Hardware Performance Counters: Ready-Made vs Tailor-Made

Abraham Peedikayil Kuruvila, Anushree Mahapatra and Ramesh Karri, Kanad Basu.

ACM Transactions on Embedded Computing Systems, September 2021.

Problem Addressed in this Paper:

Modern Processor used Hardware Performance Counters (HPC) for the system-level defense to distinguish malware from benign applications. HPC are special purpose registors used to monitor low-level micro-architectural events, e.g. branch-misses, cache-misses, and number of instruction executed. Based on HPC values, application performance has been tuned at run-time. Ready-made HPC cannot adequately characterize the full unique behaviour of monitored applicaiton. They have low granularity and unable to distinguish between similar behaved application. So they propose tailor-made HPCs that capture fine-grained micro-architectural events.

- Ready-Made HPC : track the number of instructions.
- Tailor-Made HPC : monitor the number of add, load, branch, boolean, and store instructions. It also record specific events such as branch instruction followed by load or an add instruction followed by a store.

Main points addressed in this paper,

- Demonstrate that ready-made HPCs are insufficient to uniquely identify algorithm implementations
- Proposed tailer-made HPCs for the monitored algorithm to yield hardware events that can identify the algorithms with a high accuracy.
- Develop ML based classifiers to characterize algorithms using tailor-made HPCs.

Motivation for the Problem:

HPC are used to trained the classifier for the malware detection process. HPCs cannot distinguish similar patterns malware which result in classifier making false prediction for different family group of malware. Saber and mceliece algorithm have overlapping samples for the branch-instruction indicates that both have similar HPC counts. Because of overlapping, classifier distinguish power will reduce. This will affect the precission, recall and accuracy of the classifier. Since, the HPC counts of the algorithms are similar for multiple time intervals, these limitation induce inability to effectively distinguish data representations of algorithms effectively through a ML classifier.

Motivation for the Solution:

As HPCs are unable to uniquely distinguish application. We need an alternative of HPC which having more power on collecting hardware feature values. HPC counts the number of instructions irrespect of instruction details like number of branch, load, store, add, boolean instruction. These fine grained instruction can uniquely identify the behaviour of application as these values are hardly same for two application unless they are the same application.

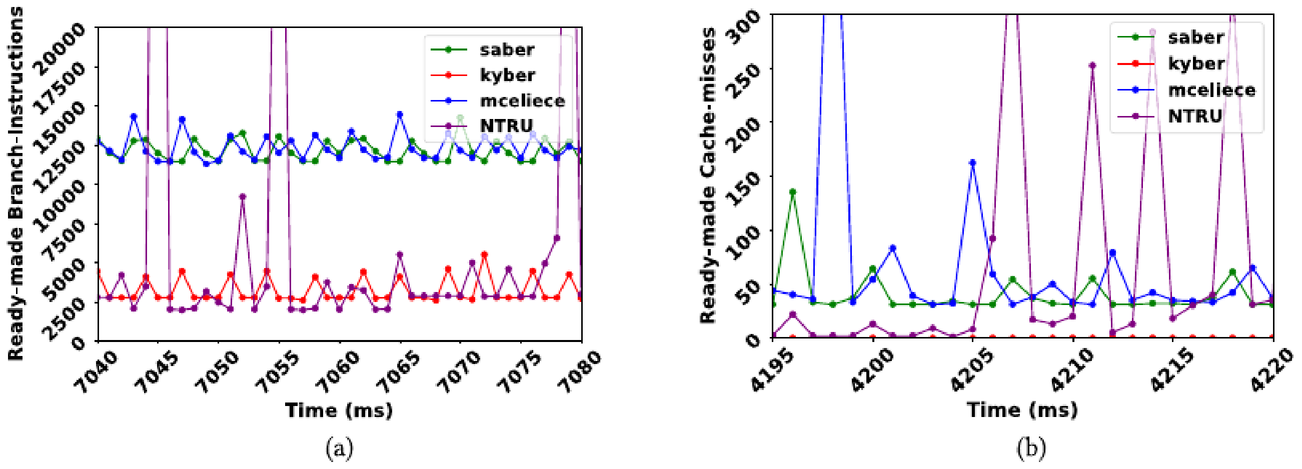

Figure 1: Ready-made HPC Traces of (a) Branch-instruction and (b) Cache-misses for NIST PQC KEM finalists.

Fig 1 shows the traces of branch instruction and cache-misses HPC values. As you can see the NTRU and Kyber traces have similar HPC counts. Crystals kyber and NTRU suffer from the same limitation. Traces have samples of similar counts for saber and mcEliece for cache-misses. These limitations induced inability to effectively distinguish data representation of algoritms.

Theory :

They proposed a Tailor-Made HPC which counts the sequences of assembly level instructions at run-time. They have captured five instruction types : branch/jump (b), load (l), store (s), arithmetic (a), and boolean (n). These instruction are common in every type of applications whether it is sorting, text editors, web browsers, malwares. They have implemented tailor-made HPCs to track the sequences of instructions in the form of "XY", where X and Y are the two instruction types and "XY" means instruction "X" is followed by instruction "Y". The tailor-made HPC feature vector is : b, l, s, a, n, ba, bl, bs, bb, bn, la, ll, ls, lb, ln, sb, sl, ss, sa, sn, ab, al, as, aa, an, nb, nl, ns, na, nn. Tailer-made HPC counts the number of these instrucitons for running applicaiton.

A disassembler is utilized to obtain the dynamic trace from the PQC binary executable. From the dynamic traces the tailor-made HPCs values are emulated using the Intel PIN tool. PIN is a dynamic binary instrumentation framework that provides a rich application programming interface to obtain assembly-level information, including instruciton and contents of the general and special purpose registers. They uses sampling interval T_s to collect the HPCs and T_s is set to 10,000 instruction. After every 10,000 instructions tailor-made HPCs are recorded and resets. They added a tailor-made HPC called program phase (pp). The pp HPC uses an instruction count register for fine-grained control on the sampling rate of the number of instrucitons counted. p*p* is incremented every T_s instructions.

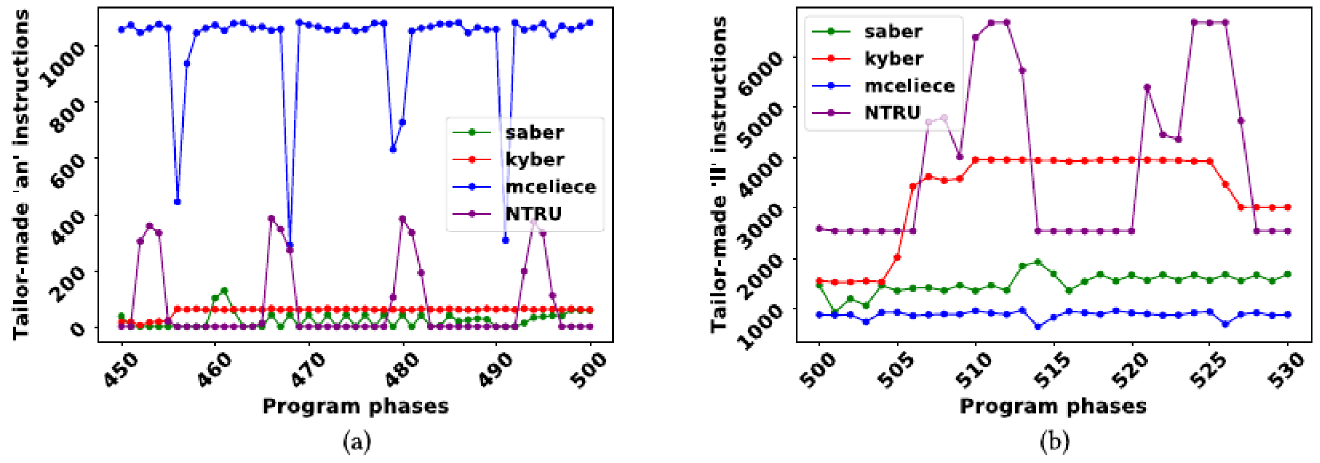

Figure 2: Tailor-made HPC traces of (a) "an" and (b) "ll" through the various phases of a program of NIST PQC KEM finalists.

In Figure 2(a), the trace of tailor-made HPC for *an* is unique to classic McEliece and distinguished it from other KEMs. *ll* instruction traces show behaviour of KEMs delineated from each other through several program phases. These distinguish traces help to increase the distinguishing power of the features for a classifier. Since tailor-made HPC can capture the unique attributes and characteristics of an application, they provide a clear separation between the algorithms. Therefore, ML model trained on tailor-made HPCs furnishes superior classifier performance relative to ready-made HPCs.

Ready-made HPCs are vulnerable to side-channel attacks to leak secret keys and confidential details of the program flow. Adversaries can monitor ready-made HPC cache events to determine a hit or miss. Since tailor-made HPC are generated be periodically sampling the dynamic assembly trace, an attacker would face a plethora of difficulties in exploiting this information to leak secret keys. It is not possible to obtain register values in the dynamic assembly instructions as specific values are stored at specific memory locations accessed through a memory address. These addresses cannot be identified from the dynamic trace.

Implementation:

They utilize Post Quantum Computing (PQC) Key Encapsulation Mechanism (KEM) algoritms as exemplar applications to highlight the benefits of tailor-made HPCs. The KEM algorithms they consider are the four NIST round-three finalists: Crystals Kyber, NTRU, Saber, and Classic McEliece. These KEM algorithms are either lattice or code-based.

They used the Intel PIN tool to collect dynamic trace for each of the PQC algoritms. They wrote a python script to sample the collected trace every 10,000 instructions and furnish the tailor-made HPCs. They trained the ML models using the scikit-learn library on this tailor-made HPC data and used

Random Forest model for the classification. They used 70% of the data for training and 30% for testing. Initially developed a Base model utilizing all the tailor-made HPCs by not constraining them to only four. We devised separte models utilizing four tailor-made HPCs selected through univariate and extra trees feature selection methods.

Critique:

Every KEM algoritm is identified using a unique subset of tailor-made HPC. Tailor-made feature trained ML classifier attained higher accuracy than ready-made HPCs. When analyzing the accuracy and recall for both feature selection methods, their highest furnished accuracy was 99.98% and the largest recall was 99.99%. ML models were competent enough to furnish a minimum of atleast 94.91% precision with just four tailor-made HPCs. These values indicate that the model has low false negatives and using tailor-made HPCs allows for successfully distinguishing KEM algorithm.

Implementation of Tailor-made HPCs has no impact on performance of processor due to the HPC unit, since it resides outside the critical path. Hardware overhead is negligible for tailor-made HPCs. We can only uitilize four feature vector at a time while using ready-made HPCs for classification. Tailor-made HPCs has no such limitation they can be extended to any number but keep as low as possible to reduce the hardware overhead. Tailor-made HPCs can certainly extended to include triple or quadruple assembly combinations. This will be beneficial when profiling complex algorithms.

Related works:

Many work has shown the studies of classification of malware by capturing low-level micro-architectural events. Researchers used various ML models, such as Neural Networks and K-Nearest Neighbors, trained on HPCs for distinguishing implementations. They proposed 2SmaRT techinque, that selects the best ready-made HPCs via feature selection and utilize a two-stage classifier for malware detection. Furthermore, HPCMalHunter was developed to produce behavioral vectors for applications through ready-made HPCs for real-time malware detection. Behavior-based Adaptive Intrusion detection in Networks was developed and proposed. This framework utilizes ready-made HPCs to amalgamate network data, including statistics as well as profiled application information in order to identify Denial of Service and Distributed Denial of Service attacks. A comparison of classification capabilities between ML models and the impact of the classifier parameters for Malware detection was presented.