Network and Distributed System Security (NDSS) Symposium
February 24, 2026, San Diego, California, USA

Beyond Conventional Triggers: Auto-Contextualized Covert Triggers for Android Logic Bombs

Ye Wang, Bo Luo, and Fengjun Li

Department of EECS
The University of Kansas

AGENDA

01

Logic Bombs and Hidden Triggers

02

SensorBomb Framework

03

Prototypes and Attack Performance Analysis

04

Evasion Evaluations and Conclusion

Logic Bombs and Their Influence

- Logic bombs represent a class of condition-triggered malicious code that remains dormant until specific conditions are met

- Real-world Impact: Logic bombs have caused significant damage to different infrastructures

Android Logic Bombs and Existing Detection

- Logic bombs keep evolving by leveraging new trigger types to evade detection
- Both dynamic and static analysis have evolved accordingly to detect new triggers
- Now empowered by ML models, they can detect a wide range of potential triggers
- Academic attention to the topic has declined

Trigger Types

- Common conditions
- System events/property
- Context-aware property
- Communication/signal

Dynamic Fuzzing

- Botnet Detection [2008]
- Droidfuzzer [MoMM 2013]
- Dynodroid [FSE 2013]
- Intellidroid [NDSS 2016]

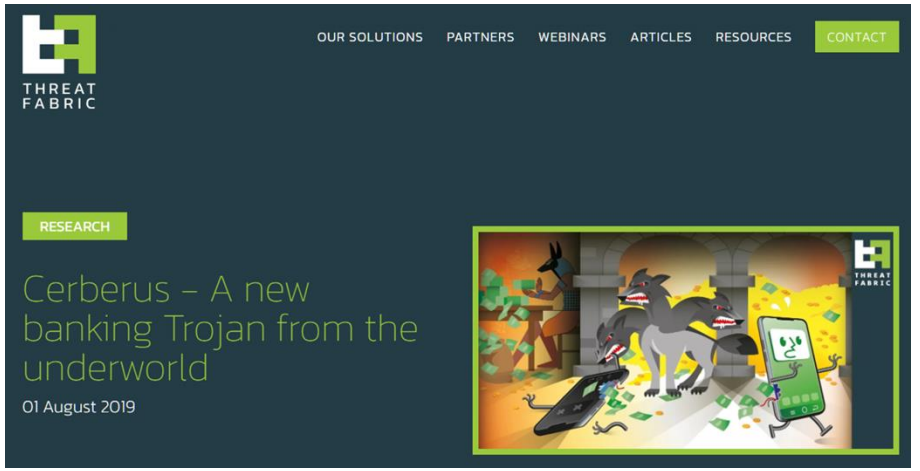
Static Analysis

- TriggerScope [S&P 2016]
- HSOMiner [NDSS 2017]
- BareCloud [USENIX SEC 2014]
- InputScope [S&P 2020]
- Difuzer [ICSE 2022]

Sensor-based Logic Bomb

- Some sensor-based logic bombs had evaded static analysis and had been released.

RQ1: how can sensor-based logic bomb evade static analysis reliably?

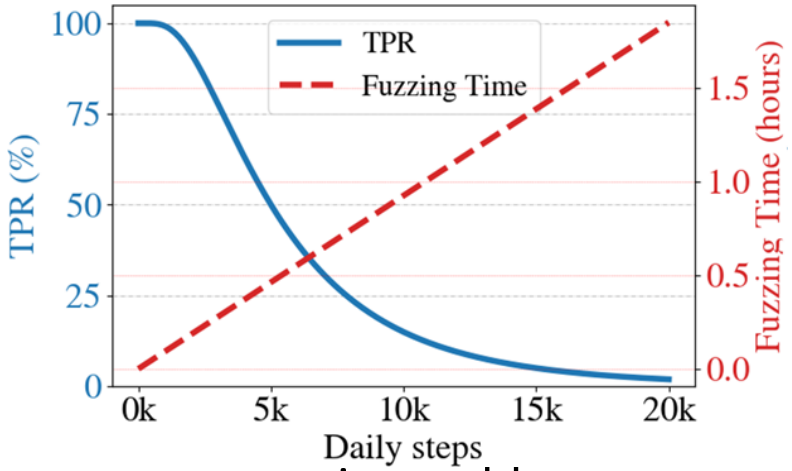


- We tested common sensors and sensitive operations they triggered by state-of-the-art logic bomb static detector Difuzer [1], some of which can indeed evade Difuzer.

HSO	contacts	calls	photos	location	microphone	network	Bluetooth	SMS
Accelerometer	√	√	√	√	√	√	√	√
Camera	√	X	√	√	√	√	X	X
Motion sensor	X	X	√	X	√	√	√	X

[1] J. Samhi, L. Li, T. F. Bissyandé, and J. Klein, "Difuzer: Uncovering suspicious hidden sensitive operations in android apps," in Proceedings of the 44th International Conference on Software Engineering, 2022, pp. 723–735.

Sensor-based Logic Bomb



- Sensor reading-based logic bomb
  - It is easy to fuzz
    - Sensors always have limited reading range
  - Low success rate
    - In the real world, attackers always satisfy success rate to remain stealthy

RQ2: how to leverage controlled dynamic-pattern-based trigger to be resistant to fuzzing test while keeping high trigger success rate

- Sensor-state-based anomaly detection has been proposed to detect all attacks leveraging sensors (e,g., 6thsense [1] and Sbtddl [2])

RQ3: how to evade sensor-state-based anomaly detection

[1] K. Sikder, H. Aksu, and A. S. Uluagac, "{6thSense}: A context aware sensor-based attack detector for smart devices," in 26th USENIX Security Symposium, 2017, pp. 397–414.

[2] S. Manimaran, V. Sastry, and N. Gopalan, "Sbtddl: A novel framework for sensor-based threats detection on android smartphones using deep learning," Computers & Security, vol. 118, p. 102729, 2022.

Auto-Contextualization

Device Contextualization for controlled dynamic-pattern-based trigger

Static Contextualization for static detection evasion

RQ1: resistant to static analysis

- Mimic legitimate sensor use

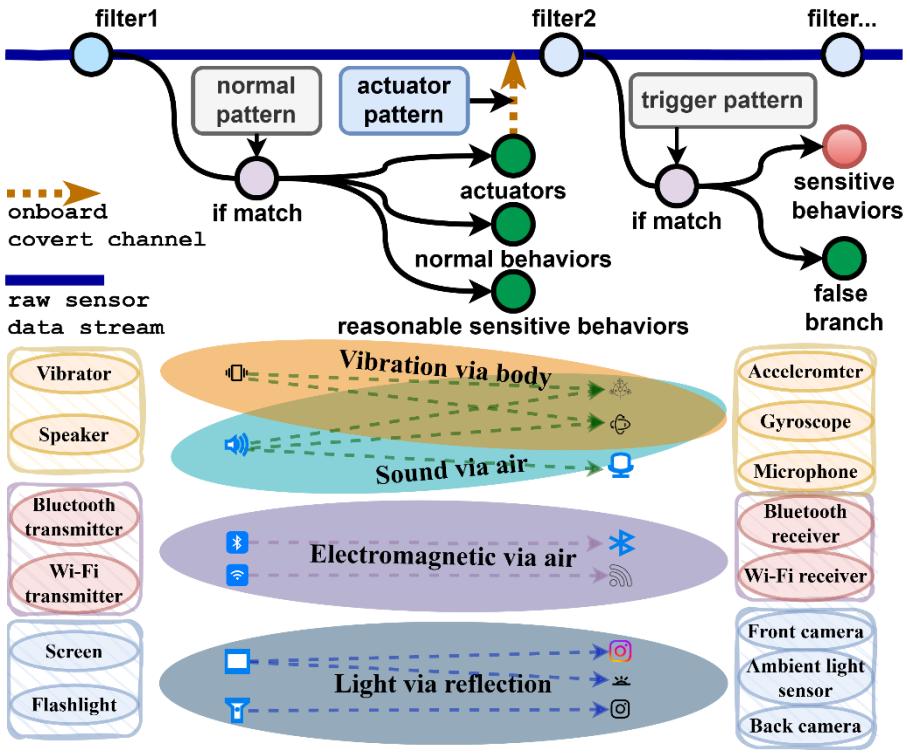
RQ2: controlled dynamic-pattern-based trigger

- Onboard actuators
- Dynamic trigger patterns
- Onboard sensor-actuator covert channel discovery

RQ3: evade sensor-state-based anomaly detection

- Measure actuator's influence on sensor readings
- Mimic legitimate actuator behavior

Behavior Contextualization for anomaly detection evasion



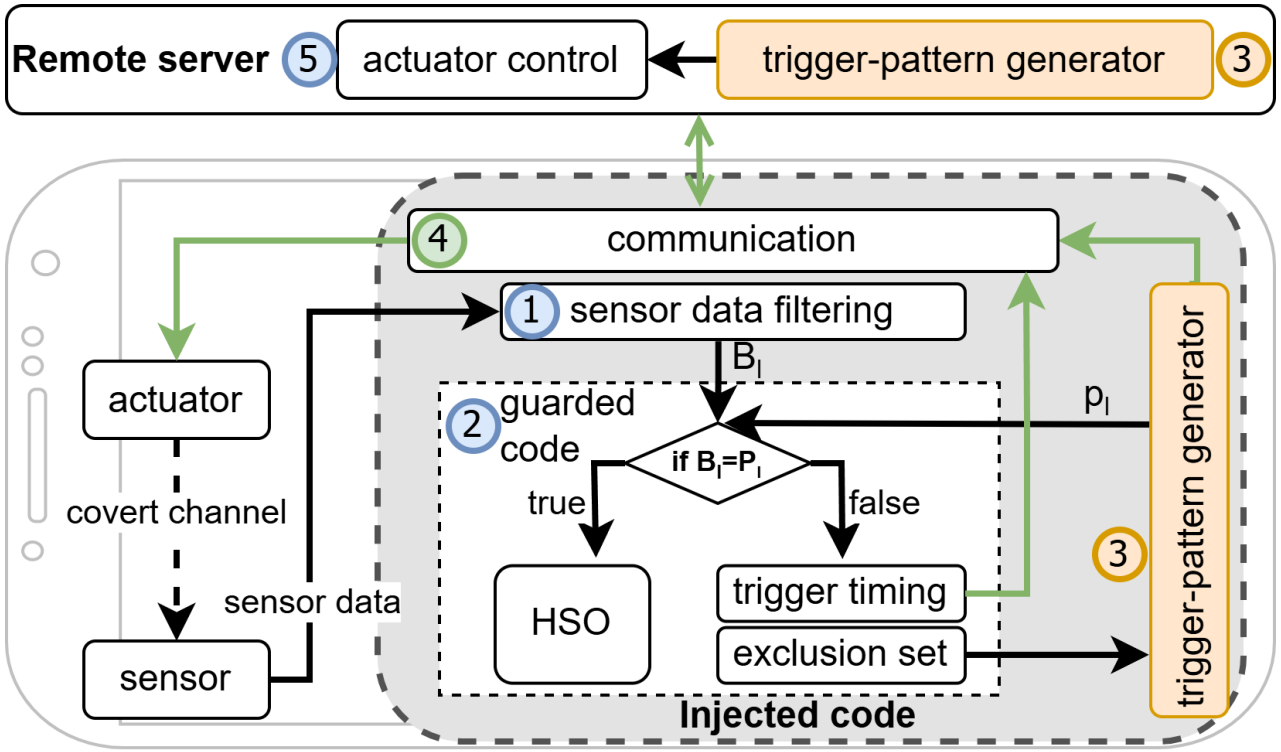
BPF and threshold-based filtering

Category	Trigger Condition	Source / Mechanism	Vibration Pattern	Semantic Intent
Fitness & Tracking	Goal achieved (steps, distance)	Background metric threshold	Short celebratory multi-pulse	Positive reinforcement
	Lap / segment completed	Distance/time threshold event	Single medium pulse	Progress milestone feedback
	Start / pause / resume / stop	Foreground UI action	Short pulse / dual-tap (pause)	Mode transition acknowledgment
	HR zone / arrhythmia alert	HR sensor + rule evaluation	Repeated short pulses	Physiological safety warning
	Inactivity reminder	Step counter + periodic timer	Soft gentle pulses	Behavioral nudging

Auto-Contextualization Components

1 Device Contextualization for Data Filtering

- Automatically configure the parameters of the onboard covert channels and the filters.
- Based on device specifications and knowledge base
- Knowledge base: parameters of existing covert channels through different media



Parameters:

- band-pass filter BPF(f)
- time window W(Δt)
- threshold θ

Example:

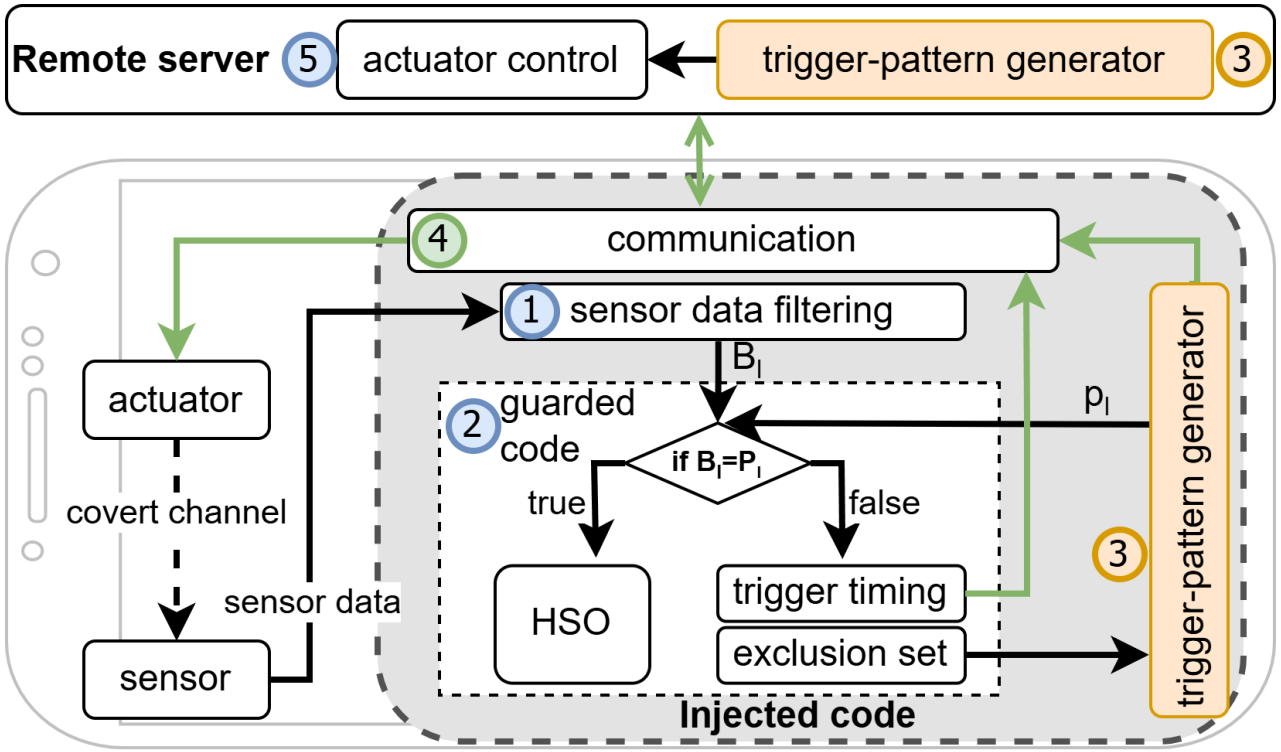
The vibration pattern [100, 200, 200, 100]

-> Binary codes 100110

Auto-Contextualization Components

2 Host App Contextualization for Guarded Code Generation

- Automatically select the sensitive methods that can evade static analysis.
- Based on the host app category, sensor type, and knowledge base.
- Infer real-time sensor status and statistical patterns.

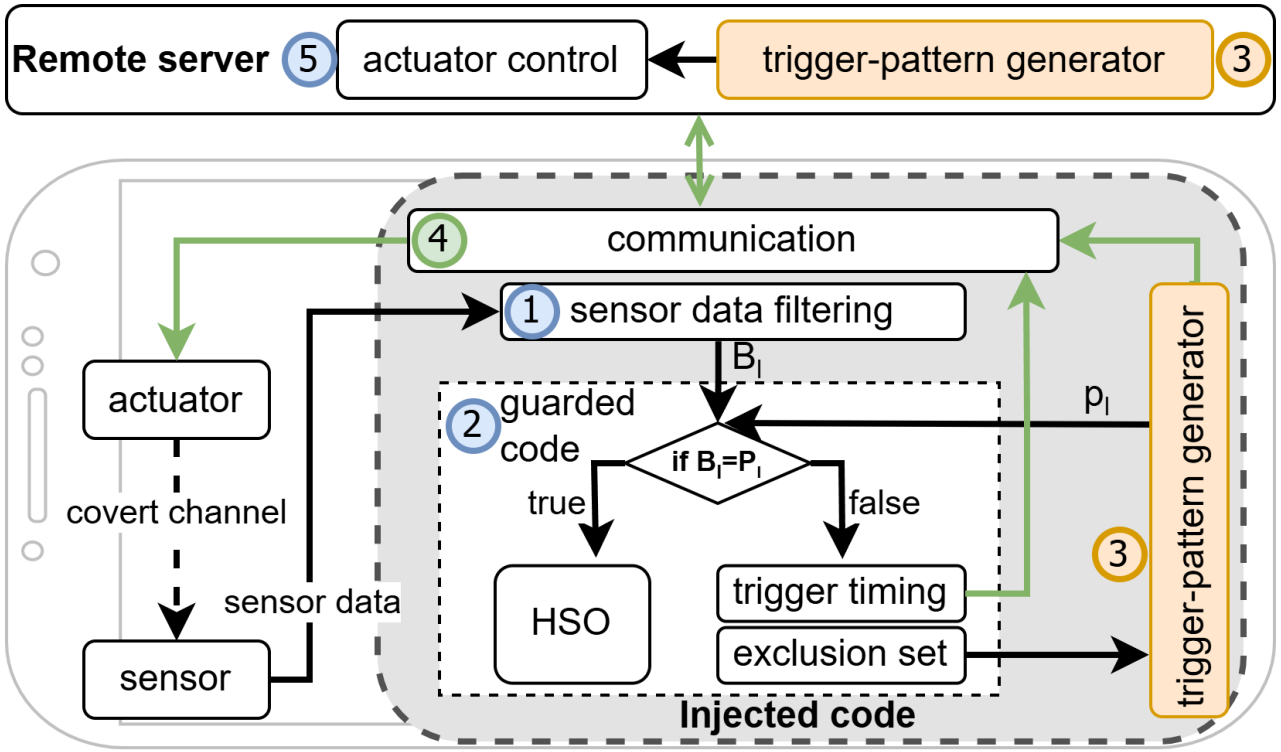


- **Static evasion**
  - Alignment: sensitive methods, number of app methods, number of sensitive methods
- **Trigger pattern exclusion**
  - Record all mismatch triggers -> patterns generated in daily behaviors
- **Optimal timing**
  - When all observed bits are zero -> no interference

Auto-Contextualization Components

3 Pattern Generation

- Optimal pattern length balancing transmission errors, false triggers, and fuzzing space
- Generate dynamic pattern to evade fuzzing test based on records
- Mimic legitimate actuator behaviors to evade anomaly detection



Optimal Pattern Length (l):

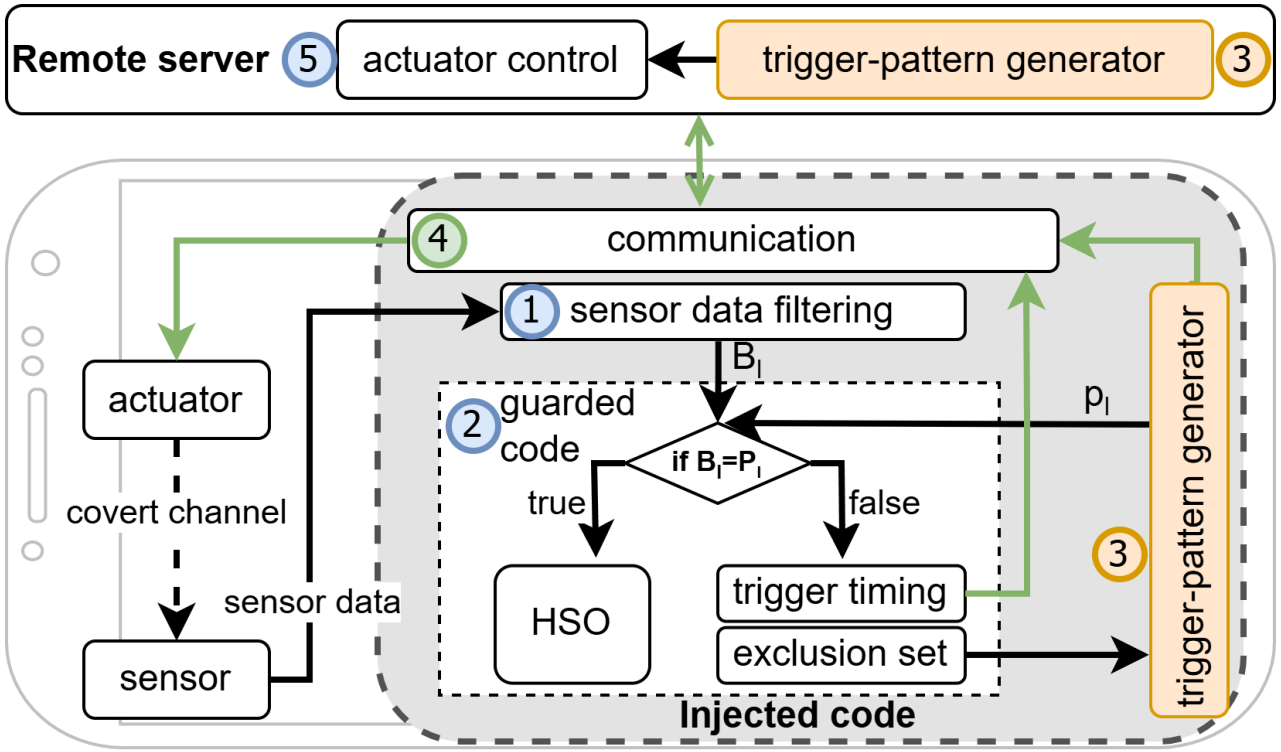
$$l^* = \arg\max_{l \in \mathcal{L}} \; w_d\,(l \ln 2) - w_f\, e^{-\alpha l} - w_e\left[1-(1-p)^l\right]$$

- p is the channel BER
- α is the exponential decay rate
- w_d, w_f, and w_e are weights
- $L = [l_{min}, l_{max}]$ denotes the soft boundary

Auto-Contextualization Components

4 Communication

- Device to server
  - Trigger timing and exclusion patterns
- Server to device
  - Command to control the actuators



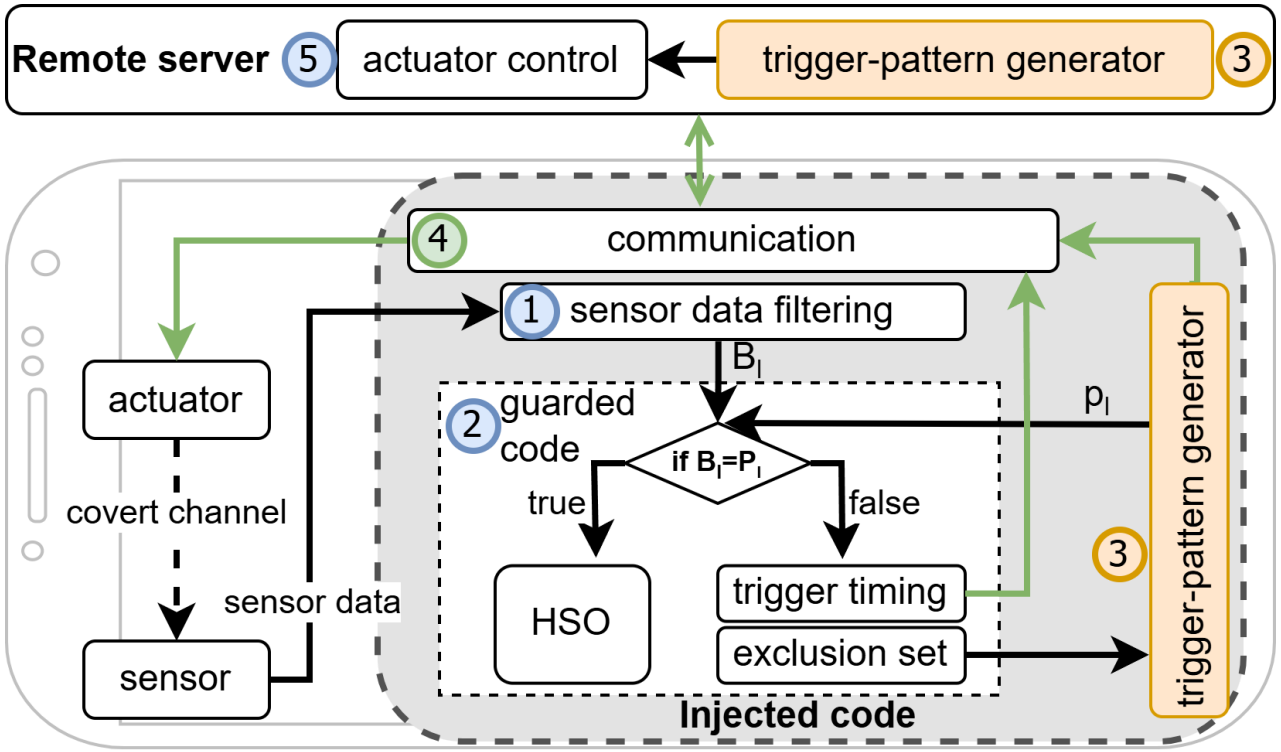
All traffic is transparent, and the content is not suspicious

- **Control actuators:**
  - JavaScript API or media content
- **Uploaded information:**
  - Plaintext and a small exclusion pattern set

Auto-Contextualization Components

5 Actuator Control

- Manage contextual and behavioral alignment to avoid user suspicion



- Contextual alignment
- Behavior alignment
- Controlled pattern variation

Example:

When a fit app meets an exercise goal, vibration notifications happen with a familiar pattern.

Evaluation Objectives

- Automated injection of SensorBombs and its robustness
- Attack success rate and false-trigger rate
  - Zero FTR and high ASR
- Resistance against static analysis, dynamic fuzzing test, and anomaly detection
  - High evasion rates

Evaluation Settings

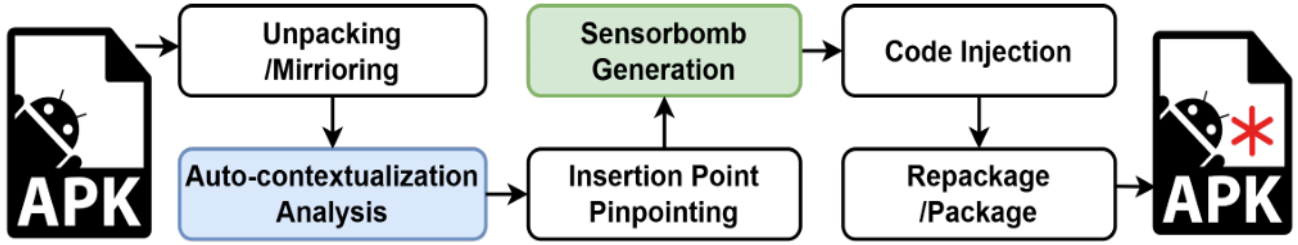
1. **Devices:** smartphones from 5 vendors with 7 models

Category	Brand	Model	API ver.	Max screen brightness	Screen size	Ratio	Vibrator axis
Low-end	Xiaomi	Mi 8	29	600 nits	6.2 in	2.08	Z-axis
	Samsung	S9	28	630 nits	5.8 in	2.06	Z-axis
	OnePlus	10T	30	950 nits	6.7 in	2.23	X-axis
	Motorola	Edge+	32	1000 nits	6.7 in	2.22	X-axis
Flagship	Samsung	S23+	33	1750 nits	6.6 in	2.17	X-axis
	Samsung	S24+	34	2600 nits	6.7 in	2.17	X-axis
	Vivo	x200	35	4500 nits	6.7 in	2.22	X-axis

2. **Large-scale injection:** injecting 3 types of triggers into 1,729 APKs from AndroZoo
3. **Prototypes:** self-developed Android apps based on three covert-channels
4. **Automatic testing: open-source** software, pipeline
5. **Manual testing:** 15 volunteers (college students under IRB)

Large-Scale Automated Injection

- **Injection pipeline**
  - APK filtering: 1,729 out of 5,000 APKs
  - Injecting using AndroBomb [1]

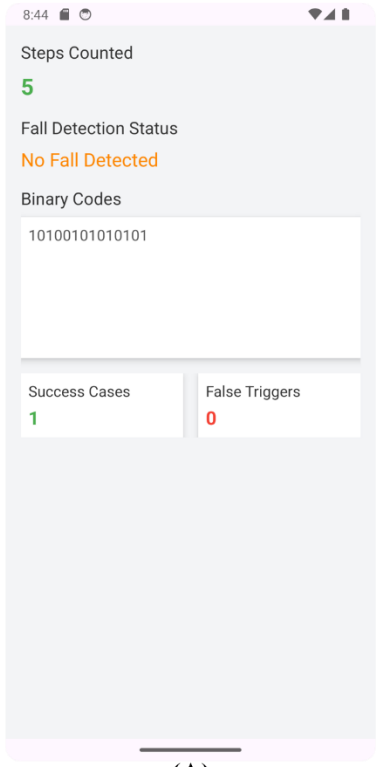


- **Testing pipeline**
  - Automatically testing: ADB-based script
  - Manual functionality checking
- **Results**
  - Successfully injected triggers into 1,403 APKs, and all passed ADB-based testing
  - Randomly selected 149 apps for manual checking and all passed the tests

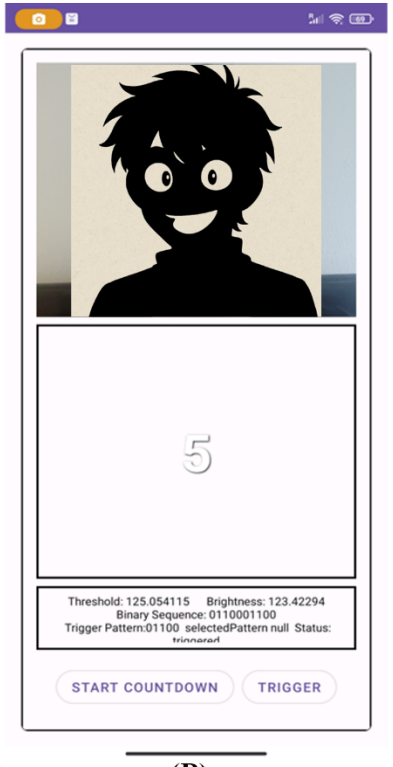
[1] J. Samhi, "AndroBomb (1.0)," 2022. [Online]. Available: https: //doi.org/10.5281/zenodo.5907924

Prototypes

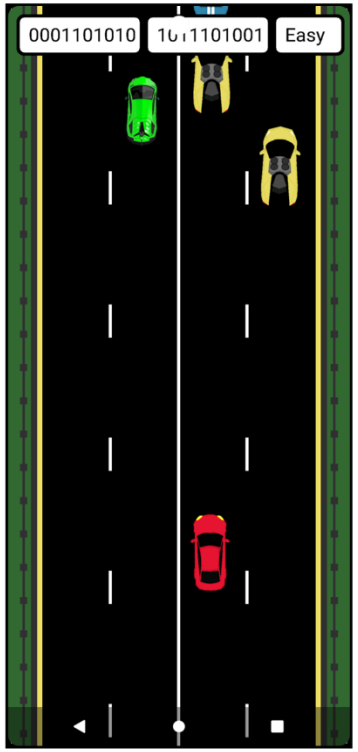
- **Accelerometer Bomb (A)**
  - Broadest attack class
- **Camera Bomb (B)**
  - Novel onboard covert channel
- **User Bomb (C)**
  - Comparison with existing real-world user behavior-based triggers



(A)



(B)



(C)

SensorBomb	Host App	Sensor Data Filtering			Actuator Logic		Trigger Enhancement	
		f	Δt	θ	actuator	behavior logic	l	timing
Accelerometer bomb	step count & fall detection	$80-100Hz$	$>20ms$	fixed	vibrator	notification	$<17bits$	stable
Camera bomb	selfie	$1Hz$	$1s$	optimal	screen	counting down	3, 5, 10	low ambient light
User bomb	car racing game	$50Hz$	on difficulty level	automatic	user	user playing	unlimited	playing

Attack Success Rate and False-Trigger Rate

- **Accelerometer bomb experiment**
  - **12 common usage scenarios [1]:**

	Static Usage Scenarios						Moving Usage Scenarios					
Behavior	Stand	Sit	Talk-Sit	Lay	Stand-Sit	Pick	Jump	Push-up	Sit-Up	Run	Stair-Up	Table-tennis
Duration	1 min	1 min	1 min	1 min	1 min	1 min	1 min	1 min	1 min	1 min	1 min	1 min
Trigger /test times	10/10 (100%)	10/10 (100%)	10/10 (100%)	10/10 (100%)	9/10 (100%)	10/10 (100%)	0 (-)	0 (-)	0 (-)	0 (-)	0 (-)	0 (-)
False trigger	0/hour	0/hour	0/hour	0/hour	0/hour	0/hour	0/hour	0/hour	0/hour	0/hour	0/hour	0/hour

  - **Results:**
    - 100% ASR due to accurate trigger timing
    - 0% false triggers due to statistics collection of false trigger patterns

[1] N. Sikder and A.-A. Nahid, "Ku-har: An open dataset for heterogeneous human activity recognition," Pattern Recognition Letters, vol. 146, pp. 46–54, 2021.

Additional Tests

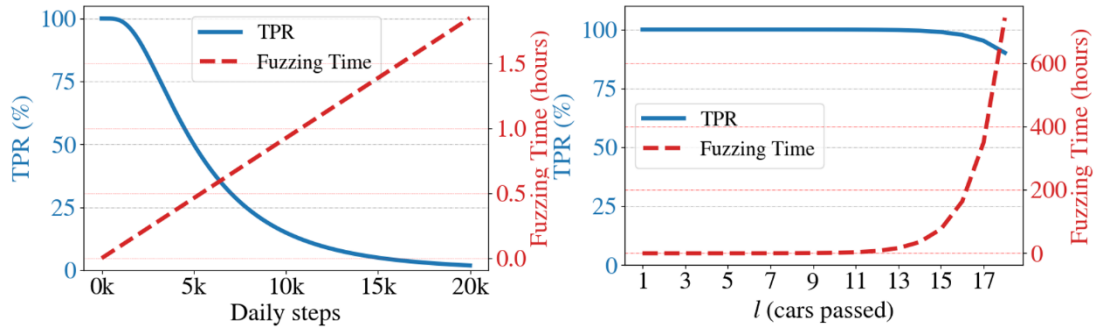
- **Camera and User bomb experiments**
  - **Camera bomb:** 5 common usage scenarios :

Scenarios		Dim bedroom	Normal light bedroom	Bright living room	Daylight office	Sunlight
Times/TPR	Low-end	120/100%	108/95.4%	8/87.5%	0/-	0/-
	Flagship	10/100%	10/100%	0/-	0/-	0/-
FPR	All	0%	0%	0%	0%	0%

  - **Results:**

    100% ASR and 0% false trigger under dim light conditions

  - **User bomb:** 15 volunteers playing a car racing game :



  - **Results:**
    - 700 hours fuzzing time while 90% success rate

Resistance to Detection

- **Evasion from static detector Difuzer [1]**
  - Y means selected by SensorBomb,
  - N means not;
  - ✓ means successfully evade Difuzer
  - × means not.

HSO	contacts	calls	photos	location	microphone	network	bluetooth	SMS
Acc bomb	Y | ✓	Y | ✓	Y | ✓	Y | ✓	Y | ✓	Y | ✓	Y | ✓	Y | ✓
Cam bomb	Y | ✓	N | ×	Y | ✓	Y | ✓	Y | ✓	Y | ✓	N | ×	N | ×
User bomb	N | ×	N | ×	Y | ✓	N | ×	Y | ✓	Y | ✓	Y | ✓	N | ×

- Results:
  - 100% evasion rate

[1] J. Samhi, L. Li, T. F. Bissyandé, and J. Klein, "Difuzer: Uncovering suspicious hidden sensitive operations in android apps," in Proceedings of the 44th International Conference on Software Engineering, 2022, pp. 723–735.

Sensor Data Fuzzing Evasion

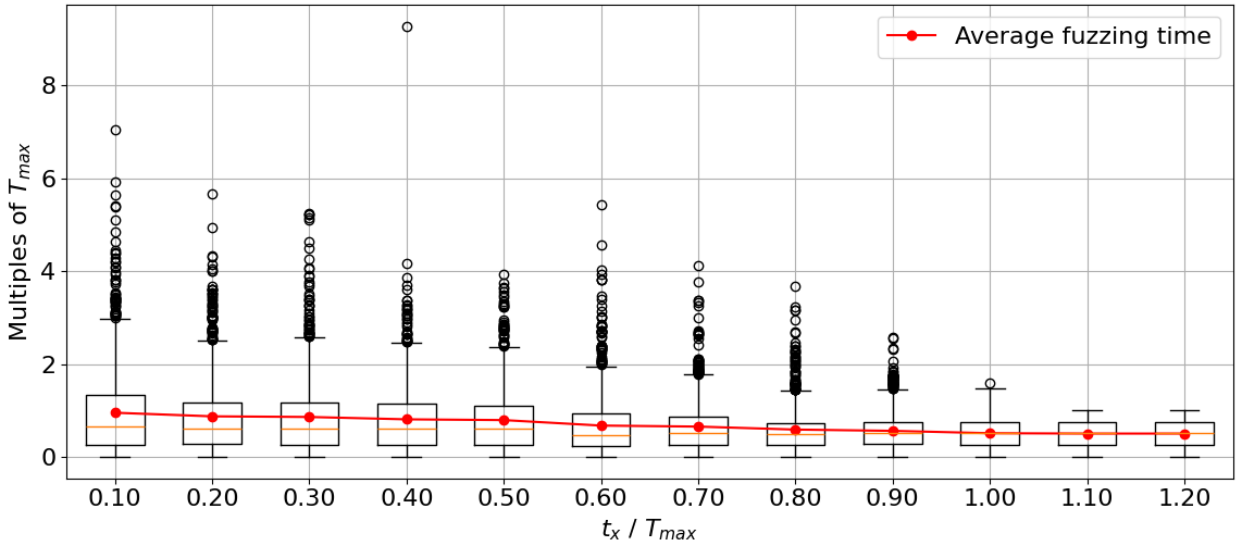
- **Theoretical Estimation**
  - We adopt a data-sweeping strategy to cover the full parameter space
  - SensorBomb variants > 100 hours
  - Sensor-reading variants ~2 hours

SensorBomb	n_θ	n_f	l (bits)	Δt (s)	T_{max}
Acc bomb	fixed	fixed	13–16	0.02–0.06	104.2 hours
Cam bomb	2	20	3, 5, 10	1	115.8 hours
User bomb	fixed	fixed	16	0.5	145.6 hours
Sensor value	**range**		**scale**	**sampling rate**	T_{max}
Ambient light [61]	0–188,000 lux		0.045 lux	50 Hz	2.32 hours
Step count [18]	0–20,000 steps		1 step	3 steps / s	1.85 hours

- **Experimental Results**
  - SensorBomb variants:

    They need more fuzzing time than theoretical estimation
  - Sensor-reading variants:

    They match the theoretical estimation



Anomaly Detection Evasion

- **Evasion from sensor-status based anomaly detection SBTDDL [1]**
  - LSTM detector
  - 12 common usage scenarios [2]:

	Static Usage Scenarios						Moving Usage Scenarios					
Behavior	Stand	Sit	Talk-Sit	Lay	Stand-Sit	Pick	Jump	Push-up	Sit-Up	Run	Stair-Up	Table-tennis
Evasion times/rate	10/10 (100%)	10/10 (100%)	10/10 (100%)	10/10 (100%)	9/10 (100%)	10/10 (100%)	0 (-)	0 (-)	0 (-)	0 (-)	0 (-)	0 (-)

  - Results:
    - 100% evasion rate

[1] S. Manimaran, V. Sastry, and N. Gopalan, "Sbtddl: A novel framework for sensor-based threats detection on android smartphones using deep learning," Computers & Security, vol. 118, p. 102729, 2022.

[2] N. Sikder and A.-A. Nahid, "Ku-har: An open dataset for heterogeneous human activity recognition," Pattern Recognition Letters, vol. 146, pp. 46–54, 2021.

Conclusion

- **Auto-Contextualized Triggers:** Novel approach that dynamically adapts to device and application context, moving logic bomb design from leveraging new triggers to deep hiding by Auto-Contextualization
- **Comprehensive Evasion:** Successfully bypasses static analysis, anomaly detection, and dynamic fuzzing test
- **Practical Validation:** Three working prototypes and large-scale injection test demonstrate real-world feasibility
- **Systematic study:** To our knowledge, it is the first systematic study of sensor-based logic bomb evasion

Thank You!

Read our paper:

Personal website:

Potential Countermeasures

- **Actuator Regulation**
- A whitelist to allow only actuator patterns found in legitimate apps
  - 98% adopted patterns consistent with common patterns
  - Fuzzing may be employed to detect the trigger
  - The attacker may still combine multiple benign patterns as a trigger
  - Pattern-based whitelist is changing for multimedia-related actuators. Such as screens and speakers.
  - Limit the device output levels.

Out of Scope Detection

- **Repackaging Detection**
  - Signature verification, integrity check.
- **Signature-Based Detection**
  - Large-scale injection will leave footprint
- **Communication Content**
  - Transmission of a small set of binary codes
  - The set is very small, e.g., 138 12-bit binary code in accelerometer bomb

Static Analysis Tools

- **TriggerScope [1]:**
  - Limited to predefined triggers (time, location, SMS)
  - High false positive rate when adding sensors to the trigger types
- **HSOMiner [2]:**
  - No public code and dataset
- **Difuzer:**
  - Open-source code

[1] JordanSamhi, “Tsopen,” https://github.com/JordanSamhi/TSOpen, 2021, accessed: 2025-12-12.

[2] X. Pan, X. Wang, Y. Duan, X. Wang, and H. Yin, “Dark hazard: Learning-based, large-scale discovery of hidden sensitive operations in android apps.” in NDSS, 2017.