



IT-Sicherheit

09 Trusted Computing

Gerrit Kalkbrenner
Teile: Norbert Pohlmann
Gerrit.Kalkbrenner@hwr-berlin.de



Trusted Computing

→ Inhalt

- **Ziele und Ergebnisse der Vorlesung**
- **Kernfunktionalitäten**
- **Sicherheitsarchitektur**
- **Trusted Network Connect**
- **Zusammenfassung**



Trusted Computing

→ Inhalt

- **Ziele und Ergebnisse der Vorlesung**
- Kernfunktionalitäten
- Sicherheitsarchitektur
- Trusted Network Connect
- Zusammenfassung



Ziele und Ergebnisse der Vorlesung

→ Trusted Computing

- Gutes Verständnis über die **Sicherheitsarchitektur** und **Sicherheitsprinzipien** von Trusted Computing erlangen.
- Erlangen der Kenntnisse über die TPM Schlüsselhierarchie, Authenticated Boot, Attestation, Binding, Sealing und Trusted Network Connect.
- Gutes Verständnis zu verschiedenen **Kernelarchitekturen** erlangen.
- Gutes Verständnis über praktische Anwendungen durch Betrachtung von **Turaya** als Beispielanwendung.



Trusted Computing

→ Inhalt

- Ziele und Ergebnisse der Vorlesung
- **Kernfunktionalitäten**
- Sicherheitsarchitektur
- Trusted Network Connect
- Zusammenfassung



Kernfunktionalitäten → Herausforderungen

- Reaktive Cyber-Sicherheitssysteme
 - „Airbag-Methode“
- Proaktive Cyber-Sicherheitssysteme
 - „Vorausschauend“
- Das Software-Problem



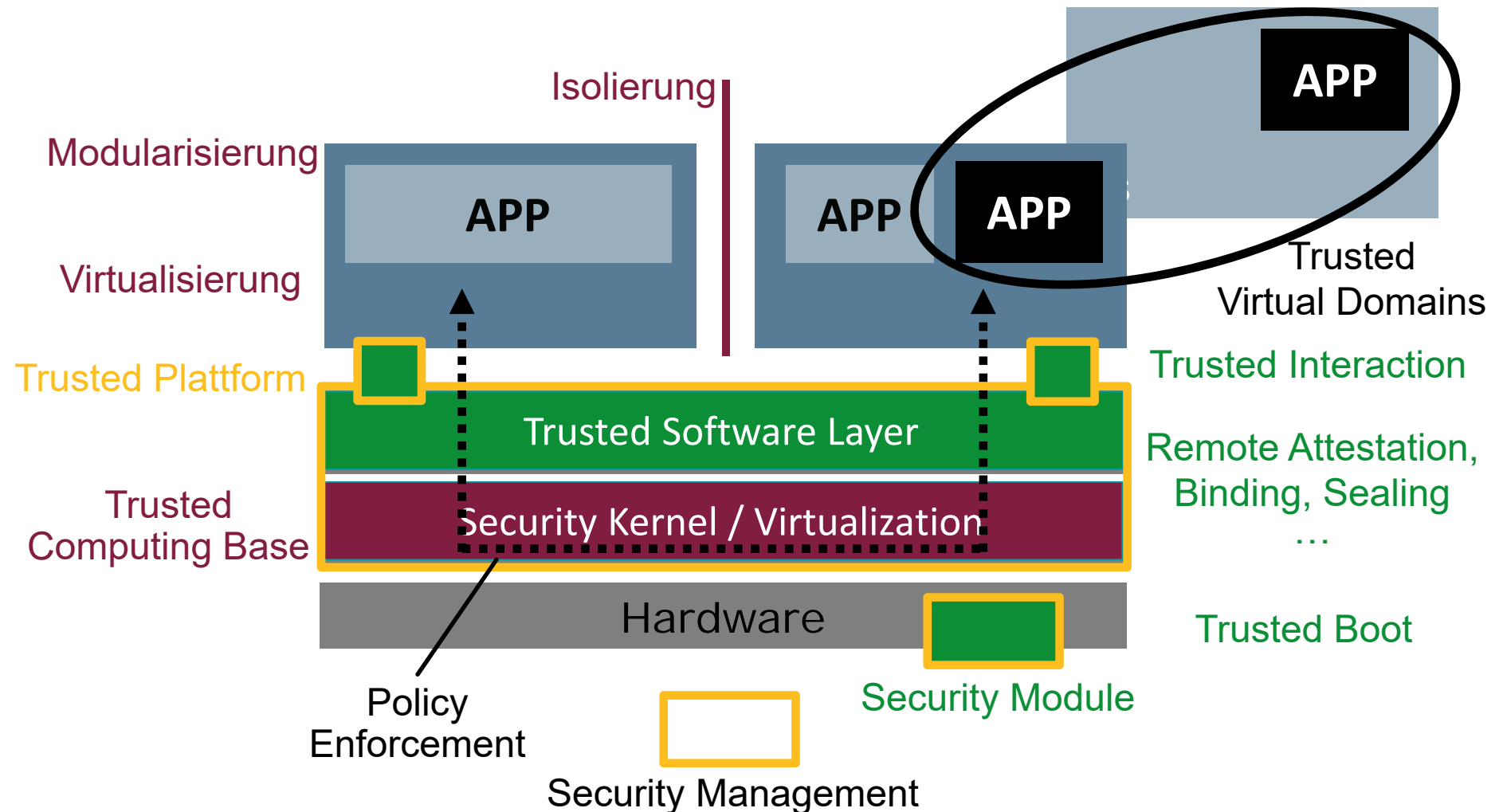


Kernfunktionalitäten → Sicherheitsprinzipien

Robustness/Modularity

Trusted Process

Integritätsprüfung





Trusted Computing

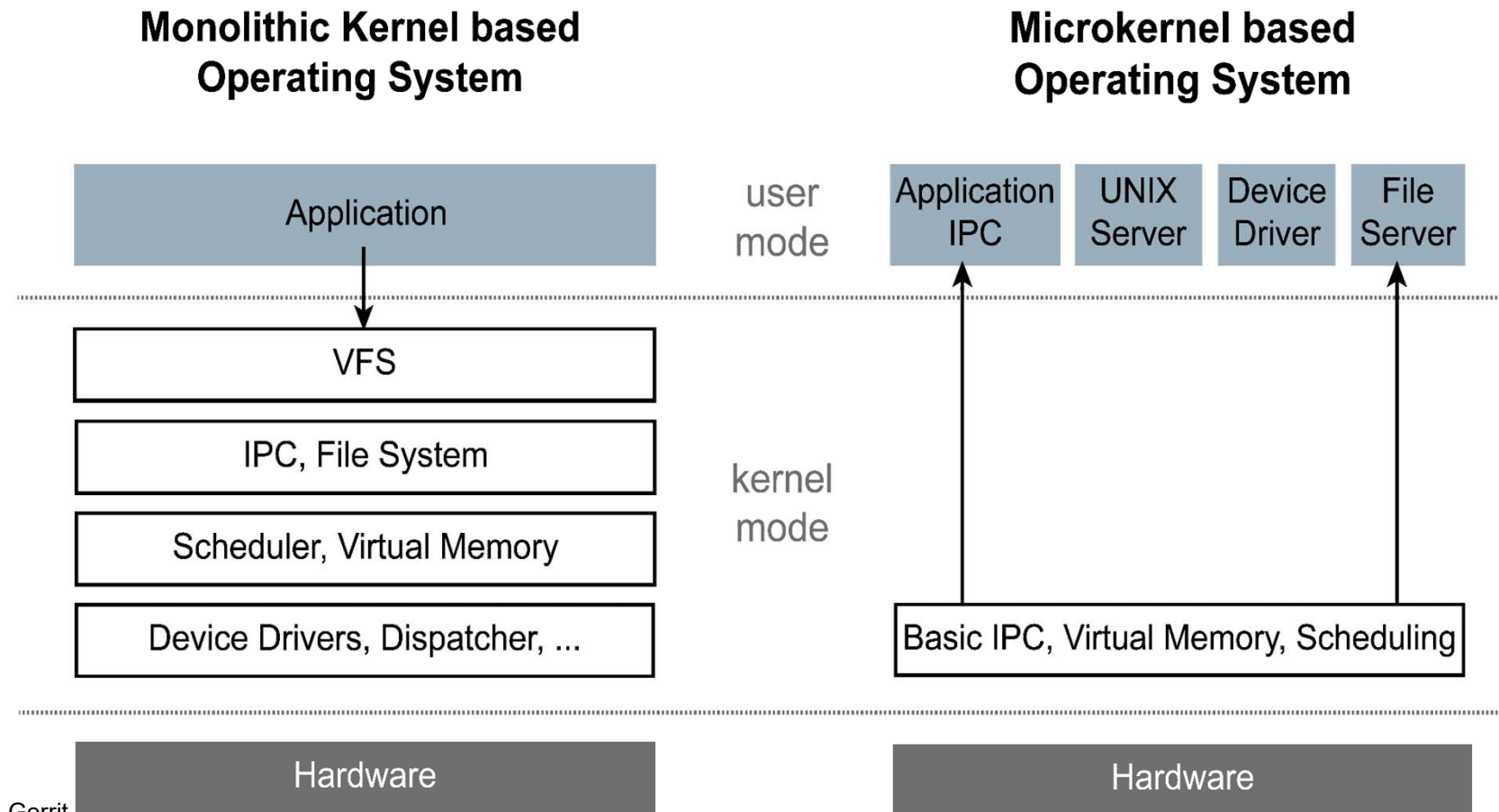
→ Inhalt

- Ziele und Ergebnisse der Vorlesung
- Kernfunktionalitäten
- **Sicherheitsarchitektur**
- Trusted Network Connect
- Zusammenfassung



Sicherheitsarchitektur

→ Kernelarchitekturen (1/3)





Sicherheitsarchitektur

→ Kernelarchitekturen (2/3)

- **Vorteile** eines monolithischen Kernels:
 - Lange etabliert
 - Gute Performance
- **Nachteile** eines monolithischen Kernels:
 - Alle Treiber vereint im Kernel-Space
 - Geringere Flexibilität
 - Höhere Komplexität
 - Wenig robust
 - Schlechte Sicherheitsmechanismen



Sicherheitsarchitektur

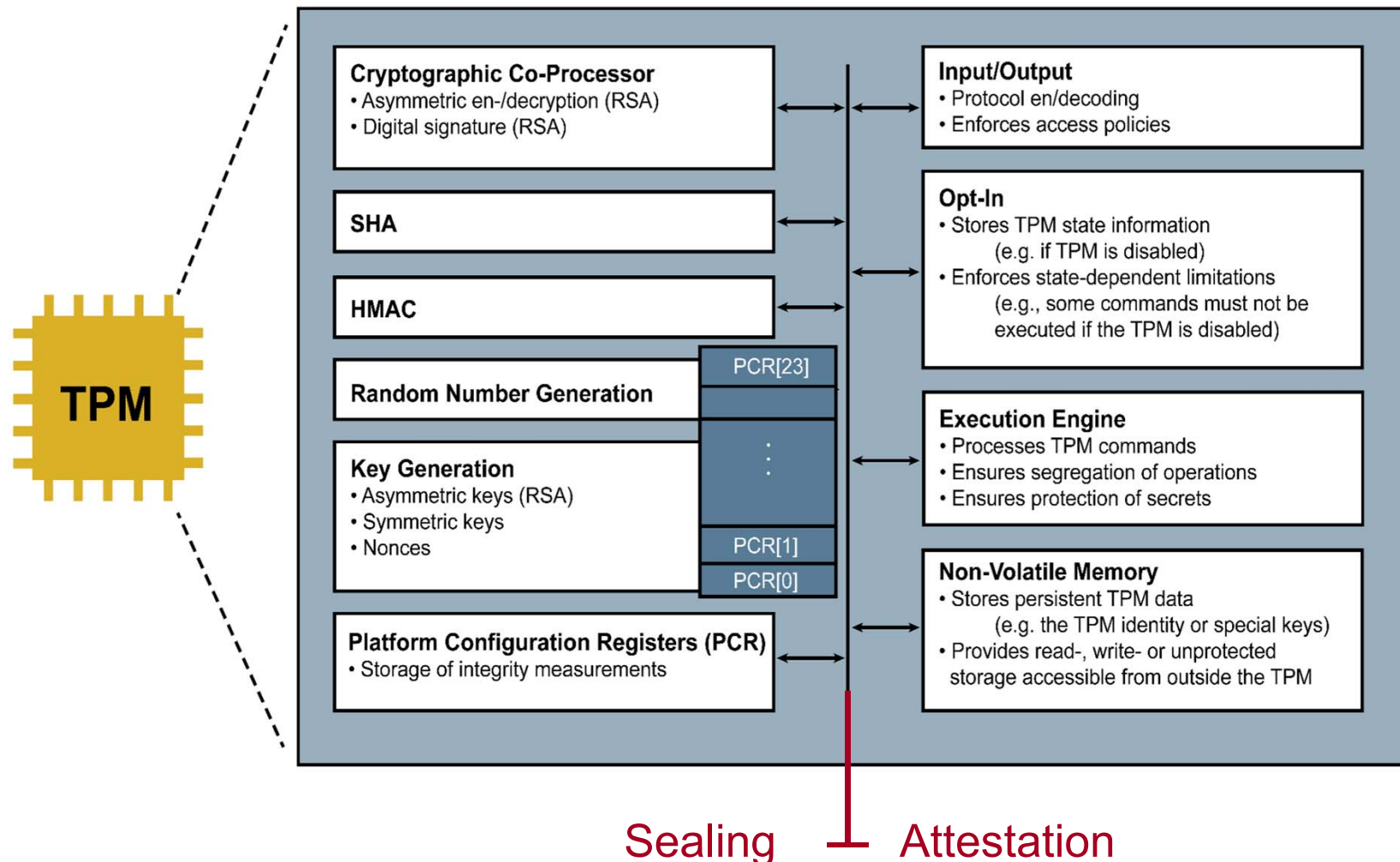
→ Kernelarchitekturen (3/3)

- **Vorteile** eines Mikrokernels:
 - Höhere Robustheit
 - Höhere Modularität
 - Höhere Flexibilität
 - Höhere IT-Sicherheit (kontrollierbare Interprozesskommunikation)
 - Weniger benötigter Speicherplatz
- **Nachteile** eines Mikrokernels:
 - Geringere Performanz, da erhöhte Kommunikation zwischen den Prozessen



HSM: Trusted Platform Module

→ Hardware-Sicherheitsanker



Sicherheitsarchitektur

→ Core Root of Trust for Measurement (CRTM)

- Messvorgang über einzelne **Systemzustände** (Hard- und Software).
 - Speicherung der Messungen in den **PCRs**.

■ Authenticated Boot:

- Systemzustände messen.
- Speicherung in den PCRs.
- Überprüfung der Integrität.

■ Secure Boot:

- Systemzustände messen.
- Überprüfung der Integrität.
- Ggf. Bootvorgang stoppen.

Entity E_0

"Transitives Vertrauen"



Sicherheitsarchitektur

→ Identitäten (1/2)

- **Endorsement Key (EK):**
 - Eindeutige TPM-Identität (nicht migrierbar).
 - RSA-Schlüsselpaar (im Herstellungsprozess erzeugt).
 - Geheimer Schlüssel im TPM gespeichert.
 - Öffentlicher Schlüssel ist datenschutzsensitiv.
 - TPM-Hersteller verwaltet PKI.
- **Endorsement Credential (EC):**
 - Elektronisches Zertifikat vom TPM-Hersteller.
 - Bestätigt ordnungsgemäße Erstellung und Einbettung des EK.
 - Bestandteile: TPM-Herstellername, TPM-Modellnummer, TPM-Version, Öffentlicher Schlüssel des EK.

→ Identitäten (2/2)

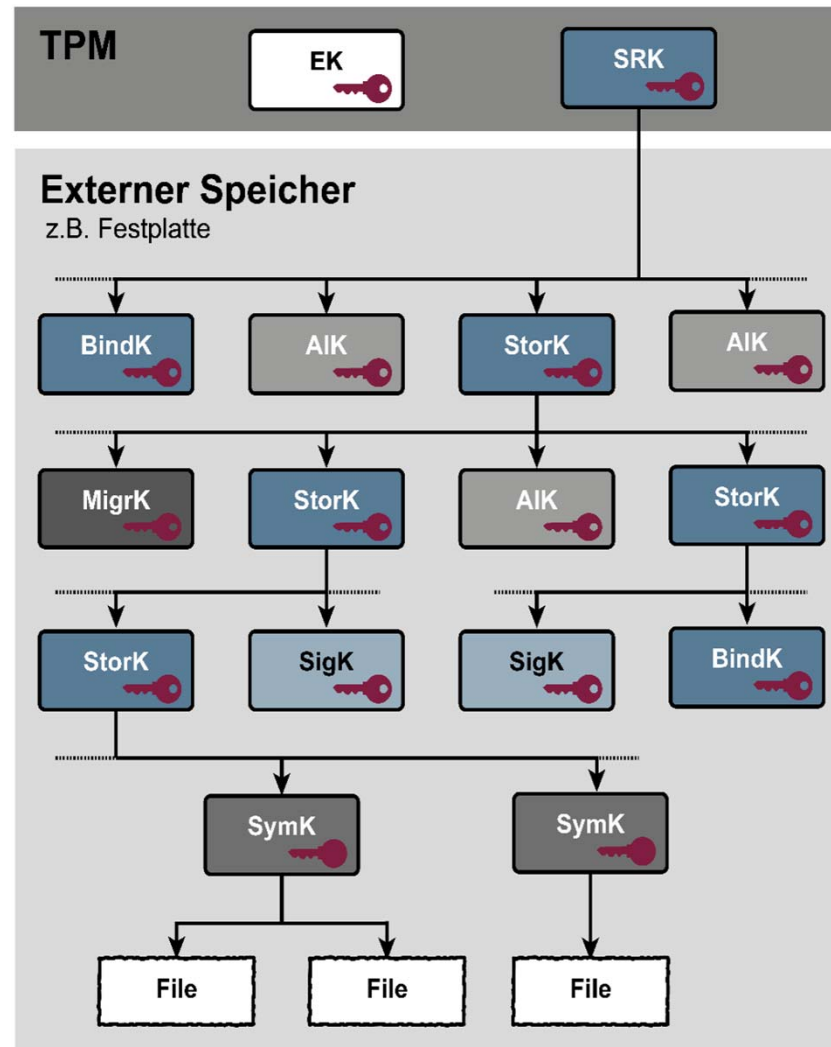
- **Platform Identität (PI):**

- Entspricht der TPM-Identität (EK).
- Physikalische oder logische Bindung des TPMs an die Plattform (z.B. mittels anlöten an das Motherboard oder Kryptographie).
- Plattform $\hat{=}$ Motherboard/IT-System.
- Plattform muss konform zur Evaluierungsrichtlinien der TCG sein
→ Conformance Credential (CC)

- **Platform Credential:**

- Elektronisches Zertifikat vom Plattform-Hersteller.
- Bestätigt gültige Verbindung zwischen TPM und Plattform
→ Trusted Plattform.
- Bestandteile: Name des Plattformherstellers, Plattformmodell und Versionsnummer, Verweise auf die EC und CC.

→ Schlüssel und deren Eigenschaften (1)





Sicherheitsarchitektur

→ Schlüssel und deren Eigenschaften (2)

- **Migratable Keys** → Auf andere Plattformen übertragbar.
- **Non-Migratable Keys** → An die Plattform gebunden.
- **Storage Root Key (SRK):**
 - Wurzel der Schlüsselhierarchie.
 - Während der Installation des TPM-Eigentümers generiert.
 - Löschung des TPM-Eigentümers → Löschung des SRK → Kein Zugriff auf die Schlüsselhierarchie mehr.
 - **Eigenschaften:**
 - Steht im nicht flüchtigen Speicher des TPMs.
 - Ist nicht migrierbar.

Sicherheitsarchitektur

→ Schlüssel und deren Eigenschaften (3)

- **Attestation Identity Keys (AIK):**
 - Verwendet für die Trusted Computing Funktion „**Attestation**“:
 - Authentische Bestätigung der Integrität einer Plattformkonfiguration (z.B. aktuelle Hard- und Softwareumgebung).
 - Nötig, da EK datenschutzsensibel ist.
 - AIKs werden vom TPM-Besitzer generiert.
 - TPM/Plattform kann mehrere AIKs besitzen (z.B. für Online-Banking, E-Mail, ...)
 - **Eigenschaften:**
 - Stehen im nicht flüchtigen Speicher des TPMs.
 - Nicht migrierbar.

Sicherheitsarchitektur

→ Schlüssel und deren Eigenschaften (4)

- **Storage Keys (StorK):**
 - Verschlüsselung von weiteren Schlüsseln und Daten außerhalb des TPMs.
 - Verwendet für die Trusted Computing Funktion „**Sealing**“:
 - Zustand der Plattform wird Teil der Verschlüsselung.
 - Entschlüsselung nur im vorher definierten Zustand möglich.
 - **Eigenschaften:**
 - RSA-Schlüsselpaar.
 - Im Allgemeinen darf die Migration zu anderen TPMs erfolgen.



Sicherheitsarchitektur

→ Schlüssel und deren Eigenschaften (5)

- **Binding Keys (BindK):**
 - Verschlüsselung von beliebigen Daten außerhalb des TPMs.
 - Entspricht der asymmetrischen Verschlüsselung.
 - **Eigenschaften:**
 - RSA-Schlüsselpaar (es können auch andere Algorithmen vom TPM unterstützt werden).
 - Im Allgemeinen darf die Migration zu anderen TPMs erfolgen.
 - Kann nur mit Binding-Befehlen verwendet werden.



Sicherheitsarchitektur

→ Schlüssel und deren Eigenschaften (6)

- **Signing Keys (SigK):**
 - Nachweis der Authentizität und Integrität von beliebigen Daten /Protokollnachrichten innerhalb und außerhalb des TPMs.
 - **Eigenschaften:**
 - RSA-Schlüsselpaar (es können auch andere Algorithmen vom TPM unterstützt werden).
 - Im Allgemeinen darf die Migration zu anderen TPMs erfolgen.

Sicherheitsarchitektur

→ Schlüssel und deren Eigenschaften (7)

TPM Key Object 

General Information

Key Type

Algorithm

Authorization Secret

Specific Information

Key Length

Key Data

Key Properties

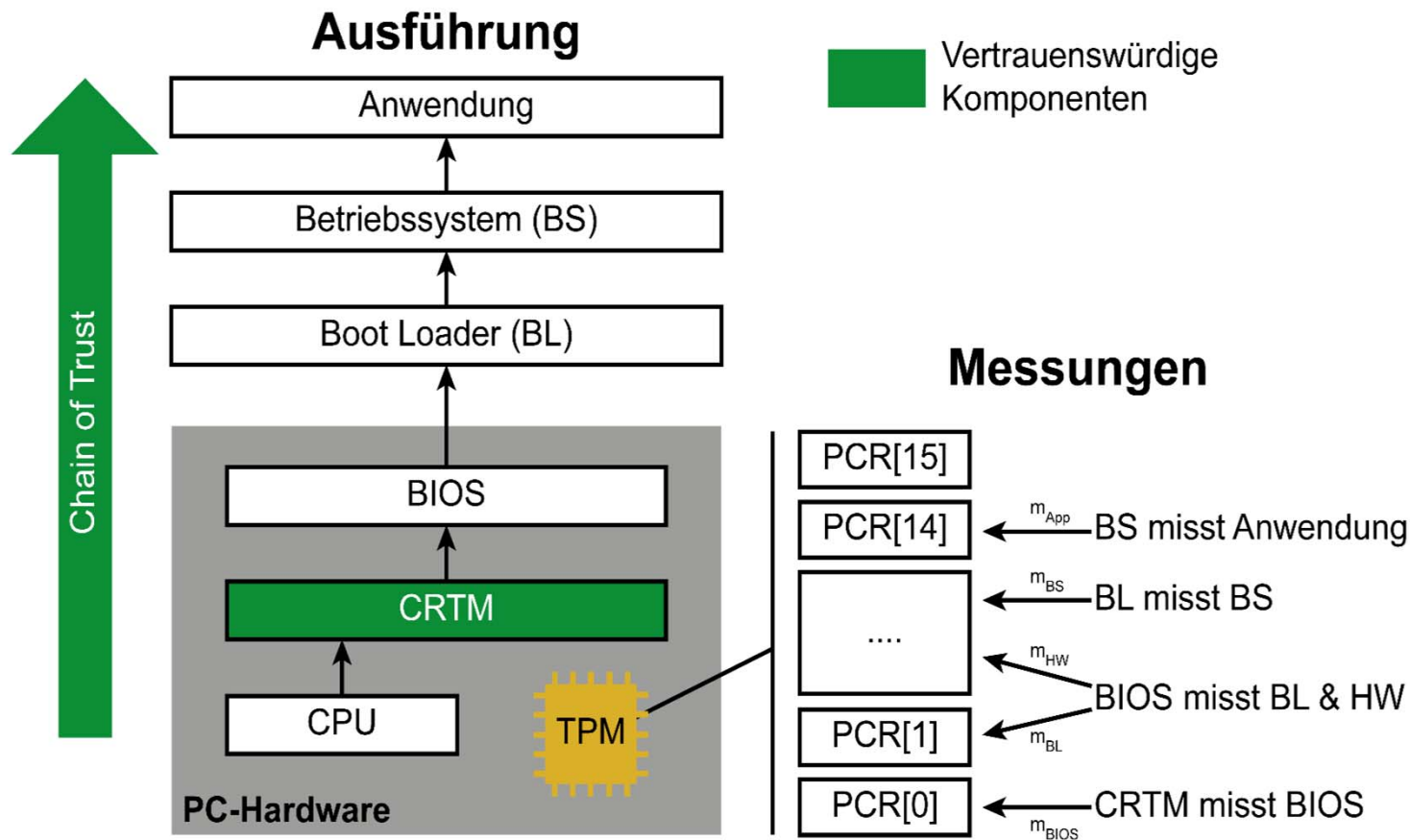
Migration

PCR Values



Sicherheitsarchitektur

→ Authenticated Boot





Sicherheitsarchitektur

→ Sealing Funktionen und Parameter

Eingabe Parameter

daten {unverschlüsselte Daten}

Ausgabe Parameter

cipher {verschlüsselte Daten}

cryptedKEY {verschlüsselter Schlüssel}

TPM Interne Funktionen und Daten

encrypt (key, daten) {symmetrischer Verschlüsselungsalgorithmus „AES“}

H (daten) {One-Way-Hashfunktion „SHA-256“}

genKey() {Schlüsselerzeugung}

SRK {Storage Root Key}

PCRs {PCR-0, PCR-1, ...} z.B. aktuell abgespeicherte PCR-Werte

plainKEY = genKEY ()

cipher = encrypt (plainKEY, (daten // H (daten // PCR-0 // ... // PCR-x))

cryptedKEY = encrypt (SRK, plainKEY // H (plainKEY))



Sicherheitsarchitektur

→ Un-Sealing Funktionen und Parameter

Eingabe Parameter

<i>cipher</i>	<i>{verschlüsselte Daten}</i>
<i>crypteKEY</i>	<i>{verschlüsselter Schlüssel}</i>

Ausgabe Parameter

<i>daten</i>	<i>{unverschlüsselte Daten}</i>
--------------	---------------------------------

TPM Interne Funktionen und Daten

<i>decrypt (key, daten)</i>	<i>{symmetrischer Verschlüsselungsalgorithmus „AES“}</i>
<i>H (daten)</i>	<i>{One-Way-Hashfunktion „SHA-256“}</i>
<i>checkPCRs (Hash-Value)</i>	<i>{vergleicht PCRs-Inhalte mit Hash-Value}</i>
<i>SRK</i>	<i>{Storage Root Key}</i>
<i>PCRs</i>	<i>{PCR-0, PCR-1, ...}</i>

plainKEY = decrypt (SRK, crypteKEY)

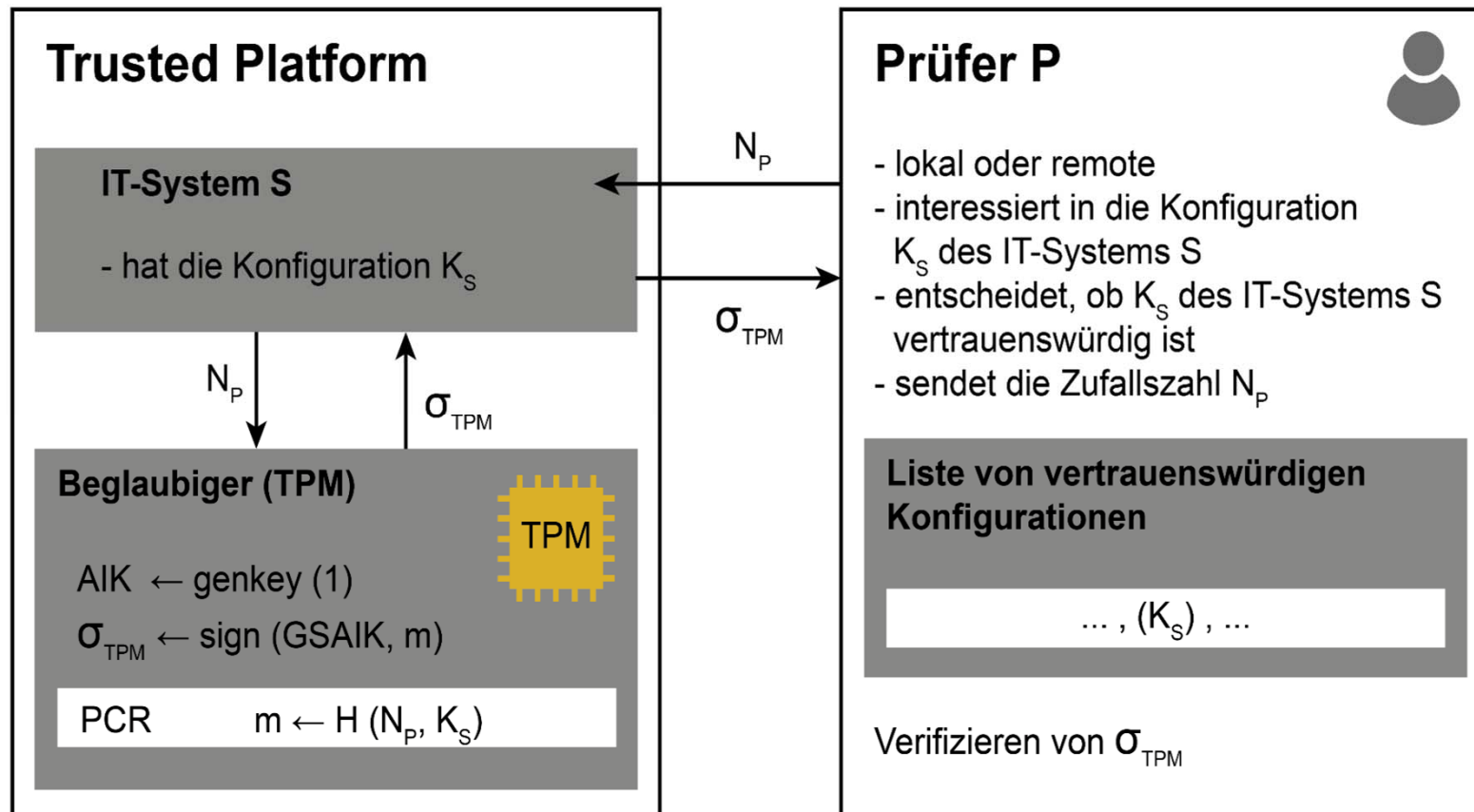
daten // H (daten // PCR-0 // ... // PCR-x) = decrypt (plainKEY, cipher)

if (checkPCRs (Hash-Value))

return daten

else

return ERROR





Sicherheitsarchitektur

→ Signaturfunktion für die Attestierung

Eingabe Parameter

random

{Zufallszahl des Prüfers $P - N_P$ }

Ausgabe Parameter

signature//certificate

{Signatur der aktuellen Systemkonfiguration des AIKs}

TPM Interne Funktionen und Daten

sign (key, daten)

{RSA-Signatur}

H (daten)

{One-Way-Hashfunktion "SHA-256"}

GSAIK

{geheimer AIK-RSA-Schlüssel}

AIK-certificate

{elektronisches Zertifikat des AIKs}

PCRs

{PCR-0, PCR-1, ...} z.B. aktuell abgespeicherte PCR-Werte

$$\sigma_{TPM} = \text{sign} (\text{GSAIK}, H (\text{random} // \text{PCR-0} // \dots // \text{PCR-x}))$$



Sicherheitsarchitektur

→ Verifikationsfunktion für die Attestierung

Eingabe Parameter

signature//certificate

{Signatur der Systemkonfiguration des AIKs}

Ausgabe Parameter

return value

{Rückgabewert}

TPM Interne Funktionen und Daten

very (key, daten)

{RSA-Signatur-Verifikation}

H (daten)

{One-Way-Hashfunktion "SHA-256"}

ÖSAIK

{öffentlicher AIK-RSA-Schlüssel}

checkPCRs (PCR-Values)

{vergleicht den Inhalt der PCRs mit den gewünschten Werten}

PCRs

{PCR-0, PCR-1, ...}

if (very (ÖSAIK, σ_{TPM})) and

if (checkCERT (certificate)) and

if (checkPCRs (Hash-Value))

return ok

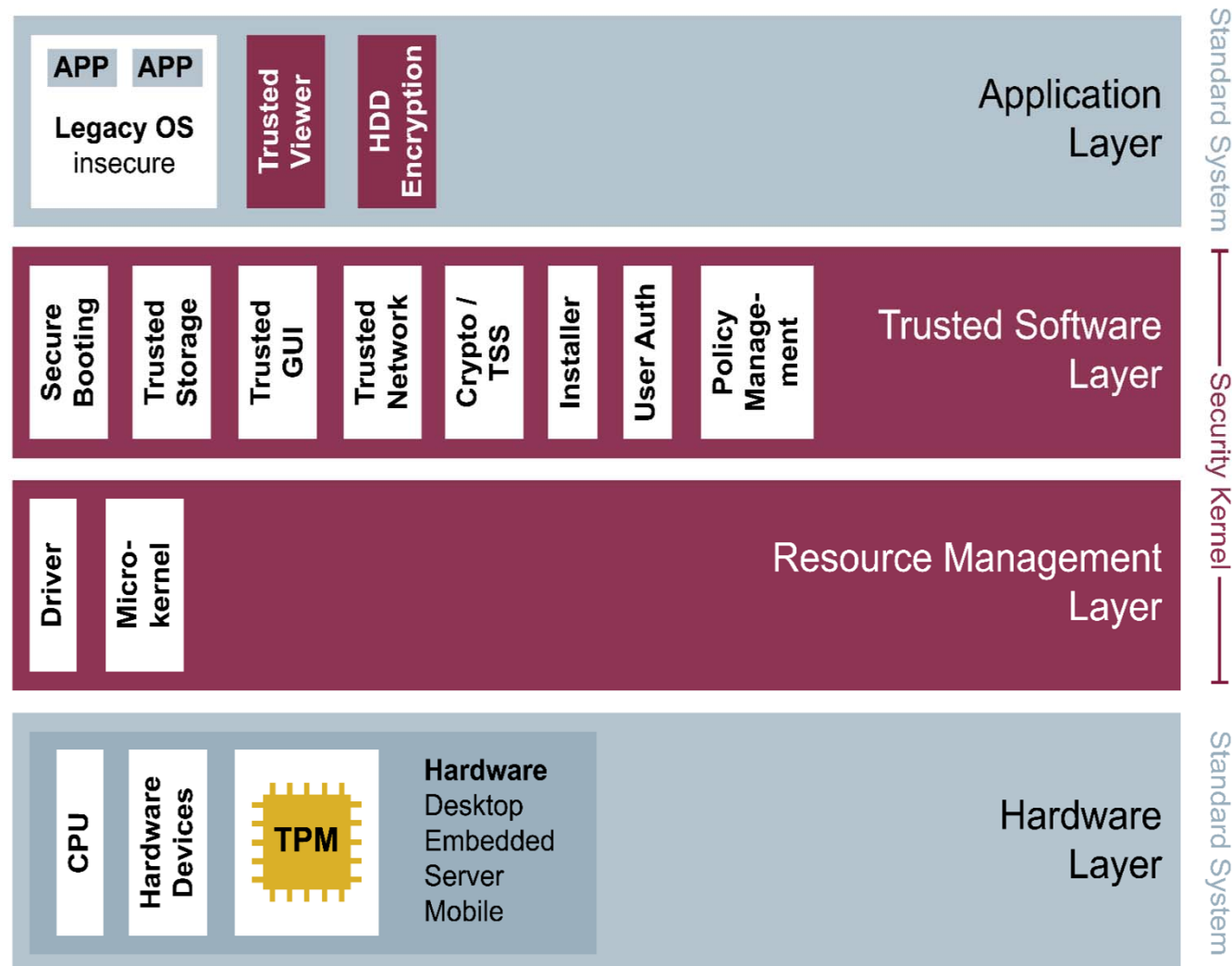
else

return ERROR



Sicherheitsarchitektur

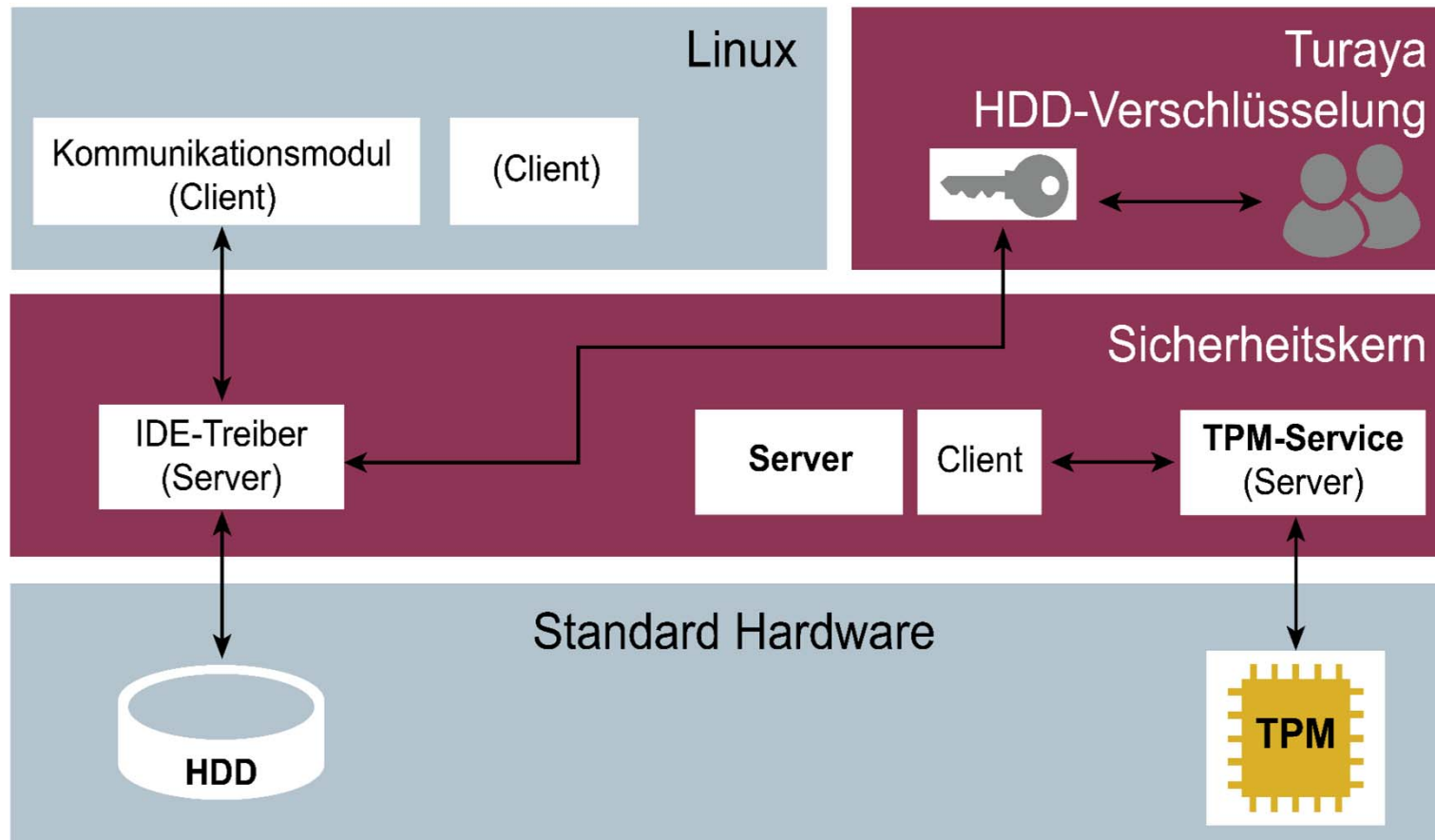
→ Trusted Platform





Sicherheitsarchitektur

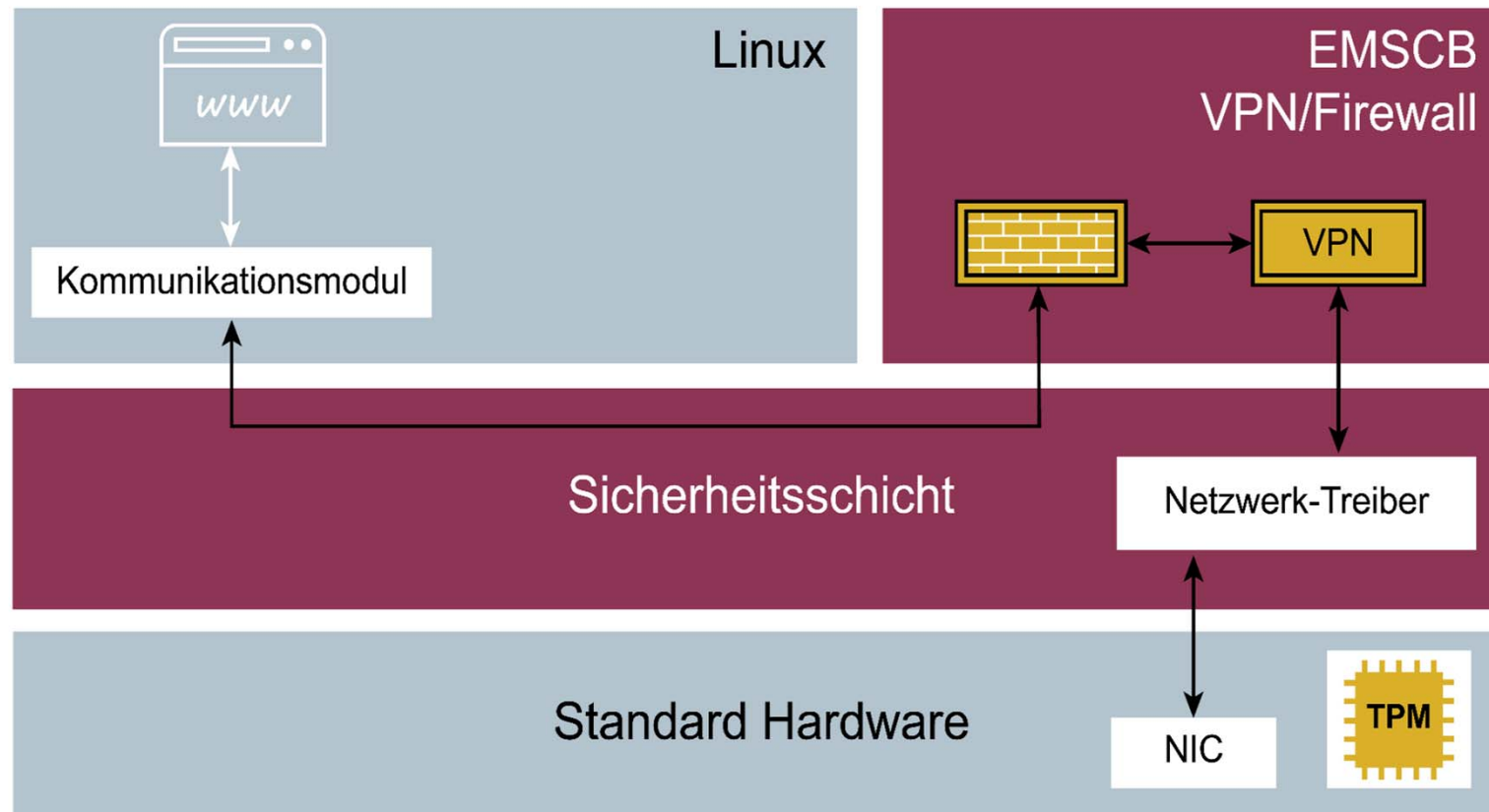
→ Beispielanwendung: Turaya.Crypt





Sicherheitsarchitektur

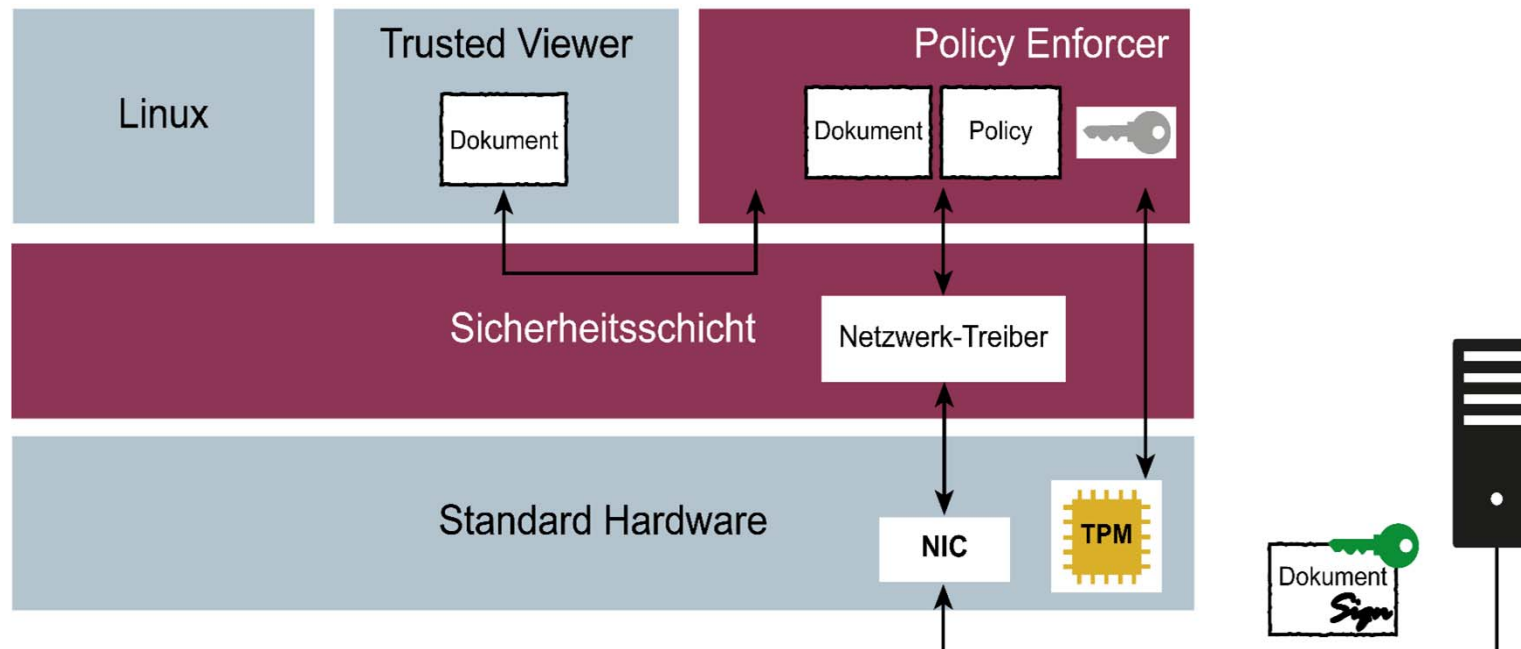
→ Beispielanwendung: Turaya.VPN





Sicherheitsarchitektur

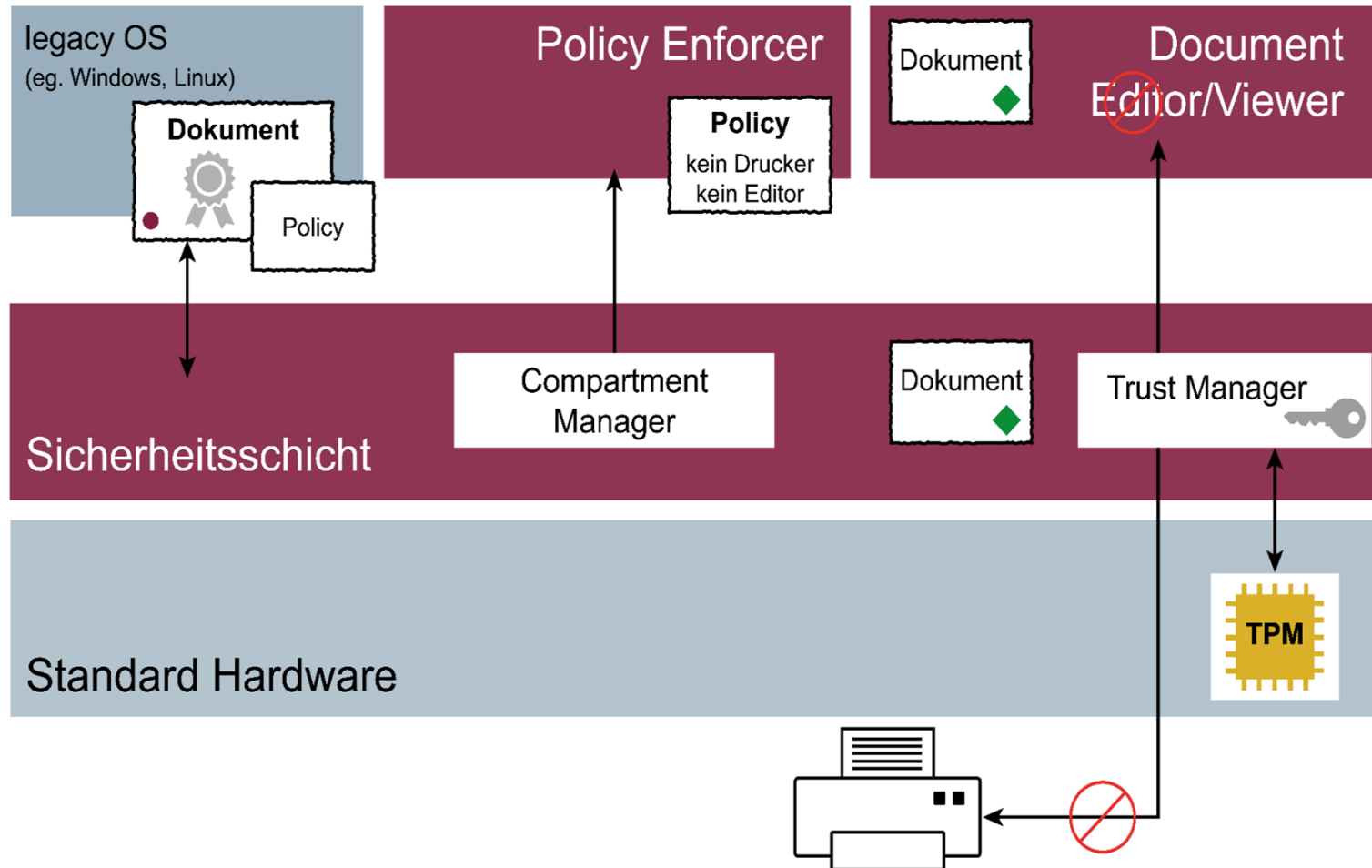
→ Beispielanwendung: Turaya.FairDRM





Sicherheitsarchitektur

→ Beispielanwendung: Turaya.FairDRM



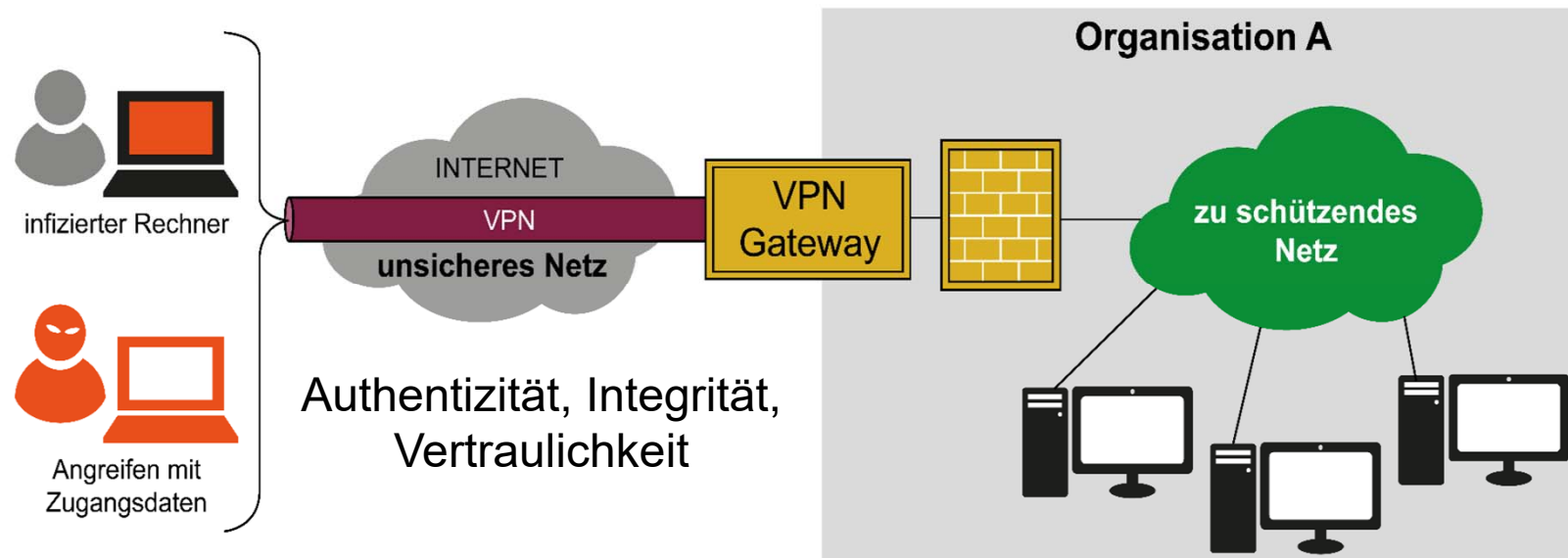
Trusted Computing

→ Inhalt

- Ziele und Ergebnisse der Vorlesung
- Kernfunktionalitäten
- Sicherheitsarchitektur
- **Trusted Network Connect**
- Zusammenfassung

Trusted Network Connect

→ Problemstellung



Lösungsansätze:

Microsoft NAP,

Cisco NAC,

Trusted Computing Group (TCG) → **Trusted Network Connect (TNC)**

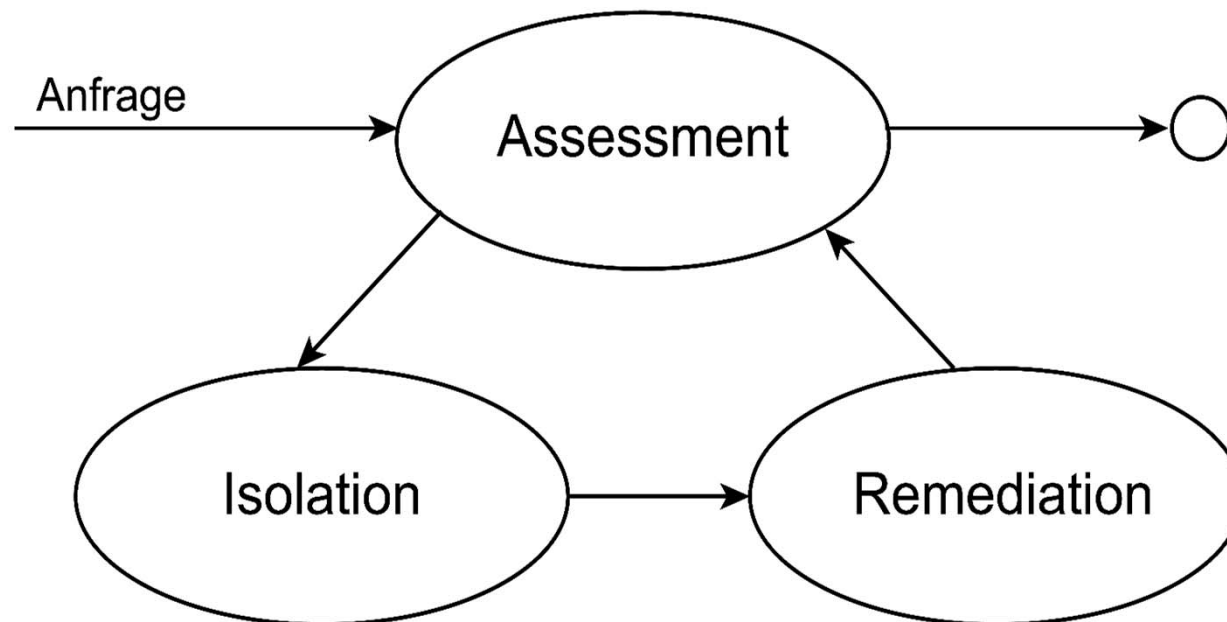
Trusted Network Connect

→ Vertrauenswürdige Netzwerkverbindungen

- Vertrauenswürdigkeit abhängig von der **Integrität**.
- **Alle** beteiligten **Kommunikationspartner** betrachten.
- **Kommunikationsrichtung** getrennt betrachten.
- Alle IT-Systeme, Infrastrukturelemente und das Umfeld der Kommunikation bewerten.
- Anforderungen in **Policies** definieren (z.B. erlaubte Konfigurationen, Betriebssysteme, Hard- und Software, ...).
- Überprüfung **vor dem Zugriff (Prävention)**.

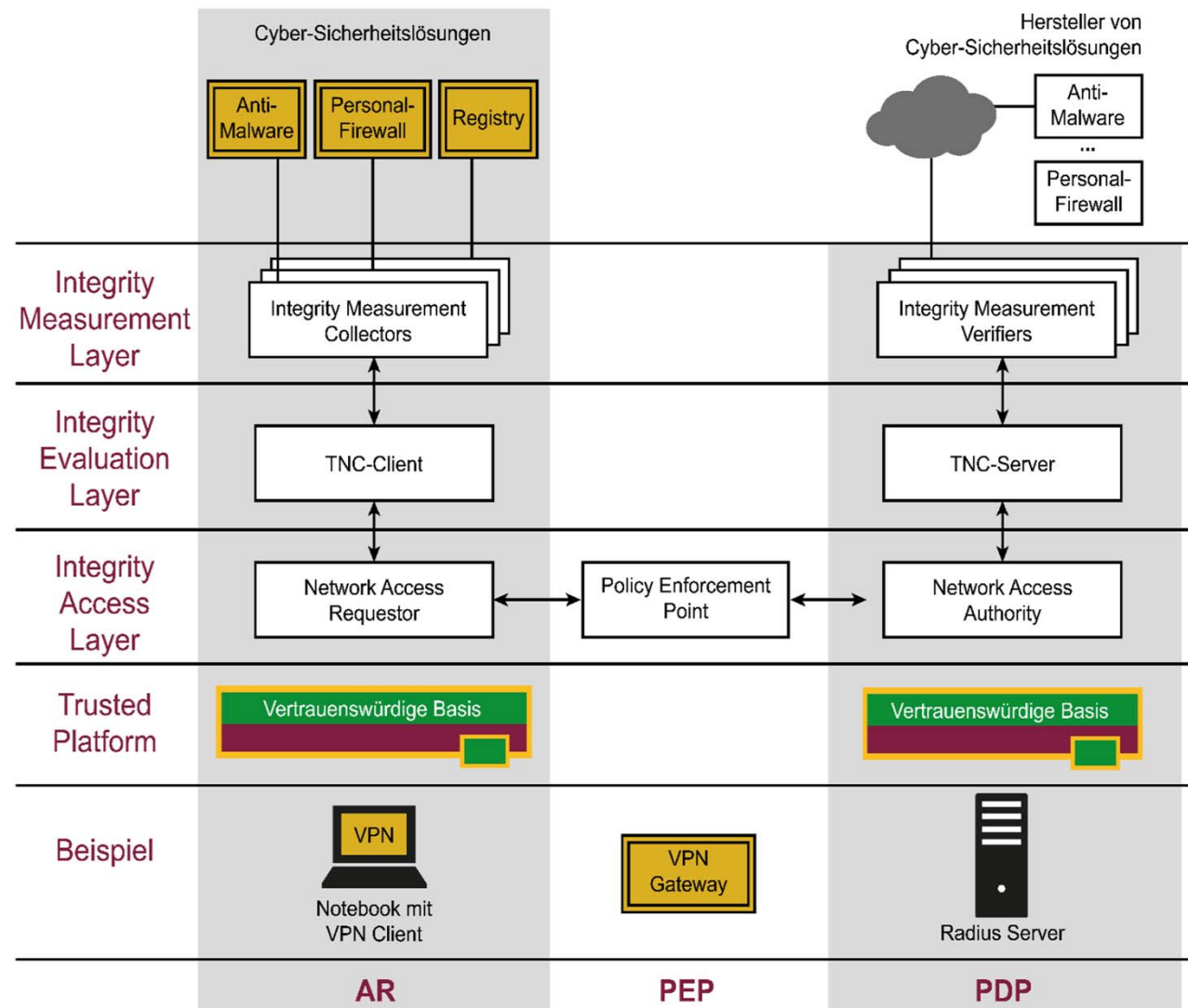


Trusted Network Connect → Phasen



Trusted Network Connect

→ Struktur





Trusted Computing

→ Inhalt

- Ziele und Ergebnisse der Vorlesung
- Kernfunktionalitäten
- Sicherheitsarchitektur
- Trusted Network Connect
- **Zusammenfassung**

Trusted Computing

→ Zusammenfassung (1/2)

- Die Kernfunktionalitäten von Trusted Computing sind:
 - **Robustheit und Modularität**
 - **Integritätsüberprüfung**
 - **Trusted Process**
 - **Trusted Plattform**
- Vor- und Nachteile der verschiedenen **Kernelarchitekturen** müssen miteinander abgewogen werden.
- **CRTM** ist die Vertrauensbasis. Das Vertrauen ist **transitiv**.
- Die **TPM Schlüsselhierarchie** ermöglicht eine sichere Speicherung von Daten, auch auf externen Speichermedien.



Trusted Computing

→ Zusammenfassung (2/2)

- Wichtige Trusted Computing Funktionen sind:
 - **Authenticated Boot**
 - **Binding**
 - **Sealing**
 - **Attestation**
- Vertrauenswürdige Netzwerkverbindungen können durch **Trusted Network Connect (TNC)** realisiert werden.
- Die Festlegung einer sicheren und vertrauenswürdigen **Systemkonfiguration** ist mit zahlreichen Schwierigkeiten (technisch und politisch) verbunden.