Advanced Custom Network Protocol Fuzzing

Joshua Pereyda

Tim Clemans

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Intro – Outline

- Us
- Fuzzing
- Network Protocol Fuzzing
- Tool Landscape
- Goals
- Intro to boofuzz
- Exercise 1

Intro – Speakers

- Joshua Pereyda
  - Software Engineer in Security
  - Experience fuzzing professionally
  - Maintains boofuzz

@jtpereyda

Intro – Speakers

- Tim Clemans
  - Software Engineer in Security
  - Experience fuzzing professionally
  - boofuzz apprentice

@tim_clemans

Intro – Fuzzing

- Sending corrupt/random/bogus data to a target to elicit failures
- Popular tools
  - American Fuzzy Lop (AFL)
  - Libfuzzer

Intro – Network Protocol Fuzzing

- Commercial Tools
  - Defensics from Synopsys
  - PeachFuzzer from Peach Tech
- Protocols Available
  - Ethernet, ARP, IPv4/6, UDP, TCP, HTTP, SSH, TLS, FTP, 61850, BGP, Bluetooth family, BACNET, CIP, IKEv2, JSON, Kerberos, Modbus, NFSv3, NTP, PTP, RADIUS, SMTP, SNMP, iSCSI, …
- Benefits of Commercial Tools
  - Ready-to-go protocol definitions
- Note: Engineering acumen required!

Intro – Why Open Source

- Custom Protocol Definition
- Better User Experience
- Benefits
  - Custom protocols without proprietary lock-in
  - Potentially better use experience
  - Cost
- May not be cost effective if a protocol can be purchased
- More Fun

Intro – Which Tool to Use?

- Do you have lots of money AND want to fuzz a well-known protocol?
  - Commercial Tool
- Do you have lots of time OR want to fuzz an obscure/custom/proprietary protocol?
  - Open Source Framework

Intro – Course Goals

- Write your own protocol specification in boofuzz
- Use your fuzzer to find bugs
- Practice reverse engineering an unknown protocol (and writing a fuzzer)

Intro – Open Source Tools

- Spike
- Sulley
- boofuzz
- Kitty

Intro – Open Source Tools

- Spike
- Sulley
- boofuzz
- Kitty

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Basic Techniques

- Understand Your Protocol
- Define Some Messages
- Connect Your Messages

Basic Techniques – Understand Your Protocol

- RFCs/Specifications
- Packet Captures
- Experimentation

boofuzz – Boilerplate

```
from boofuzz import *
# ...
session = Session(
    target=Target(
        connection=SocketConnection(target_host,
                                    target_port,
                                    proto='tcp'),
    )
)
```

boofuzz – Define Messages

```
s_initialize("user")
s_string("USER")
s_delim(" ")
s_string(username.encode('ascii'))
s_static("\r\n")
```

boofuzz – Define Messages

```
s_initialize("pass")
s_string("PASS")
s_delim(" ")
s_string(password.encode('ascii'))
s_static("\r\n")
```

boofuzz – Define Messages

```
s_initialize("stor")
s_string("STOR")
s_delim(" ")
s_string("AAAA")
s_static("\r\n")
```

boofuzz – Define Messages

```
s_initialize("retr")
s_string("RETR")
s_delim(" ")
s_string("AAAA")
s_static("\r\n")
```

boofuzz – Connect Messages

```
session.connect(s_get("user"))
session.connect(s_get("user"), s_get("pass"))
session.connect(s_get("pass"), s_get("stor"))
session.connect(s_get("pass"), s_get("retr"))
```

boofuzz – post_send()

```
def ftp_check(target, fuzz_data_logger, session, sock, *args, **kwargs):
    target.close()
    target.open()
    recv_banner(target=target,
                fuzz_data_logger=fuzz_data_logger, session=session)
    target.send('USER {0}\r\n'.format('admin'))
    reply = target.recv(10000)
    fuzz_data_logger.log_check('Checking reply matches regex
/{0}/'.format(ftp_reply_regex.pattern))
    if re.search(ftp_reply_regex, reply):
        fuzz_data_logger.log_pass('Match')
    else:
        fuzz_data_logger.log_fail('No match')
```

```
session.post_send = ftp_check
```

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Exercise 1 – Target Practice

- Target: HTTP Server on Linux
- Follow your handout
- Go!

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

More Techniques – Review

- Message Definition
- Connecting Messages
- post_send() Method
- Process Monitor (procmon)
- Reproducing Failures
- callback() Method

More Techniques – callback() Method

- Like post_send(), but called right before a node is sent/fuzzed
- Good for handling protocol interactions
- Example: FTP servers send a banner as soon as they are opened, which should be received by the client before any message is sent

More Techniques – callback() Method

```
def recv_banner(target, fuzz_data_logger, session, *args, **kwargs):
    banner = target.recv(10000)
    fuzz_data_logger.log_check('Checking banner matches regex
/{0}/'.format(ftp_reply_regex.pattern))
    if re.search(ftp_reply_regex, banner):
        fuzz_data_logger.log_pass('Match')
    else:
        fuzz_data_logger.log_fail('No match')
```

```
session.connect(s_get("user"), callback=recv_banner)
session.connect(s_get("user"), s_get("pass"))
session.connect(s_get("pass"), s_get("stor"))
session.connect(s_get("pass"), s_get("retr"))
```

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Exercise 2 – Target Practice 2

- Target: FTP Server on Windows
- Follow your handout
- Remember to review log output of feature_check() to verify your interactions
- Go!

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Reverse Engineering – Outline

- Use Wireshark
- Analyze PCAPs
- Context
- ???
- Fuzzer!

Reverse Engineering – Analysis Tips

- Look at multiple samples
- Look for static values or sometimes-changing values
- Look for data structures
- Sequences of zeros may be
  - Filler bytes for fixed length fields
  - Unused fields
- Look at request vs reply format
- Look for hints
  - What else is happening in the PCAP?

Reverse Engineering – Analysis Example

- Two request+reply pairs
- Functionally similar
- Different hosts
- We will compare
  - Request A vs Request B
  - Reply A vs Reply B
  - Request vs Reply

Request A

```
0000    00 04 00 01 00 06 00 00 17 00 e4 88 00 00 08 00
0010    45 00 00 60 e6 c3 40 00 40 06 3d c5 0a 00 01 05
0020    0a 00 01 0b d6 df 08 01 94 68 ae 69 f4 2f b6 0e
0030    80 18 00 d2 16 62 00 00 01 01 08 0a 00 0f 48 a9
0040    1c 62 1b a3 80 00 00 28 d1 82 5e 7d 00 00 00 00
0050    00 00 00 02 00 01 86 a3 00 00 00 03 00 00 00 00
0060    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Request B

```
0000    00 04 00 01 00 06 00 00 17 00 2a ed 00 00 08 00
0010    45 00 00 60 13 c7 40 00 40 06 12 af 0a 00 00 11
0020    0a 00 00 12 03 1f 08 01 ff 30 1f 56 86 fd 15 cb
0030    80 18 00 d2 14 75 00 00 01 01 08 0a 07 f3 0c 90
0040    07 f1 8c fd 80 00 00 28 5c 1c ef 1b 00 00 00 00
0050    00 00 00 02 00 01 86 a3 00 00 00 04 00 00 00 00
0060    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Request A

```
0000    00 04 00 01 00 06 00 00 17 00 e4 88 00 00 08 00
0010    45 00 00 60 e6 c3 40 00 40 06 3d c5 0a 00 01 05
0020    0a 00 01 0b d6 df 08 01 94 68 ae 69 f4 2f b6 0e
0030    80 18 00 d2 16 62 00 00 01 01 08 0a 00 0f 48 a9
0040    1c 62 1b a3 80 00 00 28 d1 82 5e 7d 00 00 00 00
0050    00 00 00 02 00 01 86 a3 00 00 00 03 00 00 00 00
0060    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Reply A

```
0000    00 00 00 01 00 06 00 00 17 00 97 11 00 00 08 00
0010    45 00 00 50 ad ae 40 00 40 06 76 ea 0a 00 01 0b
0020    0a 00 01 05 08 01 d6 df f4 2f b6 0e 94 68 ae 95
0030    80 18 00 d2 62 c3 00 00 01 01 08 0a 1c 62 1b a3
0040    00 0f 48 a9 80 00 00 18 d1 82 5e 7d 00 00 00 01
0050    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Reply A

```
0000    00 00 00 01 00 06 00 00 17 00 97 11 00 00 08 00
0010    45 00 00 50 ad ae 40 00 40 06 76 ea 0a 00 01 0b
0020    0a 00 01 05 08 01 d6 df f4 2f b6 0e 94 68 ae 95
0030    80 18 00 d2 62 c3 00 00 01 01 08 0a 1c 62 1b a3
0040    00 0f 48 a9 80 00 00 18 d1 82 5e 7d 00 00 00 01
0050    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Reply B

```
0000    00 00 00 01 00 06 00 00 17 00 b1 07 00 00 08 00
0010    45 00 00 50 99 37 40 00 40 06 8d 4e 0a 00 00 12
0020    0a 00 00 11 08 01 03 1f 86 fd 15 cb ff 30 1f 82
0030    80 18 00 d2 26 46 00 00 01 01 08 0a 07 f1 8c fd
0040    07 f3 0c 90 80 00 00 18 5c 1c ef 1b 00 00 00 01
0050    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Reply B

```
0000    00 00 00 01 00 06 00 00 17 00 b1 07 00 00 08 00
0010    45 00 00 50 99 37 40 00 40 06 8d 4e 0a 00 00 12
0020    0a 00 00 11 08 01 03 1f 86 fd 15 cb ff 30 1f 82
0030    80 18 00 d2 26 46 00 00 01 01 08 0a 07 f1 8c fd
0040    07 f3 0c 90 80 00 00 18 5c 1c ef 1b 00 00 00 01
0050    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Exercise 3 – Reverse Engineering

Instructions

- Analyze protocol format
- Create fuzz definitions
- Run fuzzer
- Iterate
- Tips
  - You have a working server – experiment!
  - Dive as far into the protocol as possible, but build your MVP first

What You Know

- Samples are from the same client, but sometimes with different settings
- This is a proprietary license management protocol – anticipate obfuscation

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Outline

- Lecture: Intro
- Lecture: Basic Techniques with boofuzz
- Exercise 1 – Target Practice
- Lecture: More Techniques
- Exercise 2 – Target Practice 2
- Lecture: Reverse Engineering
- Exercise 3 – Reverse Engineering
- Lecture: Advanced Topics & Role of Fuzzing in Offense & Defense

Advanced Topics – Outline

- Custom Transport Layers
- Future of boofuzz
- Thank you!

Advanced Topics – Custom Transport Layers

- Implement the `ITargetConnection` interface
- `SocketConnection` provides typical use case
- `SerialConnection` for serial ports

Advanced Topics – Custom Transport Layers

```
session = Session(
    target=Target(
        connection=SerialConnection(port=1,
                                    baudrate=9600),
    ),
    sleep_time=sleep_between_cases,
)
```

Advanced Topics – Future

- More robust data model – use Python construct package?
- More features
- Less bugs
  - Hot tip: If you think you found a bug in boofuzz… you probably did! :P
- Better built-in Cli
- Quality Protocol Definitions

Conclusion

- We learned how to…
  - Define protocols in boofuzz
  - Identify bugs using a network protocol fuzzer
  - Reverse engineer an unknown network protocol
- Remember:
  - Commercial tools are great if your employer can pay for them
  - Open source tools are best for custom protocols… and more fun! ☺

Thank you and happy fuzzing!

- https://github.com/jtpereyda/boofuzz
- https://github.com/jtpereyda/boofuzz-ftp
- https://github.com/jtpereyda/boofuzz-http

- @boofuzz
- @jtpereyda
- @tim_clemans