Deckblatt

# Inhaltsverzeichnis

# A Executive Summary

# B Einleitung

Die Sicherung und Wiederherstellung digitaler Daten ist ein zentraler Bestandteil moderner IT-Systeme und ein relevantes Studien- und Forschungsfeld der Informatik. Insbesondere im Kontext zunehmender Systemkomplexität und verteilter Infrastrukturen gewinnt die Frage an Bedeutung, wie Backups zuverlässig, sicher und nachvollziehbar erstellt werden können. Parallel dazu zeigen aktuelle Forschungsergebnisse und Fallstudien, dass Schadsoftware, insbesondere Ransomware, nicht nur produktive Systeme, sondern auch Sicherungsmechanismen selbst ins Visier nimmt. Somit wird die Backup-Integrität zu einem sicherheitskritischen Aspekt, der sowohl theoretisch als auch praktisch untersucht werden muss.

**Kommentiert [SM1]:** Wenn man das so schreibt sollte man Quellen nennen

Vor diesem Hintergrund ist der Aufbau eines Malware-resistenten Backup-Systems von zentraler Bedeutung, das auch im Falle eines kompromittierten Clients oder Servers funktionsfähig bleibt. In diesem Projekt soll daher eine Zero-Trust-Backup-Architektur entwickelt und in Form eines Proof of Concept (PoC) umgesetzt werden.

Kernprinzipien sind dabei:

1. **Vollständig clientseitige Verschlüsselung:**
   Der Backup-Server erhält ausschließlich Ciphertext und besitzt zu keinem Zeitpunkt Kenntnis über Entschlüsselungsinformationen.
2. **Strikte Zugriffstrennung und Rechtebegrenzung:**
   Sowohl auf Client- als auch auf Serverseite ist sichergestellt, dass Kompromittierungen nicht automatisiert zu Datenverlust führen können.
3. **Append-Only-Speicherung (WORM-Prinzip):**
   Gesicherte Daten dürfen nach erfolgter Speicherung nicht gelöscht oder überschrieben werden.
4. **Pull-Prinzip:**
   Der Backup-Server initiiert den Sicherungsprozess, während der Client nicht das Recht besitzt, aktiv auf den Server zu schreiben.

**Kommentiert [SM2]:** @Sven Matzik muss umgesetzt werden

Das Projekt umfasst die Analyse bestehender Lösungen, die Erstellung eines Kriterienkatalogs, die Auswahl geeigneter Werkzeuge und die Umsetzung in einer containerisierten Testumgebung. Die Ergebnisse werden dokumentiert und hinsichtlich ihrer Eignung für eine zukünftige produktive Nutzung bewertet.

# C Grundlagen und Methodik

## 1 Begriffsdefinitionen

- **Malware:** Oberbegriff für *malicious software*, also Schadprogramme, die unerwünschte oder schädliche Funktionen ausführen. Dazu zählen Viren, Würmer, Trojaner, Spyware etc. Eine besonders gefährliche Unterart ist **Ransomware**, die Daten auf einem System **kryptographisch verschlüsselt** oder den Zugriff darauf blockiert, um anschließend ein Lösegeld zu fordern. Ziel eines Ransomware-Angriffs ist es, die **Verfügbarkeit** von Daten zu sabotieren und das Opfer durch Datenverschlüsselung zur Zahlung zu zwingen. (Wichtig: Selbst die Zahlung garantiert nicht, dass die Daten entschlüsselt werden.)
- **Datendiebstahl:** Das unbefugte **Kopieren oder Abziehen vertraulicher Daten**. Ein Angreifer verschafft sich unerlaubt Zugang zu sensiblen Informationen, etwa um sie weiterzuverkaufen, für Identitätsdiebstahl zu missbrauchen oder das Opfer zu erpressen. Im Backup-Kontext bedeutet Datendiebstahl, dass geschützte Backup-Inhalte in falsche Hände geraten.
- **Zero Trust:** Ein Sicherheitsmodell nach dem Prinzip *„Vertraue niemals, verifiziere immer“*. **Zero-Trust-Architektur (ZTA)** verlangt für alle Zugriffe – egal ob innerhalb oder außerhalb des eigenen Netzwerks – eine strikte Identitätsprüfung und Rechtekontrolle, bevor Zugang zu Ressourcen gewährt wird. Nichts und niemand wird per se vertraut; jeder Zugriff wird kontextabhängig authentifiziert und autorisiert. Im Backup-Umfeld heißt das insbesondere: Der Backup-Server wird **nicht als vertrauenswürdig angenommen** und sieht nur verschlüsselte Daten (siehe *clientseitige Verschlüsselung*), und jede Interaktion (z.B. Restore) erfordert erneute Authentifizierung. Dieses *Assume-Breach*-Denken minimiert die Auswirkungen kompromittierter Accounts oder Geräte.
- **Append-Only** (Nur-anfügen-Modus): Betriebsmodus für Datenspeicher/Backups, in dem **einmal gespeicherte Daten nicht mehr gelöscht oder überschrieben** werden können – es kann nur hinzugefügt werden. Dies entspricht dem WORM-Prinzip (Write Once Read Many, *einmal schreiben, vielfach lesen*). WORM bezeichnet Vorkehrungen in der IT, die das Löschen oder Ändern von einmal gespeicherten Daten **dauerhaft ausschließen**, um sie vor Verlust oder Manipulation zu schützen. Solche Daten sind *unveränderlich* (*immutable*). Ein Backup-Repository im Append-Only/WORM-Modus stellt sicher, dass frühere Sicherungsstände erhalten bleiben, selbst wenn ein Angreifer Schreibzugriff erlangt – er könnte dann nur neue Daten anhängen, aber keine alten entfernen oder verschlüsseln.
- **Deduplizierende Sicherung:** Verfahren, bei dem redundante (mehrfach vorhandene) Daten im Backup erkannt und **nur einmal gespeichert** werden. Durch diese *Deduplikation* werden doppelte Datenblöcke eliminiert, bevor sie auf das Backup-Ziel geschrieben werden. Dies reduziert den Speicherbedarf und beschleunigt inkrementelle Backups, da z.B. wiederkehrende Dateien oder unveränderte Bereiche nicht mehrfach übertragen werden. Alle betrachteten Backup-Tools (restic, Kopia, BorgBackup) nutzen interne Block-Deduplikation, was effizientere, platzsparende Sicherungen ermöglicht.

## 2 Projektorganisation und methodisches Vorgehen

Das Projekt wurde in fünf logisch voneinander abgegrenzten Phasen durchgeführt. Diese waren Initialisierung, Anforderungsanalyse, Marktrecherche, Proof of Concept und Ergebnisbewertung.

Die Arbeitsteilung folgte klar definierten Rollen: Sprecher, Koordinator und Mitarbeiter. Dadurch wurden sowohl organisatorische als auch fachliche Zuständigkeiten transparent umgesetzt.

Methodisch stützte sich das Projekt auf eine Kombination aus qualitativen und quantitativen Bewertungsverfahren. Insbesondere kam der Analytic Hierarchy Process (AHP) zum Einsatz, um die Vielzahl heterogener Kriterien zu gewichten und die Entscheidungsfindung objektiv abzusichern. Ergänzend wurden Risikoanalysen sowie experimentelle Evaluierungen im Zuge des PoC durchgeführt.

# D Bedrohungsszenarien und Angriffsvektoren

Backups gelten traditionell als das letzte Sicherheitsnetz der IT-Infrastruktur. Sie sind essenziell, um nach Datenverlusten den Geschäftsbetrieb wiederherzustellen. In modernen Bedrohungslagen hat sich diese Rolle jedoch gewandelt: Backups sind nicht mehr nur der „Retter“, sondern zunehmend das primäre Ziel von Angriffen.

In einer Zero-Trust-Architektur, die davon ausgeht, dass keinem Benutzer und keinem System per se vertraut werden darf, muss das Backup-System gegen zwei Hauptkategorien von Risiken gehärtet werden: den Verlust der Verfügbarkeit (Zerstörung der Daten) und den Verlust der Vertraulichkeit (Datendiebstahl).

## 1 Verlust der Verfügbarkeit

Das primäre Ziel vieler Angreifer – insbesondere im Bereich der Cyberkriminalität und Ransomware – ist es, den maximalen Schaden anzurichten, um die Zahlungsbereitschaft der Opfer zu erzwingen. Wenn ein Unternehmen über funktionierende Backups verfügt, ist es nicht gezwungen, Lösegeld zu zahlen. Daher versuchen Angreifer gezielt, diese Option zu eliminieren.

**Gezielte Ransomware-Angriffe auf Backups**

Moderne Schadsoftware agiert nicht mehr zufällig. Sie ist so programmiert, dass sie aktiv nach Backup-Infrastrukturen sucht. Sobald ein System infiziert ist, scannt die Malware nach verbundenen Speicherorten wie Netzlaufwerken, NAS-Systemen (Network Attached Storage) oder externen Festplatten. Die Strategie ist perfide: Bevor die Produktionsdaten verschlüsselt werden, werden die Backups zerstört. Ohne eine „saubere“ Sicherung steigt der Druck auf das Opfer massiv an, da die Wiederherstellung der Geschäftsprozesse ohne den Decryption-Key der Erpresser unmöglich wird.

Die typischen Angriffspfade in diesem Szenario sind:

- Infektion und Ausbreitung (Lateral Movement): Dringt Malware in einen Server oder Arbeitsplatzrechner ein, versucht sie, alle von dort erreichbaren Speichermedien zu erfassen. Da Backup-Speicher oft als Netzlaufwerke eingebunden sind, werden sie von

der Ransomware wie normale Daten behandelt und verschlüsselt oder gelöscht. Dokumentierte Vorfälle belegen, dass Ransomware-Gruppen spezifisch darauf trainiert sind, NAS-Systeme zu identifizieren und deren Inhalte unwiederbringlich zu löschen.

- Missbrauch von Backup-Software und Credentials: Ein besonders gefährlicher Vektor ist der Missbrauch legitimer Werkzeuge. Angreifer nutzen gestohlene Administrator-Zugangsdaten (Credentials) oder Sicherheitslücken in der Backup-Software selbst (ein prominentes Beispiel hierfür ist eine Schwachstelle in Veeam Backup & Replication im Jahr 2023). Mit administrativen Rechten ausgestattet, loggen sich die Angreifer regulär in die Backup-Konsole ein. Dort können sie Sicherungsjobs manipulieren, Retention-Policies (Aufbewahrungsfristen) verkürzen oder Snapshots manuell löschen. In einigen Fällen verfügt die Ransomware selbst über Module, die API-Befehle an Backup-Lösungen senden, um Löschvorgänge auszulösen.
- Gezielte Löschbefehle durch Hacker oder Insider: Auch ohne spezialisierte Backup-Exploits können Angreifer, die administrativen Zugriff auf die Domäne (z. B. Active Directory) erlangt haben, verheerenden Schaden anrichten. Über Gruppenrichtlinien oder Skripte können Befehle an alle vernetzten Clients gesendet werden, um lokale Schattenkopien (Shadow Copies) oder verbundene Backup-Repositories zu löschen. Zudem fällt in diese Kategorie die Bedrohung durch böswillige Insider: Mitarbeiter mit administrativen Rechten könnten versuchen, Spuren eigener Vergehen zu verwischen, indem sie ältere Backup-Stände, die Beweise enthalten könnten, gezielt vernichten.

Die Konsequenz eines erfolgreichen Angriffs auf die Verfügbarkeit ist der Worst Case: Primärdaten und Backups sind gleichzeitig unbrauchbar. Dies führt zu extremen Ausfallzeiten und kann für datengetriebene Unternehmen existenzbedrohend sein.

**Nicht-böswillige Ursachen für Verfügbarkeitsverlust**

Neben aktiven Angriffen dürfen klassische Risiken nicht ignoriert werden, auch wenn sie in der Sicherheitsanalyse oft als „Basisrisiken“ gelten:

- Hardware-Ausfälle und physische Zerstörung: Festplattendefekte, Brände im Rechenzentrum oder der Diebstahl von Backup-Servern führen ebenso zu Datenverlust. Ein „Single Point of Failure“ (z. B. Aufbewahrung der Backups im selben Brandabschnitt wie die Server) ist hier das größte Risiko.
- Menschliches Versagen: Versehentliches Löschen von Backup-Ordnern oder fehlerhaft konfigurierte Backup-Pläne (z. B. wenn kritische Verzeichnisse gar nicht gesichert werden) sind häufige Fehlerquellen. Oft fallen diese Probleme erst auf, wenn eine Wiederherstellung scheitert.
-

## 2 Verlust der Vertraulichkeit

Backups sind im Wesentlichen ein Spiegelbild aller Daten eines Unternehmens. Oft enthalten sie sogar mehr Informationen als die Live-Systeme, da sie historische Datenarchive und Sicherungen mehrerer Server zentral bündeln. Dies macht sie zu einer wahren Fundgrube für Datendiebe.

Der sogenannte Angriffsbaum unterteilt die Gefahren hierbei in verschiedene Zweige:

**Unbefugter Zugriff und Exfiltration**

Gelingt es einem Angreifer, Zugriff auf den Backup-Speicher (z. B. einen Cloud-Bucket oder einen Dateiserver) zu erhalten, kann er riesige Datenmengen kopieren, ohne sofort bemerkt zu werden. Dies ist Teil der modernen „Double Extortion"-Strategie bei Ransomware:

- Die Daten werden gestohlen (Verlust der Vertraulichkeit).
- Die Daten werden verschlüsselt (Verlust der Verfügbarkeit). Selbst wenn das Opfer über weitere Backups verfügt und nicht für die Entschlüsselung zahlt, können die Kriminellen mit der Veröffentlichung der sensiblen Daten (Kundendaten, Patente, Geschäftsgeheimnisse) drohen.

**Abfangen der Datenübertragung (Man-in-the-Middle)**

Werden Backups über ungesicherte Netzwerke übertragen, könnten Angreifer den Datenstrom mitschneiden („Sniffing"). Hinweis zur Mitigation: Moderne Backup-Tools wie Restic oder Kopia adressieren dieses Risiko standardmäßig durch die Nutzung verschlüsselter Protokolle (TLS/HTTPS, SSH). Zudem setzen diese Tools oft auf eine clientseitige Verschlüsselung. Das bedeutet, die Daten werden bereits auf dem Quellsystem verschlüsselt, bevor sie überhaupt das Netzwerk verlassen. Abgefangene Datenpakete wären für den Angreifer ohne den Schlüssel wertlos.

**Diebstahl von Zugangsdaten und Schlüsseln**

Das stärkste Verschlüsselungskonzept bricht zusammen, wenn die Schlüssel nicht sicher verwahrt werden. Angreifer suchen gezielt nach:

- Passwörtern in ungesicherten Skripten auf dem Quellserver.
- API-Keys für Cloud-Speicher (z. B. AWS S3 Credentials), die versehentlich im Code hinterlegt wurden.
- Konfigurationsdateien der Backup-Software. Besitzt der Angreifer die Schlüssel, kann er die (eigentlich sicher geglaubten) verschlüsselten Backups lesbar machen, selbst wenn er sie nur kopiert hat.

## 3 Die Insider-Bedrohung

Ein oft unterschätztes Risiko sind interne Täter. Ein Administrator mit legitimen Rechten kann Backups nutzen, um unbemerkt Kopien sensibler Datenbanken zu erstellen und diese an Wettbewerber zu verkaufen oder mitzunehmen, wenn er das Unternehmen verlässt. Da Backups oft zentralisiert sind, genügt ein einziger Zugriffspunkt, um Informationen aus verschiedensten Abteilungen abzugreifen.

Die Folgen eines Vertraulichkeitsverlusts sind gravierend: Neben dem Reputationsschaden drohen empfindliche Strafen durch Datenschutzbehörden (DSGVO) und der Verlust von Wettbewerbsvorteilen.

**Weitere Bedrohungen und Schwachstellen**

Abseits der direkten Angriffe auf die Datenintegrität und -vertraulichkeit nennt der Text weitere Flanken, die ein Backup-System angreifbar machen:

- Fehlkonfigurationen: Oft sind es einfache Fehler, die Türen öffnen. Offene Ports ohne Firewall-Schutz, zu großzügig vergebene Dateiberechtigungen (sodass jeder Benutzer Backups lesen oder löschen kann) oder fehlende Verschlüsselung sind Einladungen für

Angreifer. Auch fehlende „Immutability“ (Unveränderbarkeit) ist ein Konfigurationsfehler: Wenn Backups nicht für einen gewissen Zeitraum gegen Überschreiben geschützt sind, hat Ransomware leichtes Spiel.

- Software-Bugs und Zero-Day-Exploits: Backup-Software ist komplexer Code und enthält Fehler. Kritische Sicherheitslücken können es Angreifern ermöglichen, Code mit den (meist sehr hohen) Rechten des Backup-Agenten auszuführen. Regelmäßiges Patch-Management für die Backup-Infrastruktur ist daher genauso wichtig wie für das Betriebssystem selbst.
- Physischer Zugriff: Der Diebstahl von Backup-Tapes oder Festplatten ist ein klassisches Szenario. Ohne Verschlüsselung sind die Daten auf dem Träger sofort lesbar. Doch auch mit Verschlüsselung ist der physische Verlust problematisch, da er die Verfügbarkeit einschränkt, bis Ersatz beschafft ist.
- Kombinationsangriffe: In der Realität treten diese Bedrohungen selten isoliert auf. Ein typischer Angriffsablauf („Kill Chain“) kombiniert mehrere Vektoren: Ein Angreifer nutzt Malware für den initialen Zugang, späht Passwörter aus (Credential Theft), nutzt diese für den Zugriff auf das Backup-System (Vertraulichkeitsbruch) und löscht abschließend die Sicherungen (Verfügbarkeitsbruch), um die Verschlüsselung des Primärsystems vorzubereiten.

# E Konzeption

## 1 Anforderungen an ein Zero-Trust Backup-System

### 1.1 Funktionale Anforderungen

Die funktionalen Anforderungen zielten darauf ab, ein Backup-System zu realisieren, das auch unter realistischen Angriffsbedingungen resilient bleibt. Zentrale Anforderungen waren:

- **Verschlüsselung ausschließlich auf dem Client** mittels starker, authentifizierter Algorithmen (AES-256-GCM bzw. äquivalente Verfahren).
- **Trennung der Benutzerkontexte** auf dem Client in Datenproduzent, Verschlüsselungseinheit und Pull-Benutzer.
- **Inkrementelle Sicherung** kombiniert mit Block-Deduplikation, um große Datenmengen effizient zu behandeln.
- **Pull-Prinzip**, sodass ausschließlich der Server den Datenabruf auslösen darf.
- **Unveränderbare Speicherung**, die das Löschen oder Überschreiben vorhandener Backups verhindert.
- **Wiederherstellbarkeit** in Form granularer und vollständiger Restores.

## 1.2 Nichtfunktionale Anforderungen

Nichtfunktionale Anforderungen umfassten:

- Performance, insbesondere bei großen Dateien und vielen kleinen Dateien.
- Skalierbarkeit im Hinblick auf mehrere Clients und Storage-Backends.
- Compliance-Anforderungen, insbesondere BSI-Grundschutz (INF.13, OPS.1.1.5) und allgemeine Anforderungen an Datensicherung und Wiederanlauf.
- Wartbarkeit und Automatisierbarkeit über CLI, Skripte oder systemd-Timer.

**Kommentiert [SM3]:** Realisierung über Cron-Jobs?

Die Priorisierung erfolgte anhand des MoSCoW-Modells. Sicherheitskritische Merkmale wie Verschlüsselung und Append-Only wurden als **MUSS-Kriterien** definiert.

# 2 Bewertungsmethodik und Marktrecherche

Zur systematischen Bewertung potenzieller Backup-Werkzeuge wurde zunächst ein umfassender Kriterienkatalog entwickelt. Dieser gliedert die Anforderungen des Projekts in vier zentrale Kategorien: Sicherheit, Funktionalität, Performance und Betrieb. Diese Struktur orientiert sich an etablierten Bewertungsmodellen der IT-Sicherheit und gewährleistet, dass sowohl technische Merkmale als auch Aspekte wie Wartbarkeit, Automatisierbarkeit und langfristige Nutzbarkeit berücksichtigt werden. Anschließend wurden die Ausprägungen der Kriterien mithilfe des Analytic Hierarchy Process (AHP) gewichtet. Dieses Verfahren ermöglicht eine transparente und konsistente Priorisierung, was insbesondere bei komplexen Entscheidungssituationen mit mehreren Einflussfaktoren von Vorteil ist. Die Analyse zeigte, dass sicherheitsbezogene Kriterien, insbesondere clientseitige Verschlüsselung, Zugriffstrennung, Manipulationsschutz und Unveränderbarkeit der Backups, mit jeweils 15 % die höchsten Einzelgewichte erhielten und damit den Schwerpunkt der Systembewertung darstellten.

**Kommentiert [SM4]:** Analyse weiter darstellen. Auch die Tabellen aus Bewertungsmethodik hier mit einfügen. Der Inhalt vom Auswahl kann hier auch mit rein.

Im Zuge der Marktrecherche wurden sowohl Open-Source- als auch proprietäre Backup-Lösungen analysiert, darunter Restic, Kopia, BorgBackup, Duplicati, Duplicity und verschiedene kryptografische Zusatzwerkzeuge wie rclone crypt. Die Auswahl erfolgte auf Basis einer Longlist. Durch die Formulierung und Anwendung von K.O.-Kriterien wurden anschließend diejenigen Systeme ausgeschlossen, die grundlegende Anforderungen nicht erfüllten, beispielsweise fehlende clientseitige Verschlüsselung, unzureichende Automatisierbarkeit oder fehlende Unterstützung für inkrementelle Sicherungen. Nach dieser Filterung blieben insbesondere Restic, Kopia und BorgBackup als technisch valide und funktional ausreichend leistungsfähige Shortlist-Kandidaten übrig.

Die anschließende AHP-basierte Bewertung führte zu einem klaren und nachvollziehbaren Ergebnis: Mit 49 von 50 möglichen Punkten erzielte Restic den höchsten Gesamtscore und setzte sich damit deutlich von den anderen Lösungen ab. Mehrere Eigenschaften waren hierfür ausschlaggebend. Erstens bietet Restic eine vollständig clientseitige Verschlüsselung, die sowohl Dateninhalte als auch Metadaten geschützt überträgt und speichert. Zweitens verfügt das Werkzeug über eine hochgradig effiziente Block-Deduplizierung, die insbesondere im Umgang mit großen Datenmengen erhebliche Speicher- und Performancevorteile bietet. Drittens ermöglicht Restic durch den optionalen Betrieb eines REST-Servers im Append-Only-Modus die Umsetzung eines echten WORM-Verhaltens. Dadurch werden Lösch- oder Manipulationsversuche kompromittierter Clients wirkungsvoll verhindert.

Neben den sicherheitstechnischen Gründen überzeugte Restic auch aus betrieblichen und funktionalen Gesichtspunkten. Es verfügt über eine sehr aktive Community, eine solide Dokumentation, regelmäßige Sicherheitsupdates und eine ausgezeichnete Skriptfähigkeit, die eine nahtlose Integration in automatisierte Abläufe ermöglicht. Im Vergleich dazu ist Kopia zwar funktionsreich, jedoch noch weniger etabliert. BorgBackup ist zwar ausgereift, ermöglicht jedoch keinen echten, manipulationssicheren Append-Only-Betrieb.

Damit stellt Restic aus wissenschaftlicher und technischer Perspektive die am besten geeignete Lösung für die Umsetzung eines Zero-Trust-Backup-Systems im Rahmen dieses Projekts dar.

# 3 Systemarchitektur der Zero-Trust-Lösung

Die Systemarchitektur folgt streng dem Zero-Trust-Prinzip. Demnach verfügt jede Komponente ausschließlich über die Berechtigungen, die sie zwingend benötigt.

## 3.1 Clientseitige Architektur

Der Client wurde in drei strikt getrennte Benutzerrollen aufgeteilt:

1. **user**
   Erzeugt Daten, besitzt keinerlei Berechtigungen zum Initialisieren oder Übertragen von Backups.
2. **backup_encoder**
   Darf ausschließlich Daten des Nutzers lesen und verschlüsselte Artefakte erzeugen.
   Keine Schreibrechte an Produktivdaten und kein Zugriff auf Netzwerkressourcen.
3. **Backup_puller**
   Hat nur Leserechte auf das Verschlüsselungsstaging und dient als Ziel der Pull-Operation des Servers.
   Keine Berechtigung zum Lesen von Klartextdaten.

Um eine Kompromittierung einzelner Accounts ohne unmittelbare Auswirkung auf den Backup-Bestand zu ermöglichen, wurden diese Rollen über ACLs exakt voneinander isoliert.

## 3.2 Serverseitige Architektur

Der Server verfügt lediglich über einen dedizierten Benutzer, der die Pull-Operationen ausführt. Er hat keinen Zugriff auf Produktivsysteme, verfügt über keine administrativen Privilegien und hat keinerlei Möglichkeit, Daten zu löschen oder zu überschreiben.

# 4 [Platzhalter] Append-Only-Speicherung auf dem Server

# F Proof of Concept und empirische Validierung

Zur Überprüfung der theoretisch hergeleiteten Architektur wurde ein umfassender Proof of Concept (PoC) durchgeführt. Dabei wurden die Funktionsfähigkeit, die Sicherheitseigenschaften und die Robustheit der entwickelten Zero-Trust-Backup-Lösung unter kontrollierten Bedingungen empirisch evaluiert. Der PoC wurde vollständig containerisiert umgesetzt, um Reproduzierbarkeit, Isolation und eine klare Trennung der Systemkomponenten sicherzustellen. Die Testumgebung bestand aus einem Client-Container, der alle definierten Benutzerrollen einschließlich eines ACL-basierten Rechtemodells implementiert, sowie einem Server-Container, der ausschließlich den Pull-Prozess ausführt und das Restic-Repository im Append-Only-Modus verwaltet.

Die Implementierung folgte wissenschaftlichen Anforderungen an Replizierbarkeit: Alle Konfigurationen und Skripte wurden versioniert, sodass sämtliche Testläufe im gleichen Zustand reproduzierbar ausführbar sind. Zusätzlich wurden sowohl manuelle als auch automatisierte Testszenarien ausgeführt, um verschiedene Nutzungs- und Angriffssituationen realitätsnah abzubilden.

## 1 Testkategorien und durchgeführte Prüfungen

Im Rahmen des PoC wurden mehrere Testkategorien definiert, die jeweils unterschiedliche Aspekte der Backup-Architektur validieren sollten:

**(1) Validierung der clientseitigen Verschlüsselung**

- Überprüfung, dass sämtliche Daten **vor der Übertragung** kryptografisch transformiert werden.
- Kontrolle, dass weder Server noch Pull-Benutzer zu irgendeinem Zeitpunkt Zugriff auf Klartext haben.
- Analyse der erzeugten Artefakte zur Sicherstellung, dass auch **Metadaten** (z. B. Dateinamen) verschlüsselt vorliegen.
- Vergleich der Hashwerte, um Manipulationen ausschließen zu können.

**(2) Prüfung der ACL-Restriktionen und Rollentrennung**

- Test des Lesezugriffs von backup_encoder auf Nutzerdaten, jedoch Verweigerung des Schreibzugriffs.
- Test, dass backup_puller ausschließlich verschlüsselte Staging-Daten lesen kann und keinen Zugriff auf Klartext erhält.
- Versuch, mit falschen Benutzerrollen auf Daten zuzugreifen, um die Robustheit des Rechtemodells sicherzustellen.
- Evaluierung der Isolation zwischen Client-Rollen mit Tools wie getfacl.

**(3) Funktionstest der Pull-Operation**

- Initiierung des Pull-Vorgangs durch den Server via SSH und Schlüssel-basierter Authentifizierung.

- Kontrolle, dass der Client keinerlei aktiven Beitrag zum Übertragungsprozess leistet und somit kein Angriffspunkt entsteht.
- Analyse der übertragenen Daten im Server-Repository, um Konsistenz und Vollständigkeit sicherzustellen.
- Tests fehlerhafter Verbindungen zur Bewertung der Fehlertoleranz des Systems.

**(4) Wiederherstellungstests (Restore)**

- Wiederherstellung einzelner Dateien unterschiedlicher Größe zur Prüfung granularer Restore-Fähigkeit.
- Wiederherstellung ganzer Verzeichnisse zur Analyse des Zeitverhaltens.
- Überprüfung der Integrität wiederhergestellter Dateien durch Hashvergleich.
- Simulation eines vollständigen Datenverlusts auf Clientseite und Wiederaufbau auf Basis der Backups.

**(5) Performance-Tests**

- Sicherung einer künstlich erzeugten **10-GB-Datei**, um Durchsatz und Deduplizierungseffizienz zu messen.
- Testen einer Repository-Struktur mit **mehreren tausend kleinen Dateien** (Kodi-Repositorium), da solche Strukturen oft Performanceprobleme verursachen.
- Messung der Upload-/Download-Raten sowie der CPU- und RAM-Auslastung während der Verschlüsselungsprozesse.
- Vergleich der Performance vor und nach Deduplizierung.

**(6) End-to-End-Testlauf (E2E-Automatisierung)**

- Ausführung eines vollautomatisierten Skripts, das den gesamten Backup-Zyklus abbildet:
    1. Starten der Umgebung
    2. Einrichten der ACLs
    3. Erzeugen von Testdaten
    4. Verschlüsselung
    5. Pull-Vorgang
    6. Einspielen in das Restic-Repository
    7. Restore-Test
    8. Bereinigung der Umgebung
- Dieser Test verifiziert, dass die Backup-Architektur **ohne manuelle Eingriffe lauffähig** ist und Fehler zuverlässig abgefangen werden.

## 2 Ergebnisse der empirischen Validierung

Die empirischen Tests bestätigten die theoretisch angenommene Funktionsfähigkeit der Zero-Trust-Backup-Architektur. Insbesondere ergaben sich die folgenden zentralen Ergebnisse:

1. **Robuste Verschlüsselungsprozesse:**
   Zu keinem Zeitpunkt war Klartext außerhalb des Client-Benutzerkontextes zugänglich. Alle übertragenen Daten lagen vollständig verschlüsselt vor.

2. **Durchsetzungsstarkes Berechtigungsmodell:**
   Sämtliche ACL-Regeln wirkten exakt wie vorgesehen. Unerlaubte Zugriffe führten konsistent zu Permission-Denied-Fehlern, ohne dass sicherheitskritische Informationen offengelegt wurden.

3. **Zuverlässiger Pull-Mechanismus:**
   Der Server konnte eigenständig und ohne Beteiligung des Clients vollständige Sicherungen abrufen. Fehlerhafte Authentifizierungen wurden korrekt abgewiesen.

4. **Stabile Wiederherstellung:**
   Wiederherstellungsvorgänge verliefen fehlerfrei. Integritätsprüfungen der wiederhergestellten Daten ergaben keine Abweichungen zum Ursprungsmaterial.

5. **Überzeugende Performance:**
   Sowohl große Dateien als auch viele kleine Dateien wurden effizient verarbeitet. Die Deduplizierung reduzierte den Speicherbedarf spürbar.

6. **Hohe Wiederholbarkeit:**
   Alle Testszenarien konnten konsistent repliziert werden, was die Robustheit und Verlässlichkeit der Lösung unterstreicht.

7.

## 3 Zusammenfassende Bewertung

Der Proof of Concept zeigt insgesamt, dass die entwickelte Backup-Lösung die Anforderungen eines Zero-Trust-Modells erfüllt und selbst in potenziell fehleranfälligen Szenarien zuverlässig arbeitet. Ihre Architektur ist klar strukturiert, sicherheitlich belastbar und dank der Containerisierung vollständig reproduzierbar. Die Ergebnisse untermauern, dass ein Realbetrieb mit überschaubarem Anpassungsaufwand möglich wäre.

# 4 Finales Backup- und Restore-Prozessdesign

Der finale Prozess wurde in Form eines strukturierten Ablaufs implementiert, der sich in vier Stufen gliedert:

1. **Verschlüsselungsphase**
   backup_encoder transformiert die Nutzerdaten in verschlüsselte Blobs.
   Dies erfolgt mittels Restic, das sämtliche Metadaten und Inhaltsdaten vollständig verschlüsselt, bevor sie das System verlassen.

2. **Staging-Phase**
   Die verschlüsselten Daten werden in ein dediziertes Verzeichnis geschrieben, das ausschließlich von backup_puller gelesen werden kann.
   Dadurch ist selbst bei Kompromittierung eines einzelnen Kontos kein Zugriff auf Klartext möglich.

3. **Pull-Phase**
   Der Server initiiert per SSH-Aufruf den Abruf der verschlüsselten Artefakte.
   Anschließend werden diese Daten in das Append-Only-Repository eingespielt.

4. **Restore-Phase**
   Im Wiederherstellungsfall ruft der Administrator Restic über die erzeugten Snapshots auf und kann einzelne Dateien oder komplette Systeme wiederherstellen.
   Aufgrund der Deduplizierung erfolgen sowohl Restore als auch Backup performanzoptimiert.

Der gesamte Ablauf wurde in einem finalen Skript automatisiert, das sämtliche Schritte ohne manuelle Intervention ausführt. Die Automatisierung minimiert Fehlerquellen und gewährleistet die Wiederholbarkeit.

# G Fazit und Ausblick