GWDG

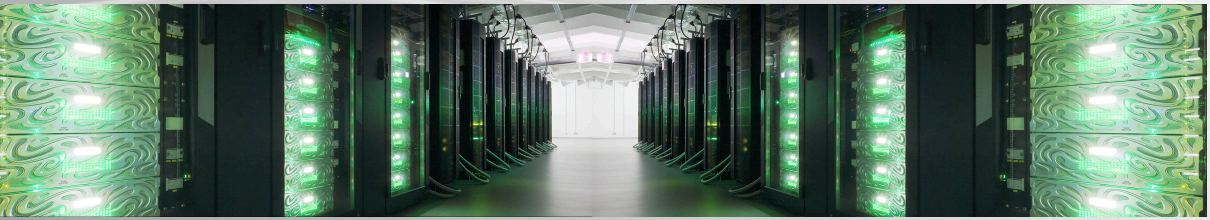
Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen

GWDG

HPC-nativer 5-SAFE Trusted Research Environment (TRE)

für skalierbare Medizinische Datenanalyse

Lars Quentin, Dr. Hendrik Nolte, Prof. Dr. Julian Kunkel



08.09.2025

GMDS 2025

Warum TREs

- **Sicherheit**
  - Datenlokalität
    - Unerlaubtes Teilen
    - Backup-Systeme (z.B. Shadow-Copies)
    - Senden via Slack/Teams/etc
  - Zeitliche Zugriffslimitation
  - Kein Monitoring
  - Laxeres Client-Management
  - Nutzerfehlverhalten
    - Privates Gerät
    - private Dropbox
    - USB-Sticks

Warum TREs

- **Sicherheit**
  - Datenlokalität
    - Unerlaubtes Teilen
    - Backup-Systeme (z.B. Shadow-Copies)
    - Senden via Slack/Teams/etc
  - Zeitliche Zugriffslimitation
  - Kein Monitoring
  - Laxeres Client-Management
  - Nutzerfehlverhalten
    - Privates Gerät
    - private Dropbox
    - USB-Sticks
- **Forschungsreproduzierbarkeit**
  - Kontrollierte Umgebung
  - Container-Images publizierbar

Warum TREs

- **Sicherheit**
  - Datenlokalität
    - Unerlaubtes Teilen
    - Backup-Systeme (z.B. Shadow-Copies)
    - Senden via Slack/Teams/etc
  - Zeitliche Zugriffslimitation
  - Kein Monitoring
  - Laxeres Client-Management
  - Nutzerfehlverhalten
    - Privates Gerät
    - private Dropbox
    - USB-Sticks
- **Forschungsreproduzierbarkeit**
  - Kontrollierte Umgebung
  - Container-Images publizierbar
- **HPC-Resourcen (inkl. GPU-Access)**
  - Big Data, z.B. Genome
  - Machine-Learning Modelle

Warum TREs

- **Sicherheit**
  - Datenlokalität
    - Unerlaubtes Teilen
    - Backup-Systeme (z.B. Shadow-Copies)
    - Senden via Slack/Teams/etc
  - Zeitliche Zugriffslimitation
  - Kein Monitoring
  - Laxeres Client-Management
  - Nutzerfehlverhalten
    - Privates Gerät
    - private Dropbox
    - USB-Sticks
- **Forschungsreproduzierbarkeit**
  - Kontrollierte Umgebung
  - Container-Images publizierbar
- **HPC-Resourcen (inkl. GPU-Access)**
  - Big Data, z.B. Genome
  - Machine-Learning Modelle
- **Zentralisierter Administrativaufwand**

Warum TREs

- **Sicherheit**
  - Datenlokalität
    - Unerlaubtes Teilen
    - Backup-Systeme (z.B. Shadow-Copies)
    - Senden via Slack/Teams/etc
  - Zeitliche Zugriffslimitation
  - Kein Monitoring
  - Laxeres Client-Management
  - Nutzerfehlverhalten
    - Privates Gerät
    - private Dropbox
    - USB-Sticks
- **Forschungsreproduzierbarkeit**
  - Kontrollierte Umgebung
  - Container-Images publizierbar
- **HPC-Resourcen (inkl. GPU-Access)**
  - Big Data, z.B. Genome
  - Machine-Learning Modelle
- **Zentralisierter Administrativaufwand**
- **Legislativer Druck** EHDS, GDNG

Five Safes

1 Safe People

- Vertrauenswürdige Organisatonen und KYC
- Absolviertes Training, Unterzeichnete Nutzervereinbarung

2 Safe Projects

- Kein Zugriff ohne Einverständnis der Data Owner

3 Safe Settings

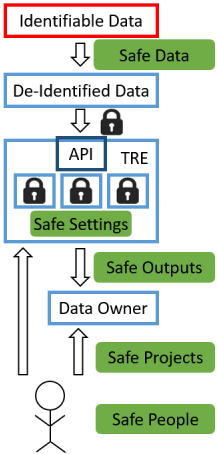
- Daten werden nur innerhalb der TRE verarbeitet
- Kein Internetzugriff, kein Download möglich

4 Safe Outputs

- *Manuelle* Genehmigung von Exporten Prä-Publikation
- Fokus: Keine Reidentifikation; Nur aggregierte Daten

5 Safe Data

- Daten vor Import bereits Pseudonymisiert



Ziele

Technisch

- Zero-Trust
- End-to-end Encryption
- Kollaboratives Arbeiten
- Multi-Layered Security
  1. POSIX
  2. Kryptografisch
- Per-Key-Encryption
- Audit-Log
- Verschlüsselung transparent ggü Nutzer!

Ziele

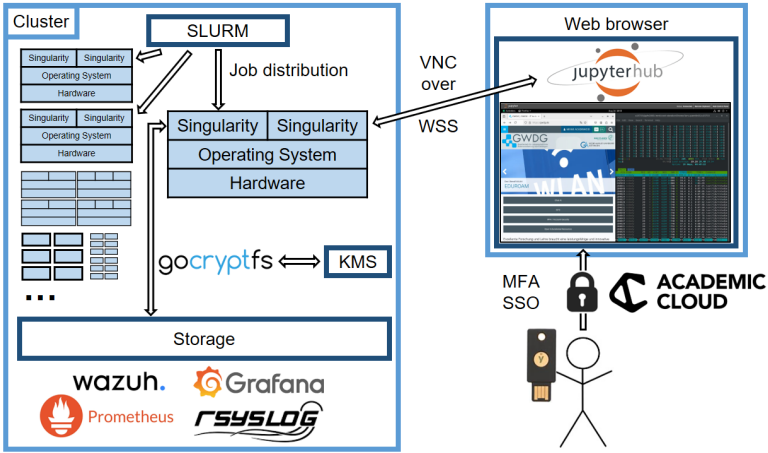
Technisch

- Zero-Trust
- End-to-end Encryption
- Kollaboratives Arbeiten
- Multi-Layered Security
  1. POSIX
  2. Kryptografisch
- Per-Key-Encryption
- Audit-Log
- Verschlüsselung transparent ggü Nutzer!

Organisatorisch

- Nutzbarkeit obwohl Sicher
- Integration in HPC-Infrastruktur
- Nutzung Organisatorischer Resourcen (insb. DIZ, FDPG)
  - Data Ownership
  - Datennnutzungsanträge (inkl. Beratung, UAC)
    $\Rightarrow$ Safe Projects
  - Pseudonymisierung
    $\Rightarrow$ Safe Data
  - Statistical Disclosure Control (SDC)
    $\Rightarrow$ Safe Outputs

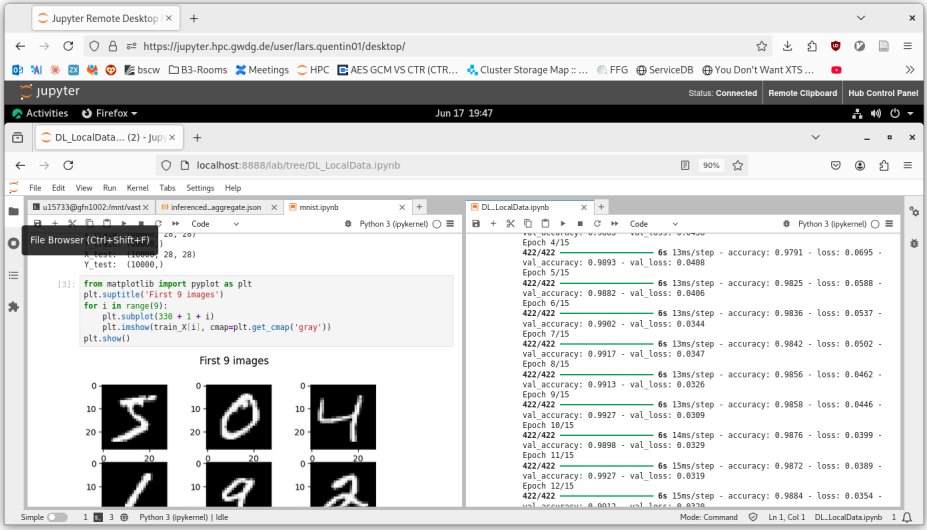
Architektur



Architektur

- JupyterHub-embedded Desktops
  - TigerVNC in Container, websockify TCP Socket ⇒ WSS, NoVNC JS Frontend
  - Somit via AcademicCloud SSO abgesichert
- Virtueller in (Singularity/Apptainer) Container, gestartet als SLURM-Job
- Internet deaktiviert, Software a-priori installiert
- Container read-only, mit Ausnahme von Gocryptfs mount.
  - Gocryptfs ist Dateisystem, was auf Betriebsystemsebene verschlüsselt
  - Fork: Per-File-Verschlüsselung, Zentralisiertes Key-Management, Audit Trails
- Daten nicht exportierbar ohne manuelle Review
- Hauptprobleme:
  1. Sicherer Upload
  2. Sicheres Key-Management

Virtueller Desktop



Zero Trust: Envelope Encryption

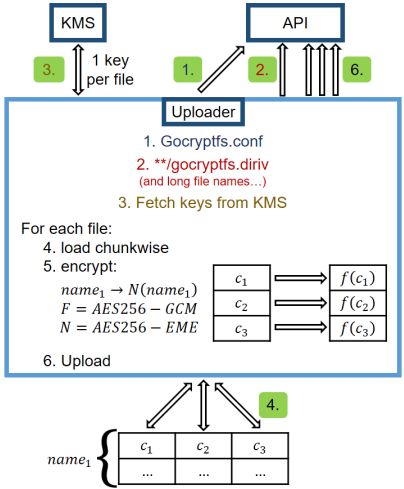
- Jede Datei ist mit einem Data Encryption Key (DEK) verschlüsselt.
- Dieser Key kann in einem Key Management System zentralisiert gespeichert werden.
  - Absicherung via Berechtigungssystem
  - Problem: Datenbank-Leak = Daten-Leak

Zero Trust: Envelope Encryption

- Jede Datei ist mit einem Data Encryption Key (DEK) verschlüsselt.
- Dieser Key kann in einem Key Management System zentralisiert gespeichert werden.
  - Absicherung via Berechtigungssystem
  - Problem: Datenbank-Leak = Daten-Leak
- Stattdessen: Alle DEKs werden verschlüsselt gespeichert
- Jeder Nutzer hat eigenen Key Encryption Key (KEK)
  - DEKs werden mit Public Key des Nutzers verschlüsselt
  - Private Key wird bei Container-Start dem Dateisystem übergeben

Upload

- Für TRE geschriebenen Uploader
  - Beliebig große Dateien
  - Multithreaded Upload und Verschlüsselung
  - Wiederaufnehmbar auf Dateiebene
- POSIX-Backend, keine ext. Dependencies (wie Redis, S3 etc)
- Nutzer live-verschlüsselt jeden chunk beim Upload
- Nutzt gocryptfs-kompatible Verschlüsselung
  - AES256-GCM für Inhalt
  - AES256-EME für Name



Sicherheits- und Disasteranalyse

- **Bekannte Sicherheitslücke:** CVE-Management bereits deployed
- **Datenleck via Zwischenablage:** Deaktiviert im VDE
- **Falsche Nutzung:** Container isoliert; Internet blockiert; Training verpflichtend
- **Falsche POSIX-Rechte:** Doppelte Berechtigungsebene via KMS
- **KMS-Datenbank geleaked:** Schlüssel nicht nutzbar ohne KEKs der Nutzer.
- **Nutzer verliert KEK:** “Tresor”-Funktion basierend auf Shamir’s Secret Sharing

Sicherheitsanalyse

- **Bösartige Forscher:** Keine Möglichkeit abzusichern da Remotezugriff erlaubt
  - Beispiel: Bildschirmaufnahme mit OCR
- **Unbemerktes Teilen von KEKs:** Da Zugriffe von überall erlaubt keine Anomalieerkennung bisher geplant.
- **Zero-Day Exploits:** Angreifer kann dann aus dem Arbeitsspeicher von goecryptfs theoretisch den KEK extrahieren.

Safe People: Nutzervereinbarung

- **Absolviertes Training**

Safe People: Nutzervereinbarung

- **Absolviertes Training**
- **Regeln einhalten**
  - ToS wurden gelesen und verstanden
  - Compliance mit Gesetzen, insb. DSGVO
  - Nur Forschung im Scope des Antrags

Safe People: Nutzervereinbarung

- **Absolviertes Training**
- **Regeln einhalten**
  - ToS wurden gelesen und verstanden
  - Compliance mit Gesetzen, insb. DSGVO
  - Nur Forschung im Scope des Antrags
- **Keine Regeln umgehen**
  - Nicht reidentifizieren
  - Keine eigenen Exportversuche
  - Keine Screenshots, ...

Safe People: Nutzervereinbarung

- **Absolviertes Training**
- **Regeln einhalten**
  - ToS wurden gelesen und verstanden
  - Compliance mit Gesetzen, insb. DSGVO
  - Nur Forschung im Scope des Antrags
- **Keine Regeln umgehen**
  - Nicht reidentifizieren
  - Keine eigenen Exportversuche
  - Keine Screenshots, ...

Safe People: Nutzervereinbarung

- **Absolviertes Training**
- **Regeln einhalten**
  - ToS wurden gelesen und verstanden
  - Compliance mit Gesetzen, insb. DSGVO
  - Nur Forschung im Scope des Antrags
- **Keine Regeln umgehen**
  - Nicht reidentifizieren
  - Keine eigenen Exportversuche
  - Keine Screenshots, ...
- **Vertraulichkeit wahren**
  - Keine Daten teilen
  - Kein Login teilen (selbst wenn beide approved da Audit-Trail)
  - Sichere Hardware (vom AG)
  - Keine Nutzung in öffentlichen Cafes
  - Während offener Session nicht unaufsichtigt

Safe People: Nutzervereinbarung

- **Absolviertes Training**
- **Regeln einhalten**
  - ToS wurden gelesen und verstanden
  - Compliance mit Gesetzen, insb. DSGVO
  - Nur Forschung im Scope des Antrags
- **Keine Regeln umgehen**
  - Nicht reidentifizieren
  - Keine eigenen Exportversuche
  - Keine Screenshots, ...
- **Vertraulichkeit wahren**
  - Keine Daten teilen
  - Kein Login teilen (selbst wenn beide approved da Audit-Trail)
  - Sichere Hardware (vom AG)
  - Keine Nutzung in öffentlichen Cafes
  - Während offener Session nicht unaufsichtigt
- **Benachrichtigung**
  - Sicherheitslücke oder Anomalie entdeckt
  - Nutzer zu viele Rechts
  - Institutswechsel

Andere Safes

- **Safe People: Trainingscurriculum**
  1. HPC Usage Training
  2. Data Awareness Training
  3. TRE Rules and Workflows

Andere Safes

- **Safe People: Trainingscurriculum**
  1. HPC Usage Training
  2. Data Awareness Training
  3. TRE Rules and Workflows
- **Safe data, Safe Projects: Ingress Workflow**
  - Normaler DIZ-Workflow: Opt. Beratung:
    Verfügbarkeitsprüfung, Antragsstellung, UAC, Bereitstellung
  - Rohdaten bleiben bei Datenbesitzer; *Nur* Pseudonymisierte Daten in TRE
    - Unterstützung durch erzwungene Checkliste (4-Augen-Prinzip)

Andere Safes

- **Safe People: Trainingscurriculum**
  1. HPC Usage Training
  2. Data Awareness Training
  3. TRE Rules and Workflows
- **Safe data, Safe Projects: Ingress Workflow**
  - Normaler DIZ-Workflow: Opt. Beratung: Verfügbarkeitsprüfung, Antragsstellung, UAC, Bereitstellung
  - Rohdaten bleiben bei Datenbesitzer; *Nur* Pseudonymisierte Daten in TRE
    - Unterstützung durch erzwungene Checkliste (4-Augen-Prinzip)
- **Safe Outputs: Egress Workflow**
  - Fokus: Keine Reidentifikation möglich, nur nicht-PII DAten
  - Bei Anfrage: Exportordner wird gefreezed, (kryptograischen) Zugriff für Data Owner
    - Unterstützung durch erzwungene Checkliste (4-Augen-Prinzip, PI und Data Owner)

Zusammenfassung

- Erstellung eines HPC-nativen TRE mit End-To-End Verschlüsselung.
  - **Five-Safe Compliant:** Durch Einbindung existierender Resourcen
  - **Souveränität:** Komplette Datenkontrolle bei Datenbesitzer
  - **Zero-Trust:** End-to-End Envelope Encryption Schema; Verschlüsselung auf Dateisystemebene
  - **Nutzerfreundlich:** Grafische Desktops, Kollaborativ, von überall, keine Clientsoftware nötig
  - **High-Performance:** Zugriff auf 100+ Core Maschinen mit aktuellen GPUs

Zusammenfassung

- Erstellung eines HPC-nativen TRE mit End-To-End Verschlüsselung.
  - **Five-Safe Compliant:** Durch Einbindung existierender Resourcen
  - **Souveränität:** Komplette Datenkontrolle bei Datenbesitzer
  - **Zero-Trust:** End-to-End Envelope Encryption Schema; Verschlüsselung auf Dateisystemebene
  - **Nutzerfreundlich:** Grafische Desktops, Kollaborativ, von überall, keine Clientsoftware nötig
  - **High-Performance:** Zugriff auf 100+ Core Maschinen mit aktuellen GPUs
- Erfolgreich absolviert:
  - Initiale Software fertig entwickelt.
  - Masterarbeit mit *allen* technischen Details abgegeben.
    $\Rightarrow$ **Gerne ansprechen wer will!**

Zusammenfassung

- Erstellung eines HPC-nativen TRE mit End-To-End Verschlüsselung.
  - **Five-Safe Compliant:** Durch Einbindung existierender Resourcen
  - **Souveränität:** Komplette Datenkontrolle bei Datenbesitzer
  - **Zero-Trust:** End-to-End Envelope Encryption Schema; Verschlüsselung auf Dateisystemebene
  - **Nutzerfreundlich:** Grafische Desktops, Kollaborativ, von überall, keine Clientsoftware nötig
  - **High-Performance:** Zugriff auf 100+ Core Maschinen mit aktuellen GPUs
- Erfolgreich absolviert:
  - Initiale Software fertig entwickelt.
  - Masterarbeit mit *allen* technischen Details abgegeben.
    $\Rightarrow$ **Gerne ansprechen wer will!**
- Nächsten Schritte:
  - Deployment auf HPC-Knoten
  - Use-Case mit SHIP-Datensatz fertigstellen.