Using Cyber Digital Twins for Automated Automotive Cybersecurity Testing

SRCNAS/STRIVE WS @ IEEE EURO S&P‘ 21

September 6, 2021

Stefan Marksteiner, Slava Bronfman, Markus Wolf, Eddie Lazebnik

The Need for Industrialized Automotive Cybersecurity Testing

- UNECE
  - Regulation R.155
  - Mandates cybersecurity and cybersecurity management
  - Requires testing of measures
  - Adopted in EU, Japan and Korea
  - Effective in EU for new types 2022 and for all new vehicles 2024
- ISO/SAE 21434
  - Cyber security management system for automotive systems
  - Risk-based approach
  - Also demands testing, however, does not specify details
  - To be supplemented for testing by ISO PWI 8477 (V&V) and ISO/SAE PWI 8475 (CAL &TAF)

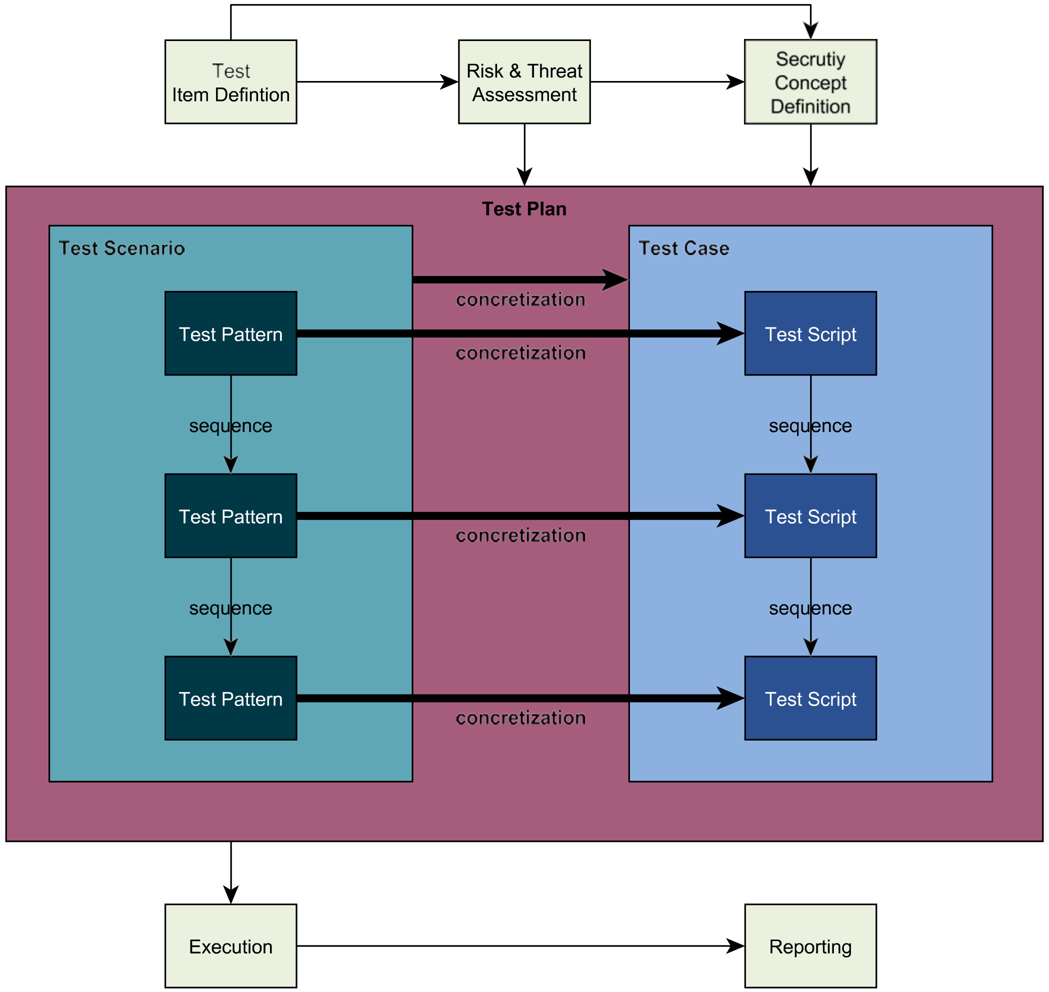
=> Need for automated testing

Why Black Box Testing?

- Providing an attacker's view
- Long supply chain – source might not be available
- Unwillingness (or inability) to disclose internals

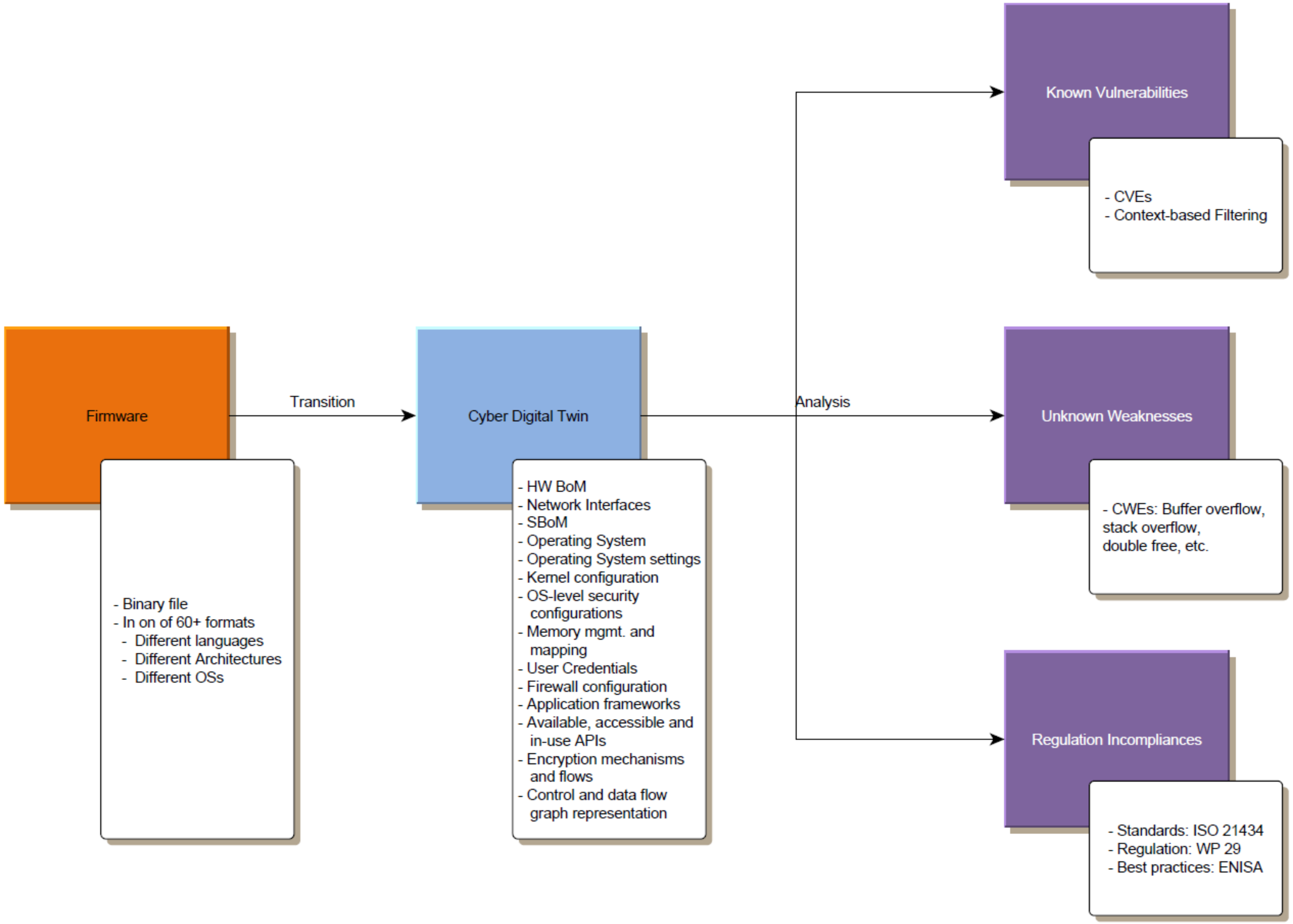
Static Approach (Previous Work)

- Generalize Existing Attacks
- Formulate Attack Scenarios in DSL (ALIA[14])
- =>SUT-Agnostic attack description
- Test Case Generation => augmenting attacks with SUT info



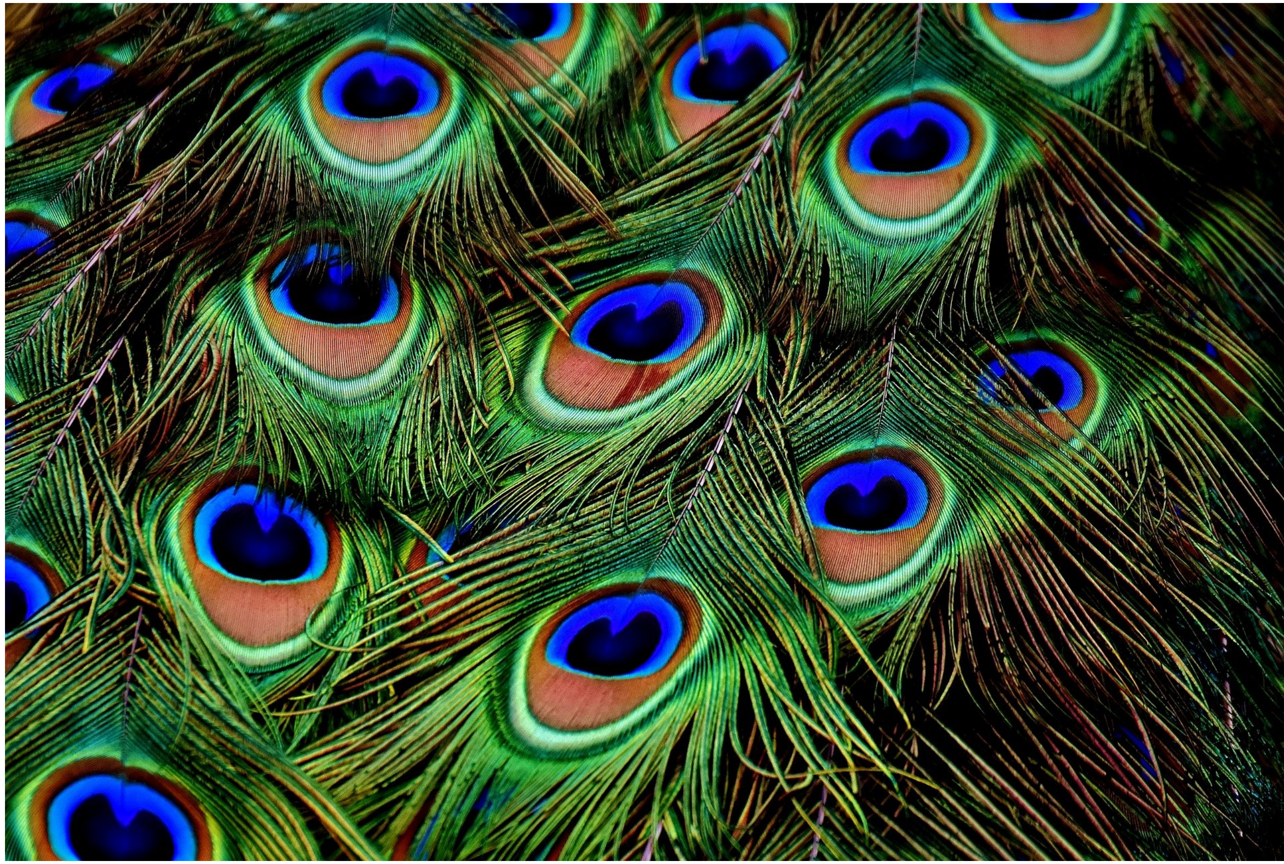
Problem: approach static - lots of a priori information needed!

Cyber Digital Twin (Previous Work[11])

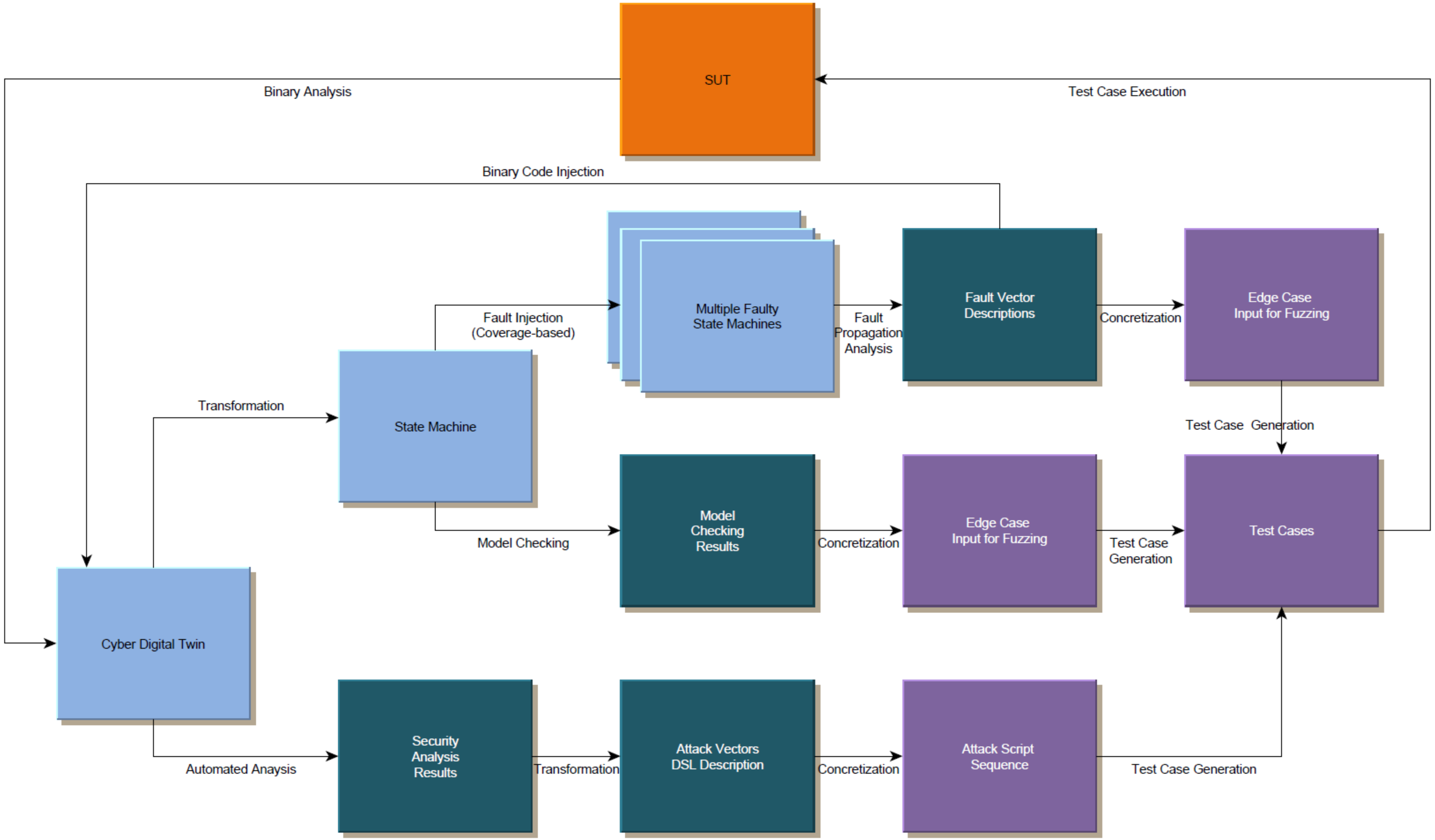


Cyber Digital Twin – Pattern Matching

- Translate binary into own machine code format
- Compare patterns of known software with parts of the binary => software BOM
- Compare patterns of known vulnerabilities (CVEs) and general flaws with parts of the binary => security analysis results



Test Case Generation



State Machine-Based Testing

- Fault injection
  - Inject Faults into the State Machine
  - Use the ones producing interesting results as test cases
- Model Checking
  - Transform model into provable form
  - Use violations as test case inputs

Binary Analysis -> Attack DSL Scripts

- Generate DSL scripts out of findings
- Use pre-prepared building blocks
  - CVEs
  - Code pieces for buffer overflows, etc.

```
ID <2> BT_Connect=TRUE
ID <4> MEASUREMENT(SPD, PRETEST)= 0
</PRECONDITIONS>
<ATTACK>
ID <1> Traget Vulns:=ACTION SCAN_IF_VULN  (Bluetooth, MA
ID <2> Shell:=ACTION EXPLOIT_BT (Target_Vulns, GetShell)
ID <3> RootShell:= ACTION OPEN_ADB_SHELL(ADB_KEY, S
ID <4> Result:=ACTION RUN_ATTACK_TOOL(RootShell, Can
</ATTACK>

<POSTCONDITIONS>
ID <2> BT_Connect=FALSE
ID <3> RootShell!=NULL
ID <4> Result=Success
ID <4> MEASUREMENT(SPD, INTEST)=200
ID <4> MEASUREMENT(SPD, POSTTEST)=0
```

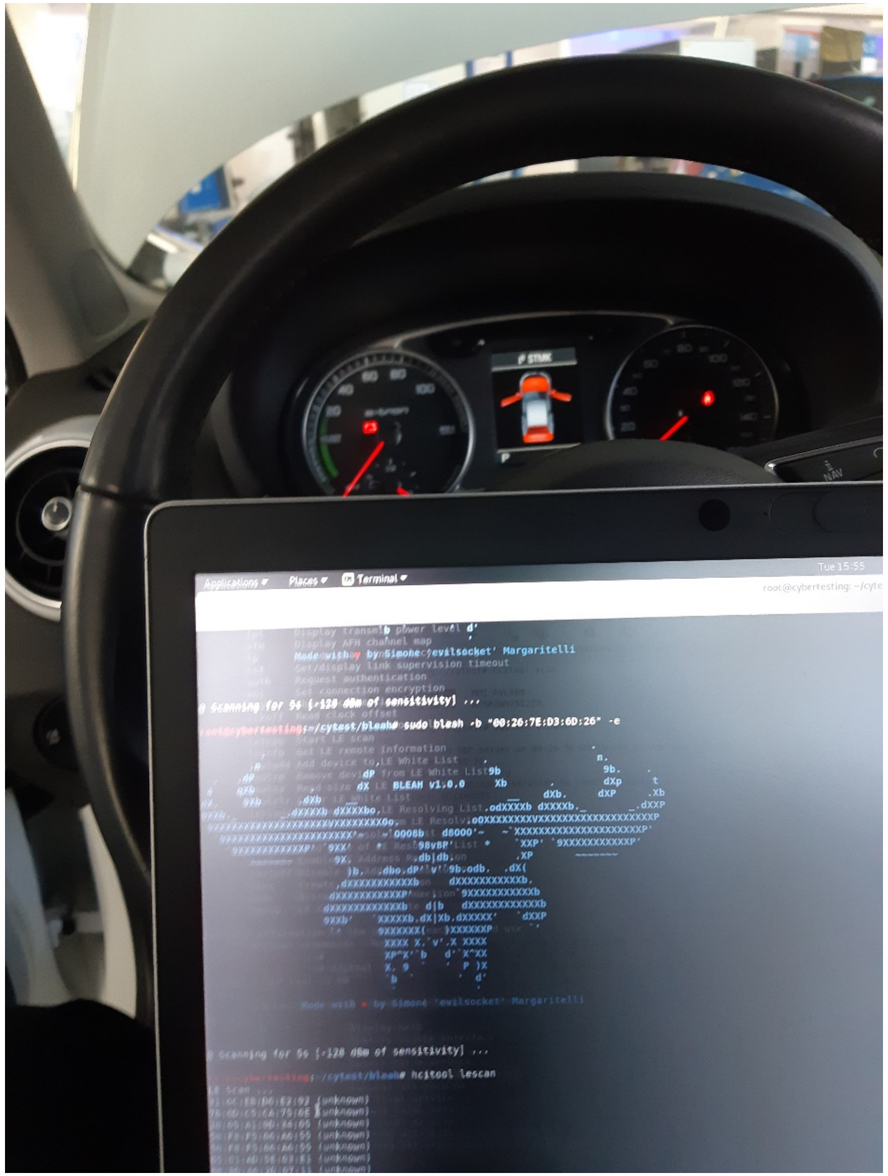
Test Execution

- Test case generation produces a JSON output that can be interpreted by an execution engine
- Principally an environment description + shell commands

```
from pyusbtin.usbtin import USBtin
from pyusbtin.canmessage import CANMessage
from time import sleep

def log_data(msg):
        print(msg)

usbtin=USBtin()
usbtin.connect("/dev/ttyACM0")
usbtin.add_message_listener(log_data)
usbtin.open_can_channel(500000,USBtin.ACTIVE)

#test_msg = CANMessage([illegible])
test_msg = CANMessage([illegible])

while(True):
        usbtin.send(test_msg)
        sleep(0.1)

#pysh = "/data/user/0/com.hipipal.qpy3/files/bin/qpython3-android5.sh"
#import subprocess
#subprocess.call([pysh,"/sdcard/usbtintest.py"])
```

Conclusion

- Concept for model-based cybersecurity testing of automotive systems
- Uses existing building blocks
- Combines
  - Dynamic model generation
  - Dynamic security analysis
  - Automated test case generation
  - Automated test execution



Thank you for your attention!

Thanks!

Stefan Marksteiner [1], Slava Bronfman [2], Markus Wolf [1], Eddie Lazebnik [2]

[1] AVL List Gmbh, {stefan.marksteiner | markus.wolf}@avl.com
[2] Cybellum Ltd., {slava | eddie}@cybellum.com

This research has received funding from the program "ICT of the Future" of the Austrian Research Promotion Agency (FFG) and the Austrian Ministry for Transport, Innovation and Technology under grant agreement No. 867558 (project TRUSTED) and within the the ECSEL Joint Undertaking (JU) under grant agreement No. 876038 (project InSecTT). The JU receives support from the European Union's Horizon 2020 research and innovation programme and Austria, Sweden, Spain, Italy, France, Portugal, Ireland, Finland, Slovenia, Poland, Netherlands, Turkey. The document reflects the author's view only and the Commission is not responsible for any use that may be made of the information it contains.