

Übungsaufgabe

Sicherheit, Datenschutz und gesellschaftliche
Verantwortung in OS: Zugriffskontrollen,
Privilege, Schutz gegen Daten- und
Identitätsdiebstahl

Universität:	Technische Universität Berlin
Kurs/Modul:	Systemprogrammierung
Erstellungsdatum:	September 6, 2025



Zielorientierte Lerninhalte, kostenlos!
Entdecke zugeschnittene Materialien für deine Kurse:

<https://study.AllWeCanLearn.com>

Systemprogrammierung

Aufgabe 1: Zugriffskontrollen, Privilege und Schutz gegen Daten- und Identitätsdiebstahl

Untersuchen Sie Grundprinzipien der Zugriffskontrolle, Privilege-Ebenen und Maßnahmen gegen Daten- bzw. Identitätsdiebstahl in Betriebssystemen. Formulieren Sie Ihre Antworten klar und knapp.

a) Definieren Sie die Konzepte - DAC (Discretionary Access Control), - MAC (Mandatory Access Control), - RBAC (Role-Based Access Control) und - ABAC (Attribute-Based Access Control). Geben Sie jeweils eine kurze Bezeichnung sowie ein typisches Anwendungsbeispiel an.

b) Szenario-Analyse Gegeben ist ein kleines Dateisystem-Szenario mit Dateien D1 und D2. Benutzerrollen: alice, bob, admin; Rollen: User, Admin. Formulieren Sie eine RBAC-Policy (in natürlicher Sprache bzw. Pseudocode), die folgende Anforderungen erfüllt: - alice soll D1 nur lesend zugreifen können, - bob soll D2 lesen und schreiben dürfen, - admin soll vollen Zugriff (lesen, schreiben, löschen) haben. Verwenden Sie eine klare Policy-Darstellung (Rollen -> Rechte -> Ressourcen).

c) Privilege-Escalation Nennen Sie zwei typische Angriffswege, die Privilege-Escalation ermöglichen (ohne Lösungsschritte auszuführen). Skizzieren Sie zwei Gegenmaßnahmen, die solche Angriffe erschweren (z. B. Capabilities, Dropping of privileges, Namespace-Isolation, sichere Sudo-Konfiguration).

d) Datenschutz und Integrität Skizzieren Sie Mechanismen, die Daten- und Identitätsdiebstahl verhindern helfen. Nennen Sie mindestens drei Maßnahmen (z. B. Verschlüsselung ruhender Daten, Verschlüsselung der Übertragung, Schlüsselmanagement, Multi-Faktor-Authentisierung, Audit-Logging) und erläutern Sie kurz deren Zweck.

Aufgabe 2: Sicherheit, Datenschutz und gesellschaftliche Verantwortung in OS

Untersuchen Sie, wie Betriebssysteme Sicherheit, Datenschutz und gesellschaftliche Verantwortung miteinander verbinden. Bearbeiten Sie die Unteraufgaben in der nachfolgenden Reihenfolge.

a) Audit-Logging und Integrität Skizzieren Sie ein Audit-Logging-Konzept für ein Multi-User-System. Welche Ereignistypen sollten protokolliert werden, welche Anforderungen an Integrität und Aufbewahrung bestehen, und wie verhindern Sie unbefugte Modifikationen der Logs?

b) Zugriffskontrollen und Least Privilege Beschreiben Sie das Prinzip *Least Privilege* im Kontext von Betriebssystemen. Welche konkreten Maßnahmen (z. B. feingranulare Berechtigungen, CAPs, Just-Enough-Administration) unterstützen dieses Prinzip in einer Systemumgebung?

c) Datenschutz vs. Sicherheit Diskutieren Sie typische Zielkonflikte zwischen Datenschutz und Systemsicherheit in OS-Designs. Welche Grundprinzipien helfen, diese Balance zu halten (z. B. Data Minimization, Privacy by Design, Auditability, Transparenz)? Geben Sie jeweils ein praktisches Beispiel aus der Praxis.

d) Gesellschaftliche Verantwortung und Nachhaltigkeit Erläutern Sie, wie sichere und datenschutzfreundliche Betriebssysteme zur gesellschaftlichen Verantwortung beitragen und welche Auswirkungen dies auf Infrastruktur, Energieverbrauch und Wartung hat. Diskutieren Sie kurz Vor- und Nachteile sowie ethische Aspekte.

Lösungen

Lösung zu Aufgabe 1: Zugriffskontrollen, Privilege und Schutz gegen Daten- und Identitätsdiebstahl

a) Definitionen

- DAC (Discretionary Access Control): Zugriffskontrollen, bei denen der Owner bzw. Berechtigungsinhaber die Rechte an Objekten (z. B. Dateien) festlegt bzw. weitervererbt. Typische Umsetzung: UNIX-Dateirechte (Benutzer, Gruppe, Andere) bzw. Dateisystem-ACLs, bei denen der Eigentümer die Verteilungsfreiheit über Rechte hat.
- MAC (Mandatory Access Control): zentrale, policies-gesteuerte Zugriffsentscheidungen unabhängig von den Wünschen einzelner Owner. Objekte besitzen Labels/Sensitivitätsstufen, und Zugriffe erfolgen strikt nach Policy (z. B. SELinux, MLS).
- RBAC (Role-Based Access Control): Zugriffsrechte werden Rollen zugeordnet und Benutzer erhalten Rollen. Die Berechtigungen ergeben sich aus der Rolle, nicht aus individueller Zuweisung. Typische Anwendung: Admin-, User-, Auditor-Rollen mit entsprechenden Rechten.
- ABAC (Attribute-Based Access Control): Zugriff entscheidet sich anhand von Attributen von Benutzer, Ressource, Umgebung (z. B. Abteilung, Standardeil, Zeitpunkt). Hohe Flexibilität bei komplexen Richtlinien.

b) RBAC-Policy (Rollen -> Rechte -> Ressourcen)

- Rollen: User, Admin
- Zuweisung:
 - alice → User
 - bob → User
 - admin → Admin
- Rechte pro Ressource:
 - D1:
 - * User: Lesen
 - * Admin: Lesen, Schreiben, Löschen
 - D2:
 - * User: Lesen, Schreiben
 - * Admin: Lesen, Schreiben, Löschen

plausible Policy-Darstellung (natürlich/richtliniennah):

- Wenn Benutzer die Rolle User tragen, dürfen sie D1 nur lesend accessieren.
- Wenn Benutzer die Rolle User tragen, dürfen sie D2 lesen und schreiben.
- Admin-Rolle hat für D1 und D2 Lese-, Schreib- und Löschrechte.

- Zuweisung: alice hat User-Rechte (D1: nur lesen), bob hat User-Rechte (D2: lesen/schreiben), admin hat Admin-Rechte (vollzugriff).

c) Privilege-Escalation

Zwei typische Angriffswege:

1. Inkorrekte oder missbrauchte SUID-/Set-UID-Binaries (z. B. Programme mit root-Rechten, die ausnutzt werden können, um Privilegien zu erhöhen).
2. Kernel- bzw. Kernel-Modul-/Treiber-Schwachstellen, mit denen ein lokaler Angreifer root-Rechte erlangen kann.

Zwei Gegenmaßnahmen (Skizze):

1. Capabilities und Privilege-Dropping:

- Verwendene POSIX-Capabilities, um privilegierte Operationen auf einzelne Fähigkeiten zu beschränken.
- Nach privilegierten Schritten explizit Privilegien ablegen (z. B. setuid-Binaries so gestalten, dass sie Privilegien nach Ausführung wieder abgeben; `prectl(PR_SET_NO_NEW_PRIVS)).Bounding-Set der Fähigkeiten und strikte Kontrolle der ENV – /PATH – Variablen.`

2. Namespace-Isolation und sichere Konfiguration:

- Einsatz von User-Namespaces bzw. Containerisierung (z. B. Docker/Podman) mit eingeschränkten Rechten.
- Absicherung durch Sicherheitsmodule (SELinux/AppArmor), Seccomp-Filter, minimale Sudo-Konfiguration (mit Protokollierung, zeitlich limitierte Sessions, weniger Umgebungsvariablen).
- Minimierung der Angriffsfläche durch sichere Sudo-/Privilegien-Verwaltung.

d) Datenschutz und Integrität

- Verschlüsselung ruhender Daten (at rest): Schutz vertraulicher Informationen, selbst bei physischem Zugriff auf Speichermedien.
- Verschlüsselung der Übertragung (in transit): Vertraulichkeit und Integrität von Daten bei Netzwerktransporten (TLS/HTTPS etc.).
- Schlüsselmanagement: Sichere Generierung, Verteilung, Rotation und Sperrung von Schlüsseln; zentrale KMS-/HSM-Einbindung.
- Multi-Faktor-Authentisierung (MFA): Erhöht die Sicherheit bei Authentisierung, reduziert Interzeption durch Diebstahl von Credentials.
- Audit-Logging: Nachvollziehbarkeit von Zugriffen und Änderungen; Nachweis der Einhaltung von Richtlinien.

Kurzbegründung der Zwecksetzung: Schutz vor unbefugtem Zugriff, Nachweisbarkeit von Aktivitäten, Minimierung von Schaden bei Kompromittierung.

Lösung zu Aufgabe 2: Sicherheit, Datenschutz und gesellschaftliche Verantwortung in OS

a) Audit-Logging und Integrität

- Audit-Logging-Konzept: Zentrale Erfassung sicherheitsrelevanter Ereignisse in einem Multi-User-System.
- Ereignistypen (Beispiele):
 - Authentifizierung (Erfolg/Fehlschlag)
 - Autorisierungsentscheidungen (Zugriffsversuche auf sensible Ressourcen)
 - Datei-/Objektzugriffe (lesen/schreiben/löschen) auf kritische Dateien
 - System- und Konfigurationsänderungen (z. B. Änderung von Berechtigungen, Kernel-/Dienstdienst-Änderungen)
 - Privilege-Escalations-Versuche
 - Netzwerkverbindungen zu relevanten Diensten
- Integrität und Aufbewahrung:
 - Integrität durch Append-Only-Logs, Sequenznummern, kryptographische Prüfsummen/HMAC, Zeitstempel.
 - Aufbewahrung gemäß gesetzlicher Vorgaben bzw. Compliance (z. B. 90–365 Tage, je nach Anforderung), ggf. Zentralisierung in einem sicheren Log-Server.
- Verhindern unbefugter Modifikation der Logs:
 - Schreibrechte absichern (Logs nur von dedizierten Prozessen beschreibbar).
 - Remote Logging bzw. Write-Once/Immutable-Store (WORM) nutzen.
 - Integritätsprüfungen (Regelmäßige Checks, Hash-Ketten, Signaturen).
 - Zeit-Synchronisation (NTP) und Schutz vor Log-Tampering durch Sicherheitsmodule.

b) Zugriffskontrollen und Least Privilege

- Prinzip des Least Privilege: Benutzer und Prozesse erhalten exakt jene Rechte, die für die Erfüllung der Aufgabe nötig sind; nichts Weiteres.
- Konkrete Maßnahmen:
 - Feingranulare Berechtigungen (ACLs, POSIX-ACLs) statt globaler root-Rechte.
 - Einsatz von Capabilities (Linux) statt vollständiger root-Rechte.
 - Just-Enough-Administration (JEA): zeitlich beschränkte, geteilte administrative Rollen; Multi-User-Administration mit getrennten Privilegien.
 - Privilege Separation: Aufgaben in separate Prozesse/Container, klare Schnittstellen, minimale Privilegien pro Komponente.

c) Datenschutz vs. Sicherheit

- Typische Zielkonflikte:
 - Umfassendes Logging kann Personal- bzw. Nutzungsdaten exponieren; Privacy vs. Auditabilität.
 - Data Minimization kann Sicherheitsforensik erschweren (weniger gespeicherte Details im Falle eines Vorfalls).
 - Verschlüsselung erhöht Komplexität der Schlüsselverwaltung und erschwert forensische Analysen.
- Grundprinzipien, um Balance zu halten:
 - Data Minimization: Erheben und Verarbeiten nur notwendiger Daten.
 - Privacy by Design: Datenschutzgrundprinzipien in Architekturen integrieren.
 - Auditability: Transparente, überprüfbare Auditpfade, ohne unnötige personenbezogene Details.
 - Transparenz: Offenlegung von Datenerfassungs- und Verarbeitungsmustern.
- Praktische Beispiele:
 - Protokollierung von Zugriffen in anonymisierter Form, soweit möglich.
 - Verschlüsselung sensibler Felder in Logs; Zugriffsmonitoring statt vollständiger Offenlegung von Inhalten.

d) Gesellschaftliche Verantwortung und Nachhaltigkeit

- Beitrag sicherer und datenschutzfreundlicher Systeme:
 - Verringerung von Angriffserosionen, Schutz von Nutzerdaten, Vertrauen in digitale Dienste.
 - Weniger Sicherheitsverletzungen bedeuten weniger wirtschaftliche Schäden und bessere öffentliche Sicherheit.
- Auswirkungen auf Infrastruktur, Energieverbrauch und Wartung:
 - Sicherheitsmaßnahmen können zusätzlichen Rechenaufwand verursachen (z. B. Verschlüsselung, MIS- oder Audit-Logging); moderner Hardware und effiziente Implementierung mildern dies.
 - Bessere Wartbarkeit und Updatemechanismen erhöhen langfristig die Lebensdauer von Systemen und reduzieren Ressourcenverbrauch durch stabile, patchbare Systeme.
- Vor- und Nachteile, ethische Aspekte:
 - Vorteile: Stärkere Privatsphäre, Schutz kritischer Infrastrukturen, Compliance mit Rechtsnormen, gesellschaftliches Vertrauen.
 - Nachteile: Höhere Kosten für Implementierung, potenzielle Hürden für kleine Akteure, Risiko von Over-Blocking oder Überwachung.
 - Ethische Aspekte: Gerechtigkeit bei Zugangs- und Nutzungsrechten, Vermeidung von Diskriminierung durch Sicherheitsmechanismen, Transparenz gegenüber Nutzern.