ICS Protocol Fuzzing: Coverage Guided Packet Crack and Generation

Zhengxiong Luo[1], Feilong Zuo[1], Yuheng Shen[1], Xun Jiao[2], Wanli Chang[3], Yu Jiang[1]

[1]Tsinghua University, [2]Villanova University, [3]University of York

UNIVERSITY of York

Outline

- Introduction
  - Background
  - Motivation
- Peach*
  - System Design
  - Evaluation
- Conclusion

Industrial Control System

- System of electronic components that control the physical operations of machines.
- Support critical infrastructure
  - Power system
  - Transportation
  - …

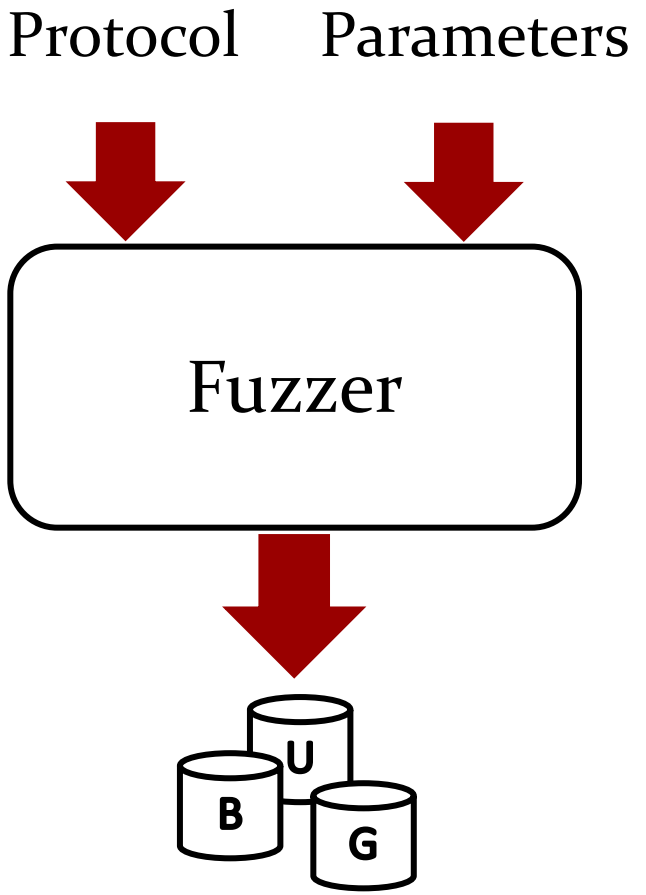
Industrial Control System Protocol

- Build communications among system components and devices.
- ICS is becoming more open, which has increased the susceptibility to attack, primarily due to greater awareness of ICS protocols.
- Frequent accidents arising from ICS protocol gravely threaten the ICS.

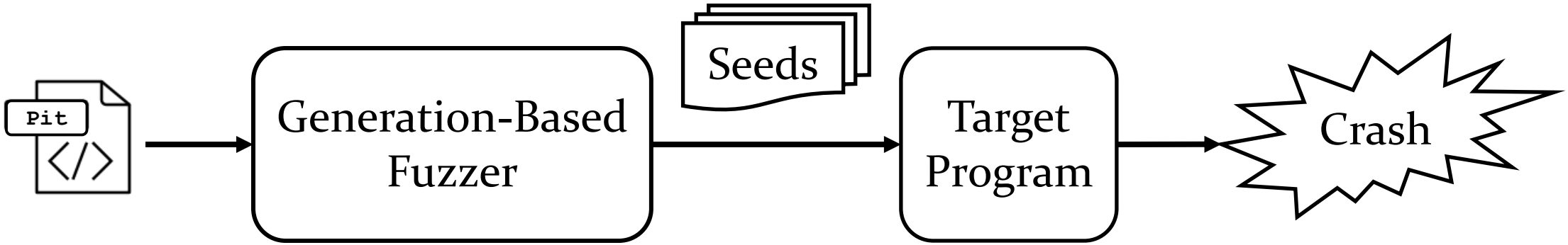
Attack	Year
Venezuela Blackout	2019
Saudi Arabia TRISIS	2017
Ukraine CRASHOVERRIDE	2016
Ukraine BLACKENERGY3	2015
German Steel Mill Cyber Attack	2014
DragonFly	2014
Havex Malware	2013
Telvent Canada Attack	2012

Fuzz Testing for ICS Protocol

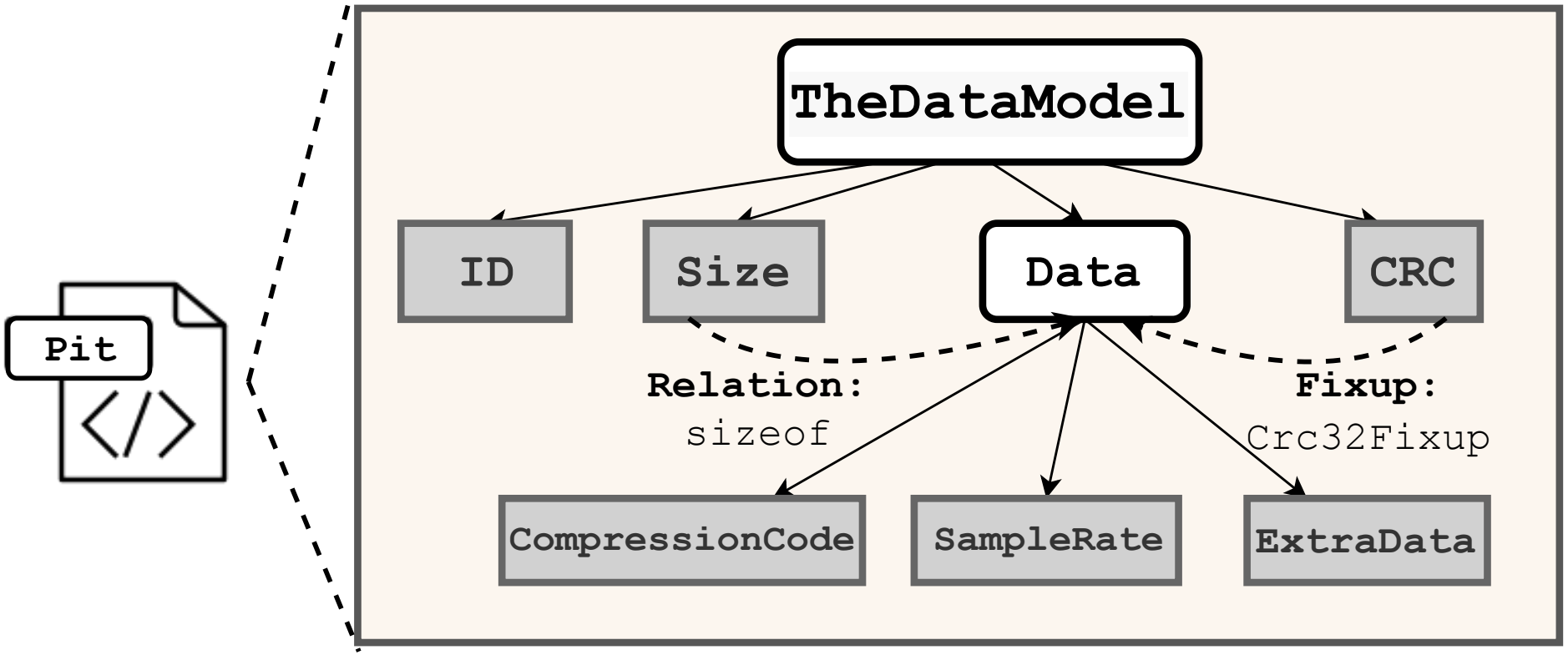
american fuzzy lop (2.52b)



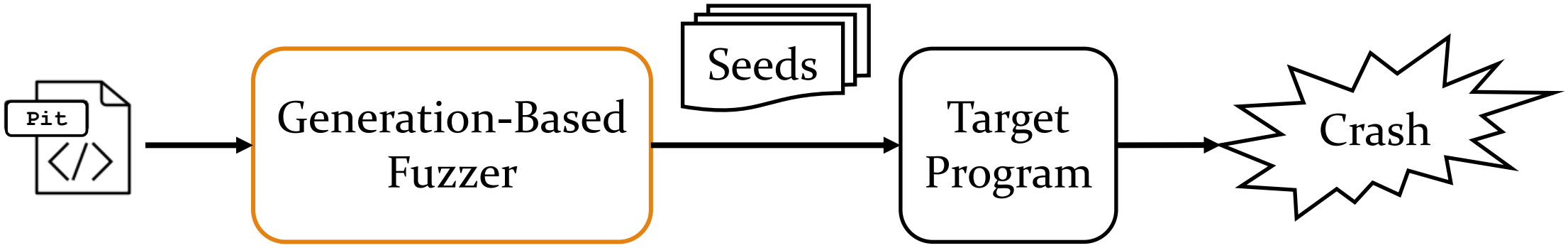
Generation-Based Fuzzing



File as a Tree

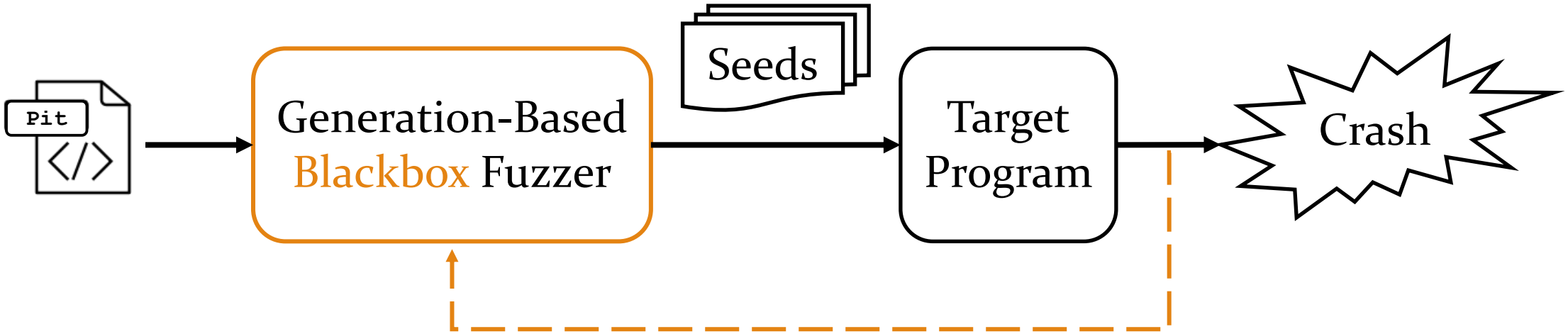


Challenges



- Generation strategy is random and pointless.

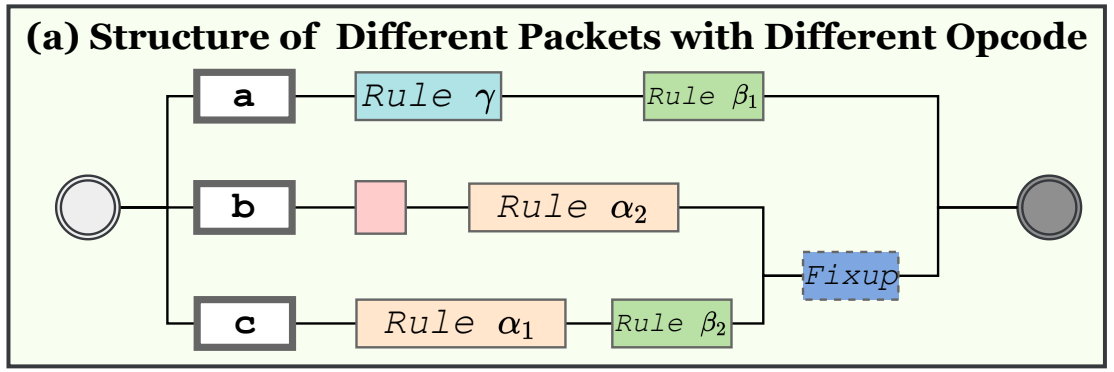
Challenges



- Blackbox: no access to the application's internals.
- Previously generated valuables seeds are discarded.

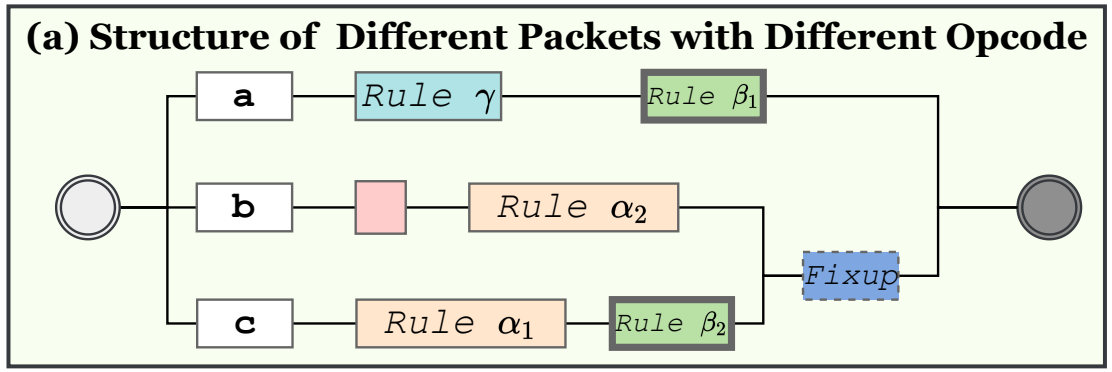
Motivation

- Opcode field: encode the instruction to be performed.



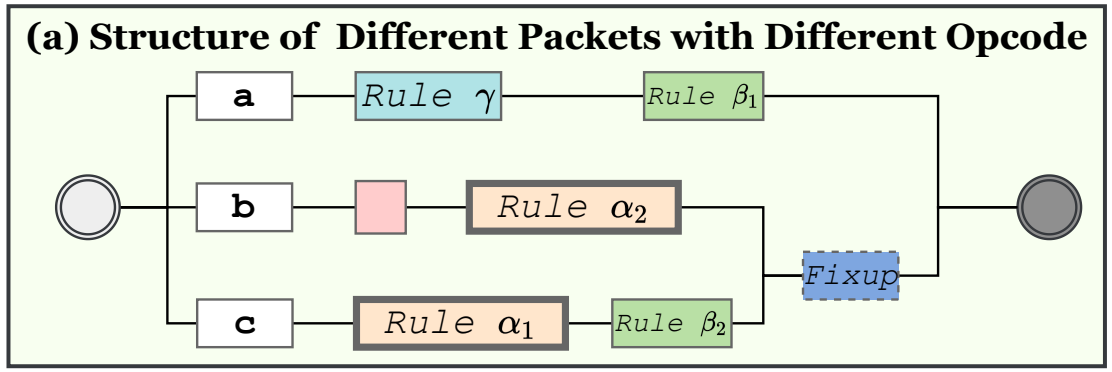
Motivation

- Opcode field: encode the instruction to be performed.
- These different types of packets would share similar data chunks.



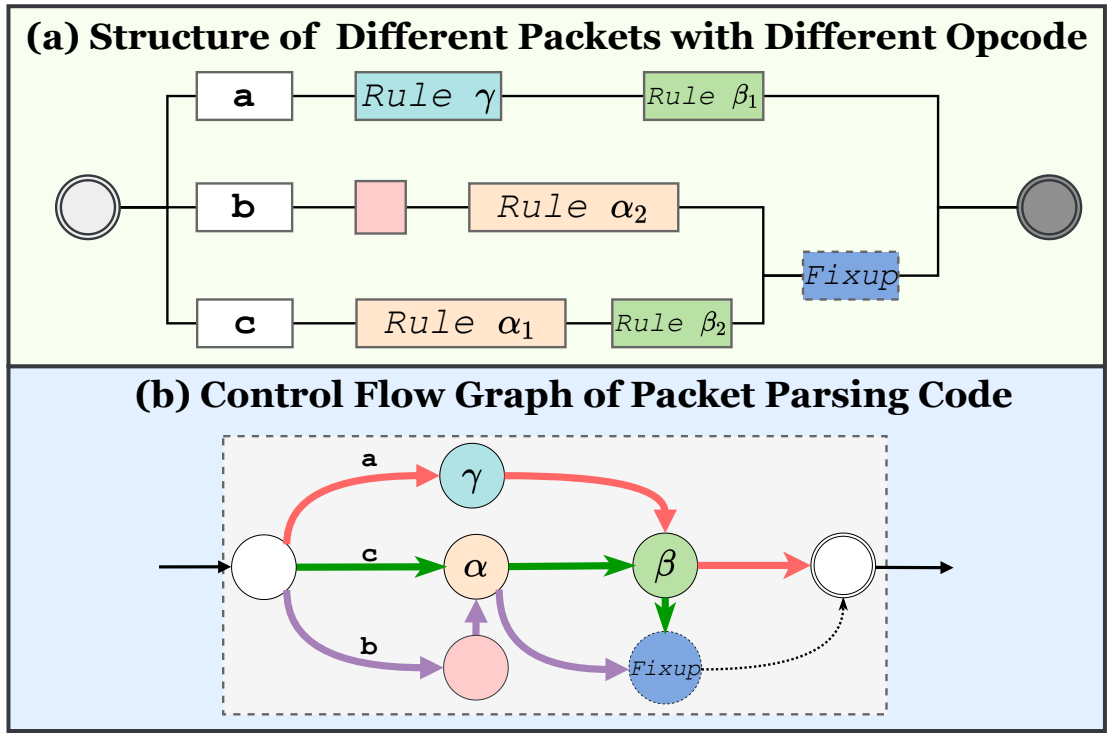
Motivation

- Opcode field: encode the instruction to be performed.
- These different types of packets would share similar data chunks.



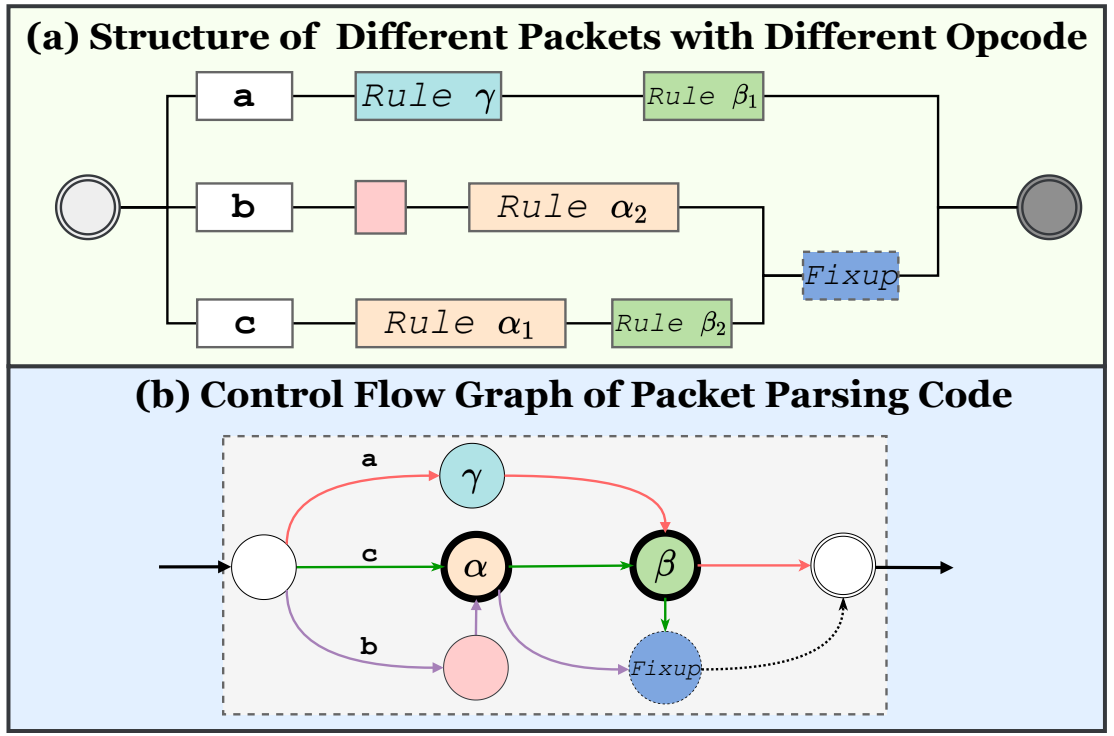
Motivation

- Opcode field: encode the instruction to be performed.
- These different types of packets would share similar data chunks.
- They would trigger different program traces.



Motivation

- Opcode field: encode the instruction to be performed.
- These different types of packets would share similar data chunks.
- They would trigger different program traces.
- Similarly, these traces would share some parsing code blocks.

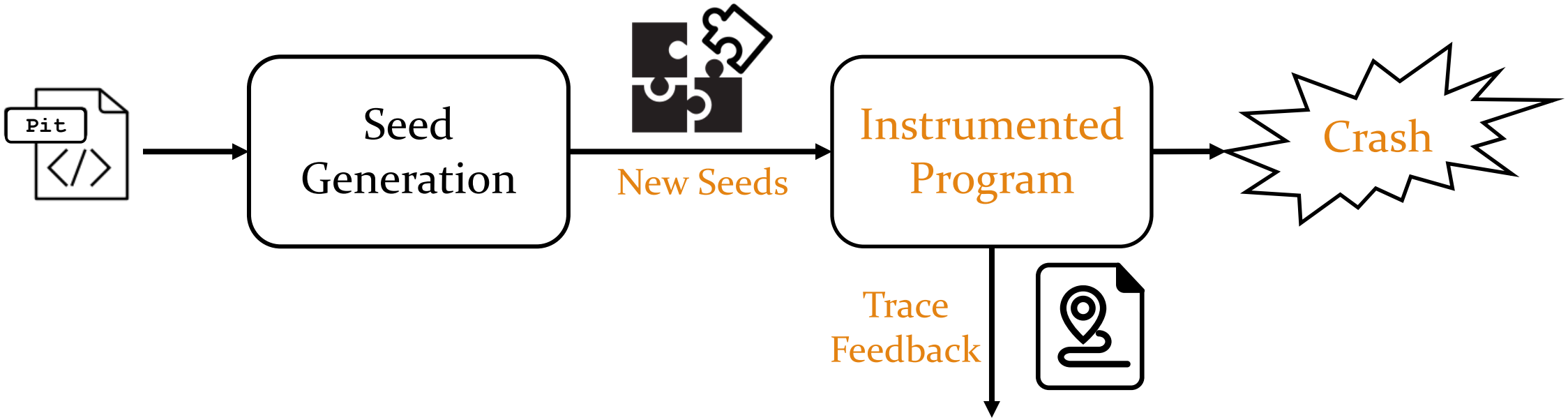


Outline

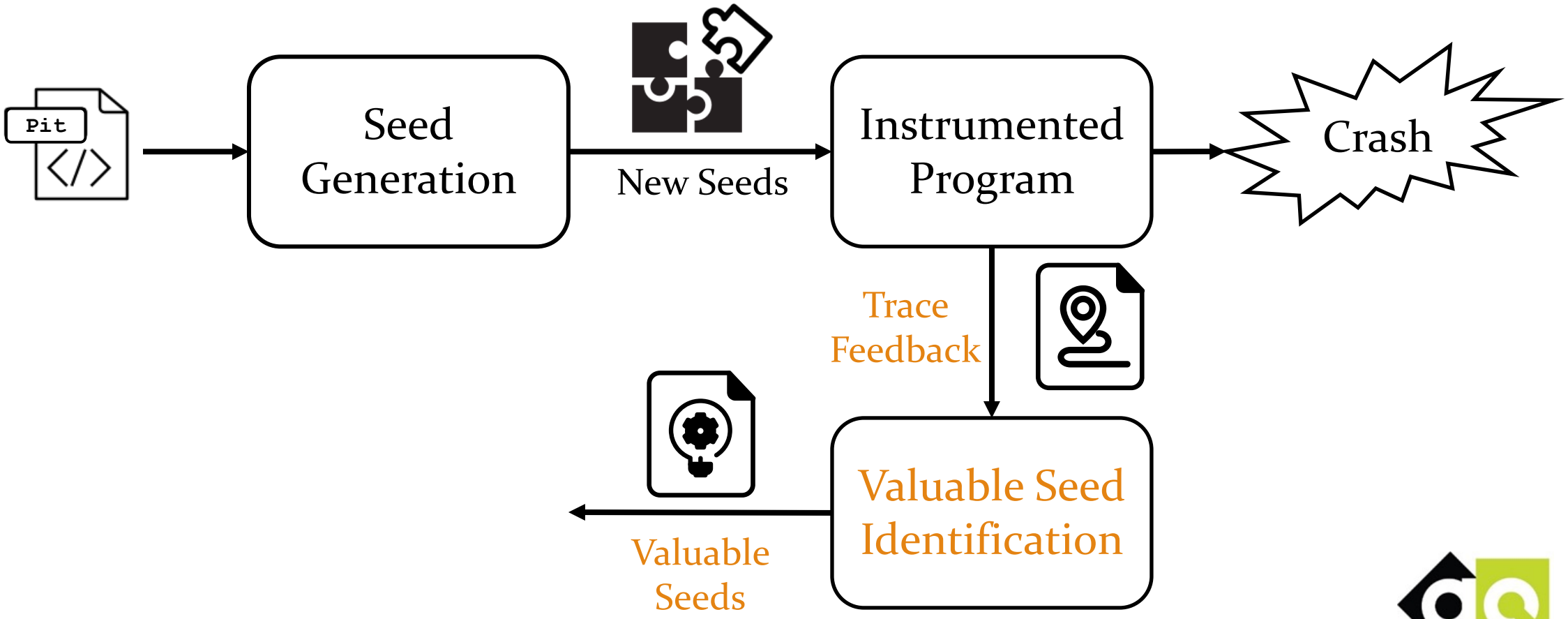
- Introduction
  - Background
  - Motivation
- Peach*
  - System Design
  - Evaluation
- Conclusion

Peach* Overview

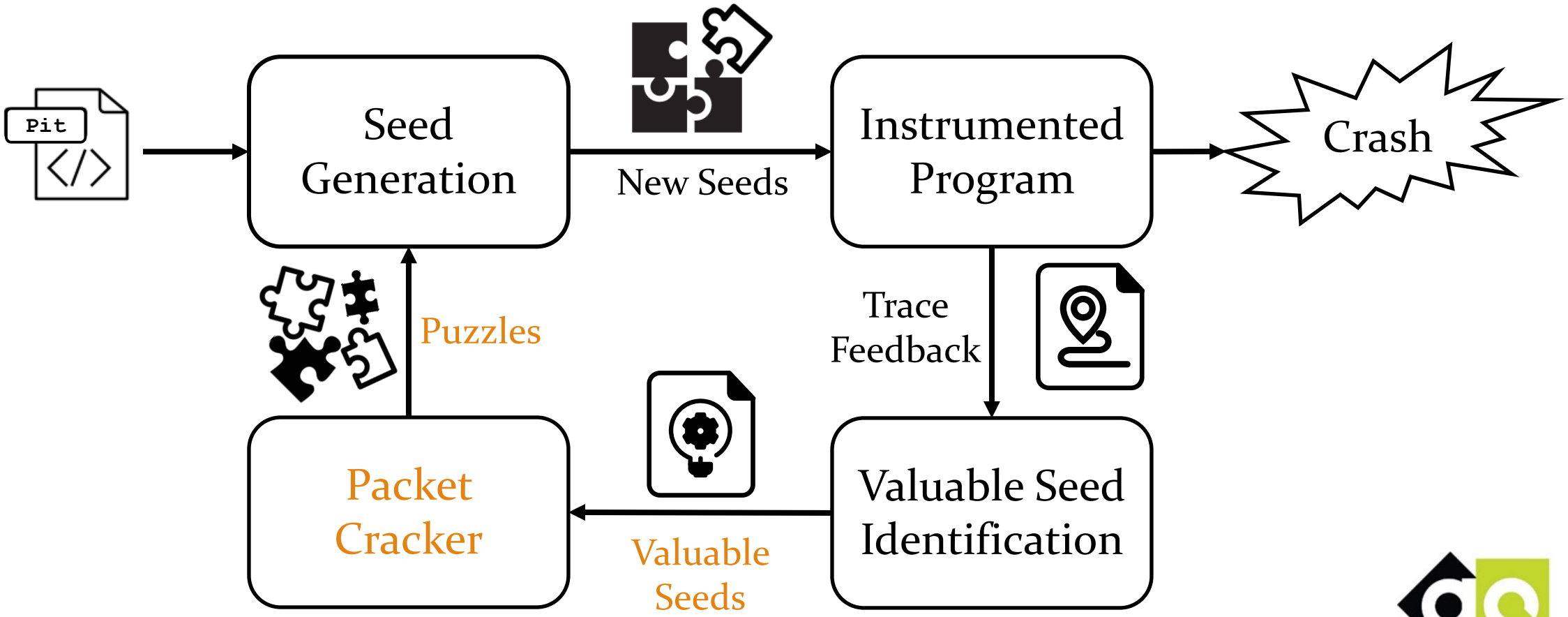
Peach* Overview



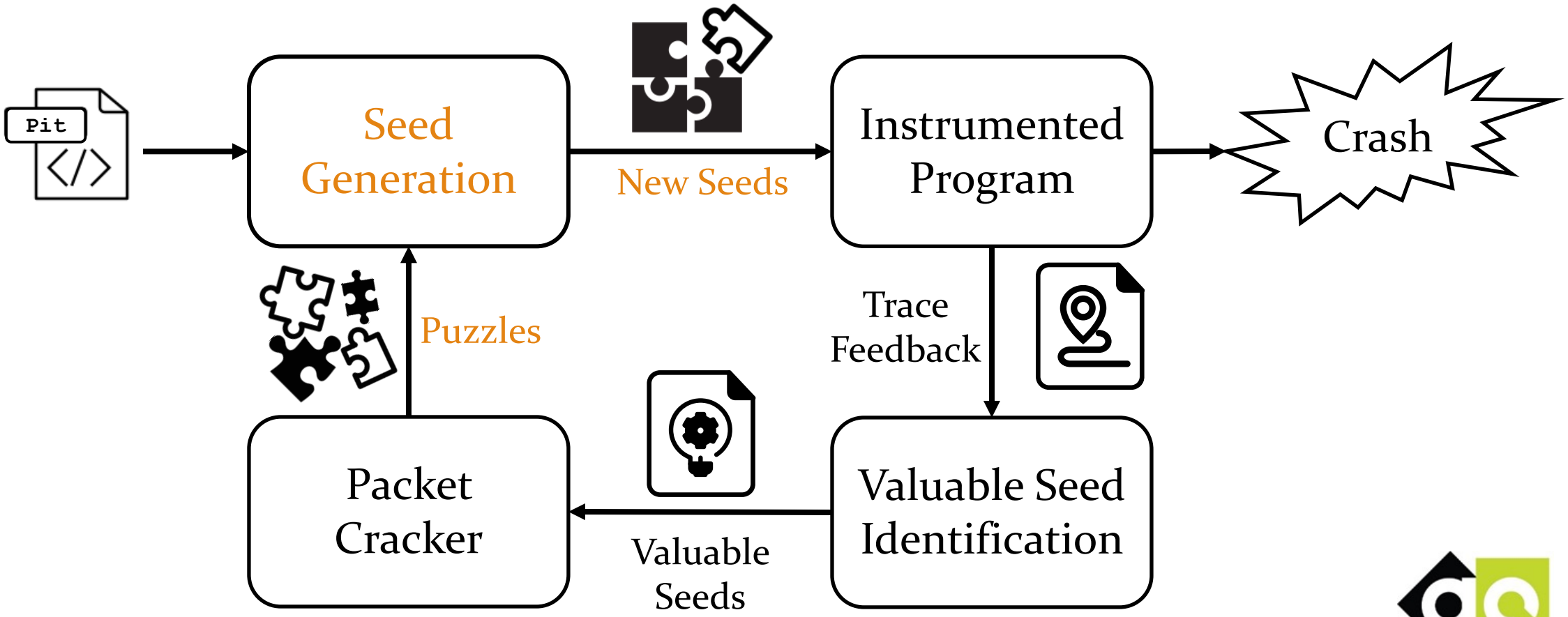
Peach* Overview



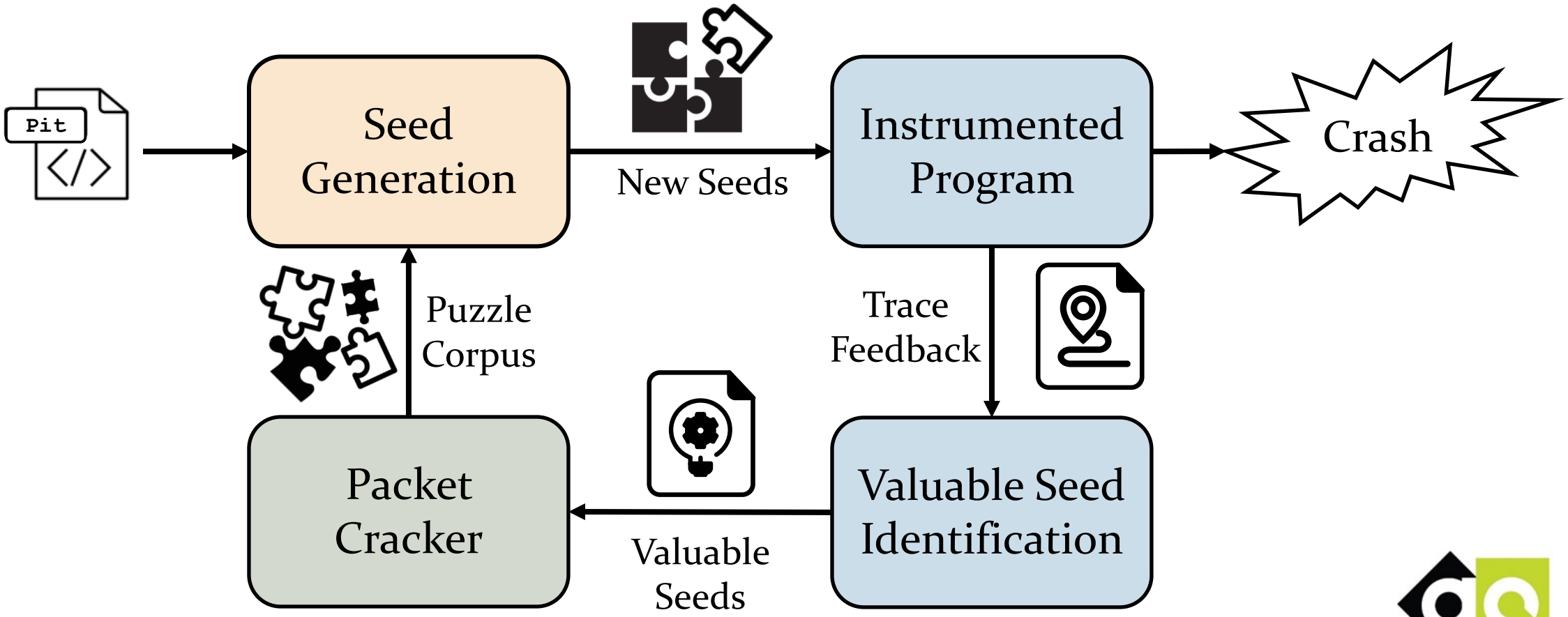
Peach* Overview



Peach* Overview

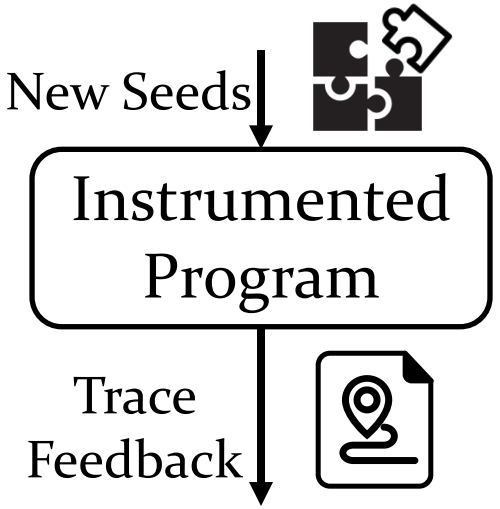


Peach* Overview



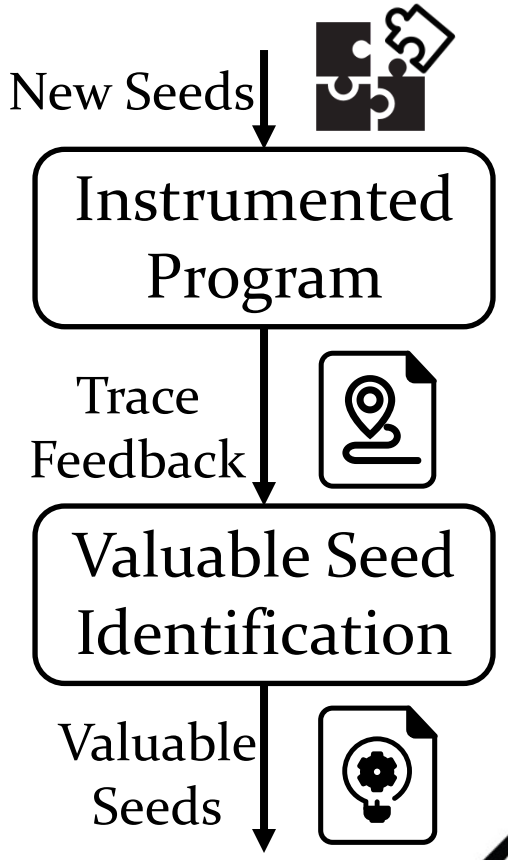
Valuable Seeds Identification

- By injecting instrumentation at branch points in the target program, the execution trace of each seed is available.



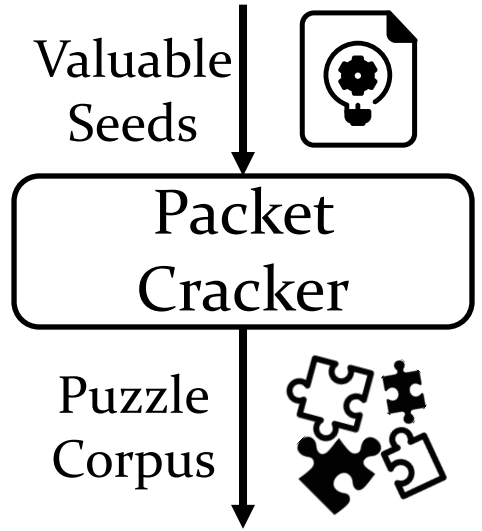
Valuable Seeds Identification

- By injecting instrumentation at branch points in the target program, the execution trace of each seed is available.
- Then, these seeds that achieve new code coverage are considered valuable.



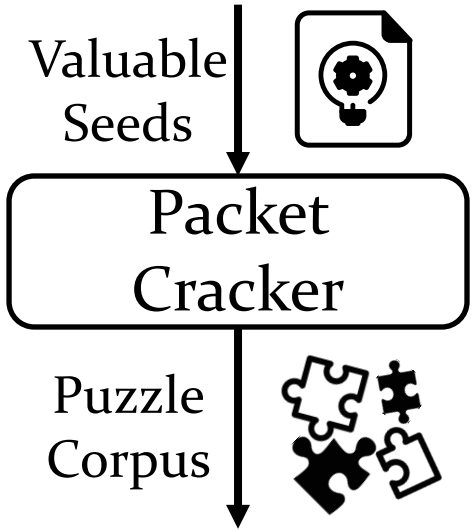
Packet Cracker

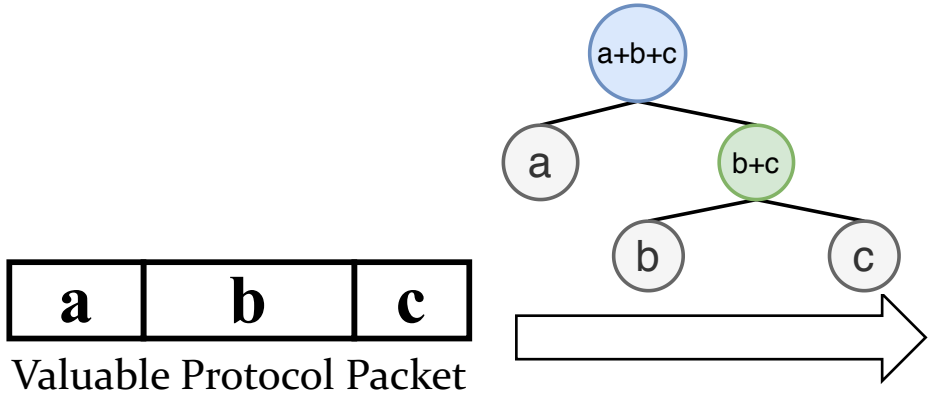
- Those valuable seeds would be cracked into puzzles to improve seed generation.

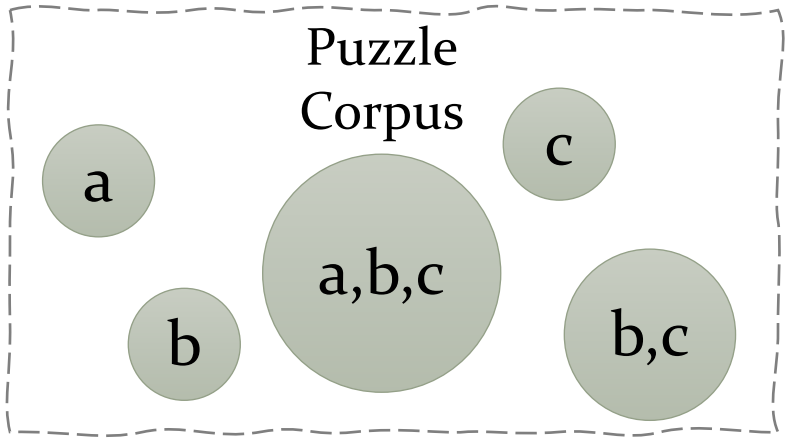


Packet Cracker

- Those valuable seeds would be cracked into puzzles to improve seed generation.

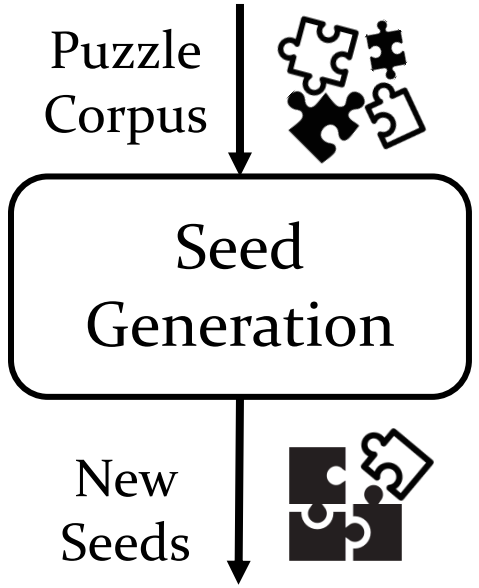






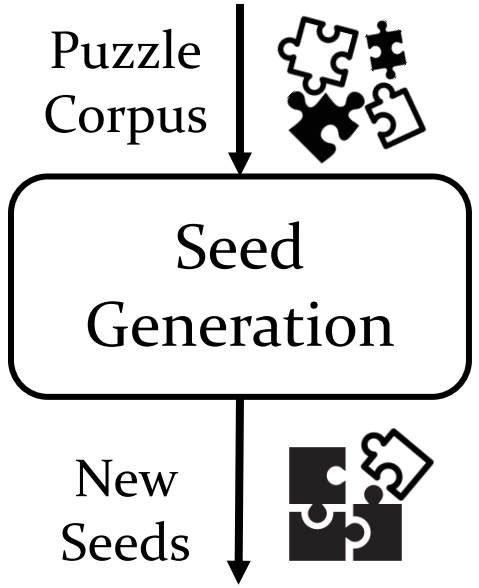
Seed Generation

- If the puzzle corpus is vacant, the seed generation strategy remains inherent.



Seed Generation

- If the puzzle corpus is vacant, the seed generation strategy remains inherent.
- Otherwise, semantic aware generation strategy would be applied:
  - For each chunk to generate, Peach* find those puzzles that conform to its rule from corpus.
  - Then, those appropriate pieces are selected to derive new seeds in preference to instantiation from input model.



Evaluation

- E1: Is Peach* more efficient than Peach, when augmented with the proposed fuzzing strategy?

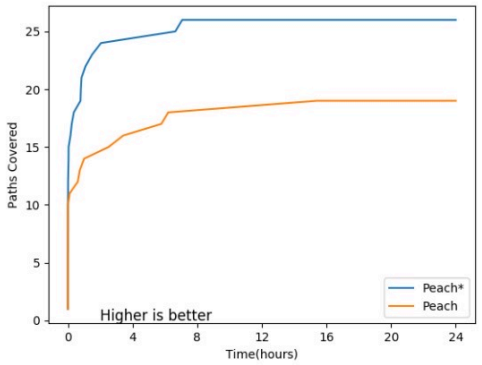
- E2: Is Peach* effective in exposing previously unknown vulnerabilities in real-world ICS protocol applications?

Evaluation Setup

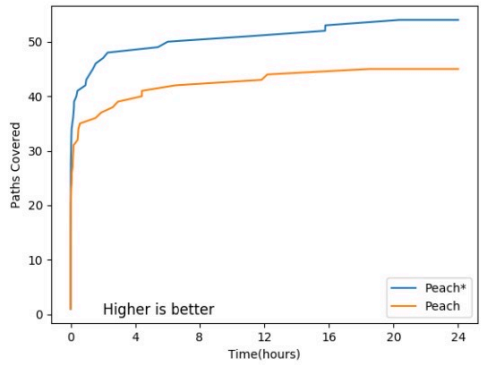
- We selected several widely used open-source implementations of ICS protocols.
- Including Modbus, IEC 61850, DNP3, etc.

- Those ICS protocols are international standard widely used in critical infrastructures.

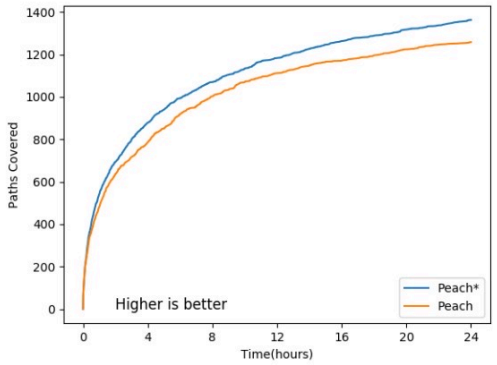
E1: Optimize Fuzzing



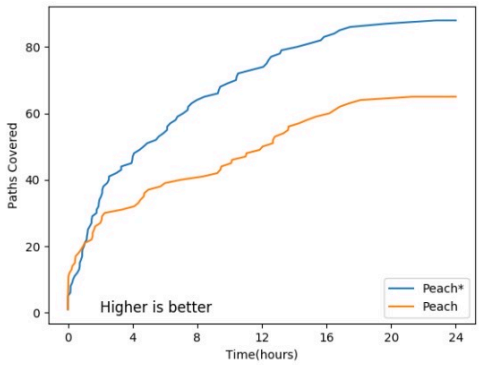
(a) **libmodbus**



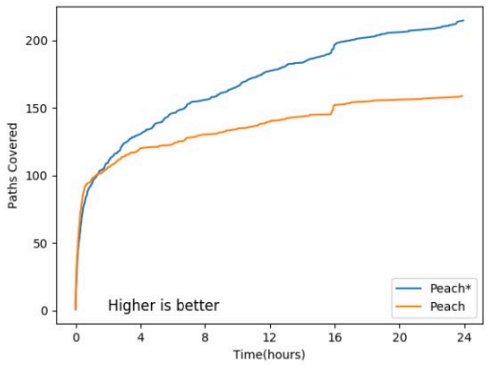
(b) **IEC104**



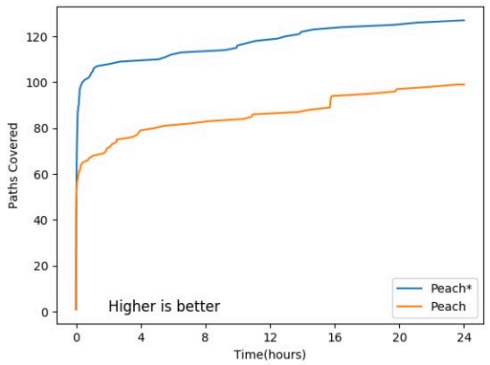
(c) **libiec61850**



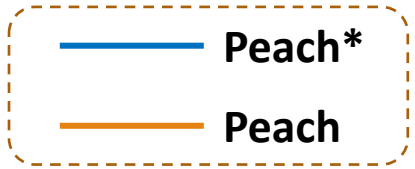
(d) **lib60870**



(e) **libiccp**

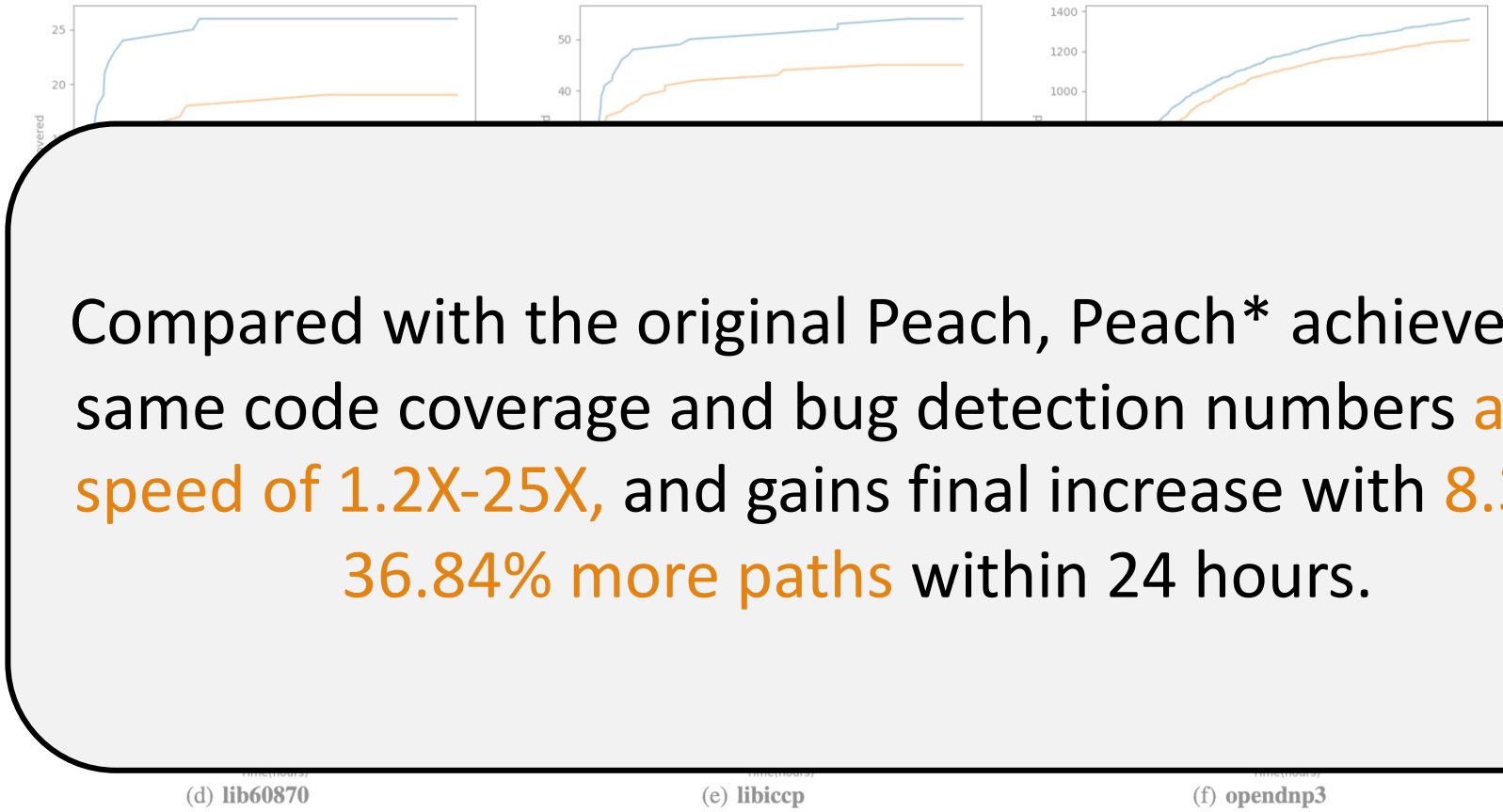


(f) **opendnp3**



Average number of paths covered by Peach* and Peach within 24 hours for 10 repetitions on each ICS protocol program.

E1: Optimize Fuzzing



Compared with the original Peach, Peach* achieves the same code coverage and bug detection numbers at the speed of 1.2X-25X, and gains final increase with 8.35%-36.84% more paths within 24 hours.

Average number of paths covered by Peach* and Peach within 24 hours for 10 repetitions on each ICS protocol program.

E2: Previously Unknown Vulnerabilities

- Peach* has exposed 9 previously unknown vulnerabilities.

TABLE I: Vulnerabilities Exposed by Peach*

Project	Vulnerability Type	Number	Status
lib60870	SEGV	3	Confirmed
libmodbus	Heap Use after Free SEGV	1 1	Confirmed
libiec_iccp_mod	SEGV Heap Buffer Overflow	3 1	Confirmed

Outline

- Introduction
  - Background
  - Motivation
- Peach*
  - System Design
  - Evaluation
- Conclusion

Conclusion

- Peach* is an automated fuzzing tool targeted for ICS protocol vulnerability detection and can perform:
  - Use coverage information as guidance;
  - Build high-quality corpus based on the cracked packet pieces;
  - Perform semantic aware generation strategy.
- Peach* outperforms Peach:
  - 10~40% coverage improvement
  - 2~25X speedup.
- Peach* has been tested on several ICS protocols and detected many serious previously-unknown vulnerabilities in Modbus, IEC 60870, ICCP, etc.

Thanks for your attention!

Q&A

luozx19@mails.tsinghua.edu.cn