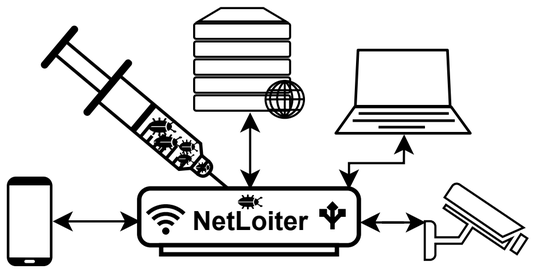
BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY



Excel@FIT 2023

NetLoiter

A Tool for Automated Testing of Network Applications using Fault-Injection

supervisor: Ing. Aleš Smrčka, Ph.D.
Bc. Michal Rozsíval
[michal.rozsival@vut.cz]

Motivation

Development (ideal) environment: fast transmission, low packet loss, reliable communication, ...
Production (real) environment: communication delay, hard and soft errors, communication interruption, ...

Applications are tested in ideal conditions. Using them in a **real environment may introduce new situations**:

- Remotely controlled vehicle – acceleration and subsequent loss of communication
- Client-Server communication with an unpredictably behaving client (e.g., poor internet signal)
- Cyber attacks (e.g., the MITM, impacting of selected packets or data modification)

Idea

Automatic simulation of real network conditions.

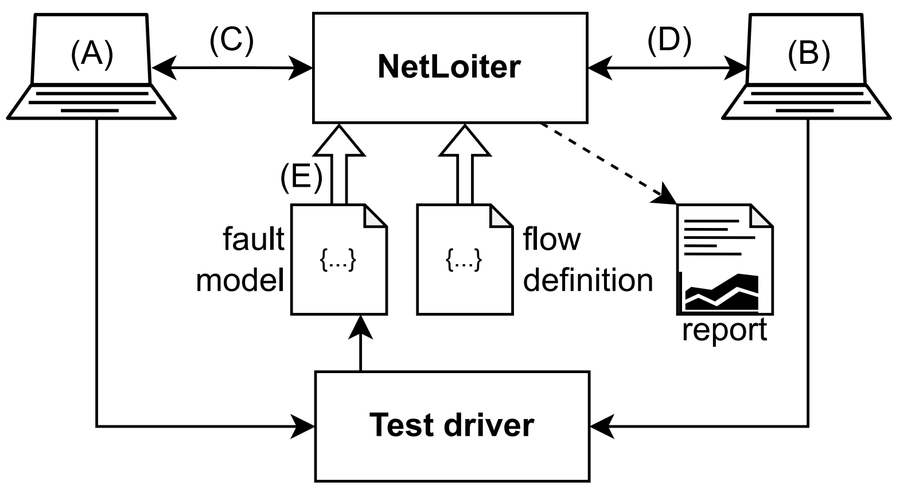


Fig. 1: Structure of the test environment with the NetLoiter.

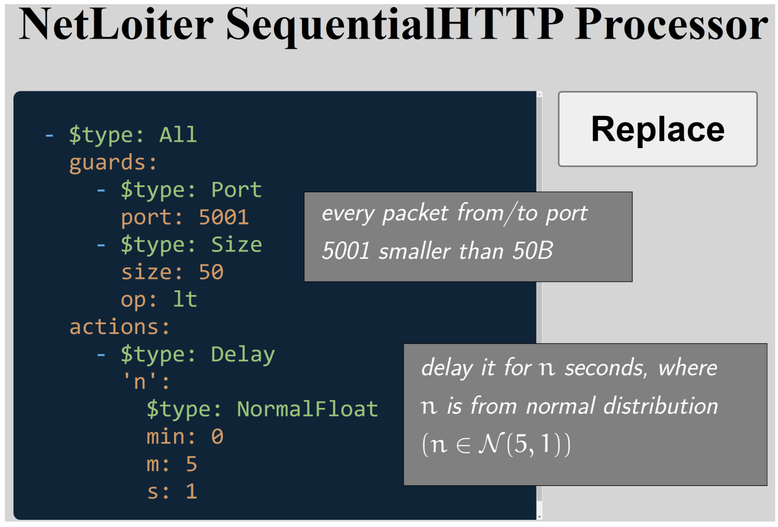


Fig. 2: The test driver can control the NetLoiter via its **RestAPI**.

Deployment

- **Visible software solution** – uses TCP/UDP socket proxy
  Requires changes in communicating applications (redirection)
- **Hidden software solution** – uses NFTables, TC or WFP
  Captures directly in kernel (requires elevated user permissions)
- **Hardware solution** – uses MITM placed in the communication

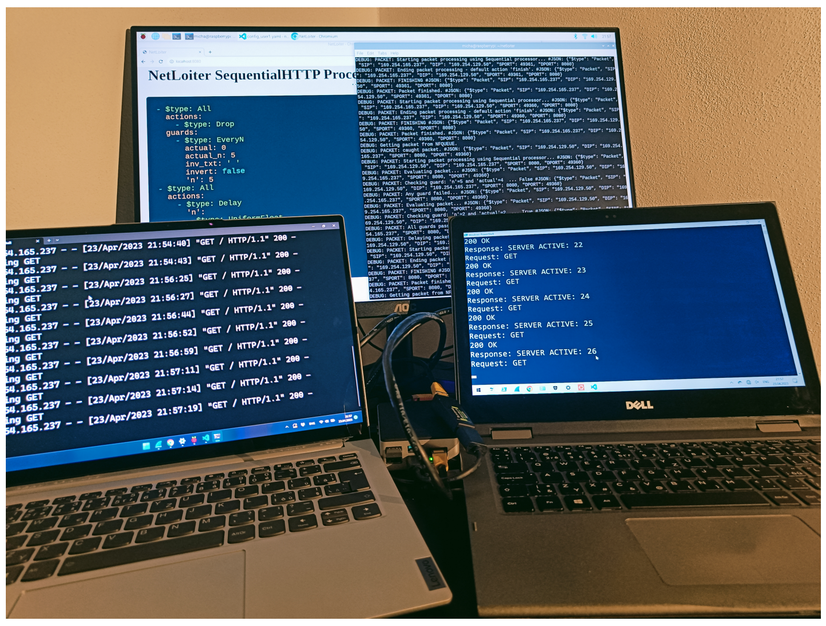


Fig. 3: HW solution using the Raspberry Pi as the man-in-the-middle.

Usage

1. Initialization of an environment (e.g., starting of tested applications)
2. Starting the NetLoiter with a flow definition to communication capturing
3. Updating the simulation scenario with the rules definition
4. Executing tests of the applications affected by the NetLoiter
5. For searching boundary setting, repeat the steps 3. and 4.

The creation of the simulation scenario can be automated!

```
def rule_prob_drop(x):
    return {
        "$type": "All",
        "guards": [ { "$type": "Prob", "x": x } ],
        "actions": [ { "$type": "Drop" } ]
    }
```

```
def test_prob_x_drop_bisection(a, b, eps):
    while a + eps < b:
        m = (a+b)/2
        requests.post(..., rule_prob_drop(m))
        if check_the_tested_applications()
            a = m + eps/2
        else:
            b = m - eps/2
    return m
```

Listing 1: **Automatic** reconfiguration of the NetLoiter with generated rules.

Functionality

- **Guards** – additional packet filtering:
  EveryN, ICMP, IP, Port, Prob, Protocol, Time, Count, TimePeriod, CountPeriod, Size, ...
- **Actions** – executed on selected packets:
  Delay, Drop, Reorder, Replicate, Restart, Finish, Throttle, BitNoise, SocketTCP, ...
- **ValueGenerators** – provides non-constant values to guards/actions:
  Uniform, Normal, Sequence, ...

Evaluation

Table 1: **Roboauto use-case** – affecting communication between the vehicle and the remote station. Table contains the search results for the communicating parties' failure limits. **The results were obtained automatically**.

Injected fault	Duplex	From RS	To RS
Delay for x [s]	$x \geq 0.4$	$x \geq 0.7$	$x \geq 0.7$
Drop every N	$N \leq 4$	$N \leq 2$	$N \leq 1$
Drop with prob. p	$p \geq 0.4$	$p \geq 0.3$	$p \geq 0.95$
Reorder N	$N \geq 12$	$N \geq 2$	$N \geq 17$

VALU3S HAS RECEIVED FUNDING FROM THE ECSEL JOINT UNDERTAKING UNDER GRANT AGREEMENT NO 876852. THE JOINT UNDERTAKING RECEIVES SUPPORT FROM THE EUROPEAN UNION'S HORIZON 2020 RESEARCH AND INNOVATION PROGRAMME AND AUSTRIA, CZECH REPUBLIC, GERMANY, IRELAND, ITALY, PORTUGAL, SPAIN, SWEDEN, TURKEY.
ECSEL Joint Undertaking
Electronic Components and Systems for European Leadership
VALU3S