RVFuzzer: Finding Input Validation Bugs in Robotic Vehicles through Control-Guided Testing

Taegyu Kim, Chung Hwan Kim, Junghwan Rhee, Fan Fei, Zhan Tu, Gregory Walkup, Xiangyu Zhang, Xinyan Deng, Dongyan Xu

USENIX Security '19

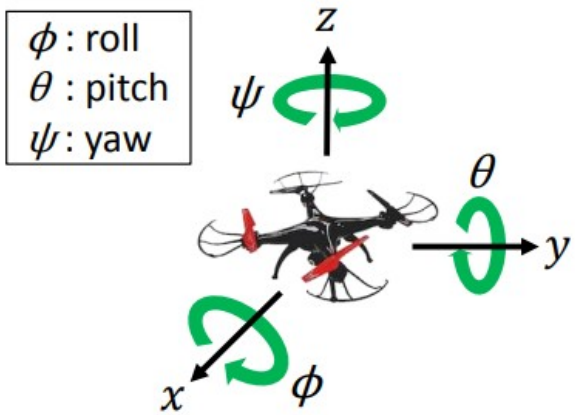
Presenter : Hyeon Heo

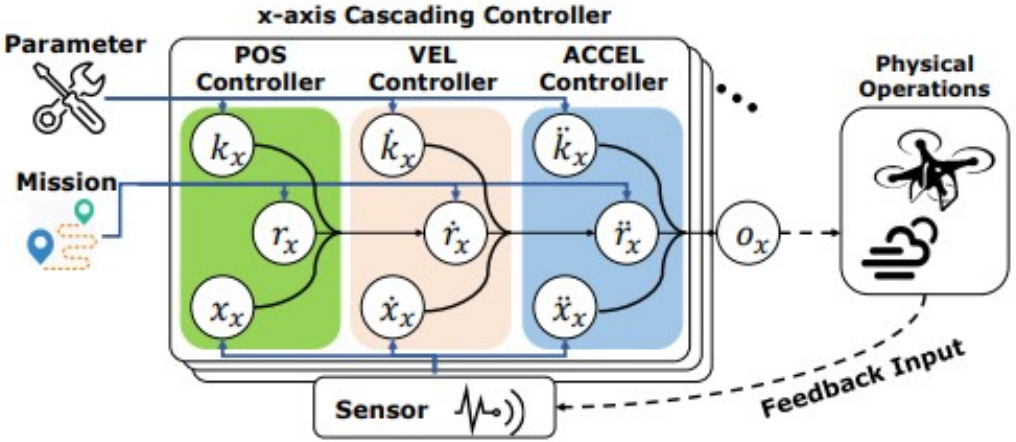
Robotic Vehicle (RV)

Landscape of RV Attacks

- Physical vulnerabilities
  - External sensor spoofing
- Syntactic bugs in software
  - e.g., memory corruption bugs
- **Control-semantic bugs in control program**

RV Control Model





RV Control Model

- RV Control Program involves
  - sensor module
  - controller module
  - mission module

- RV Control Program communicates with a ground control station (GCS) during a flight

- There are lots of control parameters in RV

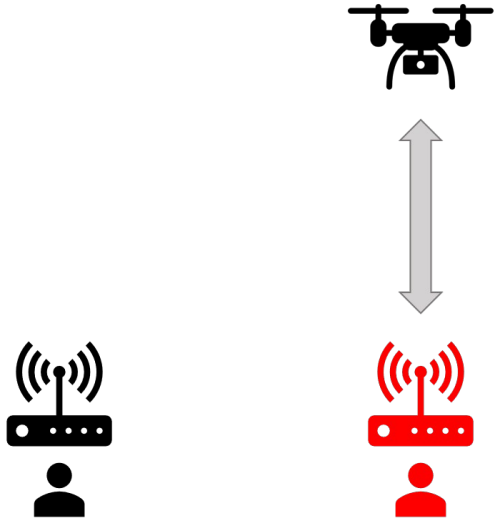
Attack Model

- Attackers can issue parameter change GCS command to victim RV
  - Cause at least one of the RV’s 6DoF controllers to malfunction

- Also, attacker can use specific environmental condition
  - For example, strong wind or sharp turn

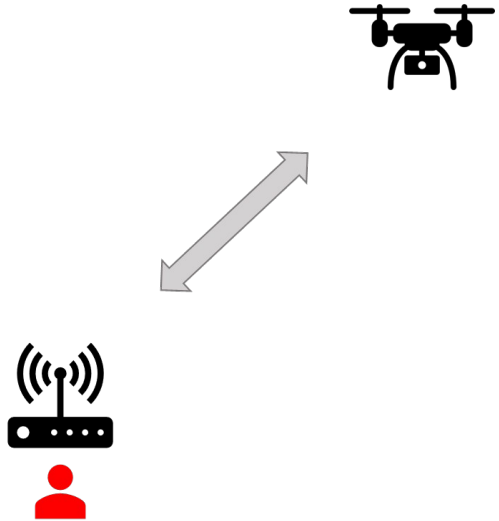
Attack Model

- External Attacker



e.g.) GCS Spoofing

- Insider Threat



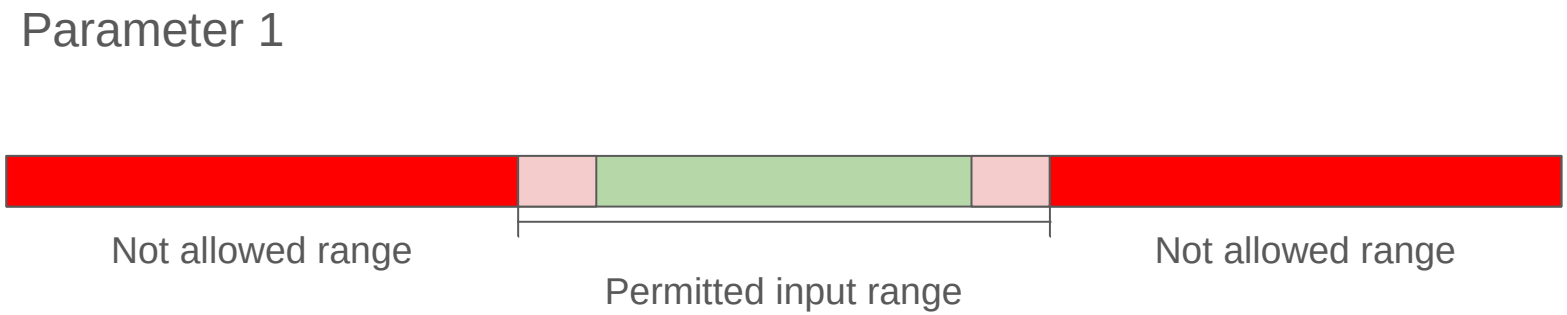
e.g.) malicious operator

Why Manipulate Parameter Values?

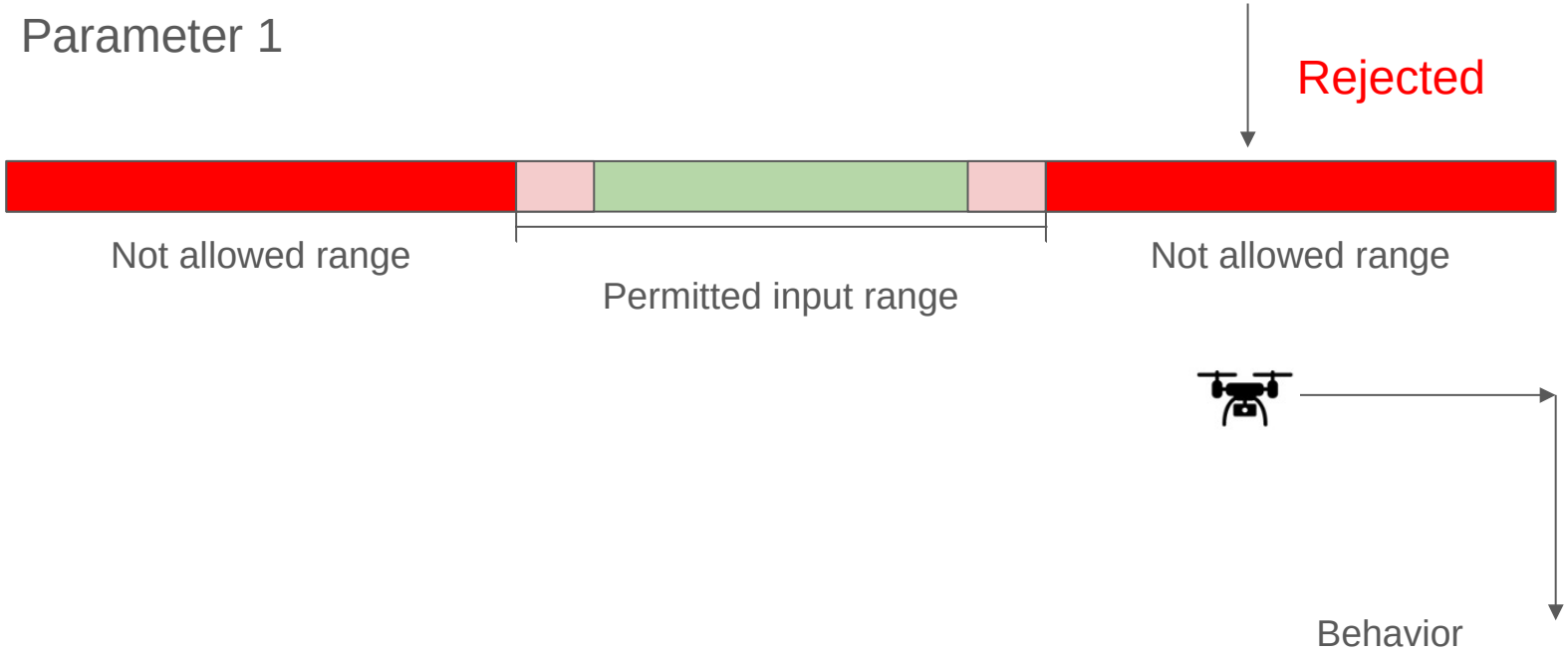
- Small footprint
  - One innocent-looking command can crash RV

- Can be conducted even though there are software bug mitigation

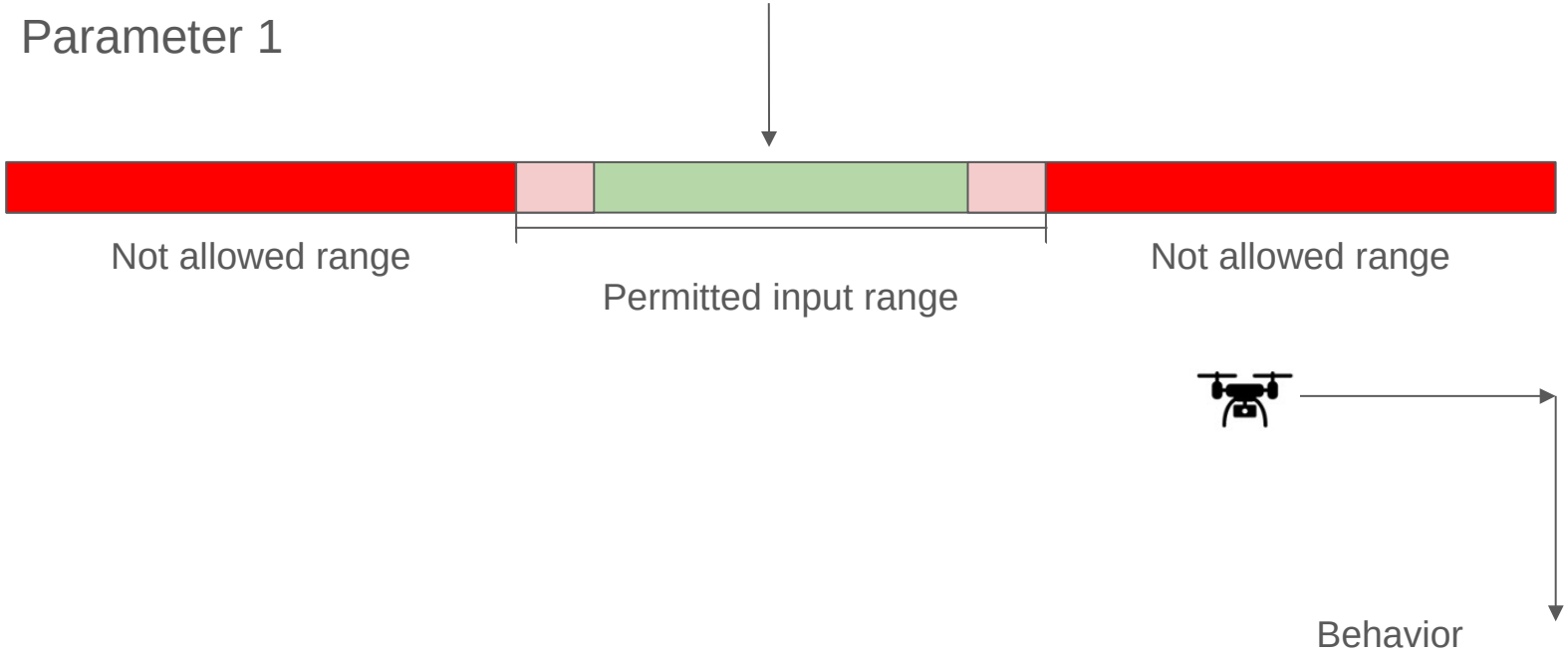
Nature of Control-Semantic Bug



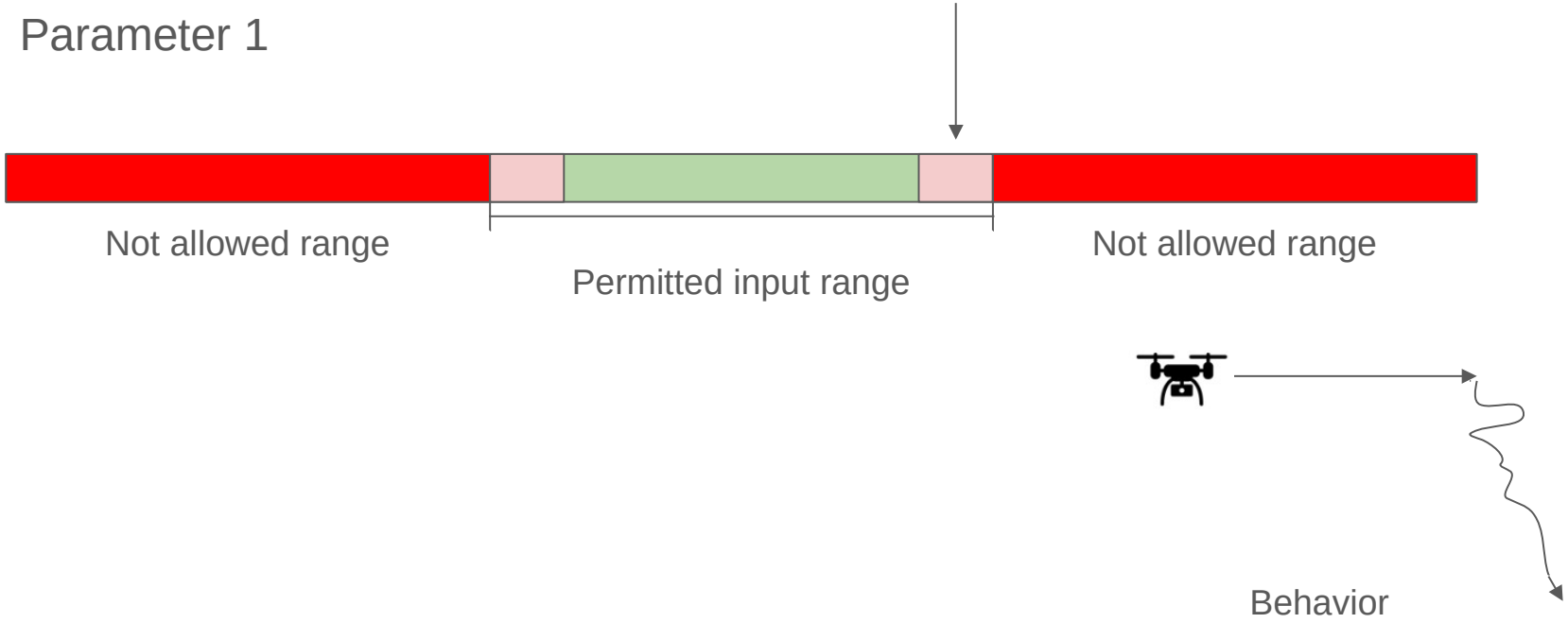
Nature of Control-Semantic Bug



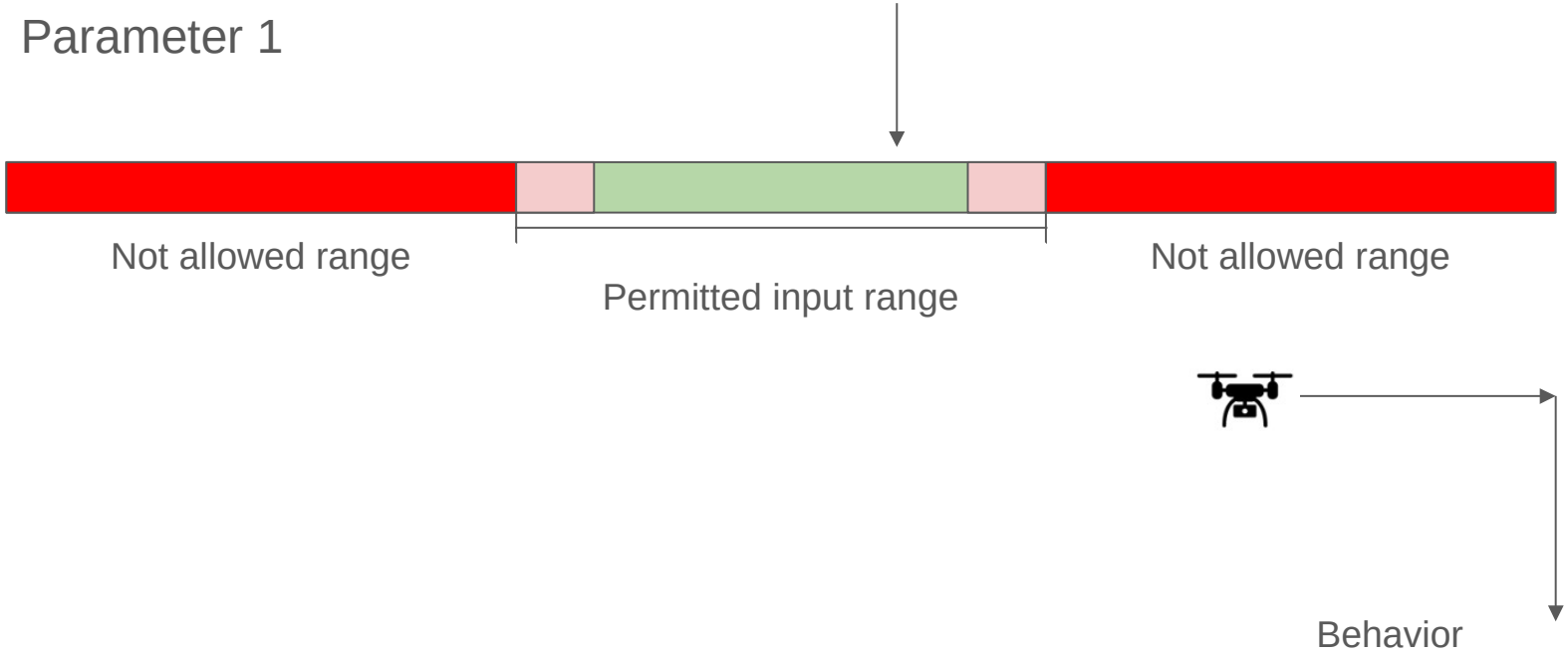
Nature of Control-Semantic Bug



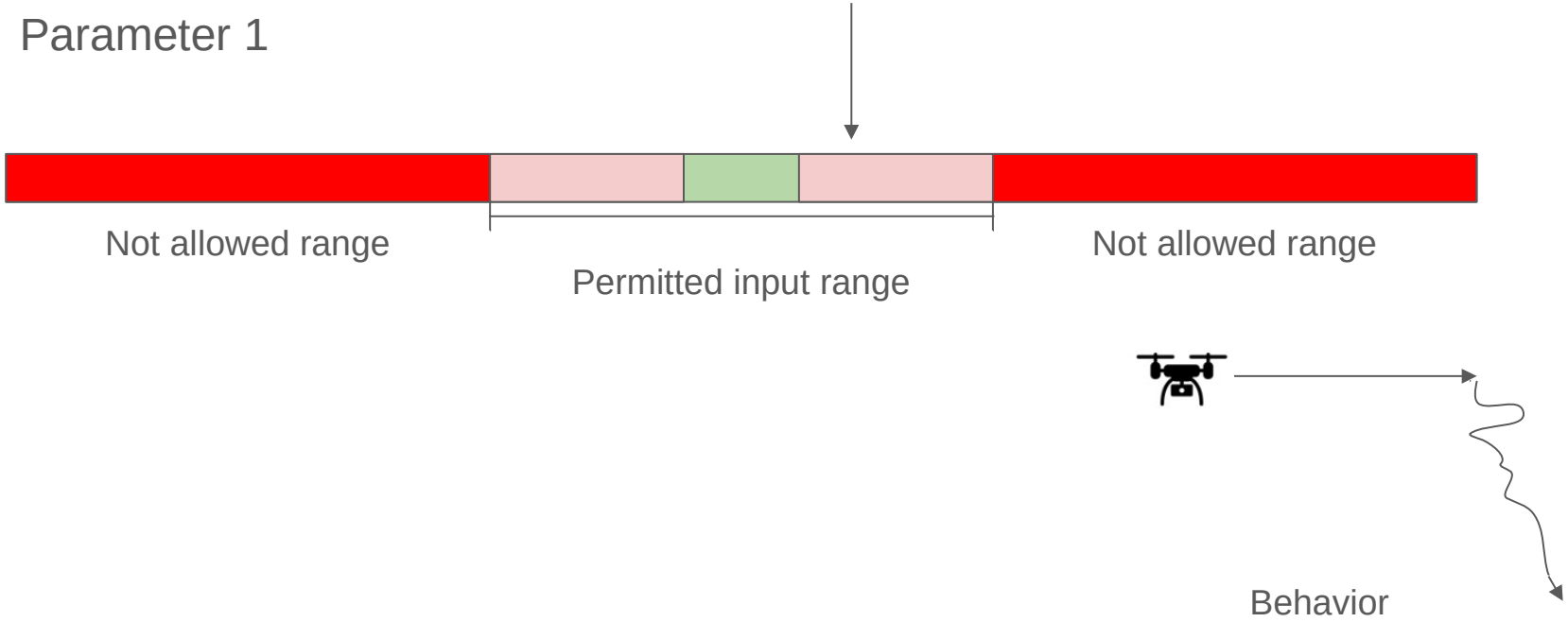
Nature of Control-Semantic Bug



Nature of Control-Semantic Bug - Without Wind



Nature of Control-Semantic Bug - With Wind



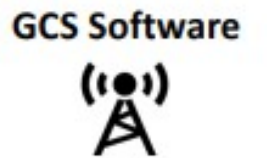
Finding the Bugs

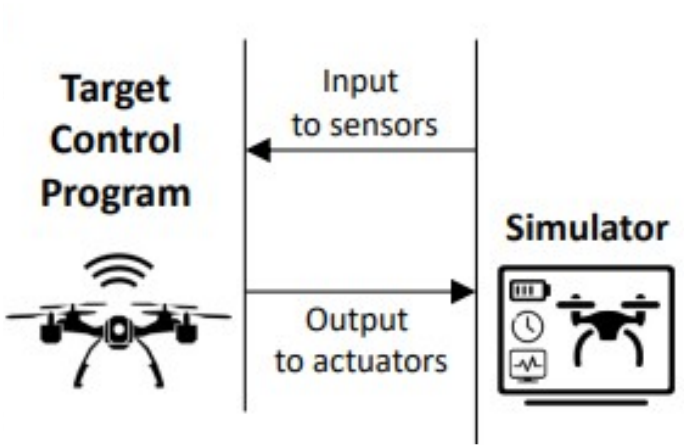
- How to detect a bug
  - Check non-transient divergence between reference state and observed state

- How to conduct fuzzing
  - For safety: Use simulator
  - For efficiency: Control-guided, feedback-directed fuzzing

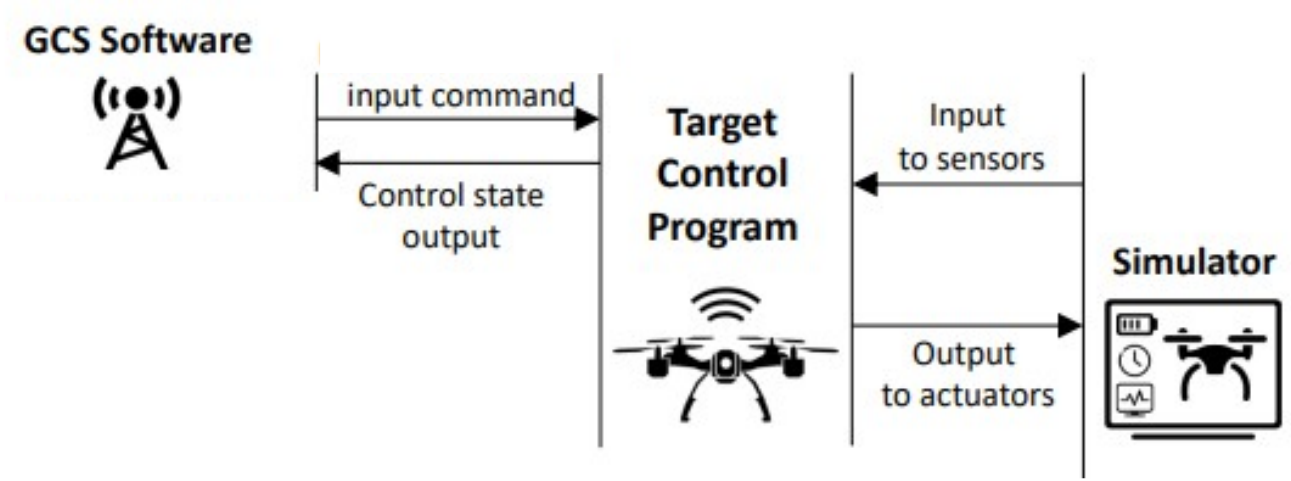
RVFuzzer

RVFuzzer

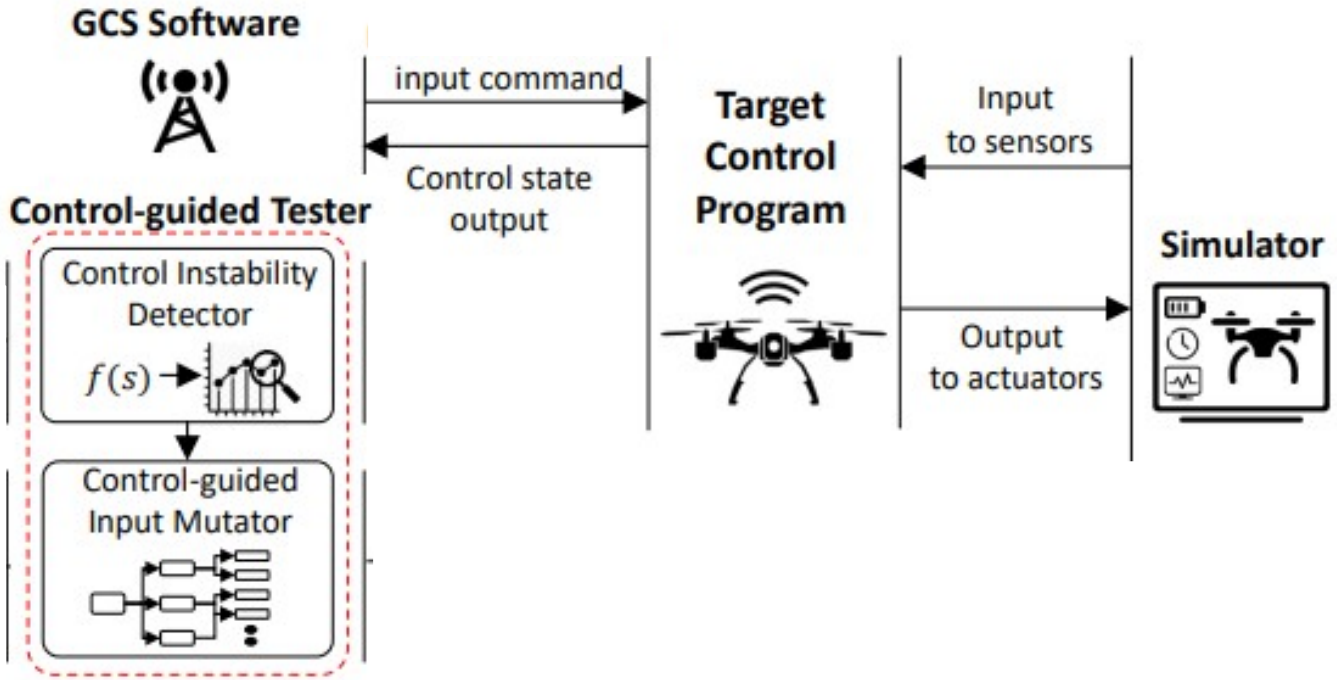

GCS Software


Target
Control
Program
Input
to sensors
Simulator
Output
to actuators

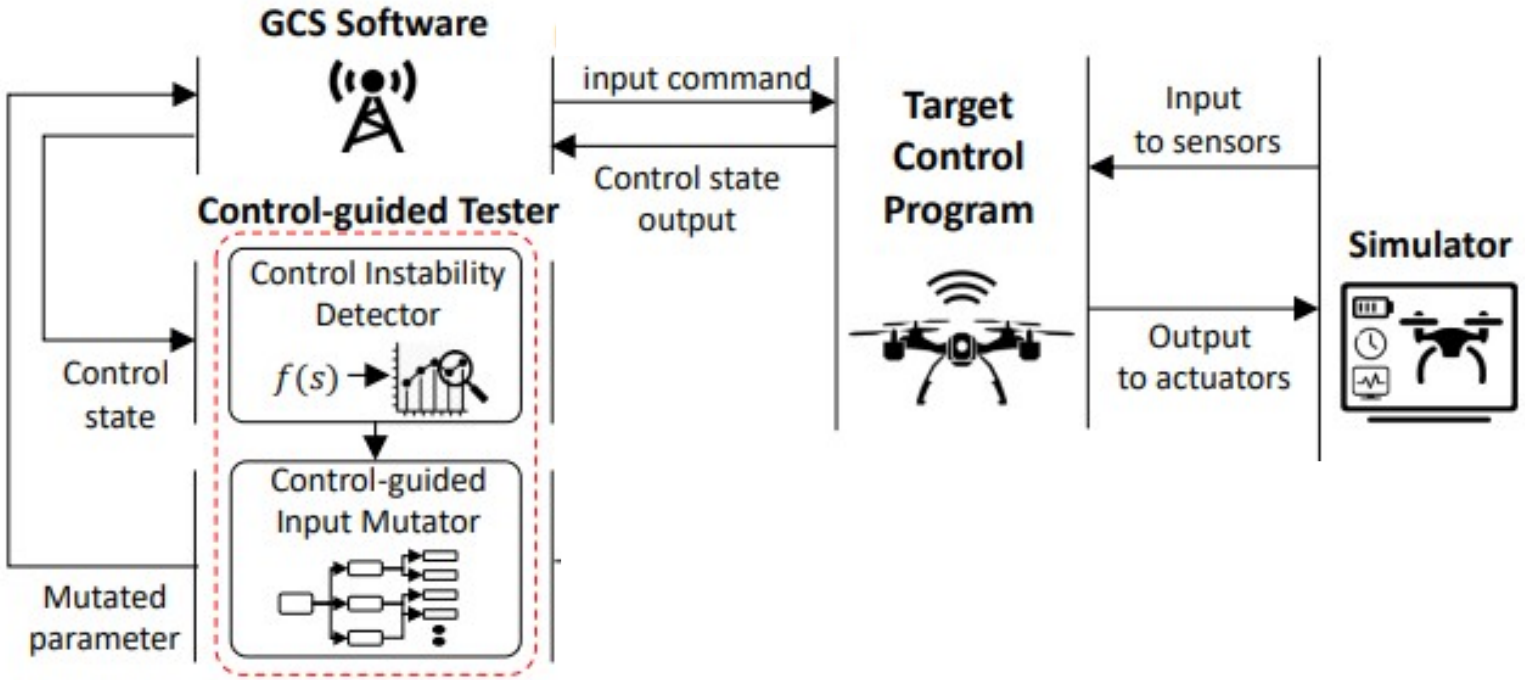
RVFuzzer


GCS Software
input command
Control state output
Target Control Program
Input to sensors
Output to actuators
Simulator

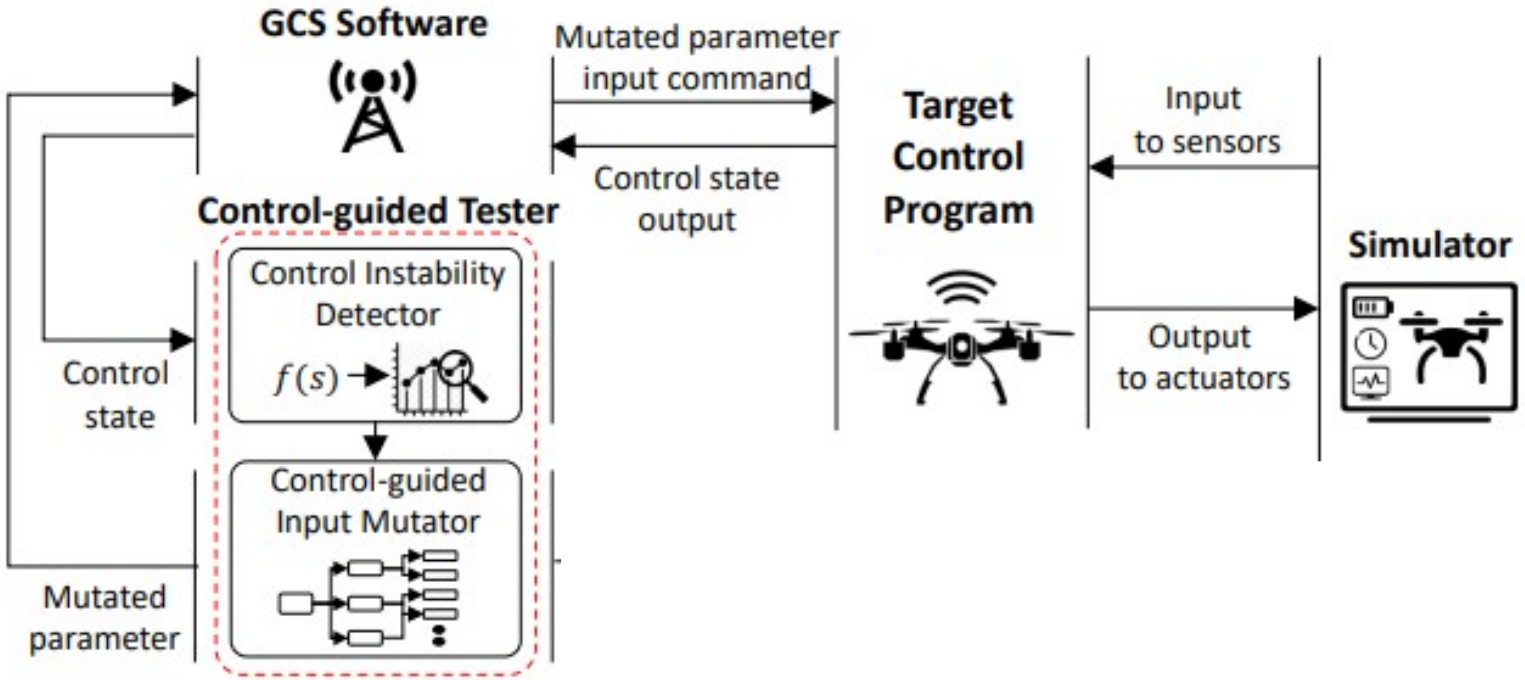
RVFuzzer


GCS Software
input command
Control state output
Target Control Program
Input to sensors
Output to actuators
Simulator
Control-guided Tester
Control Instability Detector
$f(s)$
Control-guided Input Mutator

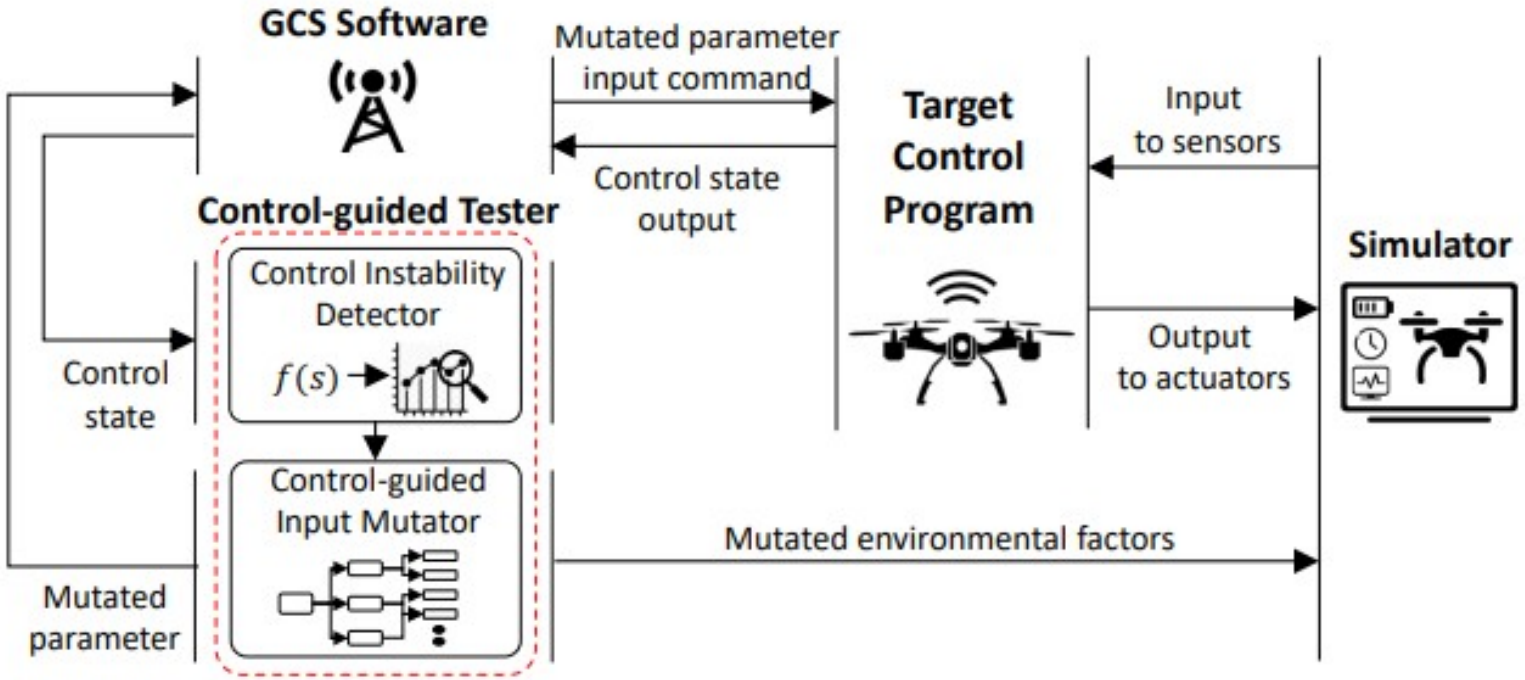
RVFuzzer


GCS Software
input command
Control state output
Target Control Program
Input to sensors
Simulator
Control-guided Tester
Control Instability Detector
$f(s)$
Control state
Control-guided Input Mutator
Mutated parameter
Output to actuators

RVFuzzer


GCS Software
Mutated parameter input command
Control state output
Target Control Program
Input to sensors
Output to actuators
Simulator
Control-guided Tester
Control Instability Detector
$f(s)$
Control state
Control-guided Input Mutator
Mutated parameter

RVFuzzer


GCS Software
Mutated parameter input command
Control state output
Target Control Program
Input to sensors
Simulator
Control-guided Tester
Control Instability Detector
$f(s)$
Control state
Output to actuators
Control-guided Input Mutator
Mutated environmental factors
Mutated parameter

Mutation

- One-dimensional Mutation
  - For determining the valid/invalid range for each control parameter independently
  - Based on binary-search
- Multi-dimensional Mutation
  - For finding extra invalid values when parameters have dependencies on other parameters
  - Perform one-dimensional mutation recursively to parameters
- Environmental Factors
  - For finding cases when external factors (e.g., wind) make a valid parameter value cause control state deviation.

Evaluation

- Target Control Programs
  - ArduPilot 3.5
  - PX4 1.8
- Simulator
  - APM simulator for ArduPilot
  - Gazebo for PX4
- GCS Program
  - QGroundControl for ArduPilot
  - MAVProxy for PX4

Bug Classification

- Range Implementation Bugs
  - When the attacker can set out-of-range parameters.

- Range Specification Bugs
  - When the bug occurs, even though the parameters are in the specified valid range.

Evaluation

Module	Sub-module	ArduPilot		PX4	
		RIB	RSB	RIB	RSB
Controller	x, y-axis position	1	0	1	1
	x, y-axis velocity	2	1	1	1
	z-axis position	1	0	1	1
	z-axis velocity	1	0	1	0
	z-axis acceleration	3	0	0	0
	Roll angle	1	0	1	1
	Roll angular rate	5	0	3	3
	Pitch angle	1	0	1	1
	Pitch angular rate	5	0	3	3
	Yaw angle	1	0	2	2
	Yaw angular rate	6	0	3	3
	Motor	0	0	3	3
Sensor	Inertia sensor	3	3	0	0
Mission	x, y-axis velocity	1	1	2	0
	z-axis velocity	2	0	4	0
	z-axis acceleration	2	0	0	0
	Roll, Pitch	1	1	1	1
Total	-	36	6	27	20

- Total **89** bugs
- **87** 0-day bugs
- Confirmed **8** bugs
- Patched **7** bugs

- RIB : Range Implementation Bugs
- RSB : Range Specification Bugs

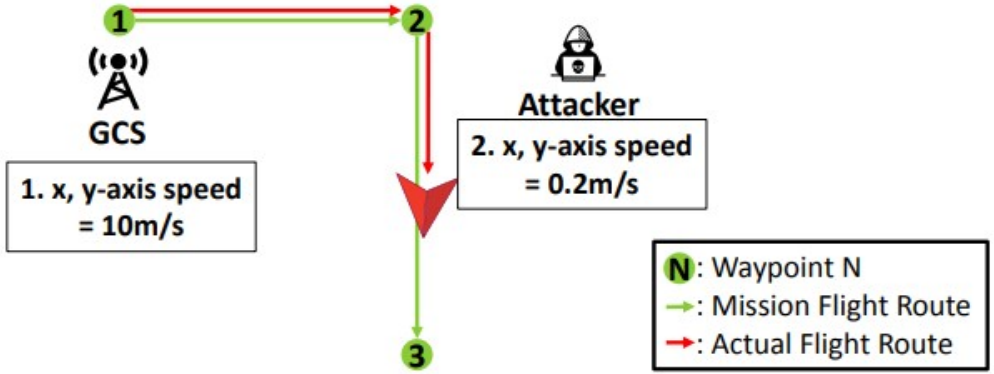
Evaluation

Control Program Module	Parameter	Physical Impacts			
		C	D	U	S
Controller	PSC_POSXY_P	✓			✓
	PSC_VELXY_P	✓	✓	✓	
	PSC_VELXY_I		✓	✓	
	PSC_POSZ_P				✓
	PSC_VELZ_P	✓			
	PSC_ACCZ_P	✓			✓
	PSC_ACCZ_I	✓	✓	✓	
	PSC_ACCZ_D	✓	✓	✓	
	ATC_ANG_RLL_P	✓			
	ATC_RAT_RLL_I	✓			
	ATC_RAT_RLL_IMAX	✓			✓
	ATC_RAT_RLL_D	✓			
	ATC_RAT_RLL_P	✓		✓	
	ATC_RAT_RLL_FF	✓		✓	
	ATC_ANG_PIT_P	✓			
	ATC_RAT_PIT_P	✓		✓	
	ATC_RAT_PIT_I	✓			
	ATC_RAT_PIT_IMAX	✓			
	ATC_RAT_PIT_D	✓			✓
	ATC_RAT_PIT_FF	✓		✓	✓
	ATC_ANG_YAW_P	✓			
	ATC_SLEW_YAW			✓	
	ATC_RAT_YAW_P			✓	
	ATC_RAT_YAW_I			✓	
	ATC_RAT_YAW_IMAX				✓
	ATC_RAT_YAW_D	✓			✓
	ATC_RAT_YAW_FF	✓		✓	
Sensor	INS_POS1_Z	✓		✓	
	INS_POS2_Z	✓		✓	
	INS_POS3_Z	✓		✓	
Mission	WPNAV_SPEED				✓
	WPNAV_SPEED_UP				✓
	WPNAV_SPEED_DN				✓
	WPNAV_ACCEL	✓			✓
	WPNAV_ACCEL_Z	✓			✓
	ANGLE_MAX	✓			✓

- In ArduPilot
  - 27 parameters can cause crash
  - 4 parameters can cause deviation form trajectory
  - 15 parameters can cause unstable movement
  - 14 parameters can cause stuck in certain location or speed

Case Study 1

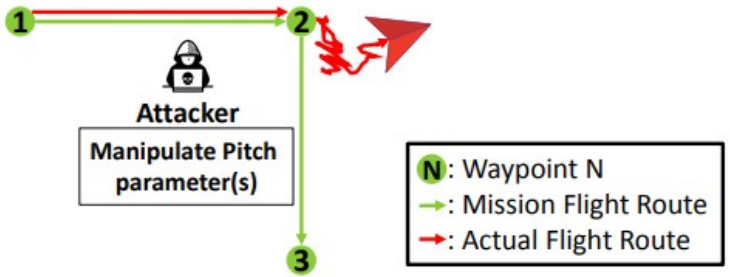
- Found by one-dimensional mutation
- Unrecoverable Slowdown
  - If the attacker sets the mission speed to 0.2m/s in (2), the mission speed does not go back to 10m/s
- Cause by input validation bug on mission speed change routine




```
1  #define WPNAV_WP_SPEED_MIN 100  //Buggy code 2
2  #define WPNAV_WP_SPEED_MIN 20   //Patched code 2
3  ...
4  void AC_WPNav::set_speed_xy(float speed_cms){
5  -if(_wp_speed_cms>=WPNAV_WP_SPEED_MIN){  //Buggy code 1
6  +if(speed_cms>=WPNAV_WP_SPEED_MIN){     //Patched code 1
7      _wp_speed_cms = speed_cms;
8      _pos_control.set_speed_xy(_wp_speed_cms);
9      ...
```

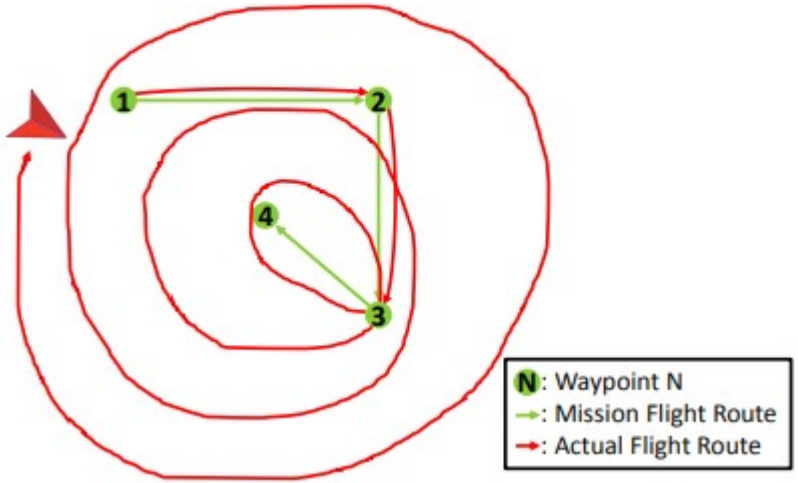
Case Study 2

- Found by multi-dimensional mutation
- Oscillation route and crash
  - By manipulating four pitch control parameters
- Cause by inter-dependency between parameters



Case Study 3

- Found by wind condition mutation
- Diverging route
  - Under strong wind condition
  - By sending a command to set the maximum tilting angle to a low value
- Cause by sharp turn, not enough time to change parameter, strong wind



Summary

- Designed RVFuzzer to find input validation bugs in control programs of RVs
- Found 89 input validation bugs using RVFuzzer
- Showed that Input validation bugs can cause even deviation or crash of RV

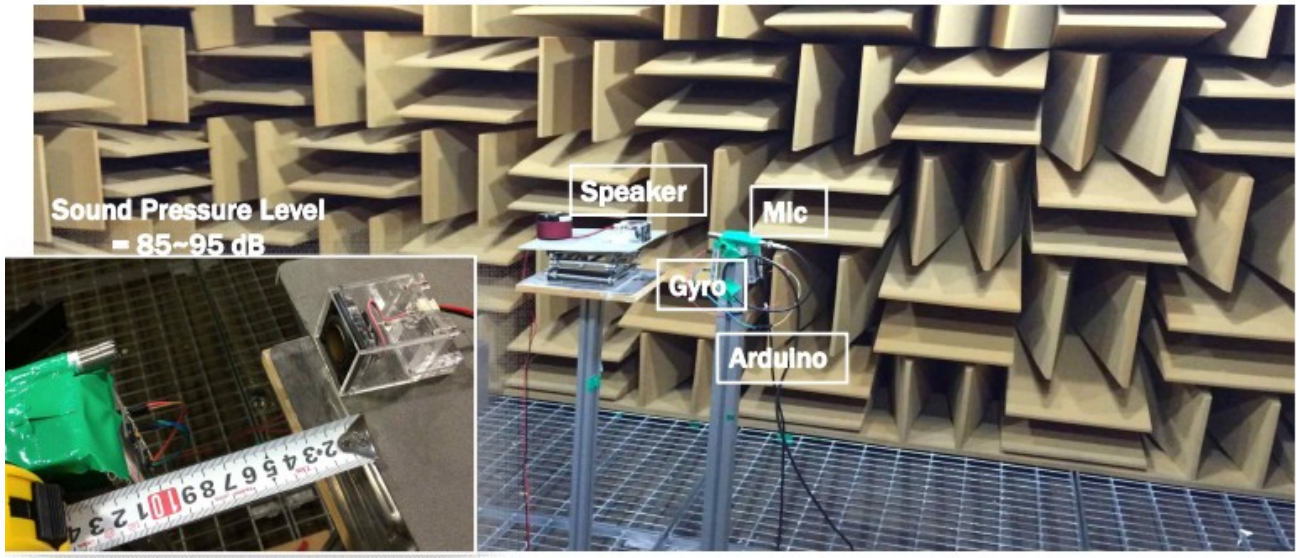
Conclusion

- It is meaningful that a new technique for testing input validation bugs in RVs was proposed.

- Attack model is too strong

Related Works

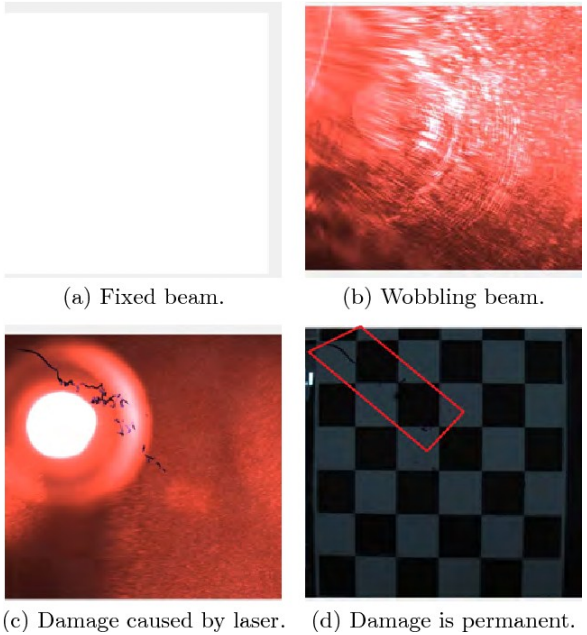
- Traditional attacks
  - Rocking drones with intentional sound noise on gyroscopic sensors



Related Works

- Traditional attacks
  - Can You Trust Autonomous Vehicles: Contactless Attacks against Sensors of Self-driving Vehicle

(a) Normal. (b) Spoofed. (c) Jammed.

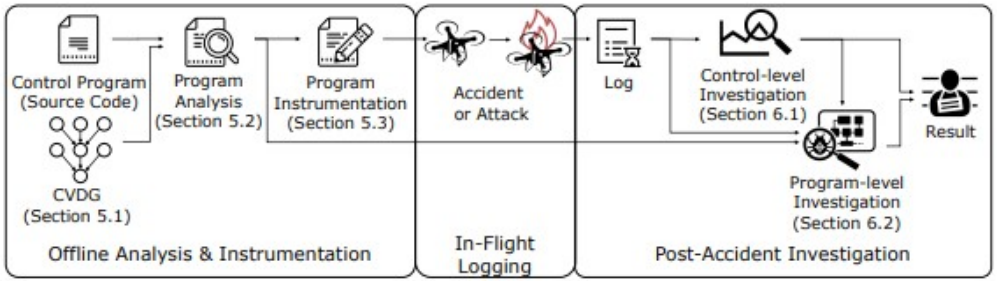


(a) Fixed beam. (b) Wobbling beam.

(c) Damage caused by laser. (d) Damage is permanent.

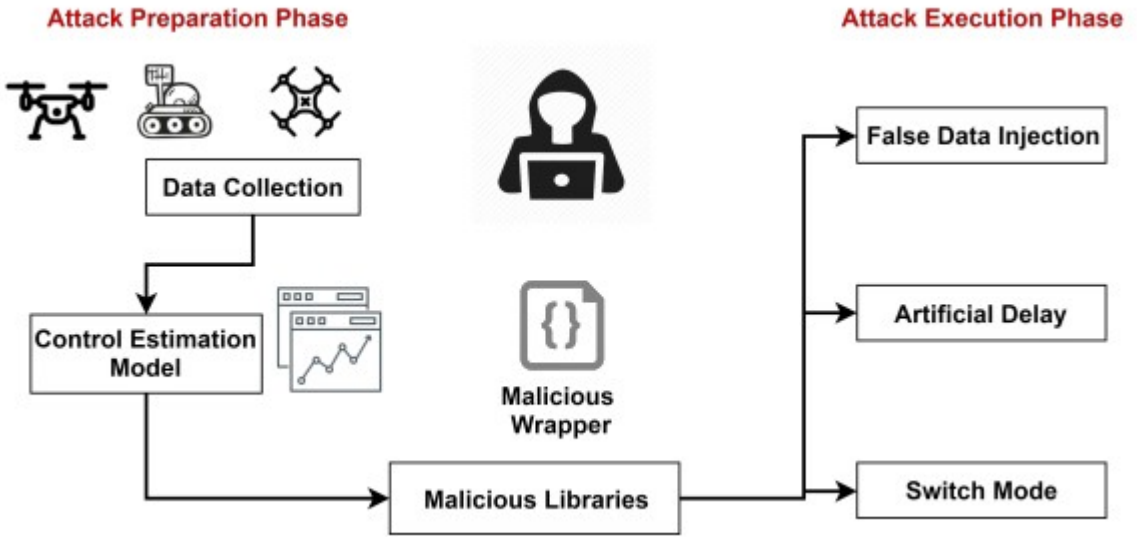
Future Works

- Control-semantic bugs
  - From Control Model to Program: Investigating Robotic Aerial Vehicle Accidents with MayDay



Future Works

- Small footprint
  - Out of control: stealthy attacks against robotic vehicles protected by control-based techniques



Q&A

Q&A - Best Questions

- Seunghyun Lee : Although the authors justify their attack model, it still isn't really convincing

- Jaehyun Ha : Can the fuzzers also discover or define the "correct behaviour" of the RVs?

- Taeung Yoon : The paper mentions environmental factors impacting the validity of input ranges. Could this impact be minimized in the software design phase, and how might that be achieved?

Thank you