

Generating Fingerprints of Network Servers and their Use in Honeypots

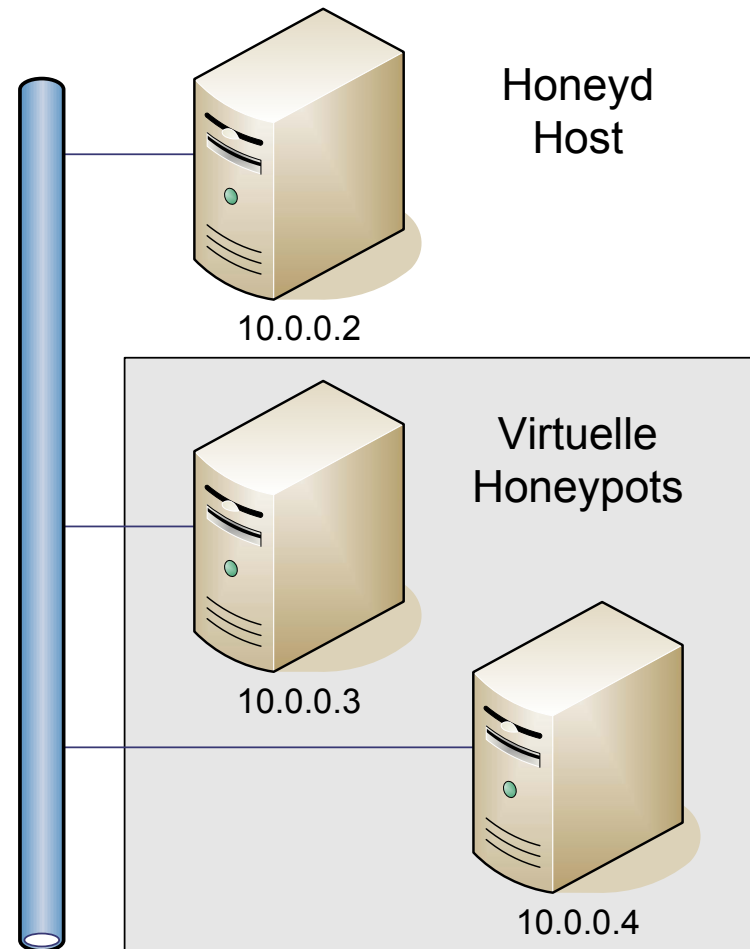
Thomas Apel

Der Überblick

- Fingerprinting von Netzwerkdiensten
 - Banner
 - Verfügbare Optionen
 - Reaktionen auf falsche Syntax
- Verwendung für Honeypots
 - Emulator für Netzwerkdienste
 - Soll automatisch erzeugt werden
 - Plug-In für Honeyd

Honeyd

- Emuliert Rechner und Infrastruktur
 - Emuliert Eigenheiten der IP-Stacks
 - Täuscht damit Netzwerkscanner
 - Benutzt Datenbanken von Nmap, p0f, Xprobe2
- Netzwerkdienste
 - Shellskripte
 - Pythonmodule
 - Weiterleitung an echte Server



Fingerprinting-Tools

- Nmap
 - Viele Protokolle
 - Prüft Banner
- Vmap
 - Viele Protokolle
 - Datenbank enthält unveränderte Antworten
- Amap
 - Viele Protokolle
 - Trigger für z.B. DNS
- Smtpscan
 - Nur 15 Tests reichen
 - Nur Statuscodes werden ausgewertet
 - Über 3000 Fingerprints
- Ftpmap
 - Ausführliche Tests
 - Testet Zufälligkeit der Portauswahl
 - Knapp 70 Fingerprints
- Und noch ein paar andere...

Beispiele: FTP

- TYPE Kommando ohne Login
 - FreeBSD FTPd
 - Linux Netkit FTPd
- CWD ohne Parameter
 - In RFC nicht definiert
 - Wechsel zu Root-Directory
 - Bleibe im aktuellen Verzeichnis
- Werden EPRT und EPSV unterstützt?
- Negative Integer als Parameter für PORT, ALLO, etc.

```
220 freebsd.domain FTP server (Version
6.00LS) ready.

TYPE A
530 Please login with USER and PASS.

TYPE X
530 Please login with USER and PASS.
500 'TYPE X': command not understood.

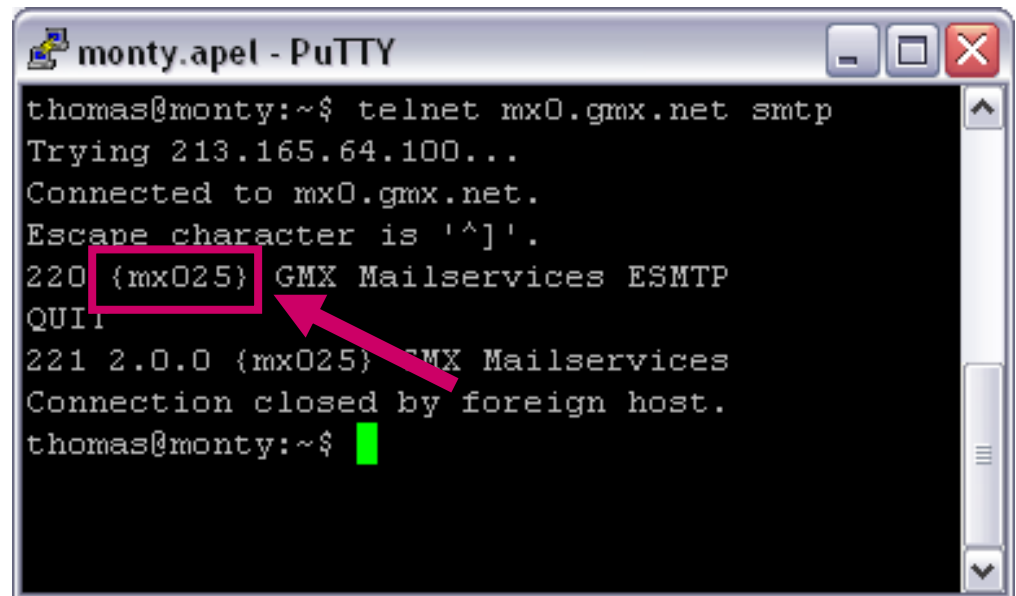
220 linux-netkit.domain FTP server
(Version 6.4/OpenBSD/Linux-ftpd-0.17)
ready.

TYPE X
530 Please login with USER and PASS.
500 'TYPE X': command not understood.

TYPE C
530 Please login with USER and PASS.
500 'TYPE C': command not understood.
500 'TYPE C': command not understood.
```

Beispiele: SMTP

- Tests aus Smtplib
 - Sind Exoten wie SOML, SAML oder TURN implementiert?
 - Mail-Adresse mit nur einer spitzen Klammer
- Server von GMX
 - Geschweifte Klammern
 - Kein FQDN
 - Damit überhaupt RFC-konform?




```
monty.apel - PuTTY
thomas@monty:~$ telnet mx0.gmx.net smtp
Trying 213.165.64.100...
Connected to mx0.gmx.net.
Escape character is '^]'.
220 {mx025} GMX Mailservices ESMTQ
QUIT
221 2.0.0 {mx025} GMX Mailservices
Connection closed by foreign host.
thomas@monty:~$
```

Existierende Emulatoren

- IIS Emulator
 - Emuliert Microsofts Internet Information Server
- smtpot.py
 - Emuliert Sendmail
- smtp.pl
 - Liegt Honeyd bei
 - Emuliert Sendmail und Postfix
 - Kann begrenzt um andere Server erweitert werden
- Probleme
 - Emulieren nur bestimmte Serverinkarnationen
 - In Handarbeit entstanden

Der Wunsch

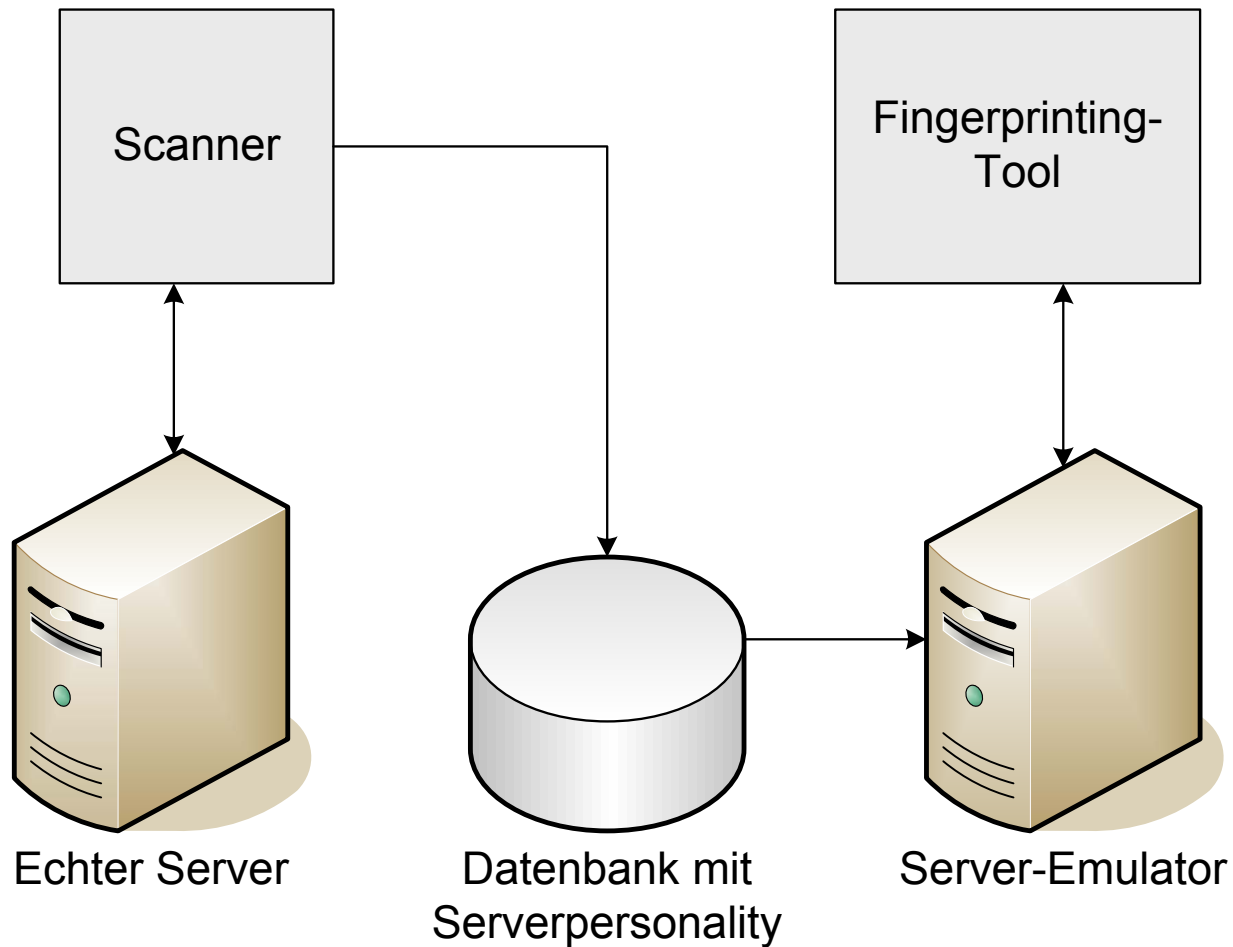


Die große,
universelle,
vollautomatische
Lösung

Der Plan

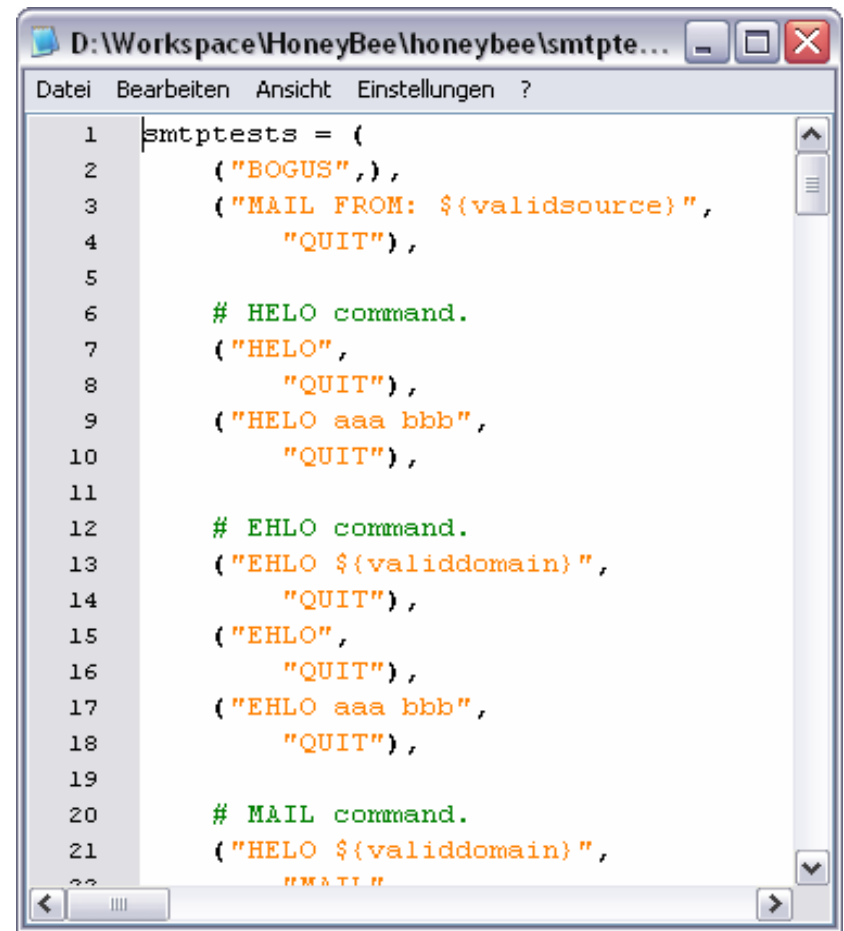
- Scanner
 - Unterhält sich mit Server
 - Testet Reaktion in verschiedenen Situationen
 - Generiert Datenbank
- Datenbank
 - Enthält Antworten für verschiedene Situationen
- Emulator
 - Findet Antworten in Datenbank
 - Geschrieben in Python
 - Nutzt Plugin-Schnittstelle von Honeyd

Das große Ganze



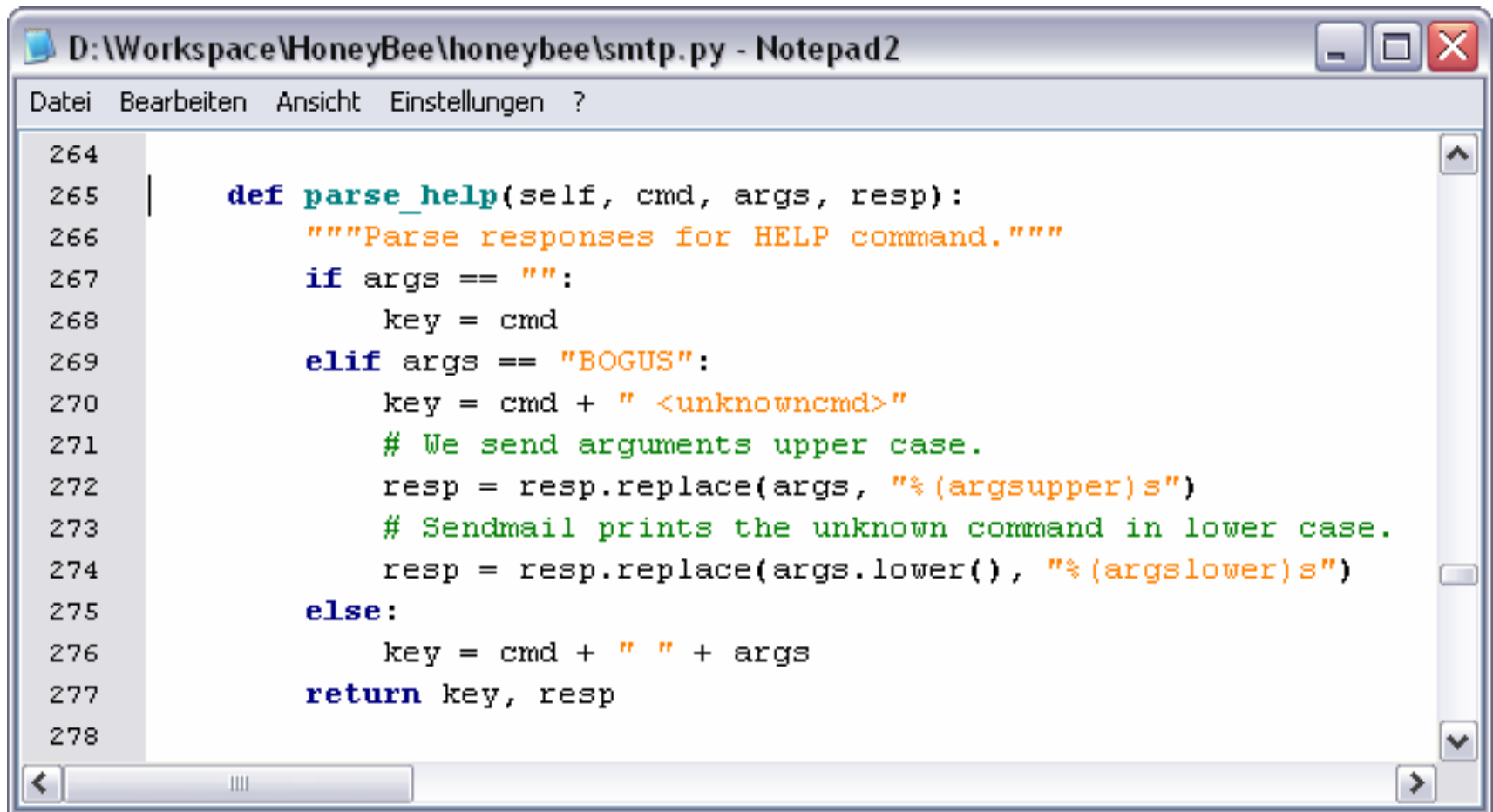
Der Scanner

- Provoziert verschiedene Situationen und speichert Antworten
- Parser pro Kommando
 - Dynamische Teile werden durch Platzhalter ersetzt
 - Falsche Parameter werden zu Klassen zusammengefasst
- Scanner muss sich seinen Zustand merken
 - Login
 - Wurde Absender gesetzt?
 - Wurde Empfänger gesetzt?



```
D:\Workspace\HoneyBee\honeybee\smtpte...
Datei Bearbeiten Ansicht Einstellungen ?
1 smtpstests = (
2     ("BOGUS",),
3     ("MAIL FROM: ${validsource}",
4         "QUIT"),
5
6     # HELO command.
7     ("HELO",
8         "QUIT"),
9     ("HELO aaa bbb",
10        "QUIT"),
11
12    # EHLO command.
13    ("EHLO ${validdomain}",
14        "QUIT"),
15    ("EHLO",
16        "QUIT"),
17    ("EHLO aaa bbb",
18        "QUIT"),
19
20    # MAIL command.
21    ("HELO ${validdomain}",
22        "MAIL")
```

Der Scanner



The image shows a Notepad2 window titled "D:\Workspace\HoneyBee\honeybee\smtp.py - Notepad2". The window contains Python code for a function named `parse_help`. The code is as follows:

```
264
265 | def parse_help(self, cmd, args, resp):
266 |     """Parse responses for HELP command."""
267 |     if args == "":
268 |         key = cmd
269 |     elif args == "BOGUS":
270 |         key = cmd + " <unknowncmd>"
271 |         # We send arguments upper case.
272 |         resp = resp.replace(args, "%(argsupper)s")
273 |         # Sendmail prints the unknown command in lower case.
274 |         resp = resp.replace(args.lower(), "%(argslower)s")
275 |     else:
276 |         key = cmd + " " + args
277 |     return key, resp
278
```

Die Datenbank

- Hashtabelle/Dictionary
 - Gespeichert in Textfile
 - Zeilenweise Tupel aus Schlüssel und Wert
 - Einlesen und speichern per repr() und eval()
- Schlüssel
 - Kommandos
 - Parameter
 - Situation bzw. Zustand
- Wert
 - Serverantwort
 - Mit Platzhaltern für dynamische Antwortteile

Die Datenbank



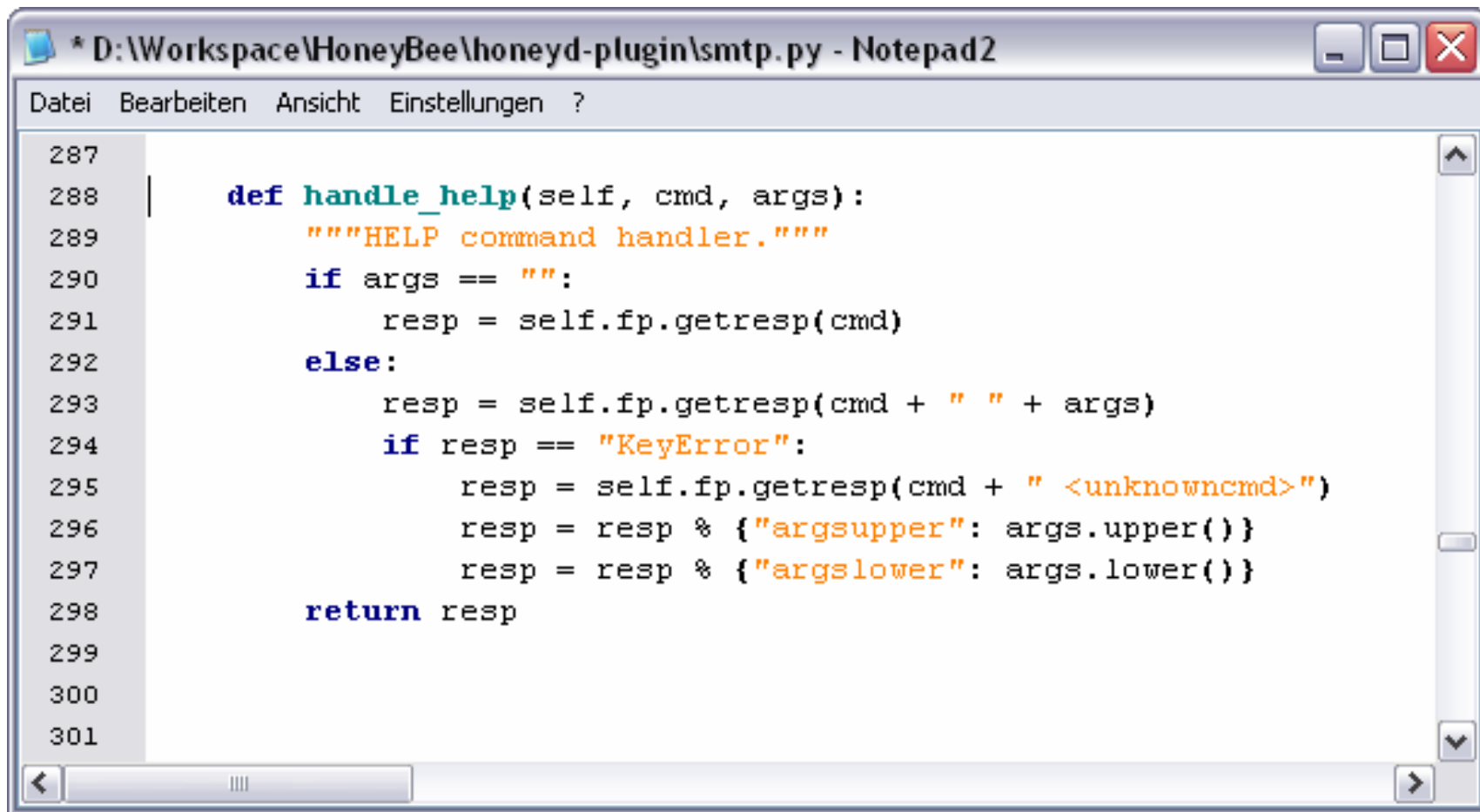
The screenshot shows a Notepad2 window with the title bar: * D:\Workspace\HoneyBee\honeyd-plugin\def\smtp\sendmail-8.13.1-freebsd - Notepad2. The window contains a list of SMTP command and response pairs, numbered 1 through 16. The commands are in single quotes, and the responses are in double quotes. The responses include status codes, error messages, and success messages.

```
1 ('HELP', '214-2.0.0 This is sendmail version 8.13.1\r\n214-2.0.0 Topics:\r\n214-2.0.0 \tHE:
2 ('HELP <unknowncmd>', '504 5.3.0 HELP topic "%(argslower)s" unknown\r\n')
3 ('HELP QUIT', '214-2.0.0 QUIT\r\n214-2.0.0 \tExit sendmail (SMTP).\r\n214 2.0.0 End of HEL:
4
5 ('RCPT nosender', '503 5.0.0 Need MAIL before RCPT\r\n')
6
7 ('RCPT', '501 5.5.2 Syntax error in parameters scanning ""\r\n')
8 ('RCPT TO:', '501 5.5.2 Syntax error in parameters scanning "TO"\r\n')
9
10 ('RCPT TO: <knownuser> withoutbrackets', '250 2.1.5 %(rcpt)s... Recipient ok\r\n')
11 ('RCPT TO: <knownuser> startbracketmissing', "553 5.0.0 %(rcpt)s>... Unbalanced '>'\r\n")
12 ('RCPT TO: <knownuser> endbracketmissing', "553 5.0.0 <%(rcpt)s... Unbalanced '<'\r\n")
13 ('RCPT TO: <knownuser> withbrackets', '250 2.1.5 <%(rcpt)s>... Recipient ok\r\n')
14
15 ('RCPT TO: <relayaddress> withbrackets', '550 5.7.1 <%(rcpt)s>... Relaying denied\r\n')
16
```

Der Emulator

- Anfragen werden in Kommando und Parameter zerlegt
- Command-Handler
 - Analysiert Anfrage
 - Berücksichtigt Zustand
 - Holt Antwort aus Datenbank
 - Ersetzt Platzhalter mit dynamischen Daten
- Steuerung
 - Übergang von einem Zustand zum nächsten
 - Zum Beispiel: Wurde Absender mit fehlender spitzer Klammer akzeptiert?
 - Fehlercodes der Antwort reichen

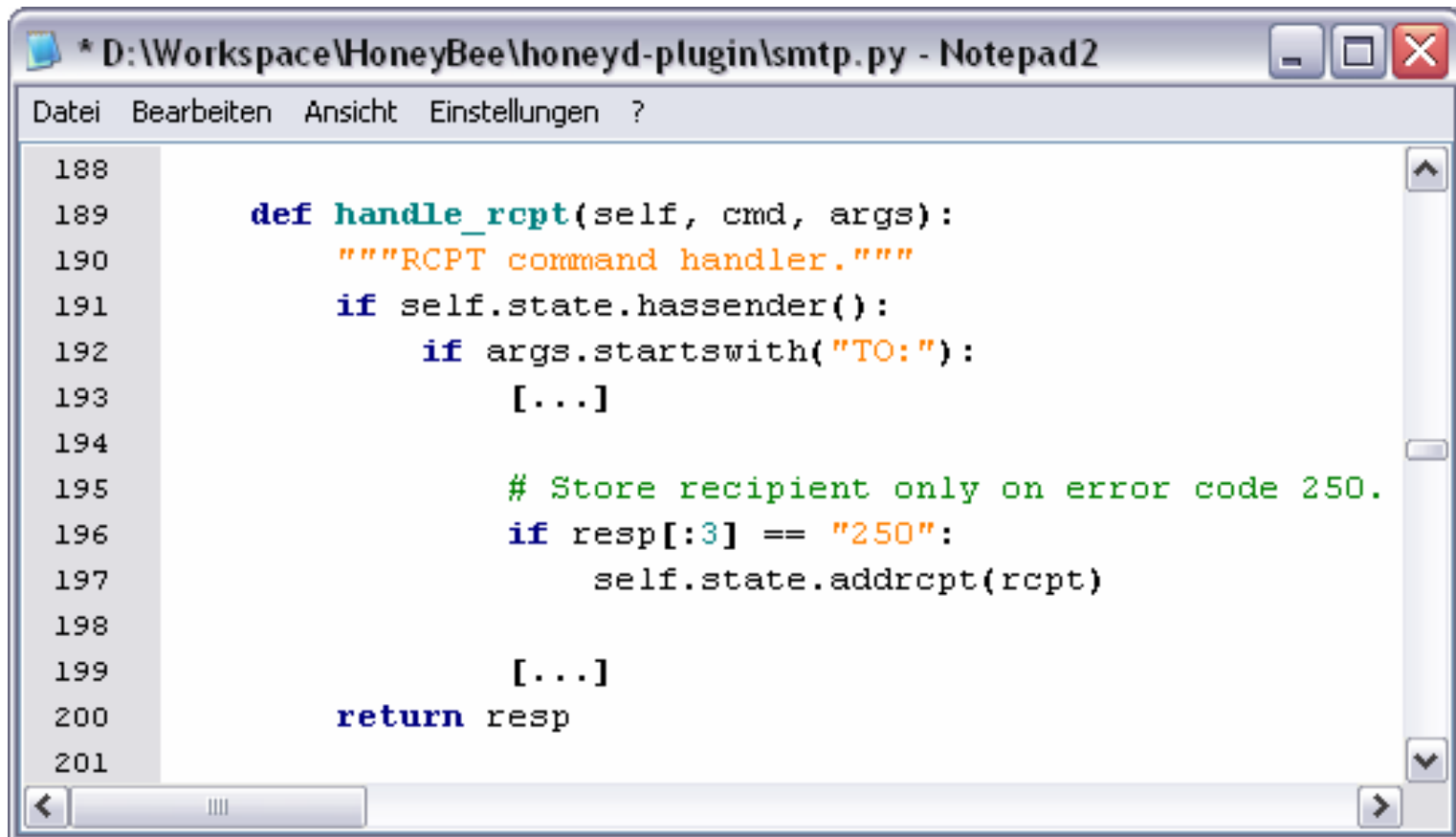
Der Emulator



The image shows a Notepad2 window with the title bar "* D:\Workspace\HoneyBee\honeyd-plugin\smtp.py - Notepad2". The window contains Python code for a help command handler. The code is as follows:

```
287
288 |   def handle_help(self, cmd, args):
289 |       """HELP command handler."""
290 |       if args == "":
291 |           resp = self.fp.getresp(cmd)
292 |       else:
293 |           resp = self.fp.getresp(cmd + " " + args)
294 |           if resp == "KeyError":
295 |               resp = self.fp.getresp(cmd + " <unknowncmd>")
296 |               resp = resp % {"argsupper": args.upper()}
297 |               resp = resp % {"argslower": args.lower()}
298 |       return resp
299
300
301
```


Der Emulator



The image shows a Notepad2 window titled "* D:\Workspace\HoneyBee\honeyd-plugin\smtp.py - Notepad2". The window contains Python code for an SMTP RCPT command handler. The code is as follows:

```
188
189     def handle_rcpt(self, cmd, args):
190         """RCPT command handler."""
191         if self.state.hassender():
192             if args.startswith("TO:"):
193                 [...]
194
195                 # Store recipient only on error code 250.
196                 if resp[:3] == "250":
197                     self.state.addrcpt(rcpt)
198
199                 [...]
200         return resp
201
```

```
thomas@vm-debian:~$ smtpscan 192.168.1.200
```

```
smtpscan version 0.5
```

```
15 tests available
```

```
3184 fingerprints in the database
```

```
Scanning 192.168.1.200 (192.168.1.200) port 25
```

```
15/15
```

```
Result --
```

```
250:501:501:250:553:553:250:214:252:502:502:502:502:250:250
```

```
Banner :
```

```
220 sendmail-emu.apel ESMTP Sendmail 8.13.1/8.13.1; Sat, 18 Dec 2004 13:15:44 +0100 (CET)
```

```
SMTP server corresponding :
```

```
- Sendmail 8.12.8/8.12.8 (with source email address checking like RBL, ...)
```

```
thomas@vm-debian:~$ smtpscan 192.168.1.78
```

```
smtpscan version 0.5
```

```
15 tests available
```

```
3184 fingerprints in the database
```

```
Scanning 192.168.1.78 (192.168.1.78) port 25
```

```
15/15
```

```
Result --
```

```
250:501:501:250:553:553:550:214:252:502:502:502:502:250:250
```

```
Banner :
```

```
220 vm-freebsd.apel ESMTP Sendmail 8.13.1/8.13.1; Tue, 18 Jan 2005 00:31:09 +0100 (CET)
```

```
SMTP server corresponding :
```

```
- Sendmail 8.12.2-8.12.5 (with source email address checking like RBL, ...)
```

```
thomas@vm-debian:~$
```

Die Schwierigkeiten

- Scanner muss zu Servern passen
 - Dynamische Stellen müssen erkannt und ersetzt werden
 - Zusätzliche Tests erfordern Anpassung der Parser
- Komplexität
 - Immer noch jede Menge Handarbeit
 - Echte Funktionalität, z.B. bei FTP
- Fehlende Funktionen von Honeyd
 - FTP-Datenverbindungen nicht möglich
 - Keine zentrale Konfiguration möglich

Das Fazit

- Das Prinzip funktioniert
 - SMTP täuscht Fingerprinter und nimmt Mails an
 - FTP täuscht Fingerprinter
- Mögliche Erweiterungen und Verbesserungen der Implementierung
 - POP3 ist noch in Arbeit
 - Sonstige Protokolle wie IMAP4, SSH?
 - Zusätzliche Tests
 - Mehr Funktionalität für Emulation