Using the Sulley Fuzzing Framework for Generation Fuzzing

Dr. Jared DeMott
CTO AND FOUNDER

@jareddemott www.vdalabs.com

Overview

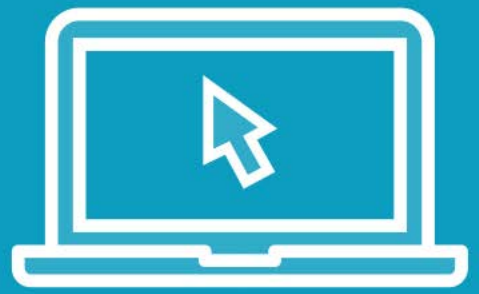
Fuzzing framework

Commercial tools

- Synopsys
- Peach

Sulley

Demo



Fuzzing with Sulley

Generation Fuzzing

Need a good data model

- Protocol knowledge or “Intelligent fuzzing”
  - Allows the fuzzer to get *deep* into the protocol
  - More expensive to build

Intelligent Fuzz Tools

Old: SPIKE

- Manually create a definition file that (mostly) describes the protocol
  - Based on either the RFC, a network sniff, or reverse engineering

Unmaintained: Sulley

- TLS example code in fuzzing book

Peach community

- Uses a sample input
- Parses with definition file to create output

Drawbacks of Frameworks

Requirements

- **network_monitor.py**: CORE Pcapy, CORE Impacket
- **process_monitor.py**: PaiMei
  - PaiMei has a bunch of requirements
    - Pydbg, python, ctypes, etc.
- **vmcontrol.py**: VMWare

Learning curve

- Understanding how
  - Pieces of framework fit together
  - Write data/state protocol description
  - Setup the monitors and make the whole thing “go”

Which Framework?

In-house

Open source

- Sulley, Peach community
  - Not well maintained

Commercial

- Synopsys
- Peach Pro

Sulley

Network fuzzing

- **Data Representation**
  - Break down the protocol into individual *requests* and represent that as blocks
- **Session**
  - Link requests together to form a *session*, attach the various available Sulley *agents* and fuzz
- **Post Mortem**
  - Review crash results
  - Replay individual test cases

```
from sulley import *

s_initialize("HTTP BASIC")

s_group("verbs", values=["GET", "HEAD", "POST", "TRACE"])

if s_block_start("body", group="verbs"):
    s_delim(" ")
    s_delim("/")
    s_string("index.html")
    s_delim(" ")
    s_string("HTTP")
    s_delim("/")
    s_string("1")
    s_delim(".")
    s_string("1")
    s_static("\r\n\r\n")
s_block_end("body")
```

Sulley Request to Fuzz a HTTP server

```
from sulley import *

from requests import myhttp


sess = sessions.session(session_filename="audits/myhttp/myhttp.session",
sleep_time=.25, log_level=10)


target = sessions.target("192.168.0.104", 80)

target.procmon = pedrpc.client("127.0.0.1",  26002) #started separately


sess.add_target(target)

sess.connect(s_get(“HTTP BASIC"))

sess.fuzz()
```

Types Matter

Numbers

- s_byte|word|dword|qword

Strings

- s_string: fuzz lib

Delimiters

- s_delim: ‘ ‘, ‘\’, ‘<‘, in ascii protocols

Static

- s_static: String that must be present

Fixers

- Checksums, hash, len, encoders, etc.

Summary

Introduced fuzzing framework

- Data model
- Monitoring

Continue with Peach next