Malware are stealthy and hard-to-detect by performance analysis of programs. Various evasion mechanisms adopted by these malwares make them still succeed to execute, affecting precious data. Recent trends try to detect their presence by performing behavioral analysis using dedicated software to quarantine such malicious codes. Often observing high-level information, like system calls, have been evaded by smart malwares. In addition to it, identification and blocking of runtimes at the earliest along with recovering the contents of the already encrypted files is an open challenge. It is proved that despite advancing in encryption systems, the prominent runtimes leave a trail in the address of I/O and file-systems.

Advantages of HPCs

Hardware Performance Counters (HPCs) have certain advantages in detecting the presence of malwares as compared to other system call based detection methods.

- HPCs provide more sensitive information of a system behavior than system calls.
- HPCs are difficult to manipulate by the malware writer.
- Easily accessible in most of the Linux based systems.

Monitoring Features

The feature monitored in the detection mechanism is a tuple of (Indicator, Observer).

- Indicator: Benign Operating System library executables, like ls, netstat, ps, who, psad.
- Observer: Low-level hardware events, like cycles, instructions, cache-references, cache-misses, branches, branch-misses.

Malware Detection

To highlight the role of low-level hardware events detected from Hardware Performance Counters (HPCs) in detecting the existence of malware execution with the help of two case studies:

- Developing a statistical lightweight tool, in the context of an embedded platform, to evaluate the potential of a program under test of being a malware.
- Developing a very fast detection methodology for popular ransomware on standard desktops.

Ransomware Detection

Hardware events likely to change because of the symmetric and asymmetric key encryptions of runtimes are monitored. Generally the symmetric encryption affects the cache based events while the asymmetric encryptions affect the instruction and branching events.

Behaviour in Frequency Domain

Figure 8: Variation of Amplitude in frequency domain of HPCs in presence of SPEC and Wannacry

Second Autoencoder Behavior

Figure 9: Reconstruction Errors in the Second Autoencoder

First Autoencoder Behavior

Figure 10: Reconstruction Errors in the First Autoencoder

Table 1: Average Training and Detection time for different models

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Time (mS)</th>
<th>Detection Time (µS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian Naive Bayes</td>
<td>14.3784</td>
<td>255.8256</td>
</tr>
<tr>
<td>Multilayer Perceptron</td>
<td>14.1558</td>
<td>287.8915</td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>378.8816</td>
<td>1777.9335</td>
</tr>
<tr>
<td>Random Forest</td>
<td>839.0549</td>
<td>549.9807</td>
</tr>
</tbody>
</table>

Table 2: Resource requirement on x86 and ARM processors (per sec)

<table>
<thead>
<tr>
<th>Processor</th>
<th>CPU Usage</th>
<th>MEM Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>x86</td>
<td>14.405%</td>
<td>13.285%</td>
</tr>
<tr>
<td>ARM Cortex-A9</td>
<td>16.788%</td>
<td>10.380%</td>
</tr>
</tbody>
</table>