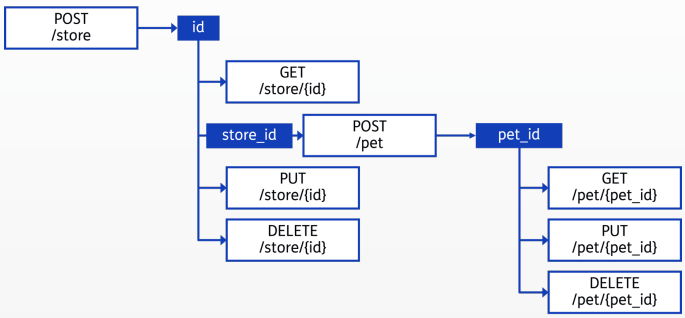
Do REST API Fuzzers Need to Be Different Than Traditional Fuzzers?

Stefan van den Berg and Cristian Daniele – TNO and Radboud University

Stateless or Stateful?

REST API systems are built to be stateless.

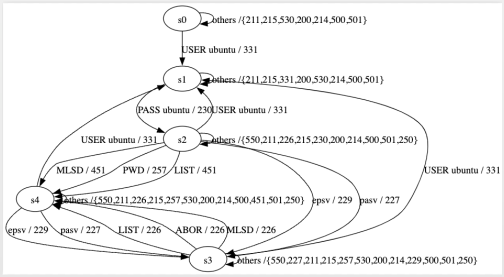
However, API systems still maintain a degree of statefulness through the persistence of resources and through relations between parameters.



REST API endpoints

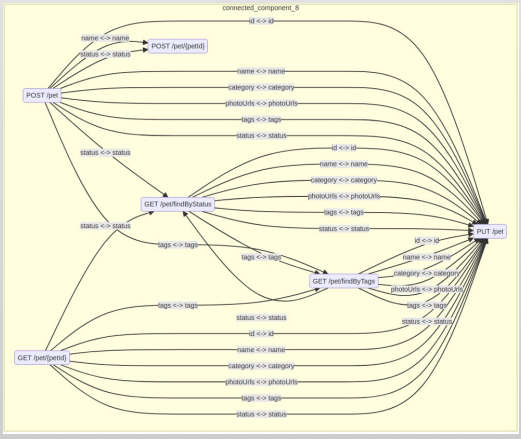
Statefulness in REST APIs

We usually represent the statefulness of a system through a state model that describes the system's knowledge at a given point in time.



Stateful System

However, when representing the stateful nature of a REST API system, we need to represent the existence of resources and their relationships.



Stateful REST API System

Different fuzzers and metrics

This difference requires using specialized fuzzers and tailored metrics to properly fuzz REST API systems.

SUT	Fuzzer used	Endpoint coverage	Response coverage	Deviation from the specification	Requests sent	Server errors
dropx	EvoMaster	7/7	Na	6	4651	0
dropx	RESTler	7/7	Na	Na	2705	0
dropx	WuppieFuzz	7/7	7/80	0	6053	0
ebi	EvoMaster	0/13	Na	13	14491	0
ebi	RESTler	7/13	Na	Na	7306	0
ebi	WuppieFuzz	0/13	8/69	17	612	4
ibanapi	EvoMaster	0/3	Na	0	9326	0
ibanapi	RESTler	0/3	Na	Na	246	0
ibanapi	WuppieFuzz	3/3	3/15	0	15981	0
ideaconsult	EvoMaster	8/13	Na	12	4248	13
ideaconsult	RESTler	0/13	Na	Na	8058	1
ideaconsult	WuppieFuzz	9/13	12/74	32	11722	12
namsor	EvoMaster	5/84	Na	80	7041	1
namsor	RESTler	2/84	Na	Na	7429	0
namsor	WuppieFuzz	4/84	115/339	63	12132	1
nexmo	EvoMaster	5/6	Na	6	8042	0
nexmo	RESTler	6/6	Na	Na	228	0
nexmo	WuppieFuzz	6/6	6/21	11	16504	0
portfoliooptimizer	EvoMaster	1/83	Na	80	9736	0
portfoliooptimizer	RESTler	0/83	Na	Na	240	0
portfoliooptimizer	WuppieFuzz	11/83	11/257	164	19173	0
transavia	EvoMaster	0/5	Na	5	14798	0
transavia	RESTler	0/5	Na	Na	103	0
transavia	WuppieFuzz	0/5	5/22	7	18825	0
waterlinked	EvoMaster	0/38	Na	27	5955	0
waterlinked	RESTler	23/38	Na	Na	9655	0
waterlinked	WuppieFuzz	21/38	34/91	8	9359	1

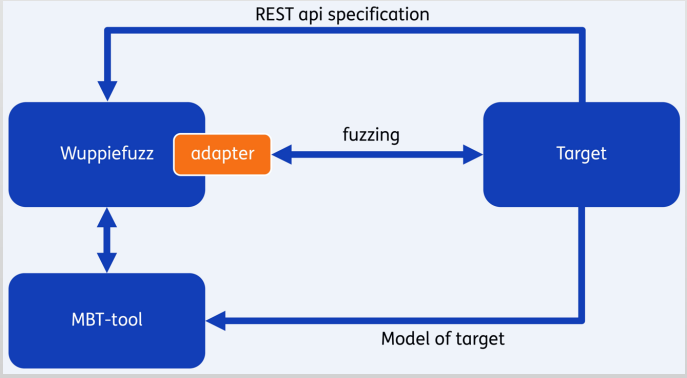
Fuzzing with MBT

With stateful fuzzing, we notice that we are building up the state for each input (multiple requests are made to flow through the graph).

Model-based testing (MBT) tracks the state through a comprehensive test. The disadvantage is that MBT only tests within the specification. Fuzzing is testing how the system responds to unexpected inputs.

Combining both techniques can test the application more completely than each method can do on its own.

We have three ways to combine the techniques: either the MBT-tool is in the lead, WuppieFuzz is in the lead, or they alternate who is in the lead.



MBT and Fuzzing loop