



Angewandte Computer- und Biowissenschaften



**HOCHSCHULE
MITTWEIDA**
University of Applied Sciences

Evaluierung von KI-basierten Modellen zur automatisierten Schwachstellenanalyse im Rahmen von Penetrationstests

Katharina Erler

[hs-mittweida.de](https://www.hs-mittweida.de)

Agenda

- ① Problem & Motivation & Ziel der Arbeit
- ② Grundlagen
- ③ Methodik
- ④ Praxistests & Ergebnisse
- ⑤ Vergleich & Bewertung
- ⑥ Fazit & Ausblick

Problem & Motivation

Problem:

- Pentests: teuer & zeitaufwendig
- Wachsende Angriffsflächen (Cloud, Microservices, DevOps)

Motivation:

- Prozesse automatisieren
- Abläufe beschleunigen

Ziel der Arbeit

Forschungsfrage

Wie können KI-gestützte Modelle typische Pentest-Aufgaben unterstützen, beschleunigen oder automatisieren?

Ziele:

- Realitätsnahe Praxistests durchführen
- Stärken & Schwächen vergleichen
- Bewertungsrahmen entwickeln

2. Grundlage

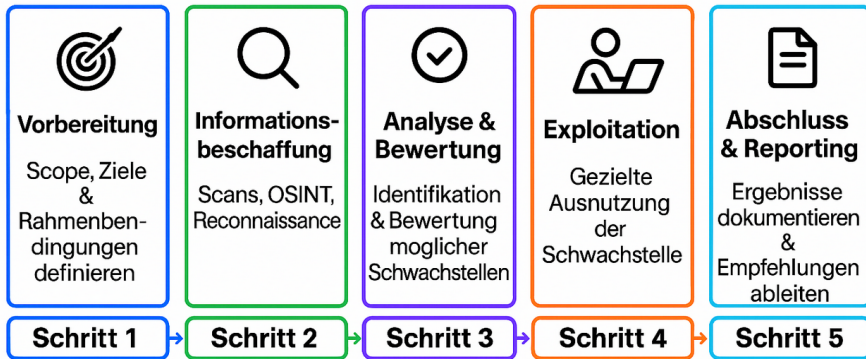
Grundlage 1/3

Was sind Penetrationstests?

- **Definition:** Simulation realer Angriffe auf IT-Systeme
- **Zweck:** Schwachstellen frühzeitig finden & bewerten
- **Arten:**
 - ▶ Black-Box (keine Vorkenntnisse)
 - ▶ Grey-Box (teilweise Infos)
 - ▶ White-Box (vollständige Infos)

Grundlage 2/3

Pentest-Phasen (BSI-Modell)



Grundlage 3/3

KI im Pentest-Kontext

- **Künstliche Intelligenz (KI)**
 - ▶ Muster erkennen
 - ▶ Prozesse automatisieren
- **Large Language Models (LLMs)**
 - ▶ Exploit-Ideen & Payloads generieren
 - ▶ Automatisierte Dokumentation
- **Schwächen**
 - ▶ Halluzinationen (falsche, aber plausible Antworten)
 - ▶ Begrenztes Kontextfenster

3. Methodik

Welche Tools wurden getestet?

- **RamiGPT**
 - ▶ Fokus: Privilege Escalation (Linux/Windows)
 - ▶ Kombination von KI-Logik & klassischen Tools (LinPEAS, BeRoot)
- **PentestGPT (CLI)**
 - ▶ Open Source (Projekt GreyDGL)
 - ▶ Strukturierte Workflows, textbasiert in der Konsole
- **PentestGPT (Web)**
 - ▶ Kommerziell (HackerAI LLC)
 - ▶ Dialogorientiert, direkt im Browser nutzbar

Alle getesteten Tools nutzen GPT-LLMs

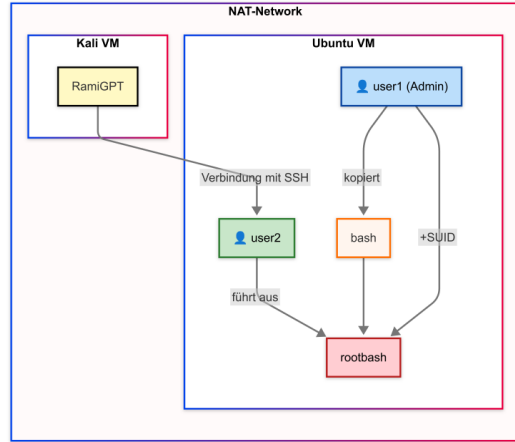
4. Praxistests: RamiGPT-Privilege Escalation

Was ist Privilege Escalation?

- **Ziel:** Unberechtigter Root/Admin-Zugriff
- Post-Exploitation-Phase
- **Typische Techniken (nach MITRE ATT&CK):**
 - ▶ Exploitation von Systemschwachstellen (T1068)
 - ▶ Missbrauch von Sudo/SetUID (T1548)
 - ▶ Manipulation von Zugriffstokens (T1134)
 - ▶ Nutzung gültiger, privilegierter Accounts (T1078)

RamiGPT – Szenario

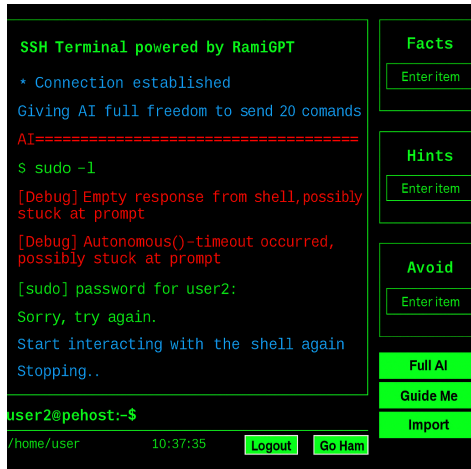
- **Ziel:** Root-Rechte über rootbash
- **Setup:** Ubuntu-VM mit absichtlich fehlerhafter SetUID



Full-AI-Modus (Fehlschlag)

- Startet sudo-Befehle
- Bleibt bei Passwort hängen
- rootbash nicht erkannt

Ergebnis: Full-AI scheitert



The screenshot displays the 'Full AI' interface. On the left, a terminal window titled 'SSH Terminal powered by RamiGPT' shows the following text: 'Connection established', 'Giving AI full freedom to send 20 comands', 'AI=====', 's sudo -l', '[Debug] Empty response from shell,possibly stuck at prompt', '[Debug] Autonomous()-timeout occurred, possibly stuck at prompt', '[sudo] password for user2:', 'Sorry, try again.', 'Start interacting with the shell again', 'Stopping..', and 'user2@pehost:~\$'. On the right, there is a sidebar with buttons for 'Facts', 'Hints', 'Avoid', 'Full AI' (highlighted in red), 'Guide Me', and 'Import'. Each of the first three buttons has an 'Enter item' button below it. At the bottom of the terminal, there is a status bar showing '/home/user', the time '10:37:35', and 'Logout' and 'Go Ham' buttons.

Halb-automatischer Ablauf (1/2)

- Nutzerhilfe: `ls` → Kontext
- RamiGPT findet rootbash
- Führt aber nicht selbst aus

Ausgabe nach `find / -perm /4000`

```
$ ls
rootbash snap
Giving AI full freedom to send 20 commands
AI
request#1=====
$ find / -perm /4000 2>/dev/null
[Debug] Empty response from shell, possibly stuck at prompt
[Debug] Autonomous() - timeout occurred, possibly stuck at prompt

/usr/bin/sudo
/usr/bin/passwd

/snap/core22/2045/usr/bin/passwd
/snap/core22/2045/usr/bin/su
/snap/core22/2045/usr/bin/sudo

/home/user2/rootbash

$/bin/sh: 5: !/bin/sh: not found
$ uid=1001(user2) gid=1001(user2) groups=1001(user2),100(users)
$
Stopping..
```

Halb-automatischer Ablauf (2/2)

- Nutzer: `ls -la` → Datei erkannt
- Tool startet `./rootbash`
- Ergebnis: Root-Rechte (`uid=0/root`)

Fazit: Erfolg nur mit Nutzerhilfe

4. Praxistests: PentestGPT (CLI)

Grundlagen OWASP Top 10 (A01–A03)

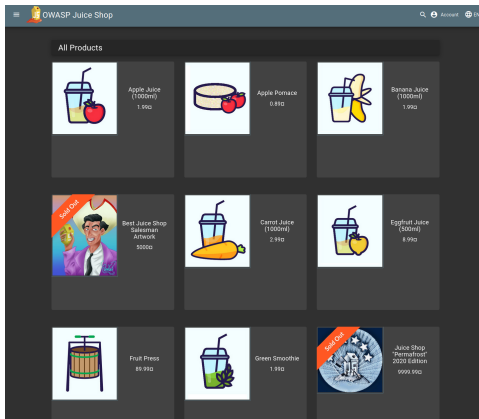
- **A01:** Zugriffskontrollen umgehen (JWT, IDOR)
- **A02:** Schwache / unsichere Kryptografie
- **A03:** Unsichere Eingaben (SQLi, XSS)

OWASP Top 10 – 2021

A01:2021	Broken Access Control
A02:2021	Cryptographic Failures
A03:2021	Injection
A04:2021	Insecure Design (New)
A05:2021	Security Misconfiguration
A06:2021	Vulnerable and Outdated Components
A07:2021	Identification and Authentication Failures
A08:2021	Software and Data Integrity Failures
A09:2021	Security Logging and Monitoring Failures
A10:2021	Server-Side Request Forgery

PentestGPT (CLI) – Szenarien

- **Zielsystem:** OWASP Juice Shop
- **Interaktion:**
 - ▶ Prompt (z. B. SQLi)
 - ▶ CLI-Befehl next
 - ▶ Ausführung im Zielsystem
 - ▶ Feedback & nächsten Schritt



Test – Exploit-Vorschlag

- Vorschläge für konkrete Exploits (z. B. SQLi)
- Liefert fertige Payloads wie
' OR '1'='1'

```
**Possible Tasks:**  
4. Exploit Identification -  
  4.1 Test for Reflected XSS in Customer Feedback Feature -  
  4.2 Test for Stored XSS in Customer Feedback Feature -  
  4.3 Test for SQL Injection in Customer Feedback Feature -  
  4.4 Evaluate Access Controls in Customer Feedback Feature -  
5. Troubleshooting and Resolution -  
  5.1 Diagnose and Resolve Tool Malfunction -  
    5.1.1 Review Tool Configurations -  
    5.1.2 Examine System and Tool Logs -  
    5.1.3 Verify Connectivity and Dependencies -  
    5.1.4 Seek Additional Resources or Support -  
  
**Favored Sub-task**: 4.3 Test for SQL Injection in Customer Feedback Feature -  
  
**Explanation**:  
Manually input typical SQL injection payloads such as ' OR '1'='1' or ' OR 1=1 --'  
into the input fields in the customer feedback form. Observe responses and any changes in  
application behavior, such as retrieving more data than expected, to identify if SQL  
injection vulnerabilities exist.
```

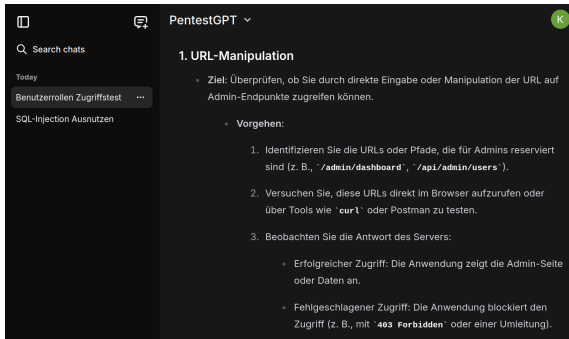
PentestGPT (CLI) – Ergebnisse

- **Stärken:**
 - ▶ valide Payloads
 - ▶ strukturierte Tests
- **Schwächen:**
 - ▶ keine Automatisierung
 - ▶ Kein Reporting-Export, nur Logfiles
- **Fazit:** gutes Assistenztool für Ausbildung & strukturierte Tests

4. Praxistests: PentestGPT (WEB)

PentestGPT (Web) – Szenarien

- **Zielsystem:** OWASP Juice Shop
- **Schwachstellen:** A01–A03
- **Interaktion:**
 - ▶ Chat-ähnlich (Dialog)
 - ▶ Prompts DE/EN
 - ▶ Ad-hoc-Analysen im Browser



PentestGPT (Web) – Ergebnisse

- **Stärken:**

- ▶ Schnelle & präzise Antworten (inkl. Exploits)
- ▶ Keine Installation, einfache Bedienung
- ▶ Kontextsensitiv (Deutsch/Englisch)

- **Schwächen:**

- ▶ Keine Automatisierung
- ▶ Kein Reporting (nur Chat)
- ▶ Voller Funktionsumfang kostenpflichtig

- **Fazit:** Praktisch für schnelle Analysen, weniger für strukturierte Tests

Vergleich & Evaluierung der Tools

Vergleich der Tools

Bewertung nach 6 Kriterien (0–2 Punkte, max. 12):

Kriterium	RamiGPT	PentestGPT (CLI)	PentestGPT (Web)
Schwachstellenabdeckung	●	●	●
Exploit-Vorschlag	●	●	●
Automatisierungsgrad	●	●	●
Kontextverständnis	●	●	●
Reporting & Protokollierung	●	●	●
Kosten-Nutzen-Verhältnis	●	●	●
Gesamtpunkte	5	8	7

Ergebnisse im Überblick

- **RamiGPT** → unreif, nur für Privilege Escalation interessant
- **PentestGPT (CLI)** → methodisch klar, gute Payloads, aber rein manuell
- **PentestGPT (Web)** → schnell & flexibel, aber limitiert ohne Automatisierung

Fazit & Ausblick

Fazit

- **KI = Unterstützung, kein Ersatz**
- **Nutzen:**
 - ▶ Effizienzsteigerung bei Routineaufgaben
 - ▶ Unterstützung in Ausbildung & Training
- **Grenzen:**
 - ▶ Keine vollständige Automatisierung
 - ▶ Grenzen in komplexen Szenarien

Ausblick

- **Integration in DevSecOps**
 - kontinuierliche Sicherheitsprozesse
- **Technische Weiterentwicklung**
 - größere Kontexte, spezialisierte Modelle,
- **Ausbildung**
 - KI als interaktiver Trainingspartner
- **Langfristig**
 - KI erweitert Pentests
 - skalierbar & adaptiv

Quellen / Tools

- ① RamiGPT (GitHub): <https://github.com/M507/RamiGPT>
- ② PentestGPT – CLI (GitHub): <https://github.com/GreyDGL/PentestGPT>
- ③ PentestGPT – Web: <https://pentestgpt.ai/>
- ④ OWASP Juice Shop (GitHub): <https://github.com/juice-shop/juice-shop>
- ⑤ Burp Suite (PortSwigger): <https://portswigger.net/burp/documentation/desktop/getting-started/download-and-install>

Vielen Dank

Katharina Erler



**HOCHSCHULE
MITTWEIDA**
University of Applied Sciences

Hochschule Mittweida
University of Applied Sciences
Technikumplatz 17 | 09648 Mittweida
Angewandte Computer- und
Biowissenschaften

hs-mittweida.de