Improvements for stateful fuzzing

Presented by Martin Vivian

Promoter : Axel Legay

UCLouvain

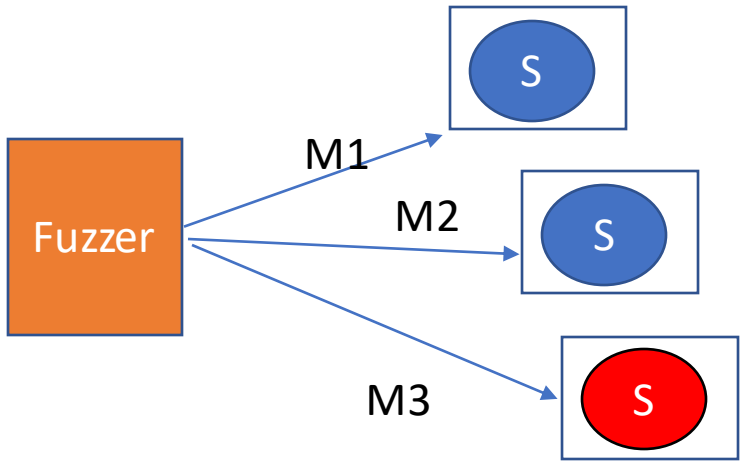
What is fuzzing

- Fuzzing is an efficient testing method to discover vulnerabilities in a system. This approach consists of an automated generation of inputs for a program.

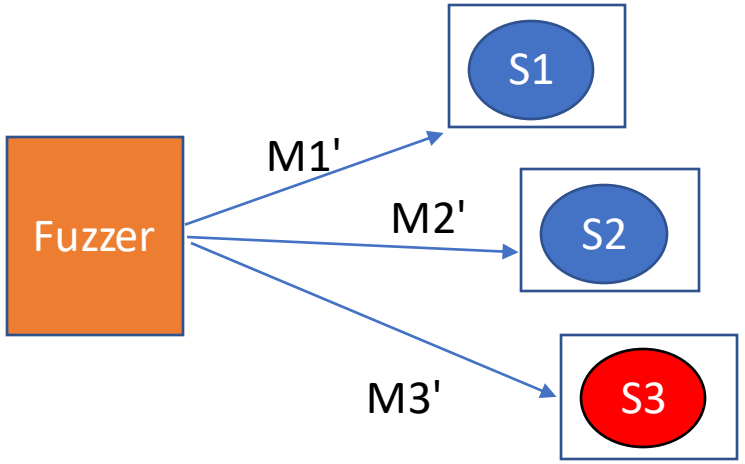
- Two types of programs stateful and stateless

Stateful and Stateless

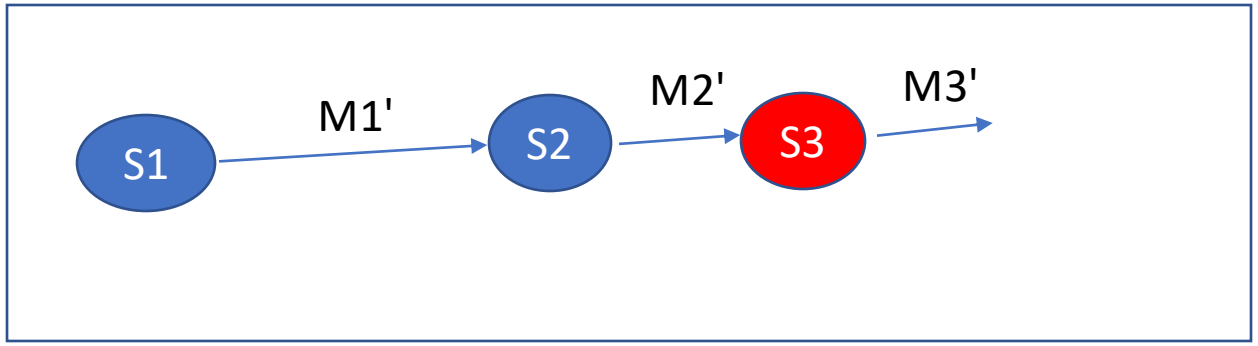
Stateless fuzzing



Stateful fuzzing

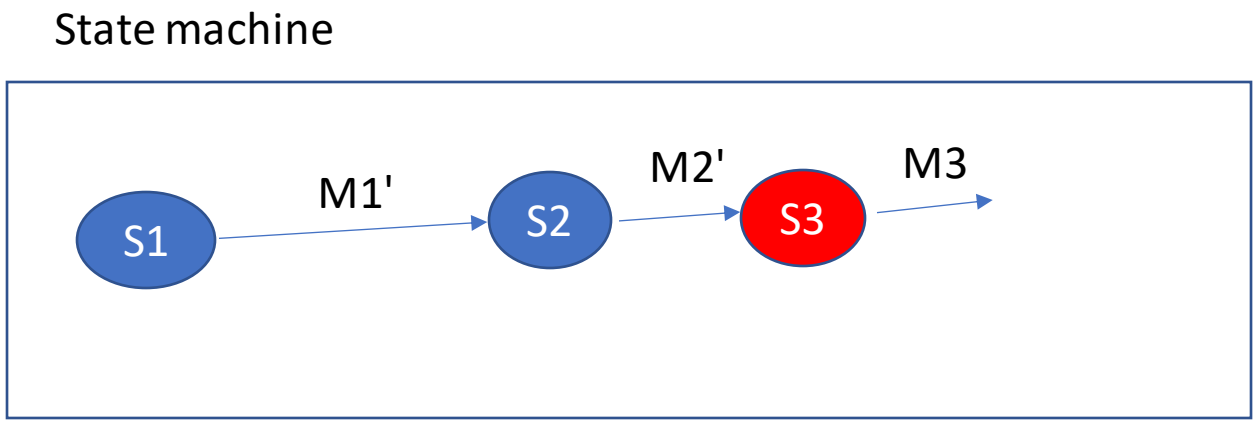


State machine



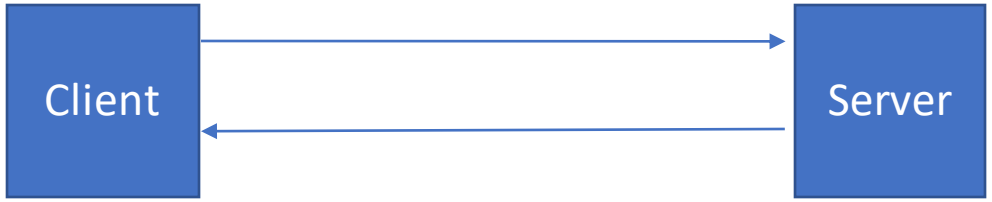
Stateful

- Each message have their grammar
- Order of the message in this example, we must send M1' before M2' to reach S3



Case study

- Two entities that communicate (client/server)
- Trafic not encrypted
- The client must be tested

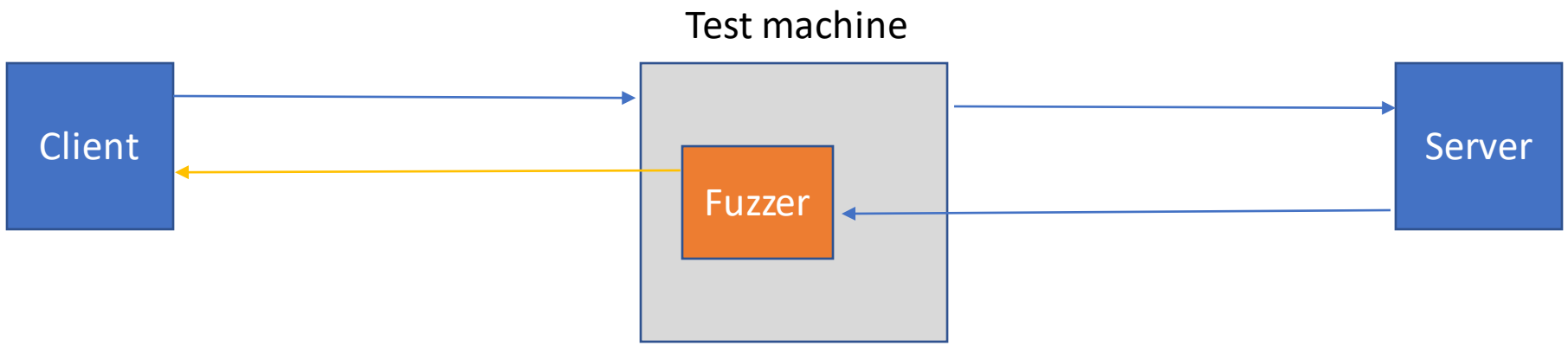


Problem

- Proprietary protocol
- No source code
- Some knowledge about the structure of the frame
- No knowledge about the state machine

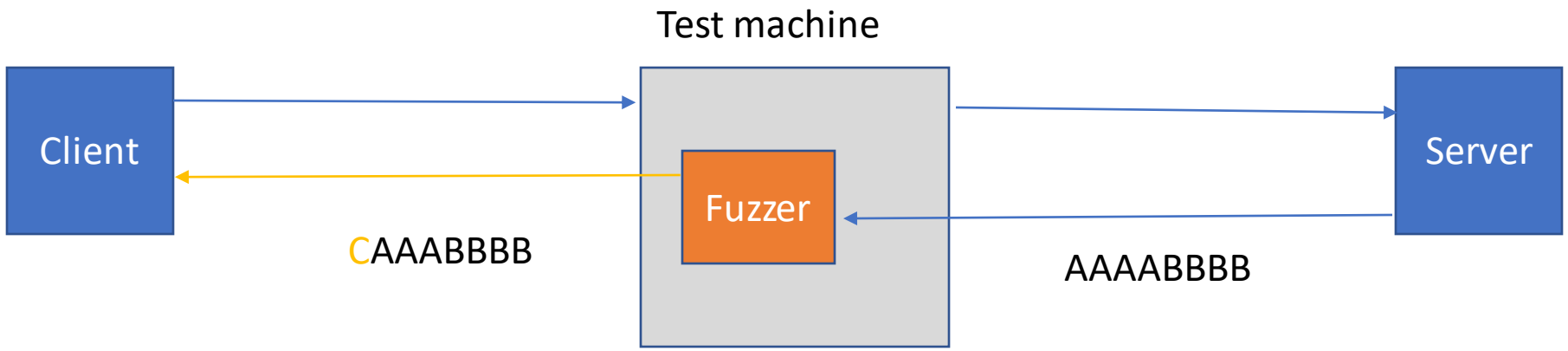
Solution 1 : Fuzzing in MITM

- The fuzzer is between the client and the server
- The fuzzer modifies traffic between the fuzzer and the client



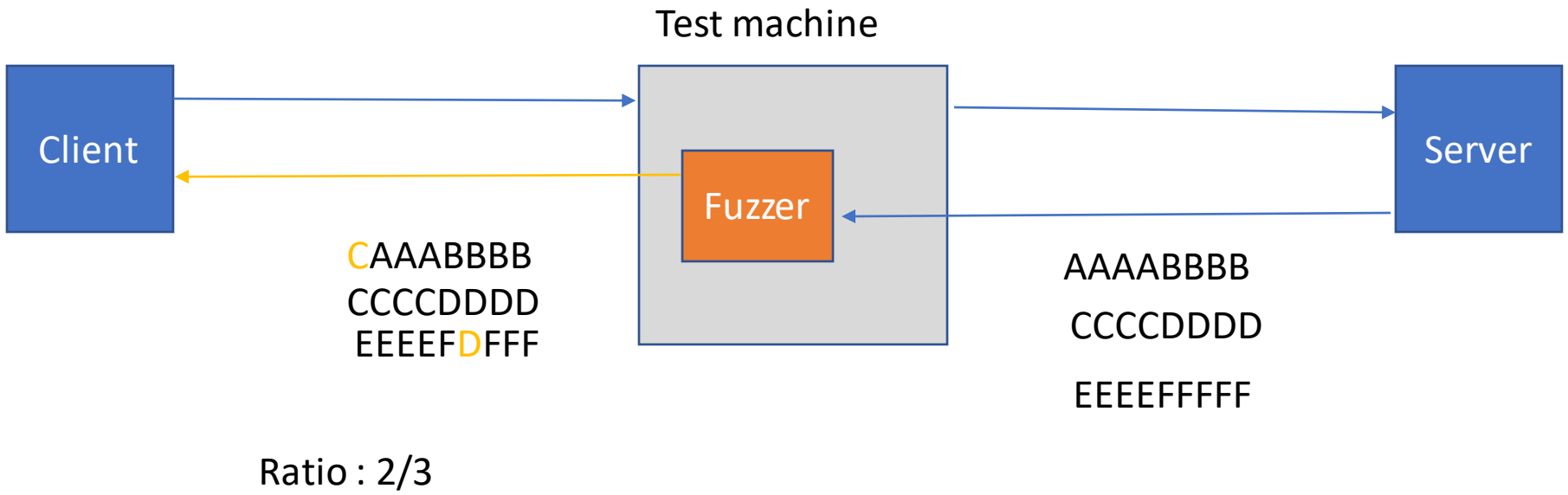
Description

- Fuzz the client
- Fuzz a small part of the received request
- Some parameters in the frame were recalculated like crc



Description

- Only a ratio of requests are fuzzed to be able to reach and test the other states



Solution 2 : Pulsar

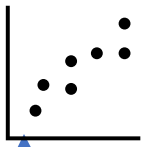
- Software developed by the University of Gottingen
- This software is for fuzzing proprietary protocol

How it works ?

- Find the state machine of the system from networking traces
- Deduce the rules for the transition state and messages templates
- Fuzz a program according to the deduced model

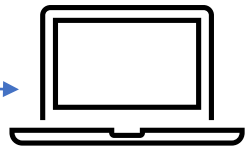
Pulsar process

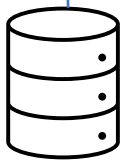
Model : templates, messages rules, states



Fuzzer

Messages generated by respecting the model





Dataset : Networking traces

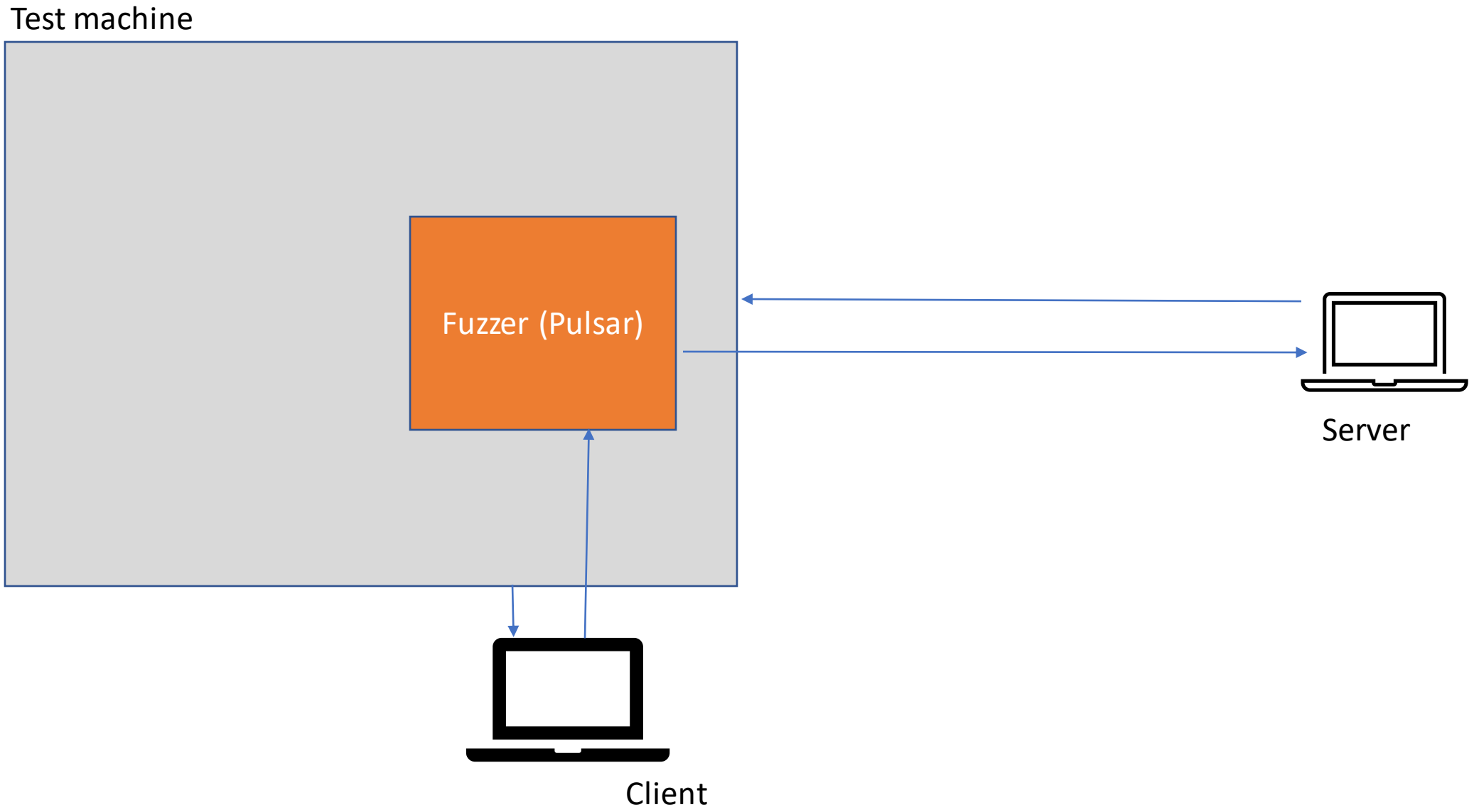
Results

- First approach
  - Memory management
  - Some bugs implementation
- Pulsar approach
  - It does not work for complex protocols

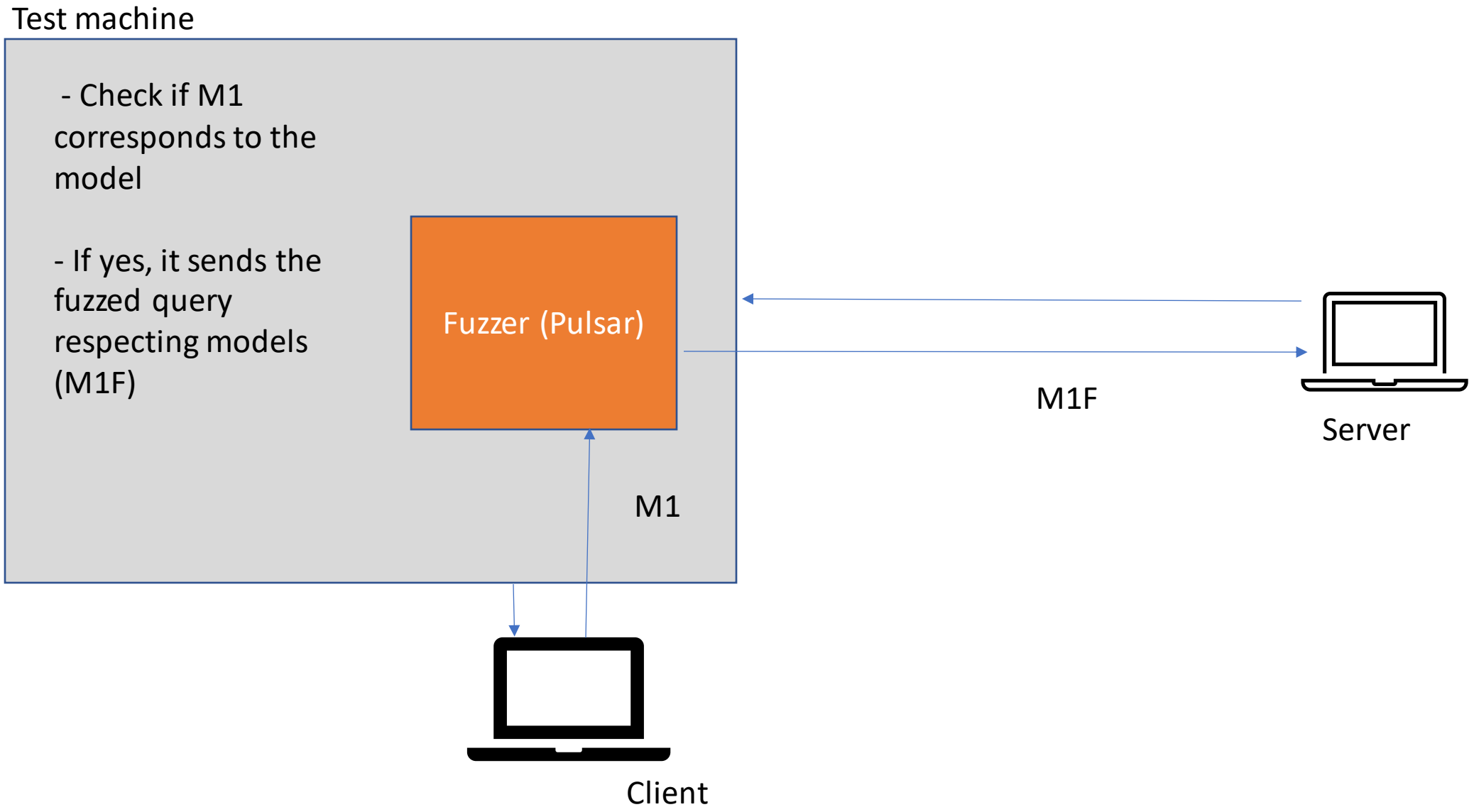
Comparison between the two approaches

	MITM	Pulsar
Rules, templates	-	+
Correct inputs	+	-
Efficient fuzzing	-	+
Time to configure	-	+

Pulsar in Mitm



Pulsar in Mitm



Expected results

- An efficient fuzzer compatible with the maximum of the stateful program and then the less loss of time to configure it
- Able to find new bugs or vulnerabilities in systems
- Possibilities to improve the initial model during the execution

Future work

- Manage different types of packets (packets control, telemetry...).
  - For the moment packets contol falsifies the model
- Work on differents IP layer
- Use information from code sources when available
- Improvement of the quality of the models
  - better recognition of rules, recognition of crc...
- Work with encrypted traffic

Questions ?

Thank you for your attention