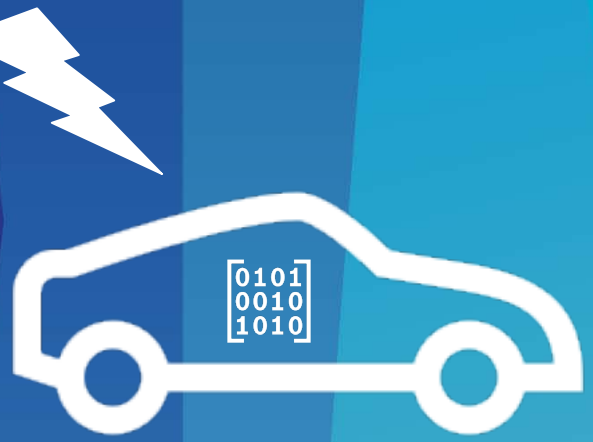
WHAT'S DIFFERENT ABOUT FUZZING AUTOMOTIVE SOFTWARE?



FuzzCon Europe 2020

2020-09-08

RAKSHITH AMARNATH
CORPORATE RESEARCH
ROBERT BOSCH GMBH

BOSCH

What's different about fuzzing automotive software?
What is this talk about?

- Why fuzz automotive software?
- What's the difference?
- How to cope with the differences?
- Conclusions and Takeaways

BOSCH

What's different about fuzzing automotive software?

- Why fuzz automotive software?
- What’s the difference?
- How to cope with the differences?
- Conclusions and Takeaways

BOSCH

What's different about fuzzing 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

What's different about fuzzing automotive software?

- Why fuzz automotive software?
- **What’s the difference?**
- How to cope with the differences?
- Conclusions and Takeaways

BOSCH

What's different about fuzzing 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

Challenge: High entry barrier for fuzzing automotive applications

BOSCH

What's different about fuzzing 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

Challenge: Novel fuzzing techniques are required to fuzz stateful embedded software

What's different about fuzzing automotive software?

Difference #3: Integration of 3rd Party 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

Challenge: Scalable approaches needed for fuzzing native automotive binaries

BOSCH

What's different about fuzzing automotive software?

Why fuzz automotive software?

What’s the difference?

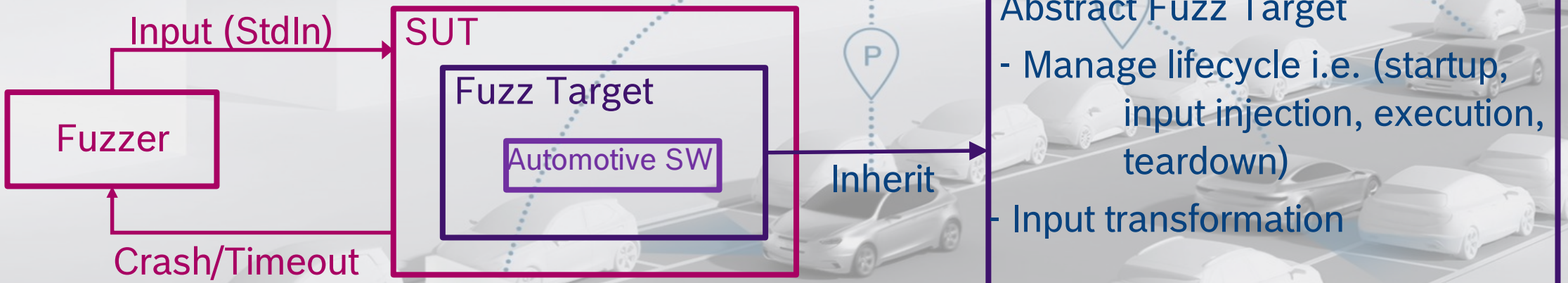
How to cope with the differences?

Conclusions and Takeaways

BOSCH

What's different about fuzzing automotive software?

Solution: Lower the barrier



Automotive Software:
- Non-trivial input format required
- Special setup phase required

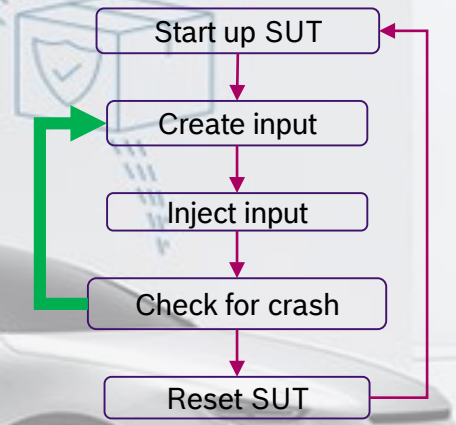
Fuzz Target:
- Directly affects quality of fuzzing
- Most effort in connection to fuzzer + writing fuzz target

Separate security specific activities from projects to lower the overhead for security testing

What's different about fuzzing automotive software?

Solution: Provide support for stateful fuzzing (ongoing research)

- **Motivation:**
  - The more (automation in finding internal states), the better
- **What will stateful fuzzing help with?**
  - Allows systematic testing for sequences of inputs leading to security vulnerability
  - (Ideally) no knowledge or specification of “critical” internal states necessary
- **Open research avenues?**
  - State identification
  - Building sequences
  - Picking a state to fuzz



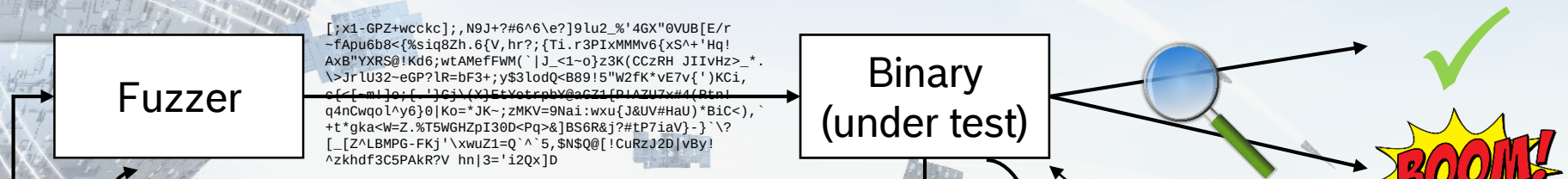
Recent research that addresses stateful aspects for network protocols in Servers:

AFLNET: A Greybox Fuzzer for Network Protocols

Fuzzing stateful programs has potential for further research in the context of embedded software

What's different about fuzzing automotive software?

Solution: Fuzz native binaries (future work)



Recent research addressing smart fuzzing on RISC-V based virtual prototypes:

Fuzzing Embedded Binaries leveraging SystemC-based Virtual Prototypes

Going ahead, systematic and scalable approaches for fuzzing automotive binaries are needed

What's different about fuzzing automotive software?

Why fuzz automotive software?

What’s the difference?

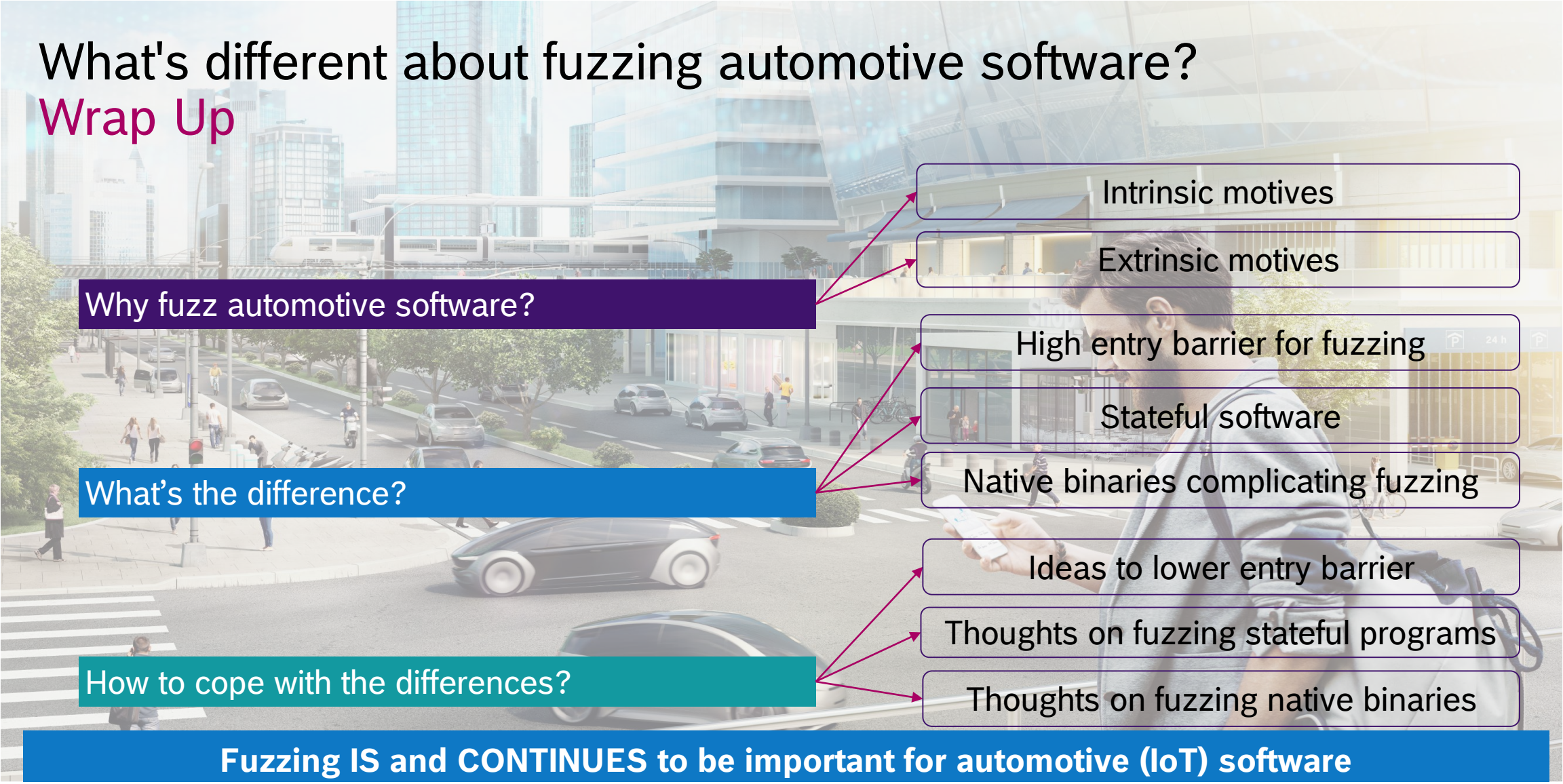
How to cope with the differences?

Conclusions and Takeaways

BOSCH

What's different about fuzzing automotive software?

Wrap Up

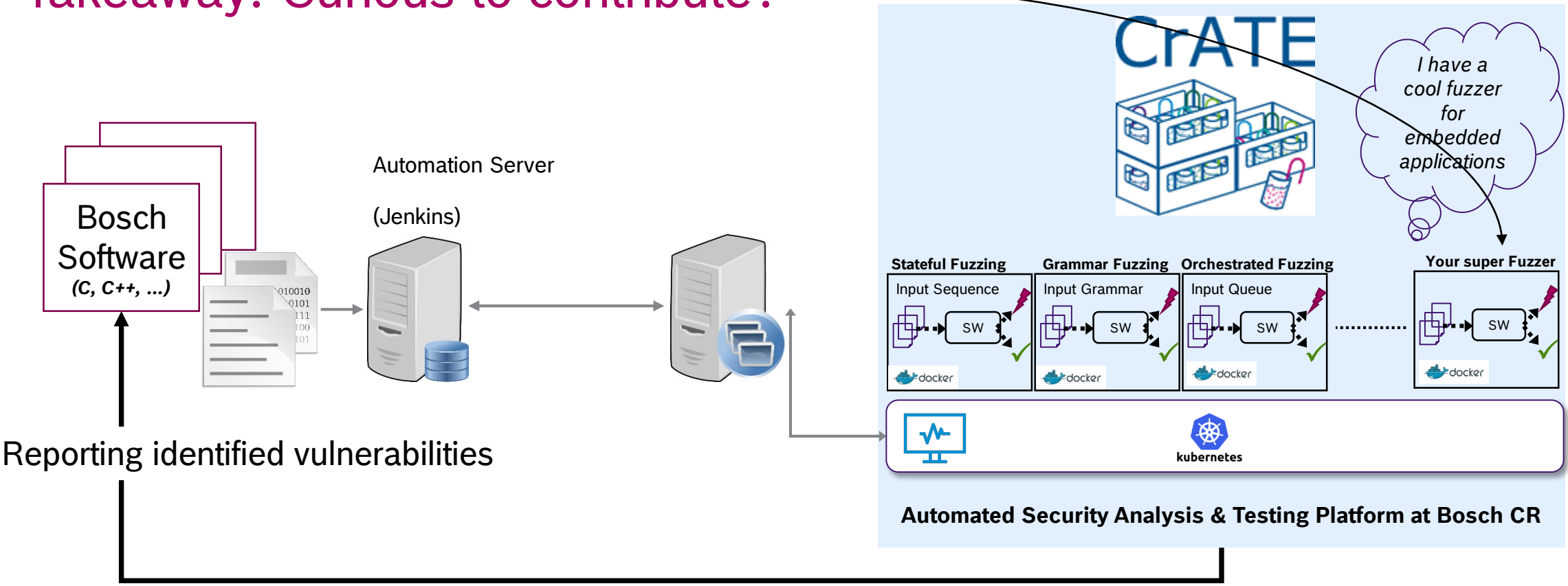


- Why fuzz automotive software?
  - Intrinsic motives
  - Extrinsic motives
- What's the difference?
  - High entry barrier for fuzzing
  - Stateful software
  - Native binaries complicating fuzzing
- How to cope with the differences?
  - Ideas to lower entry barrier
  - Thoughts on fuzzing stateful programs
  - Thoughts on fuzzing native binaries

Fuzzing IS and CONTINUES to be important for automotive (IoT) software

What's different about fuzzing automotive software?

Takeaway: Curious to contribute?



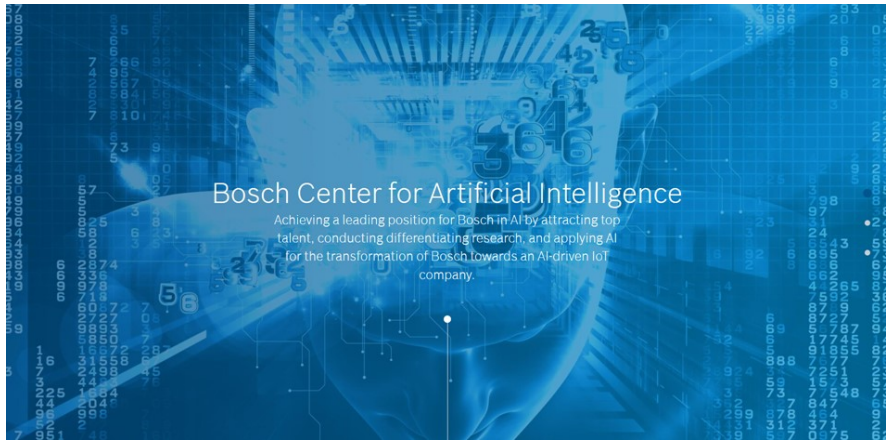
Extensible platform enables contributions from the fuzzing community

What's different about fuzzing automotive software?
Acknowledgements and Contact

- This talk wouldn’t have been possible without the ideas and great work of my team of experts, researchers and engineers. I sincerely thank the following people:
  - Dr. Simon Greiner
  - Dr. Christopher Huth
  - Mark Giraud
  - Dr. Nishant Sinha
  - Anjali Bhat and
  - Rajat Jaiswal
- Management support for the topic has been tremendous so I also sincerely thank:
  - Dr. Paul Duplys and
  - Dr. Andreas Nauerz

Feel free to reach out to me on LinkedIn or ✉ firstname[dot]lastname[at]bosch[dot]com

What's different about fuzzing automotive software?
Time for Questions



We are hiring!

Our Mantra: DevSecOps

Join my team

#software security

- Research Scientist
- Research Scientist
- PhD position