2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA) | 978-1-7281-8470-8/20/$31.00 ©2020 IEEE | DOI: 10.1109/ICMLA51294.2020.00101

# HybriDG: Hybrid Dynamic Time Warping and Gaussian Distribution Model for Detecting Emerging Zero-Day Microarchitectural Side-Channel Attacks

Han Wang[1], Hossein Sayadi[2], Avesta Sasan[3], Setareh Rafatirad[3], and Houman Homayoun[1]
[1]University of California, Davis, CA, USA
[2]California State University, Long Beach, CA, USA
[3]George Mason University, Fairfax, VA, USA
[1]{hjlwang, hhomayoun}@ucdavis.edu, [2]{hossein.sayadi}@csulb.edu,[3]{asasan, srafatir}@gmu.edu

***Abstract*—Microarchitectural Side-channel Attacks (SCAs) benefit from emerging hardware vulnerabilities in modern microprocessors to steal critical information from users, posing great security threats to computer systems. Several recent studies have focused on using low-level features captured from built-in Hardware Performance Counter (HPC) registers to implement accurate Machine Learning (ML)-based SCAs detectors. Nonetheless, existing ML-based SCAs detectors required prior knowledge of attacks applications to detect the pattern of side-channel attacks using a variety of microarchitectural features. In particular, the existing solutions have ignored to address the challenge of detecting sophisticated unknown (zero-day) SCAs at run-time which is a more challenging issue in today's computer systems. In addition, prior works analyzed a limited number of ML classifiers without thoroughly evaluating the detection effectiveness and computational complexity of the detectors. In response, we propose *HybriDG*, a hybrid lightweight model consisting of Dynamic Time Warping (DTW) followed by a Gaussian distribution model to accurately detect both known and unknown emerging SCAs at run-time. Our experimental results demonstrate that *HybriDG* achieves 100% detection accuracy for known attacks and 99.5% detection accuracy for unknown attacks which is significantly outperforming traditional ML algorithms, deep learning, and time series classification models by up to 80% for unknown and 8% known attack detection.**

***Index Terms*—Side-Channel Attacks, Dynamic Time Warping, Gaussian Distribution, Zero-Day Attacks, Detection**

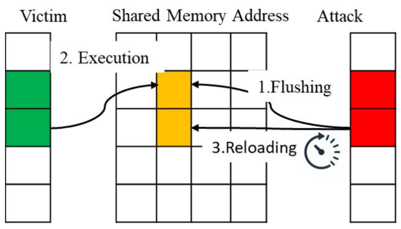
## I. Introduction

The past decades have witnessed a significant boost in the complexity of computing systems to support the increasing computation and performance demand. To this aim, various components (e.g., cache memories, branch predictors, out-of-order execution units) are implemented in processors architectures to minimize the CPU stalls and enhance the performance. Despite the provided performance benefits, these solutions could cause new microarchitectural vulnerabilities that have been exploited by new types of attacks that observe side-channel information by causing interference. Microarchitectural Side-Channel Attacks (SCAs) primarily exploit hardware vulnerabilities to infer sensitive and confidential information [11], [37]. The proliferation of computing devices in mobile computing and Internet-of-Things (IoT) domains further exacerbates the impact of emerging cybersecurity threats implying the necessity of protecting legitimate users from these attacks. Hence, there exists an emerging need to address the security risks and challenges posed by such harmful attacks, calling for effective low-cost security countermeasure which can accurately identify SCAs with minor overhead.

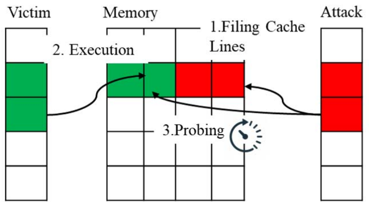
To address the SCAs issues, some prior works [34], [33], [30], [29] propose mitigation methods to alleviate the influence of side-channel attacks. However, such solutions either require extra hardware design and/or are only effective to a subset of SCAs, ignoring the impact of zero-day attacks. Other studies [8], [39], [21], [9], [6], [3], [31], [32], [28] propose the use of microarchitectural pattern analysis captured through Hardware Performance Counters (HPCs) to detect the existence of side-channel attacks. For instance, [8] offers an HPC monitoring model of both victim and attack applications to detect the SCAs. Based on the obtained HPCs, the HPC events of victims' and attacks' traces are correlated. Similarly, in [39] the authors present CloudRadar framework which aims at detecting cross-VM SCAs by making use of HPC patterns. This work comprises two phases: a) identifying sensitive applications by comparing offline generated signatures based on HPCs, and b) correlating victim HPCs and attack HPCs.

While there has been much progress in terms of identifying various SCAs in the last few years, there are still two major challenges involved with existing SCAs detectors. First, the fast-paced development of the emerging SCAs to circumvent the current detection techniques has not been properly addressed. Hence, the existing methods are not able to detect the unknown (zero-day) attacks. Secondly, existing works on SCAs detection have primarily focused on one or a few ML techniques for attack detection [8], [39], [21]. Such an analysis leaves a void in terms of performance of attack detection, as various ML classifiers yield different performance in detecting various types of attacks [27], [22], [32].

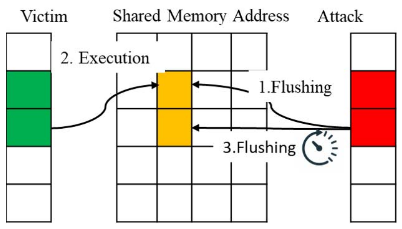
In order to solve the aforementioned challenges, in this work we propose *HybriDG*, a hybrid Dynamic Time Warping (DTW) and Gaussian distribution model to accurately detect both known and unknown emerging side-channel attacks at run-time. *HybriDG* employs DTW time-series classification to calculate the similarities of victim under no attack (VNA) and victim under attack (VA) HPCs traces and then applies Gaussian distribution to set a threshold of the similarities based on the optimal false alarm rate. During the testing part, only victim HPCs features are collected and fed to



(a) Flush+Reload



(b) Prime+Probe



(c) Flush+Flush

Fig. 1: Working principle of three emerging cache-based side-channel attacks

the dynamic time warping model. Lastly, the distance from DTW is compared with the threshold decided by Gaussian distribution to detect if the victim application is under attack or under no attack condition. *HybriDG* shows a significantly high detection accuracy for both known and unknown attacks with a low false positive rate eliminating the need to collect the attacks' HPCs data for training the SCAs detector.

## II. Background

In this section, we introduce the background on microarchitectural side-channel attacks and different types of ML classifiers that are used for attacks detection.

### A. Side-channel Attacks

The emergence of different hardware components such as cache memory, branch predictor, etc. to enhance the modern microprocessors' performance, have led to the exposure of new hardware vulnerabilities in the systems. Such exposure makes a unique opportunity to spy on victim applications and infer sensitive information by side-channel attacks.

*1) Flush+Reload:* The researches in [36], [7] exploit the vulnerability of page de-duplication technique by monitoring the memory access lines in the shared pages. This attack targets the Last-Level Cache in the CPU and flushes out victim applications' data in the cache and waits for the victim application to execute, as shown in Figure 1-(a). After flushing the cache, the attacker attempts to access the data and measures the accessing time (latency). Shorter accessing time denotes that the victim application has accessed the data; otherwise, it has not been accessed.

*2) Prime+Probe:* Without the memory de-duplication restriction, Prime+Probe [19] attack could be applied to a wider range of systems. This type of SCA consists of two different stages: Prime and Probe as shown in Figure 1-(b). In the Prime stage, the attacker builds the eviction sets which are grouped cache sets causing potential conflict with victim applications, and then fills the cache with the eviction sets. Next, the attacker waits for the victim applications' execution and then re-accesses the eviction sets (Probe stage). If the accessing time is long enough, it means that the victim application has accessed the data.

*3) Flush+Flush:* This SCA relies on the execution time of the flush instruction [11]. Unlike prior attacks, Flush-Flush does not make any memory accesses, nor does it rely on the data's access latency. The execution time of flush instruction depends on whether the data is stored in the cache. Flush-Flush uses the execution time of the subsequent flush instruction following the victim application's execution, as shown in Figure 1-(c). The longer execution time of the flush instruction indicates that the corresponding data was brought to the cache and later was accessed by the victim application.

### B. Time-series Classification

Since our work proposes a time-series classification based on HPCs data to detect the SCAs, we will introduce dynamic time warping and compare it with other time-series classification methods. There are several methods proposed for mining the time-series data including the dynamic time warping (DTW) [20], Shaplet [38], [12], Bag of Patterns(BOP) [18], Symbolic Aggregate approXimation (SAX) [17], etc. DTW calculates the distance between two time-series sequences and attempts to align two different sequences while giving the optimal one with the shortest distances. As a result, DTW achieves the measurement of the similarity between temporal sequences with different speed. SAX is another time-series algorithm that first transforms data into the Piecewise Aggregate Approximation (PAA) representation, which will be symbolized into a sequence of discrete strings. Next, SAX deploys an approximate distance function that lower bounds the Euclidean distance. Shaplet algorithm introduces a new concept called "shaplet" a sub-sequence extracted from one of the time-series sequences. The Shaplet algorithm selects the shaplet according to the ability of the shaplet to classify time-series data.

### C. Deep Learning

For the comprehensive analysis of the effectiveness of our proposed detector, we further examine Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) recurrent neural networks for detecting SCAs based on HPCs temporal traces.

*1) Fully Convolutional Neural Network (FCN):* A convolutional layer in a deep neural network learns patterns of local structure in the input signal and can further learn feature representations over a sequence of input data [15]. In our implementation, the final layer of a CNN classifies an input sequence as "under no attack" or "under attack" as a function of the HPCs detected over the entire sequence. The HPCs are translated into integer values in a one-dimensional vector, and are then processed by a CNN with one-dimensional convolutional layers. FCN is based on the convolutional neural network (CNN) technique where models employ continuous convolution layers to extract time-series features.

*2) Long Short-Term Memory(LSTM):* For the LSTM [13], each HPC trace includes a time-ordered sequence of HPCs on which an LSTM network detects temporal patterns that are important for discriminating between ”under attack” and ”under no attack”. To this aim, one and two layers of LSTM neurons with various numbers of neurons per layer were explored. Each node learns a different sequence pattern and a collection of sequence pattern detectors from all the nodes connected to the output layer is used to classify each HPC temporal sequence.

TABLE I: Selected Monitoring HPCs List

| | |
|---|---|
| L1 HIT | L1 MISSES |
| L2 HIT | L2 MISSES |
| L3 HIT | L3 MISSES |
| All BRANCHES RETIRED | BRANCHES MISPREDICTED |

### *D. Hardware Performance Counters (HPCs) Data*

Hardware performance counters are special-purpose registers built-in modern microprocessors that count hardware-related events such as instructions executed, cache misses suffered, or branches mispredicted [27], [10], [28]. HPCs data are extensively used to auto-tune/profile applications [16], analyze and optimize performance [5], malware detection [27], [26] and SCAs detection [23], [32], [31].

This work only employs victim applications’ HPCs and collects HPCs data by separate processing cores instead of counting them by application’s thread. For this purpose, a PAPI-based HPCs monitoring tool is employed and one HPC sample is collected every 50 microseconds. We have collected 8 HPC events listed in Table I to train our detector. These HPCs are selected based on the attacks’ design methodology influencing the cache and branch predictor units. As mentioned in Section I, both HPCs of victim under no attack and victim under attack are needed to collect. Hence, victim applications are first executed solely to obtain victim under no attack data. And then, victim and attack applications mentioned in Section IV-A are executed concurrently to build victim under attack dataset.

## III. Proposed Methodology

In this section, we present the details of our proposed *HybriDG* model to detect the known and unknown side-channel attacks. Figure 4 depicts an overview of the proposed *HybriDG*. As seen, the proposed *HybriDG* detector contains two major parts: a) offline data collection and HPC features evaluation, b) distance threshold determination (T) with dynamic time warping and Gaussian distribution, and online prediction with a threshold (T) that each part is discussed in details below.

### *A. Threat Model*

The proposed *HybriDG* is designed to secure multi-core computing processors against SCAs that employ inclusive and

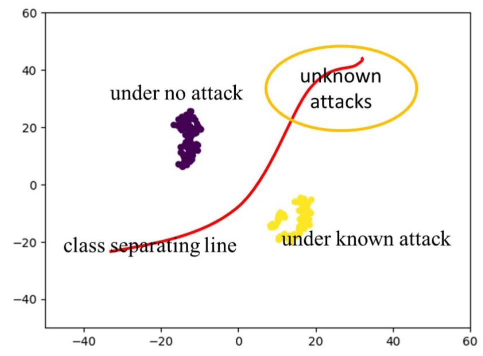


Fig. 2: TSNE plot for victim under no attack and victim under known attacks samples

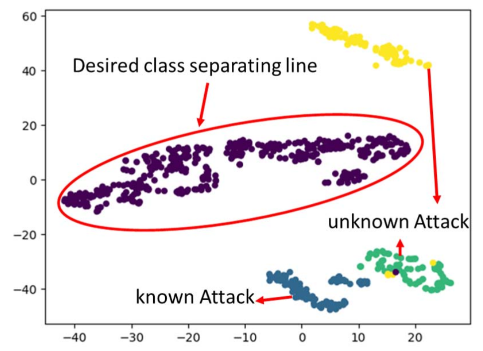


Fig. 3: TSNE plot with desired classifying line for victim under no attack, known attack, and unknown attack samples

non-inclusive caches on the Intel processor architectures, in which malicious and benign processes use shared libraries. In addition, it considers that the applications are executing in a Linux-based single OS environment, where both victims and attacks reside in the physical machine, on different processing cores. The system can face with different types of SCAs including shared-memory based SCAs such as Flush+Flush, Flush+Reload, and none shared-memory based SCAs such as Prime+Probe. In order to create the known and unknown threat conditions, two out of the three considered SCAs are used as known attacks, and the third attack is used for modeling unknown attacks. As a result, the known attacks can be profiled, and the corresponding run-time HPCs information is stored in database for training ML models. And the unknown attack is deployed to test the effectiveness of ML-based detectors built in the proposed *HybriDG* in detecting zero-day SCAs.

### *B. Classifying Unknown Dataset*

As shown in Figure 2, victim under no attack and under known attacks temporal sequences are plotted using TSNE algorithm. It can be observed that under no attack and under

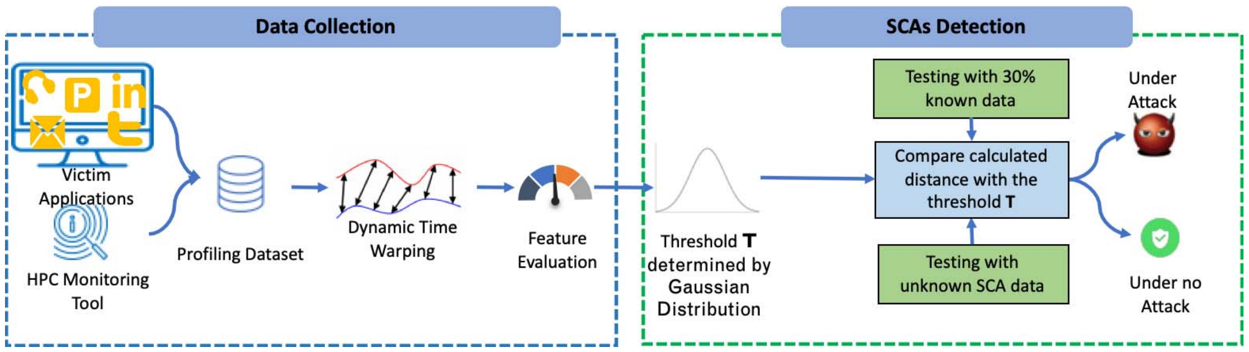

Data Collection
SCAs Detection
Victim Applications
HPC Monitoring Tool
Profiling Dataset
Dynamic Time Warping
Feature Evaluation
Threshold **T** determined by Gaussian Distribution
Testing with 30% known data
Compare calculated distance with the threshold **T**
Testing with unknown SCA data
Under Attack
Under no Attack

Fig. 4: Overview of *HybriDG*, the proposed hybrid Model for detecting emerging zero-day SCAs

known attacks samples can be easily separated. In addition, to conduct binary classification, prior classification models which are trained with under no attacks and under known attacks dataset could construct a line shown in Figure 2, which separates samples into two classes. However, unknown attacks might locate on both sides of the line, which results in the misleading and accuracy degradation of the ML classifiers. As a result, to achieve a high detection accuracy, we need to construct a classification line, as shown in Figure 3, which defines the threshold of ”under no attack” and any samples outside the line is classified as ”under attack”.

## C. Dynamic Time Warping (DTW)

DTW was introduced into classification problems and time-series mining by Berndt and Clifford [2], where dynamic programming was used to align sequences with different lengths. The main idea of DTW is to find an optimal match between two sequences by allowing a nonlinear mapping of one sequence to the other sequence and minimizing the distance between two sequences. Considering two HPC temporal sequences A and B as follows: A(a1, a2,..., an) and B(b1, b2,..., bm), DTW finds an optimal warping path between A and B by using dynamic programming to calculate the minimum cumulative distance $\gamma$ (n, m), where $\gamma$ (i, j) is defined in Equation 1:

$$\gamma(i,j) = (ai, bj)^2 + \min \begin{cases} \gamma(i-1,j) \\ \gamma(i-1,j-1) \\ \gamma(i,j-1) \end{cases} \quad (1)$$

This allows DTW to practically match similar shape sequences together, even though the sequences may be shifted or out of phase, and this qualification has further made DTW viable for a multitude of time-series classification problems. Keogh and Pazzani [14] considered replacing the value of each data point with the first derivative within the dynamic time warping distance function to resolve such a problem.

In this work, DTW is employed to calculate the similarities among victim under no attack HPC sequences and among victim under no attack and attack HPC sequences. For the following sections, the two distances calculated when victim application are under no attack and attack are abbreviated as VNAD (Victim under No Attack Distance) and VAD (Victim under Attack Distances).

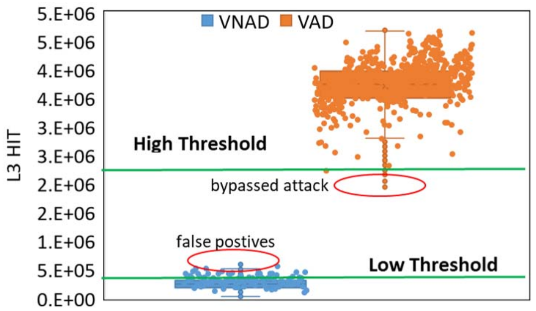


Fig. 5: Different threshold influences(VNAD: victim under no attack distances; VAD: victim under attack distances.)

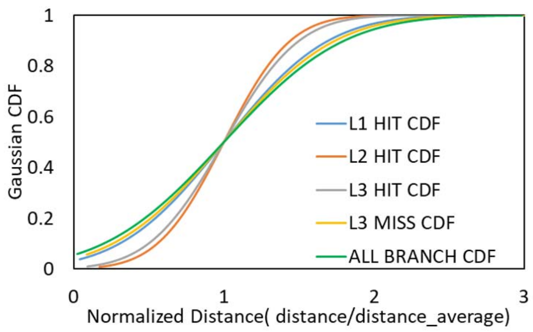


Fig. 6: Gaussian distribution of various HPCs temporal traces

## D. HPCs Evaluation

As mentioned, a threshold of distances calculated by DTW needs to be decided. As shown in Figure 5, if a large threshold value is selected, some attacks could potentially bypass the detector. Whereas, if the threshold is too small, then some ”under no attack” samples could be classified as ”under attack”, leading to a higher number of false positives in the classification result. Hence, the goal here is to carefully set a confident threshold that not only could divide VA and VNA sequences, but also is strict enough for detecting further unknown attacks. To this aim, we should choose the HPC features whose temporal sequences' distances determined by DTW are distributed densely with the smallest possible variance. For this purpose, the CDF (cumulative distribution) of Gaussian distribution is used to plot the distances of HPCs sequences, as shown in Figure 6. Each HPC distances are normalized with the HPC's average distance to eliminate the influence of different HPCs value ranges. The Figure 6 shows that the L2 HIT feature value converges to 1 faster than other features indicating that by setting the same threshold, the L2 HIT can achieve higher VNA predicted correctly as "under no attack" compare to other HPC features. Table III-D further shows the theoretical false positives when using 1.5 as a threshold across different HPC features. It can be found that the L2 HIT could achieve higher VNA detection accuracy and lower false positive rate. Therefore, in this work, we select the L2 HIT feature for building the classifiers according to the methodology mentioned above.

TABLE III: Theoretical false positive rate and corresponding L2 threshold value

| Threshold Setting | Theoretical False Positive Rate | L2 Threshold |
|---|---|---|
| $\mu$+$\sigma$ | 1/10 | 263245 |
| $\mu$+2$\sigma$ | 1/100 | 330946 |
| $\mu$+3$\sigma$ | 1/1000 | 398647 |
| $\mu$+4$\sigma$ | 1/10000 | 398647 |
| $\mu$+5$\sigma$ | 1/100000 | 398647 |
| $\mu$+6$\sigma$ | 1/1000000 | 601752 |

## E. Gaussian Process for Threshold Determination

After choosing the most suitable HPC feature, the threshold needs to be set. The Gaussian distribution of HPCs temporal traces' distance value can estimate the percentage of points with a larger distance value than a certain threshold, which is a false positive rate. For example, the percentage of points with a value larger than $(\mu+\sigma)$ is 10%. Hence, the theoretical false positive rate is 10% when the threshold is $(\mu + \sigma)$ according to equation 2. In this work, we also study different thresholds that influence the final prediction result. The details of different thresholds are listed in Table III based on the HPC choice of L2 HIT. As shown in Figure 5, the smaller the threshold is, the higher the theoretical false positive rate is, and the larger the threshold is, the higher the possibility of missing "under attack" detection is. Therefore, choosing an optimal value to meet the false positive rate requirement and maintain high "under attack" detection accuracy at the same time is crucial. In this work, we choose $(\mu+3\sigma)$ as a threshold to evaluate the proposed approach, meaning that the theoretical false positive rate is considered as 0.001.

$$f(x) = \frac{1}{\sqrt{(2\pi\sigma)}} * e^{\frac{(x-\mu)}{2\sigma^2}} \tag{2}$$

## IV. Experimental Results and Evaluation

In order to comprehensively evaluate the effectiveness of the proposed detector various classification algorithms from three main categories of ML are selected and compared with *HybriDG* in terms of detection accuracy for known and unknown attacks, detection latency, and efficiency analysis. The selected ML classifiers are listed in Table IV. As shown, for the thorough analysis of suitability of standard ML techniques for zero-day SCAs, we have implemented different types of classifiers, including traditional classifiers (tree-based J48, support vector machine based SGD, rule-based OneR), Deep learning classifier (LSTM, FCN, LSTM-FCN, ALSTM-FCN), and Time-series classifier (DTW, BOPF, Shaplet).

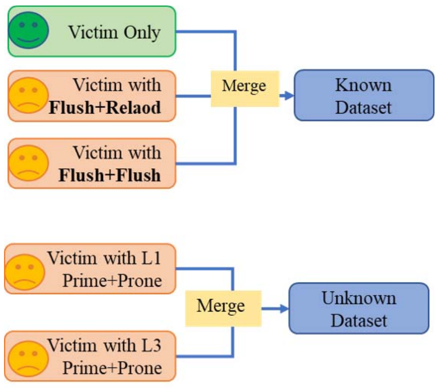


Fig. 7: Testing datasets

## A. Experimental Setup

The experiments are conducted on an Intel i5-3470 desktop with four cores and 8GB DRAM, three-level cache system. In this on-chip cache memory subsystem, while L1 and L2 caches are exclusively separated, L3 cache memory is inclusive and shared among all cores meaning that flushing out the data in L3 cache will also remove the data in L1 providing vulnerability for LLC cache attacks to be exploited. In addition, the operating system is Ubuntu 16.0.4 LST with Linux kernel 4.13. In order to perform our experiments, AES and RSA are used as victim applications, and Flush+Reload, Flush+Flush, Prime+Probe are used as attacks. In order to evaluate the effectiveness of detecting unknown attacks, Flush+Reload and Flush+Reload are used as known attacks, and the remaining applications are used as unknown attacks that are not included in the training dataset.

## B. Testing Dataset

As introduced in Section III, victim under no attack and under attack sequences are fed into the DTW model, and a threshold determines whether the victim is under attack or not. To model real-world application scenarios, all SCA applications experimented in this work are divided into known attacks and unknown attack sets. To this aim, as illustrated in Figure 7, victim only and victim with known attacks form the known dataset, and 70% of the known dataset is then employed as a training dataset to choose the most effective HPC and determine the threshold. And the remaining 30% of the reference dataset is used as a testing dataset to model unknown zero-day attacks. Therefore, the testing results are

TABLE IV: Selected classifiers

| Traditional Classifier | | | Time-series Classifier | | | Deep Learning | | | | Proposed Classifier |
|---|---|---|---|---|---|---|---|---|---|---|
| J48 | SGD | OneR | 1NN-DTW | BOPF | Shapelet | LSTM | FCN | LSTM-FCN | ALSTM-FCN | *HybriDG* |

presented in terms of detection accuracy for the known dataset and the unknown dataset.

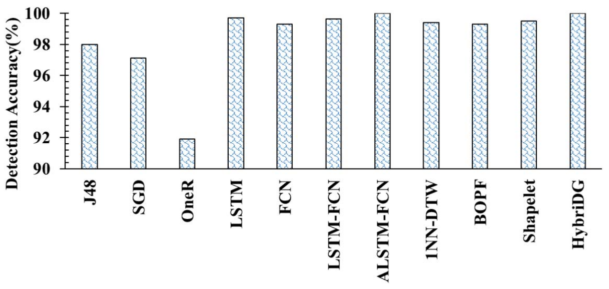


Fig. 8: Known attacks detection accuracy of various classifiers

### C. Known Dataset Testing Results

*1) Detection Accuracy:* Figure 8 presents the detection accuracy (rate of the correctly classified samples) of all evaluated and proposed classifiers. It can be observed that traditional classifiers, deep learning-based classifiers, and time-series based classifiers can achieve above 90% detection accuracy for the known dataset. This observation validates the plot given by TSNE algorithm in Figure 2, which shows that "under no attack" and "under known attacks" samples are located separately and can be easily classified by means of standard ML algorithms. The distribution in Figure 2 demonstrates that even by applying a simple classifier like rule-based OneR we could achieve 92% detection accuracy with only 1 HPC feature. It is also noticeable that a more complex deep learning and timer-series based classifiers outperform the traditional classifiers, achieving above 99% accuracy. Among all implemented ML-sbased detectors, the proposed detector *HybriDG* obtains the highest detection accuracy of 100%.

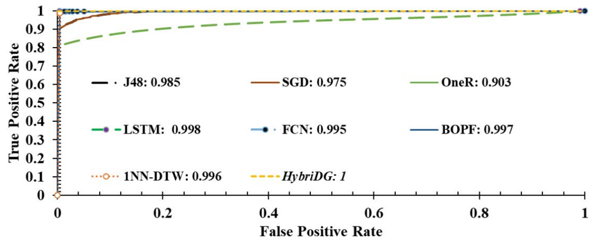


Fig. 9: ROC Curve and AUC value of various classifiers

*2) Classification Robustness:* Receiver Operating Characteristics (ROC) Curve is produced by plotting the fraction of true positives rate versus the fraction of false positives for a binary classifier. The best possible classifier would thus yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing 0% false positives and 100% true positives. The Area under the ROC Curve (AUC) metric corresponds to the probability of correctly identifying "under attack" and "under no attack" and robustness is referred to how well the classifier distinguishes between the two classes, for all possible threshold values. Higher AUC indicates better robustness for ML classifiers. Due to space limitation, in Figure 9 we show the results for two classifiers from deep learning and time-series categories and compare them with traditional and the proposed *HybriDG* detector using the ROC Curve and corresponding AUC values. Similar to detection accuracy, it can be observed in Figure 9 that the *HybriDG* outperforms all other classifiers in terms of AUC, having the highest AUC and smallest distance to point (0,1) in ROC curve. Both deep learning and time-series classifiers yield to high robustness as well with slightly lower AUC value. It is also notable among three traditional classifiers, the rule-based OneR model delivers the lowest AUC value.

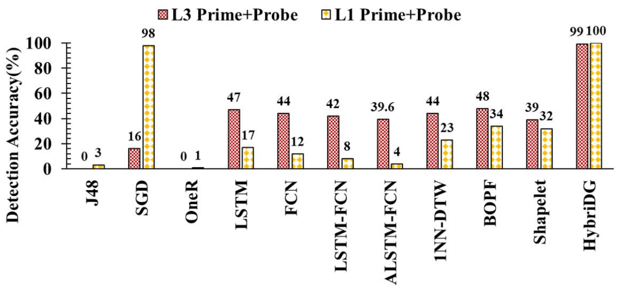


Fig. 10: Unknown attacks detection accuracy

### D. Unknown Dataset Testing Results

The unknown dataset contains victim under unknown attacks only (L3/L1 Prime+Probe) and is used for testing if the proposed SCAs detector is able to accurately detect zero-day attacks or not. As shown in Figure 10, L3 Prime+Probe and L1 Prime+Probe are unknown attacks, and the detection accuracy for each of them are demonstrated. As seen, the proposed *HybriDG* is substantially outperforming all other ML classification models achieving 100% for L1 Prime+Probe and 99% accuracy for L3 Prime+Probe detection. Moreover, one could observe that most of the experimented ML classifiers are not able to detect unknown attacks with high accuracy, delivering less than 50% detection accuracy, except SGD (77.8% and 98% for L3 Prime+Probe and L1 Prime+Probe, respectively). Though SGD shows better detection performance compare to the rest, it is still significantly less accurate and robust than the *HybriDG* detector highlighting the effectiveness of our proposed hybrid model for detecting emerging zero-day SCAs.

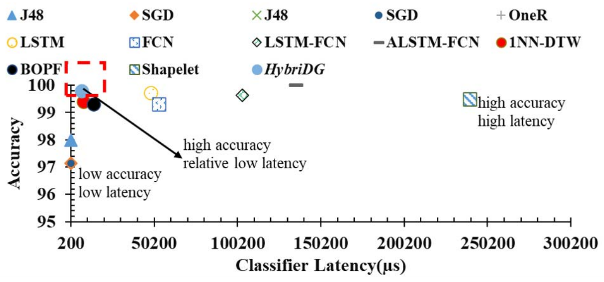


Fig. 11: Efficiency comparison among various classifiers for known dataset

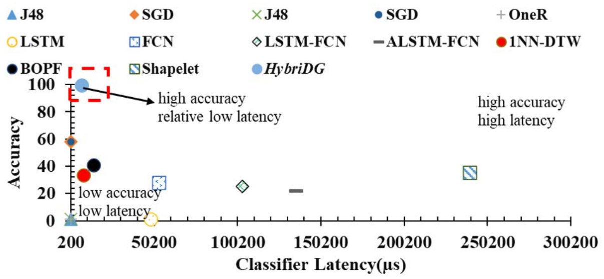


Fig. 12: Efficiency comparison among various classifiers for unknown dataset

### *E. Efficiency Analysis: Accuracy vs. Latency*

Figure 11 and Figure 12 present the trade-off analysis between detection accuracy and detection latency (computational costs) for known and unknown datasets, respectively. In Figure 12, accuracy is the average detection accuracy of L1 Prime+Probe and L3 Prime+Probe. It can be found that the proposed *HybriDG* achieves high detection accuracy (100% and 99.5%) with relatively low latency compared to deep learning or time-series classifiers for both known and unknown datasets. Although traditional classifiers, like SGD can also achieve high accuracy (around 98%) for known dataset with lowest latency, its accuracy for unknown dataset (zero-day SCAs) is significantly low (only 57%), which still exposes computer systems to potential new attacks and security exploits. Also, while deep learning and time-series classifiers achieve high detection accuracy for the known dataset, they lack the ability to detect unknown SCAs with high accuracy and further result in higher latency than the *HybriDG* detector.

## V. Relevant Works

SCADET [25] proposes to analyze the executable binary file and detect if there is any potential risk for Prime Probe attacks. It shows 100% attacks detection with 7.4% false positive rate. However, it only evaluated with Prime Probe and analysis time at the granularity of minute. CaSym [4] first gets a program from a compiler IR and performs symbolic execution. Then, it monitors the programs and cache status. The cache status is used to build a formula that is fed to SMT solver to analyze whether side-channels exist. It can help to identify whether cartographic applications have side-channel vulnerabilities and then apply preloading to mitigate such vulnerabilities.

Prior detection work [1] monitors HPCs trace of both victim and attack processes and compares the effectiveness of three ML classifiers: neural network, decision tree C4.5, and K nearest neighbors. The work in [24] proposes a detection system containing one analytic server and one or more monitored computing devices to detect SCAs, including Spectre and Meltdown. The analytic server receives HPCs data from monitored devices and identifies suspicious core activity. Once captured, application level monitor will be deployed on the computing devices taking corrective actions as soon as finding suspicious application activity. Recent work [35] uses cache latency to build cache occupancy of victims and attacks in which based on the cache occupancy relation of the two processes, SCAs can be deduced.

Chiappetta et al. [8] monitors applications' HPCs at microsecond scale and then compares the effectiveness of three different machine learning-based methods to detect SCAs at real-time with HPC features including finding a correlation between victims and attacks, building supervised machine learning models based on HPCs from victims and attacks, and detecting anomalies by validating attack HPCs as samples and other processes as outliers. However, HPCs data from attacks could not be collected due to embedded execution of attacks within benign applications. CloudRadar [39] first identifies whether an application running in victim virtual machine is a cryptographic application by measuring distances between HPCs temporal traces of running applications and cryptographic applications stored in training database. Once identified as sensitive applications, HPCs from untrusted VM are correlated with the protected VM's applications. While both works have reported high detection performance for known attacks, they rely on the knowledge of attacks and have ignored to address the challenge of detecting zero-day side-channel attacks which is a more challenging security threat.

## VI. Conclusion

In this work, we propose a hybrid Dynamic Time Warping (DTW) and Gaussian distribution model called *HybriDG* to accurately detect both known and unknown emerging microarchitectural side-channel attacks at run-time. The proposed methodology calculates the similarities between victim HPCs traces collected from two conditions: victim under attack and victim under no attack which are indicated by distances from the DTW model. Then, according to the distribution of victims under no attacks and victims under known attacks, only one HPC feature is selected, and the optimal threshold is identified according to the estimated false positive rate. After setting the threshold, the HPC temporal sequences will be fed into the DTW model to calculate the distance, which is compared with the threshold to identify whether a victim is under attack or not. Our comprehensive evaluation indicates that *HybriDG* achieves 100% detection accuracy for known attacks and 99.5% detection accuracy for unknown attacks using the most prominent microarchitectural feature (L2 HIT) outperforming standard ML models by up to 80% for unknown and 8% known attack detection.

## ACKNOWLEDGMENT

This research was supported in part by DARPA SSITH program under the award number 97774952.

## REFERENCES

[1] ALLAF, Z., ADDA, M., AND GEGOV, A. A comparison study on flush+ reload and prime+ probe attacks on aes using machine learning approaches. In *UK Workshop on Computational Intelligence* (2017), Springer, pp. 203–213.

[2] BERNDT, D. J., AND CLIFFORD, J. Using dynamic time warping to find patterns in time series. In *KDD workshop* (1994), vol. 10, Seattle, WA, pp. 359–370.

[3] BRIONGOS, S., IRAZOQUI, G., MALAGÓN, P., AND EISENBARTH, T. Cacheshield: Detecting cache attacks through self-observation. In *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy* (2018), pp. 224–235.

[4] BROTZMAN, R., AND ET. AL. Casym: Cache aware symbolic execution for side channel detection and mitigation. In *CaSym: Cache Aware Symbolic Execution for Side Channel Detection and Mitigation*, IEEE, p. 0.

[5] CHEN, D., VACHHARAJANI, N., HUNDT, R., LIAO, S.-W., RAMASAMY, V., YUAN, P., CHEN, W., AND ZHENG, W. Taming hardware event samples for fdo compilation. In *Proceedings of the 8th annual IEEE/ACM international symposium on Code generation and optimization* (2010), ACM, pp. 42–52.

[6] CHEN, S., ZHANG, X., REITER, M. K., AND ZHANG, Y. Detecting privileged side-channel attacks in shielded execution with déjá vu. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security* (2017), ACM, pp. 7–18.

[7] CHIAPPETTA, M., SAVAS, E., AND YILMAZ, C. Xlate: https://www.vusec.net/projects/xlate/.

[8] CHIAPPETTA, M., SAVAS, E., AND YILMAZ, C. Real time detection of cache-based side-channel attacks using hardware performance counters. *Applied Soft Computing 49* (2016), 1162–1174.

[9] DEPOIX, J., AND ALTMEYER, P. Detecting spectre attacks by identifying cache side-channel attacks using machine learning. *Advanced Microkernel Operating Systems* (2018), 75.

[10] DINAKARRAO, S. M. P., AMBERKAR, S., BHAT, S., DHAVLLE, A., SAYADI, H., SASAN, A., HOMAYOUN, H., AND RAFATIRAD, S. Adversarial attack on microarchitectural events based malware detectors. In *Proceedings of the 56th Annual Design Automation Conference 2019* (2019), pp. 1–6.

[11] GRUSS, D., AND ET.AL. Flush+ flush: a fast and stealthy cache attack. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment* (2016).

[12] HILLS, J., LINES, J., BARANAUSKAS, E., MAPP, J., AND BAGNALL, A. Classification of time series by shapelet transformation. *Data Mining and Knowledge Discovery 28*, 4 (2014), 851–881.

[13] KARIM, F., MAJUMDAR, S., DARABI, H., AND CHEN, S. Lstm fully convolutional networks for time series classification. *IEEE access 6* (2017), 1662–1669.

[14] KEOGH, E. J., AND PAZZANI, M. J. Derivative dynamic time warping. In *Proceedings of the 2001 SIAM international conference on data mining* (2001), SIAM, pp. 1–11.

[15] LECUN, Y., BENGIO, Y., ET AL. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks 3361*, 10 (1995), 1995.

[16] LIM, R., CARRILLO-CISNEROS, D., ALKOWAILEET, W., AND SCHERSON, I. Computationally efficient multiplexing of events on hardware counters. In *Linux Symposium* (2014), Citeseer, pp. 101–110.

[17] LIN, J., KEOGH, E., WEI, L., AND LONARDI, S. Experiencing sax: a novel symbolic representation of time series. *Data Mining and knowledge discovery 15*, 2 (2007), 107–144.

[18] LIN, J., AND LI, Y. Finding structural similarity in time series data using bag-of-patterns representation. In *International Conference on Scientific and Statistical Database Management* (2009), Springer, pp. 461–477.

[19] LIU, F., AND ET.AL. Last-level cache side-channel attacks are practical. In *SP* (2015), IEEE, pp. 605–622.

[20] MÜLLER, M. Dynamic time warping. *Information retrieval for music and motion* (2007), 69–84.

[21] MUSHTAQ, M., ET AL. Nights-watch: A cache-based side-channel intrusion detector using hardware performance counters. In *Proceedings of the 7th International Workshop on Hardware and Architectural Support for Security and Privacy* (2018), ACM, p. 1.

[22] NARUDIN, F. A., ET AL. Evaluation of machine learning classifiers for mobile malware detection. *Soft Computing* (2016), 343–357.

[23] NOMANI, J., AND SZEFER, J. Predicting program phases and defending against side-channel attacks using hardware performance counters. In *Proceedings of the Fourth Workshop on Hardware and Architectural Support for Security and Privacy* (2015), ACM, p. 9.

[24] PRADA, I., IGUAL, F. D., AND OLCOZ, K. Detecting time-fragmented cache attacks against aes using performance monitoring counters. *arXiv preprint arXiv:1904.11268* (2019).

[25] SABBAGH, M., FEI, Y., WAHL, T., AND DING, A. A. Scadet: A side-channel attack detection tool for tracking prime-probe. In *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)* (2018), IEEE, pp. 1–8.

[26] SAYADI, H., GAO, Y., MOHAMMADI MAKRANI, H., MOHSENIN, T., SASAN, A., RAFATIRAD, S., LIN, J., AND HOMAYOUN, H. Stealthminer: Specialized time series machine learning for run-time stealthy malware detection based on microarchitectural features. In *Proceedings of the 2020 on Great Lakes Symposium on VLSI* (New York, NY, USA, 2020), GLSVLSI '20, Association for Computing Machinery, p. 175–180.

[27] SAYADI, H., PATEL, N., PD, S. M., SASAN, A., RAFATIRAD, S., AND HOMAYOUN, H. Ensemble learning for effective run-time hardware-based malware detection: A comprehensive analysis and classification. In *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)* (2018), IEEE, pp. 1–6.

[28] SAYADI, H., WANG, H., MIARI, T., MAKRANI, H. M., ALIASGARI, M., RAFATIRAD, S., AND HOMAYOUN, H. Recent advancements in microarchitectural security: Review of machine learning countermeasures. In *2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS)* (2020), IEEE, pp. 949–952.

[29] WANG, H., SAYADI, H., MOHSENIN, T., ZHAO, L., SASAN, A., RAFATIRAD, S., AND HOMAYOUN, H. Hybrid-shield: Accurate and efficient cross-layer countermeasure for run-time detection and mitigation of cache-based side-channel attacks. IEEE/ACM International Conference on Computer-Aided Design.

[30] WANG, H., SAYADI, H., MOHSENIN, T., ZHAO, L., SASAN, A., RAFATIRAD, S., AND HOMAYOUN, H. Mitigating cache-based side-channel attacks through randomization: A comprehensive system and architecture level analysis. In *DATE* (2020), IEEE.

[31] WANG, H., SAYADI, H., RAFATIRAD, S., SASAN, A., AND HOMAYOUN, H. Scarf: Detecting side-channel attacks at real-time using low-level hardware features. In *IOLTS* (2020), IEEE.

[32] WANG, H., SAYADI, H., SASAN, A., RAFATIRAD, S., MOHSENIN, T., AND HOMAYOUN, H. Comprehensive evaluation of machine learning countermeasures for detecting microarchitectural side-channel attacks. In *Proceedings of the 2020 on Great Lakes Symposium on VLSI*, p. 181–186.

[33] WANG, Z., AND LEE, R. B. A novel cache architecture with enhanced performance and security. In *Proceedings of the 41st annual IEEE/ACM International Symposium on Microarchitecture*.

[34] WANG, Z., AND LEE, R. B. New cache designs for thwarting software cache-based side channel attacks. In *ACM SIGARCH Computer Architecture News* (2007).

[35] YAO, F., FANG, H., DOROSLOVACKI, M., AND VENKATARAMANI, G. Towards a better indicator for cache timing channels. *arXiv preprint arXiv:1902.04711* (2019).

[36] YAROM, Y. Mastik: A micro-architectural side-channel toolkit. *Retrieved from School of Computer Science Adelaide: http://cs. adelaide. edu. au/˜ yval/Mastik* (2016).

[37] YAROM, Y., AND FALKNER, K. Flush+ reload: A high resolution, low noise, l3 cache side-channel attack. In *USENIX Security Symposium* (2014), vol. 1, pp. 22–25.

[38] YE, L., AND KEOGH, E. Time series shapelets: a new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (2009), ACM, pp. 947–956.

[39] ZHANG, T., ZHANG, Y., AND LEE, R. B. Cloudradar: A real-time side-channel attack detection system in clouds. In *International Symposium on Research in Attacks, Intrusions, and Defenses* (2016), Springer, pp. 118–140.