NoC-level Threat Monitoring in Domain-Specific Heterogeneous SoCs with SoCurity

Naorin Hossain*, Alper Buyuktosunoglu*, John-David Wellman*, Pradip Bose*, and Margaret Martonosi†

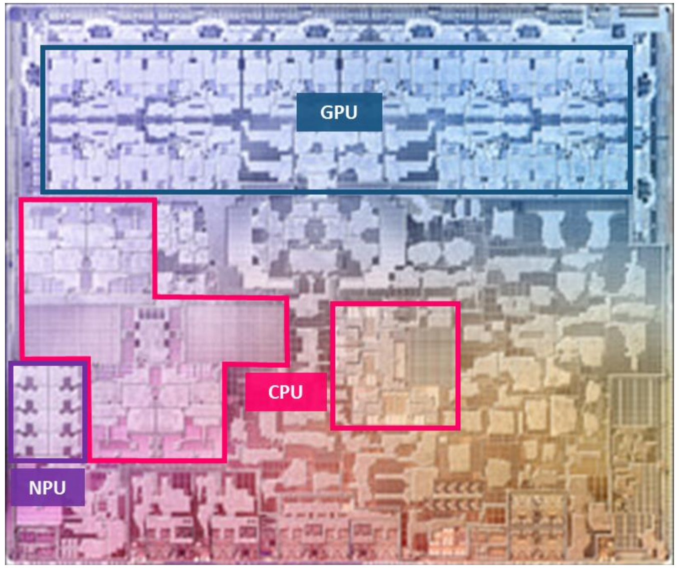
**IBM Research* *†Princeton University*

June 30, 2024

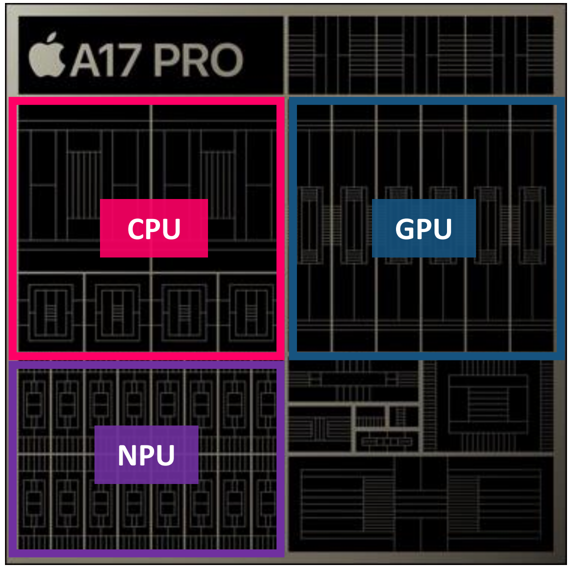
DOSSA-6

SoCs are everywhere – and are increasingly designed for specialized uses

Apple M3 SoC (2023 MacBook Pro, iMac)



Apple A17 Pro SoC (2023 iPhone 15 Pro)



Images from Apple

Tesla Full Self-Driving Chip (2019)

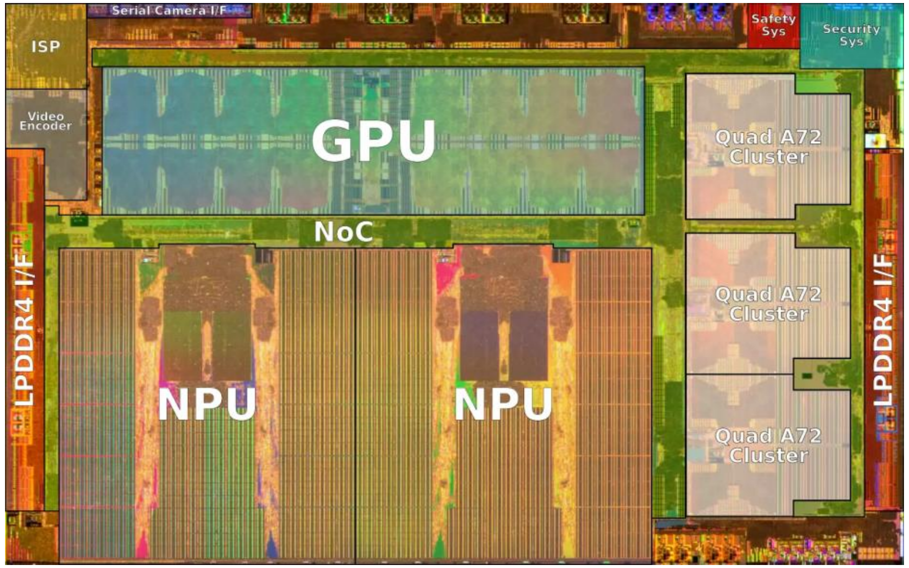
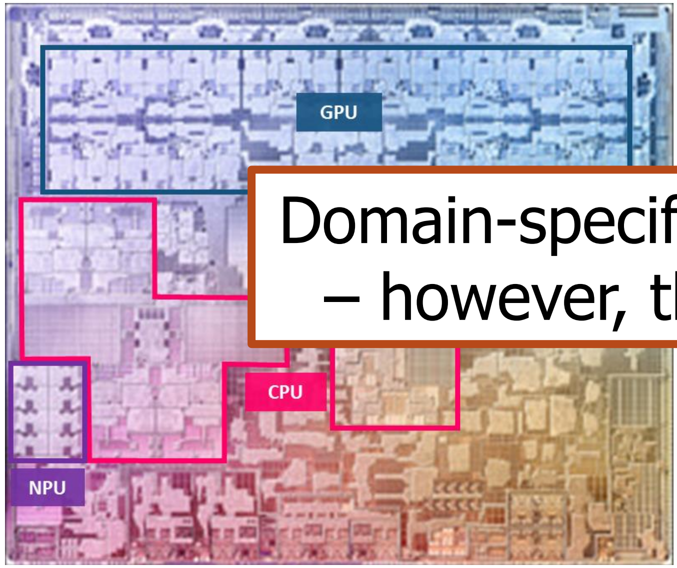


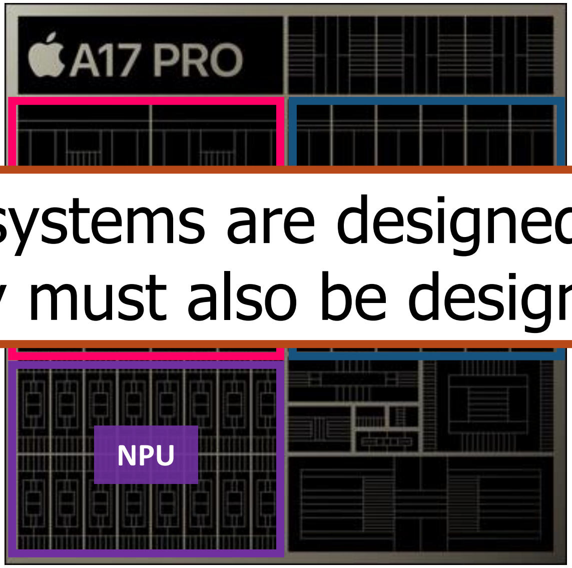
Image from WikiChip

SoCs are everywhere – and are increasingly designed for specialized uses

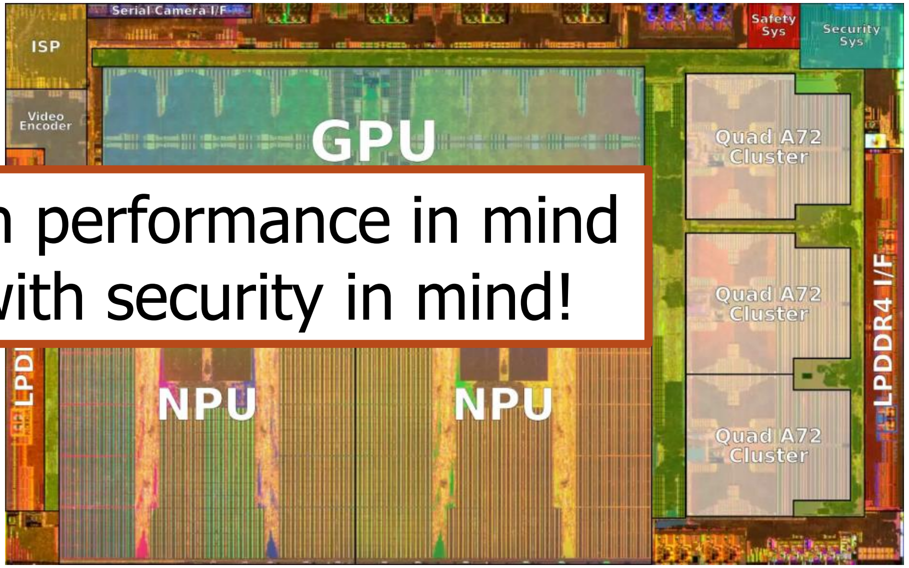
Apple M3 SoC (2023 MacBook Pro, iMac)



Apple A17 Pro SoC (2023 iPhone 15 Pro)



Tesla Full Self-Driving Chip (2019)



Images from Apple

Image from WikiChip

Domain-specific systems are designed with performance in mind – however, they must also be designed with security in mind!

Design complexity ↑
Security complexity ↑

Outline

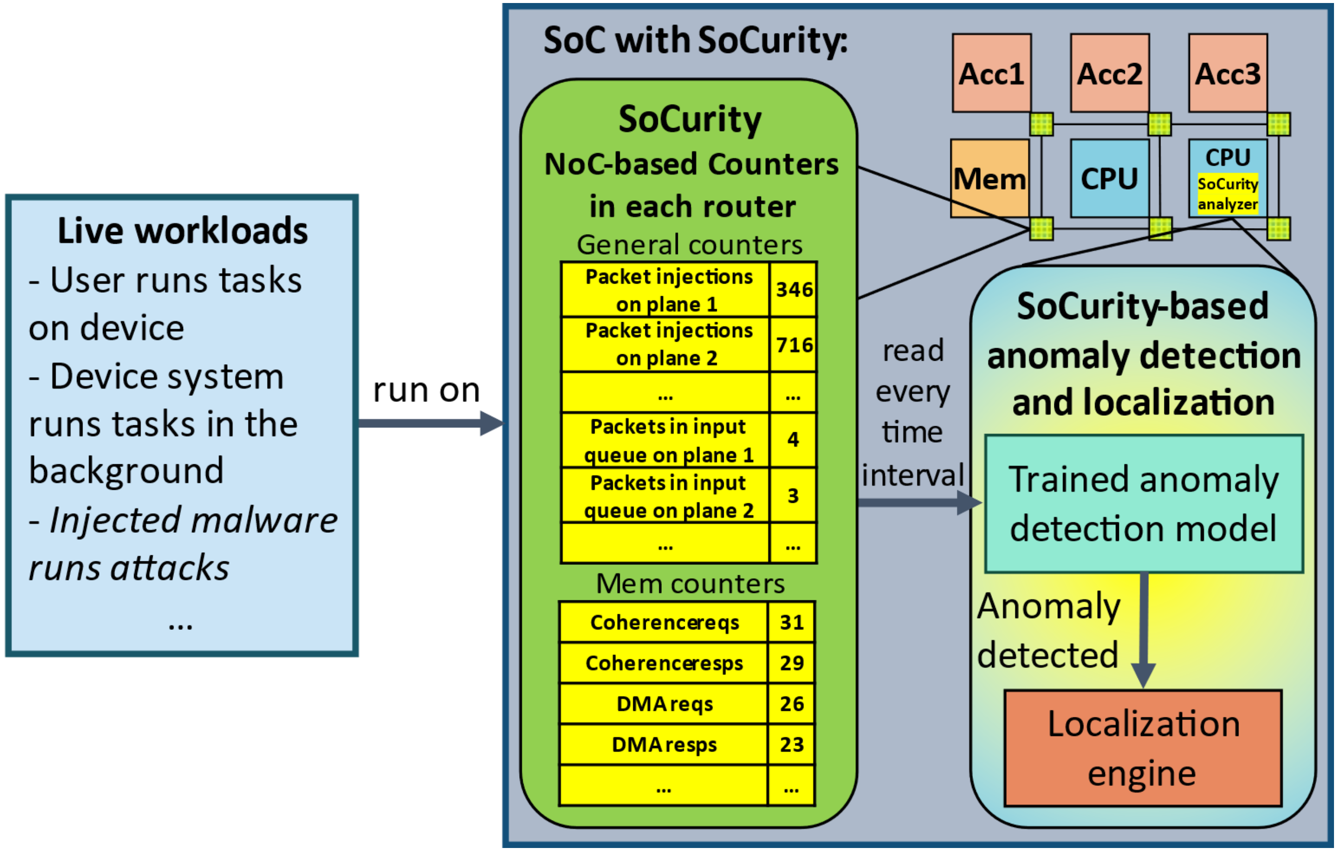
- Intro and motivation
- SoCurity overview
- Case study on connected autonomous vehicle SoCs
- Applying SoCurity to other threats and reliability
- Summary

Domain-specific heterogeneous SoCs can be compromised by availability attacks like DoS

- SoCs are specially designed to meet a device's performance needs. Performance gains are achieved by:
  - Parallelizing tasks across on-chip computing units
  - Speeding up computations with specialized hardware
- **Availability attacks:** reduce availability of system resources so they cannot be used for intended purpose
  - Can significantly slow down system, preventing realtime deadlines from being met
  - E.g., **denial-of-service (DoS) attacks** which "deny" access to available services by flooding system with requests to occupy the targeted services
- **Goal:** Build systems with security in mind to defend against such attacks.

We used the ESP framework to monitor **NoC-based hardware counters** and devised a tool for **detecting and localizing anomalous activity** in the SoC in **realtime**.

SoCurity: enhancing SoC security with NoC monitoring



Live workloads
- User runs tasks on device
- Device system runs tasks in the background
- *Injected malware runs attacks*
...

run on

SoC with SoCurity:

Acc1 Acc2 Acc3 Mem CPU CPU SoCurity analyzer

SoCurity NoC-based Counters in each router

General counters

Packet injections on plane 1	346
Packet injections on plane 2	716
...	...
Packets in input queue on plane 1	4
Packets in input queue on plane 2	3
...	...

Mem counters

Coherence reqs	31
Coherence resps	29
DMA reqs	26
DMA resps	23
...	...

read every time interval

SoCurity-based anomaly detection and localization

Trained anomaly detection model

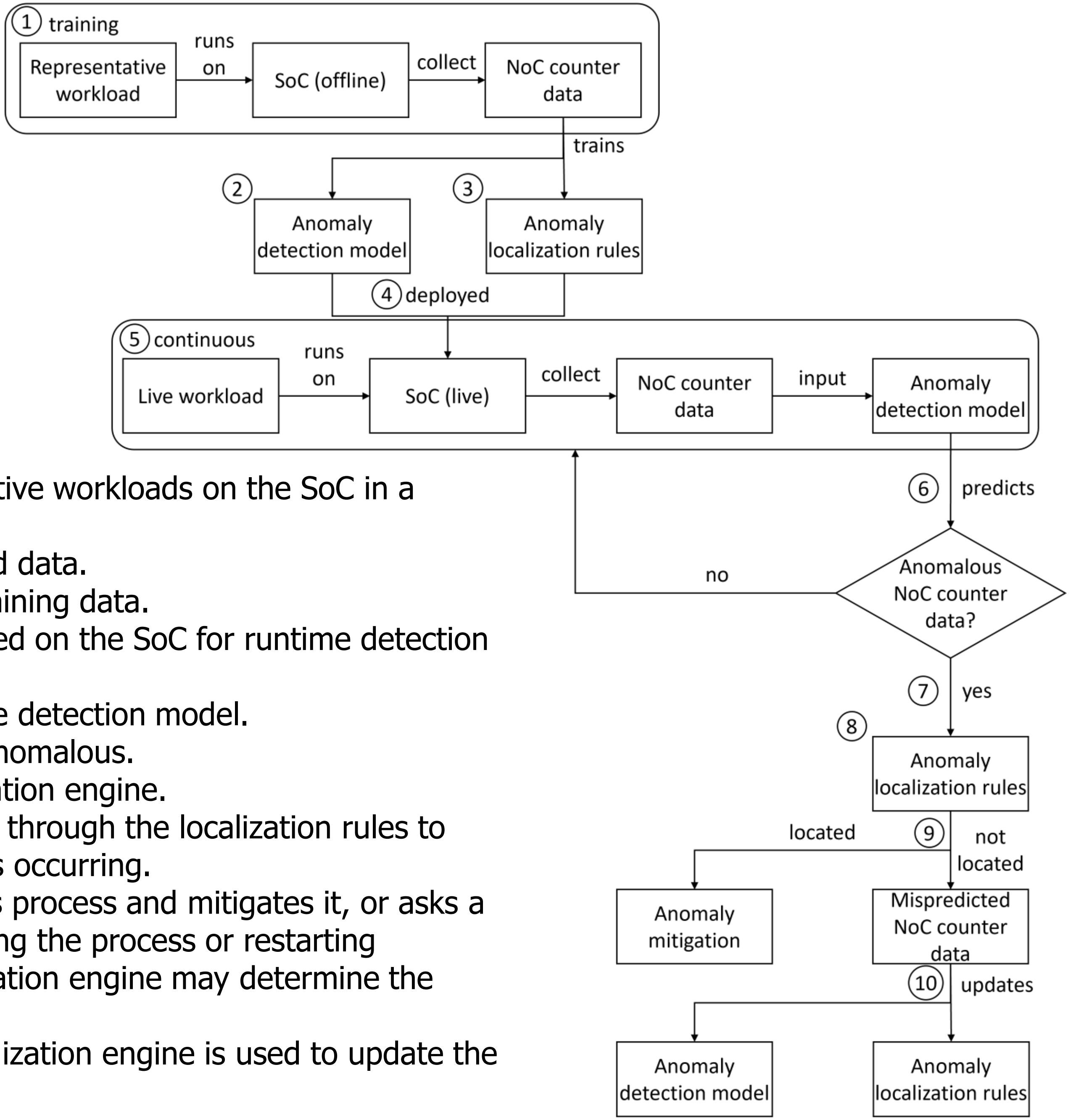
Anomaly detected

Localization engine

- **HW counters at each NoC router**
  - Monitor activity throughout SoC
  - No requirements for internal design of SoC components → black box, reconfigurable
- **Anomalous activity detection**
  - Lightweight semi-supervised anomaly detection model to monitor NoC counters
  - Trained with benign data
  - Can detect existing and novel future availability attacks, HW issues
- **Anomalous activity localization**
  - NoC counters indicate where anomalous activity is occurring
  - Explainable ML technique for determining specific anomalous counters to localize anomaly

Naorin Hossain, Alper Buyuktosunoglu, John-David Wellman, Pradip Bose, Margaret Martonosi. "SoCurity: A Design Approach for Enhancing SoC Security", IEEE CAL 2023.

SoCurity full process



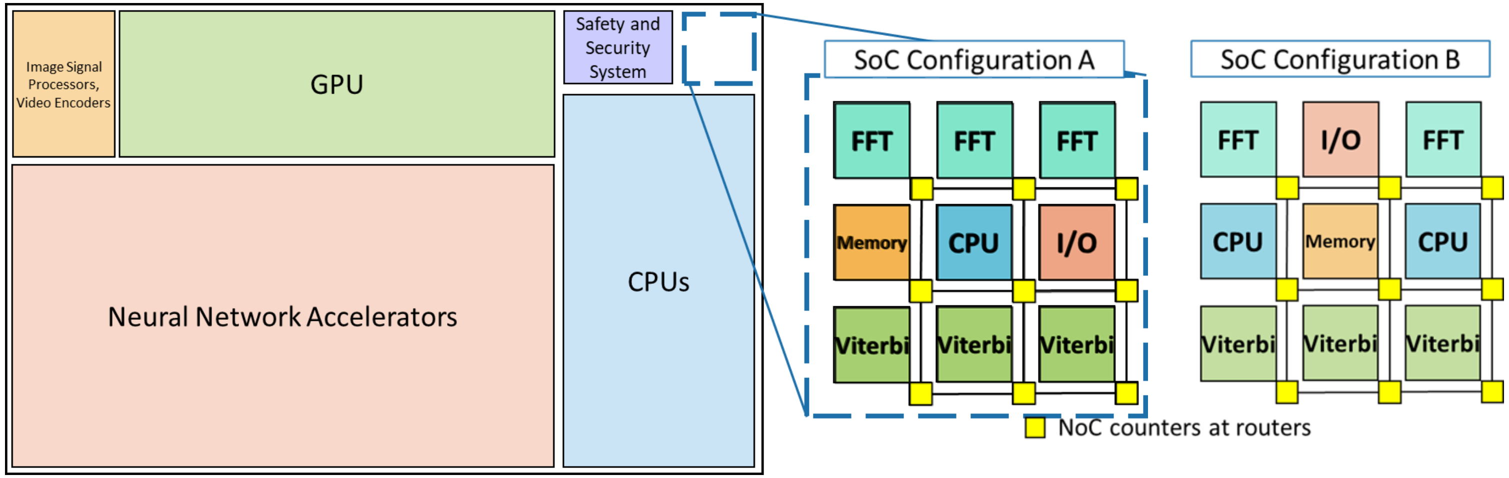
1. NoC counter data collected while running representative workloads on the SoC in a secure, offline setting.
2. Anomaly detection model is trained with the collected data.
3. Localization rules are extracted from the collected training data.
4. Trained detection and localization models are deployed on the SoC for runtime detection and localization.
5. NoC counter data is periodically read and input to the detection model.
6. The detection model predicts if the counter data is anomalous.
7. If the data is anomalous, it is provided to the localization engine.
8. The localization engine runs anomalous counter data through the localization rules to determine where in the SoC the anomalous activity is occurring.
9. The localization engine finds the detected anomalous process and mitigates it, or asks a trusted supervisor or OS to mitigate it, (e.g., by ending the process or restarting impacted SoC components). Alternatively, the localization engine may determine the data was mispredicted so no action is needed.
10. Counter data labeled benign by the detection or localization engine is used to update the detection and localization models over time.

SoCurity Case study: detect, localize DoS attacks on connected autonomous vehicle (CAV) SoCs

- CAVs share data on surroundings with nearby vehicles, smart infrastructure, and the cloud
- Complex SoCs, real-world deadlines → **availability attacks** like DoS can severely compromise safety.
- We ran 4 DoS attacks on the CAV SoCs with 4 semi-supervised anomaly detection models.

***One-class* semi-supervised detection models:**

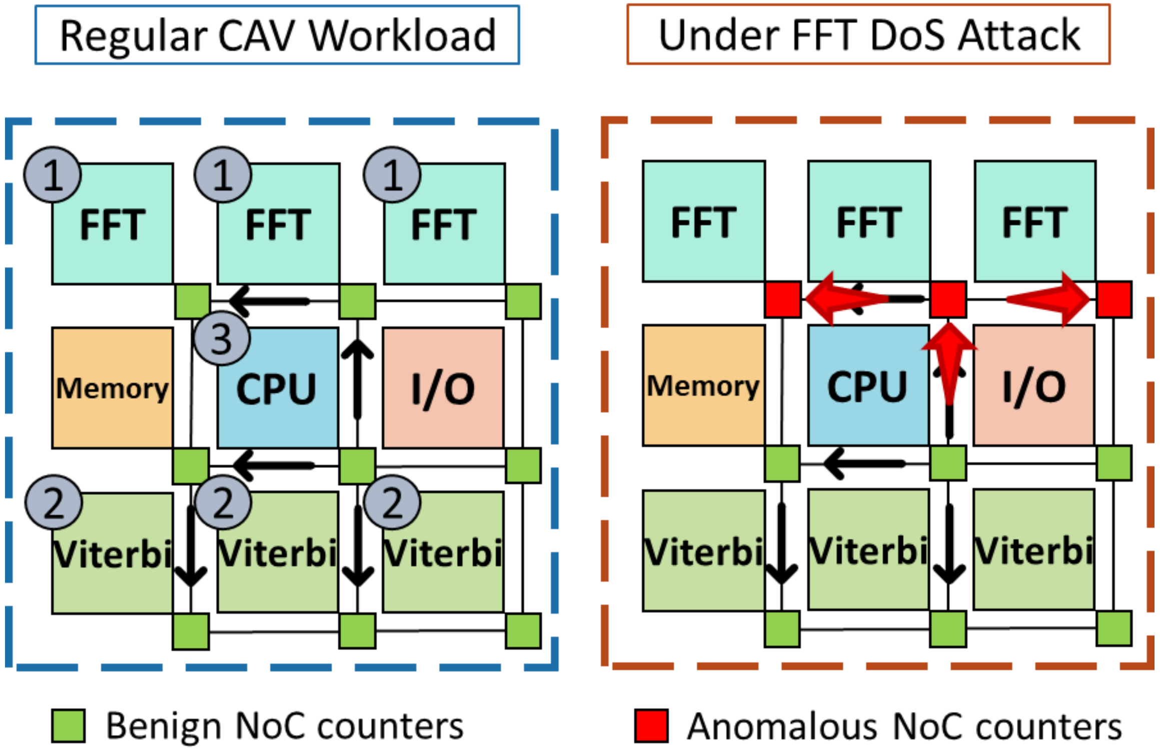
- One-class nearest neighbors (OCNN)
- One-class support vector machines (OCSVM)
- Isolation forest (iForest)
- Local outlier factor (LOF)



Experiments performed on SoC for CAV communication subsystem emulated on FPGA with ESP framework. SoCurity NoC counters implemented using ESP to monitor:

- General NoC activity
- Memory accesses
- Accelerator usage

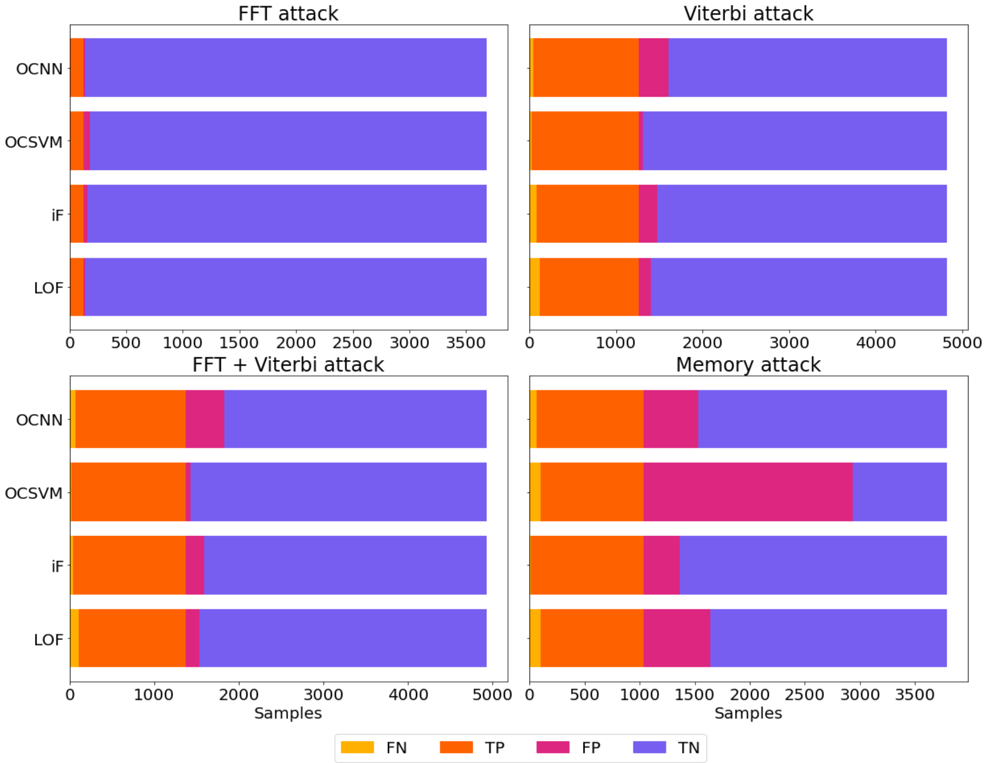
Representative CAV workload and targeted DoS attacks used to train and test SoCurity detection system



- Regular CAV workload – completes the following tasks per time step:
  1. **FFT:** distance calculations based on radar readings.
  2. **Viterbi decoder:** decoding messages received from other CAVs, the cloud, etc.
  3. **CPU:** combine learnings from 1 and 2 to plan and execute actions.
- DoS attacks
  - **FFT:** 1000 tasks sent to all FFT accelerators
  - **Viterbi:** 1000 tasks sent to all Viterbi accelerators
  - **FFT + Viterbi:** 1000 tasks sent to all FFT accelerators and 1000 tasks sent to all Viterbi accelerators
  - **Memory:** 500,000 memory requests are made for random addresses in a space that is twice the size of the LLC
- SoCurity counter data sampled for detection every 100ms on FPGA @ 78MHz

Results: Offline and online experiments show fast, effective DoS attack detection, localization with SoCurity

Results from offline experiments:



Accuracy correlated with impact of attack on workload.

Results: Offline and online experiments show fast, effective DoS attack detection, localization with SoCurity

Results from realtime experiments:

Attack on SoC A	Samples to detect	Accuracy	FPR
FFT	0.96	0.96	0.04
Viterbi	2.82	0.87	0.05
FFT + Viterbi	1.06	0.87	0.06
Memory	2.22	0.60	0.04

Fast detection across all attacks → only a few detection cycles required.

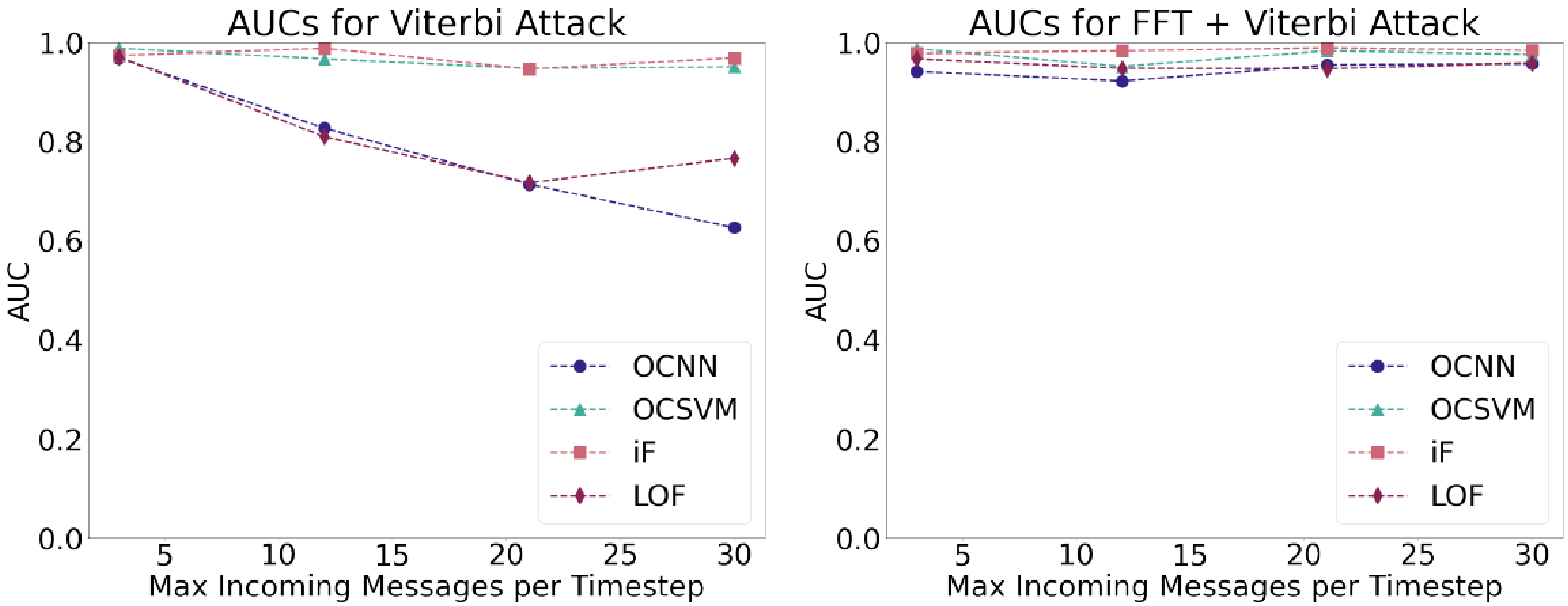
Results: Offline and online experiments show fast, effective DoS attack detection, localization with SoCurity

Results from localization experiments:

Attack on SoC B	**Attack Correctly Localized**	**Other Anomaly Localized**	**Undetermined**
FFT	**0.94**	0.03	0.03
Viterbi	**0.88**	0.11	0.01
FFT + Viterbi	**0.95**	0.03	0.02
Memory	**0.99**	0.00	0.01

High accuracy in pinpointing
attack locations within SoC

SoCurity robustness to variability



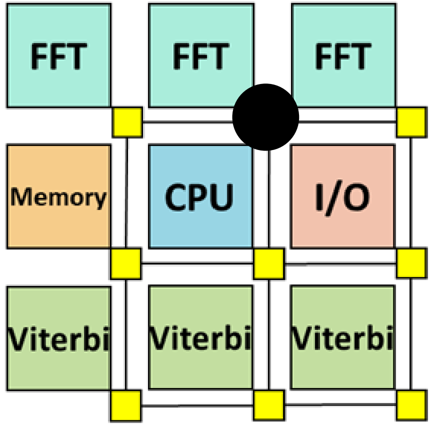
(a) Viterbi Attack

(b) FFT + Viterbi Attack

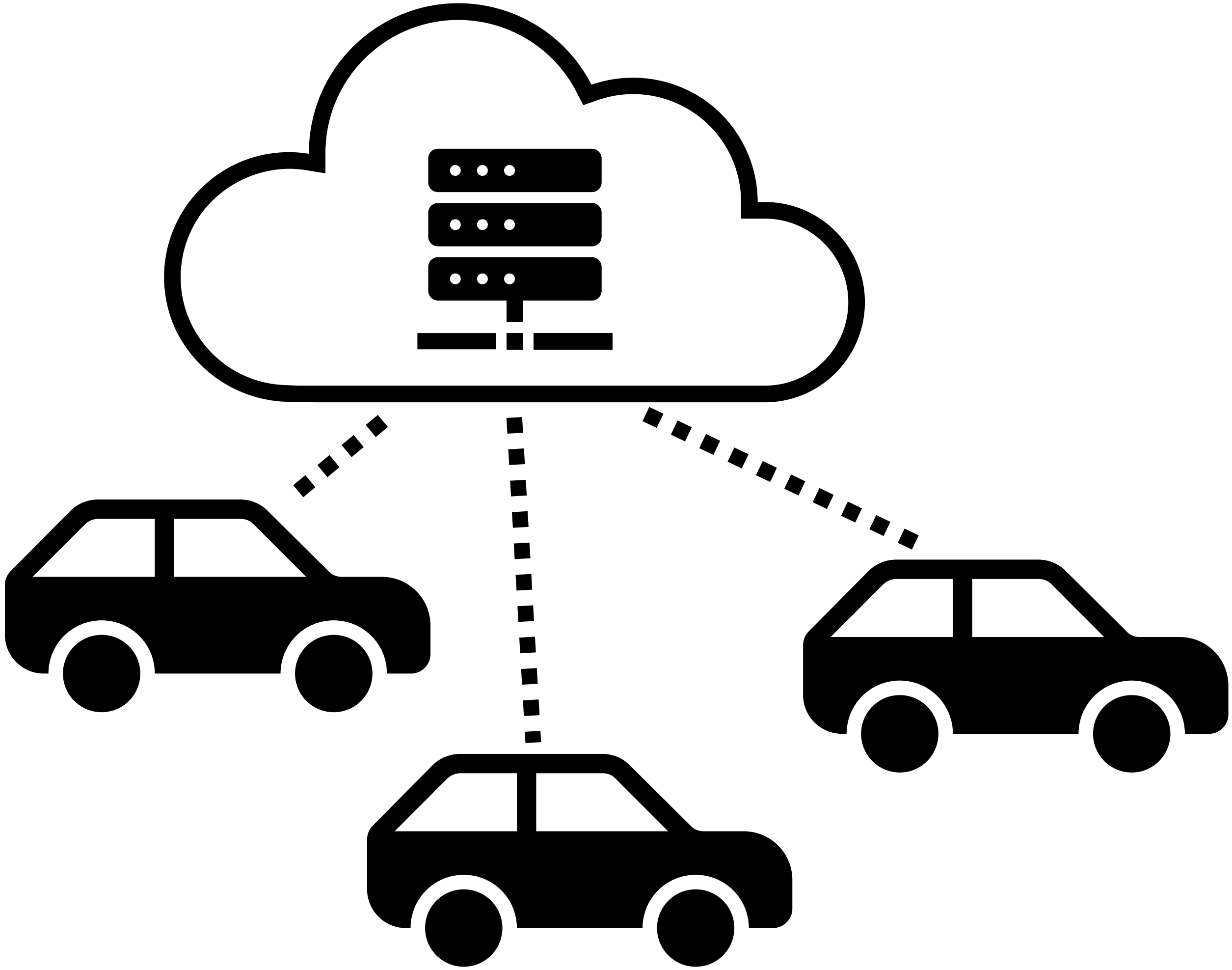
Accuracy remained high with OCSVM and iForest models when detecting Viterbi-based attacks when variability of base workload Viterbi activity was increased.

Applying SoCurity for other threats

- NoC counters can enable detection of anomalous activity caused by other hardware attacks in the SoC as well
    - Resource usage attacks such as cache-based side channels
        - Spectre
        - Meltdown
        - etc.
    - *Lack of resource usage* attacks
        - E.g., black hole router attacks



Applying SoCurity for improving system reliability



- Another system concern, particularly in safety-critical domains: reliability!
- Silent data corruption (SDC)
  - Detrimental at large scale (e.g., data center)
  - Many edge devices are cloud-backed
- SoCurity counters may be able to detect erroneous behaviors due to
  - Cloud-provided corrupt data
  - Internal SDC (e.g., corrupted data transmissions between components)

In Summary

- **SoCurity**: lightweight, broadly applicable design enhancement for adding built-in security to heterogeneous SoCs
- **Case study on CAV systems**: fast and accurate DoS attack detection and localization enabled by SoCurity using ML techniques
- SoCurity can be applied to enhance security beyond availability attacks:
  - Side channel attack detection
  - Black hole router attack detection
  - Silent data corruption detection and localization