



# Abgabedokument Lab1

## Einführung in Security

194.157 – 2024 W

20. Dezember 2024

Team 44

Name	MatrNr.
Kevin Csele	12122544
Clemens Schneider	MATRIKELNUMMER
Luka Twaroch	MATRIKELNUMMER
Wen Long Zhou	MATRIKELNUMMER
Ramin Shaikh	12123657

# Inhaltsverzeichnis

<b>1</b>	<b>Der Service war auch schon besser ...</b>	<b>5</b>
1.1	Achtung! Streng geheim! . . . . .	5
1.2	Eine schräge Nummer . . . . .	5
1.3	Was letzte Preis? . . . . .	5
1.4	IBANs sollte man verbannen! . . . . .	5
<b>2</b>	<b>Wireless Time Travel</b>	<b>5</b>
2.1	Vier zukunftssichere Handschläge . . . . .	5
2.2	Code der Zukunft . . . . .	5
2.3	Ungewöhnlich verschlüsselte Botschaft . . . . .	5
2.4	Geheimnisvoller Zugang: superboss . . . . .	6
2.5	Unbrauchbarer Schlüssel . . . . .	6
2.6	Einen Schlüssel für einen Schlüssel! Echt jetzt?! . . . . .	6
2.7	Verborgenes Protokoll . . . . .	6
<b>3</b>	<b>Bot Bot Bot Bot</b>	<b>6</b>
3.1	I keep you my little secret ... . . . .	6
<b>4</b>	<b>Cäsars Schlüsselbund</b>	<b>6</b>
4.1	Schlüssel. Knacken. . . . .	6
4.2	Passwörter Retten. . . . .	6
<b>5</b>	<b>Paranoider Mozart</b>	<b>7</b>
5.1	MozART. . . . .	7
<b>6</b>	<b>Zertifiziertes Durcheinander</b>	<b>7</b>
6.1	Zertifizieren ist schwer . . . . .	7
<b>7</b>	<b>Zeitreise durch das World Wide Web</b>	<b>8</b>
7.1	Wieder Elvis . . . . .	8
7.2	CäsarMussWeg! MussCäsarWeg? . . . . .	8
7.3	dackboor. . . . .	8
7.4	Schlechtes Timing (Time Travel Edition) . . . . .	8
7.5	Sorcerer ... ? . . . . .	8
<b>8</b>	<b>Seitlich fließend</b>	<b>8</b>
8.1	Newton und Co KG. . . . .	8
<b>9</b>	<b>Antike Mobile Security</b>	<b>8</b>
9.1	iTimeTravel . . . . .	8
9.2	AND(roid)ERS . . . . .	9
<b>10</b>	<b>Babycam Espionage</b>	<b>9</b>
10.1	The Rise of the HuManoiD5 . . . . .	9
10.2	ETA . . . . .	9

<b>11 Das Social Media der Zukunft</b>	<b>9</b>
11.1 Der vergiftete Passwort Reset . . . . .	9
11.2 Accountübernahme . . . . .	9
<b>12 Hidden Timelines</b>	<b>9</b>
12.1 Phantom Domain . . . . .	9
<b>13 Vault Voyage</b>	<b>10</b>
13.1 That's all your vault! . . . . .	10
<b>14 Wikinger Overflow</b>	<b>10</b>
14.1 Überlauf. Hand drauf. . . . .	10
14.2 Typisch Typing ... Stufe 1 . . . . .	10
14.3 Typisch Typing ... Stufe 2 . . . . .	10
14.4 Typisch Typing ... Stufe 3 . . . . .	10
<b>15 Tap to the Future</b>	<b>10</b>
15.1 Tick Tock Tap . . . . .	10
<b>16 So viele</b>	<b>10</b>
16.1 Das Device ist heiß . . . . .	10
16.2 Persona non grata . . . . .	11
16.3 Eine Frage der Kommunikation . . . . .	11
16.4 Treffpunkt . . . . .	11
16.5 Alles dokumentiert! . . . . .	11
16.6 Es geht immer um Inhalte . . . . .	11
<b>17 Web of Treats</b>	<b>11</b>
17.1 Mitgliedschaftsnr. . . . .	11
17.2 Geheimer Artikel . . . . .	11
17.3 Überfüllt . . . . .	11
17.4 A shell in the forest? . . . . .	12
17.5 Elvis . . . . .	12
<b>18 Das. Beste. Text. Adventure. Aller. Zeiten.</b>	<b>12</b>
18.1 Time to travel! . . . . .	12
18.2 Mein Name? . . . . .	12
18.3 Ein PIN! . . . . .	12
18.4 Ach ... ein Schlüssel . . . . .	12
18.5 Flag! . . . . .	12
<b>19 Passwörter werden wir auch nie los, oder?!</b>	<b>12</b>
19.1 Gute Idee, um ein Passwort zu verstecken?! . . . . .	12
19.2 Call Julius ... äh. John. . . . .	13
19.3 Nicht nur Ziffern, sonder auch ...? . . . . .	13
19.4 /etc/ANTIK? . . . . .	13
19.5 Sicher sicher? . . . . .	13
19.6 Zeitlose Liste . . . . .	13

19.7 (Image)magic(k)	13
19.8 Auch in Zukunft ein schweres Passwort?	13
<b>20 Franz Joseph und die Kommandozeile</b>	<b>13</b>
20.1 Stage	13
20.2 Stagee	15
20.3 Stageee	16
20.4 Stageeee	17
20.5 Stageeeee	18
20.6 Stageeeeee	19
20.7 Stageeeeeee	20
20.8 Stageeeeeeee	20
20.9 Stageeeeeeeee	20
20.10Stageeeeeeeeeee	20
20.11Stageeeeeeeeeeee	20
20.12Stageeeeeeeeeeeee	20
<b>21 Ueberschrift 1</b>	<b>20</b>
21.1 Hinweise	20
<b>22 Beispiele</b>	<b>21</b>
22.1 Source Code formatieren	21
22.2 Bilder	22

# **1 Der Service war auch schon besser ...**

## **1.1 Achtung! Streng geheim!**

Um diese Aufgabe zu lösen, hat es genügt, das besagte PDF im Browser zu öffnen. Der "streng geheime" String befand sich im Titel des Tabs.

## **1.2 Eine schräge Nummer**

Die Rechnungsnummer wurde zwar von einem schwarzen Rechteck verdeckt, ließ sich jedoch ganz einfach kopieren, indem man die betroffene Stelle markiert -> Strg + C

## **1.3 Was letzte Preis?**

Selbes Spiel, auch der Preis ließ sich ganz simpel herauskopieren.

## **1.4 IBANs sollte man verbannen!**

Um den IBAN aufzudecken, habe ich PDF-XChange verwendet, um das schwarze Rechteck mit dem Objektbearbeitungswerkzeug zu entfernen.

# **2 Wireless Time Travel**

## **2.1 Vier zukunftssichere Handschläge**

Nicht gelöst.

## **2.2 Code der Zukunft**

Nicht gelöst.

## **2.3 Ungewöhnlich verschlüsselte Botschaft**

Nicht gelöst.

## **2.4 Geheimnisvoller Zugang: superboss**

Nicht gelöst.

## **2.5 Unbrauchbarer Schlüssel**

Nicht gelöst.

## **2.6 Einen Schlüssel für einen Schlüssel! Echt jetzt?!**

Nicht gelöst.

## **2.7 Verborgenes Protokoll**

Nicht gelöst.

## **3 Bot Bot Bot Bot**

### **3.1 I keep you my little secret ...**

Nicht gelöst.

## **4 Cäsars Schlüsselbund**

### **4.1 Schlüssel. Knacken.**

Nicht gelöst.

### **4.2 Passwörter Retten.**

Nicht gelöst.

## 5 Paranoid Mozart

### 5.1 MozART.

Nicht gelöst.

## 6 Zertifiziertes Durcheinander

### 6.1 Zertifizieren ist schwer

Um den Certificate Signing Request zu erstellen habe ich den folgenden Befehl verwendet: `openssl req -newkey rsa:4096 -sha512 -config openssl.cnf -out csr.csr`

`-subj "/CN=12123657-Intermediate-CA-WS2024/OU=ESSE-Lab1-Exercise"`

`req` ist der Befehl um einen CSR zu erstellen;

`-newkey rsa:4096` spezifiziert, dass ein neuer Key (4096-Bit RSA) erstellt werden soll;

`-sha512` gibt an, dass `sha512WithRSAEncryption` als Signaturalgorithmus verwendet werden soll;

mit `-config` wird angegeben, welches config file zu verwenden ist;

`-out` bestimmt das output-file und

`-subj "/CN=12123657-Intermediate-CA-WS2024/OU=ESSE-Lab1-Exercise"` definiert die gewünschten Namens-Parameter im CSR.

Das config file dient dazu, die nötigen X509v3 Parameter zu setzen und sieht aus wie folgt:

```
[ req ]
2  default_bits           = 4096
   default_md             = sha512
4  default_keyfile        = privkey.pem
   distinguished_name     = req_distinguished_name
6  req_extensions         = v3_req

8  [ req_distinguished_name ]

10 [ v3_req ]
   subjectKeyIdentifier = hash
12 basicConstraints = critical, CA:true, pathlen:0
   keyUsage = critical, Certificate Sign, CRL Sign
```

Listing 1: openssl.cnf

Nach Ausführung des oben genannten Befehls, wird der CSR in der Datei `csr.csr` gespeichert, diese wurde im Abgabetool eingereicht.

## **7 Zeitreise durch das World Wide Web**

### **7.1 Wieder Elvis**

Nicht gelöst.

### **7.2 CäsarMussWeg! MussCäsarWeg?**

Nicht gelöst.

### **7.3 dackboor.**

Nicht gelöst.

### **7.4 Schlechtes Timing (Time Travel Edition)**

Nicht gelöst.

### **7.5 Sorcerer ... ?**

Nicht gelöst.

## **8 Seitlich fließend**

### **8.1 Newton und Co KG.**

Nicht gelöst.

## **9 Antike Mobile Security**

### **9.1 iTimeTravel**

Nicht gelöst.



## **9.2 AND(roid)ERS**

Nicht gelöst.

# **10 Babycam Espionage**

## **10.1 The Rise of the HuManoiD5**

Nicht gelöst.

## **10.2 ETA**

Nicht gelöst.

# **11 Das Social Media der Zukunft**

## **11.1 Der vergiftete Passwort Reset**

Nicht gelöst.

## **11.2 Accountübernahme**

Nicht gelöst.

# **12 Hidden Timelines**

## **12.1 Phantom Domain**

Nicht gelöst.

## **13 Vault Voyage**

### **13.1 That's all your vault!**

Nicht gelöst.

## **14 Wikinger Overflow**

### **14.1 Überlauf. Hand drauf.**

Nicht gelöst.

### **14.2 Typisch Typing ... Stufe 1**

Nicht gelöst.

### **14.3 Typisch Typing ... Stufe 2**

Nicht gelöst.

### **14.4 Typisch Typing ... Stufe 3**

Nicht gelöst.

## **15 Tap to the Future**

### **15.1 Tick Tock Tap**

Nicht gelöst.

## **16 So viele**

### **16.1 Das Device ist heiß**

Nicht gelöst.

## **16.2 Persona non grata**

Nicht gelöst.

## **16.3 Eine Frage der Kommunikation**

Nicht gelöst.

## **16.4 Treffpunkt**

Nicht gelöst.

## **16.5 Alles dokumentiert!**

Nicht gelöst.

## **16.6 Es geht immer um Inhalte**

Nicht gelöst.

# **17 Web of Treats**

Nicht gelöst.

## **17.1 Mitgliedschaftsnr.**

Nicht gelöst.

## **17.2 Geheimer Artikel**

Nicht gelöst.

## **17.3 Überfüllt**

Nicht gelöst.

## **17.4 A shell in the forest?**

Nicht gelöst.

## **17.5 Elvis**

Nicht gelöst.

# **18 Das. Beste. Text. Adventure. Aller. Zeiten.**

## **18.1 Time to travel!**

Nicht gelöst.

## **18.2 Mein Name?**

Nicht gelöst.

## **18.3 Ein PIN!**

Nicht gelöst.

## **18.4 Ach ... ein Schlüssel**

Nicht gelöst.

## **18.5 Flag!**

Nicht gelöst.

# **19 Passwörter werden wir auch nie los, oder?!**

## **19.1 Gute Idee, um ein Passwort zu verstecken?!**

Nicht gelöst.

## **19.2 Call Julius ... äh. John.**

Nicht gelöst.

## **19.3 Nicht nur Ziffern, sonder auch ...?**

Nicht gelöst.

## **19.4 /etc/ANTIK?**

Nicht gelöst.

## **19.5 Sicher sicher?**

Nicht gelöst.

## **19.6 Zeitlose Liste**

Nicht gelöst.

## **19.7 (Image)magic(k)**

Nicht gelöst.

## **19.8 Auch in Zukunft ein schweres Passwort?**

Nicht gelöst.

# **20 Franz Joseph und die Kommandozeile**

## **20.1 Stage**

Bei diesem Beispiel musste man sich mit dem Befehl `ssh e12122544@tese.esse-teaching.at -p 12345` in tese einloggen und von dort mit dem Befehl `ssh e1sec_team44@10.10.201 -p 22044` zum vorgebenen Host verbinden. Hier gab es eine "welcome.txt"

Datei welche Beschrieb dass ich mich in den user stage00 einloggen soll und dort die Aufgabe machen soll. Die Aufgabe war es einen username mit verstecktem Passwort zu finden. Für diese Stage haben mich die folgenden Schritte zum Ziel geführt.

Nach dem verbinden zur vorgegebenen Maschine:

- Ausführen von `ls -la`
- Interessanten versteckten Ordner gefunden
- In den Ordner gewechselt mit `cd`
- Erneut `ls -la` ausgeführt
- Interessante versteckte Datei gefunden
- Inhalt der Datei ausgegeben
- Fertig

Lösung:

- Username: stage01
- Passwort: bi0owaiK6ieK

Das folgende Bild zeigt die ausgeführten Befehle in der Kommandozeile.



```
key — stage00@cmdbox44: ~/.what_is_this — ssh e12122544@tese.esse-teaching.at -p 12345 — 110x25
[stage00@cmdbox44:~]$ ls -la
total 24
drwxr-x--- 3 root stage00 4096 Dec 19 03:07 .
drwxr-xr-x 1 root root    4096 Dec 19 03:07 ..
-rw-r----- 1 root stage00 220 Mar 29 2024 .bash_logout
-rw-r----- 1 root stage00 3526 Mar 29 2024 .bashrc
-rw-r----- 1 root stage00 807 Mar 29 2024 .profile
drwxr-xr-x 2 root root    4096 Dec 19 03:07 .what_is_this
[stage00@cmdbox44:~]$ cd .what_is_this/
[stage00@cmdbox44:~/.what_is_this]$ ls -la
total 12
drwxr-xr-x 2 root root    4096 Dec 19 03:07 .
drwxr-x--- 3 root stage00 4096 Dec 19 03:07 ..
-rw-r--r-- 1 root root    132 Dec 19 03:07 .hidden
[stage00@cmdbox44:~/.what_is_this]$ cat .hidden
Ich sagte doch, es gibt auch einfache Aufgaben.

Deine nächste Aufgabe findest du hier:

Username: stage01
Passwort: bi0owaiK6ieK
[stage00@cmdbox44:~/.what_is_this]$
```

Abbildung 1: Lösungsweg "Stage"

## 20.2 Stagee

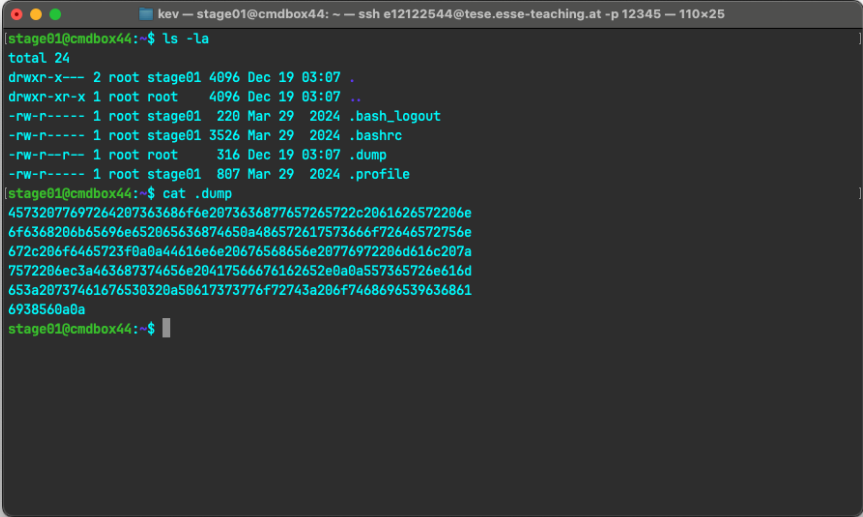
Dieses Beispiel hatte dieselbe Aufgabe wie die vorige, und zwar ein verstecktes Passwort finden. Hier war ich schon auf der richtigen Maschine eingeloggt, ich musste nurmehr user wechseln welchen ich aus der vorigen Ausgabe erhalten habe. Für diese Stagee haben mich die folgenden Schritte zum Ziel geführt:

- Einloggen mit dem gegebenen Benutzer: `su -l stage01`
- Ausführen von `ls -la`
- Interessante Datei `.dump` gefunden, die in "Stage" nicht vorhanden war
- Dateinhalt mit `cat .dump` ausgegeben
- Die Hexadezimaldaten mit einem Hex-Decoder decodiert
- Fertig

Lösung:

- Username: stage02
- Passwort: othie9chai8V

Das folgende Bild zeigt die ausgeführten Befehle in der Kommandozeile.



```
kevin@stage01@cmdbox44: ~ -- ssh e12122544@tese.esse-teaching.at -p 12345 -- 110x25
[stage01@cmdbox44:~]$ ls -la
total 24
drwxr-x--- 2 root stage01 4096 Dec 19 03:07 .
drwxr-xr-x 1 root root    4096 Dec 19 03:07 ..
-rw-r----- 1 root stage01 220 Mar 29 2024 .bash_logout
-rw-r----- 1 root stage01 3526 Mar 29 2024 .bashrc
-rw-r----- 1 root root    316 Dec 19 03:07 .dump
-rw-r----- 1 root stage01 887 Mar 29 2024 .profile
[stage01@cmdbox44:~]$ cat .dump
45732077697264207363686f6e2073636877657265722c2061626572206e
6f6368206b65696e652065636874650a486572617573666f72646572756e
672c206f6465723f0a0a44616e6e20676568656e20776972206d616c207a
7572206e63a463687374656e2041756667616265206a0a557365726e616d
653a20737461676530320a50617373776f72743a206f7468696539636861
6938560a0a
[stage01@cmdbox44:~]$
```

Abbildung 2: Lösungsweg "Stagee"

## 20.3 Stageee

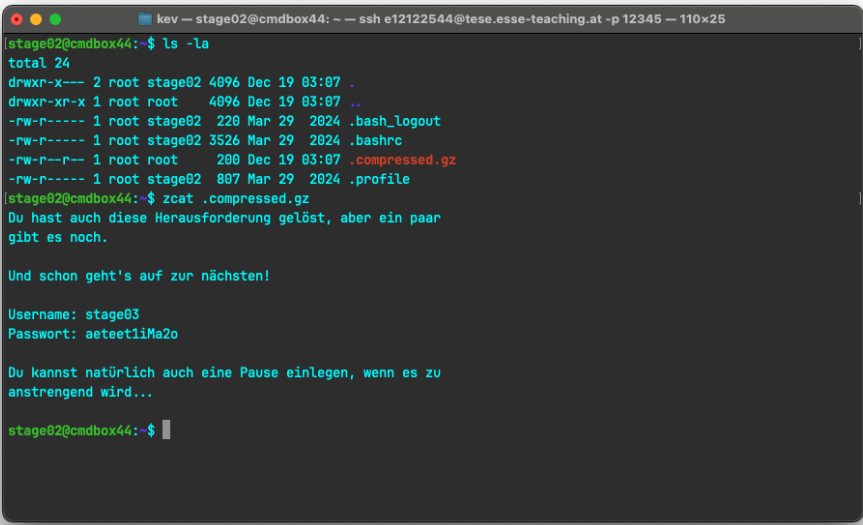
Bei diesem Beispiel war es wieder dasselbe. Für diese Stageee haben mich die folgenden Schritte zum Ziel geführt:

- Einloggen mit dem gegebenen Benutzer: `su -l stage02`
- Ausführen von `ls -la`
- Interessante `.compressed.gz` Datei gefunden
- Konnte sie nicht mit `gunzip` entpacken, daher Inhalt mit `zcat` ausgelesen
- Inhalt wird ausgegeben
- Fertig

Lösung:

- Username: stage03
- Passwort: aeteet1iMa2o

Das folgende Bild zeigt die ausgeführten Befehle in der Kommandozeile.



```
kevin@stage02@cmdbox44: ~ -- ssh e12122544@tese.esse-teaching.at -p 12345 -- 110x25
[stage02@cmdbox44:~]$ ls -la
total 24
drwxr-x--- 2 root stage02 4096 Dec 19 03:07 .
drwxr-xr-x 1 root root    4096 Dec 19 03:07 ..
-rw-r----- 1 root stage02 220 Mar 29 2024 .bash_logout
-rw-r----- 1 root stage02 3526 Mar 29 2024 .bashrc
-rw-r--r-- 1 root root    200 Dec 19 03:07 .compressed.gz
-rw-r----- 1 root stage02 807 Mar 29 2024 .profile
[stage02@cmdbox44:~]$ zcat .compressed.gz
Du hast auch diese Herausforderung gelöst, aber ein paar
gibt es noch.

Und schon geht's auf zur nächsten!

Username: stage03
Passwort: aeteet1iMa2o

Du kannst natürlich auch eine Pause einlegen, wenn es zu
anstrengend wird...

stage02@cmdbox44:~$
```

Abbildung 3: Lösungsweg "Stageee"



## 20.4 Stageeee

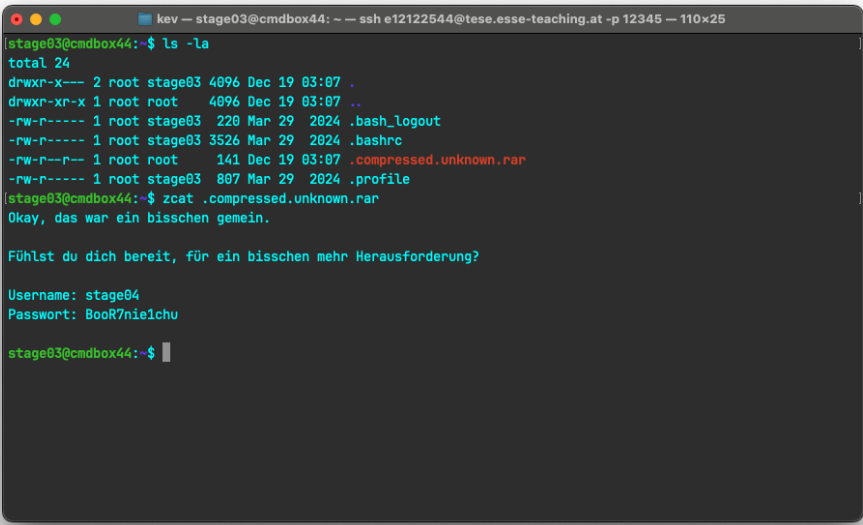
Bei diesem Beispiel war es wieder dasselbe. Für diese Stageee haben mich die folgenden Schritte zum Ziel geführt:

- Einloggen mit dem gegebenen Benutzer: `su -l stage03`
- Ausführen von `ls -la`
- Interessante `.compressed.unknown.rar` Datei gefunden
- `zcat` auf die Datei ausgeführt
- Inhalt wird ausgegeben
- Fertig

Lösung:

- Username: stage04
- Passwort: BooR7nie1chu

Das folgende Bild zeigt die ausgeführten Befehle in der Kommandozeile.



```
kev — stage03@cmdbox44: ~ — ssh e12122544@tese.esse-teaching.at -p 12345 — 110x25
[stage03@cmdbox44:~]$ ls -la
total 24
drwxr-x--- 2 root stage03 4096 Dec 19 03:07 .
drwxr-xr-x 1 root root    4096 Dec 19 03:07 ..
-rw-r----- 1 root stage03 220 Mar 29 2024 .bash_logout
-rw-r----- 1 root stage03 3526 Mar 29 2024 .bashrc
-rw-r----- 1 root root    141 Dec 19 03:07 .compressed.unknown.rar
-rw-r----- 1 root stage03 807 Mar 29 2024 .profile
[stage03@cmdbox44:~]$ zcat .compressed.unknown.rar
Okay, das war ein bisschen gemein.

Fühlst du dich bereit, für ein bisschen mehr Herausforderung?

Username: stage04
Passwort: BooR7nie1chu

stage03@cmdbox44:~$
```

Abbildung 4: Lösungsweg "Stageeee"

## 20.5 Stageeeee

Bei diesem Beispiel war es wieder dasselbe. Für diese Stageee haben mich die folgenden Schritte zum Ziel geführt:

- Einloggen mit dem gegebenen Benutzer: `su -l stage04`
- Ausführen von `ls -la`
- Interessante `.encrypted` Datei gefunden
- `cat` auf die Datei ausgeführt, um den Inhalt auszugeben
- Inhalt scheint verschlüsselt zu sein
- Sieht nach Base64 aus
- In Base64-Decoder eingegeben (Ausgabe siehe 5)
- Zufällige Zeichen deuten darauf hin, dass es komprimiert sein könnte
- Mit Base64-Befehl entschlüsselt, entpackt und direkt auf die Konsolenausgabe ausgegeben, da das Schreiben in Dateien in diesem Verzeichnis nicht erlaubt ist. Folgender Befehl wurde verwendet: `base64 -d .encrypted | gunzip`
- Fertig

Lösung:

- Username: stage05
- Passwort: eifietiey2Go

**Decode from Base64 format**  
Simply enter your data then push the decode button.

---

H4sIAAAAAAAAAXMsQrCMBCH8T1PcZtLCSK4dBZcXXRP7d/rQXqB3JWgz+bmi5mu3we/SzJaxWlG  
pQeqPZf8+5ohb8rUUiWI0iTWB5QYuTB0oM/GYEzQGMJdZ2oQ7wJj8YPFuFdD1bRiJPPEOJ7DLZm1  
Un3s5kvggvfpWkL4A+CemlyDAAAA

☐ For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8  Source character set.

☐ Decode each line separately (useful for when you have multiple entries).

☒ Live mode OFF Decodes in real-time as you type or paste (supports only the UTF-8 character set).

**< DECODE >** Decodes your data into the area below.

0=OqK "t \tOAz7拙 K2ZIF =loR%\$ 0tL jgi cYWCJb\$8-RjKZB \0000

Abbildung 5: Ergebnis der base64 Dekodierung von dem Inhalt der Datei .encrypted

```

kev — stage04@cmdbox44: ~ — ssh e12122544@tese.esse-teaching.at -p 12345 — 110x25
[stage04@cmdbox44:~]$ ls -la
total 24
drwxr-x--- 2 root stage04 4096 Dec 19 03:07 .
drwxr-xr-x 1 root root    4096 Dec 19 03:07 ..
-rw-r----- 1 root stage04 220 Mar 29 2024 .bash_logout
-rw-r----- 1 root stage04 3526 Mar 29 2024 .bashrc
-rw-r----- 1 root root    183 Dec 19 03:07 .encrypted
-rw-r----- 1 root stage04 807 Mar 29 2024 .profile
[stage04@cmdbox44:~]$ cat .encrypted
H4sIAAAAAAAAAXMsQrCMBCH8T1PcZtLCSK4dBZcXXRP7d/rQXqB3JWgz+bmi5mu3we/SzJaxWlG
pQeqPZf8+5ohb8rUUiWI0iTWB5QYuTB0oM/GYEzQGMJdZ2oQ7wJj8YPFuFdD1bRiJPPEOJ7DLZm1
Un3s5kvggvfpWkL4A+CemlyDAAAA
[stage04@cmdbox44:~]$ base64 -d .encrypted | gunzip
Das mit der Verschlüsselung war ein bisschen gelogen, zugegeben.

Und weiter geht's...

Username: stage05
Passwort: eifietiey26o

stage04@cmdbox44:~$

```

Abbildung 6: Lösungsweg "Stageeeee"

## 20.6 Stageeeeeee

Nicht gelöst.

## 20.7 Stageeeeeeee

Nicht gelöst.

## 20.8 Stageeeeeeeee

Nicht gelöst.

## 20.9 Stageeeeeeeeeee

Nicht gelöst.

## 20.10 Stageeeeeeeeeeee

Nicht gelöst.

## 20.11 Stageeeeeeeeeeeee

Nicht gelöst.

## 20.12 Stageeeeeeeeeeeeeee

Nicht gelöst.

# 21 Ueberschrift 1

## 21.1 Hinweise

*Hinweise:*

- Verwenden Sie entweder diese deutsche Version oder die englische Version in `protocol.tex`.
- Setzen Sie alle Variablen nach *FOR STUDENTS* in der `.tex` Datei.
- Ersetzen Sie die Platzhalter für Ihre Namen und MatNr.

- Löschen Sie diese Sektion über Hinweise und die folgenden Beispiel-Kapitel.
- Achten Sie auf geforderte Formate und Anforderungen an die Dateinamen.
- Führen Sie `pdflatex` mindestens zweimal aus, damit die Referenzen und Seitenzahlen richtig im PDF dargestellt werden.
- Sie können dazu auch das Makefile verwenden: `make de`.

## 22 Beispiele

### 22.1 Source Code formatieren

Es folgen einige Beispiele wie Sourcecode in diesem Dokument formatiert und referenziert werden kann (siehe Listing 2 auf Seite 21 und siehe Listing 3 auf Seite 21).

Ebenso können kurzer Code oder kurze Befehle direkt in der Zeile in einem `lstinline` Block mit typengleicher Schrift formatiert werden.

```

/*
2  * Just an example C-file.
  */
4
#include <stdio.h>
6
int global_variable = 1;
8 #ifdef DEBUG
int another_global_variable = 1;
10 #endif
12
/*
  * Some comment
14 */
int main(void)
16 {
    temp_variable = 4711;
18    another_variable = 0815;
20
    printf("foo bar baz %02d", temp_variable);
22
    return 1;
}

```

Listing 2: Example C/C++ file

```

#!/bin/bash
2 echo "Bash version ${BASH_VERSION}..."
for i in {0..10..2}
4 do
    echo "Welcome $i times"
6 done

```

```
8 echo "some very very very very very very very very very very ↵  
    very very very very very very very very very very very ↵  
    long string"  
10 exit 0;
```

Listing 3: Example bash script

## 22.2 Bilder

Es folgen einige Beispiele wie Bilder in diesem Dokument eingefuegt werden koennen (siehe [Abbildung 7 auf Seite 22](#)).

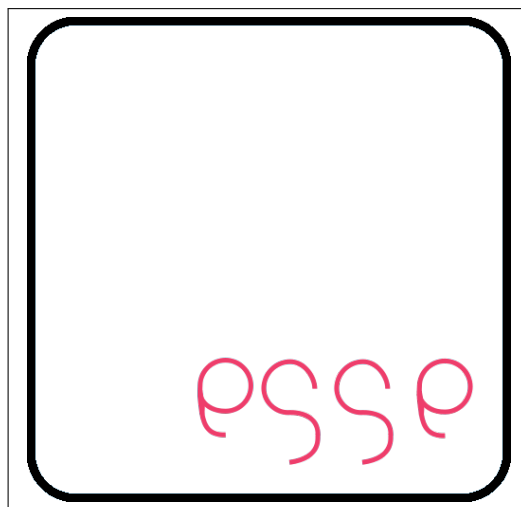


Abbildung 7: ESSE Logo