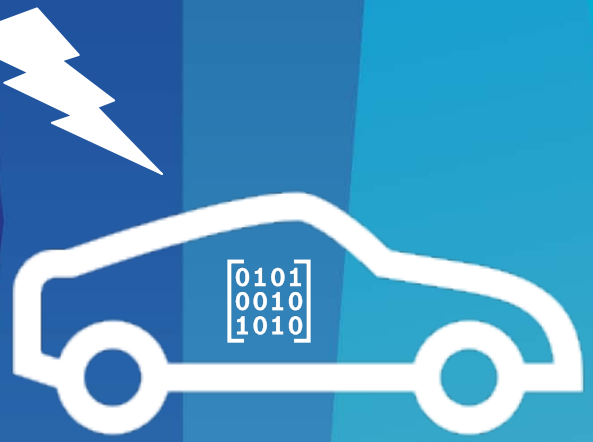
RETHINKING FUZZING FOR AUTOMOTIVE SOFTWARE



FuzzCon Europe Automotive Edition

2020-12-02

RAKSHITH AMARNATH
CORPORATE RESEARCH
ROBERT BOSCH GMBH

BOSCH

Rethinking Fuzzing for Automotive Software

What is this talk about?

- Why fuzz automotive software?
- How is automotive software different?
- Why rethink fuzzing for automotive software?
- Thoughts about rethinking
- Conclusions and takeaways

BOSCH

Rethinking Fuzzing for Automotive Software

- Why fuzz automotive software?
- How is automotive software different?
- Why rethink fuzzing for automotive software?
- Thoughts about rethinking
- Conclusions and takeaways

BOSCH

Rethinking Fuzzing for Automotive Software

Why fuzz automotive software?

- Take proactive measures
  - Continuous fuzzing of automotive software (especially for SAE automation level 3 and above)
- Technically effective
  - Fuzzing also makes technical sense: to combat large number of false positives of static analysis, finding subtle bugs that go beyond simple coding guideline checks etc.

Fuzzing makes sense and just doing it!

- Compliance to internal/future norms
  - Internal guidelines and norms at companies
  - Future Norms e.g. ISO/SAE 21434
- Requirements from customers
  - Automotive manufacturers emphasize that fuzz testing be done

Fuzzing gets imposed and having to do it!

Intrinsic and extrinsic motivations for fuzzing automotive software

Rethinking Fuzzing for Automotive Software

- Why fuzz automotive software?
- **How is automotive software different?**
- Why rethink fuzzing for automotive software?
- Thoughts about rethinking
- Conclusions and takeaways

BOSCH

Rethinking Fuzzing for Automotive Software

Difference #1: Platform dependence

Peculiarities

- Stronger platform dependence due to resource constraints and optimizations
- Well established Hardware-in-the-Loop (HiL) testing
- Customary use of certified compilers

Impediments

- Stubbing necessary to eliminate platform dependencies
- Execution on Linux is not always possible out of the box
- Complex build chains (e.g. code generation) require special treatment
- Non trivial interfaces make creation of fuzz targets challenging

High entry barrier for fuzzing automotive applications

BOSCH

Rethinking Fuzzing for Automotive Software

Difference #2: Statefulness

Peculiarity

- Automotive software is predominantly stateful -> a set of state transitions have to occur before something interesting happens

Impediments

- Traditional fuzzers target software with single input whereas stateful programs require sequence of inputs
- Adequate consideration of evolving program state during input generation is missing
- Resetting (cleaning up resources for) the software that runs on its own (in infinite loop) is challenging

Novel fuzzing techniques are required to fuzz stateful embedded software

Rethinking Fuzzing for Automotive Software

Difference #3: Integration of multi-party software including binaries

Peculiarity

- Automotive software often involves multiparty software (binaries) from different vendors

Impediments

- Unavailability of source code makes it hard to achieve the performance equivalent of state of the art modern fuzzers
- Executing native automotive binaries virtually is not always possible using prominent instruction set emulators like QEMU, UniCorn, Gem5

Scalable approaches needed for fuzzing automotive applications

BOSCH

Rethinking Fuzzing for Automotive Software

- Why fuzz automotive software?
- How is automotive software different?
- **Why rethink fuzzing for automotive software?**
- Thoughts about rethinking
- Conclusions and takeaways

BOSCH

Rethinking Fuzzing for Automotive Software

Now why rethink after all?

Where fuzzing excels today

- Analysis of software with access to source-code
- Optimizations to fuzzing techniques (efficiency)
- Protocol fuzzing tools and vibrant community of bleeding edge open source tools for software fuzzing

Where fuzzing needs rethinking!

- **Automotive projects also integrate 3rd party software binaries w/o access to source-code**
- **Enabling automotive projects to start fuzzing is more crucial than optimizing fuzzing**
- **Open source tools impose restrictions that only handful of automotive projects satisfy**

Automotive software benefits from effective techniques first and efficient techniques later

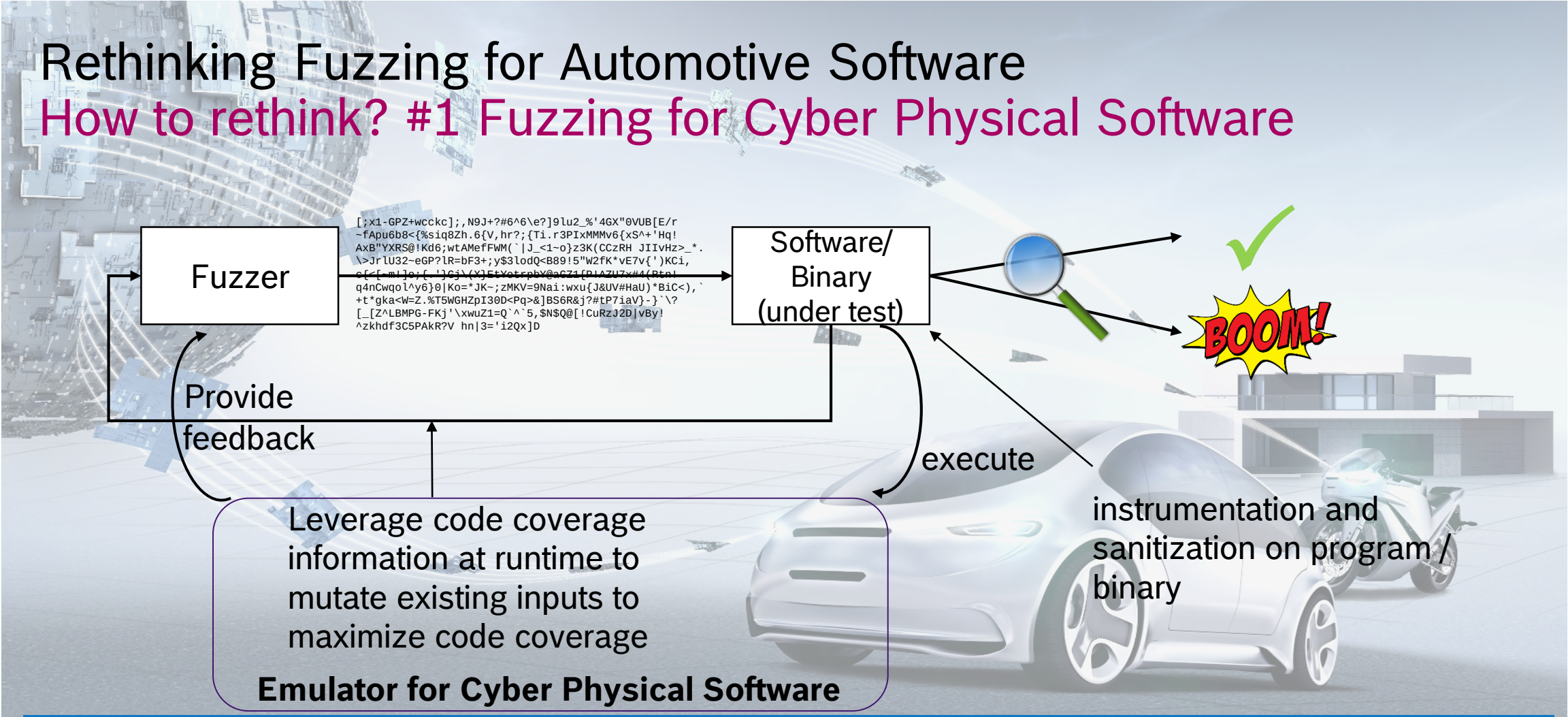
Rethinking Fuzzing for Automotive Software

- Why fuzz automotive software?
- How is automotive software different?
- Why rethink fuzzing for automotive software?
- **Thoughts about rethinking**
- Conclusions and takeaways

BOSCH

Rethinking Fuzzing for Automotive Software

How to rethink? #1 Fuzzing for Cyber Physical Software



Systematic approaches required to enable fuzz testing for Cyber Physical Software

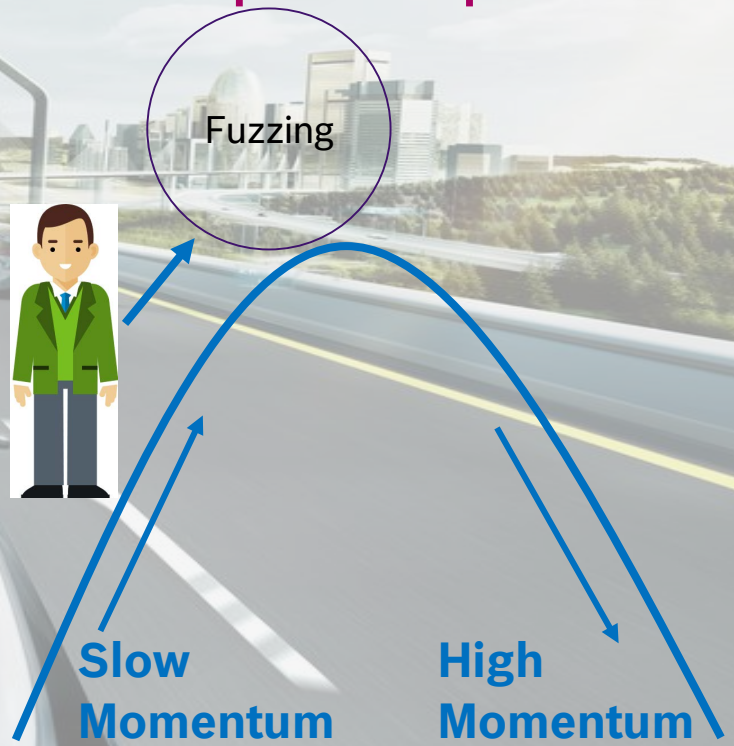
Rethinking Fuzzing for Automotive Software

How to rethink? #2 Anchor fuzzing into SW development process

You could excel in building a fuzzing platform

BUT ...

Anchoring Fuzzing into our SW development activities is crucial for sustaining the momentum



Make fuzzing a part of CI/CD and anchor it to become an integral part of SW development

Rethinking Fuzzing for Automotive Software

- Why fuzz automotive software?
- How is automotive software different?
- Why rethink fuzzing for automotive software?
- Thoughts about rethinking
- **Conclusions and takeaways**

BOSCH

Rethinking Fuzzing for Automotive Software

Wrap Up

Why fuzz automotive software?

- Intrinsic motives
- Extrinsic motives

How is automotive software different?

- High entry barrier for fuzzing
- Stateful software
- Native binaries complicating fuzzing

Why rethink?

- Effectiveness over efficiency

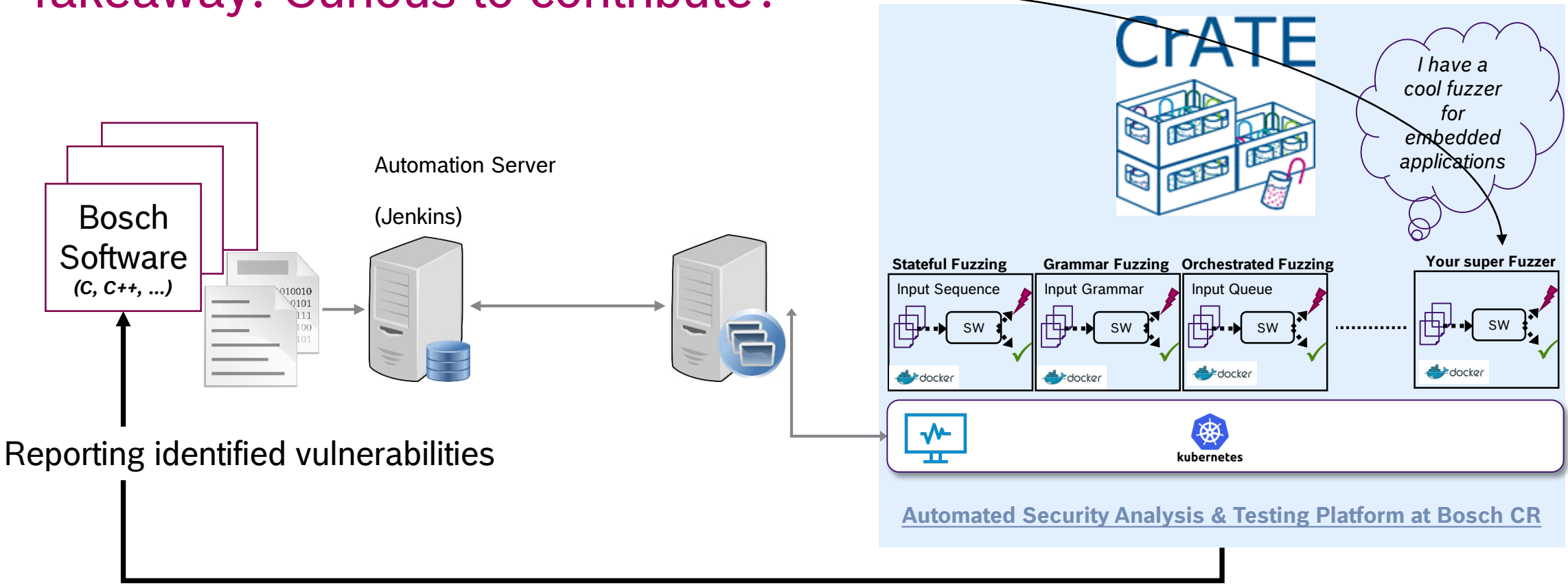
How to rethink?

- Fuzzing Cyber Physical Software
- Anchor Fuzzing into SW development

Cyber Physical Software – Fuzzing's next frontier!

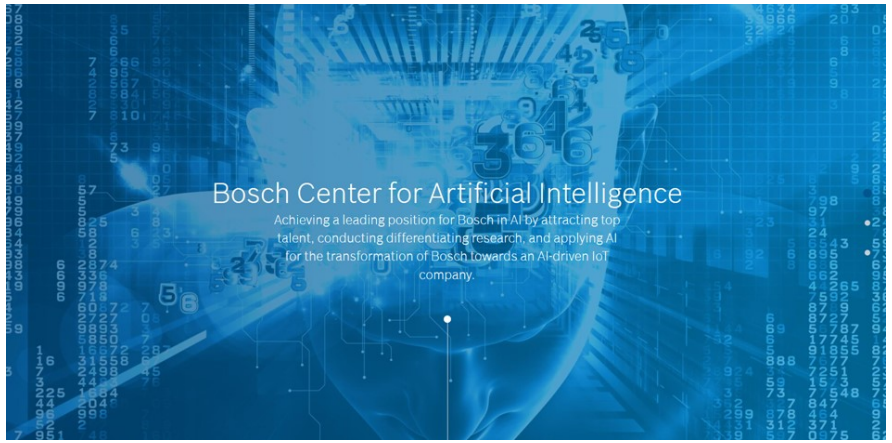
Rethinking Fuzzing for Automotive Software

Takeaway: Curious to contribute?



Extensible platform enables contributions from the fuzzing community

Rethinking Fuzzing for Automotive Software
Time for Questions



Feel free to reach out to me:

LinkedIn or

firstname[dot]lastname[at]bosch[dot]com