# Unleash the Power: Non-Invasive On-Chip Malware Detection in Heterogeneous IoT Systems by Leveraging Side-Channels

Anonymous Author(s)

***Abstract*—As heterogeneous systems become more common and diverse in IoT and CPS settings, securing these systems against malware has become a daunting task. To combat this, real-time hardware and/or (hardware-)software malware detection has gained popularity. *Hardware* malware detectors are effective but often require invasive changes to the CPU, hence limiting their usefulness in diverse settings. *Software* methods are non-invasive but often come with large performance overheads and/or disruptions to the main functionality of the device.**

**This poster proposes SideGuard, a new, non-invasive approach for detecting malware by analyzing the system's internal power consumption. With a tailored power sensor, our method utilizes this measured power consumption signal as a stand-in for program behavior. It collects training data, understanding how signals should appear in different program sections during proper execution. It then monitors execution, identifying instances where the observed signal deviates from the expected ones. For monitoring, the crucial idea is to *indirectly* measure power using customized sensors on an embedded FPGA or co-processor common in modern heterogeneous IoT systems. Notably, the monitoring unit (e.g., embedded FPGA) doesn't need a direct CPU connection but simply shares the power source, offering a *key advantage*: the malware detection unit requires no CPU changes, resulting in zero performance, power, and area overhead for the main CPU. Implementing this idea requires addressing *several new challenges* compared to prior work. Specifically, we introduce a new software-signal processing *co-design* approach. Results show that our approach can achieve $>$95% accuracy in detecting real-world malware. As heterogeneous IoT systems become more common, we believe our method is a strong contender for securing future hardware systems.**

## I. Introduction

Computing systems, especially embedded and smart IoT systems, are increasingly targeted by malware [1]. There are various methods to detect malware. Among them, hardware malware detectors (HMD) have gained attention in recent years [2]–[4].

HMDs typically function by incorporating two key elements into the system: *monitoring logic* that gathers hardware-level information (e.g., performance counters) about the application, and a *classifier* to identify potential malware and anomalies. While highly effective, the main drawback of HMDs is the need for invasive changes to the device's CPU and/or its underlying system. Despite being feasible for many systems, it is not suitable or even possible for others, especially those already in use, older systems, and custom heterogeneous systems in which the system integrator lacks control over the internal design of each hardware component and can only add/remove components. Given the shift toward more *heterogeneity*, where various components including CPUs, FPGAs, and sensors are integrated on the same device/chip, there is a *strong demand* for techniques that can match the capabilities and performance of HMDs without invasive modifications.

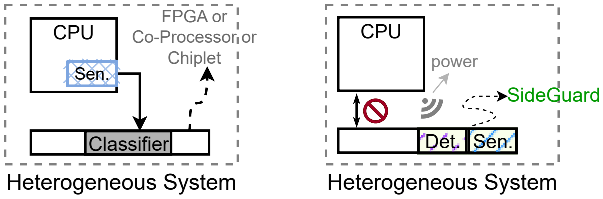


Fig. 1. Conventional hardware malware detectors (left) vs. our proposed method (right). Instead of collecting performance counters from the CPU and using a classifier, our method indirectly measures power consumption. No connection between the monitor and CPU is needed.

To address this demand, we propose a new on-device *non-invasive* malware detection method called SideGuard. The key insight is leveraging on-chip power side-channel as a means to *indirectly* monitor the system, and using a new detection algorithm that utilizes this data to detect anomalies. As shown in Figure 1, instead of collecting hardware-related features from the CPU, SideGuard indirectly measures the power consumption (i.e., a side-channel signal) of the CPU using customized sensors ("Sen") implemented as a *separate* component. This data is then used by our detection algorithm ("Det.") also implemented on the same unit, collectively creating an on-device malware detection unit.

The ***key advantage*** of this method is that it doesn't require any hardware support from the CPU or any connection to it, unlike current HMDs. This feature broadens its suitability for various systems with SoCs. In such setups, the malware detection unit can reside in a distinct component (an FPGA) or as a separate IP (in an SoC). Additionally, as the detection module is physically separated from the CPU, it creates an "air gap," further enhancing robustness.

Designing and implementing SideGuard involves ***several new contributions***. First, although the utilization of on-chip power sensors (e.g., ring oscillators) has been employed previously for side-channel attacks and hardware Trojan detection [5]–[9], this study stands out as the ***first*** to apply on-chip power consumption for *dynamic program monitoring* and consequently for detecting malware. Unlike earlier studies, dynamic program monitoring *introduces entirely new research inquiries*.

Second, creating SideGuard involves tackling *two new*

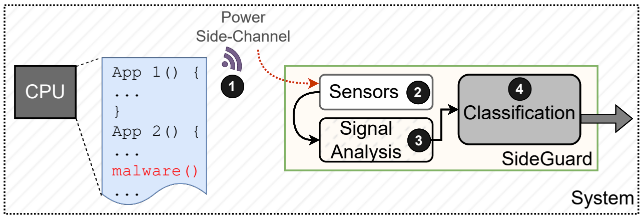


Fig. 2. Software running on a CPU creates unintentional power consumption fluctuations (i.e., power side-channels). SideGuard captures this information using an array of on-chip power sensors and leverages a signal analysis and detection algorithm to find anomalies.

*research challenges*. The initial one is devising and executing the malware detection algorithm. The fundamental distinction between our approach and current classifiers for malware detection lies in the *nature of our features*. Traditional HMDs utilize hardware performance counters, which offer a naturally discrete feature space. In contrast, our problem deals with a continuous time-series data feature space. Also, in *contrast to* previous methods that used on-chip power monitoring for detecting Trojans, our task (dynamic program monitoring) demands a significantly more intricate detection algorithm. The difficulty stems from the need to monitor a *diverse range of software applications and types of malware*. This is a departure from the monitoring of just a *single* application (such as cryptographic cores) and a *single* malicious behavior, as seen in Trojan detection techniques.

The next challenge involves the system and hardware design aspects of SideGuard. Two main questions need addressing here. Firstly, how to design the sensors and manage continuous data effectively to reduce storage usage? Secondly, considering the power and storage constraints on the device (especially for embedded and IoT devices) and the requirement for an advanced signal processing method to handle time-series data, how can a detection algorithm be implemented efficiently in terms of both area and power usage?

We systematically analyze the effectiveness of our detection framework using various malware on a real SoC system, a DE1-SoC board.

## II. System Design Overview

**Threat Model and Assumption.** We focus on malware detection for *heterogeneous* "smart" embedded/IoT devices such as robotic devices, medical devices, and smart home systems. We target devices equipped with a system-on-chip (SoC) and/or heterogenous 2.5D systems, comprising various IPs and/or chips/chiplets. These components include sensors, actuators, and processing elements where one or multiple cores are controlled by an operating system. We assume that our detection framework is implemented on the system using a hardware component such as an embedded FPGA (eFPGA) and/or a co-processor implemented as a separate IP and/or chip. It's worth noting that comparable assumptions were made in previous hardware malware detection frameworks, utilizing the eFPGA/co-processor to implement the classifier [2]–[4], [10]. Therefore, SideGuard doesn't introduce new hardware; instead, it suggests a method to *repurpose* the existing hardware.

We assume that the system is initially secure. The system, however, can get compromised as it starts executing various applications. Once it is compromised, the adversary controls the entire CPU and kernel OS. Furthermore, we assume that SideGuard and its underlying hardware (i.e., eFPGA) is part of the root-of-trust (RoT) and can only be re-programmed through a secure update. Further, the RoT is additionally protected from an adversary since the monitoring framework is physically separate from the CPU and not controlled by the OS (i.e., *air-gapped*). Providing this air-gap eliminates the possibility of the monitor being infected by the same attack vectors that have compromised the host system.

For detecting malicious activities, SideGuard doesn't possess a priori knowledge about the type of attack or its power signatures and detection solely depends on the signals gathered by the sensors during monitoring. Further, SideGuard always maintains accurate reference models for malware-free signatures. These models are stored internally and remain uncompromised. The models, however, can be updated through a secure update, if needed. Moreover, we assume that the adversary is familiar with the system and program(s), including any existing vulnerabilities, and can manipulate the system by sending random inputs.

**System Overview.** The high level design of SideGuard is shown in Figure 2. Internally, SideGuard consists of three main components: a sensor array, a signal analysis unit, and a classifier. We briefly explain each in the following.

The first component is the *on-chip power sensors* (❷). An essential feature of SideGuard is its complete non-invasiveness. Therefore, the sensors (power or other types) should not have direct connections to the CPU. Instead, their design should allow indirect monitoring of the CPU (when running different applications). To achieve this, we utilize a time-to-digital converter (TDC) primitive [8]. These sensors are capable of tracking alterations in power usage within the shared power distribution network (PDN) by sensing changes in the delay of a propagating signal through a chain of buffers or other logic, thereby capturing the behavior of various applications operating on the host CPU (❶). A primary challenge in our design is the balance between sensor circuit size and precision. While more sensors provide more accurate data, they also occupy additional space and consume more power. We opt for a design with 32-bit granularity.

The second element is a signal analysis module. The fundamental distinction between our problem and current HMD solutions is our analysis involving a continuous time-domain signal. Consequently, we present a novel signal analysis algorithm (❸ and ❹). In our design, a crucial strategy is adopting a co-design approach. This means tailoring our signal analysis strategy to match the observed behavior of the target software.

## References

[1] M. Antonakakis *et al.*, "Understanding the mirai botnet," in *26th {USENIX} security symposium*, 2017, pp. 1093–1110.

[2] J. Demme *et al.*, "On the feasibility of online malware detection with performance counters," *ACM SIGARCH computer architecture news*, vol. 41, no. 3, pp. 559–570, 2013.

[3] K. Basu *et al.*, "Preempt: Preempting malware by examining embedded processor traces," in *DAC, 2019*, 2019, pp. 1–6.

[4] K. N. Khasawneh *et al.*, "Rhmd: Evasion-resilient hardware malware detectors," in *MICRO, 2017*, 2017, pp. 315–327.

[5] I. Giechaskiel *et al.*, "C 3 apsule: Cross-fpga covert-channel attacks through power supply unit leakage," in *S&P, 2020*. IEEE, 2020, pp. 1728–1741.

[6] A. Boutros *et al.*, "Neighbors from hell: Voltage attacks against deep learning accelerators on multi-tenant fpgas," in *ICFPT, 2020*. IEEE, 2020, pp. 103–111.

[7] Z. Xie, S. Li, M. Ma, C.-C. Chang, J. Pan, Y. Chen, and J. Hu, "Deep: Developing extremely efficient runtime on-chip power meters," in *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, 2022, pp. 1–9.

[8] H. Ma *et al.*, "On-chip trust evaluation utilizing tdc-based parameter-adjustable security primitive," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 10, pp. 1985–1994, 2020.

[9] M. Zhao and G. E. Suh, "Fpga-based remote power side-channel attacks," in *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 229–244.

[10] C. Konstantinou *et al.*, "Hpc-based malware detectors actually work: Transition to practice after a decade of research," *IEEE Design & Test*, vol. 39, no. 4, pp. 23–32, 2022.

# Unleash the Power: Non-Invasive On-Chip Malware Detection in Heterogeneous IoT Systems by Leveraging Side-Channels

**Fatemeh Arkannezhad**, Pooya Aghanoury, Justin Feng, Hossein Khalili, Nader Sehatbakhsh

Secure System and Architectures (SysArch) Lab
ECE Departement - University of California, Los Angeles (UCLA)

## Background

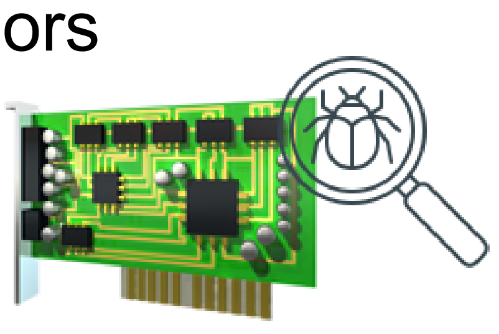
- Increasing Malware Threats
- Demand for Non-Invasive Solutions

Hardware Malware Detectors (HMD)

- Monitoring logic
- Classifier

Heterogeneous Systems

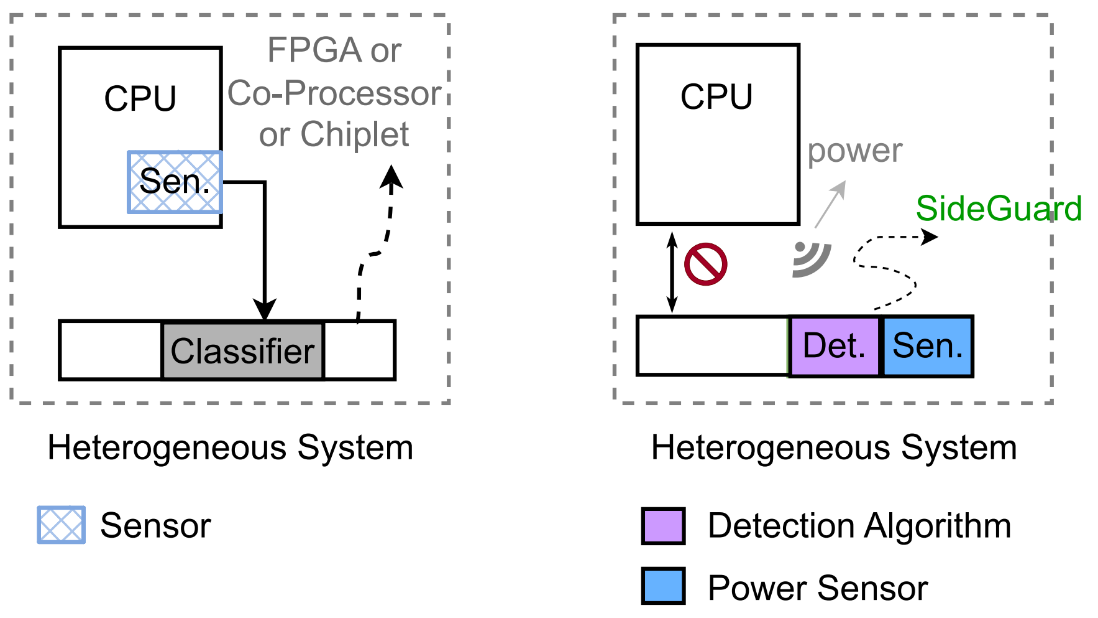
- CPUs
- FPGAs
- Sensors



## Problem

**Hardware Monitoring**

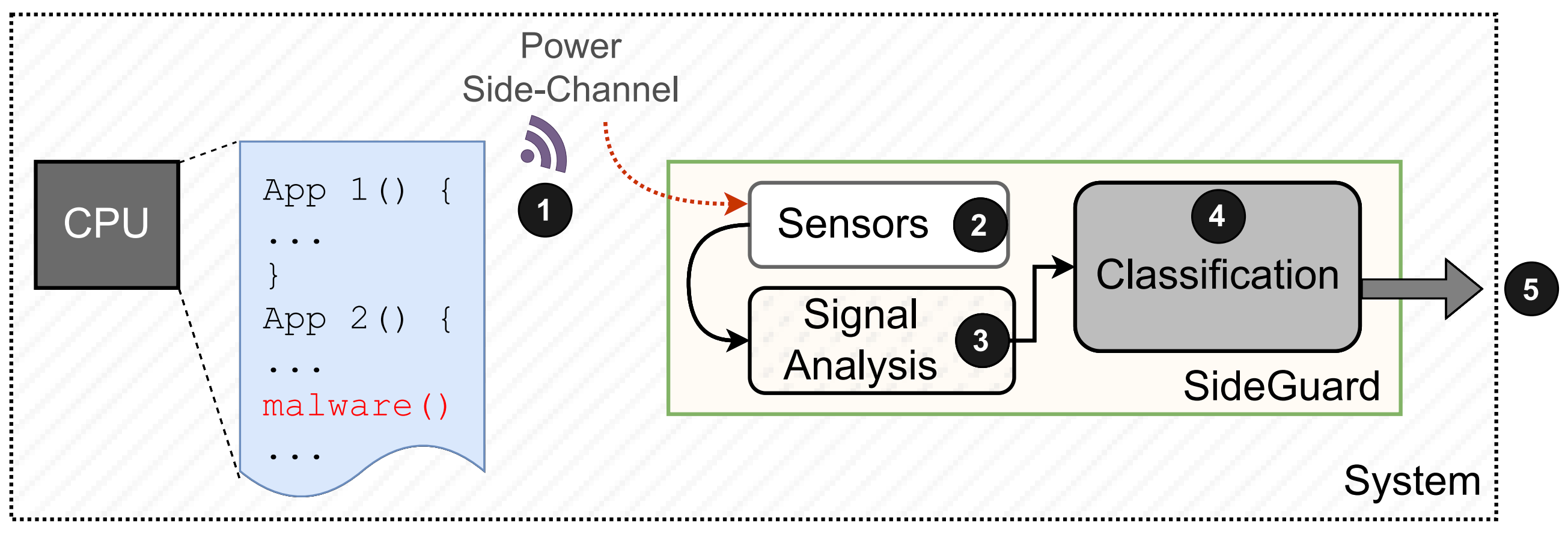
- SW-based
  - Performance Overhead
- HMD
  - Invasive Changes
- Limited Robustness
- Scalability



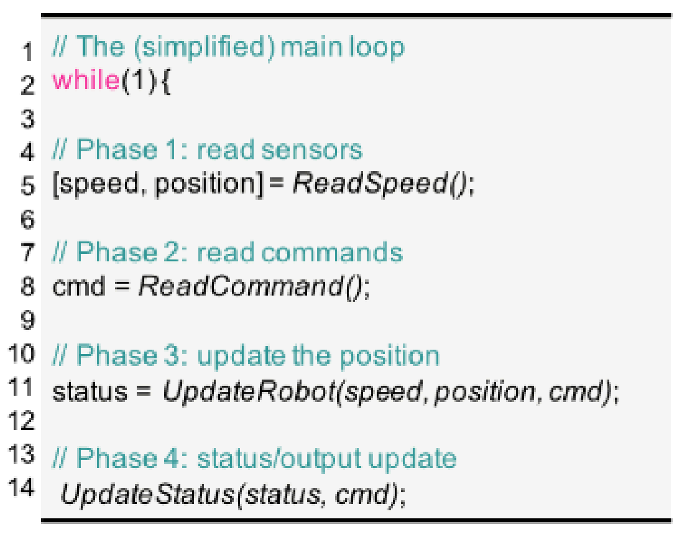
## Solution

**SIDEGUARD**

❶ Execute the CPU applications
❷ Collect on-chip Power Sensor signal
3,4 Analyze and Classify the signal
❺ Detects the malware

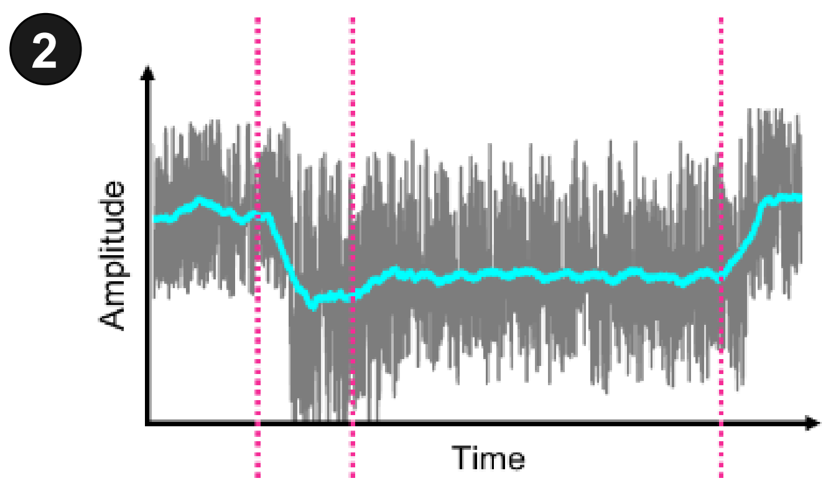


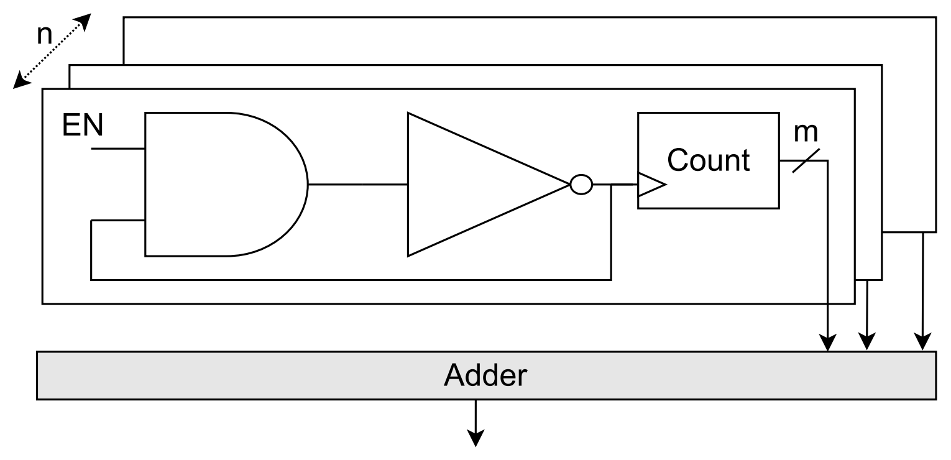
1



```
// The (simplified) main loop
while(1){

// Phase 1: read sensors
[speed, position] = ReadSpeed();

// Phase 2: read commands
cmd = ReadCommand();

// Phase 3: update the position
status = UpdateRobot(speed, position, cmd);

// Phase 4: status/output update
 UpdateStatus(status, cmd);
```

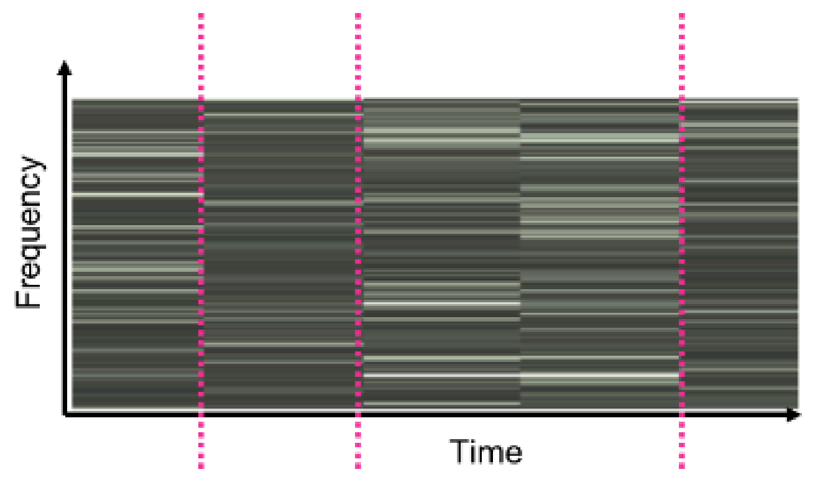
2



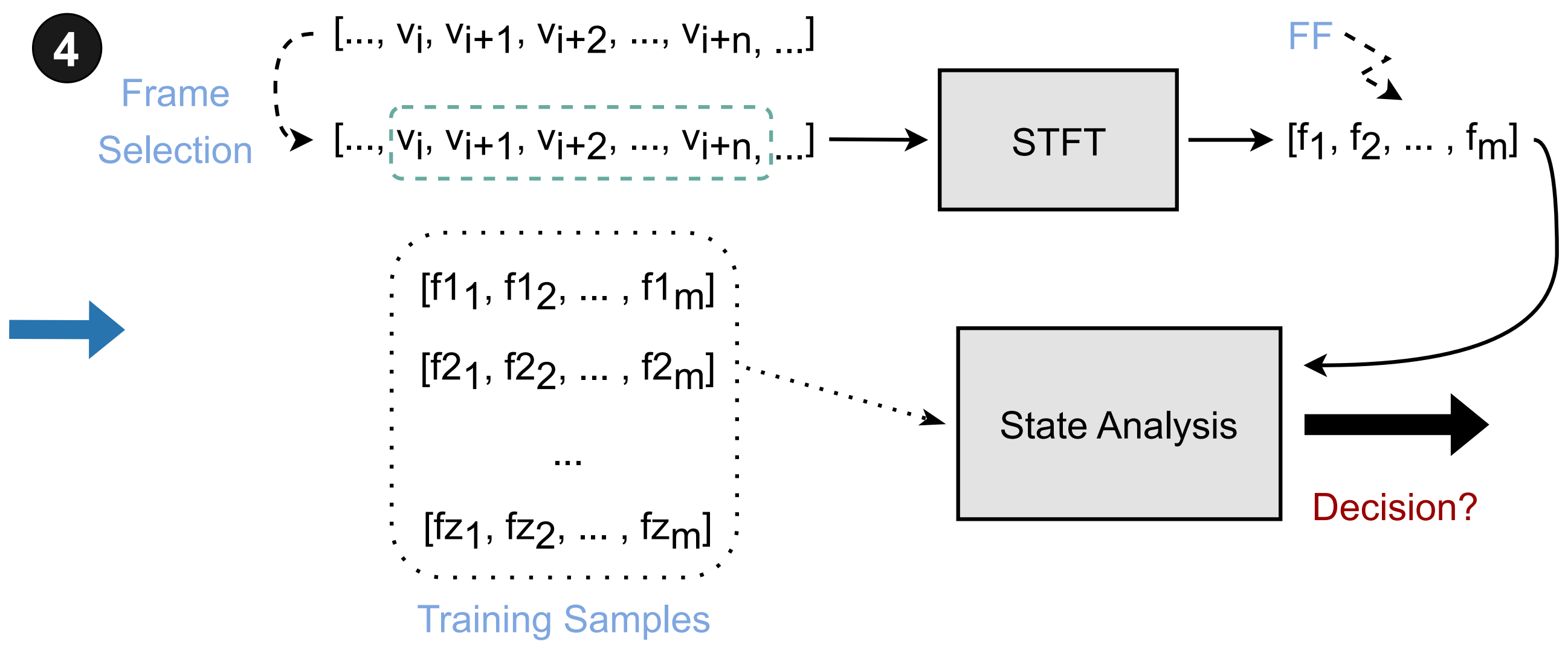


Power Sensor

3

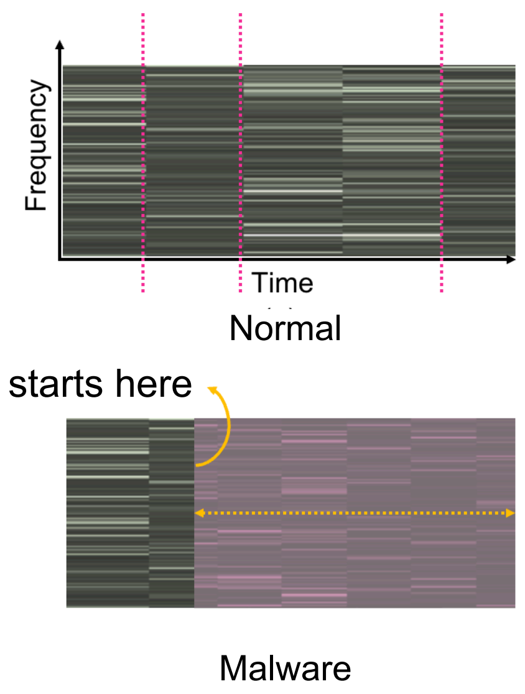


4


Frame Selection
[..., $v_i$, $v_{i+1}$, $v_{i+2}$, ..., $v_{i+n}$, ...]
[..., $v_i$, $v_{i+1}$, $v_{i+2}$, ..., $v_{i+n}$, ...]
STFT
FF
[$f_1$, $f_2$, ... , $f_m$]
[$f1_1$, $f1_2$, ... , $f1_m$]
[$f2_1$, $f2_2$, ... , $f2_m$]
...
[$fz_1$, $fz_2$, ... , $fz_m$]
Training Samples
State Analysis
Decision?

5

Detection Algorithm:
$\{FF_i\}_{i=t}^{t+L}$ is a region (of size L) if:
$\forall i \in \{t \leq i < t+l\} : Dist(FF_i, FF_{i+1}) < TH,$
Where $Dist(,)$ is defined as:
$Dist(FF_i, FF_{i+1}) = \sum_{j=1}^{K}(FF_{i_{sortA}}[j] - FF_{i+1_{sortA}}[j])^2/j.$



## Results

**MiBench:**

- Ransomware
- Mirai botnet

| Bitcount | | Susan | | FFT | | Basicmath | | qsort | |
|---|---|---|---|---|---|---|---|---|---|
| FP | FN | FP | FN | FP | FN | FP | FN | FP | FN |
| 1.7% | <0.1% | 4.1% | <0.1% | 3.5% | <0.1% | 2.9% | <0.1% | 3.1% | <0.1% |

Robustness

| Experiment | Average Change in Accuracy | |
|---|---|---|
| | FP | FN |
| Same-Type Board (DE1) | +2.7% | 0% |
| Different Board (Zynq) | +0.3% | 0% |

## Summary

**SIDEGUARD**

- Malware detection for IoT and embedded systems
- Non-Invasive Technique
  - Power sensors integrated into an embedded FPGA
- Efficient Detection
  - Real-time detection
  - No performance, power, or area overhead on the main CPU
- Robustness
  - Combines software and signal analysis
- Scalability

Contact:
https://ssysarch.ee.ucla.edu/
fatemeharkan@ucla.edu

IEEE/ACM Workshop on the Internet of Safe Things
45th IEEE Symposium on Security and Privacy
May 2024, San Francisco, California, USA