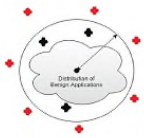
LAMBDA: Lightweight Assessment of Malware for emBeddeD Architectures

Research Objective

To propose a framework for runtime anomaly detection on embedded systems - The framework is capable of performing anomaly detection in a hierarchical manner (i.e. application level, operating system level and processor micro-architecture level) by harnessing the information available at various levels to detect malicious exploits.

Motivation



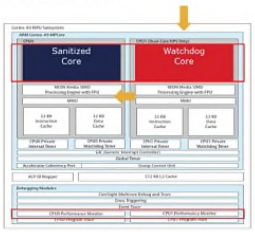
Measure the distance of a program under test from the characteristics of a given set of benign programs.

If the distance is less than a previously defined threshold value, the target program can be treated as a benign program otherwise the program is a malware.

Detection Approach

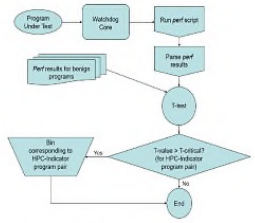
Use (HPC, Indicator) for monitoring.

Control Flow



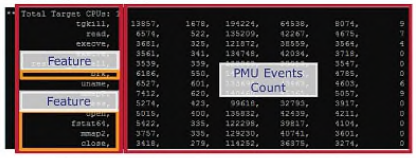
Dual core setup
- Watchdog Core: to monitor all the processes.
- Sanitized Core: to run non-malicious processes.

Creating Bins



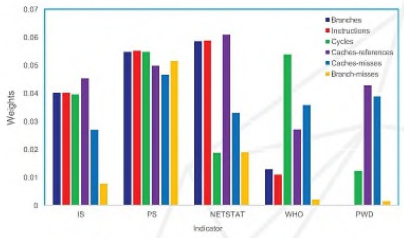
Emphasize on critical HPC-Indicator pair
- Performed to give more weightage on important performance counters and indicator programs.

Data Collection



Hardware Performance Counters
- More efficient to detect Kernel modifying rootkits.
- Easily accessible in most of the Linux based systems.

Normalized Weights

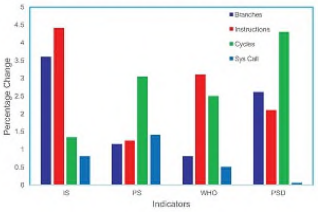


Determine weights during training
- Normalised weights help at runtime to determine distance of malware using statistical T-test.

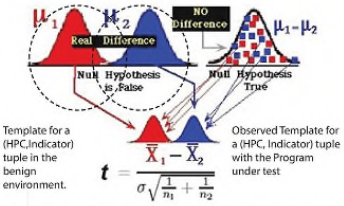
Scoring at Runtime

Calculate the amount of maliciousness of a program under test
- Create bins for program under test at runtime.
- Multiplication of the trained weights with these bins produces score for the program under test.
- Score greater than a pre-defined threshold value signifies the malicious behaviour of the program.

HPC provides more sensitivity & Increased Protection



Run time Statistical T-test



Template for a (HPC,Indicator) tuple in the benign environment.

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sigma\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

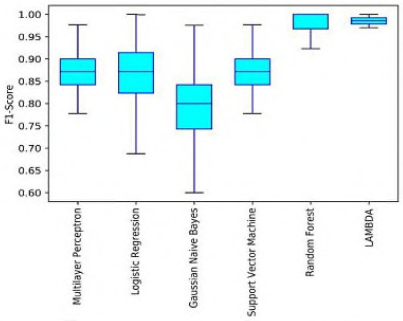
Observed Template for a (HPC, Indicator) tuple with the Program under test

The null hypothesis of two equal means is rejected when the test statistic |z| exceeds a threshold of 4.5, which ensures a confidence of 0.99999.

Advantages for HPC observation:
Difficult to manipulate HPC values by the malware.
More sensitive when observed in conjunction with system calls.
Results in better false positives and negatives

Advantages of the approach
- Monitoring only system calls doesn't provide any significant information but monitoring HPCs does. Significant changes can be observed in presence of malware.
- Enables semantic based malware detection.
- Supports multi-core environment.

T-test vs. Machine Learning based Detection



-- Box plot showing F1-Scores for various machine learning based techniques along with LAMBDA

-- It can be observed that variation in F1-Score is less for LAMBDA

--Signifies LAMBDA is consistent over different train-test pairs

Advantage over Training and Detection Time

	Model Building Time (in milli seconds)	Detection Time (in milliseconds)
Multi Layer Perceptron	2036.91	10.24
Gaussian Naïve Bayes	7.17	10.43
Random Forest	85.95	91.39
LAMBDA	43.72	15.38

Sai Praveen Kadiyala, Muhamed Fauzi Bin Abbas,
Debdeep Mukhopadhyay,
Siew-Kei Lam, Thambipillai Srikanthan