REQUIREMENT ENGINEERING FOR MALWARE DETECTION USING HARDWARE PERFORMANCE COUNTERS

7/18/23

Elly Khodaei

Presentation for CS 846

UNIVERSITY OF WATERLOO | FACULTY OF ENGINEERING

What are hardware performance counters?

Hardware performance counters or performance monitoring counters are **built in registers** in computer processors.

every time a software runs in a computer, the computer will save some values as events in these registers. For example: the number of branch-misses, the number of cache-hits, instructions, are some examples of performance events that will be counted in the processor.

Depending on the system, there is a different number of performance counters available to monitor. For example in AMD processors there are a number of over 100 performance counters available.

Some hardware performance counters

```
Performance counter stats for 'system wide':

       158,211.75 msec cpu-clock                 #   24.018 CPUs utilized
           64,126      context-switches          #  405.318 /sec
              467      cpu-migrations            #    2.952 /sec
            3,829      page-faults               #   24.202 /sec
    4,625,715,211      cycles                    #    0.029 GHz                      (33.25%)
      384,058,056      stalled-cycles-frontend   #    8.30% frontend cycles idle     (33.25%)
      768,028,149      stalled-cycles-backend    #   16.60% backend cycles idle      (33.26%)
    3,271,461,441      instructions              #    0.71  insn per cycle
                                                 #    0.23  stalled cycles per insn  (33.29%)
      652,368,421      branches                  #    4.123 M/sec                    (33.34%)
       27,916,510      branch-misses             #    4.28% of all branches          (33.37%)
    1,430,046,036      L1-dcache-loads           #    9.039 M/sec                    (33.37%)
       77,314,863      L1-dcache-load-misses     #    5.41% of all L1-dcache accesses  (33.37%)
   <not supported>      LLC-loads
   <not supported>      LLC-load-misses
    1,040,439,083      L1-icache-loads           #    6.576 M/sec                    (33.37%)
       20,307,222      L1-icache-load-misses     #    1.95% of all L1-icache accesses  (33.37%)
       13,516,700      dTLB-loads                #   85.434 K/sec                    (33.37%)
        2,969,423      dTLB-load-misses          #   21.97% of all dTLB cache accesses  (33.37%)
        2,371,970      iTLB-loads                #   14.992 K/sec                    (33.36%)
          930,809      iTLB-load-misses          #   39.24% of all iTLB cache accesses  (33.34%)
       41,881,049      L1-dcache-prefetches      #  264.715 K/sec                    (33.28%)
   <not supported>      L1-dcache-prefetch-misses

       6.587219954 seconds time elapsed
```

Why do we need to use these hardware performance counters (hpc) ?

- Malware attack different kinds of programs and softwares.
- Software malware detectors could also be the target of a malware attack
- According to this we need a tool that we could trust better.
- But hardware registers cannot be attacked by the malware, Since they are hardware components in the processor.
- Therefore they are a more accurate tool.

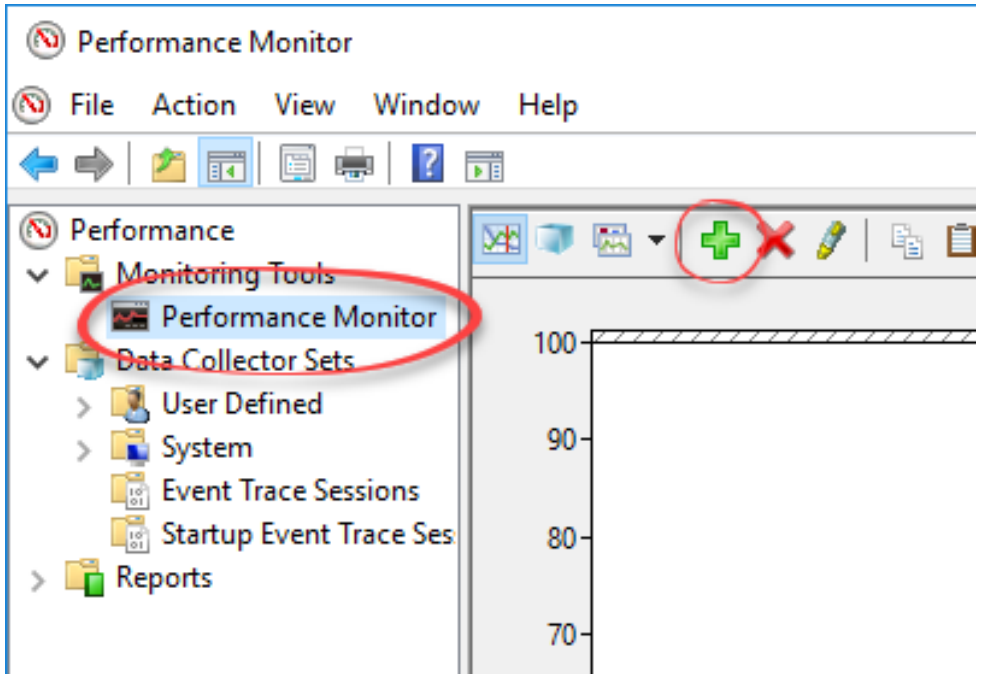
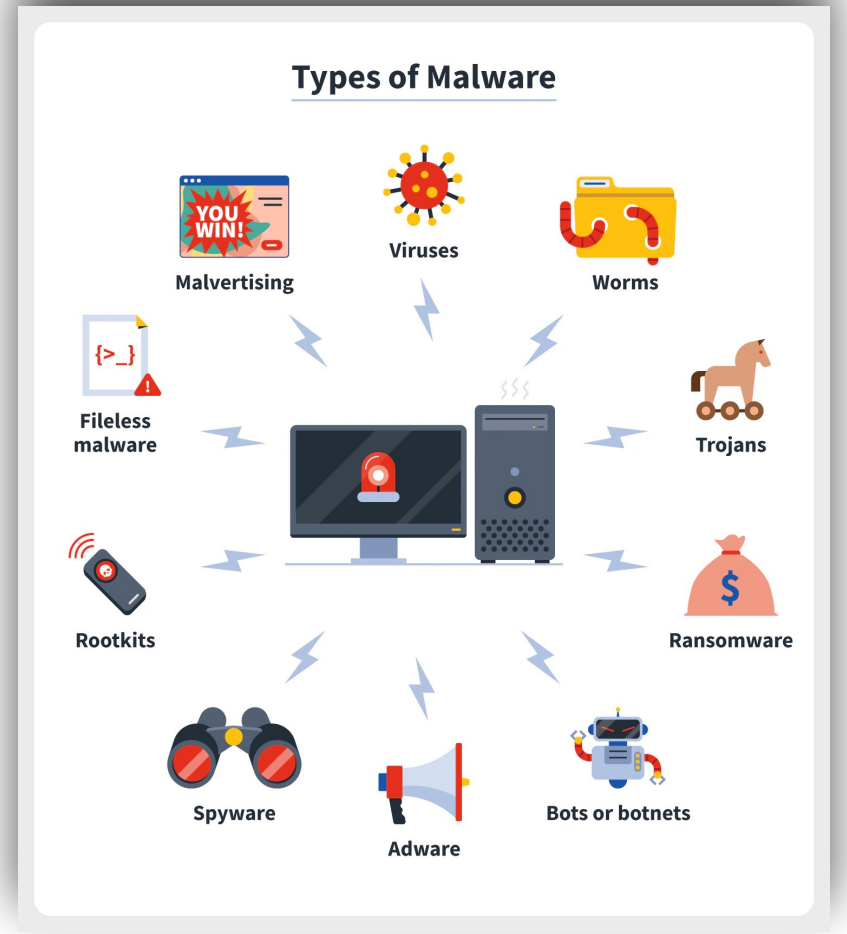


Image 1: The performance monitoring application in windows

How does malware detection with hardware performance counters work?

- First we have to have some malware binaries.they could be ELF, Exec etc and regular system softwares as benign-wares
- Second, we have to run them in a sandboxed environment- so that we won't destroy our own system!!! These environments could be: Virtual machines, Containers like dockers etc.
- Third, while we are running the malware, we have to collect the performance counters trace that we want and save them in a file.



What's next? What happens after collecting our performance counter data?

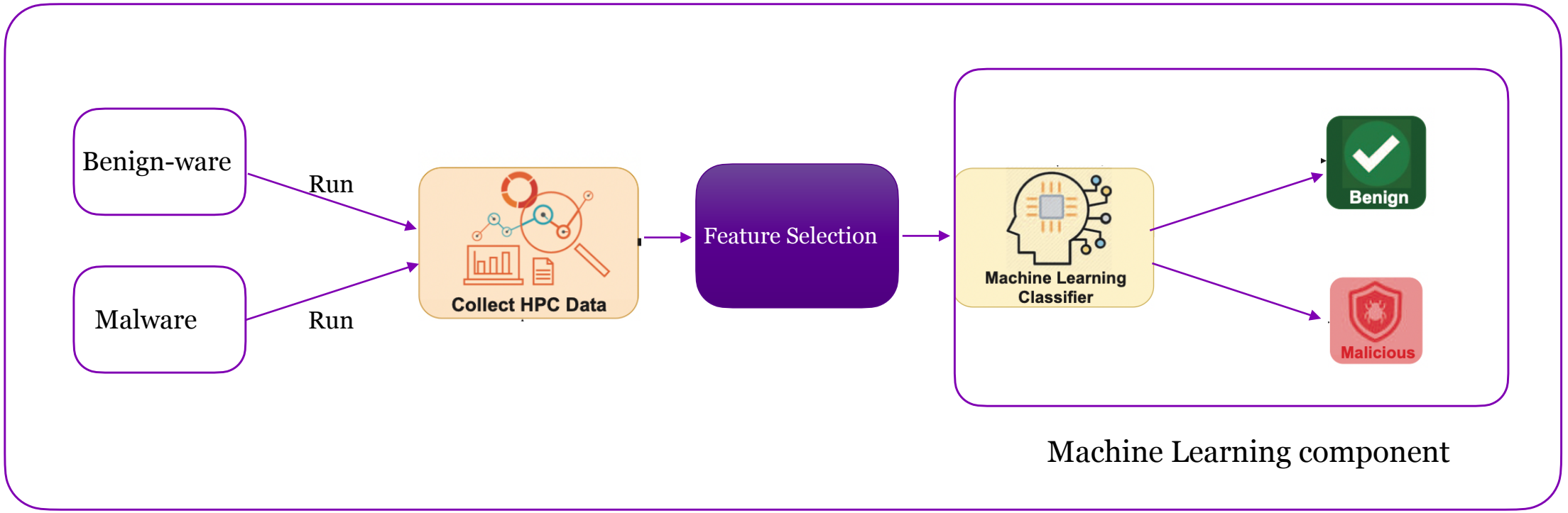
After we collect our data, we have to do a feature selection, since we have more than a 100 of performance counters to look at we should select the ones that are more important to us, and would explain the data better.

After feature selecting we have to feed our feature selected data to a machine learning model.

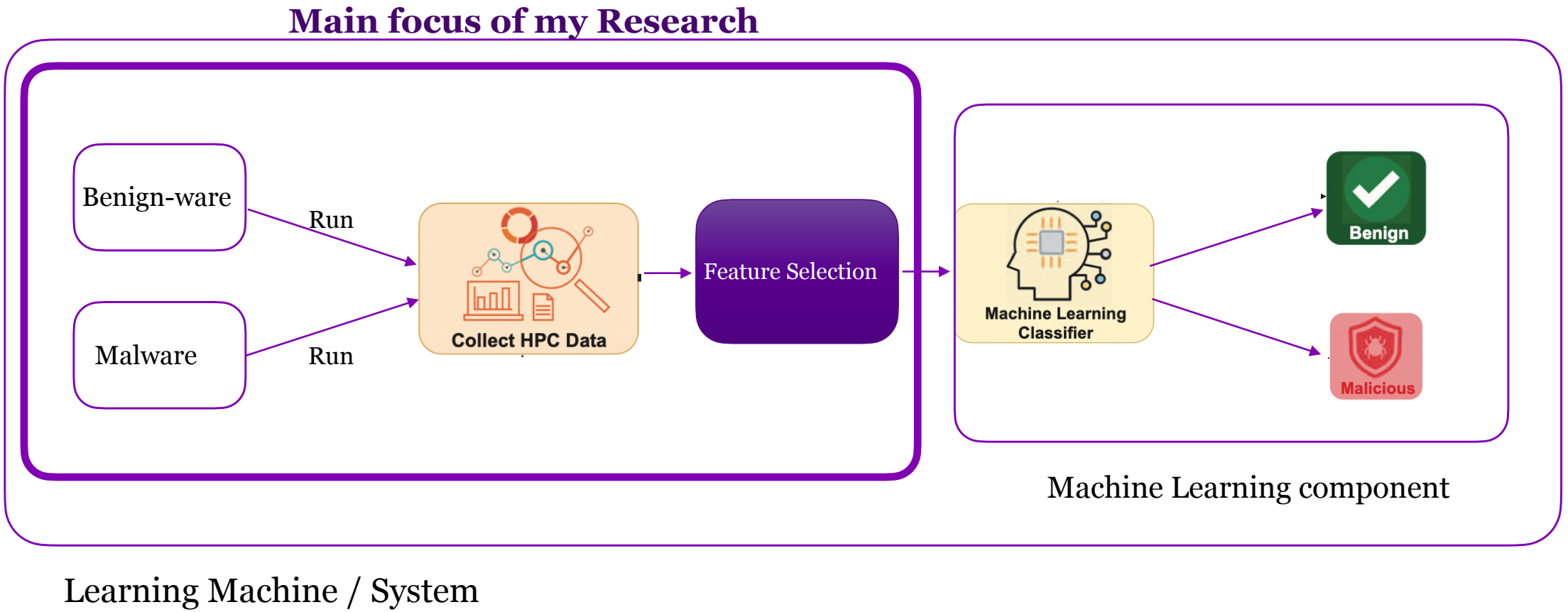
The model should be trained with benign-ware and the malware and for testing

the machine learning model, should be able to detect an anomaly happening and detect the malware.

Detailed model of the system



Detailed model of the system



Layers of the System

- **1**. To run the malware and benign-ware
- **2**. Collect performance counters
- **3**. Feature selection on the collected performance counters

- 4. Machine Learning detector

Previous work

- A lot of prev work use virtual machines or virtual boxes for their experiments, and they are not that accurate.
- Most of the prev work do not consider the performance counter data as a multi-variate time series data. And the feature selection methods that they use are not for time series feature selection specifically.
- They used fixed performance counters for every malware family, but each malware appears in the performance counters differently. So the feature selection should be different for each malware family.
- …..

What is Requirement Engineering?

- Requirements engineering is the process of eliciting stakeholder needs and desires and developing them into an agreed-upon set of detailed requirements that can serve as a basis for all subsequent development activities.
- Requirements engineering is a process of gathering and defining what services should be provided by the system.

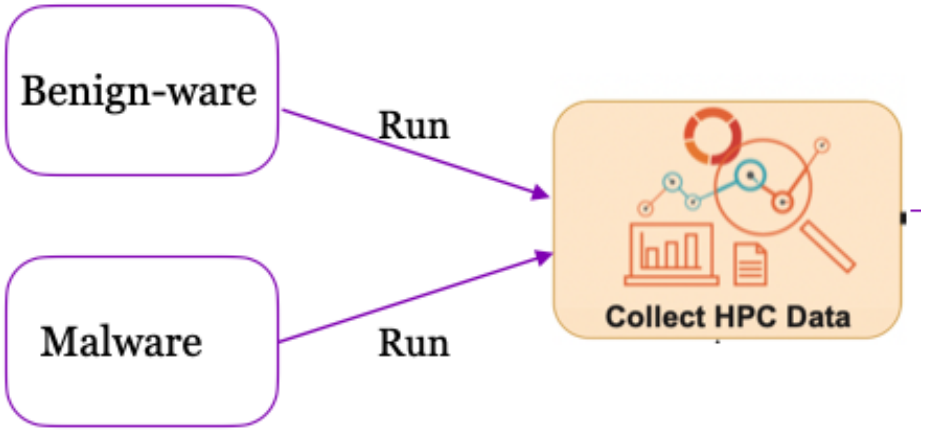
Who are the Stakeholders here?

- Anyone that is using a pc
- Companies
- And my Supervisor ! :)

Requirements of each layer of the system

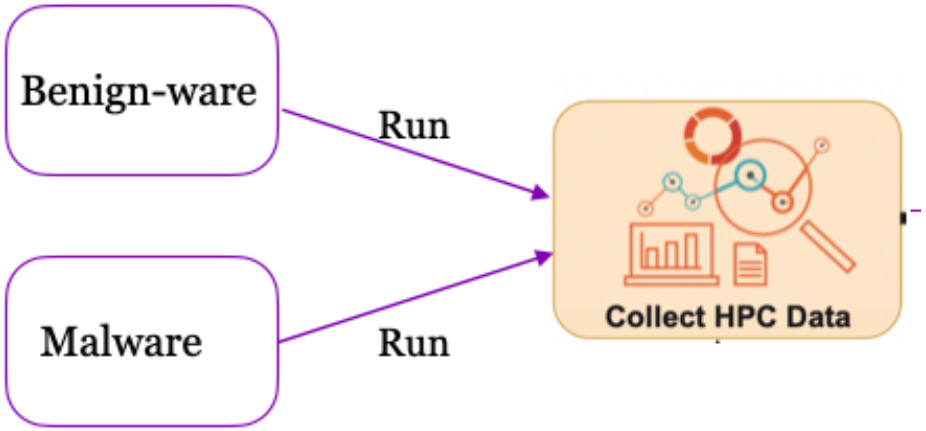
Requirements for collecting the HPC

- Since we want to train a malware detector model that is able to detect different malware families, we have to have multiple malware samples in each family. Ransomware, Backdoors, …
- We have to have compatible software applications/Benign-ware samples as well to be able to train a model.
  Example : for a ransomware malware that encrypts files on a system we need to have an encrypting application as a benign-ware, for example zip, compress. Etc



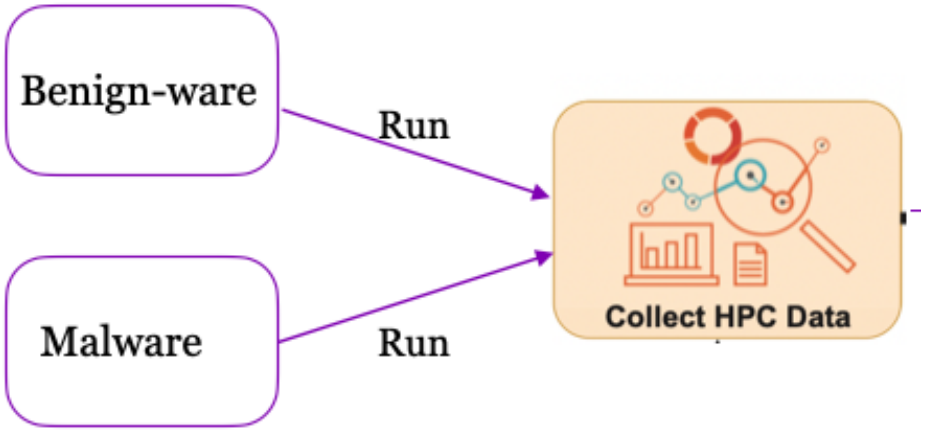
Requirements for collecting the HPC

- **It's not that easy to know that a malware is running** ! That's why there are multiple virus detectors.
- Even if we collect some malware samples that we are sure that they act malicious, different malware could act different on each system. Malware usually don't show themselves. And for a malware to run we have to know the dependencies they need, example: they may need specific libraries to be able to run and not all of them are able to install their dependencies.



Requirements for collecting the HPC

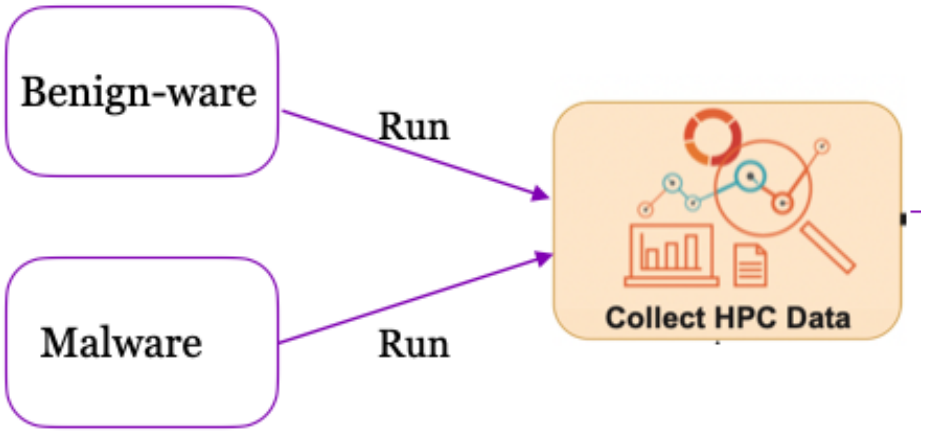
- It's not that easy to know that a malware is running !
- The malware should be compatible with our system, the version, the architecture, etc.
- So in order to create an accurate dataset out of the Malware samples, we have to be sure that the malware are running. And this is not an easy task because other than their dependencies some malware like root-kits or some backdoors are designed to be undetectable.



Requirements for collecting the HPC

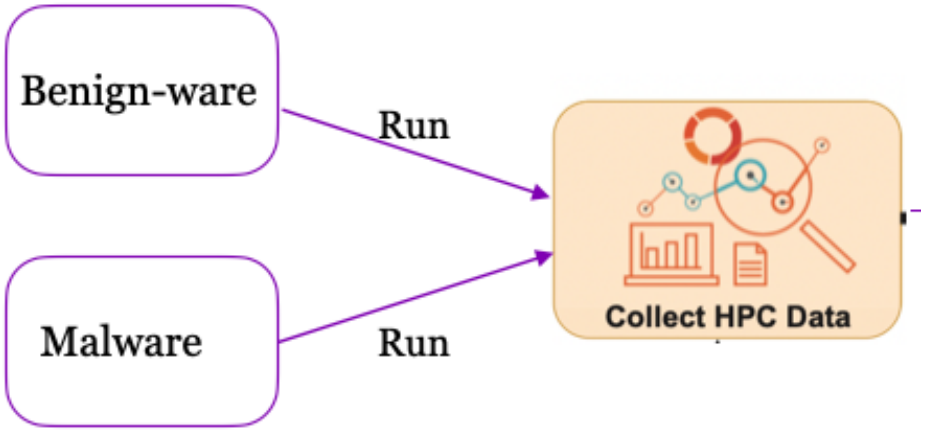
- We need different families of malware.
- We have to have multiple benign-ware benchmarks.

- The environment that we choose to run our malware is important, some environments like virtual machines or virtual boxes add a considerable amount of noise to the performance counter data and we don't want that.



Requirements for collecting the HPC

- Most of the malware need internet connection to work.
- Malware look at the environments they are in, they would look for specific directories, internet connection, common softwares, .. and some would not start working if they discover that the system is not for personal use, and it's a virtual env.
- The environment should be as close as it can be to a personal computer.
- While the malware is running, application s should run in background.



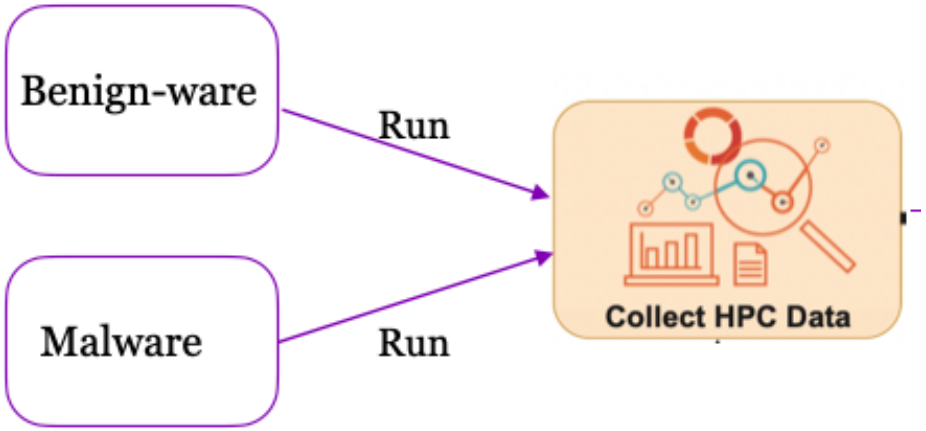
Personal experience !

- I was Running malware samples, and a couple of days later we got an email from Information Security Services, that they detected a Cryptocurrency Miner Checkin, and they disabled our ports.

- Even though there are security risks with running malware with a system connected to the internet, we have to give access to the malware since we have to be sure that they are working like they would in a regular environment.
- But the ransomware attack wasn't me ! :)

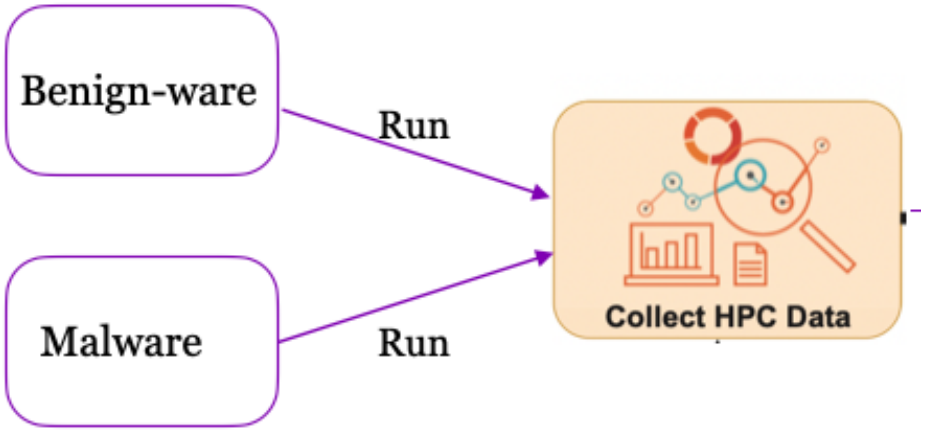
Requirements for collecting the HPC

- There are more that 100 performance counter available, we need to pick a subset of them for example we have to have at least 30 of these performance counters. To perform a feature selection in the next level.
- Depending on the system we can monitor only 4-6 performance counters at the same time, so we have to run each malware multiple times. (To get 30 -> 7,8)
- System needs to reset every time a new malware is going to run.



Requirements for collecting the HPC

- System needs to reset each time because the prev malware traces should not affect the new malware.

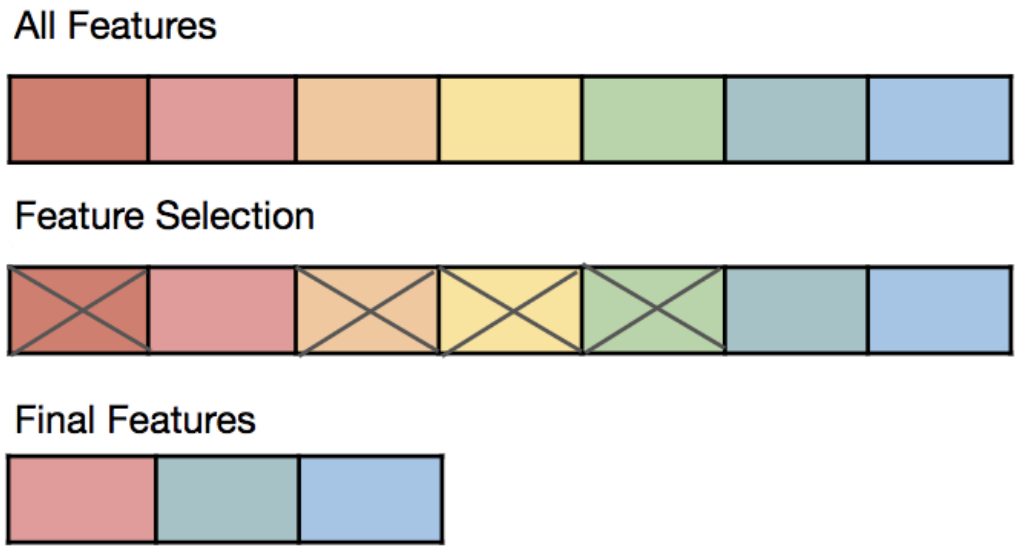


RE for the feature selection

- After we collect the set of performance counters as our data set. We have to do feature selection on them.
- The data that we are dealing with is a multi-variate time series data, therefore the feature selection methods that we use should be suitable for this purpose.
- Feature selection methods may work better for different malware. For example: the final Accuracy of the ML model, when we use F1 for MalF1, but we would get higher accuracy if we use F2 for MalF4

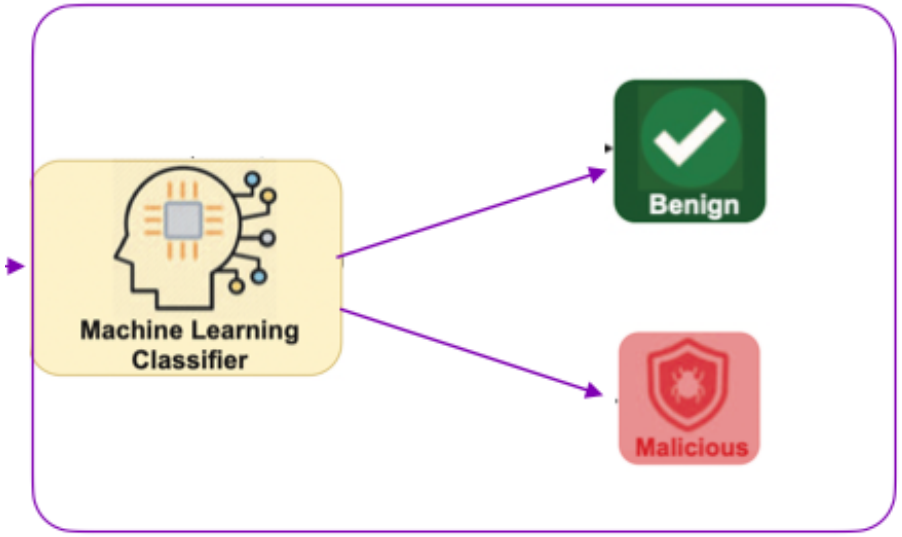
RE for the feature selection

- We should run each feature selection method for Each malware family that we gathered.



How do we evaluate our system?

- After Selecting the features, we split our data into training and testing. And we train the machine learning model with both the benign-ware and the malware.
- After training the model we can evaluate the feature selection by the accuracy o the model for classifying an application as benign or malicious.



Thank you !

ELLY KHODAEI

References

UNIVERSITY OF
WATERLOO

FACULTY OF ENGINEERING