SIEMENS

Corporate Technology

Fuzzing Proprietary Protocols

A Practical Approach

Dr. Thomas Pröll
Siemens CERT

Overview

Fuzzing: What is it?
Proprietary Protocols
Fuzzing Proprietary Protocols
Results

SIEMENS

Fuzzing: What is it?

Fuzz testing or **fuzzing** is a **software testing** technique that provides **invalid**, unexpected, or random data to the **inputs of a program**. If the program fails (for example, by crashing or failing built-in code assertions), the defects can be noted.

(Source: http://en.wikipedia.org/wiki/Fuzz_testing)

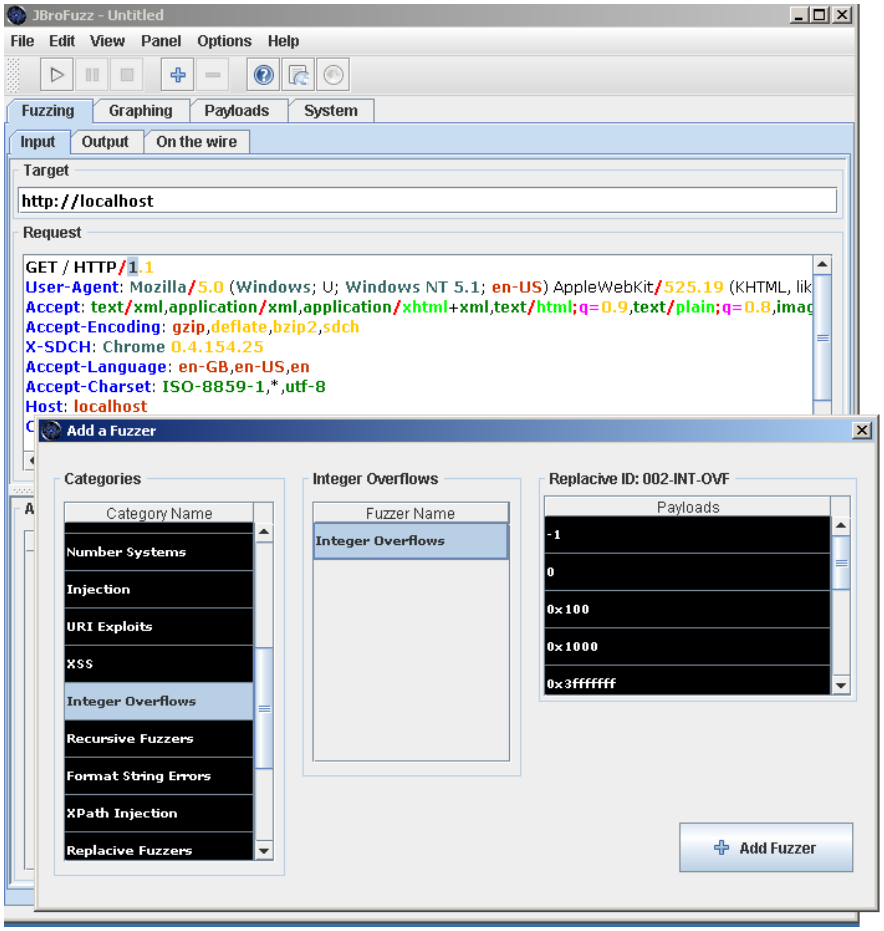
Fuzz testing or **Fuzzing** is a Black Box software testing technique, which basically consists in **finding implementation bugs** using malformed/semi-malformed data injection in an **automated fashion**.

(Source: http://www.owasp.org/index.php/Fuzzing)

Fuzzing: What is it?

Fuzzing is protocol dependant:

- Encoding (ASCII ↔ Binary)
- Architecture (Stateful ↔ Stateless)
- Specification (Open ↔ Proprietary)
- Complexity
  - Sequence numbers
  - Time stamps
  - Handshakes
  - Checksums
  - Encryption
  - Compression



Proprietary Protocols

Proprietary protocols are used in:

- Industrial environment (Electricity, Oil&Gas, Transportation, SCADA etc.)
- Instant messengers (Skype, MSN, QQ etc.)
- Online games
- As HTTP payload
- …

SIEMENS

Fuzzing Proprietary Protocols 1

Fuzzing by completely random data with Netcat:

```
cat /dev/urandom | nc 192.168.0.1 1234
```

Pro:
- Simple
- Combines flooding and fuzzing

Contra:
- Stateless
- Most simple input validation will filter it out

Very effective baseline check

SIEMENS

Fuzzing Proprietary Protocols 2

Fuzzing by replay with tcpreplay:

- Capture pure client traffic e.g. with wireshark
- Insert random errors e.g. with editcap
- Repair TCP/UDP/IP checksums with editcap
- Replay the traffic with tcpreplay/flowreplay

Pro:

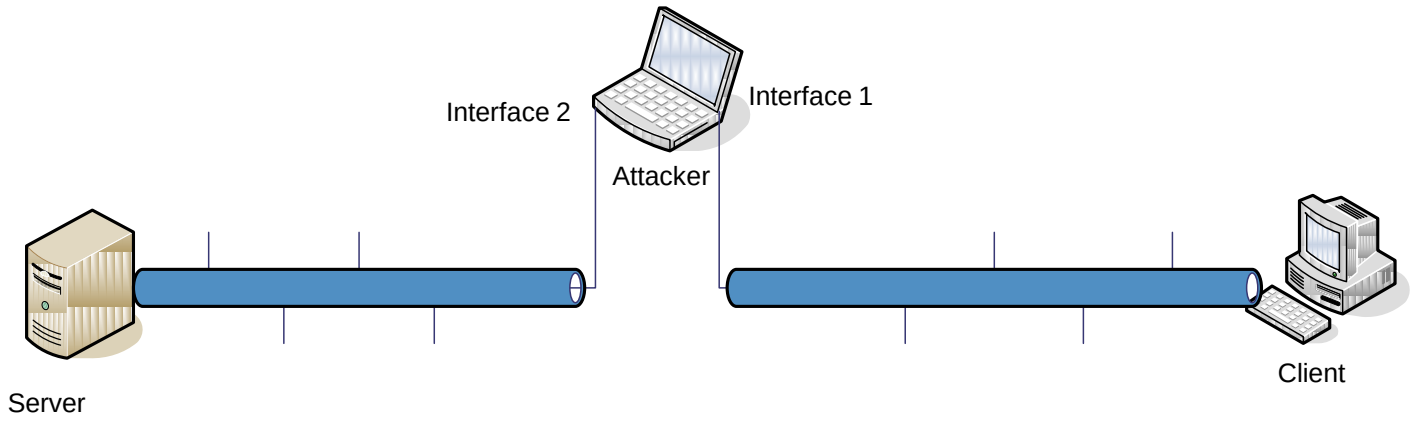
- Traffic looks much like original traffic
- Avoids some input validation checks

Contra:

- Sequence numbers (TCP and application specific)
- Application specific checks (checksums, time stamps etc.)
- Mostly stateless

Fuzzing Proprietary Protocols 3

Man-in-the-middle or Inline Fuzzing:



- Capture traffic on interface x
- Manipulate
- Replay traffic on interface y

Fuzzing Proprietary Protocols 3

Inline Fuzzing Pro:

- Stateful, handshakes work
- Sequence numbers, time stamps etc. match
- Protocol stack used without own implementation

Inline Fuzzing Contra:

- Application layer checksums
- With encryption, only decryption algorithm attacked

SIEMENS

Fuzzing Proprietary Protocols 3

Inline fuzzing caveats:

- Keep track, which hosts are on which interface
- Adjust TCP/UDP/IP checksums
- Ways to influence the manipulation:
  - Where to manipulate (Packet header? Packet Payload?)
  - What to manipulate (Random? String replacement? Overflows?)
  - When to manipulate (Handshake? Payload?)
  - How much to manipulate (50%? 0.01%?)
- Logging of manipulations

SIEMENS

Fuzzing Proprietary Protocols 3

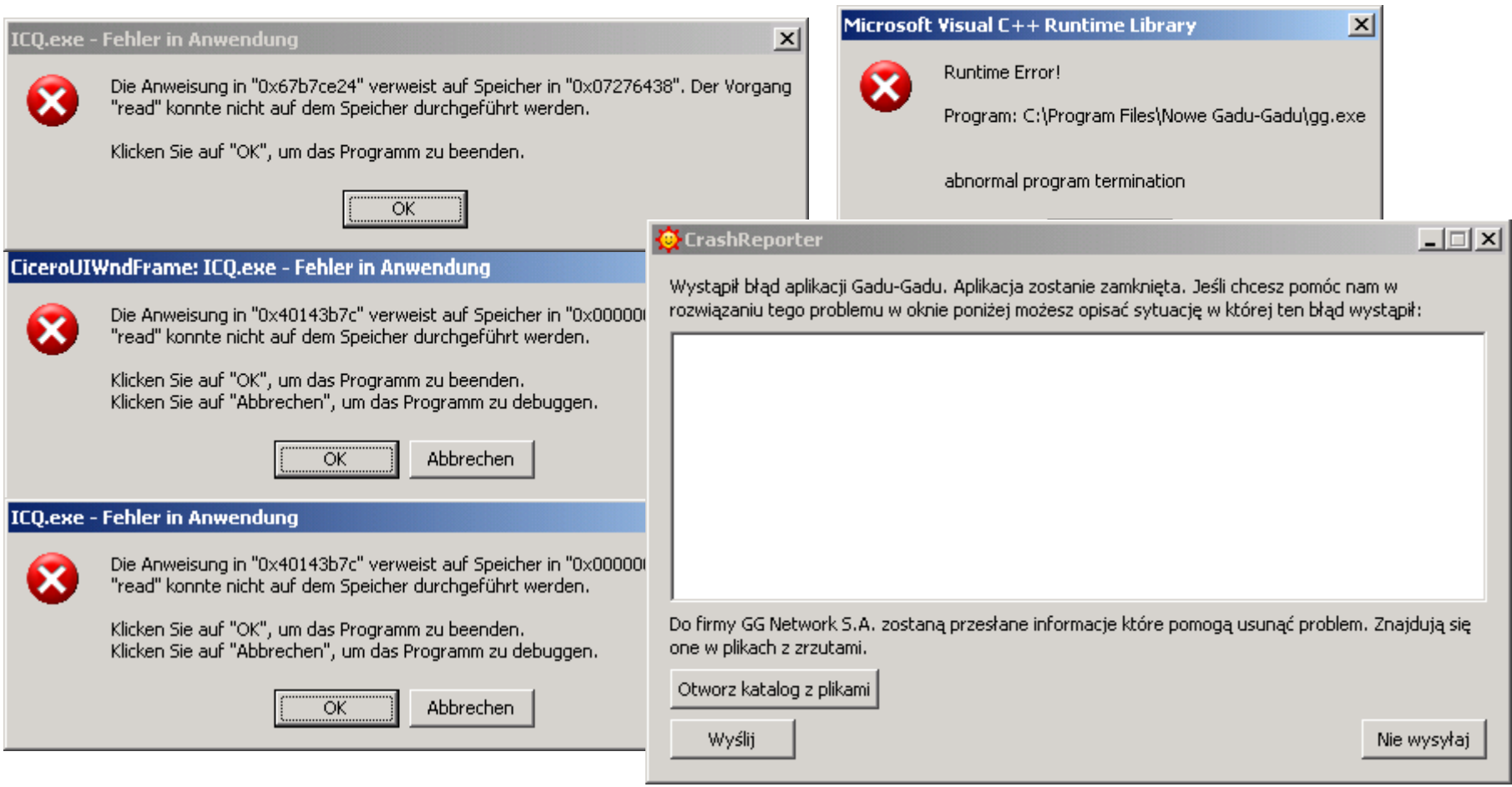
Available software for Inline Fuzzing:

- quefuzz (AKA querub)
- Proxyfuzz
- Achilles Satellite (commercial)

Siemens CERT implementations:

- tcpdump → editcap → tcpreplay (slow, only limited ports)
- tcpbridge addon (C-Code)

Results



Used programs

JBroFuzz:	http://sourceforge.net/projects/jbrofuzz/
Netcat:	http://netcat.sourceforge.net/
Proxyfuzz:	http://theartoffuzzing.com/
Quefuzz:	http://code.google.com/p/quefuzz/
TCPdump:	http://www.tcpdump.org/
TCPreplay:	http://tcpreplay.synfin.net/
Wireshark:	http://www.wireshark.org/

Contact

Dr. Thomas Pröll

Siemens AG
Corporate Technology
Corporate Research and Technologies
CT T DE IT1
Otto-Hahn-Ring 6
81739 München, Deutschland

thomas.proell@siemens.com

