RUHR UNIVERSITÄT BOCHUM RUB

VUSec

CISPA

Nyx-Net

Network Fuzzing with Incremental Snapshots

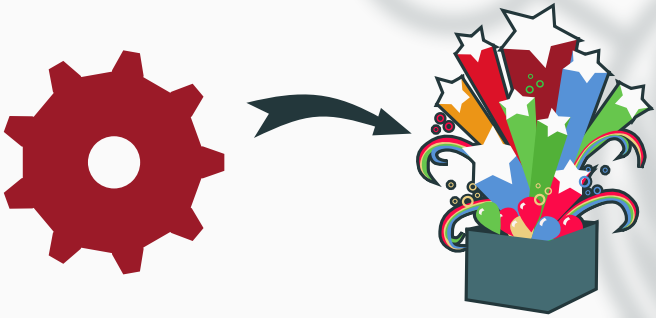
Sergej Schumilo[1], Cornelius Aschermann[1], Andrea Jemmett[2], Ali Abbasi[1] and Thorsten Holz[3]

[1] Ruhr-Universität Bochum
[2] Vrije Universiteit Amsterdam
[3] CISPA Helmholtz Center for Information Security

Fuzzing in a Nutshell

Fuzzing

Fuzzing



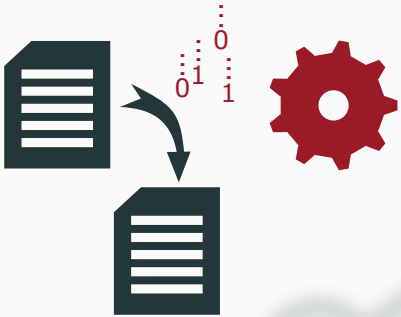
Fuzzing

Modern Fuzzers

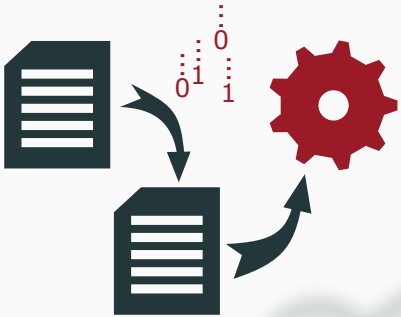
Modern Fuzzers

Modern Fuzzers

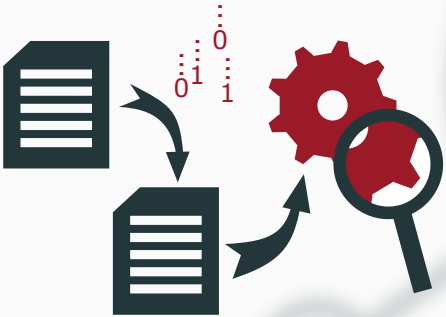
Modern Fuzzers



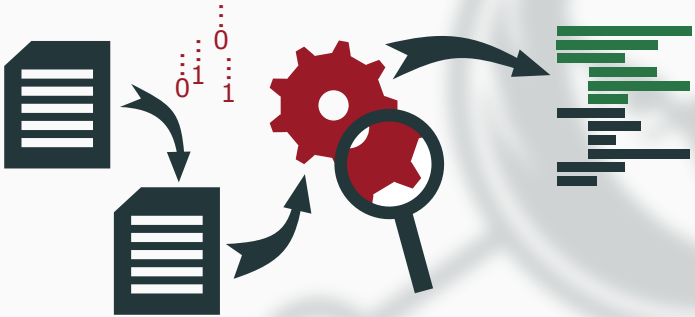
Modern Fuzzers



Modern Fuzzers

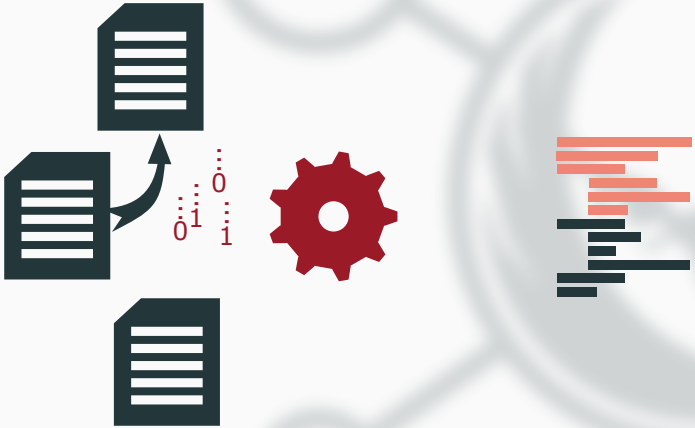


Modern Fuzzers

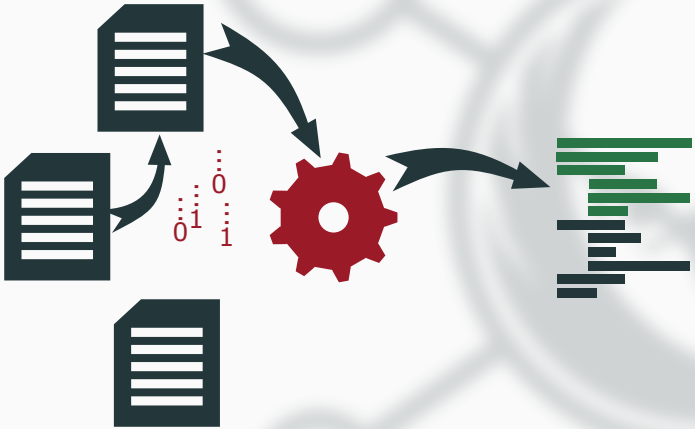


Modern Fuzzers

Modern Fuzzers



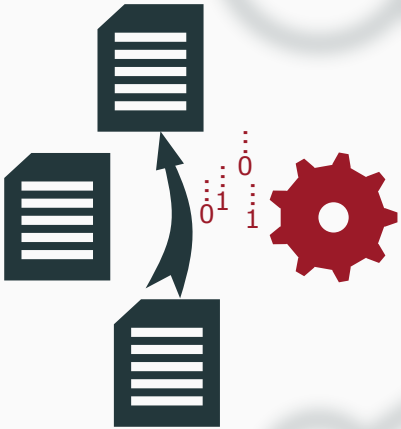
Modern Fuzzers



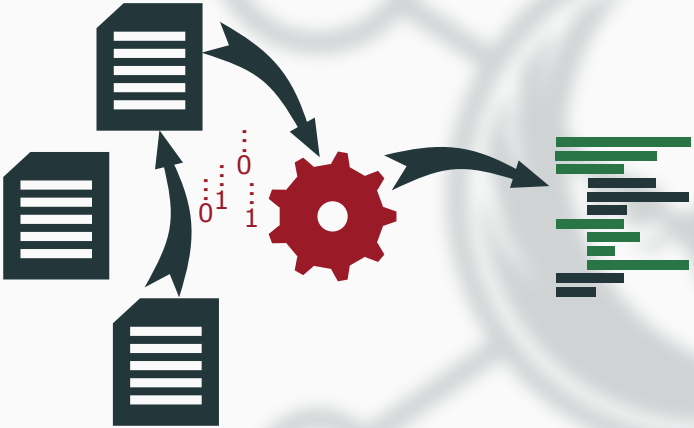
Modern Fuzzers

Modern Fuzzers

Modern Fuzzers



Modern Fuzzers

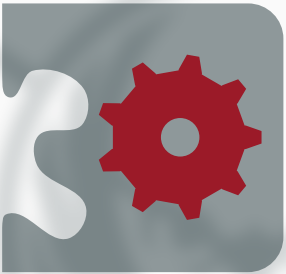


Modern Fuzzers

Motivation

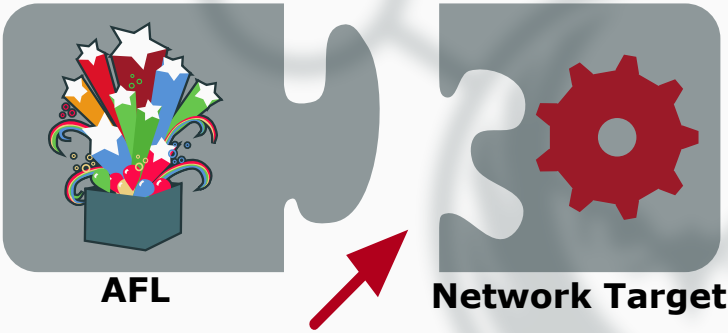
Motivation

AFL

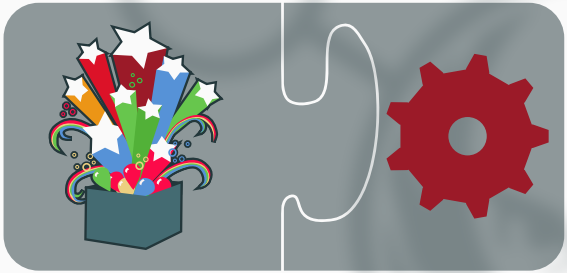


Network Target

Motivation



Motivation



Pham et al. **"AFLNET: A Greybox Fuzzer for Network Protocols"** (ICST 2020)

Motivation

AFLNet's limitations:

- **requires target modifications**

  (to enable "state machine inference")

Motivation

AFLNet's limitations:

- **requires target modifications**

  (to enable "state machine inference")

- **slow execution** (≈10 execs/sec)

  (due to costly init and de-init procedures)

Motivation

AFLNet's limitations:

- **requires target modifications**

  (to enable "state machine inference")

- **slow execution** (≈10 execs/sec)

  (due to costly init and de-init procedures)

- **poor scalability**

  (same network port, same FS, ...)

Can we do better?

Our Approach

Our Approach

Nyx-Net is an extension of Nyx

https://nyx-fuzz.com

Our Approach

Nyx-Net is an extension of Nyx

- **Target runs in a VM**

  (scalability)

https://nyx-fuzz.com

Our Approach

Nyx-Net is an extension of Nyx

- **Target runs in a VM**

  (scalability)

- **Fast snapshots**

  (determinism)

https://nyx-fuzz.com

Our Approach

Nyx-Net is an extension of Nyx

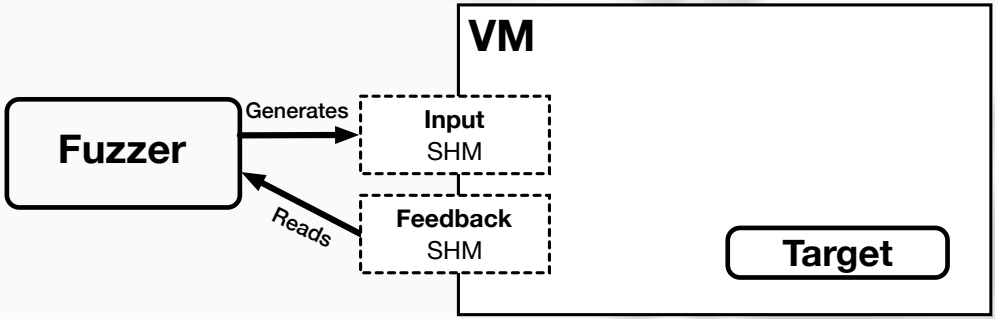
- **Target runs in a VM**
  (scalability)
- **Fast snapshots**
  (determinism)
- **Snapshots & emulation layer**
  (performance)

https://nyx-fuzz.com

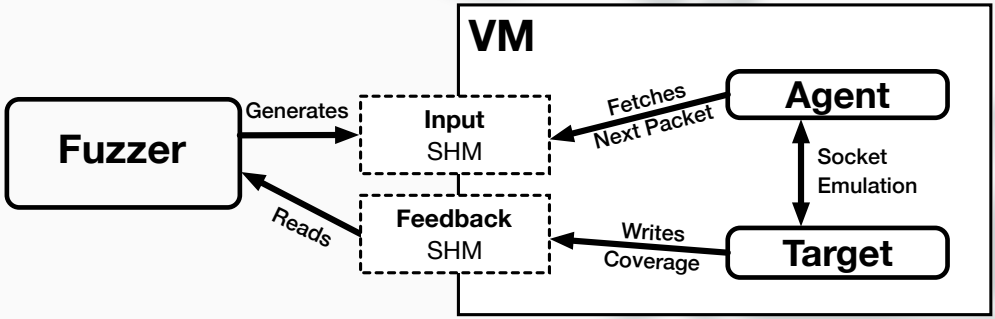
Our Approach: Architecture

Host

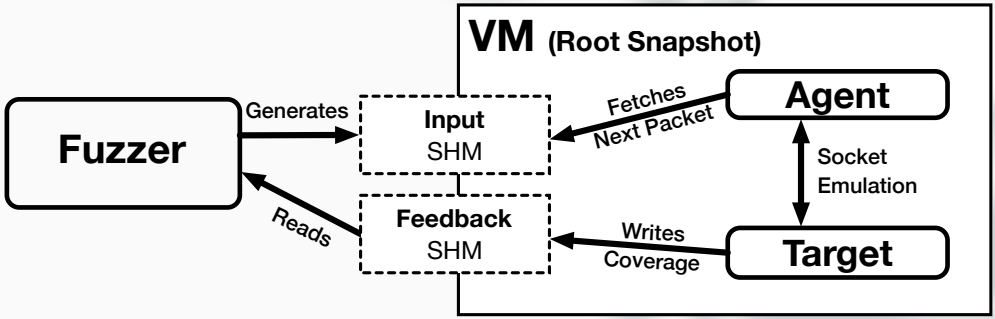
Our Approach: Architecture



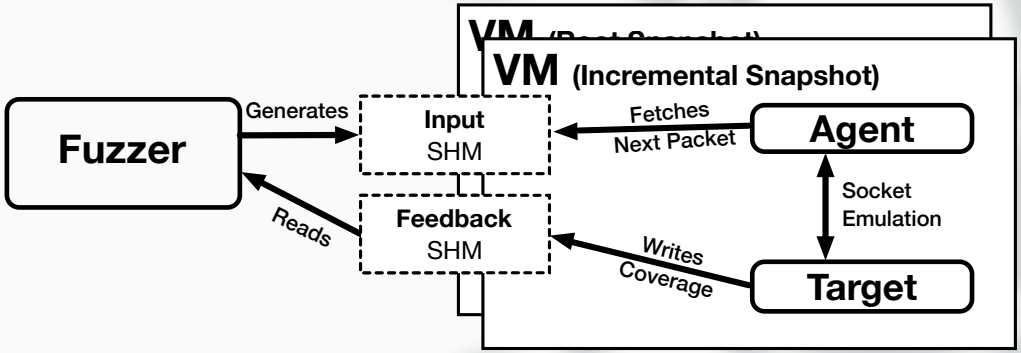
Our Approach: Architecture



Our Approach: Architecture



Our Approach: Architecture



Our Approach: Emulation

glibc emulation layer:

Our Approach: Emulation

glibc emulation layer:

- **avoid kernel as much as possible**

  (improves overall performance)

Our Approach: Emulation

glibc emulation layer:

- **avoid kernel as much as possible**

  (improves overall performance)

- **fuzzing inputs passed via "syscalls"**

  (root snapshot created on first "request")

Our Approach: Emulation

glibc emulation layer:

- **avoid kernel as much as possible**

  (improves overall performance)

- **fuzzing inputs passed via "syscalls"**

  (root snapshot created on first "request")

- **covers most I/O functions**

  (read, recv, recvmsg, select, poll, ...)

Approach: Incremental Snapshots

Input (sequence of messages):

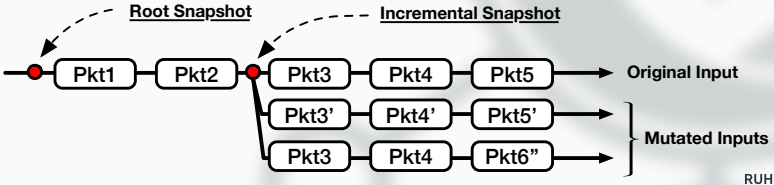
Approach: Incremental Snapshots

Input (sequence of messages):

Approach: Incremental Snapshots

Input (sequence of messages):

Skip message sequence prefix:

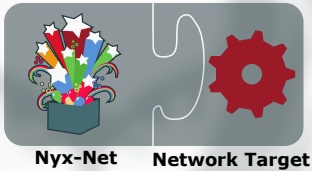


Use Cases

Use Case: Network Fuzzing

Fuzzing-harness for network targets

TCP/UDP as wells server/client support



Use Case: Network Fuzzing

Fuzzing-harness for network targets

TCP/UDP as wells server/client support

Supports complex targets

(fork(), file system changes, ...)

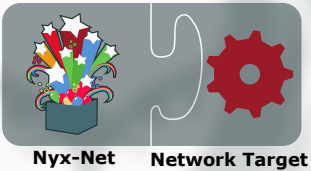
Use Case: Network Fuzzing

Fuzzing-harness for network targets

TCP/UDP as wells server/client support

Supports complex targets

(fork(), file system changes, ...)



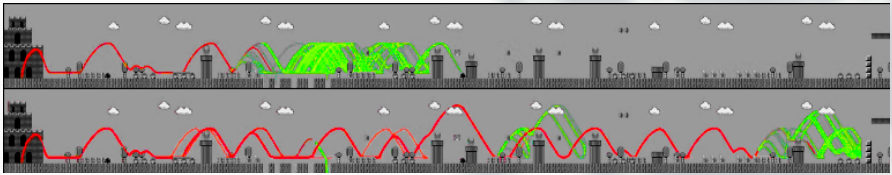
High test throughput (exec/sec)

no need to establish and close connections

Use Case: Super Mario

Solve levels by a fuzzer

Ijon: Exploring Deep State Spaces via Fuzzing (Aschermann et al., IEEE S&P 2022)



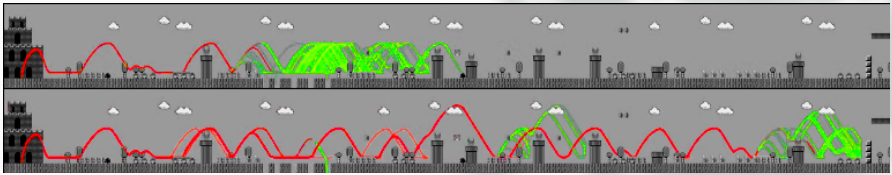
Use Case: Super Mario

Solve levels by a fuzzer

Ijon: Exploring Deep State Spaces via Fuzzing (Aschermann et al., IEEE S&P 2022)

Accelerate Ijon by incremental snapshots

Keystrokes are represented by single "messages"



Use Case: Firefox

Long standing problem at Mozilla:

https://blog.mozilla.org/attack-and-defense/2021/01/27/effectively-fuzzing-the-ipc-layer-in-firefox/

Use Case: Firefox

Long standing problem at Mozilla:

https://blog.mozilla.org/attack-and-defense/2021/01/27/effectively-fuzzing-the-ipc-layer-in-firefox/

- **Emulation-Layer subset** enables fuzzing of the IPC-Subsystem

Use Case: Firefox

Long standing problem at Mozilla:

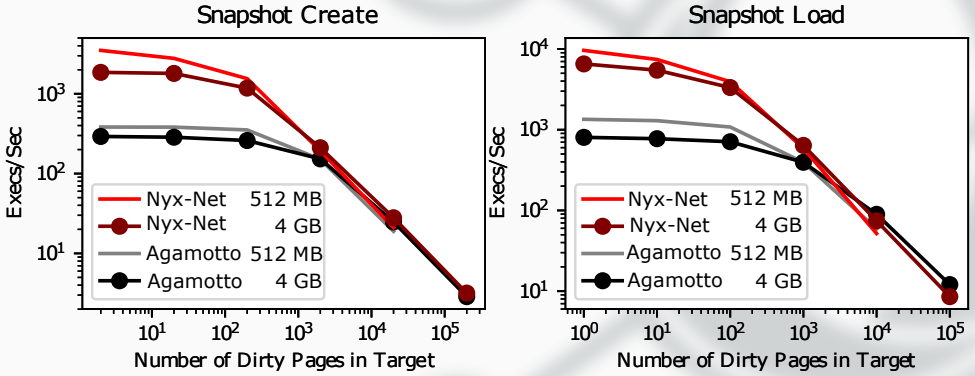
https://blog.mozilla.org/attack-and-defense/2021/01/27/effectively-fuzzing-the-ipc-layer-in-firefox/

- **Emulation-Layer subset** enables fuzzing of the IPC-Subsystem
- Fuzzing of a **fully intilized Browser**

Evaluation

Evaluation: Snapshots

Nyx-Net vs Aggamotto (Song et al., USENIX Security 2020)



Evaluation: ProFuzzBench

Test suite for network fuzzers

ProFuzzBench: A Benchmark for Stateful Protocol Fuzzing - Natella et al., ISSTA 2021

Evaluation: ProFuzzBench

Test suite for network fuzzers

ProFuzzBench: A Benchmark for Stateful Protocol Fuzzing - Natella et al., ISSTA 2021

Set of differnt network targets:

FTP (pure-ftpd, proftpd, ...)

DNS (dnsmsq)

SMTP Server (exim)

DICOM (dcmtk)

...

Evaluation: ProFuzzBench

Crashes found by each fuzzer in PFB (24h experiments)

		Nyx-Net		
Target	**AFLNet**	none	balanced	aggressive
dcmtk	✓	(✓)	(✓)	✓
dnsmasq	✓	✓	✓	✓
exim	-	✓	✓	✓
live555	✓	✓	✓	✓
proftpd	-	✓	✓	✓
pure-ftpd	*	-	-	-
tinydtls	✓	✓	✓	✓

Evaluation: ProFuzzBench

Results compared to AFLNet:

Evaluation: ProFuzzBench

Results compared to AFLNet:

- Up to **100x faster** due to snapshots and emulation layer

Evaluation: ProFuzzBench

Results compared to AFLNet:

- Up to **100x faster** due to snapshots and emulation layer
- **2x higher test throughput** due to incremental snapshots

Evaluation: ProFuzzBench

Results compared to AFLNet:

- Up to **100x faster** due to snapshots and emulation layer
- **2x higher test throughput** due to incremental snapshots
- **More coverage** in 24h compared to AFLNet and other fuzzers

Evaluation: Super Mario

Results compared to Ijon:

Evaluation: Super Mario

Results compared to Ijon:

- **Solves levels up to 4x faster**
  (due to snapshots and no process initalization)

Evaluation: Super Mario

Results compared to Ijon:

- **Solves levels up to 4x faster**
  (due to snapshots and no process initalization)

- **Up to 10x - 30x faster** with incremental snapshots

Evaluation: Firefox

- **3 bugs** found during development

Evaluation: Firefox

- **3 bugs** found during development
- awarded a **special bug bounty** ($20.000)

Evaluation: Firefox

- **3 bugs** found during development
- awarded a **special bug bounty** ($20.000)
- **used at Mozilla** by its security team

Evaluation: Firefox

- **3 bugs** found during development
- awarded a **special bug bounty** ($20.000)
- **used at Mozilla** by its security team
- plans to integrate Nyx-Net into their **CI-Pipeline**

Conclusion

Conclusion

New approach: emulation & VM snapshots

Outperforms state-of-the-art (AFLNet)

Auto-harnessing for network targets

Solved a long-standing problem at Mozilla

Thank You!

Q & A

sergej.schumilo@rub.de

https://github.com/RUB-SysSec/nyx-net