From One Thousand Pages of Specification to Unveiling Hidden Bugs: Large Language Model Assisted Fuzzing of Matter IoT Devices

Xiaoyue Ma, Lannan Luo, Qiang Zeng

George Mason University, USA

USENIX Security 2024 • Philadelphia, USA

GEORGE MASON UNIVERSITY

Background

- **Matter** is an open, uniform IoT standard
  - Backed by **200+** companies, such as Amazon, Google, Apple
  - A Google Hub can control an Amazon plug, and vice versa

- **Our work**: To discover bugs and vulnerabilities in Matter devices

Forbes

Matter And Thread
Win The IoT Connectivity Wars

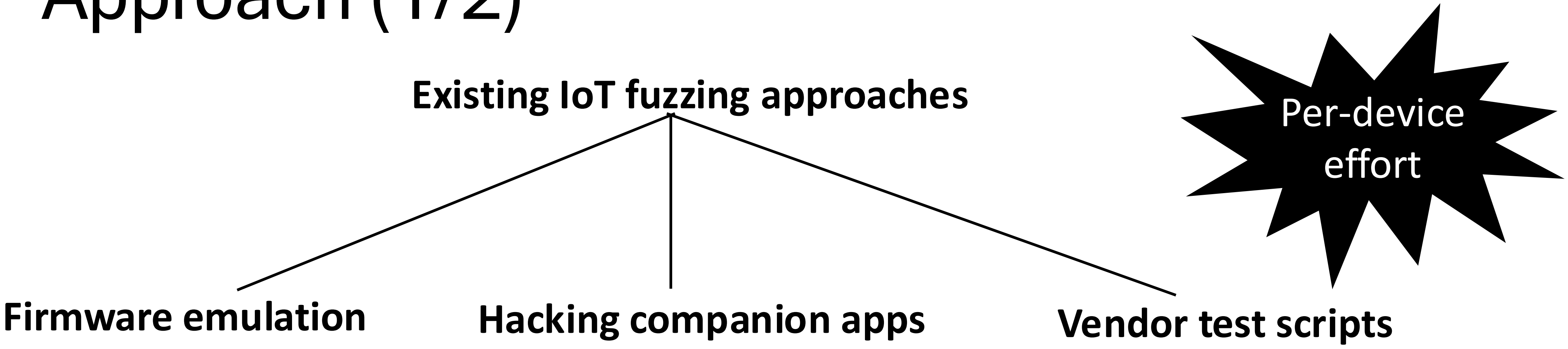
TIME

The Best Inventions
of 2021

THE VERGE

Matter’s plan to save
the smart home

Approach (1/2)



Existing IoT fuzzing approaches

Firmware emulation

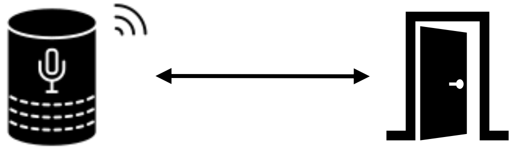
Hacking companion apps

Vendor test scripts

Per-device effort

- **Our insight:** A Matter device can be controlled by a Matter controller

- **Our approach:** Sending test messages from a controller, called **controller-based fuzzing**
  - Inspired by HubFuzzer [Ma, et al., MobiSys’23]
  - No emulation, no app hacking, no need to collect test scripts



Approach (2/2)

- **Our observation:** The Matter specification contains rich information
  - Valid parameter values
  - Command effect
  - Expected response

- **Direction:** It is promising to make use of the information in the specification for test input generation.

Challenges

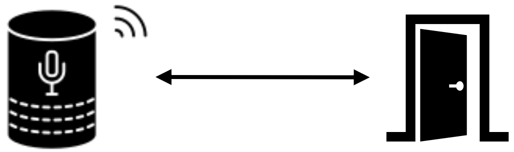
- Challenge 1: Command coverage
- Challenge 2: Sheer volume of specification
- Challenge 3: Stateful bugs
- Challenge 4: Non-crash bugs

Challenge 1: Command coverage

- A fuzzer should test all the commands of a device

- **Observation**: When a controller adds a device, the device declares all supported commands

- Build a fuzzer within a controller and extract the supported commands from pairing messages

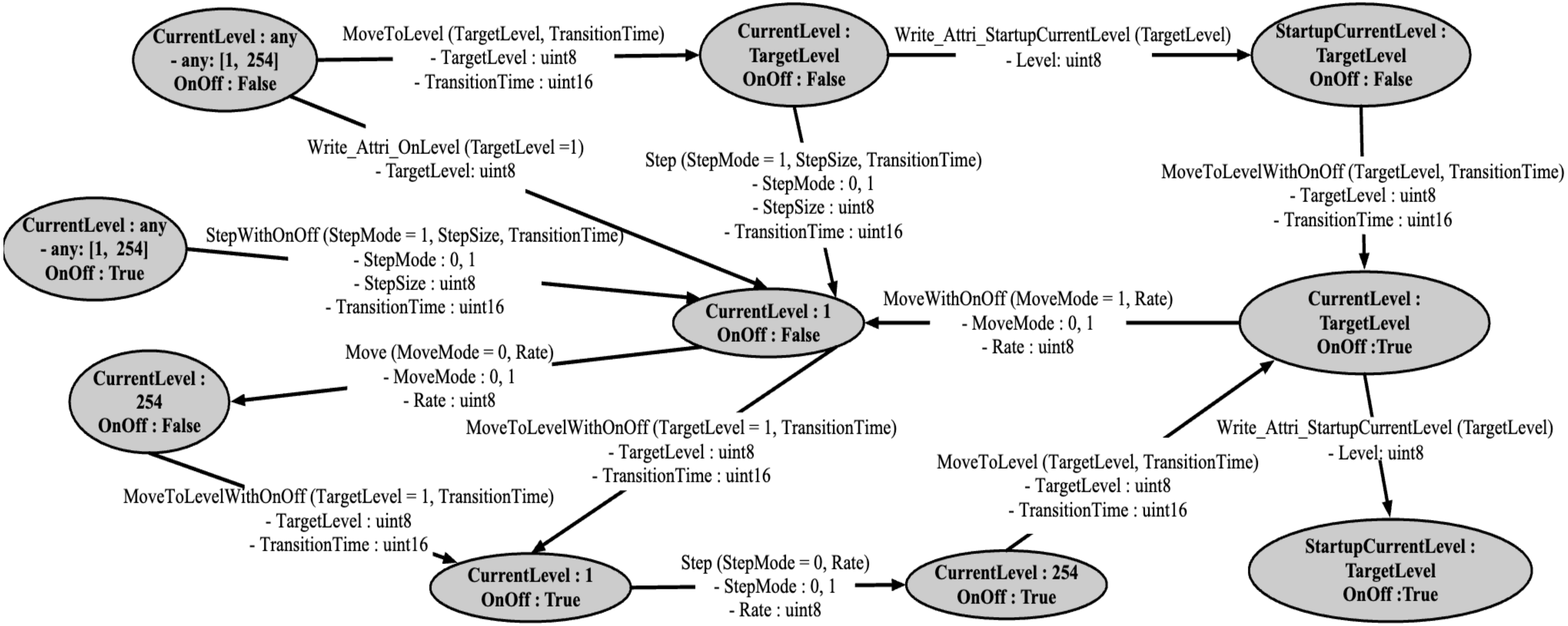


Challenge 2: Sheer volume of specification
Challenge 3: Stateful bugs

- Matter specification contains 1258 pages

- Commands only make sense when the device is at a specific state
  - Represented in finite-state-machines (FSMs)

- Large Language Model (LLM) Assisted Fuzzing



CurrentLevel : any
- any: [1, 254]
OnOff : False

MoveToLevel (TargetLevel, TransitionTime)
- TargetLevel : uint8
- TransitionTime : uint16

CurrentLevel :
TargetLevel
OnOff : False

Write_Attri_StartupCurrentLevel (TargetLevel)
- Level: uint8

StartupCurrentLevel :
TargetLevel
OnOff : False

Write_Attri_OnLevel (TargetLevel =1)
- TargetLevel: uint8

Step (StepMode = 1, StepSize, TransitionTime)
- StepMode : 0, 1
- StepSize : uint8
- TransitionTime : uint16

MoveToLevelWithOnOff (TargetLevel, TransitionTime)
- TargetLevel : uint8
- TransitionTime : uint16

CurrentLevel : any
- any: [1, 254]
OnOff : True

StepWithOnOff (StepMode = 1, StepSize, TransitionTime)
- StepMode : 0, 1
- StepSize : uint8
- TransitionTime : uint16

CurrentLevel : 1
OnOff : False

MoveWithOnOff (MoveMode = 1, Rate)
- MoveMode : 0, 1
- Rate : uint8

CurrentLevel :
TargetLevel
OnOff :True

Move (MoveMode = 0, Rate)
- MoveMode : 0, 1
- Rate : uint8

CurrentLevel :
254
OnOff : False

MoveToLevelWithOnOff (TargetLevel = 1, TransitionTime)
- TargetLevel : uint8
- TransitionTime : uint16

Write_Attri_StartupCurrentLevel (TargetLevel)
- Level: uint8

MoveToLevel (TargetLevel, TransitionTime)
- TargetLevel : uint8
- TransitionTime : uint16

MoveToLevelWithOnOff (TargetLevel = 1, TransitionTime)
- TargetLevel : uint8
- TransitionTime : uint16

CurrentLevel : 1
OnOff : True

Step (StepMode = 0, Rate)
- StepMode : 0, 1
- Rate : uint8

CurrentLevel : 254
OnOff : True

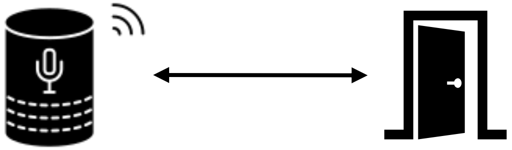
StartupCurrentLevel :
TargetLevel
OnOff :True

Example: FSM for the LevelControl cluster

Challenge 4: Non-crash bugs

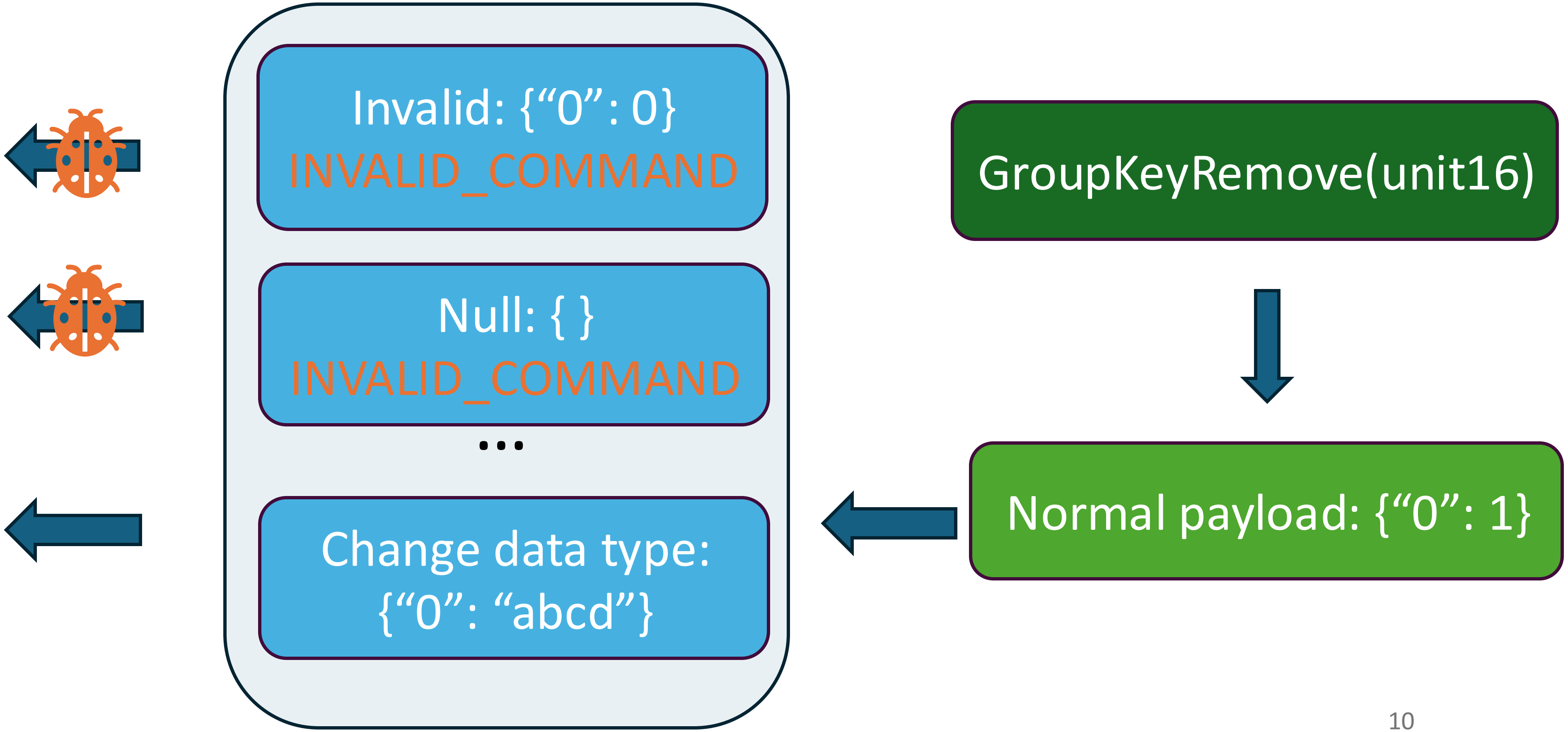
- It is feasible to collect the program execution information inside a device
  - Brach coverage
  - Path condictiones
  - Function return values

- Leverage command semantics
  - Querying attributes modified by command execution



Fuzzing policies

- FSM-guided test generation



Invalid: {“0”: 0}
INVALID_COMMAND

Null: { }
INVALID_COMMAND

…

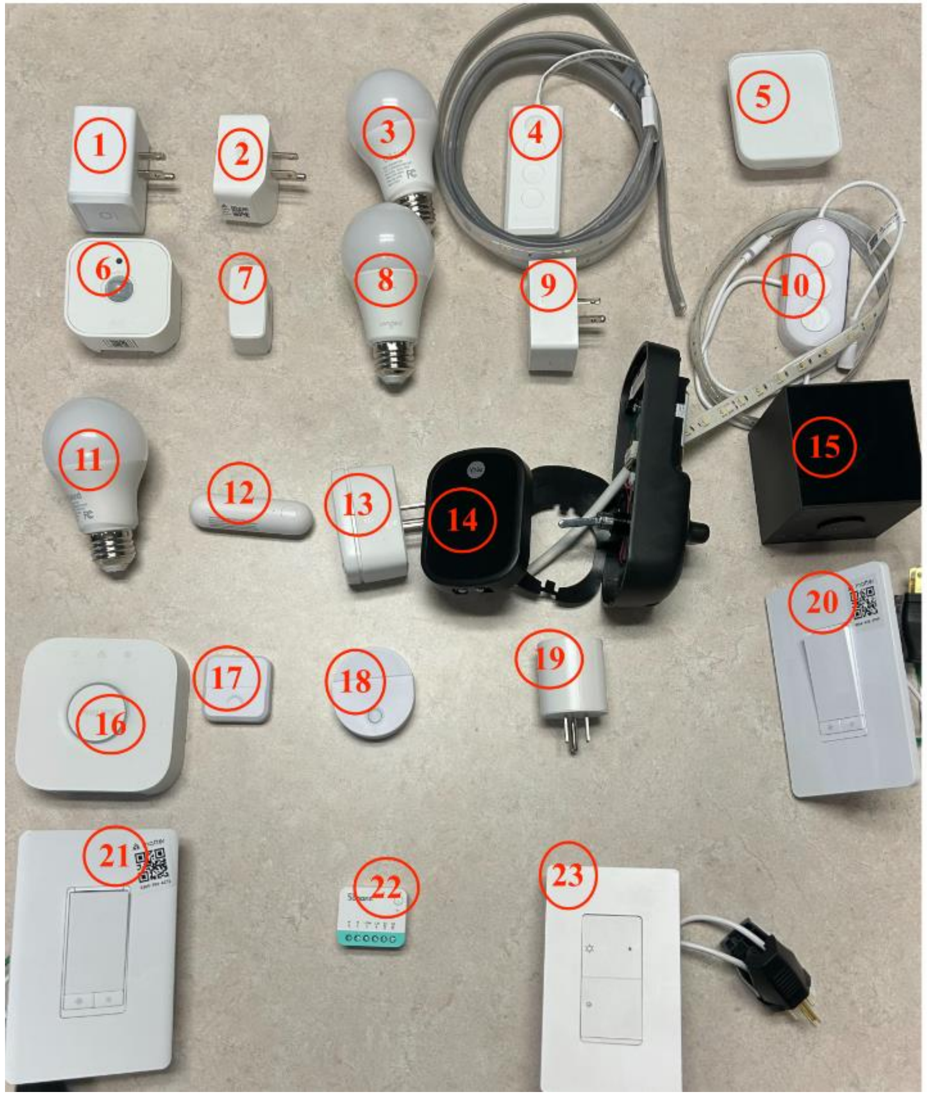
Change data type:
{“0”: “abcd”}

GroupKeyRemove(unit16)

Normal payload: {“0”: 1}

Evaluation

- 23 Matter devices
  - Smart switches,
  - Lighting,
  - Locks,
  - Sensers,
  - Hubs
  - …



147 new bugs

3 CVEs

0 can be found using SNIPUZZ (prior state of the art)

Example - Non-crashed bugs

- State sensitive bug
  - Govee Lighting device wrongly accepted and execute
  - Initial state: Highest hue level
  - Action: MoveHue up with 0 rate, meaning no change
  - Expected Behavior: Should reject and respond INVALID_COMMAND
  - Actual Behavior: Device accepted and state was changed

Summary

- The first Matter fuzzer: **mGPTFuzz**
- Controller-based fuzzing architecture
- LLM-assisted fuzzing: stateful, non-crash bugs
- 147 new bugs, 61 zero-day, 3 CVEs

Q&A

Xiaoyue Ma (xma9@gmu.edu)

GEORGE MASON UNIVERSITY