MUSTARD - Adaptive Behavioral Analysis for Ransomware Detection

Davide Sanvito, Giuseppe Siracusano, Roberto Gonzalez, Roberto Bifulco

NEC Laboratories Europe

Problem

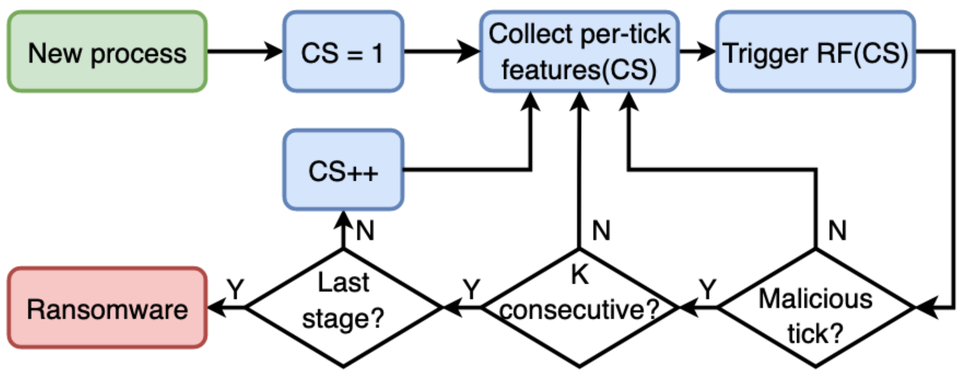
- Ransomwares can be detected with behavioural analysis on filesystem operations
- Monitoring all the operations for all the processes can introduce a significant overhead
  - E.g. time to copy 10GB file with `dd` in Linux ext4:
    up to 20% slower when updating per-process counters with eBPF
  - Potential limited applicability in real settings

Idea

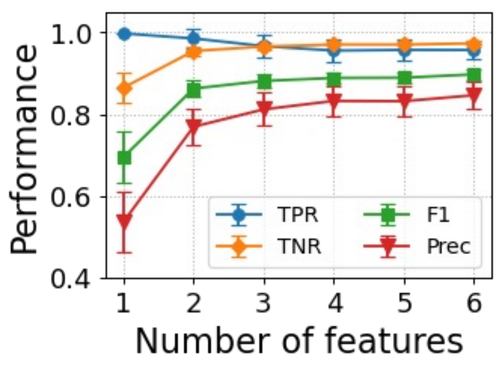
- The type and number of features monitored affect the overhead
  - Number of underlying low-level OS interfaces
  - Compute-intensive features (e.g. write entropy)
- Most of the processes are benign
  - A limited number of processes have a ransomware-like behaviour
- **Can we keep monitoring lightweight for most processes and dynamically adapt it only for those with suspicious behaviour?**

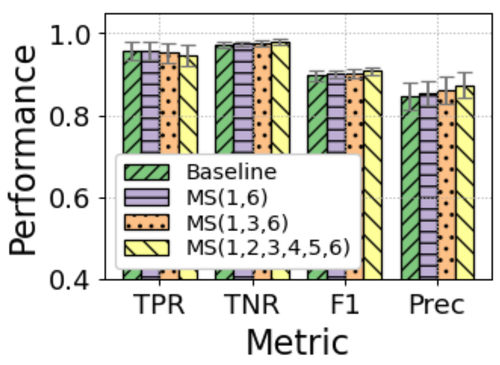
MUSTARD

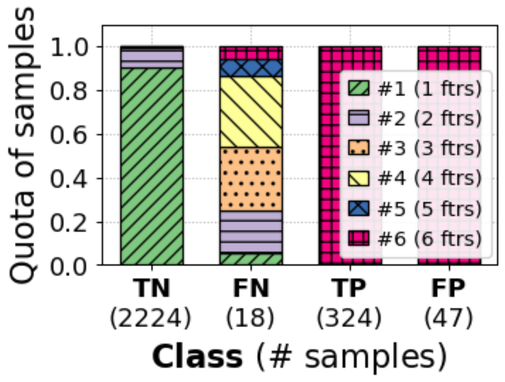
- **Mu**lti-**St**age **A**daptive **R**ansomware **D**etection
- It cascades increasingly complex ML models (stages) with an increasing number of input features
- Most monitored processes use few simpler features and models
- Only suspected ransomwares activate more complex ones
- MUSTARD workflow
  - Monitor a new process with a single or few feature(s) and periodically run the ML model of the first stage
  - Move to next stages (i.e. collect more features and select a more complex model) based on the process behaviour
  - Declare a process as ransomware once it positively triggers the ML model of the last stage



Results







	Detection latency (baseline)	*Additional* ticks wrt baseline		
		MS(1,6)	*MS(1,3,6)*	*MS(1,2,3,4,5,6)*
min	3.0	+3.0	+6.0	+15.0
avg	8.2	+4.4	+8.6	+20.1
95p	41.2	+14.0	+17.0	+45.0
max	241.0	+49.0	+151.0	+166.0

- Dataset: 6 features derived from filesystem events [1]
- Baseline and per-stage ML model based on [2]
  - Events are slotted in *ticks*
  - Per-tick features are fed to a binary Random Forest classifier
  - Process declared ransomware after K consecutive malicious ticks
  - The baseline model uses all the features all the time
- Preliminary implementation
  - Static sequence of stages and corresponding features
  - Early stages should
    - minimize the monitoring overhead (avoid heavy monitoring for most processes)
    - prioritize Recall over Precision (minimize FN over FP)

MUSTARD can keep detection performance, reducing the monitoring overhead at the cost of a small increase in the detection latency

Next steps

- Refine the monitoring adaptation logic: monitoring overhead vs risk mitigation trade-off
- Perform a complete evaluation of the monitoring costs by collecting a new dataset
- Full implementation (e.g. based on eBPF framework)

References:

[1] ShieldFS dataset: http://shieldfs.necst.it

[2] A. Continella, A. Guagnelli, G. Zingaro, G. De Pasquale, A. Barenghi, S. Zanero, and F. Maggi, *"ShieldFS: a self-healing, ransomware-aware filesystem"*, In ACSAC 2016

This work has received funding from the European Union's H2020 research and innovation programme under grant agreements n. 883335 ("PALANTIR") and n. 101017171 ("MARSAL").