablage des Volume-Schlüssels in NVRAM des TPM....46
4.3.3 M3.3 - Lokale „Remote“ Attestation via Smartcard....48
4.3.4 M3.4 - Gegenseitige Attestierung über ein „Trusted Device“, z.B. via Smartphone....51
4.3.5 M3.5 - Gegenseitige Attestierung via Smartcard und Klasse-3-Leser (i). .54
4.3.6 M3.6 - Gegenseitige Attestierung via Smartcard und Klasse-3-Leser (ii). 57
4.3.7 M3.7 – Mehrstufige, gegenseitige Attestierung durch Bilder....60
4.3.8 M3.8 - Sealing des Volume-Key + Smartcard und Klasse-1-Leser....62
4.4 Secure Boot ohne Trusted Computing....65
4.4.1 M4.1 - Sicheres BIOS....65
4.5 Secure Boot mit Trusted Computing....68
4.5.1 M5.1 - Unter Verwendung des DRTM und Intel TXT / AMD SVM....68
5 Zusammenfassung....70
5.1 Produkte und Dienstleistungen fürs Geschäftsmodell....71
5.2 Empfehlung....71
5.3 Vergleich – Bedrohungen, Maßnahmen und Szenarien....73

# 1 Problembeschreibung

Heutige Computersysteme besitzen keine einfach umzusetzende Möglichkeit, die auf der Plattform laufende Software sicher zu starten. Rein funktionell ist der Startprozess – ab dem Einschaltzeitpunkt des PC bis zum Erreichen des geladenen Betriebssystems – wohl definiert und bereitet technisch keine Probleme, jedoch sind die Bootkomponenten bzw. die zu startenden Programme im Allgemeinen nicht vor Manipulation geschützt. Einem Angreifer ist es daher prinzipiell erst einmal möglich, von einem beliebigen Bootmedium zu starten, den Bootloader, das zu ladende Betriebssystem oder installierte Anwendungen auf einer Festplatte auszulesen, zu manipulieren oder ganz zu ersetzen. Ein Benutzer bzw. Administrator hat zunächst keine Möglichkeit, diese gewollten (z.B. durch einen Angreifer) oder ungewollten (z.B. durch Hardware-Fehler) Modifikationen zu bemerken.

Eine gängige, erste Schutzmaßnahme stellt das Setzen eines Start- sowie eines BIOS-Passworts dar. Das Start-Passwort kann genutzt werden, um den PC vor unberechtigten Startvorgängen zu schützen, zum Hochfahren des Systems muss zwingend dieses Passwort eingegeben werden. Über das BIOS-Passwort lässt sich der Zugang zum sowie Modifikationen am BIOS nur durch gültige Benutzer (d.h. Inhaber des BIOS-Passworts) durchführen, wie z.B. die Änderung der Bootreihenfolge. Als letzte Variante bieten die ATA-Spezifikationen[1] ein Sicherheitskonzept namens „ATA security mode feature set“ vor. Hier lässt sich der Zugriff auf die Festplatte durch ein Passwort schützen. In der Regel kann ein Benutzer im BIOS das Setzen des Passworts veranlassen. Vor einem Startvorgang muss die Festplatte dann zunächst freigeschaltet werden, bevor von ihr Daten gelesen werden können. Bei einem Startvorgang würde das Passwort daher durch das BIOS abgefragt werden.

Diese Passwort-basierten Lösungsansätze bieten jedoch keinen ausreichenden Schutz, vor allem nicht im Hinblick auf Vertraulichkeit:

- BIOS-Passwörter von (zumindest) PC-Mainboards können durch das Setzen von Steckbrücken (*Jumper*) zurückgesetzt werden. Zusätzlich existieren für einige BIOS-Versionen Standard-Passwörter, die Angreifern ebenfalls Zugriff auf das BIOS gewähren. Alternativ kann auch das BIOS bzw. der Flash-Speicher, in dem das BIOS gespeichert ist, getauscht werden.
- Start-Passwörter werden im gleichen Bereich gespeichert wie BIOS-Passwörter, sodass auch hier die gleichen Risiken und Umgehungsmöglichkeiten gelten.
- Das Setzen von BIOS- oder Startpassworten schützt die Festplatte nicht, wenn ein Angreifer diese aus dem System entnehmen und in ein anderes System einbauen kann.
- Bzgl. der Festplattenpasswörter gibt es zwar keine Möglichkeit, diese physikalisch zurückzusetzen, jedoch existieren für Festplattenhersteller

1 ATA-3 Spezifikation, Kapitel 6.5 „Security mode feature set“, siehe http://www.t10.org/t13/project/d2008r7b-ATA-3.pdf

Möglichkeiten zum Zurücksetzen. Im übrigen schützt das Passwort lediglich den Zugriff auf die Daten, die Daten selbst liegen immer noch im Klartext auf dem Datenträger vor, sodass z.B. unter Laborbedingungen die einzelnen Speicherscheiben ausgelesen werden können. Alternativ lässt sich auch ein neuer, ungeschützter Festplattencontroller an die Festplatte montieren, um so Zugang zu den Daten zu erhalten.

Um die auf dem System befindliche Software und Daten zu schützen, empfiehlt sich der Einsatz einer Festplattenverschlüsselung. Diese kann entweder durch den Einsatz einer Hardwarelösung realisiert werden, wie z.B. Festplatten mit direkt eingebauter Verschlüsselung[2] oder aber durch Software-basierte Verschlüsselungslösungen. Bei letzterem Fall gibt es verschiedene Ansätze, eine Festplatte zu verschlüsseln:

- Verschlüsseln einer einzelnen (Daten-)Partition,
- Verschlüsseln einer (Daten-)partition sowie der Systempartition,
- Verschlüsseln der gesamten Festplatte (*Full-Disc-Encryption* (FDE)).

Der letzte Fall ist zugleich der komplexeste Fall: In diesem Szenario wird eine gesamte Festplatte inkl. Betriebssystem, Anwendungen und Nutzerdaten verschlüsselt. Das problematische beim Einsatz einer Software-basierten FDE ist jedoch, dass ohne Entschlüsselungsroutine und den dazugehörigen Schlüssel nicht von der Festplatte gelesen werden kann. Für ein konventionelles BIOS, welches von Haus aus keinerlei Informationen über Algorithmus oder Betriebsart mitbringt, ist das Lesen und somit das Starten von einer FDE-verschlüsselten Festplatte nicht möglich.

Eine gängige Lösung für dieses Problem besteht darin, nicht die gesamte Festplatte (d.h. angefangen von Sektor 0 bis einschließlich des letzten Sektors) zu verschlüsseln, sondern einen kleinen Bereich am Anfang der Festplatte im Klartext zu belassen. In diesem Bereich – dem Master Boot Record (Sektoren 0-62) – werden die Partitionstabelle sowie ein Bootloader eingebracht. Der Bootloader enthält die nötigen Entschlüsselungsroutinen, um auf den Rest der Festplatte zugreifen zu können und so das Betriebssystem zu starten.

Üblicherweise fragt hierzu der Bootloader den Benutzer nach seinem Passwort, welches dann als Eingabe für eine „Password-Based Key Derivation Function“ (PBKDF) dient. Als Ergebnis dieser Funktion wird ein Schlüssel ausgegeben, welcher dann zur Entschlüsselung genutzt wird.

## *1.1 Bedrohte Werte*

Sämtliche bedrohten und damit zu schützenden Werte sind in AP3 definiert. Um aber konkret den Gefahren von Angriffen (siehe Kapitel 3) auf den Startvorgang (sog. Pre-Boot-Angriffe) zu begegnen, werden hier erneut die relevanten Werte aufgeführt, die

2 http://www.fujitsu.com/global/news/pr/archives/month/2008/20080421-01.html
http://www.seagate.com/ww/v/index.jsp?locale=de-DE&name=Seagate_Secure_Technology:_Selbstverschl%C3%BCsselnde_Laptop-Festplatten_von_National_Security_Agency_qualifiziert&vgnextoid=9942aa95bf92a110VgnVCM100000f5ee0a0aRCRD

vor und während des Startvorgangs besonders geschützt sein müssen. Kapitel 3 beschreibt existierende Angriffe gegen TrueCrypt und andere, gängige Festplattenverschlüsselungslösungen mit FDE.

**Direkt bedrohte Werte**

- Daten (abhängig vom Volume-Key)
  - Betriebssystem
  - Anwendungen
  - Nutzerdaten (NfD)
- Volume-Key

**Indirekt bedrohte Werte**

- Hardware
  - Tastatur (durch Keylogger)
  - TPM (durch Hardware-Angriffe)
  - Smartcard-Leser (durch Manipulation)
  - RAM (durch Auslesen)
- Software (durch Manipulation)
  - Master Boot Record (MBR)
  - Bootloader
  - BIOS Firmware
- Authentifizierungsmerkmale (durch Ausspähen / Auslesen)
  - Passwort (siehe z.B. 3.5.1 oder 3.5.2)
  - PIN
  - Smartcard

## *1.2 Authentisierung*

Eines der größten Probleme hinsichtlich der FDE besteht darin, vor dem eigentlichen Startvorgang eine „Vertrauensbeziehung“ zwischen der zu startenden Plattform und dem davor sitzenden Benutzer herzustellen.

### 1.2.1 Benutzer gegenüber Plattform

Um die verschlüsselte Plattform zu starten, wird das Authentifizierungsmerkmal des Benutzers zur Ableitung des Volume-Keys benötigt. Dies kann indirekt als Authentisierung des Benutzers gegenüber der Plattform gewertet werden, denn die Plattform offenbart (implizit) nur ihre Geheimnisse, wenn die gültigen Credentials bereit gestellt wurden. Als Authentifizierungsmerkmal kommen drei Arten in Frage,

welche zusätzlich untereinander kombiniert werden können:

- (*was man weiß*)
  - Komplexes Passwort (mit hoher Entropie)
  - PIN, z.B. in Verbindung mit einer Smartcard
- *(was man hat)*
  - Smartcard
  - Ausweis
    - Neuer Personalausweis
    - Truppenausweis der Bundeswehr
    - elektronischer Dienstausweis
- (*was man kann*)
  - Biometrie, z.B. Fingerabdruck
  - Eingabeverhalten, z.B. Verzögerung zwischen jedem Tastendruck

### 1.2.2 Plattform gegenüber Benutzer

Die Authentifizierung des Benutzers stellt in der Regel kein Problem dar. Gibt ein Benutzer das falsche Passwort ein oder stellt nicht die benötigten Authentifizierungsmerkmale bereit, so lässt sich der Volume-Key nicht rekonstruieren und ein Zugriff auf die verschlüsselte Festplatte ist nicht möglich. Die entgegengesetzte Richtung, d.h. die Authentifizierung der Plattform gegenüber dem Benutzer ist hier schon komplizierter. Dieser Authentisierung ist insofern ein hoher Stellengrad beizumessen, da ein Benutzer sicher sein muss, wem er seine Credentials präsentiert. Gelingt es z.B. einem Angreifer, dem Benutzer ein modifiziertes System „unterzuschieben“, und gibt der Benutzer sein Passwort in das modifizierte System ein, so kann der Angreifer an die benötigten Informationen gelangen, um sich Zugriff auf die Daten zu verschaffen.

Diese Problematik ist insofern als besonders schwerwiegend zu bewerten, da ein Angreifer diese Informationen an vielen Stellen während des Bootprozesses abgreifen kann, z.B.:

- mit Hilfe eines Keyloggers zwischen Tastatur und PC (hier helfen nur organisatorische Maßnahmen)
- durch ein modifiziertes BIOS, welches Tastatureingaben speichert
- durch einen modifizierten Bootloader, welcher Tastatureingaben speichert

Ein vollständiger Angriffsbaum ist in AP3 zu finden, eine Auflistung bereits existierender Angriffe ist in Kapitel 3 dargestellt.

Im Rahmen dieses Arbeitspaketes können Angriffe, wie das [optische / akustische / via Keylogger] Auslesen von Benutzereingaben oder sonstige Hardware-

Modifikationen nicht betrachtet werden. Angriffen dieser Art muss durch organisatorische Maßnahmen (z.B. Siegel, Räumlichkeiten, Mitarbeiterschulung etc.) vorgebeugt werden. Die hier beschriebenen Lösungen zielen darauf ab, Software-basierte Pre-Boot-Angriffe zu:

a) erkennen

b) verhindern

Dazu müssen Lösungen erarbeitet werden, welche eine Vertrauenskette (*chain of trust*) durch den gesamten Startvorgang herstellen und zu dem Zeitpunkt, an welchem der Benutzer seine Credentials vorzeigt, diesen (explizit oder implizit) von der Korrektheit seiner Konfiguration überzeugt. Konkret bedeutet dies:

- Der Benutzer hat seine unmodifizierte[3] Hardware vor sich.
- Die Integrität des BIOS ist gewährleistet.
- Die Integrität des MBR ist gewährleistet.
- Die Integrität des Bootloaders ist gewährleistet.

Im Folgenden werden zunächst einige grundlegende Begriffe und Technologien definiert und erläutert, welche technologisch zur Lösung der Pre-Boot-Problematik beitragen können. Darüber hinaus werden existierende Angriffe und Lösungsmöglichkeiten für TrueCrypt skizziert. Im Anschluss erfolgt eine Empfehlung, welche der dargestellten Lösungsmöglichkeiten geeignet für den Einsatz mit TrueCrypt sind.

3 Mit unmodifizierter Hardware ist gemeint, dass der Benutzer die selbe Hardware mit der selben Start-Software vor sich hat. Unter Software fällt hier insbesondere auch die Firmware von Hardware-Komponenten, welche unmittelbar am Startvorgang beteiligt ist, d.h. die Codeteile eines ROMs, welche vor dem Starten **ausgeführt** werden müssen, beispielsweise Grafik-BIOS, SCSI-BIOS, Raid-Controller-BIOS etc.

# 2 Begriffsdefinitionen

## *2.1 Trusted Computing*

In diesem Zusammenhang wird der Begriff Trusted Computing in Anlehnung an die Definition durch die TCG verwendet, d. h. es werden Komponenten und Mechanismen beschrieben, die in der Spezifikation der TCG definiert oder zumindest dazu kompatibel sind und dazu beitragen, die Vertrauenswürdigkeit einer Plattform zu erhöhen.

### 2.1.1 Trusted Computing Group

Die *Trusted Computing Group* (TCG)[4] ist ein internationales Konsortium aus verschiedenen Industrie- und Forschungseinrichtungen, mit dem Ziel, offene Standards und Spezifikationen im Bereich *Trusted Computing* zu entwickeln. Die TCG wurde im Jahre 2003 von AMD, Hewlett-Packard, IBM, Intel und Microsoft gegründet – damals noch unter der Bezeichnung *Trusted Computing Platform Alliance* (TCPA). Innerhalb der TCG existieren verschiedene Arbeitsgruppen, die sich mit den folgenden Themenbereichen beschäftigen:

### 2.1.2 TPM Work Group

Die TPM Arbeitsgruppe spezifiziert die Kernkomponente im Bereich des *Trusted Computing* – das *Trusted Platform Module* (TPM) (siehe Kapitel 2.3). Hierzu hat die TCG verschiedene Spezifikationen erstellt, welche die Funktionalität, die Schnittstellen und die vom TPM zur Verfügung gestellte Funktionalität inkl. der zugehörigen Kommandos definiert.

### 2.1.3 Server Work Group

Die Aufgabe der *Server Work Group* ist es, Definitionen, Spezifikationen, Anleitungen und technische Voraussetzungen bereitzustellen, die die Implementierungen von TCG-Technik in Servern betreffen.

### 2.1.4 Mobile Phone Work Group

Die *Mobile Phone Work Group* arbeitet an der Anpassung des TCG-Konzeptes für mobile Geräte. Die Arbeitsgruppe erweitert die Spezifikationen der TCG um Spezifikationen, die spezielle Eigenschaften von mobilen Endgeräten, wie z.B. eingeschränkte Ressourcen, wechselnde Netzanbindung oder auch spezielle Nutzungsmodelle berücksichtigen.

### 2.1.5 Infrastructure Work Group

Die *Infrastructure Work Group* arbeitet an der Erweiterung und Integration der TCG-

4 http://www.trustedcomputinggroup.org

Spezifikationen für die Bereiche Internet- und Intranet-Infrastrukturen, um unterschiedliche Geschäftsmodelle in offenen heterogenen Plattformumgebungen zu ermöglichen. Die Arbeitsgruppe legt Rahmenbedingungen, Schnittstellen und Bedingungen fest, um Probleme in der Infrastruktur zu überwinden und überlegt sich u.a. Lösungen für:

- Repräsentationen von Vertrauensankern
- Vertrauensketten
- Schlüsselverwaltungen (engl. *Key Lifecycle Services*)
- Sicherheitsrichtlinien

### 2.1.6 Trusted Network Connect Work Group

Die *Trusted Network Connect* (TNC) Arbeitsgruppe hat eine offene Architektur zur Integritätsüberprüfung via Rechnernetze spezifiziert und veröffentlicht. Die TNC-Architektur ermöglicht es Administratoren aktiver Netzkomponenten, Sicherheitsrichtlinien während oder nach Herstellung einer Netzverbindung durchzusetzen.

### 2.1.7 Storage Work Group

Die *Storage Work Group* arbeitet basierend auf existierenden TCG-Techniken und -Philosophien und widmet sich der Entwicklung von Spezifikationen für die Sicherheit von Massenspeichersystemen. Ein Ziel ist es, Spezifikationen zu erstellen, so dass technikunabhängige Sicherheitsdienste oberhalb eines Festplattenadapters definiert werden können, welche z. B. ATA, Serial ATA, SCSI, FibreChannel, USB-Sticks, IEEE 1394, Network Attached Storage (NAS) oder Wechsel-Medien umfassen, aber nicht auf diese beschränkt sind. Diese Arbeitsgruppe der TCG arbeitet mit anderen Industrieorganisationen zur Spezifizierung von Massenspeichern zusammen, um den Einflussbereich der TCG-Technik zu vergrößern.

## 2.2 Bootstrapping

Die Bezeichnung *bootstrapping* wird oft durch das gebräuchlichere Wort *booting* (oder *booten*) abgekürzt. Das Booten ist der notwendige Vorgang, um die grundlegende Hardware einer Plattform zu initialisieren (normalerweise wird das vom BIOS übernommen) sowie das Starten des Betriebssystems durch den Bootloader.

### 2.2.1 Trusted Boot (Authenticated Boot)

*Trusted Boot* bezeichnet die Sicherheitseigenschaft einer Bootstrap-Architektur gemäß des Bootstrap-Modells der TCG. Das bedeutet, dass alle am Bootvorgang beteiligten Komponenten vor der Ausführung derart gemessen werden, dass **entfernte** Stellen sie verifizieren können. Die Messergebnisse werden an geeigneter Stelle sicher protokolliert. Der Startvorgang selbst wird nicht beeinflusst. Nachdem die Startsequenz beendet ist, können diese Protokolle zur Überprüfung des Systemzustandes verwendet werden. Durch die Analyse der Protokolle können nun andere Entitäten über die Vertrauenswürdigkeit eines Computersystems entscheiden. Zu beachten ist hierbei, dass *Trusted Boot* in keiner Weise garantiert, dass sich das Gerät in einem sicheren Zustand befindet, denn der Bootvorgang wird nicht unterbrochen, selbst wenn Komponenten modifiziert wurden. *Trusted Boot* wird auch als *Authenticated Boot* bezeichnet.

### 2.2.2 Secure Boot

*Secure Boot* prüft ein auszuführendes Programm noch bevor es gestartet wird anhand von Sicherheitsrichtlinien, und vermeidet so, dass unerlaubte Software auf einem Gerät gestartet wird. Die Sicherheitsrichtlinien umfassen ein Identifizierungsmerkmal von autorisierter Software, wobei z.B. Prüfsummen als Identifikationsmerkmal verwendet werden können. Liegt kein passendes Identifikationsmerkmal vor, wird der Startvorgang abgebrochen. Gewöhnlich wird diese Technik dazu verwendet, um **lokal** abzusichern, dass nur gültige Software auf einer Plattform gestartet wird und diese sich so in einem sicheren Zustand befindet. Die Technik des Secure Boot kann dazu verwendet werden, um sichere geschlossene Plattformen aufzubauen, die nur eine beschränkte Menge an ausführbarer Software booten.

## 2.3 Trusted Platform Module

Die heutzutage weit verbreitetste Version eines TPM ist ein kleiner, manipulationssicherer (*tamper-resistant*) Chip, welcher fest in eine PC-Plattform integriert ist, z.B. durch Aufstecken oder Auflöten auf das Mainboard oder durch Integration in andere PC-Komponenten, wie z.B. die Northbridge oder die CPU.
Der Vorteil beim Auflöten / Aufstecken eines TPM liegt darin, dass man die freie Wahl des TPM-Hersteller hat. Ein großer Nachteil ist jedoch, dass durch Hardware-Manipulation das TPM zur Laufzeit zurückgesetzt oder ggf. die Kommunikation mit anderen PC-Komponenten (z.B. LPC-Bus) mitgehört werden kann. Das direkte Einbringen eines TPM in z.B. die Northbridge hat den Vorteil, dass das TPM weder getauscht, noch abgehört oder manipuliert werden kann, jedoch muss man bei dieser Lösung dem Chip-Hersteller vollends vertrauen. Für die in Kapitel 4 vorgestellten Lösungen sind auch aufgesteckte bzw. aufgelötete TPMs ausreichend, sofern durch organisatorische Maßnahmen wie Siegel sichergestellt ist, dass der PC nicht geöffnet wurde.
Das TPM ist in der Lage, kryptographische Schlüssel in einer isolierten Umgebung zu erzeugen, zu verwenden und zu speichern. Darüber hinaus besitzt ein TPM neben einer Kryptographieeinheit (z.B. RSA und SHA-1) zum Signieren und Verschlüsseln, einen Zufallszahlengenerator sowie eine begrenzte Menge an sicherem Speicher.
Heutzutage werden TPMs von vielen unterschiedlichen Herstellern entwickelt, u.a. von Atmel, Broadcom, Infineon, Intel, Nuvotun, Sinosun, ST-Microelectronics und Winbond (zuvor National Semiconductor).

Im Folgenden werden die wesentlichen Eigenschaften des TPM erläutert:

### 2.3.1 Platform Configuration Register

Jedes TPM besitzt internen, flüchtigen Speicher, um gemessene Werte einer Plattform zu speichern. Hierzu sieht ein TPM sogenannte Platform Configuration Register (PCR) vor. Diese speichern lediglich diskrete Werte in Form von SHA-1-Ergebnissen. In der ersten TPM-Spezifikation 1.1b wurde eine Mindest-Anzahl von 16 PCRs gefordert, die aktuelle Version 1.2 schreibt bereits 24 Konfigurationsregister vor. Jedes PCR stellt 20 Byte Speicher (160 Bit) zur Verfügung. Dieser Speicher befindet sich innerhalb des TPM-Chips in einer isolierten und somit geschützten Umgebung, sodass diese nicht von außen manipuliert oder gelöscht werden können. Ziel der PCRs ist es, eine Integritätsmessung (z.B. der Plattform oder der darauf laufenden Software) sicher zu speichern, ohne dass diese Messwerte ersetzt, manipuliert oder gelöscht werden können. Sie werden dazu verwendet, einer externen Partei die gemessene Konfiguration zu beweisen.
Dadurch, dass der Speicherplatz in jedem PCR auf die Größe eines SHA-1 Hashwertes begrenzt ist, werden Folgemessungen in einer Hash-Kette abgespeichert. Es ist also nicht möglich, direkt einen Hashwert in ein PCR zu schreiben, sondern der neue Wert eines PCR wird berechnet durch:

$$PCR_{i,Neu} = SHA\text{-}1(PCR_{i,Alt} \,||\, SHA\text{-}1(Messobjekt))$$

### 2.3.2 Roots Of Trust

Um eine reguläre Computerplattform in eine vertrauenswürdige Computerplattform (d.h. *Trusted Computing Platform*) zu überführen, ist es notwendig, den aktuellen Systemzustand zu messen und zu protokollieren. Darüber hinaus muss es möglich sein, diese Messergebnisse einer externen Partei zugänglich zu machen und dieser so den aktuellen Systemstatus zu attestieren. Um diese Ziele zu erfüllen, müssen neue Komponenten in die existierende Plattform hinzugefügt werden:

1. Root of Trust for Measurement (RTM)
2. Root of Trust for Reporting (RTR)
3. Root of Trust for Storage (RTS)

Diese Komponenten müssen implizit vertrauenswürdig sein, denn sie bilden die Wurzel einer Vertrauenskette zur Messung, Speicherung und Übermittlung der Messwerte. Daher ist es notwendig, diese Komponenten auch manipulationssicher in die Plattform zu integrieren. Bei einem TPM geschieht dies u.a. durch das Auflöten bzw. Einbringen des Chip in das Mainboard. Hierbei ist anzumerken, dass ein Chip, welcher lediglich auf ein Mainboard aufgelötet ist, diesen Anforderungen nicht genügt, denn für einen Angreifer, der Zugriff auf die PINs des Chip hat, ist es möglich, Inhalte ggf. mitzuschneiden, zu verändern oder den Chip im laufenden Betrieb zurückzusetzen. Daher hat sich z.B. der Chip-Hersteller Intel dazu entschieden, das TPM direkt in seinen Northbridge-Chipsatz (bei ICH9) zu integrieren.
Um den *Roots of Trust* zu vertrauen, muss ein Benutzer implizit den entsprechenden TPM-Herstellern, BIOS-Herstellern und Plattform-Integratoren vertrauen. Um das Vertrauen in die Plattform zu erhöhen, kann ein Benutzer die mitgelieferten Zertifikate (siehe Kapitel 2.3.6) überprüfen und dann entscheiden, ob er der ausstellenden Instanz vertraut.

#### *Root Of Trust For Measurement*

Die Vertrauenswurzel für Messungen muss bedingungslos als vertrauenswürdig eingestuft werden, denn sämtliche Beurteilungen über den Vertrauenszustand einer Plattform hängen maßgeblich von den Ergebnissen dieser Messungen ab. Laut der TCG Spezifikation existieren zwei unterschiedliche RTMs, ein statisches RTM und ein dynamisches RTM. Diese werden im Folgenden erläutert:

#### *Core Root Of Trust For Measurement (CRTM / SRTM)*

Das statische RTM (SRTM – Begriff aus TPM Spezifikation 1.2) oder auch das Kern-RTM (CRTM - Begriff aus TPM Spezifikation 1.1b) ist eine unveränderbare, ausführbare Komponente, welche die Kontrolle über den Host unmittelbar nach einem Plattform-Reset übernimmt. Das CRTM ist für die Messung einer jeden Plattform-Transition verantwortlich, d.h. das CRTM muss jeglichen Code vor der Ausführung messen. Die Ergebnisse werden dann an Root Of Trust For Reporting / Root Of Trust For Storage weitergeleitet. Da die Vertrauenswürdigkeit der gesamten Plattform unmittelbar von dieser Komponente abhängt, empfiehlt die TCG, das CRTM in einem sicheren Bereich auf der Plattform unterzubringen. Unmittelbar nach

dem Zurücksetzen oder Starten einer Plattform misst das CRTM zunächst das BIOS und überträgt dann die Kontrolle. Das BIOS ist dann für die weiteren Messungen mit Hilfe des CRTM verantwortlich. Der Ablauf eines Bootprozesses mit CRTM wird in Kapitel 2.4 dargestellt. Die Hauptaufgabe des CRTM liegt darin, eine statische, ununterbrochene Vertrauenskette vom Einschalten der Plattform bis zum Betriebssystem herzustellen. Die ermittelten Messwerte werden im TPM-Chip manipulationsgeschützt gespeichert.

Um zu ermöglichen, dass das CRTM die erste Komponente ist, welche beim Starten ausgeführt wird, existieren 2 Möglichkeiten der Plattform-Integration:

a) CRTM ist im BIOS-Boot Block: In diesem Szenario wird das BIOS in zwei Bereiche aufgeteilt: ein BIOS-Boot Block (das eigentliche CRTM) sowie das POST BIOS
b) Das CRTM ist das gesamte BIOS, bzw. umgekehrt: Das BIOS übernimmt die vollständige Funktion des CRTM.

Letztere Variante hat den Nachteil, dass bei einem BIOS-Update das CRTM mit aktualisiert wird. Bei Variante a) ist es möglich, das POST BIOS alleine zu ändern.

### *Dynamic Root Of Trust For Measurement (DRTM)*

Das dynamische RTM (DRTM – erst verfügbar bei TPM 1.2) ist ebenfalls ein unveränderbarer Teil der Plattform. Im Gegensatz zum CRTM ist es beim DRTM jedoch nicht nötig, unmittelbar beim Start involviert zu sein. Das DRTM ermöglicht es, zu einem beliebigen Zeitpunkt die Plattform mit Hilfe eines sicheren Bootloaders in einen sicheren Modus zu bringen. Diese Transition verwendet neue Techniken wie z.B. Intels TXT-Erweiterung. Bei einem Wechsel in den sicheren Modus werden sämtliche Anwendungen gestoppt und IRQs abgeschaltet. Der sichere Bootloader startet dann eine neue (dynamische) Vertrauenskette, die nicht mehr von der statischen Kette des CRTM abhängt.

Die einzige Gemeinsamkeit zwischen CRTM und DRM ist die Verwendung des RTR, wie es vom TPM zur Verfügung gestellt wird.

### *Root Of Trust For Reporting*

Diese Vertrauenswurzel beschützt die Messwerte auf sichere Art und Weise. Konkret geschieht dies darin, dass die Messungen in den PCRs des TPM abgelegt werden. Diese können darüber hinaus externen Parteien signiert übermittelt werden, um so den Zustand der Plattform zu attestieren (z.B. mittels DAA).

Tabelle 1 zeigt eine Auflistung der verfügbaren PCRs in einem TPM Version 1.2 sowie die zugehörige definierte Verwendung.

| PCR Index | Alias |
|---|---|
| **0-15** | **Static RTM** |
| 0 | CRTM, BIOS und Plattform-Erweiterungen |
| 1 | Plattform Konfiguration |
| 2 | Option ROM Code |
| 3 | Option ROM Configuration und Daten |
| 4 | IPL Code *(IPL=Initial Program Loader - Bootloader)* |
| 5 | IPL Code Configuration und Daten |
| 6 | State Transitionen und Ereignisse |
| 7 | Reserved for future usage |
| 8-15 | unspecified |
| **17-22** | **Dynamic RTM** |
| *16* | *Debug* |
| 17 | Locality 4 |
| 18 | Locality 3 |
| 19 | Locality 2 |
| 20 | Locality 1 |
| 21 | T/OS Controlled (*T/OS = Trusted OS*) |
| 22 | T/OS Controlled |
| 23 | Anwendungsspezifisch |

*Tabelle 1: Belegung der PCR-Register*

***Root Of Trust For Storage***

Die RTS-Komponente muss vertrauenswürdigen Inhalt des TPM beschützen. Ein Beispiel ist die Speicherung von TPM-Schlüsseln. Da das TPM nur einen begrenzten Speicherplatz bietet, ist es Aufgabe des RTS, Daten vertraulich und integritätsgeschützt extern (d.h. außerhalb des TPM) abzuspeichern.
Darüber hinaus stehen RTR und RTS in engem Zusammenhang, denn es ist durchaus möglich, durch das RTS zu speichernde Daten an Eigenschaften aus dem RTR zu knüpfen. Daher sieht die TCG vor, dass sich RTR und RTS auf einem Chip befinden. Im Falle des TPM bedeutet dies, dass RTR und RTS in einem *Package* sind, d.h. ohne externe Busse miteinander kommunizieren können.

### 2.3.3 TPM-Shielded Keys

Das TPM besitzt eine Kryptographie-Komponente, mit der man u.a. RSA-Schlüssel erzeugen kann. Darüber hinaus kann man für jeden erzeugten Schlüssel festlegen,

zu welchen Zwecken dieser eingesetzt werden darf. Das TPM unterschiedet hier vereinfacht ausgedrückt zwischen Verschlüsselungs- und Signaturschlüsseln. Darüber hinaus kann man noch festlegen, ob diese Schlüssel migrierbar sein dürfen, d.h. ob sie ggf. in ein anderes TPM übertragen werden können. Laut Spezifikation ist die Anzahl an Schlüsseln, die ein TPM erzeugen und verwalten kann, unbegrenzt. Dadurch, dass das TPM aber nur eine stark begrenzte Menge an nicht-flüchtigem Speicher besitzt, muss es eine Möglichkeit geben, diese Schlüssel sicher auf externen Datenträgern abzulegen. Dies kann mit Hilfe des RTS und einem speziellen Speicherschlüssel – dem *Storage Root Key* (s. Erklärung weiter unten) erreicht werden und so ermöglichen, eine Schlüsselhierarchie aufzubauen. Ein kryptographischer Schlüssel wird hierzu im TPM erzeugt, d.h. in einer isolierten, geschützten Umgebung und mit gutem Zufallszahlengenerator. Der Schlüssel wird dann mit dem öffentlichen Teil des Speicherschlüssels verschlüsselt und nach außen übermittelt. Das Betriebssystem kann diesen Schlüssel dann abspeichern. Dadurch, dass er verschlüsselt ist, kann er extern nicht benutzt werden. Zur Verwendung muss er zurück in das TPM – und damit in die geschützte Umgebung – geladen werden. Das TPM entschlüsselt den Schlüssel dann mit seinem privaten Schlüssel und kann so den Schlüssel wieder verwenden. Um dies zu realisieren, besitzt jedes TPM zwei Schlüssel, welche fest im TPM gespeichert sind:

### *Endorsement Key*

Jedes TPM wird mit einem „eingebauten“ nicht-migrierbaren 2048-Bit RSA-Schlüssel, dem so genannten *Endorsement Key* (EK) ausgeliefert, welcher beim Herstellungsprozess innerhalb des TPMs generiert oder außerhalb des TPMs durch den Hersteller erzeugt und dann ins TPM eingebracht wurde. „Eingebaut“ meint, dass es unmöglich ist, diesen Schlüssel des TPMs zu kopieren oder zu löschen[5]. Dadurch kann das TPM und die zugehörige Plattform eindeutig identifiziert werden. Die Entität, die den EK erzeugt, erzeugt eine Berechtigungsbeglaubigung (*Endorsement Credential*), welche den Beweis liefert, dass der EK ordentlich erstellt und in einem echten TPM integriert wurde. Dadurch, dass der EK die Plattform-Identität eindeutig nachweisen kann, ist der EK relevant für z.B. die Privatsphäre. Während der Erzeugung des *Storage Root Key* (siehe folgender Abschnitt) wird der öffentliche Teil des EK benötigt, um die notwendigen Benutzer- und Besitzergeheimnisse in das TPM einzubringen. Die Geheimhaltung des EK und das Vertrauen auf den EK ist essentiell.

### *Storage Root Key*

Der *Storage Root Key* (SRK) ist ebenfalls ein 2048-Bit RSA Schlüsselpaar. Er wird als Speicherschlüssel verwendet, d.h. mit Hilfe des SRK ist es möglich, TPM-geschützte Schlüssel so zu verschlüsseln, dass sie außerhalb des TPMs aufbewahrt werden können. Dies erzeugt eine Schlüsselhierarchie auf einem Speichermedium, wie beispielsweise einer Festplatte. Der SRK kann das TPM nicht verlassen (da er ebenfalls ein nicht migrierbarer Schlüssel ist). Der SRK wird bei der Inbesitznahme

5 Je nach TPM ist es unter Umständen dennoch möglich, den EK zu löschen und einen neuen EK zu erzeugen. Jedoch verliert das TPM dann seine Beweiskraft, da durch das fehlende *Endorsement Credential* des Herstellers zu dem neuen EK nicht mehr nachgewiesen werden kann, ob sich der EK in einem echten TPM befindet.

der Plattform (*Take Ownership*) erzeugt. Er kann bei einer erneuten Inbesitznahme der Plattform neu erzeugt werden, was allerdings die gesamte alte Schlüsselhierarchie zerstört, und somit sowohl alle mit Hilfe eines *Storage Keys* verschlüsselten Daten unbrauchbar macht.

### 2.3.4 Nicht-flüchtiger Speicher (Non-Volatile RAM)

Jedes TPM der Version 1.2 besitzt mindestens 512 Byte nicht-flüchtigen Speichers, welcher zum sicheren Ablegen von Daten in einer geschützten Umgebung benutzt werden kann. Die Menge des NVRAMs ist nach oben nicht begrenzt, Herstellern steht es frei, auch mehr Speicher anzubieten. Da dieser Speicherplatz aber produktionsbedingt sehr kostenintensiv ist, beläuft sich die Menge an NVRAM in der Regel auf wenige Kilobyte.
Sicherer Speicher bedeutet hier, dass sich der Speicherbereich in einer isolierten Umgebung (*shielded location*) innerhalb eines manipulationssicheren Chips befindet (*tamper-resistant*). Dieser Speicher kann nur durch fest definierte Befehle gelesen oder geschrieben werden. Darüber hinaus existiert ein Zugriffskontrollsystem. So kann man den Zugriff auf den NVRAM an eine bestimmte Konfiguration binden, d.h. nur unter gewissen Umständen ist ein Zugriff überhaupt erst möglich.

### 2.3.5 Sealing

Die *Sealing*-Funktionalität (zu deutsch: Versiegeln (von Daten)) ist ein Verschlüsselungsmechanismus, welcher Daten mit Hilfe eines asymmetrischen RSA-Schlüssels[6] verschlüsselt und (optional) nur unter einer explizit definierten Konfiguration wieder freigibt. Das bedeutet, dass man während des Versiegelungs-Vorgangs angeben kann, unter welchen Bedingungen (d.h. unter welcher PCR-Konfiguration) sich die Daten wieder entschlüsseln lassen dürfen. Da das Versiegeln und Entschlüsseln innerhalb des TPM geschieht, ist es nicht möglich, versiegelte Daten auf oder für einen anderen Rechner zu erzeugen, sie müssen im Ziel-TPM generiert werden. Dies ermöglicht vor allem, Daten explizit an eine bestimmte Plattform zu binden. Die versiegelten Daten können zusätzlich noch mit einem Passwort geschützt werden, sodass im sichersten Falle die Daten nur auf der Ursprungsplattform, in einem wohl-definierten Zustand mit einem korrekten Passwort wiederhergestellt werden können.
Die Menge an Daten, die mittels des TPM pro Versiegelungsbefehl versiegelt werden können, liegt bei wenigen Byte (ca. 150 Byte). Daher wird oft ein Zwischenschritt unternommen, indem große Mengen an Daten mittels eines symmetrischen Schlüssels verschlüsselt werden und lediglich dieser Schlüssel dann versiegelt wird.

6 Zum Sealing kann lediglich ein nicht-migrierbarer TPM-Storage-Key verwendet werden (TPM-Spec. 1.2, Teil 3, Seite 61ff). Die möglichen Schlüssellängen sind 512-, 1024- oder 2048-Bit.

### 2.3.6 TPM Vertrauensbeglaubigung

Eine vertrauenswürdige Plattform wird mit Vertrauensbeglaubigungen (d.h. digitalen Zertifikaten) ausgeliefert, die eine Aussage darüber treffen, dass ihre Komponenten unter Einhaltung der TCG-Spezifikationen konstruiert wurden:

#### *Endorsement Credential*

Wie schon erwähnt, soll die Berechtigungsbeglaubigung (*Endorsement Credential*) einen Beweis dazu liefern, dass der *Endorsement Key* ordnungsgemäß erzeugt und in ein echtes TPM eingebettet wurde. Dieses wird durch die Entität angefertigt, die den EK erzeugte. Diese Berechtigungsbeglaubigung enthält den Namen des TPM-Herstellers, die TPM-Seriennummer, die TPM-Version und den öffentlichen Teil des EK.

#### *Conformance Credential*

Die Konformitätsbeglaubigung (*Conformance Credential*) wird ausgestellt, für Plattformen, die einen TPM enthalten. Sie soll aufzeigen, dass der *Trusted Building Block* (TBB – bestehend aus RTR/RTS/RTM) und dessen Implementierung der Prüfungen (z.B. *chain of trust*) vertrauenswürdig ist. Die Konformitätsbeglaubigung enthält keine sensiblen Informationen oder Informationen, die dazu verwendet werden können, eine spezifische Plattform eindeutig zu identifizieren. Das *Conformance Credential* sollte von einer vertrauenswürdigen Instanz ausgestellt werden (z.B. einem Evaluator, einem (TPM-/BIOS-)-Hersteller oder einem Händler).

#### *Platform Credential*

Die Plattform-Vertrauensbeglaubigung (*Platform Credential*) wird durch den Plattformhersteller, Anbieter oder eine unabhängige Entität erteilt. Sie soll aufzeigen, dass die Plattform ein TPM, wie durch das *Endorsement Credential* beschrieben, sowie einen TBB, wie durch das *Conformance Credential* beschrieben, enthält. Die Plattform-Vertrauensbeglaubigung enthält den Namen des Herstellers der Plattform, die Plattform-Modellnummer und -Modellversion und Referenzen auf das *Endorsement Credential* und die *Conformance Credentials*. Das *Platform Credential* ist kritisch bezüglich der Privatsphäre, da es Informationen enthält, die zur eindeutigen Identifizierung einer spezifischen Plattform verwendet werden können.

## 2.4 Chain of Trust

Die Integrität einer Plattform kann nur gewährleistet werden, wenn die Vertrauenskette (*chain of trust*) nicht unterbrochen wird. Die Verankerung dieser Kette besteht aus der Kombination aus CRTM – in modernen PCs eine Erweiterung des BIOS – und dem TPM.
Eine funktionierende Vertrauenskette ist in mindestens einem der folgenden Szenarien erforderlich:

a) Jemand möchte die Integrität seiner eigenen Plattform absichern und auf vertrauliche Daten nur dann zugreifen können, falls das System unverändert im definierten Zustand ist.
b) Jemand möchte die eigene Plattformkonfiguration einer externen Stelle beweisen. Ein mögliches Szenario ist ein Home-Office-Mitarbeiter, der sich mit dem firmeninternen Netzwerk verbinden möchte, wobei die Firma den Zugang nur dann gewährt, wenn bestimmte Sicherheitsrichtlinien eingehalten werden und das IT-System des Mitarbeiters in einem ordnungsgemäßen Zustand ist.

Da die Vertrauenswürdigkeit der gesamten Plattform von jedem einzelnen Glied der Vertrauenskette abhängt, muss immer sichergestellt werden, dass diese nicht unterbrochen wird.

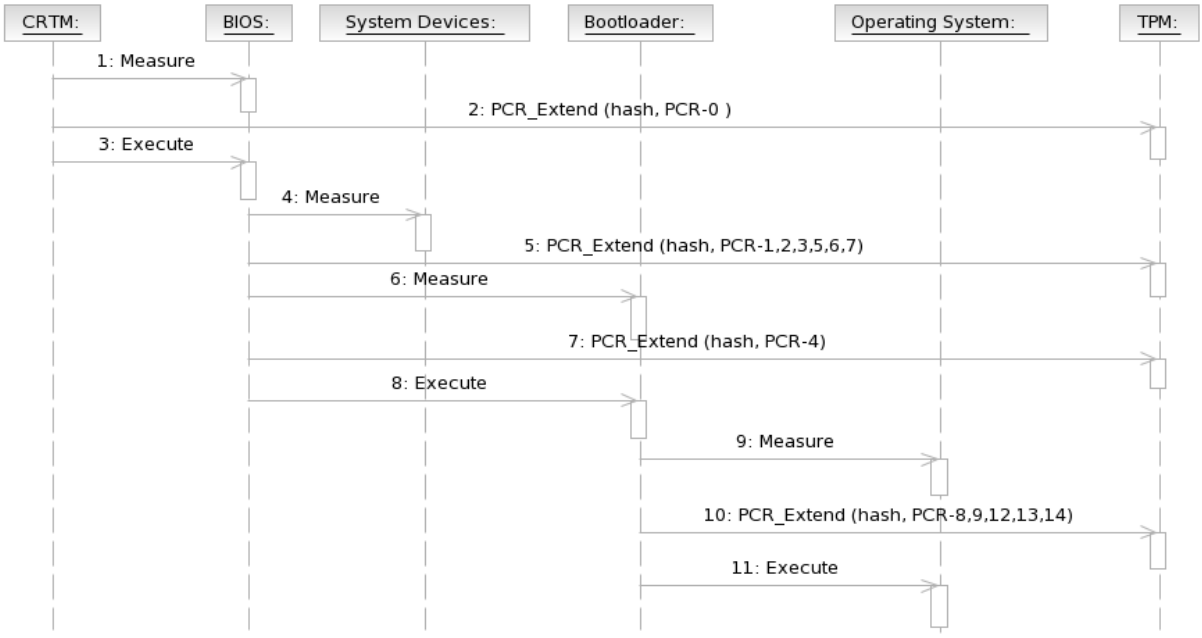


*Abbildung 1: Ablauf eines vertrauenswürdigen Bootvorgangs mit CRTM*

**Erforderliche Schritte**. Abbildung 1 zeigt die erforderlichen Schritte zum Aufbau einer gültigen Vertrauenskette bei einem TCG-basierten Bootvorgang nach dem Einschalten des IT-Systems.

1. **CRTM:** Das CRTM misst das BIOS und schreibt das Ergebnis in das TPM.
2. **BIOS:** Das BIOS misst sämtliche auszuführende Firmware (etwa das Option-ROM der Grafikkarte) und den Bootloader (der im MBR des Bootmediums gespeichert ist). Das Ergebnis wird ebenfalls im TPM gespeichert.
3. **Bootloader:** Der Bootloader setzt die Vertrauenskette fort, indem er alle geladenen Systemdateien misst und ebenfalls das Ergebnis in das TPM schreibt.
4. **Betriebssystemkern:** An dieser Stelle muss das Betriebssystem dafür Sorge tragen, dass die Vertrauenskette fortgesetzt wird. Dies ist erforderlich, da sich das System während der Ausführung verändert.

Nach Abschluss des Bootvorgangs ist das Betriebssystem geladen und jeder ausgeführte Code wurde überprüft und der Messwert im TPM gespeichert. Nun kann die Integrität der Plattform verifiziert oder einer anderen Stelle attestiert werden. Da die TCG-Definition von *Trusted Boot* direkt mit Abschluss des Bootvorganges endet, obliegt es dem nun laufenden Betriebssystems, die Vertrauenskette fortzusetzen.

## *2.5 Smartcard*

### 2.5.1 Grundlagen

Smartcards sind dem Namen nach sogenannte intelligente Karten. Davon gibt es grundsätzlich zwei Kategorien, die sog. Speicherkarten und die Prozessorkarten.

Beide sind physikalisch und elektrisch spezifiziert im ISO-Standard 7816 Teil 1,2 bzw. Teil 3. Sie werden in einen Kartenleser eingesteckt, dort kontaktiert und können so über das Lesegerät mit einem PC kommunizieren.

Auf Speicherkarten lassen sich lediglich Daten speichern und wieder abrufen. Für den hier vorgesehenen Zweck werden daher Prozessorkarten verwendet. Diese enthalten eine eigenständige Laufzeitumgebung mit Prozessor, Arbeitsspeicher, Langzeitspeicher usw. Sie sind von verschiedenen Herstellern mit unterschiedlichen Betriebssystemen erhältlich. Besonders gut eignen sie sich für kryptographische Anwendungen, da sich solch ein kleines System sehr gut absichern und evaluieren lässt. So ist je nach Modell die Hardware besonders gehärtet gegen invasive und nichtinvasive physikalische Angriffe, es sind außerdem spezielle Speicherbereiche für kryptographische Schlüssel, und Beschleunigungsfunktionen für Kryptoalgorithmen auf diesen Karten enthalten.

### 2.5.2 Leser

Der Zugriff auf Smartcards erfolgt durch den PC vornehmlich durch Nutzung des PC/SC Standards. Dieser bietet eine einheitliche Schnittstelle zu Anwendungen welche Smartkarten nutzen wollen. Zu den Lesegeräten wird in der Regel ein Treiber geliefert, der den PC/SC Standard unterstützt, neben Windows sind in der Regel auch Treiber auch für Linux verfügbar.

Bei den Lesegeräten selbst unterscheidet man verschiedene Sicherheitsklassen:

- Klasse 1 – Diese Geräte stellen lediglich eine Verbindung zwischen der Smartcard und dem Rechner her.
- Klasse 2 – Diese stellen über die Funktionen des Klasse-1-Lesers hinaus eine eigene Zahlentastatur zur Verfügung, über die eine PIN eingeben werden kann. Diese wird dann direkt (d.h. ohne Umweg über den eventuell kompromittierten PC) an die Smartcard gesandt wird. So kann ein Ausspähen der PIN von der PC-Software aus wirksam verhindert werden.
- Klasse 3 – Diese Kartenleser stellen über die Funktionen des Klasse-2-Lesers hinaus ein Display zur Verfügung. Dieses Display kann exklusiv durch die Smartcard angesteuert werden. Damit können Informationen zur Verifikation angezeigt werden, die vom (eventuell kompromittierten) PC weder ausgelesen noch verändert werden können. Klasse-3-Leser sind nicht sehr verbreitet und bei fast allen Modellen gibt es zusätzlich die Möglichkeit, den Inhalt des Displays vom PC aus zu steuern.

### 2.5.3 Kommunikation

Die Kommunikation mit Chipkarten funktioniert mittels Nachrichten, den sog. APDUs, (Abk. für „*Application Protocol Data Unit*“, spezifiziert in ISO7816 Teil 4) Hierbei ist zuerst einmal lediglich das Format dieser Nachrichten festgelegt, nicht ihr Inhalt.

Dabei kann eine Smartcard nach diesem Standard immer nur auf eine Anfrage (*command APDU*) eine Antwort (*response APDU*) schicken, nicht jedoch von sich aus aktiv werden.

Solche APDUs sind teilweise in den Standards fertig spezifiziert, teilweise kann jeder Smartcard-Entwickler eigene APDUs definieren.

Hierbei ist zu beachten, dass normale APDUs nicht mehr als 256 Byte Daten in einem Kommando transportieren können. Bis vor einiger Zeit stellte dies kein Problem dar, da kein Kommando mehr als 256 Byte Daten benötigte. Die Grenze wird durchbrochen durch die Verwendung von RSA mit großen Schlüssellängen, dort sind keine Signaturen mit Schlüssellängen größer als 1856 BIT möglich.

Die Beschränkung hebt die Verwendung von „*Extended Length APDUs*“ auf, die aber derzeit noch nicht von allen Kartenlesern implementiert wurden. Bei der Beschaffung ist daher darauf zu achten, dass nicht nur die eingesetzten Karten entsprechend große Schlüssellängen unterstützen, sondern auch die Kartenleser und deren Treiber.

### 2.5.4 PKCS #11

In den Standard PKCS#11 („*Public-Key Cryptography Standard #11*“, auch „*cryptoki*“ genannt) wird eine Programmierschnittstelle in der Programmiersprache C (bzw. C++) zum abstrahierten Zugriff auf Kryptofunktionen von Krypto-Token definiert. Diese API wird jeweils vom Tokenhersteller in Form einer Bibliothek bereitgestellt. Die Bibliothek widerum kommuniziert mit dem Token. Dies bedeutet in diesem Fall das Umsetzen auf konkrete APDUs der speziellen Smartcard und die Kommunikation mit der entsprechenden Smartcard. Dies ist jedoch keinesfalls fest in PKCS#11 definiert, die unterliegende Kommunikation kann auch mit einem völlig anders spezifizierten Token stattfinden.

In PKCS #11 sind die Objekte (Daten, Schlüssel), Kryptoperationen usw., im Hinblick auf kryptographische Token beschrieben.

Momentan wird bei TrueCrypt lediglich die Funktion verwendet, auf einem durch eine PIN geschützten Datenspeicher ein oder mehrere Keyfiles (Dateien) abzulegen. Diese werden im Anschluss von der Karte nur dann an den Leser übermittelt, wenn zuvor die PIN richtig eingegeben wurde. Weitergehende Kryptofunktionen werden in TrueCrypt derzeit nicht benutzt.

### 2.5.5 Einsatz

Prinzipiell ist es möglich, in einer Karte nahezu beliebige eigene Funktionen zu implementieren und diese über die beschriebenen Schnittstellen zur Verfügung zu stellen. Damit kann dann z.B. eine Funktion „*Remote Attestation*“(siehe 4.3.3) realisiert werden. Hierzu eignen sich insbesondere Smartcards mit einem eigenen

Java-Betriebssystem, wie zum Beispiel die JCOP (*Java Card OpenPlatform*) Karten von NXP.

### 2.5.6 RFID

In der neuesten Zeit haben sich neben den kontaktbehafteten Karten auch kontaktlose Smartcards etabliert. Hier kommuniziert die Karte mit dem Lesegerät ausschließlich über Funk; auch die Stromversorgung der Karte erfolgt hierbei drahtlos. Werden diese kontaktlosen Smartcards zum Berechnen von kryptografischen Funktionen (z.B.: RSA oder AES) eingesetzt, so wird in der Regel der ISO 14443 Standard genutzt, bei dem die Übertragungsfrequenz 13.56 MHz beträgt.

Die maximale Entfernung zwischen Karte und Lesegerät liegt hier bei ca. 10cm. Unter guten Umständen lässt sie sich erhöhen, eine Entfernung von über einem Meter ist jedoch technisch nicht möglich.

Im Gegensatz zu einer kontaktbehafteten Karte, die in ein Lesegerät gesteckt werden muss, hat der Besitzer bei einer kontaktlosen Smartcard nicht zu jeder Zeit die Kontrolle über die Nutzung der Karte. Führt er die Karte beispielsweise in seiner Brieftasche mit sich, kann unbemerkt eine Kommunikation zur Karte aufgebaut werden (z.B.: in der U-Bahn). Ist in einem solchen Fall der Besitz der Karte das alleinige Authentifizierungsmerkmal (z.B.: für eine Türöffnung) so kann mit einem mobilen Gerät, einer mobilen Internetverbindung und einem Kartensimulator die Karte von einem Angreifer sozusagen mit einem „Verlängerungskabel“ unbemerkt genutzt werden. Aus diesem Grund ist der Einsatz von RFID Karten nicht in jedem Fall ratsam.

Beim Einsatz innerhalb einer Festplattenverschlüsselung in Zusammenhang mit einer PIN-Eingabe sind im Prinzip zwei Punkte zu beachten:

1. Ausspähbarkeit der Daten (PIN und Verschlüsselungsschlüssel), die über Funk übertragen werden.
2. Privatsphäre: Durch Besitz einer Karte, die unbemerkt ausgelesen werden kann, können unter Umständen Bewegungsprofile erstellt werden, da die Karten alle eine eindeutige Seriennummer besitzen, über die die RFID Kommunikation aufgebaut wird. Hier können jedoch ähnlich wie beim elektronischen Reisepass zufällige Seriennummern bei jeder Aktivierung generiert werden.

# 3 Existierende Angriffe

Auf die in Kapitel 1.1 genannten Werte werden in AP3 detaillierte Angriffsbäume aufgezeigt. Dass diese Angriffe in der Realität tatsächlich vorkommen, soll dieses Kapitel untermauern. Einigen Angriffen kann durch organisatorische Maßnahmen und Annahmen begegnet werden. Diese sind ebenfalls in AP3 definiert und hier den existierenden Angriffen zugeordnet.

## *3.1 Passwort per Brute-Force angreifen*

Hier versucht der Angreifer, das Passwort des Volume Headers zu erraten. Hierzu werden systematisch folgende Schritte ausgeführt, bis das korrekte Passwort gefunden wurde:

1. Passwort erzeugen
2. Schlüssel aus Passwort (und ggf. Salt) ableiten
3. Volume Header entschlüsseln
4. Prüfsumme des entschlüsselten Headers verifizieren

In Schritt 1 kann zur Erzeugung der Passwörter sowohl ein Wörterbuch verwendet werden (*Dictionary Attack*), als auch ein Algorithmus, der systematisch alle Passwörter eines bestimmten Passwortraums (z.B. alle alphanumerischen Zeichen) nacheinander erzeugt. Auch Kombinationen aus beiden Verfahren sind möglich.

**Anmerkung:** Ist das Zielsystem unter Kontrolle eines Angreifers, existieren gegen diesen Angriff keine Lösungsmöglichkeiten, die Brute-Force generell verhindern. Es gibt z.B. bereits Implementierungen, um TrueCrypt per Brute-Force anzugreifen (Quelle: *http://www.accessdata.com/decryptionTool.html#dna,* verteiltes Brute-Force Tool, dass auch TrueCrypt unterstützt).

Durch den Einsatz von PBKDF-2 und einem Salt in Schritt 2 kann jedoch die Geschwindigkeit eines Brute-Force Angriffs erheblich reduziert werden, da PBKDF-2 kostspielige Berechnungen erfordert und der Einsatz eines Salts die Vorberechnung verhindert. Es muss lediglich sichergestellt werden, dass das gewählte Passwort nicht so simpel ist, dass es binnen weniger Versuche erraten werden kann.

## *3.2 Passwort per Keylogger erhalten*

Ein Keylogger dient der Aufzeichnung der Tastatureingaben des Benutzers mit dem Ziel, das Passwort für das verschlüsselte Volume zu erhalten. Keylogger können sowohl in Form von Hardware (d.h. zwischen Tastatur und Gerät) als auch in Software realisiert werden.

Bekannte Formen von Hardware-Keyloggern sind z.B. sogenannte PS/2 oder USB Keylogger, die zwischen Tastaturstecker und Rechner angebracht werden, und durch ihre Größe kaum auffallen. Für Laptops sind ähnliche Vorrichtungen unterhalb der Tastatur denkbar. Weiterhin existieren Keylogger in Form von PCI/Mini-PCI-Steckkarten, die auch in Laptops einsetzbar sind. (Quelle:

http://www.keyghost.com/USB-Keylogger.htm sowie http://www.keyghost.com/PCI-MPCI-Keylogger.htm, Hardware-Keylogger für verschiedene Einsatzgebiete).

Software-basierte Keylogger hingegen versuchen, die Eingaben auf verschiedenen Ebenen innerhalb des Rechners abzufangen. Bereits bevor ein Betriebssystem gebootet wurde, kann ein Programm den Interrupt 0x9h (Tastatureingabe) abfangen und so eventuelle Eingaben für eine Pre-Boot-Authentifizierung abhören. (Quelle: http://www.castledragmire.com/Posts/BIOS_Level_Key_Logger, Beschreibung und Code für einen Keylogger auf Interrupt-Basis mit TrueCrypt als Ziel).

Innerhalb des Betriebssystems gibt es weitere zahlreiche Möglichkeiten zum Abhören der Eingaben, z.B. über einen Gerätetreiber im Kernel Mode. (Quelle: http://www.viruslist.com/de/analysis?pubid=200883538#pt16, Auflistung von Möglichkeiten zur Implementierung eines Keyloggers im User Mode oder Kernel Mode).

**Anmerkung:** Siehe Annahme 22.

## 3.3 Passwort aus der Auslagerungsdatei auslesen

Befindet sich das Passwort in einem nicht speziell geschützten Speicherbereich des Programms, so kann das Passwort oder der davon abgeleitete Schlüssel vom Betriebssystem ausgelagert werden. Liegt die Auslagerungsdatei des Betriebssystems auf einer unverschlüsselten Partition, so kann ein Angreifer diese Informationen auch nachträglich aus der Auslagerungsdatei extrahieren. Das Passwort kann hier je nach Implementierung des Zielprogramms anhand der umliegenden Strukturen in der Auslagerungsdatei aufgespürt werden. Für abgeleitete Schlüssel (z.B. AES Key Schedule) existieren ebenfalls fertige Programme, die diese entdecken und extrahieren können (siehe dazu auch 3.4.3 ).

(Quelle: http://jessekornblum.livejournal.com/246616.html, Plugin für „Volatility“ zum Extrahieren von TrueCrypt Passwörtern aus Speicherabbildern).

**Anmerkung:** Siehe Annahme 28.

## 3.4 Passwort aus dem Hauptspeicher auslesen

### 3.4.1 via DMA und FireWire

Die FireWire Spezifikation (OHCI-1394) erlaubt einem externen FireWire-Gerät unter anderem direkte Lese- und Schreibzugriffe auf den Speicher des angeschlossenen Hostrechners (DMA). Diese Möglichkeit kann dazu missbraucht werden, den Speicher des Rechners direkt nach Passwörtern oder sonstigem Schlüsselmaterial zu durchsuchen (mit Methoden wie in 3.4.3 ). Für diese Art von Angriff existieren ebenfalls frei verfügbare Implementierungen und Geräte.

(Quelle: http://md.hudora.de/presentations/#firewire-pacsec, Präsentation, Tools und Demonstration).

**Anmerkung:** Siehe Annahme 20.

### 3.4.2 via Tastasturpuffer des BIOS

Solange kein Betriebssystem gestartet wurde, ist das BIOS für die Bereitstellung von Tastatureingaben zuständig (z.B. auch für eine Eingabe während der Pre-Boot-Authentifizierung). Hierfür hat das BIOS einen Puffer, der bei gängigen Implementierungen ca. 16 Zeichen fasst. Üblicherweise wird dieser Puffer nicht gelöscht, wenn das Betriebssystem die Kontrolle über die Eingaben übernimmt. Mit verschiedenen Techniken ist es daher möglich, die letzten Eingaben aus diesem Puffer zu Lesen und so z.B. das Passwort der Pre-Boot Authentifizierung zu erhalten. Dies hängt unter anderem damit zusammen, dass einige Puffer als *Rolling Buffer* implementiert werden. So wird nach dem Abrufen eines Zeichen aus dem Buffer lediglich der Pointer um ein Zeichen weitergerückt, nicht aber der freigewordenen Speicherplatz unmittelbar wiederverwendet. Erst wenn der Puffer einmal durchlaufen ist, werden alte Speicherinhalte wieder überschrieben.

Diese Schwachstelle konnte bis TrueCrypt 5.0 ausgenutzt werden. Seit TrueCrypt Version 5.1 ist dieses Problem behoben, d.h. TrueCrypt selbst sorgt für ein Löschen des Tastaturpuffers (siehe AP3).

(Quelle: *http://www.ivizsecurity.com/research/preboot/preboot_whitepaper.pdf*, „Bypassing pre-boot authentication passwords by instrumenting the BIOS keyboard buffer“, veröffentlicht auf *DEFCON 16*).

**Anmerkung:** Siehe Annahmen 22, 24.

### 3.4.3 via Coldboot-Mechanismen

Der Arbeitsspeicher (*RAM*) eines Rechners enthält zur Laufzeit das Passwort bzw. Schlüsselmaterial der eingesetzten Verschlüsselung. Oft wird fälschlicherweise angenommen, diese Informationen wären sofort verschwunden, sobald der Rechner neugestartet bzw. ausgeschaltet wird. In der Tat verlieren gängige DRAM Module ihre Daten nachdem kein Strom mehr anliegt. Dieser Prozess setzt zwar sofort ein, führt jedoch erst nach einigen Sekunden zu einem starken Datenverlust. Durch externe Kühlung kann der Datenverlust weiter verlangsamt werden. Da der Verfall der Daten sofort einsetzt (wenn auch langsam), sind bei der Suche eventuelle Bitfehler zu berücksichtigen. Es existieren fertige, frei verfügbare Implementierungen, die beispielsweise AES Key Schedules oder RSA Schlüssel mit Bitfehlern im Speicher finden und diese korrekt extrahieren können. Mittels dieser Techniken sind zwei Arten von Angriffen möglich:

- Neustart der laufenden Maschine und Booten eines eigenen Betriebssystems (z.B. von USB) um den Speicher auszulesen (mit oder ohne Kühlung möglich).
- Ausbau der DRAM-Module aus einem laufenden oder in den Standby-Modus (ACPI S3) versetzten Gerät und Einbau in ein Zweitgerät, um sie dort auszulesen (generell nur mit Kühlung möglich).

Beide Angriffe sind mit vergleichsweise wenig Aufwand durchzuführen und daher auch praktisch einsetzbar.

(Quelle: *http://citp.princeton.edu/memory/*, Forschungsbericht, Tools und Demos).

**Anmerkung:** Siehe Annahmen 20, 24, 25.

## *3.5 Passwort durch Manipulation am Bootloader protokollieren*

Die im Folgenden vorgestellten Angriffe haben zum Ziel, in Besitz des während des Startvorgangs benötigten Passworts zu erlangen. Während Kapitel 3.3 und 3.4 passive Methoden verwenden, um nachträglich an das Passwort zu gelangen. Sind die in diesem Kapitel vorgestellten Angriffe analog zu Kapitel 3.2 mit dem Unterschied, dass hier anstelle der physischen Schnittstelle der Bootloader manipuliert wird, um das Passwort zu protokollieren.

### 3.5.1 Evil-Maid Angriff auf TrueCrypt

Bei dieser Art von Angriff wird entweder durch Manipulation des MBR oder durch zusätzliches Laden von Code über andere Vektoren (z.B. PCI EEPROM) beim Bootvorgang Schadcode geladen, der die Tastatureingaben der Pre-Boot Authentifizierung abfängt. Technisch gesehen handelt es sich hierbei um eine Form von Keylogger (siehe Kapitel 3.2 ). Eine fertige Implementierung dieses Angriffs existiert bereits für TrueCrypt. (Quelle: *http://theinvisiblethings.blogspot.com/2009/10/evil-maid-goes-after-truecrypt.html*, Beschreibung und Implementierung des Evil-Maid Angriffs für TrueCrypt).

**Anmerkung:** Siehe Annahmen 20, 21, 22, 23, 31, 32.

### 3.5.2 Reproduktion des Bootloaders (optisch) am Beispiel von Bitlocker

Die Windows-eigene Festplattenverschlüsselung „Bitlocker“ verwendet ein TPM. Zum einen nutzt es das TPM, um den Festplattenschlüssel (und damit indirekt die Festplatte) an die aktuelle Hardware-Plattform zu binden. Zum anderen wird der Festplattenschlüssel an die Plattformkonfiguration gebunden und so nur dann freigeben, wenn die Integrität der *Chain of Trust* sichergestellt wurde. Wird nun der Bootloader manipuliert, kann dieser zwar das Geheimnis (d.h. den Festplattenschlüssel) nicht rekonstruieren, jedoch merkt der Benutzer dies erst, nachdem er das dazu benötige Passwort eingegeben hat.

Der hier vorgestellte Angriff macht sich genau dieses Problem zu Nutze. Zunächst wird der existierende Bootloader (im MBR) durch einen manipulierten Bootloader ersetzt. Dieser startet eine optisch identische Oberfläche, sodass der Benutzer den Unterschied zum ursprünglichen Bootloader nicht bemerkt und nach Aufforderung sein Passwort eingibt. Der modifizierte Bootloader protokolliert diese Benutzereingaben und speichert sie ab. Durch die Manipulation am Bootloader kann der Startvorgang nicht erfolgreich fortgesetzt werden, daher wird der manipulierte Bootloader mit einer *Restore* Funktion ausgestattet, die den Original-Bootloader wiederherstellt. Nach erfolgreicher Protokollierung des Passworts wird erst die Wiederherstellung und dann ein Neustart ausgelöst. Daraufhin kann das System normal gestartet werden. Ein Angreifer kann dann die Maschine entwenden, das gespeicherte Passwort auslesen und mit Hilfe des originalen Bootloaders starten.

Diese Implementierung ist angreifbar, weil zwar der TPM zum *Sealen* verwendet wird, aber die Integrität des Bootloaders nie extern validiert wird.

(Quelle: *http://testlab.sit.fraunhofer.de/bitlocker-skimming/*, Präsentation und Forschungsbericht zu BitLocker Schwachstellen).

**Anmerkung:** Siehe Annahmen 20, 21, 22, 23.

### 3.5.3 Stoned Bootkit

Das *Stoned Bootkit* ist vom Aufbau her ähnlich zum Evil-Maid Angriff (3.5.1 ). Hierbei wird der MBR so verändert, dass Schadsoftware geladen wird. Diese verändert die *Interrupt Handler* so, dass die nachfolgende Software inkl. Betriebssystem diese Software nicht entdecken kann. Da die Schadsoftware vollen Zugriff auf das System hat, kann sie sowohl Passwörter protokollieren, als auch direkt Einfluss auf das Betriebssystem nehmen.

(Quelle: *http://www.stoned-vienna.com/downloads/Paper.pdf*, Beschreibung des Stoned Bootkits).

**Anmerkung:** Siehe Annahmen 20, 21, 22, 23, 31, 32.

# 4 Lösungsmöglichkeiten

Dieses Kapitel zeigt verschiedene Lösungsmöglichkeiten auf, um Datenträger mit *Full-Disc-Encryption* sicher zu starten. Hierzu werden verschiedene Voraussetzungen an Hard- und Software getroffen, welche in den einzelnen Lösungsmöglichkeiten aufgeführt sind.

Die Aufwände der einzelnen Lösungen (d.h. insbesondere Design, Entwicklung, Installation und Tests) werden jeweils kategorisiert in die folgenden Schwierigkeitsgrade:

- **Hoch**: Um diese Lösung zu realisieren, sind aufwändige Maßnahmen nötig, d.h. es wird der Einsatz einer Spezialhardware gefordert, welche ggf. sogar erst noch entwickelt werden muss. Als geschätzer Realisierungsaufwand können hier mehrere Personenmonate gerechnet werden.
- **Mittel**: Bei Lösungen dieser Art ist keine neue Hardware notwendig, vorhandene Hardwarekomponente (wie z.B. Smartcards oder TPMs) werden verwendet. Softwarekomponenten (wie z.B. Bootloader) müssen teilweise aufwändig angepasst werden, sodass hier mit einem Aufwand von mehreren Personenwochen zu rechnen ist.
- **Gering**: Hierbei handelt es sich um Lösungen mit geringem Aufwand, d.h. eine Realisierung kann in einigen Personentagen umgesetzt werden.

Wie in Kapitel 1.2 beschrieben, ist es notwendig, ein bidirektionales Vertrauensverhältnis zwischen der Plattform und dem Benutzer herzustellen, d.h. sowohl die Plattform muss authentifiziert werden als auch der Benutzer.

Um sicherzustellen, dass der Benutzer sein Passwort nicht in eine manipulierte Maschine eingibt, muss zunächst eine Plattform-Authentifikation durchgeführt werden, um sicher zu gehen, dass diese sich in einem definierten Zustand befindet. Hierzu gibt es 3 Möglichkeiten:

- Booten von einem externen Medium, welches sich immer unter der Kontrolle des Benutzers befindet (z.B. USB-Stick)
- Secure Boot
- Trusted Boot

Für den Fall, dass Trusted Boot durchgeführt wird, muss eine Verifikation der gestarteten Konfiguration durch eine externe Entität erfolgen. Dies kann sich bei einer verfügbaren Netzwerkverbindung um einen Server handeln, welcher die Konfiguration verifiziert. Für den Online-Fall werden die folgenden drei Komponenten definiert:

- **File-Server:**
  Auf dem File-Server können z.B. signierte Boot-Images liegen.
- **Attestation-Server:**
  Der Attestation-Server verifiziert die von einer Plattform übermittelte Konfiguration (d.h. die PCR-Werte sowie die TPM-Signatur).

- **Management-Server:**
  Der Management-Server stellt Entschlüsselungsschlüssel zur Verfügung.

Falls keine Netzwerkverbindung zur Verfügung steht, muss eine andere Komponente die Funktiones des Attestation-Server sowie des Management-Servers übernehmen, z.B. eine Smartcard oder ein Smartphone. Wichtig ist, dass diese über einen separaten Ausgabekanal zum Benutzer verfügen, um diesen sicher über das Verifikationsergebnis zu informieren (wie z.B. ein Display).

Als letzte Variante kann man die Plattform-Integrität noch implizit nachweisen, indem man Daten an die vertrauenswürdige, bekannte Ziel-Konfiguration bindet (via Sealing) oder einen Bereich im NVRAM anlegt, welcher nur unter dieser gültigen Ziel-Konfiguration les- bzw. schreibbar ist.

Ist die Plattformkonfiguration sichergestellt, muss sich im nächsten Schritt der Benutzer authentifizieren. Dies kann er dann entweder an der verifizierten Plattform tun (z.B. mittels Passworteingabe) oder aber er kann ebenfalls einen sicheren, externen Kanal nutzen, wie z.B. eine Smartcard mit Klasse-3-Leser, an der er seine PIN eingibt.

Abbildung 2 zeigt zunächst die möglichen Abläufe auf, bevor die einzelnen Lösungsmöglichkeiten in den Folgekapiteln weiter spezifiziert werden.

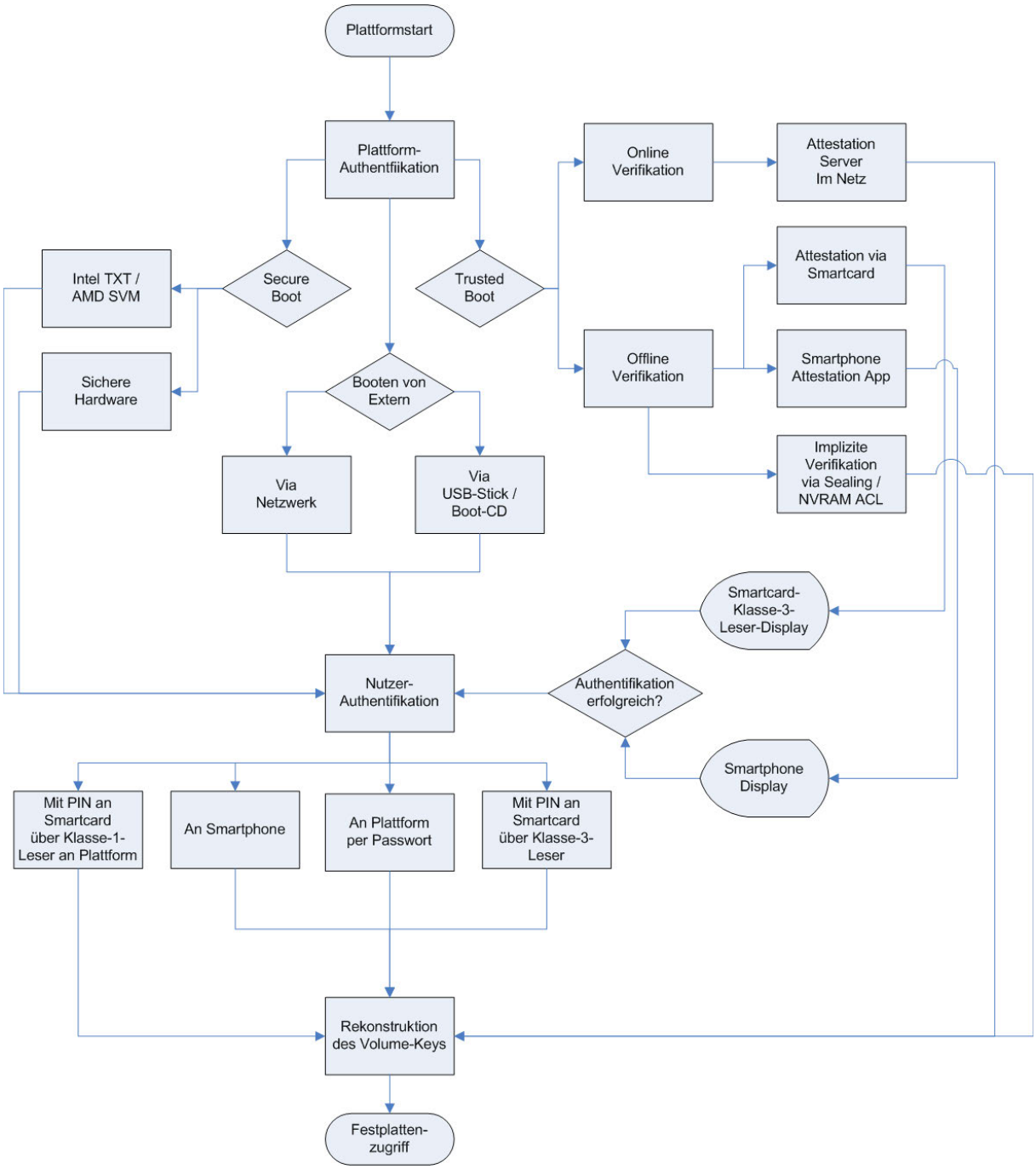


*Abbildung 2: Mögliche Pfade zum sicheren Starten einer Plattform*

Darüber hinaus kommen verschiedene Lösungsansätze in Frage, welche zusätzliche Hardware oder Plattform-Erweiterung benötigen. Konkret sind dies Smartcards zur Authentisierung von Benutzern sowie der Einsatz von *Trusted Computing* (TC) zur Authentisierung von Plattformen. TC benötigt zwingend eine C*hain of Trust*, um die Software vor dem Starten zu messen und zu verifizieren. Hier wird zusätzlich noch folgende Unterscheidung getroffen:

- **Trusted Boot**:
  - benötigt ein TPM der Version 1.1b oder 1.2
  - benötigt ein Static RTM (SRTM/CRTM) inkl. TCG-BIOS
  - benötigt einen Bootloader, der um TCG-Befehle so erweitert wurde, dass Daten vor der Ausführung gemessen werden können (z.B. TrustedGRUB[7], GRUB-IMA[8]). Eine Erweiterung des TrueCrypt-Bootloaders um TCG-Messungen ist ebenfalls denkbar.
- **Secure Boot:**
  - benötigt ein TPM der Version 1.2
  - benötigt ein Dynamic RTM (DRTM)
  - benötigt die Hardwareerweiterungen Intel-TXT oder AMD-SVM
  - benötigt einen sicheren Bootloader, um dynamisch in den sicheren Modus zu wechseln (z.B. tboot[9], Oslo[10]). Eine Erweiterung des TrueCrypt-Bootloaders um die neuen Prozessor- und TCG-Befehle ist ebenfalls denkbar.

In den folgenden Lösungen werden zusätzlich die Angriffspfade aus AP3 aufgezeigt, welche durch die jeweilige Lösung ausgeschlossen werden können. Sämtliche nicht genannten Pfade behalten jedoch weiterhin Ihre Gültigkeit. Ebenso ist es wichtig zu erwähnen, dass die Annahmen aus AP3 weiterhin gelten, vor allem diejenigen, die explizit den Angriffen in Kapitel 3 zugeordnet wurden.

Die folgenden Angriffspfade können während der folgenden Lösungen (ganz oder teilweise) detektiert bzw. eliminiert werden.

- B 3.1.1 Header beschädigen
- B 4.2.1.1 Schlüssel aus Volume Header erlangen
- B 6.1.1. Passworteingabe ausspähen
- B 9.1.1.3.1 Bootloader manipulieren
- B 9.1.1.3.1.1.1 Booten von externem Medium
- B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).

7 http://sourceforge.net/projects/trustedgrub/
8 http://sourceforge.net/projects/trousers/
9 http://sourceforge.net/projects/tboot/
10 http://os.inf.tu-dresden.de/~kauer/oslo/

- B 9.1.1.3.1.1.2 Booten von Cardbus
- B 9.1.1.3.3.1.1 Hardware komplett austauschen
- B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)
- B 9.1.1.3.3.2.1 Firmwareveränderungen
- B 9.1.1.3.3.2.1.1.1 Eigenes System booten
- B 9.1.2.1.2.1.1 Partition hinzufügen
- B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle
- B 9.1.2.2.1 Booten über Netzwerk
- B 9.2.2.2.1.1.1 Booten von extern
- B 9.2.2.2.1.1.2 Austausch der Festplatte
- B 13.1.1.1.1 Manipulation am Binärcode

## 4.1 Lösungsansätze ohne Trusted Computing

### 4.1.1 M1.1 - Booten von externem Medium

| | |
|---|---|
| ***Beschreibung*** | Eine FDE-verschlüsselte Festplatte wird von einem externen Boot-Medium gebootet |
| ***Voraussetzungen*** | [X] Externes Bootmedium<br>[X] USB-Stick<br>[X] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[ ] TPM<br>[ ] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[ ] Sealing<br>[ ] NVRAM<br><br>[ ] Smartcard<br>[ ] Smartcard-Leser mit PinPad<br>[ ] Smartcard-Leser mit Display<br><br>[ ] Netzwerkzugriff<br>[ ] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[ ] Sonstiges: _____ |
| ***Annahmen*** | • Bootmedium befindet sich immer unter der Kontrolle des Benutzers *(organisatorische Maßnahme).*<br>• Kein Volume-Header auf Festplatte *(administrative Maßnahme).* |
| ***Aufwand*** | [ ] Hoch<br>[ ] Mittel<br>[X] Gering<br>[X] Zentrales Rollout möglich<br>[X] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz<br>[ ] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz<br>[X] Schutz von VS an mobilem VS-Arbeitsplatz<br>[X] Schutz inaktiver VS bei Verbringung<br>[X] Schutz der VS bei Abhandenkommen / Diebstahl<br>[X] Sicheres Löschen von Datenträgern mit VS |
| ***Vorgehensweise*** | 1. Administrator installiert Bootloader und Volume-Header auf externes Medium. |

| | |
|---|---|
| | 2. Administrator entfernt Bootloader und Volume-Header von FDE-Disk.<br>3. Administrator konfiguriert BIOS der Plattform so, dass nur von diesem Medium gestartet werden darf.<br>4. Benutzer legt Boot-Medium ein (z.B. Boot-CD / USB-Stick).<br>5. Benutzer startet Plattform.<br>6. Bootloader fragt Benutzer nach seinem Passwort.<br>7. Bootloader benutzt Volume-Header aus Boot-Medium sowie das eingegebene Passwort zum Entschlüsseln der Festplatte.<br><br>Zur Verbesserung dieser Vorgehensweise kann der Benutzer beispielsweise auf USB-Sticks zurückgreifen, welche zunächst per Fingerabdruck freigeschaltet werden müssen. |
| ***Ausgeschlossene Angriffspfade*** | • B 3.1.1 Header beschädigen<br>• B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 13.1.1.1.1 Manipulation am Binärcode |
| ***Restrisiken*** | • B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte |

### 4.1.2 M1.2 - Booten vom Netzwerk

| | |
|---|---|
| ***Beschreibung*** | Eine FDE-verschlüsselte Festplatte wird über das Netzwerk gebootet |
| ***Voraussetzungen*** | [X] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[X] Netzwerk (PXE-Boot)<br><br>[ ] TPM<br>[ ] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[ ] Sealing<br>[ ] NVRAM<br><br>[ ] Smartcard<br>[ ] Smartcard-Leser mit PinPad<br>[ ] Smartcard-Leser mit Display<br><br>[X] Netzwerkzugriff<br>[X] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[ ] Sonstiges: _____ |
| ***Annahmen*** | • Boot-Image befindet sich auf dem File-Server der Organisation.<br>• Kein Volume-Header auf Festplatte *(administrative Maßnahme).*<br>• Die Netzwerkkarte, insbesondere die MAC-Adresse, kann nicht manipuliert werden.<br>• Die Netzverbindung muss physikalisch gesichert sein.<br>• Reine Intranet-Lösung in statischer Umgebung. |
| ***Aufwand*** | [ ] Hoch<br>[ ] Mittel<br>[X] Gering<br>[X] Zentrales Rollout möglich<br>[X] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz<br>[X] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz<br>[ ] Schutz von VS an mobilem VS-Arbeitsplatz<br>[ ] Schutz inaktiver VS bei Verbringung<br>[X] Schutz der VS bei Abhandenkommen / Diebstahl<br>[X] Sicheres Löschen von Datenträgern mit VS |

| | |
|---|---|
| ***Vorgehensweise*** | 1. Administrator erstellt PXE-Bootimage inkl. Bootloader und Volume-Header.<br>2. Administrator legt Boot-Image auf File-Server ab.<br>3. Administrator konfiguriert Plattform so, dass nur per Netzwerk gebootet werden darf.<br>4. Administrator konfiguriert Netzwerk so, dass ein Rechner nur direkt mit dem File-Server kommunizieren darf (z.B. über Routing-Regeln, MAC-Filter. Ein Zugriff von außen auf den File-Server wird unterbunden. Der File-Server ist so konfiguriert, dass er einer anfragenden MAC-Adresse nur das exakt zugewiesene Image zukommen lässt.<br>5. Benutzer startet Plattform.<br>6. Netzwerkkarte lädt Boot-Image von File-Server.<br>7. Boot-Image startet Bootloader.<br>8. Bootloader fragt Benutzer nach seinem Passwort.<br>9. Bootloader benutzt Volume-Header aus Boot-Medium sowie das eingegebene Passwort zum Entschlüsseln der Festplatte. |
| ***Ausgeschlossene Angriffspfade*** | • B 3.1.1 Header beschädigen<br>• B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle |
| ***Restrisiken*** | • B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte<br>• B 13.1.1.1.1 Manipulation am Binärcode |

### 4.1.3 M1.3 - Messung des Speichers mittels DMA

### 4.1.4

| | |
|---|---|
| ***Beschreibung*** | Der Arbeitsspeicher des PC wird durch eine vertrauenswürdige PCI-Express-Karte überwacht. Der Arbeitsspeicher wird mittels DMA ausgelesen und mit Referenzwerten verglichen. |
| ***Voraussetzungen*** | [ ] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[ ] TPM<br>[ ] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[ ] Sealing<br>[ ] NVRAM<br><br>[ ] Smartcard<br>[ ] Smartcard-Leser mit PinPad<br>[ ] Smartcard-Leser mit Display<br><br>[ ] Netzwerkzugriff<br>[ ] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[X] Sonstiges: PCI-Express-Karte |
| ***Annahmen*** | • Eine (neu zu entwickelnde) PCI-Express-Karte, welche Bootloader, Volume-Header und Schlüssel in einem sicheren Speicher bereithält.<br>• Bootloader und Volume-Header sind auf Festplatte existent. |
| ***Aufwand*** | [X] Hoch<br>[ ] Mittel<br>[ ] Gering<br>[ ] Zentrales Rollout möglich<br>[ ] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz<br>[X] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz<br>[X] Schutz von VS an mobilem VS-Arbeitsplatz<br>[ ] Schutz inaktiver VS bei Verbringung<br>[ ] Schutz der VS bei Abhandenkommen / Diebstahl<br>[ ] Sicheres Löschen von Datenträgern mit VS |
| ***Vorgehensweise*** | 1. Die Plattform startet, bis der Bootloader den Benutzer nach seinem Passwort fragt. |

| | |
|---|---|
| | 2. Die PCI-Express-Karte hat mittels DMA Zugriff auf sämtliche Speicherbereiche des PC.<br>3. Sie überwacht durch ihre Firmware den Arbeitsspeicher zum Bootzeitpunkt und vergleicht die Speicherinhalte von Bootloader und Volume-Header.<br>4. Stimmen die im Speicher befindlichen Daten mit den gespeicherten Referenzwerten überein, kann die Karte den Volume-Schlüssel bereitstellen.<br>5. Die Plattform startet. |
| ***Ausgeschlossene Angriffspfade*** | • B 3.1.1 Header beschädigen<br>• B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 13.1.1.1.1 Manipulation am Binärcode |
| ***Restrisiken*** | • B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte |

## 4.2 Online-Lösungen mit Trusted Computing

### 4.2.1 M2.1 - Remote Attestation mit Schlüsselversand

| | |
|---|---|
| ***Beschreibung*** | Eine Plattform attestiert seine Konfiguration einem Server übers Netzwerk. Stimmt die Konfiguration, und ist diese als gültig und vertrauenswürdig eingestuft, übermittelt ein Server den notwendigen Volume-Key an die Plattform. |
| ***Voraussetzungen*** | [ ] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[X] TPM<br>[X] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[ ] Sealing<br>[ ] NVRAM<br><br>[ ] Smartcard<br>[ ] Smartcard-Leser mit PinPad<br>[ ] Smartcard-Leser mit Display<br><br>[X] Netzwerkzugriff<br>[ ] zum File-Server<br>[X] zum Attestation-Server<br>[X] zum Management-Server<br><br>[ ] Sonstiges: _____ |
| ***Annahmen*** | • TPM ist initialisiert.<br>• TPM hat TPM-Signaturschlüssel.<br>• Attestation-Server besitzt öffentlichen Teil des Signaturschlüssels.<br>• Attestation-Server besitzt Referenz-PCR-Werte. |
| ***Aufwand*** | [ ] Hoch<br>[X] Mittel<br>[ ] Gering<br>[X] Zentrales Rollout möglich<br>[X] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz<br>[X] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz<br>[X] Schutz von VS an mobilem VS-Arbeitsplatz<br>[X] Schutz inaktiver VS bei Verbringung<br>[X] Schutz der VS bei Abhandenkommen / Diebstahl |

| | |
|---|---|
| | [X] Sicheres Löschen von Datenträgern mit VS |
| ***Vorgehensweise*** | 1. Die Plattform führt Trusted Boot durch.<br>2. Der Bootloader (oder ein Mini-Linux) stellt eine sichere VPN-Verbindung zum Attestation-Server her.<br>3. Der Attestation-Server sendet eine Challenge (*Nonce*) und fordert die vom TPM signierten PCR-Werte an (*TPM_Quote*).<br>4. Das TPM signiert die Challenge sowie seine PCR-Werte mit dem Signaturschlüssel.<br>5. Die Plattform sendet die TPM-Antwort an den Attestation-Server.<br>6. Der Attestation-Server verifiziert zunächst die Signatur mit dem gespeicherten öffentlichen TPM-Signaturschlüssel. Dann verifiziert er, ob die empfangene Challenge in der Signatur mit der gesendeten übereinstimmt (*Anti-Replay*). Zuletzt überprüft der Attestation-Server, ob die signierten PCR-Werte einer gültigen Konfiguration entsprechen.<br>7. Falls ja, beauftragt der Attestion-Server den Management-Server damit, den Volume-Key an die Plattform zu übertragen.<br>8. Die Plattform entschlüsselt die Festplatte und startet. |
| ***Anmerkung*** | Diese Lösung beinhaltet noch **keine** Benutzerauthentifikation, stimmt die Konfiguration der Plattform, kann der aktuelle Besitzer die Maschine starten. Eine Verfeinerung dieses Szenarios erfolgt in Kapitel 4.3.3, 4.3.5, 4.3.6 und 4.3.7. |
| ***Ausgeschlossene Angriffspfade*** | • B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk |

| | |
|---|---|
| | • B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte<br>• B 13.1.1.1.1 Manipulation am Binärcode |
| ***Restrisiken*** | • B 3.1.1 Header beschädigen |

## 4.3 Offline-Lösungen mit Trusted Computing

### 4.3.1 M3.1 - Sealing des Volume-Key

| | |
|---|---|
| ***Beschreibung*** | Eine Plattform versiegelt den Entschlüsselungsschlüssel mit Hilfe des TPM und bindet diesen so an die Plattform und an eine bestimmte Konfiguration. Nur wenn die Konfiguration integer ist, ist der Volume-Key entschlüsselbar. |
| ***Voraussetzungen*** | [ ] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[X] TPM<br>[X] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[X] Sealing<br>[ ] NVRAM<br><br>[ ] Smartcard<br>[ ] Smartcard-Leser mit PinPad<br>[ ] Smartcard-Leser mit Display<br><br>[ ] Netzwerkzugriff<br>[ ] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[ ] Sonstiges: _____ |
| ***Annahmen*** | • TPM ist initialisiert.<br>• TPM hat TPM-Speicherschlüssel.<br>• Benutzer verwendet **kein** Passwort.<br>• Passwort bzw. Keyfile wird mittels der Sealing-Funktionalität des TPM an die Bootloader-Konfiguration und das BIOS gebunden. |
| ***Aufwand*** | [ ] Hoch<br>[X] Mittel<br>[ ] Gering<br>[ ] Zentrales Rollout möglich<br>[X] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz<br>[ ] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz<br>[X] Schutz von VS an mobilem VS-Arbeitsplatz<br>[ ] Schutz inaktiver VS bei Verbringung |

| | |
|---|---|
| | [ ] Schutz der VS bei Abhandenkommen / Diebstahl<br>[X] Sicheres Löschen von Datenträgern mit VS |
| ***Vorgehensweise*** | 1. Die Plattform führt Trusted Boot durch.<br>2. Der Bootloader versucht, den Volume-Key zu *unseal*en.<br>3. Stimmt die aktuelle Konfiguration mit der erwarteteten Konfiguration überein, ist der *Unseal*-Vorgang erfolgreich.<br>4. Der Bootloader entschlüsselt die Festplatte.<br>5. Die Plattform startet. |
| ***Anmerkung*** | Diese Lösung beinhaltet noch **keine** Benutzerauthentifikation, stimmt die Konfiguration der Plattform, kann der aktuelle Besitzer die Maschine starten. Eine Verfeinerung dieses Szenarios erfolgt in Kapitel 4.3.3, 4.3.5, 4.3.6 und 4.3.7. |
| ***Ausgeschlossene Angriffspfade*** | • B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte<br>• B 13.1.1.1.1 Manipulation am Binärcode |
| ***Restrisiken*** | • B 3.1.1 Header beschädigen |

### 4.3.2 M3.2 - Ablage des Volume-Schlüssels in NVRAM des TPM

| | |
|---|---|
| ***Beschreibung*** | Eine Plattform speichert den Entschlüsselungsschlüssel mit Hilfe des TPM in einer sicheren Umgebung so ab, dass dieser nur unter einer bestimmten Konfiguration auslesbar ist. |
| ***Voraussetzungen*** | [ ] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[X] TPM<br>[X] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[ ] Sealing<br>[X] NVRAM<br><br>[ ] Smartcard<br>[ ] Smartcard-Leser mit PinPad<br>[ ] Smartcard-Leser mit Display<br><br>[ ] Netzwerkzugriff<br>[ ] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[ ] Sonstiges: _____ |
| ***Annahmen*** | • TPM ist initialisiert.<br>• TPM hat NVRAM-Bereich unter Zielkonfiguration angelegt.<br>• Benutzer verwendet **kein** Passwort.<br>• Passwort bzw. Keyfile wird sicher im NVRAM des TPM abgelegt. Das NVRAM wird hingegen so konfiguriert, dass es nur unter der gültigen Bootloader+BIOS-Konfiguration lesbar ist. |
| ***Aufwand*** | [ ] Hoch<br>[X] Mittel<br>[ ] Gering<br>[ ] Zentrales Rollout möglich<br>[X] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz<br>[ ] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz<br>[X] Schutz von VS an mobilem VS-Arbeitsplatz<br>[ ] Schutz inaktiver VS bei Verbringung<br>[ ] Schutz der VS bei Abhandenkommen / Diebstahl<br>[X] Sicheres Löschen von Datenträgern mit VS |

| | |
|---|---|
| ***Vorgehensweise*** | 1. Die Plattform führt Trusted Boot durch.<br>2. Der Bootloader versucht, den Volume-Key aus dem NVRAM auszulesen.<br>3. Stimmt die aktuelle Konfiguration mit der erwarteten Konfiguration überein, ist ein Lese-Vorgang auf dem NVRAM möglich.<br>4. Der Bootloader entschlüsselt die Festplatte.<br>5. Die Plattform startet. |
| ***Anmerkung*** | Diese Lösung beinhaltet noch **keine** Benutzerauthentifikation, stimmt die Konfiguration der Plattform, kann der aktuelle Besitzer die Maschine starten. Eine Verfeinerung dieses Szenarios erfolgt in Kapitel 4.3.3, 4.3.5, 4.3.6 und 4.3.7. |
| ***Ausgeschlossene Angriffspfade*** | • B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte<br>• B 13.1.1.1.1 Manipulation am Binärcode |
| ***Restrisiken*** | • B 3.1.1 Header beschädigen |

### 4.3.3 M3.3 - Lokale „Remote“ Attestation via Smartcard

| | |
|---|---|
| ***Beschreibung*** | Eine Plattform attestiert sich gegenüber einer Smartcard und beweist dieser seine korrekte Konfiguration. Ist die Verifikation durch die Smartcard erfolgreich, übermittelt die Smartcard den benötigten Volume-Key an die Plattform. |
| ***Voraussetzungen*** | [ ] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[X] TPM<br>[X] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[X] Sealing<br>[ ] NVRAM<br><br>[X] Smartcard<br>[ ] Smartcard-Leser mit PinPad<br>[ ] Smartcard-Leser mit Display<br><br>[ ] Netzwerkzugriff<br>[ ] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[ ] Sonstiges: _____ |
| ***Annahmen*** | • TPM ist initialisiert.<br>• TPM hat TPM-Signaturschlüssel.<br>• Smartcard fungiert als lokaler Attestation-Server.<br>• Smartcard besitzt öffentlichen Teil des TPM-Signaturschlüssels.<br>• Smartcard besitzt Referenz-PCR-Werte.<br>• Smartcard besitzt Volume-Key. |
| ***Aufwand*** | [ ] Hoch<br>[X] Mittel<br>[ ] Gering<br>[X] Zentrales Rollout möglich<br>[X] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz<br>[X] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz<br>[X] Schutz von VS an mobilem VS-Arbeitsplatz<br>[X] Schutz inaktiver VS bei Verbringung<br>[X] Schutz der VS bei Abhandenkommen / Diebstahl<br>[X] Sicheres Löschen von Datenträgern mit VS |

| | |
|---|---|
| ***Vorgehensweise*** | 1. Die Plattform führt Trusted Boot durch.<br>2. Der Bootloader (oder ein Mini-Linux) stellt eine Verbindung zur Smartcard her.<br>3. Die Smartcard sendet eine Challenge (*Nonce*) und fordert die vom TPM signierten PCR-Werte an (*TPM_Quote*).<br>4. Das TPM signiert die Challenge sowie seine PCR-Werte mit dem TPM-Signaturschlüssel.<br>5. Die Plattform sendet die TPM-Antwort an die Smartcard.<br>6. Die Smartcard verifiziert zunächst die Signatur mit dem gespeicherten öffentlichen TPM-Signaturschlüssel. Dann verifiziert sie, ob die empfangene Challenge in der Signatur mit der gesendeten übereinstimmt (Anti-Replay). Zuletzt überprüft die Smartcard, ob die signierten PCR-Werte einer gültigen Konfiguration entsprechen.<br>7. Falls ja, sendet die Smartcard den Volume-Key an die Plattform.<br>8. Die Plattform entschlüsselt die Festplatte und startet. |
| ***Anmerkung*** | Diese Lösung ist mit den Lösungen aus Kapitel 4.3.1 und 4.3.2 kombinierbar, wenn der Volume-Key in zwei Hälften aufgeteilt, wobei eine Hälfte im TPM und die zweite Hälfte auf der Smartcard gespeichert wird. Auf diese Weise kann zusätzlich noch eine Benutzerauthentifikation per PIN an der Smartcard erreicht werden (siehe 4.3.8). |
| ***Ausgeschlossene Angriffspfade*** | • B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte |

| | |
|---|---|
| | • B 13.1.1.1.1 Manipulation am Binärcode |
| ***Restrisiken*** | • B 3.1.1 Header beschädigen |

### 4.3.4 M3.4 - Gegenseitige Attestierung über ein „Trusted Device“, z.B. via Smartphone

| | |
|---|---|
| ***Beschreibung*** | Eine Plattform attestiert sich gegenüber einem vertrauenswürdigen Gerät, z.B. ein Smartphone. Konkret wird eine Attestierungssoftware auf dem Smartphone ausgeführt, welcher die Plattform seine korrekte Konfiguration beweist. Ist die Verifikation erfolgreich, informiert das Smartphone den Benutzer über den Vertrauenszustand der Plattform, dieser kann dann dort sein zum Entschlüsseln des Volume-Key benötigtes Passwort eingeben. |
| ***Voraussetzungen*** | [ ] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[X] TPM<br>[X] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[X] Sealing<br>[ ] NVRAM<br><br>[ ] Smartcard<br>[ ] Smartcard-Leser mit PinPad<br>[ ] Smartcard-Leser mit Display<br><br>[ ] Netzwerkzugriff<br>[ ] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[X] Sonstiges: _Smartphone mit Attestierungssoftware_____ |
| ***Annahmen*** | • TPM ist initialisiert.<br>• TPM hat TPM-Signaturschlüssel.<br>• Smartphone fungiert als lokaler Attestation-Server.<br>• Smartphone besitzt öffentlichen Teil des TPM-Signaturschlüssels.<br>• Smartphone besitzt Referenz-PCR-Werte. |
| ***Aufwand*** | [ ] Hoch<br>[X] Mittel<br>[ ] Gering<br>[X] Zentrales Rollout möglich<br>[X] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz<br>[X] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz |

| | |
|---|---|
| | [X] Schutz von VS an mobilem VS-Arbeitsplatz<br>[X] Schutz inaktiver VS bei Verbringung<br>[X] Schutz der VS bei Abhandenkommen / Diebstahl<br>[X] Sicheres Löschen von Datenträgern mit VS |
| ***Vorgehensweise*** | 1. Die Plattform führt Trusted Boot durch.<br>2. Der Bootloader fordert den Benutzer auf, eine Challenge einzugeben.<br>3. Der Benutzer startet seine Attestierungssoftware auf dem Smartphone. Diese generiert eine Challenge (welche einfach über die Tastatur einzugeben ist, z.B. eine 6-stellige alphanumerische PIN).<br>4. Der Benutzer gibt die PIN ein.<br>5. Der Bootloader hasht die PIN und nutzt diese als *Nonce* für das *TPM_Quote*-Kommando. Der Bootloader fordert die vom TPM signierten PCR-Werte an.<br>6. Der Bootloader generiert einen Barcode von der TPM-Antwort und zeigt diesen auf dem Bildschirm an.<br>7. Der Benutzer lässt den Barcode durch die Attestierungssoftware scannen. Intern wertet die Attestierungssoftware dann die Ergebnisse der Signaturprüfung, PCR-Prüfung und Nonce-Prüfung aus. Sind alle Werte korrekt, zeigt das Smartphone dem Benutzer eine entsprechende Nachricht an.<br>8. Der Bootloader fragt nach dem Passwort zum Entschlüsseln des Volume-Key.<br>9. Der Benutzer gibt sein Passwort ein.<br>10. Die Plattform entschlüsselt die Festplatte und startet. |
| ***Anmerkung*** | Diese Lösung lässt sich noch wie folgt erweitern:<br>• Erweiterung um 4.3.1 bzw. 4.3.2 möglich.<br>• Verfügt das Smartphone über eine Online-Anbindung, kann auch eine Online-Überprüfung der Verifikationsergebnisse erfolgen.<br>• Alternativ kann das Benutzerpasswort auch in das Smartphone eingegeben werden, falls ein Datenaustausch vom Smartphone zur Plattform hergestellt werden kann. |
| ***Ausgeschlossene Angriffspfade*** | • B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen |

| | |
|---|---|
| | • B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte<br>• B 13.1.1.1.1 Manipulation am Binärcode |
| ***Restrisiken*** | • B 3.1.1 Header beschädigen |

### 4.3.5 M3.5 - Gegenseitige Attestierung via Smartcard und Klasse-3-Leser (i)

| | |
|---|---|
| ***Beschreibung*** | Eine Plattform und ein Benutzer attestieren sich gegenseitig. Zunächst beweist die Plattform dem Benutzer, dass sie in einem korrekten Zustand ist. Dann attestiert sich der Benutzer durch seine PIN an der Smartcard. Die Smartcard übermittelt bei Korrektheit den benötigten Volume-Key an die Plattform. |
| ***Voraussetzungen*** | [ ] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[X] TPM<br>[X] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[X] Sealing<br>[ ] NVRAM<br><br>[X] Smartcard<br>[X] Smartcard-Leser mit PinPad<br>[X] Smartcard-Leser mit Display<br><br>[ ] Netzwerkzugriff<br>[ ] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[ ] Sonstiges: _____ |
| ***Annahmen*** | • TPM ist initialisiert.<br>• TPM hat TPM-Speicherschlüssel.<br>• Smartcard hat ein Benutzergeheimnis signiert.<br>• Smartcard besitzt Volume-Key (geschützt durch Benutzer-PIN).<br>• Benutzer legt während der Installation ein nur ihm bekanntes Geheimnis fest.<br>• Dieses lässt er durch die Smartcard signieren.<br>• Während der Installation der FDE wird das signierte Geheimnis an die Plattform und die Bootkonfiguration *gesealt*. |
| ***Aufwand*** | [ ] Hoch<br>[X] Mittel<br>[ ] Gering<br>[X] Zentrales Rollout möglich<br>[X] Zentrales Management möglich |
| ***Passende*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz |

| | |
|---|---|
| ***Anwendungsszenarien*** | [X] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz<br>[X] Schutz von VS an mobilem VS-Arbeitsplatz<br>[X] Schutz inaktiver VS bei Verbringung<br>[X] Schutz der VS bei Abhandenkommen / Diebstahl<br>[X] Sicheres Löschen von Datenträgern mit VS |
| ***Vorgehensweise*** | 1. Die Plattform führt Trusted Boot durch.<br>2. Der Bootloader versucht, das Benutzergeheimnis zu *unsealen*.<br>3. Ist die Konfiguration in Ordnung, so wird das signierte Benutzergeheimnis an die Smartcard gesandt.<br>4. Die Smartcard verifiziert die eigene Signatur.<br>5. Stimmt die Signatur, zeigt die Smartcard die Nachricht auf dem Display des Klasse-3-Lesers an.<br>6. Der Benutzer überprüft, ob die angezeigte Nachricht dem von ihm eingangs festgelegten Passwort entspricht.<br>7. Ist es korrekt, gibt der Benutzer seine PIN am Klasse-3-Leser ein und entschlüsselt so den in der Smartcard gespeicherten Volume-Key.<br>8. Die Plattform startet. |
| ***Anmerkung*** | Diese Lösung ist mit den Lösungen aus Kapitel 4.3.1 und 4.3.2 kombinierbar, wenn der Volume-Key in zwei Hälften aufgeteilt wird, wobei eine Hälfte im TPM und die zweite Hälfte auf der Smartcard gespeichert wird (siehe 4.3.8). |
| ***Ausgeschlossene Angriffspfade*** | • B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern |

| | |
|---|---|
| | • B 9.2.2.2.1.1.2 Austausch der Festplatte<br>• B 13.1.1.1.1 Manipulation am Binärcode |
| ***Restrisiken*** | • B 3.1.1 Header beschädigen |

### 4.3.6 M3.6 - Gegenseitige Attestierung via Smartcard und Klasse-3-Leser (ii)

| | |
|---|---|
| ***Beschreibung*** | Eine Plattform und ein Benutzer attestieren sich gegenseitig. Zunächst beweist die Plattform dem Benutzer, dass sie in einem korrekten Zustand ist. Dann attestiert sich der Benutzer durch seine PIN an der Smartcard. Die Smartcard übermittelt bei Korrektheit den benötigten Volume-Key an die Plattform. |
| ***Voraussetzungen*** | [ ] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[X] TPM<br>[X] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[ ] Sealing<br>[X] NVRAM<br><br>[X] Smartcard<br>[X] Smartcard-Leser mit PinPad<br>[X] Smartcard-Leser mit Display<br><br>[ ] Netzwerkzugriff<br>[ ] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[ ] Sonstiges: _____ |
| ***Annahmen*** | • TPM ist initialisiert.<br>• TPM hat NVRAM-Bereich unter Zielkonfiguration angelegt.<br>• Smartcard hat ein Benutzergeheimnis signiert.<br>• Smartcard besitzt Volume-Key (geschützt durch Benutzer-PIN).<br>• Benutzer legt während der Installation ein nur ihm bekanntes Geheimnis fest.<br>• Dieses lässt er durch die Smartcard signieren.<br>• Während der Installation der FDE wird im NVRAM des TPM ein Bereich reserviert, welcher nur unter der korrekten Bootkonfiguration lesbar ist. Dort wird das signierte Benutzergeheimnis abgelegt. |
| ***Aufwand*** | [ ] Hoch<br>[X] Mittel<br>[ ] Gering<br>[X] Zentrales Rollout möglich |

| | |
|---|---|
| | [X] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz<br>[X] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz<br>[X] Schutz von VS an mobilem VS-Arbeitsplatz<br>[X] Schutz inaktiver VS bei Verbringung<br>[X] Schutz der VS bei Abhandenkommen / Diebstahl<br>[X] Sicheres Löschen von Datenträgern mit VS |
| ***Vorgehensweise*** | 1. Die Plattform führt Trusted Boot durch.<br>2. Der Bootloader versucht, das Benutzergeheimnis aus dem NVRAM zu lesen.<br>3. Ist die Konfiguration in Ordnung, so wird das signierte Benutzergeheimnis an die Smartcard gesandt.<br>4. Die Smartcard verifiziert die eigene Signatur.<br>5. Stimmt die Signatur, zeigt die Smartcard die Nachricht auf dem Display des Klasse-3-Lesers an.<br>6. Der Benutzer überprüft, ob die angezeigte Nachricht dem von ihm eingangs festgelegten Passwort entspricht.<br>7. Ist es korrekt, gibt der Benutzer seine PIN am Klasse-3-Leser ein und entschlüsselt so den in der Smartcard gespeicherten Volume-Key.<br>8. Die Plattform startet. |
| ***Anmerkung*** | Diese Lösung ist mit den Lösungen aus Kapitel 4.3.1 und 4.3.2 kombinierbar, wenn der Volume-Key in zwei Hälften aufgeteilt wird, wobei eine Hälfte im TPM und die zweite Hälfte auf der Smartcard gespeichert wird (siehe 4.3.8). |
| ***Ausgeschlossene Angriffspfade*** | • B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk |

| | |
|---|---|
| | • B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte<br>• B 13.1.1.1.1 Manipulation am Binärcode |
| ***Restrisiken*** | • B 3.1.1 Header beschädigen |

### 4.3.7 M3.7 – Mehrstufige, gegenseitige Attestierung durch Bilder

| | |
|---|---|
| ***Beschreibung*** | Eine Plattform und ein Benutzer attestieren sich gegenseitig in einem dreistufigen Verfahren. Zunächst beweist der Benutzer der Plattform, dass der korrekte Benutzer vor der Plattform sitzt. Dann beweist die Plattform dem Benutzer, dass sie in einem korrekten Zustand ist. Ist der Benutzer überzeugt, die korrekte Plattform vor sich zu haben, gibt er sein Passwort zum Entschlüsseln des Volume ein. |
| ***Voraussetzungen*** | [ ] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[X] TPM<br>[X] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[X] Sealing<br>[ ] NVRAM<br><br>[ ] Smartcard<br>[ ] Smartcard-Leser mit PinPad<br>[ ] Smartcard-Leser mit Display<br><br>[ ] Netzwerkzugriff<br>[ ] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[ ] Sonstiges: _____ |
| ***Annahmen*** | • TPM ist initialisiert.<br>• TPM hat TPM-Speicherschlüssel.<br>• TPM-Speicherschlüssel ist mit einem Passwort versehen.<br>• Benutzer hat ein frei gewähltes Bild an die Bootkonfiguration ge*seal*t. |
| ***Aufwand*** | [ ] Hoch<br>[X] Mittel<br>[ ] Gering<br>[X] Zentrales Rollout möglich<br>[X] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz<br>[X] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz<br>[X] Schutz von VS an mobilem VS-Arbeitsplatz<br>[X] Schutz inaktiver VS bei Verbringung<br>[X] Schutz der VS bei Abhandenkommen / Diebstahl |

| | |
|---|---|
| | [X] Sicheres Löschen von Datenträgern mit VS |
| ***Vorgehensweise*** | 1. Die Plattform führt Trusted Boot durch.<br>2. Beim Booten fragt der Bootloader nach dem Passwort für den TPM-Speicherschlüssel.<br>3. Der Benutzer gibt das erste Passwort ein.<br>4. Die Plattform verwendet das Passwort, um den TPM-Speicherschlüssel ins TPM zu laden und das Benutzerbild zu *unseal*en.<br>5. Ist der Vorgang erfolgreich, wird dem Benutzer sein Bild und ein Passwort-Dialog angezeigt.<br>6. Ist der Benutzer durch die Darstellung seines Bildes überzeigt, seine korrekte Plattform vor sich zu haben, gibt er sein Passwort zum Entschlüsseln des Volume-Key ein.<br>7. Die Plattform startet. |
| ***Anmerkung*** | Diese Lösung kann noch erweitert werden, wenn statt nur eines Bildes (was ein Angreifer leicht anschauen und rekonstruieren könnte) mehrere Bilder (z.B. in Form einer 5x5 Matrix, abgespeichert wird. Der Benutzer kann dann beim Starten willkürlich eine Auswahl treffen, welche ihm dann angezeigt wird. |
| ***Ausgeschlossene Angriffspfade*** | • B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte<br>• B 13.1.1.1.1 Manipulation am Binärcode |
| ***Restrisiken*** | • B 3.1.1 Header beschädigen |

### 4.3.8 M3.8 - Sealing des Volume-Key + Smartcard und Klasse-1-Leser

| | |
|---|---|
| ***Beschreibung*** | Der Volume-Key ist in zwei Hälften aufgeteilt. Der erste Teil wird durch die Plattform mit Hilfe des TPM versiegelt und so an die Konfiguration der Plattform gebunden. Der zweite Teil des Volume-Key wird auf einer Smartcard in Form eines Keyfiles abgelegt. |
| ***Voraussetzungen*** | [ ] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[X] TPM<br>[X] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[X] Sealing<br>[ ] NVRAM<br><br>[X] Smartcard<br>[ ] Smartcard-Leser mit PinPad<br>[ ] Smartcard-Leser mit Display<br><br>[ ] Netzwerkzugriff<br>[ ] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[ ] Sonstiges: _____ |
| ***Annahmen*** | • TPM ist initialisiert.<br>• TPM hat TPM-Speicherschlüssel.<br>• Benutzer verwendet **kein** Passwort.<br>• Der Volume-Key wird in zwei Hälften zerteilt.<br>• Volume-Key-1 wird mittels der Sealing-Funktionalität des TPM an die Bootloader-Konfiguration und das BIOS gebunden.<br>• Volume-Key-2 wird in Form eines Keyfiles auf die Smartcard des Benutzers gelegt. Diese ist mit einer PIN geschützt. |
| ***Aufwand*** | [ ] Hoch<br>[X] Mittel<br>[ ] Gering<br>[X] Zentrales Rollout möglich<br>[X] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz<br>[X] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz |

| | |
|---|---|
| | [X] Schutz von VS an mobilem VS-Arbeitsplatz<br>[ ] Schutz inaktiver VS bei Verbringung<br>[X] Schutz der VS bei Abhandenkommen / Diebstahl<br>[X] Sicheres Löschen von Datenträgern mit VS |
| ***Vorgehensweise*** | 1. Die Plattform führt Trusted Boot durch.<br>2. Der Bootloader versucht, den ersten Teil des Volume-Key-1 zu *unseal*en.<br>3. Stimmt die aktuelle Konfiguration mit der erwarteteten Konfiguration überein, ist der *Unseal*-Vorgang erfolgreich.<br>4. Der Bootloader fordert den Benutzer auf, seine Smartcard in den Smartcard-Klasse1-Leser einzulegen.<br>5. Der Bootloader fragt den Benutzer nach seiner PIN. Dieser gibt die PIN über die normale PC-Tastatur ein.<br>6. Der Bootloader sendet die PIN an die Smartcard.<br>7. Stimmt die PIN, sendet die Smartcard das Keyfile, welches den Volume-Key-2 enthält, an den Bootloader.<br>8. Der Bootloader rekonstruiert den benötigten Volume-Key aus dem ungesealten Volume-Key-1 und dem Smartcard-Keyfile Volume-Key-2.<br>9. Bei Erfolg entschlüsselt der Bootloader die Festplatte und die Plattform startet.<br>10. Bei Misserfolg ist eine Kompromittierung der Plattform nicht ausgeschlossen, der Benutzer ist hierüber in Kenntnis zu setzen und er muss aufgefordert werden, seine PIN zu ändern. |
| ***Anmerkung*** | • Bei dieser Lösung hat die Plattform noch nicht seine Integrität gegenüber dem Benutzer bewiesen. Dies erfolgt lediglich implizit, wenn die Plattform nach Punkt 10) startet. Es ist also durchaus möglich, dass ein Angreifer den Bootloader manipuliert, um so an die PIN des Benutzers zu gelangen (vgl. Angriff 3.5).<br>• Beim Umschlüsseln muss auf den Smartcards aller berechtigter Benutzer das Keyfile erneuert werden. Daher wird der Einsatz in Kombination mit einem zentralen Management empfohlen.<br>• Alternativ besteht die Möglichkeit, über eine weitere Indirektion des Volume-Keys eine Aktualisierung der Smartcards zu umgehen. So könnte man z.B. den Volume-Key mit einem Intermediate-Volume-Key verschlüsseln und lediglich diesen Intermediate-Volume-Key in zwei Hälften aufsplitten. Nach einem Umschlüsseln müsste dann lediglich der neue Volume-Key mit dem Intermediate-Volume-Key verschlüsselt werden und eine Entschlüsselung ist auch ohne Update der Smartcards und der Plattform noch möglich. Näheres wird in einem Schlüssel-Management-Konzept erarbeitet. |
| ***Ausgeschlossene Angriffspfade*** | • B 4.2.1.1 Schlüssel aus Volume Header erlangen |

| | |
|---|---|
| | • B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte<br>• B 13.1.1.1.1 Manipulation am Binärcode |
| ***Restrisiken*** | • B 3.1.1 Header beschädigen<br>• B 6.1.1. Passworteingabe ausspähen |

## 4.4 Secure Boot ohne Trusted Computing

### 4.4.1 M4.1 - Sicheres BIOS

| | |
|---|---|
| ***Beschreibung*** | Es wird ein neues BIOS[11] für die Plattform entwickelt, welches die auszuführenden Komponenten zuvor verifiziert. Dazu müssen die Komponenten über eine Signatur von einer vertrauenswürdigen Instanz verfügen. |
| ***Voraussetzungen*** | [ ] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[ ] TPM<br>[ ] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[ ] Sealing<br>[ ] NVRAM<br><br>[ ] Smartcard<br>[ ] Smartcard-Leser mit PinPad<br>[ ] Smartcard-Leser mit Display<br><br>[ ] Netzwerkzugriff<br>[ ] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[X] Sonstiges: Sicheres BIOS |
| ***Annahmen*** | • Sicherer BIOS-Chip mit etwas *Secure Storage.*<br>• Komponenten (z.B. Bootloader) werden durch eine vertrauenswürdige Instanz signiert. Die Signaturen werden an die Komponente angehängt oder parallel dazu gespeichert.<br>• Der zugehörige, öffentliche Verifikationsschlüssel ist im Secure Storage abgelegt.<br>• Können die Signaturen, z.B. vom Bootloader, nicht parallel abgespeichert werden, müssen diese als Referenz mit im Secure Storage abgelegt werden. |
| ***Aufwand*** | [X] Hoch<br>[ ] Mittel<br>[ ] Gering |

11 Vgl. Aegis – Bill Arbaugh: http://www.cis.upenn.edu/~waa/aegis.ps

| | |
|---|---|
| | [ ] Zentrales Rollout möglich<br>[ ] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz<br>[X] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz<br>[X] Schutz von VS an mobilem VS-Arbeitsplatz<br>[ ] Schutz inaktiver VS bei Verbringung<br>[ ] Schutz der VS bei Abhandenkommen / Diebstahl<br>[ ] Sicheres Löschen von Datenträgern mit VS |
| ***Vorgehensweise*** | 1. Das BIOS lädt eine Komponente.<br>2. Das BIOS berechnet einen Hashwert über den geladenen Bereich.<br>3. Das BIOS überprüft, ob die zur Komponente gehörige Signatur gültig ist.<br>4. Das BIOS überprüft, ob der Referenzwert in der Signatur mit dem Gemessenen identisch ist.<br>5. Ist die geladene Software in Ordnung, startet die Plattform. |
| ***Anmerkung*** | Diese Lösung beinhaltet noch **keine** Benutzerauthentifikation. Ebenfalls problematisch sind hier Punkte wie *Update* oder *Revocation.* |
| ***Ausgeschlossene Angriffspfade*** | • B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte<br>• B 13.1.1.1.1 Manipulation am Binärcode |

| | |
|---|---|
| ***Restrisiken*** | • B 3.1.1 Header beschädigen |

## 4.5 Secure Boot mit Trusted Computing

### 4.5.1 M5.1 - Unter Verwendung des DRTM und Intel TXT / AMD SVM

| ***Beschreibung*** | |
|---|---|
| ***Voraussetzungen*** | [ ] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[X] TPM<br>[ ] Trusted Boot (CRTM)<br>[X] Secure Boot (DRTM + Intel TXT)<br>[ ] Sealing<br>[ ] NVRAM<br><br>[ ] Smartcard<br>[ ] Smartcard-Leser mit PinPad<br>[ ] Smartcard-Leser mit Display<br><br>[ ] Netzwerkzugriff<br>[ ] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[ ] Sonstiges: _____ |
| ***Annahmen*** | • Die Plattform wurde nicht komplett ausgetauscht<br>• TPM ist initialisiert.<br>• Chainloading von Secure Boot zu OS ist möglich. |
| ***Aufwand*** | [X] Hoch<br>[ ] Mittel<br>[ ] Gering<br>[X] Zentrales Rollout möglich<br>[X] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [X] Schutz von VS an stationärem VS-Arbeitsplatz<br>[X] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz<br>[X] Schutz von VS an mobilem VS-Arbeitsplatz<br>[X] Schutz inaktiver VS bei Verbringung<br>[X] Schutz der VS bei Abhandenkommen / Diebstahl<br>[X] Sicheres Löschen von Datenträgern mit VS |
| ***Vorgehensweise*** | 1. Die Plattform führt Secure Boot durch.<br>2. Beim Booten überprüft die Hardware die Integrität des Bootloaders und sonstiger Komponenten.<br>3. Der Bootloader fordert den Benutzer auf, sein Passwort |

| | |
|---|---|
| | einzugeben. Diese Meldung würde nicht erscheinen, wenn Secure Boot nicht erfolgreich gewesen wäre.<br>4. Der Benutzer gibt sein Passwort ein.<br>5. Der Bootloader entschlüsselt den Volume-Key und startet das OS. |
| ***Anmerkung*** | Hier kann auch eine Sequenz aus Trusted Boot und Secure Boot erfolgen. Beim Secure Boot können ebenfalls Daten wie Volume-Key im TPM (NVRAM) abgelegt sein oder an eine DRTM-Konfiguration ge*seal*t sein. |
| ***Ausgeschlossene Angriffspfade*** | • B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte<br>• B 13.1.1.1.1 Manipulation am Binärcode |
| ***Restrisiken*** | • B 3.1.1 Header beschädigen |

# 5 Zusammenfassung

In diesem Arbeitspaket wurden zunächst die grundlegenden Bausteine beschrieben, welche zum Schutz des Startvorganges sowie für die Realisierung einer Festplattenverschlüsselung mit Full-Disc-Encryption notwendig und hilfreich sein können.

Darüber hinaus wurden existierende Angriffe beschrieben und untersucht, ob die hier vorgestellten Lösungen gegen die existierenden Angriffe schützen können oder nicht.

In Kapitel 4 werden verschiedene Lösungsmöglichkeiten präsentiert, sowohl Online-Lösungen (d.h. mit Netzwerkanschluss) als auch Offline-Lösungen.

Die vielversprechendsten Lösungen nutzen die neue *Trusted-Computing* -Funktionalität basierend auf einem *Trusted Platform Module* (TPM) und einem *Root of Trust* (CRTM/SRTM/DRTM).

Die wünschenswerteste Lösung sind die *Secure Boot*-Verfahren aus Kapitel 4.4 und 4.5. Diese erfordern jedoch entweder die Entwicklung neuer Hardware oder aber sie setzen auf spezielle Hardware-Erweiterungen wie z.B. Intel's TXT-Technologie. Daher sind diese für einen großflächigen Einsatz in einem existierenden, heterogenen Feld sehr unwahrscheinlich.

Derzeit scheinen die Lösungen, welche auf *Trusted Boot* in Kombination mit der Attestation-Funktionalität am sinnvollsten zu sein. Diese Lösung lässt sich jeweils kombinieren mit:

- *Sealing*: Abspeichern eines Geheimnisses an die Plattform-Konfiguration.
- *NVRAM*: Abspeichern eines Geheimnisses in einem Bereich des Trusted Storage innerhalb des TPM, welches nur unter einer gültigen Konfiguration les- bzw. schreibbar ist.
- *Attestation*: Beweisen der eigenen Plattform-Integrität an eine externe Partei. Als „externe Parteien“ können verschiedene Gegenstellen realisiert werden:
  - *Online*: z.B. ein zentraler Server.
  - *Offline*: z.B. eine Smartcard oder eine Smartphone-Anwendung, welche die Verifikation für den Server übernimmt.

Alle drei Varianten (Sealing, NVRAM, Attestation) sind auf die Korrektheit der PCRs angewiesen.

**Anmerkung:**

Die Lösungen basieren alle auf der Plattform-Konfiguration, d.h. konkret auf den Inhalten der *Platform Configuration Register* (PCRs), welche innerhalb des TPMs gespeichert sind. Wie in Tabelle 1 dargestellt, existieren für unterschiedliche Bereiche eigene Register. So werden z.B. die Bootloader-Messungen in PCRs 4 und 5 gespeichert, während BIOS- und ROM-Messungen in PCRs 0-3 abgespeichert werden. Zusätzliche Messungen (z.B. durch einen modifizierten TrueCrypt-Bootloader oder TrustedGRUB) können PCRs 8-15 nutzen. Je nachdem, welche

PCRs in den Attestation-Vorgang mit einbezogen werden, kann eine Plattform-Überprüfung fehlschlagen, wenn sich z.B. der Bootloader, das BIOS oder die Partitionstabelle, z.B. durch ein Update, geändert hat. Daher sind Management-Komponenten nötig, welche im Falle von Updates dafür Sorge tragen, sowohl die Attestation-Server, Smartcards oder Smartphones auf den neuen Stand zu bringen, als auch die Geheimnisse, welche an die Plattform-Konfiguration gesealt sind, auf den neuen Systemstatus zu aktualisieren. Es ist unabdingbar, die neuen Werte im Vorfeld zu berechnen und vor dem eigentlichen Update die Authentifizierungsmerkmale zu aktualisieren.

## *5.1 Produkte und Dienstleistungen fürs Geschäftsmodell*

Folgende Produkte und dazugehörige Dienstleistungen können im Rahmen des Geschäftsmodells zur Realisierung der Pre-Boot-Lösungen hergestellt und vertrieben werden:

- Eigener TrueCrypt-Bootloader mit folgenden Erweiterungen:
  - Trusted Boot + Attestation-Funktionalität (Signieren / TPM_Quote)

    *oder*
  - Secure Boot

    *und*
  - Anbindung an Smartcard-Leser

    *oder*
  - Barcodegenerator

    *oder*
  - Netzwerkzugriff
- Smartcard-OS mit Attestation-Funktionalität
- Smartphone-Anwendung mit Attestation-Funktionalität
- Secure BIOS, z.B. Linux-BIOS in tamper-resistant Chip
- PCI Express-Karte
- Zentrales Management mit Pairing-Software (SC / TPM)

## *5.2 Empfehlung*

Für die Realisierung von Pre-Boot-Authentication für TrueCrypt empfehlen wir die folgenden Lösungen:

1. M3.3 - Lokale „Remote“ Attestation via Smartcard
2. M3.4 - Gegenseitige Attestierung über ein „Trusted Device“, z.B. via Smartphone
3. M3.5 - Gegenseitige Attestierung via Smartcard und Klasse-3-Leser (i)

4. M3.6 - Gegenseitige Attestierung via Smartcard und Klasse-3-Leser (ii)
5. M3.8 - Sealing des Volume-Key + Smartcard und Klasse-1-Leser

Die erste Lösung hat den Vorteil, dass ein Benutzer nicht in die Verifikation involviert ist, d.h. die Plattform und die Smartcard des Benutzers sind für die Attestierung verantwortlich. Dies lässt eine hohe Benutzerakzeptanz zu, da ein Benutzer lediglich seine Smartcard in sein Gerät stecken muss, um seine Plattform zu starten.

Bei der zweiten Lösung erfolgt eine Attestierung der Plattform-Konfiguration gegenüber einem Smartphone. Ist diese erfolgreich, wird der Benutzer darüber informiert und er kann sein Passwort in den Bootloader eingeben. Dies führt dann zu einer impliziten Authentisierung des Benutzers, da dieser im Besitz des Passwortes ist.

Die nächsten beiden Lösungen haben ebenfalls den Vorteil, dass sich Plattform und Benutzer gegenseitig authentifizieren. Für den Benutzer existiert jedoch eine weit höhere Sicherheit, da er sich an dem Klasse-3-Leser authentifiziert und somit Angriffen wie Keylogger an seiner Plattform vorbeugt. Darüber hinaus wird der Benutzer mit Hilfe des Displays über den Plattform-Zustand informiert. Hinsichtlich des Benutzerkomforts muss der Benutzer lediglich eine PIN eingeben, alles Weitere geschieht automatisch, insbesondere muss der Volume-Key nicht auf der Plattform gespeichert sein.

Die letzte Variante 5) ist eine abgeschwächte Form der Varianten 3) und 4). Es erfolgt ebenfalls eine beidseitige Authentisierung, diese erfolgt jedoch implizit, d.h sobald der Startvorgang erfolgreich ist, weiß man, dass sowohl die Plattform-Integrität korrekt ist, als auch der richtige Benutzer (mit seiner richtigen Smartcard) am System eingeloggt ist. Diese deckt jedoch nicht alle Bedrohungen ab, ist aber leichter umzusetzen.

## 5.3 Vergleich – Bedrohungen, Maßnahmen und Szenarien

Abbildung 3 zeigt einen Vergleich der aufgeführten Lösungen inkl. einer Auflistung der eliminierten Bedrohungen und der passenden Anwendungsszenarien.

| | M1.1 | M1 2 | M1.3 | M2.1 | M3.1 | M3.2 | M3 3 | M3.4 | M3 5 | M3.6 | M3.7 | M3.8 | M4.1 | M5 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| B 3.1 1 | | | | | | | | | | | | | | |
| B 4.2 1.1 | | | | | | | | | | | | | | |
| B 6.1 1 | | | | | | | | | | | | | | |
| B 9.1 1.3.1 | | | | | | | | | | | | | | |
| B 9.1 1.3.1 1.1 | | | | | | | | | | | | | | |
| B 9.1 1.3.1 1.1.3 | | | | | | | | | | | | | | |
| B 9.1 1.3.1 1.2 | | | | | | | | | | | | | | |
| B 9.1 1.3.3 1.1 | | | | | | | | | | | | | | |
| B 9.1 1.3.3 1.2.1 | | | | | | | | | | | | | | |
| B 9.1 1.3.3 2.1 | | | | | | | | | | | | | | |
| B 9.1 1.3.3 2.1.1 1 | | | | | | | | | | | | | | |
| B 9.1 2.1.2 1.1 | | | | | | | | | | | | | | |
| B 9.1 2.1.2 2.2.3 | | | | | | | | | | | | | | |
| B 9.1 2.2.1 | | | | | | | | | | | | | | |
| B 9.2 2.2.1 1.1 | | | | | | | | | | | | | | |
| B 9.2 2.2.1 1.2 | | | | | | | | | | | | | | |
| B 13.1 1.1.1 | | | | | | | | | | | | | | |

| Authentifizierung | 1 → | 1 → | 0 | 1 ← | 1 ← | 1 ← | 1 ← | 2 | 2 | 2 | 2 | (2) | 1 ← | 2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Stat. Arbeitspl. | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| Mehrb. Arbeitspl. | ✘ | ✔ | ✔ | ✔ | ✘ | ✘ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| Mobiler Arbeitspl. | ✔ | ✘ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| Verbringung | ✔ | ✘ | ✘ | ✔ | ✘ | ✘ | ✔ | ✔ | ✔ | ✔ | ✔ | ✘ | ✘ | ✔ |
| Diebstahl | ✔ | ✔ | ✘ | ✔ | ✘ | ✘ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✘ | ✔ |
| Sicheres Löschen | ✔ | ✔ | ✘ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✘ | ✔ |

| | |
|---|---|
| **B** | = Bedrohung |
| **M** | = Maßnahme |
| **0** | = Keine Authentifizierung |
| **1 →** | = Einseitige Authentifizierung, Benutzer ggü. Plattform |
| **1 ←** | = Einseitige Authentifizierung, Plattform ggü. Benutzer |
| **2** | = Beidseitige Authentifizierung |
| **(2)** | = Beidseitige Authentifizierung (implizit, falls Bootvorgang erfolgreich) |
| ✔ | = Maßnahme für Szenario geeignet |
| ✘ | = Maßnahme für Szenario nicht geeignet |
| **grün** | = Bedrohung wird durch Maßnahme vollständig eliminiert |
| **gelb** | = Bedrohung wird durch Maßnahme teilweise eliminiert, mindestens detektiert |
| **rot** | = Bedrohung wird durch Maßnahme nicht eliminiert |
| **türkis** | = nicht zutreffend |

*Abbildung 3: Vergleich Maßnahmen, Bedrohungen und Szenarien*

## Anhang - Vorlage zu Pre-Boot-Lösungen

| | |
|---|---|
| ***Beschreibung*** | |
| ***Voraussetzungen*** | [ ] Externes Bootmedium<br>[ ] USB-Stick<br>[ ] Boot-CD (r/o)<br>[ ] Netzwerk (PXE-Boot)<br><br>[ ] TPM<br>[ ] Trusted Boot (CRTM)<br>[ ] Secure Boot (DRTM + Intel TXT)<br>[ ] Sealing<br>[ ] NVRAM<br><br>[ ] Smartcard<br>[ ] Smartcard-Leser mit PinPad<br>[ ] Smartcard-Leser mit Display<br><br>[ ] Netzwerkzugriff<br>[ ] zum File-Server<br>[ ] zum Attestation-Server<br>[ ] zum Management-Server<br><br>[ ] Sonstiges: _____ |
| ***Annahmen*** | |
| ***Aufwand*** | [ ] Hoch<br>[ ] Mittel<br>[ ] Gering<br>[ ] Zentrales Rollout möglich<br>[ ] Zentrales Management möglich |
| ***Passende Anwendungsszenarien*** | [ ] Schutz von VS an stationärem VS-Arbeitsplatz<br>[ ] Schutz von VS an Mehrbenutzer-VS-Arbeitsplatz<br>[ ] Schutz von VS an mobilem VS-Arbeitsplatz<br>[ ] Schutz inaktiver VS bei Verbringung<br>[ ] Schutz der VS bei Abhandenkommen / Diebstahl<br>[ ] Sicheres Löschen von Datenträgern mit VS |
| ***Vorgehensweise*** | |
| ***Ausgeschlossene Angriffspfade*** | • B 3.1.1 Header beschädigen<br>• B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium |

| | |
|---|---|
| | • B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte<br>• B 13.1.1.1.1 Manipulation am Binärcode |
| ***Restrisiken*** | • B 3.1.1 Header beschädigen<br>• B 4.2.1.1 Schlüssel aus Volume Header erlangen<br>• B 6.1.1. Passworteingabe ausspähen<br>• B 9.1.1.3.1 Bootloader manipulieren<br>• B 9.1.1.3.1.1.1 Booten von externem Medium<br>• B 9.1.1.3.1.1.1.3 BIOS austauschen (nur, falls CRTM nicht Teil des BIOS ist).<br>• B 9.1.1.3.1.1.2 Booten von Cardbus<br>• B 9.1.1.3.3.1.1 Hardware komplett austauschen<br>• B 9.1.1.3.3.1.2.1 Interne Hardware teilweise austauschen (falls das TPM davon betroffen ist)<br>• B 9.1.1.3.3.2.1 Firmwareveränderungen<br>• B 9.1.1.3.3.2.1.1.1 Eigenes System booten<br>• B 9.1.2.1.2.1.1 Partition hinzufügen<br>• B 9.1.2.1.2.2.2.3 Fehler über Partitionstabelle<br>• B 9.1.2.2.1 Booten über Netzwerk<br>• B 9.2.2.2.1.1.1 Booten von extern<br>• B 9.2.2.2.1.1.2 Austausch der Festplatte |

| | |
|---|---|
| | • B 13.1.1.1.1 Manipulation am Binärcode |