Inhalt

[JSON Web Tokens 1](#_Toc184032280)

[Peer-to-Peer Token-Austausch 1](#_Toc184032281)

[Access Control mit JWT 1](#_Toc184032282)

[Adaptive Multi-Faktor-Authentifizierung (MFA) 2](#_Toc184032283)

[Failsafe-Token für Offline-Zugriff 2](#_Toc184032284)

[Token-Splitting für geteilte Verantwortung 2](#_Toc184032285)

[Passwortlose Authentifizierung 3](#_Toc184032286)

[Selbstheilendes Authentifizierungssystem 3](#_Toc184032287)

[Token mit KI-gestütztem Risiko-Score 4](#_Toc184032288)

[Netzwerksicherheit: Intrusion Detection System (IDS) 4](#_Toc184032289)

[Innovatives Intrustion Detection System mithilfe von maschinellem Lernen 9](#_Toc184032290)

# JSON Web Tokens

## Peer-to-Peer Token-Austausch

Idee: Benutzer können sicher Tokens untereinander übertragen, um Zugang zu Ressourcen zu teilen.

Umsetzung:

Eine Token-Transferfunktion erlaubt es Benutzern, ihre Berechtigungen sicher weiterzugeben (z. B. "Ich erteile diesem Benutzer Leserechte für meine Daten").

Der Transfer wird durch einen digitalen Schlüssel oder ein OTP validiert.

Token-Sichtbarkeit und -Kontrolle durch ein Dashboard.

Vorteil: Ideal für kollaborative Systeme oder geteilte Workflows.

## Access Control mit JWT

Idee: Über custom Claims des JWT können Autorisierungen geregelt werden und so Zugriffskontrolle implementiert werden.

## Adaptive Multi-Faktor-Authentifizierung (MFA)

**Idee:** MFA wird dynamisch basierend auf Risiko und Kontext hinzugefügt oder entfernt.

* **Umsetzung:**
  + Standard-Login erzeugt ein JWT mit Basisberechtigungen.
  + Bei risikobehafteten Aktionen (z. B. Zugriff auf sensible Daten) wird MFA automatisch aktiviert.
  + Das System entscheidet in Echtzeit, wann zusätzliche Sicherheit erforderlich ist.
* **Vorteil:** Bessere Nutzererfahrung durch Reduzierung unnötiger MFA-Eingriffe.

## Failsafe-Token für Offline-Zugriff

**Idee:** Benutzer können spezielle Offline-Tokens erstellen, die eingeschränkten Zugriff bieten, falls keine Internetverbindung besteht.

* **Umsetzung:**
  + Token mit eingeschränkter Funktionalität wird im Voraus generiert (z. B. Zugriff auf lokal gespeicherte Daten).
  + Offline-Tokens haben eine feste Ablaufzeit und spezifische Einschränkungen.
  + Synchronisation erfolgt, sobald die Verbindung wiederhergestellt ist.
* **Vorteil:** Sicherer Zugriff in Situationen ohne Internet, z. B. auf Reisen.

## Token-Splitting für geteilte Verantwortung

**Idee:** Ein Token kann zwischen mehreren Benutzern geteilt werden, wobei alle zustimmen müssen, um kritische Aktionen durchzuführen.

* **Umsetzung:**
  + Ein Haupt-Token wird in Untertokens aufgeteilt, die von verschiedenen Parteien signiert werden müssen.
  + Beispiel: Zwei Manager müssen ihre Tokens verwenden, um eine Auszahlung zu genehmigen.
  + Die Transaktion wird nur ausgeführt, wenn alle Tokens signiert sind.
* **Vorteil:** Ideal für Anwendungen mit geteilten Verantwortlichkeiten wie Finanzen oder Administration.

## Passwortlose Authentifizierung

Idee: Kombinieren Sie JWT mit passwortlosen Login-Mechanismen.

Umsetzung:

Benutzer authentifizieren sich via E-Mail-Link, OTP oder Biometrie.

Nach erfolgreicher Authentifizierung wird ein JWT generiert.

Optionale Gerätebindung erhöht die Sicherheit.

Vorteil: Reduzierte Abhängigkeit von Passwörtern und erhöhte Benutzerzufriedenheit.

## Selbstheilendes Authentifizierungssystem

Idee: Ein Token-System, das bei Sicherheitsproblemen automatisch reagiert und sich "selbst heilt."

Umsetzung:

Erkennung von Sicherheitsproblemen (z. B. Token-Manipulation, Login von verdächtigen Standorten).

Automatische Token-Invalidierung oder temporäre Einschränkung des Kontos.

Benutzer erhalten eine "Wiederherstellungsaktion", um ihre Identität zu verifizieren und Zugriff zurückzugewinnen.

Vorteil: Minimierung des Risikos durch proaktive Sicherheitsmaßnahmen.

## Token mit KI-gestütztem Risiko-Score

Idee: Tokens enthalten einen durch KI generierten Risiko-Score, der in Echtzeit angepasst wird und die Zugriffsmöglichkeiten des Benutzers bestimmt.

Umsetzung:

Integration eines ML-Modells, das Nutzeraktivitäten analysiert (z. B. ungewöhnliche IPs, Login-Zeiten, Gerätetypen).

Der Risiko-Score beeinflusst die Berechtigungen (z. B. Zugriff auf sensiblere Daten nur bei niedrigem Risiko-Score).

Eine Benutzeraktion kann erforderlich sein, um einen hohen Risiko-Score zu senken (z. B. Verifikation per MFA).

Vorteil: Dynamische Sicherheitsmaßnahmen, die sich an reale Bedrohungen anpassen.

# Netzwerksicherheit: Intrusion Detection System (IDS)

Beschreibung: Implementiere ein IDS (z. B. mit Snort) und teste dessen Fähigkeit, Anomalien zu erkennen.

Plan:

Woche 1: Grundlagenrecherche zu IDS und Snort.

Woche 2: Installation und Konfiguration von Snort in einer Testumgebung.

Woche 3: Entwicklung eines Regelwerks für die Angriffserkennung.

Woche 4: Durchführung von Testangriffen (z. B. Port-Scanning, Brute-Force).

Woche 5: Analyse der IDS-Erkennungsrate und Verbesserung des Regelwerks.

Woche 6: Abschlussbericht und Präsentation.

**Detaillierte Beschreibung: Token-Splitting für geteilte Verantwortung**

**Idee:**

Token-Splitting ermöglicht es, einen einzigen Zugriffstoken in mehrere Teile aufzuteilen, die auf verschiedene Benutzer verteilt werden. Sensible Aktionen oder Ressourcen erfordern die Zustimmung aller Parteien, die jeweils einen Teil des Tokens besitzen. Dieses Konzept ist ideal für Szenarien, in denen geteilte Verantwortung oder Mehrparteien-Autorisierung notwendig ist.

**Funktionen:**

1. **Token-Aufteilung:**
   * Ein JWT wird in **Sub-Tokens** aufgeteilt, die spezifische Informationen (z. B. Benutzer-ID, Zeitstempel) und eine Signatur enthalten.
   * Jeder Sub-Token ist nur in Kombination mit den anderen gültig.
2. **Zusammenführung von Sub-Tokens:**
   * Für Aktionen, die geteilte Zustimmung erfordern, müssen alle Sub-Tokens an den Server zurückgesendet und validiert werden.
   * Die Aktion wird nur durchgeführt, wenn alle Teile authentisch und vollständig sind.
3. **Aktionserlaubnis:**
   * Berechtigungen können granular gestaltet werden:
     + **Optional:** Nur ein Teil der Beteiligten muss zustimmen (z. B. 2 von 3).
     + **Vollständig:** Alle Beteiligten müssen zustimmen.
4. **Sicherheitsmaßnahmen:**
   * Sub-Tokens haben Ablaufzeiten.
   * Sie sind kryptographisch gesichert, sodass Manipulation erkannt wird.

**Anwendungsfälle:**

1. **Finanztransaktionen:**
   * Große Summen erfordern die Zustimmung mehrerer Manager.
2. **Ressourcenfreigabe:**
   * Zugriff auf kritische Daten oder Tools nur mit geteiltem Einverständnis.
3. **Kollaborative Prozesse:**
   * Genehmigung von Dokumenten oder Verträgen durch mehrere Parteien.

**Projektplan für Token-Splitting**

**Woche 1: Grundlagen und Planung**

1. **Anforderungen festlegen:**
   * Analyse spezifischer Anwendungsfälle (z. B. Finanzwesen, Team-Collaboration).
   * Entscheidung, ob alle oder nur ein Teil der Token erforderlich sind (Mehrheitsregel).
2. **Technologiestack definieren:**
   * **Backend:** Node.js mit Express oder Python mit FastAPI/Django.
   * **Datenbank:** PostgreSQL oder MongoDB für Token- und Benutzerverwaltung.
   * **Kryptographie:** JWT-Bibliothek (z. B. jsonwebtoken) und HMAC/Asymmetrische Signaturmethoden.
3. **Prototyp-Setup:**
   * Basis-JWT-System einrichten (Login, Token-Generierung).
   * Docker-basierte Entwicklungsumgebung einrichten.

**Woche 2: Token-Aufteilung entwickeln**

1. **Aufteilungslösung implementieren:**
   * API-Endpunkt erstellen, um ein Haupt-Token in mehrere Sub-Tokens aufzuteilen.
   * Sub-Tokens enthalten:
     + Benutzer-Identifikation.
     + Berechtigungsinformationen.
     + Signatur des Haupt-Tokens.
2. **Sicherheitsmaßnahmen integrieren:**
   * Ablaufzeit für jedes Sub-Token.
   * Überprüfung der kryptographischen Integrität (z. B. HMAC-Signaturen).
3. **Datenbankmodell:**
   * Schema für Sub-Tokens erstellen:
     + Referenz zum Haupt-Token.
     + Status (genutzt/aktiv/verfallen).

**Woche 3: Zusammenführung und Validierung**

1. **Zusammenführungs-API erstellen:**
   * Endpunkt entwickeln, der mehrere Sub-Tokens entgegennimmt und deren Integrität überprüft.
   * Validierungslogik:
     + Alle Teile müssen übereinstimmen.
     + Ablaufzeiten und Berechtigungen müssen stimmen.
2. **Mehrheitsregel-Optionen:**
   * Mechanismus implementieren, der Aktionen bei Zustimmung eines Teils der Beteiligten erlaubt (z. B. 2 von 3 Sub-Tokens).
3. **Erfolg und Fehlerhandling:**
   * Rückmeldung für Benutzer (z. B. „Aktion genehmigt“, „Token fehlt“).
   * Sicherheitsmaßnahmen gegen Manipulation (z. B. Replay-Angriffe).

**Woche 4: Frontend und Benutzererfahrung**

1. **Dashboard-Design:**
   * Übersicht über aktive und geteilte Tokens.
   * Visualisierung der Zustimmungsstatus (z. B. „2 von 3 zugestimmt“).
2. **Token-Aufteilung und -Zusammenführung:**
   * Oberfläche für Benutzer, um Token aufzuteilen.
   * Benutzerfreundlicher Workflow für die Zustimmung (z. B. Klick auf „Genehmigen“-Button).
3. **Benachrichtigungen:**
   * E-Mail/Push-Benachrichtigungen bei anstehenden Aktionen, die Zustimmung erfordern.

**Woche 5: Sicherheit und Tests**

1. **Angriffsvektoren absichern:**
   * Replay-Schutz für Sub-Tokens.
   * Schutz gegen Brute-Force-Versuche (Rate Limiting).
   * Token-Encryption, um sensible Informationen zu schützen.
2. **Testfälle:**
   * Unit-Tests für Aufteilung, Zusammenführung und Validierung.
   * Szenarien testen:
     + Fehlende oder abgelaufene Sub-Tokens.
     + Manipulierte Sub-Tokens.
3. **Load-Tests:**
   * Systemverhalten bei hohem Anfragenvolumen prüfen.

**Woche 6: Optimierung und Deployment**

1. **System-Feinschliff:**
   * Performance-Optimierungen (z. B. Caching von häufig genutzten Tokens).
   * UI-Verbesserungen basierend auf Feedback.
2. **Deployment:**
   * Bereitstellung auf einer Cloud-Plattform (z. B. AWS, Heroku, Azure).
   * HTTPS einrichten, um sichere Kommunikation zu gewährleisten.
3. **Dokumentation:**
   * API-Dokumentation erstellen.
   * Benutzeranleitung für das Token-Splitting-System.

**Technologische Herausforderungen und Lösungen**

* **Herausforderung:** Synchronisation der Sub-Tokens bei parallelen Anfragen.  
  **Lösung:** Mutex-Mechanismen oder Datenbank-Locking verwenden.
* **Herausforderung:** Sicherheit gegen Token-Manipulation.  
  **Lösung:** Kryptographische Signaturen und Ablaufzeiten nutzen.
* **Herausforderung:** Benutzerfreundlichkeit in komplexen Prozessen.  
  **Lösung:** Intuitive Benutzeroberfläche mit klaren Statusanzeigen entwickeln.

**Ergebnis:**

Das Endprodukt ist ein hochsicheres und flexibles Token-Splitting-System, das ideal für Anwendungen mit geteilten Verantwortlichkeiten ist. Dieses System kann in Branchen wie Finanzdienstleistungen, Gesundheitswesen oder Team-Collaboration-Tools integriert werden, wo maximale Sicherheit und Zusammenarbeit erforderlich sind.

# Innovatives Intrustion Detection System mithilfe von maschinellem Lernen

Quellen:

Performance analysis of machine learning models for intrusion detection system using Gini Impurity-based Weighted Random Forest (GIWRF) feature selection technique (<https://cybersecurity.springeropen.com/articles/10.1186/s42400-021-00103-8>)

Machine learning-based network intrusion detection for big and imbalanced data using oversampling, stacking feature embedding and feature extraction (<https://journalofbigdata.springeropen.com/articles/10.1186/s40537-024-00886-w>)

Datasets: **KDD Cup 1999 Data**

**Projektplan für die Implementierung eines innovativen Intrusion Detection Systems (IDS)**

Das Ziel dieses Projekts ist die Entwicklung eines innovativen Intrusion Detection Systems, das traditionelle Mustererkennungsmethoden durch moderne Technologien wie maschinelles Lernen (ML), Anomalieerkennung und adaptive Sicherheitsmechanismen erweitert. Das System soll in der Lage sein, sowohl bekannte als auch unbekannte Bedrohungen effektiv zu erkennen und darauf zu reagieren.

**Merkmale des innovativen IDS:**

1. **Hybride Bedrohungserkennung:**
   * Kombination aus signaturbasierten Methoden (für bekannte Bedrohungen) und anomaliemodellbasierten Methoden (für unbekannte Angriffe).
2. **Echtzeit-Datenverarbeitung:**
   * Verwendung von Technologien wie Apache Kafka zur Verarbeitung großer Datenströme.
3. **Adaptives Lernen:**
   * Einsatz eines ML-Modells, das aus Netzwerkdaten kontinuierlich lernt.
4. **Verhaltensbasierte Analyse:**
   * Analyse des Benutzer- und Netzwerkverhaltens, um verdächtige Aktivitäten zu erkennen.
5. **Automatisierte Reaktion:**
   * Integration eines Systems zur automatisierten Abschottung betroffener Systeme oder Benachrichtigung des Administrators.

**Projektplan über 6 Wochen**

**Woche 1: Planung und Grundlagen**

1. **Anforderungen definieren:**
   * Umfang und Funktionen des IDS klären.
   * Anwendungsfälle spezifizieren (z. B. Netzwerkangriffe, Insider-Bedrohungen).
2. **Technologiestack auswählen:**
   * **Backend:** Python (FastAPI, Flask) oder Go.
   * **Datenverarbeitung:** Apache Kafka für Streaming.
   * **ML-Framework:** TensorFlow oder PyTorch.
   * **Datenbank:** Elasticsearch für Log-Speicherung und Abfragen.
3. **Prototyp-Setup:**
   * Entwicklungsumgebung konfigurieren (Docker, Kubernetes für Skalierbarkeit).
   * Logging-System einrichten (z. B. ELK-Stack: Elasticsearch, Logstash, Kibana).

**Woche 2: Datenerfassung und Verarbeitung**

1. **Netzwerk-Traffic überwachen:**
   * Sensoren zur Erfassung von Datenpaketen einrichten (z. B. Zeek oder Suricata).
   * Protokolle und Metriken festlegen (z. B. IP-Adressen, Ports, Datenpaketgrößen).
2. **Streaming-Datenpipeline:**
   * Aufbau eines Systems, das Netzwerkdaten in Echtzeit verarbeitet (Apache Kafka oder RabbitMQ).
3. **Datenvorverarbeitung:**
   * Implementierung eines Moduls zur Bereinigung und Normalisierung der Daten.
   * Extraktion relevanter Merkmale (z. B. Anzahl der Verbindungen pro Sekunde).

**Woche 3: Implementierung der Bedrohungserkennung**

1. **Signaturbasierte Erkennung:**
   * Aufbau einer Datenbank mit bekannten Bedrohungssignaturen (z. B. Snort-Regeln).
   * API zur Erkennung von Angriffen basierend auf diesen Signaturen.
2. **Anomalieerkennung:**
   * Entwicklung eines ML-Modells zur Erkennung von Anomalien.
   * Training des Modells auf historischen Netzwerkdaten:
     + Clustering-Algorithmen (z. B. Isolation Forest).
     + Deep-Learning-Ansätze für komplexe Muster.
3. **Testen und Validieren:**
   * Evaluation der Genauigkeit der Erkennung gegen simulierte Angriffe.

**Woche 4: Reaktionsmechanismen**

1. **Automatisierte Aktionen:**
   * Implementierung von Skripten zur automatischen Blockierung (z. B. Firewall-Regeln, Benutzerlogouts).
   * Integration eines Warnsystems (E-Mails, Slack-Benachrichtigungen).
2. **Erweiterte Alarmierung:**
   * Dashboard für die Visualisierung von Angriffen und Anomalien (z. B. Grafana oder Kibana).
   * Echtzeit-Benachrichtigungen bei erkannten Bedrohungen.
3. **Simulationstests:**
   * Durchführung von Angriffsszenarien (z. B. DDoS, Port Scanning).
   * Überprüfung, ob Reaktionen korrekt ausgelöst werden.

**Woche 5: Optimierung und Sicherheit**

1. **Feintuning des ML-Modells:**
   * Optimierung der Parameter des Modells für höhere Genauigkeit.
   * Einsatz von Cross-Validation und Hyperparameter-Tuning.
2. **Skalierung:**
   * Sicherstellen, dass das System auch bei hohem Datenaufkommen stabil bleibt.
   * Load-Tests durchführen.
3. **Sicherheitsmaßnahmen:**
   * Verschlüsselung der Kommunikationskanäle (z. B. TLS).
   * Schutz vor Angriffen auf das IDS selbst (z. B. Input-Sanitization).

**Woche 6: Abschluss und Deployment**

1. **Tests und Dokumentation:**
   * End-to-End-Tests für verschiedene Szenarien.
   * Erstellung einer detaillierten Dokumentation für Entwickler und Administratoren.
2. **Deployment:**
   * Bereitstellung des Systems in einer Testumgebung.
   * Deployment in einer produktionsähnlichen Umgebung.
3. **Schulung und Übergabe:**
   * Schulung für Administratoren, wie sie das IDS überwachen und konfigurieren können.
   * Übergabe der fertigen Software.

**Ergebnis:**

Nach 6 Wochen wird ein innovatives IDS bereitstehen, das:

* Echtzeit-Bedrohungen erkennt und darauf reagiert.
* Sich an neue Bedrohungen durch adaptives Lernen anpasst.
* Skalierbar und sicher ist.

Dieses System eignet sich für Organisationen, die ihre Netzwerksicherheit proaktiv überwachen und unbekannte Angriffe erkennen möchten.