Themen zur Computersicherheit

Sicherheit in der Cloud

PD Dr. Reinhard Bündgen
buendgen@de.ibm.com

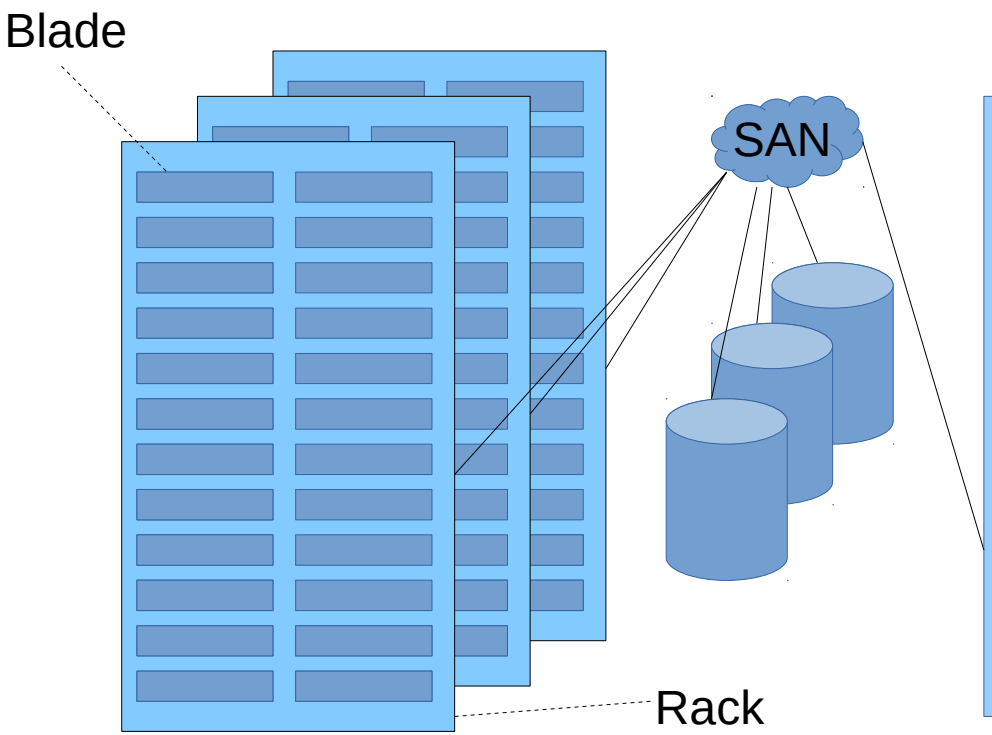
Wirt- und Gastsysteme

- klassisch: Konten auf Servern (Universität, Unternehmen)
  - HW, BS (SW), Speichermedien gehören vertrauenswürdiger Organisation
- Internetcafé
  - HW, BS, SW gehören fremder Organisation
  - Netzzugang zu „eigenem“ Server
- Cloud (privat/öffentlich)
  - Speicher Cloud
    - Speichermedien gehören fremder Organisation
    - eigener Rechner samt BS und SW
  - IaaS (Infrastructure as a service) Cloud
    - Speichermedien, (virtuelle) HW, evtl. Hypervisor/VMM gehören fremder Organisation
    - eigene BS und SW
  - PaaS (platform aaS) Cloud
    - Speichermedien, (virtuelle) HW, evtl. Hypervisor/VMM und BS gehören fremder Organisation
    - eigene SW; BS wird selbst betrieben
  - SaaS (software aaS) Cloud
    - Speichermedien, (virtuelle) HW, evtl. Hypervisor/VMM, BS, SW und HW gehören fremder Organisation
    - SW wird selbst betrieben (evtl. im Rahmen eines Containers)

Wirtsysteme (Hosts)

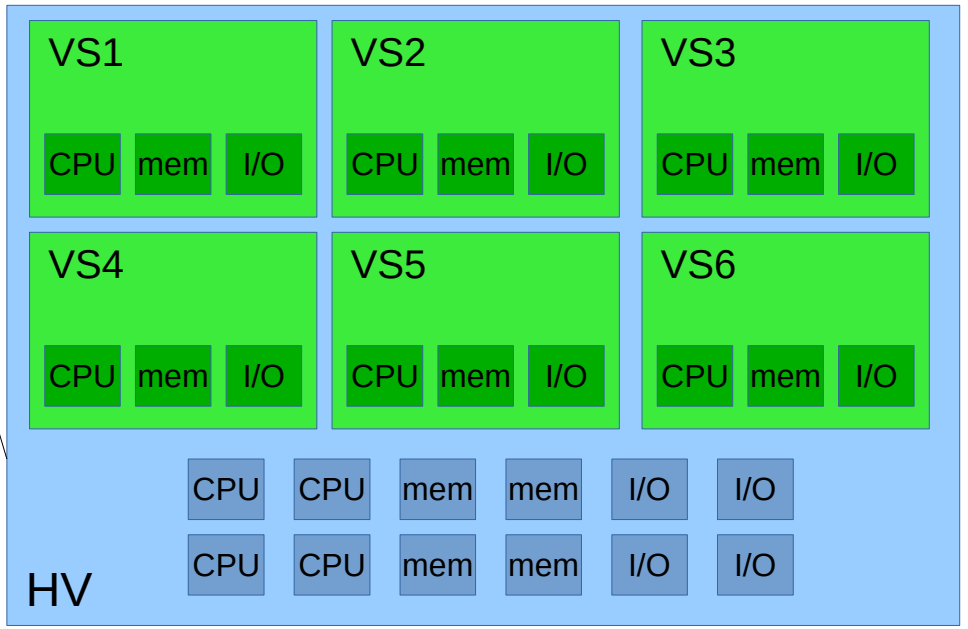
Blade Systeme

- Blade = Rechner auf einer Karte mit geringen Abmessungen (1U: 480mm breit, 44 mm hoch)
- viele Blades können in ein Rack gesteckt werden
- Cloudbetreiber, verwaltet Blades und überlässt Blades den Cloudkunden



Virtuelle Systeme

- Hypervisor (HV) oder virtual machine monitor (VMM) emuliert viele virtuelle Server (VS) auf einer HW
- jeder virtuelle Server hat virtuellen Speicher, CPUs und E/A Geräte und kann nicht auf virtuelle Ressourcen anderer virtueller Server zugreifen
- Cloudbetreiber verwaltet Server mit HV und virtuelle Server und überlässt virtuelle Server den Cloudkunden



Angriffe auf Gastsysteme

Angriffspunkte

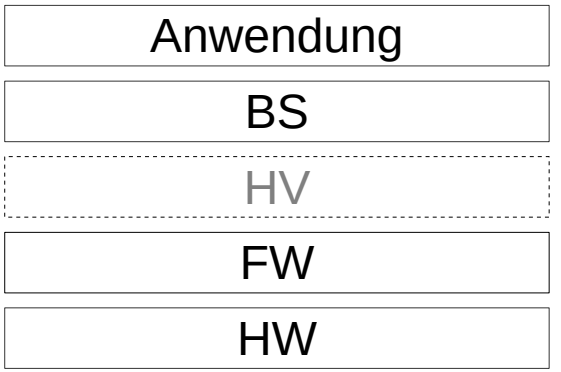
- Netzwerk Kommunikation
  - abhören, unautorisierte Nachrichten einfügen
- HW, FW, HV, OS, SW
  - bösartige Änderungen (z.B Installation von Keyloggern)
- Speichernetzwerk (Leitungen, Switches)
  - abhören, unautorisierte Nachrichten einfügen
- Speichersysteme
  - unautorisiertes Lesen und Ändern
- HV
  - unbefugtes login,
  - unbefugte Operationen an Virtuellen Maschinen

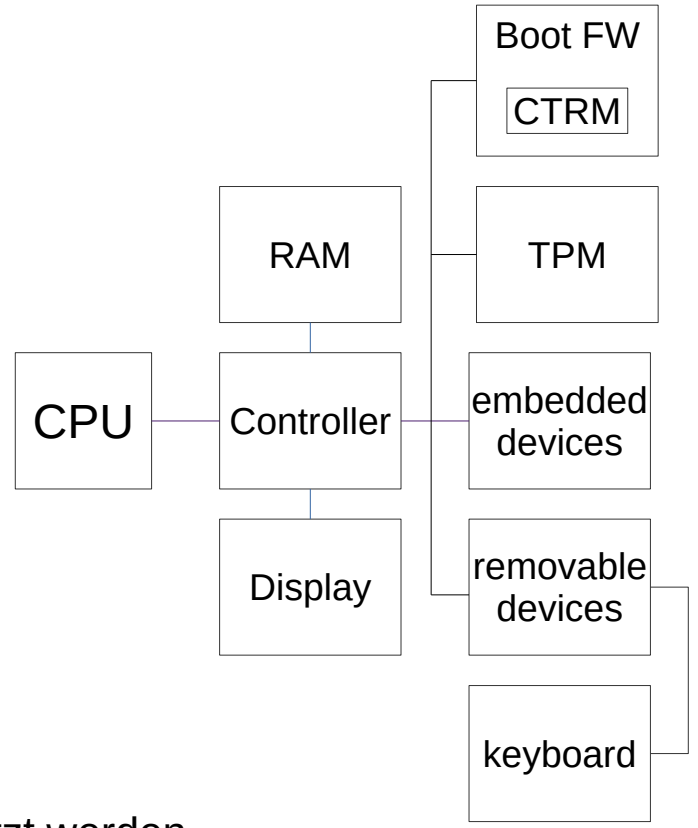
Angreifer

- Betreiber (Administrator)
  - bösartig
  - schlampig
  - erpresst
- Außenseiter
  - Ausnutzung von Sicherheitslücken
    - in HW,FW, HV, OS, SW
    - Prozessschwächen des Betreibers bei Administration
    - Erpressung des Betreibers

Trusted Computing (TC)

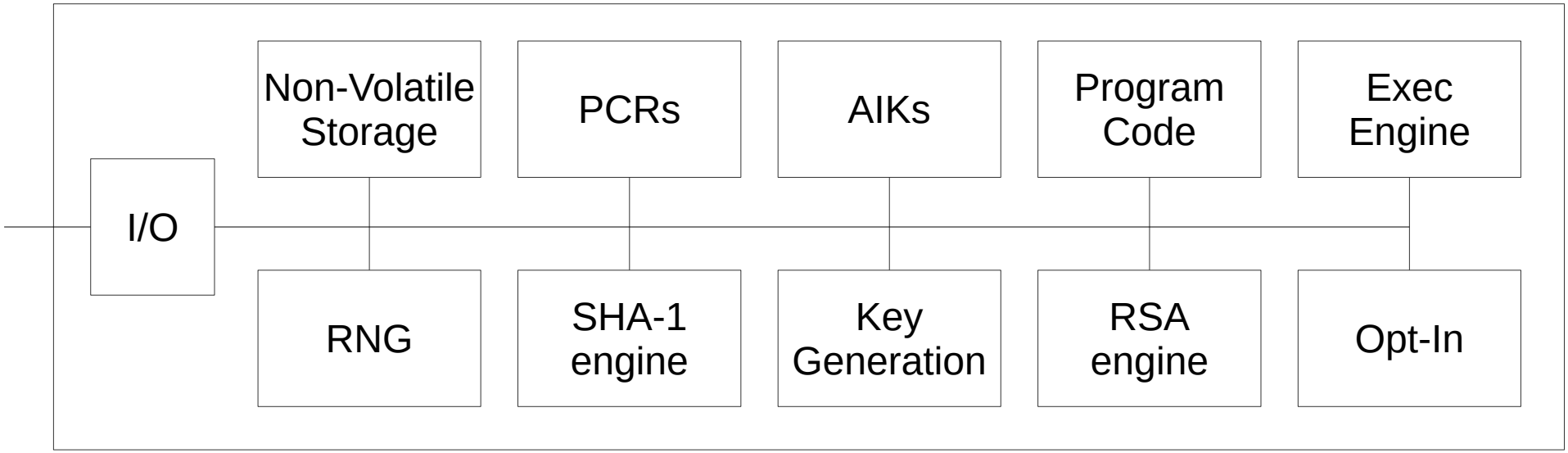
- Ziele:
  - Vertrauenswürdigkeit aller Hardware (HW), Firmware(FW) und Software (SW) Schichten eines Computersystems *festzustellen*
  - Blockade von nicht-vertrauenswürdigen Systemen*
- standardisiert durch TCG
- Benötigt
  - einen vertrauenswürdigen HW Chip: Trusted Platform Module (TPM)
    - passiv, „tamper protected“
  - geänderte Boot FW mit einer Core Root of Trust for Measurement (CRTM) Komponente
  - weitere SW Schichten, die TC unterstützen





*) dies kann auch für digitial rights management (DRM) genutzt werden

Trusted Platform Module



- Platform Configuration Registers (PCR)
  - kann signierten Hash eines Messpunktes aufnehmen
- Attestation Identity Key (AIK)
  - signiert TPM Daten

- kryptographische Funktionen
  - Zufallszahlgenerator (RNG)
  - Schlüsselgenerator
  - SHA1, RSA
- TPM 2.0 unterstützt auch
  - SHA-2, SHA-3, 3DES, AES, ECC

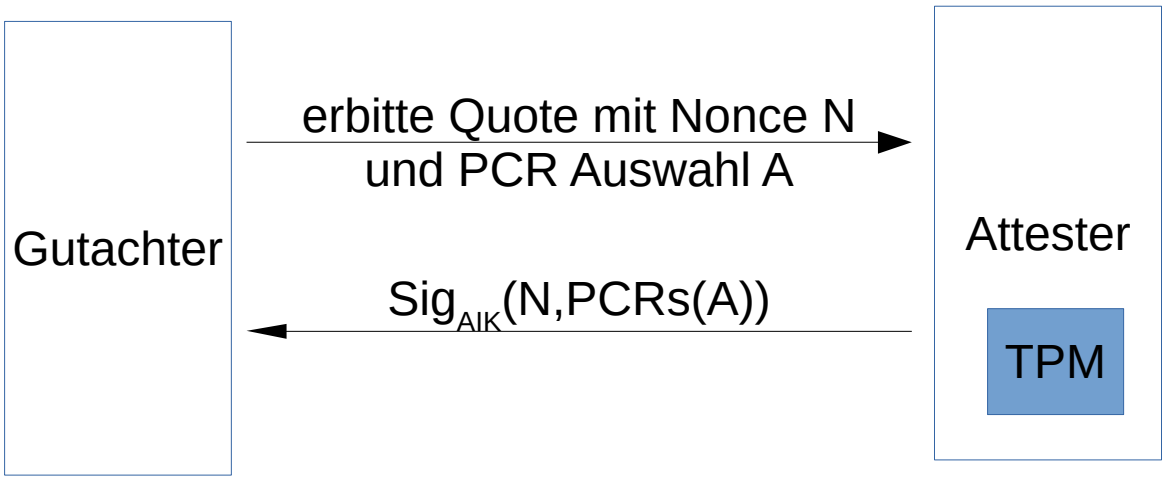
Vetrauensbeweise

Vertrauensbasis (root of trust)

- RTM root of trust for Measurement -> FW oder CPU Code der in trusted mode abläuft
- RTR root of trust for reporting -> TPM -> endorsement key (EK)
- RTS root of trust for strorage -> TPM -> storage root key (SRK)

Attestation

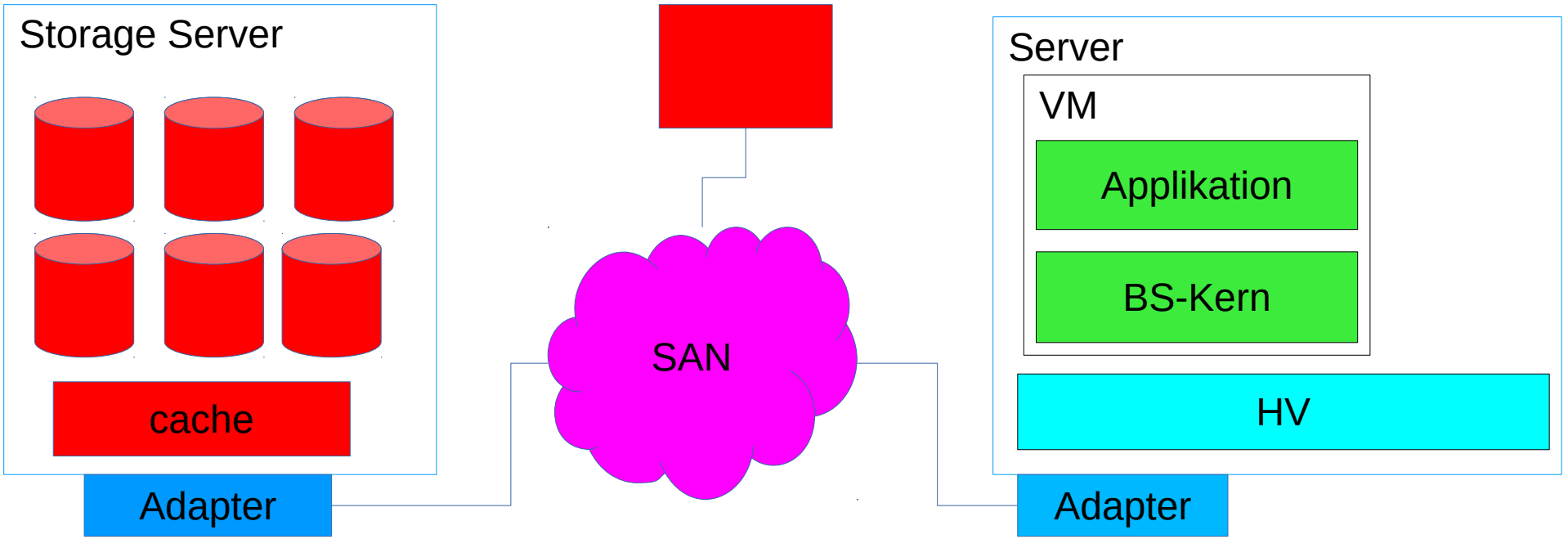
- verifizierbarer Beweis für externe Partei (Gutachter, appraiser)
- Quote
  - Nonce + PCR Inhalte signiert mit AIK



Vertrauenswürdiger Bootprozess

- Möglicher Bootprozess
  1. CRTM misst HW Konfiguration und FW (Bootloader) und schreibt Hash des Messpunktes in PCR,
  2. CRTM startet Booloader wenn Messung OK
  3. Booloader läd BS misst zu startendes BS und schreibt Hash des Messpunktes in PCR
  4. Bootloader started BS, wenn Messung OK
  5. BS läd Applikation und misst Applikation und schreibt Hash des Messpunktes in PCR
  6. BS startet Applikation, wenn Messung OK
- baut Vertrauenskette (chain of trust) auf
- Jede Schicht, kann überprüfen, ob darunter liegende Schicht vertrauenswürdig ist und gegebenenfalls eine Berechnung abbrechen
  - vergleiche Messwert der Schicht mit entsprechenden im TPM abgelegten PCR

Datenverschlüsselung



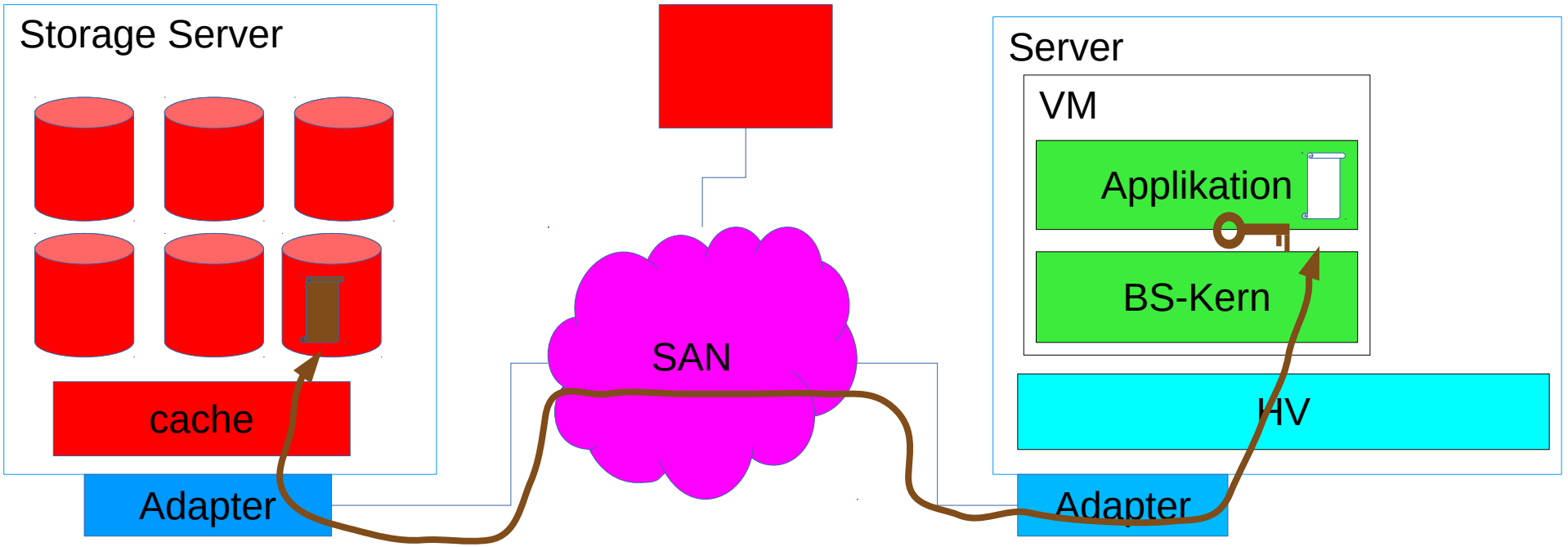
- Wo wird ver- und entschlüsselt?
  - Storage Server, Adapter, Server, HV, BS-Kern Applikation
- Wer erzeugt, die Schlüssel?
  - Kunde, (Storage/Cloud) Betreiber, System (per Zufall)
- Wem gehören die Schlüssel?
  - Kunde, (Storage/Cloud) Betreiber
- Backups? Datenumzug?

Angriffspunkte

- Storage server
- SAN
- Server / HV

- Angreifer
  - Betreiber
  - Außenseiter

Ende-zu-Ende Datenverschlüsselung



- Wo wird ver- und entschlüsselt?
  - BS-Kern oder Applikation
- Wer erzeugt, die Schlüssel?
  - Kunde oder VM (per Zufall)
- Wem gehören die Schlüssel?
  - Kunde
- Backups? Datenumzug?

abgesicherte Angriffspunkte

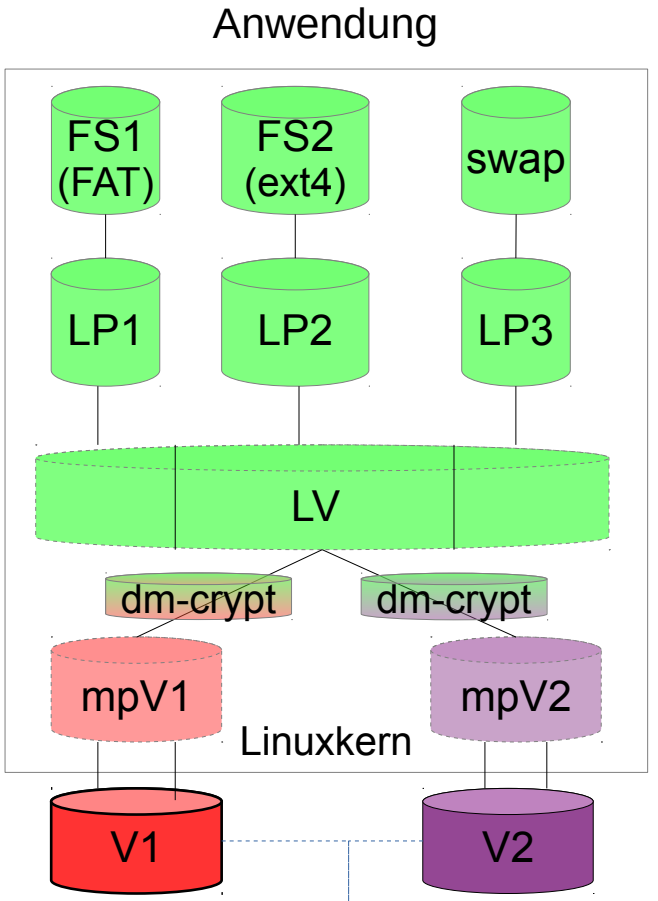
- Storage server
- SAN
- (Server / HV)

gegen Angreifer

- Betreiber von Storage Server und SAN
- Außenseiter

Beispiel: Linux mit dm-crypt & LUKS

- Linux Blockgeräte speichern Daten in gleich großen Blöcken.
- Linux-Blockgerätetreiber schreiben und lesen Blöcke von Blockgeräten
- Linux-Blockgräteteiber beschreiben virtuelle Blockgeräte
- in Linux kann man Blockgeräte(treiber) stapeln
- dm-crypt ist ein Blockgerätetreiber, der gelesene Blöcke entschlüsselt und zu schreibende Blöcke verschlüsselt
- LUKS (Linux Unified Key Setup) beschreibt einen (Klartext) Formatblock für eine verschlüsseltes Blockgerät, er enthält u.a.
  - Benutze Chiffre (einschl. Betriebsmodus)
  - 1 – 8 Kopien des Schlüssel, jede Kopie verschlüsselt mit einem (anderen) Passwort
- Vor der Montage des Blockgerätes muss dem BS ein Passwort, mit dem der Schlüssel im Formatblock verschlüsselt ist, gegeben werden.
- Ende-zu-Ende Verschlüsselung:
  - Schlüssel nur im BS in Klartext lesbar
  - alle Daten, die BS verlassen sind verschlüsselt



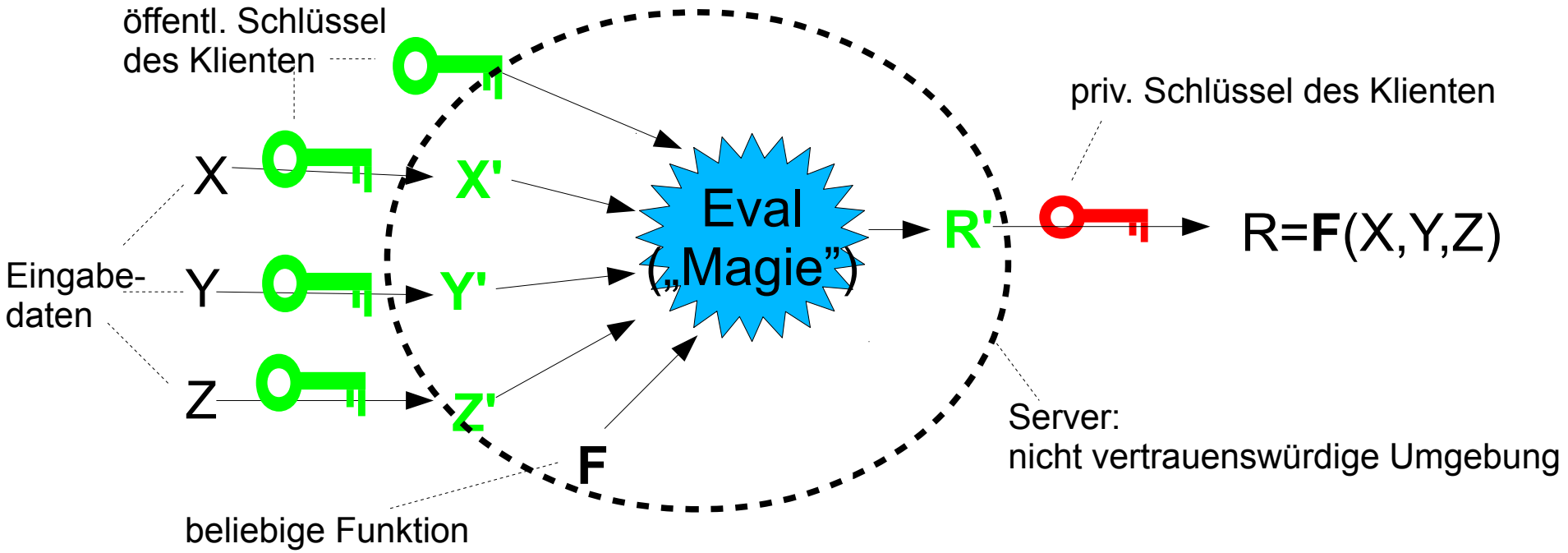
Header: cipher specification, salt, PBKDF iterations, master key digest, ...
Key material 1
Key material 2
Key material 3
...
Key maretial 8
payload encrypted by master key

Sicheres Rechnen

- Kann der Betreiber eines Systems oder einer Cloud meine Berechnungen beobachten?
- Angriffe:
  - Beobachtung (Änderung) des Speichers durch
    - HW Besitzer (mit Zugang zur HW-Konsole) -> RAM
    - Administratoren von HVs -> Gastspeicher
    - BS Administratoren -> Prozessadressraum
  - Beobachtung (Manipulation) von Serviceanfragen
    - HV: Gast-Intercepts/Unterbrechungen
    - BS: Systemrufe/Unterbrechungen
- Absicherung:
  - vollhomomorphe Verschlüsselung
  - HW-Unterstützung für vertrauliches Rechnen

Vollhomomorphe Verschlüsselung

- Berechnung einer beliebigen Funktion auf verschlüsselten Daten
- Craig Gentry: *Fully Homomorphic Encryption Using Ideal Lattices*, STOC'09



Achtung:

- Funktion = logische Gatter (keine Flip-flops!)
- sehr große Zeit- und Speicher(!)komplexität
- Verhinderung von Seitenkanalangriffen: worst case Komplexität

HW Lösungen für vertrauliches Rechnen

- Rechnen mit geschützten Daten
  - nur Code aus geschütztem Bereich und vertrauenswürdige HW oder FW sieht geschützte Daten im Klartext
- alle Daten außerhalb des Caches sind verschlüsselt:
  - IBM: SecureBlue -> Spielekonsole
- Hauptspeicher enthält verschlüsselte und unverschlüsselte Bereiche
  - Intel Security Guard Extensions (SGX)
  - Rick Boivie: *„Secure Blue++: CPU Support for Secure Execution“* IBM research Report RC25287, 2012
- Geschützte Daten im Hauptspeicher durch Zugriffsbeschränkungen geschützt
  - Szefer & Lee: *„Architectural Support for Hypervisor-Secure Virtualization“* in Proceedings of ASPLOS 2012

SGX

- Nicht vertrauenswürdige SW
  - erzeugt Enklave
  - schreibt initiale Daten in Enklave
  - startet Enklave
  - kann nach dem Start Daten in Enklave nicht mehr lesen
- Vertrauenswürdige HW / FW
  - legt Kontrollstrukturen für Enklave an
  - verschlüsselt Daten von gestarteter Enklave
  - kann signierte Messung der Enklavendaten ausgeben