PerSpectron: Detecting Invariant Footprints of Microarchitectural Attacks with Perceptron

Samira Mirbagher-Ajorpaz[1,2], Gilles Pokam[3],
Esmaeil Mohammmadian-Koruyeh[4], Elba Garza[1]
Nael Abu-Ghazaleh[4], Daniel A. Jiménez[1]

Texas A&M University[1], UC San Diego[2], Intel Labs[3] and UC Riverside [4]

53rd IEEE/ACM International Symposium on Microarchitecture

Outline

1. Introduction
2. Background
3. Motivation
4. Feature Selection
5. Example
6. Hardware Design
7. Results
8. Conclusion

Spectre and Meltdown attacks on the Headlines in 2018

Meltdown and Spectre: 'worst ever' CPU bugs affect virtually all computers

Everything from smartphones and PCs to cloud computing affected by major security flaw found in Intel and other processors – and fix could slow devices

Quote from:
https://www.theguardian.com/technology/2018/jan/04/meltdown-spectre-worst-cpu-bugs-ever-found-affect-computers-intel-processors-security-flaw

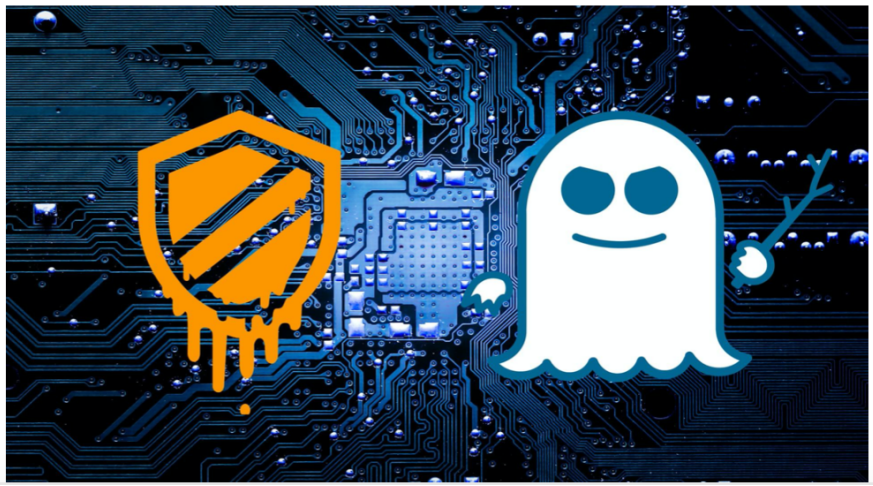
Jan 2018

Intel admits a load of its CPUs have Spectre v2 flaw that can't be fixed

And won't fix Meltdown *nor* Spectre for 10 product families covering 230-plus CPUs

Quote from:
https://www.theregister.com/2018/04/04/intel_spectre_microcode_updates/

Apr 2018

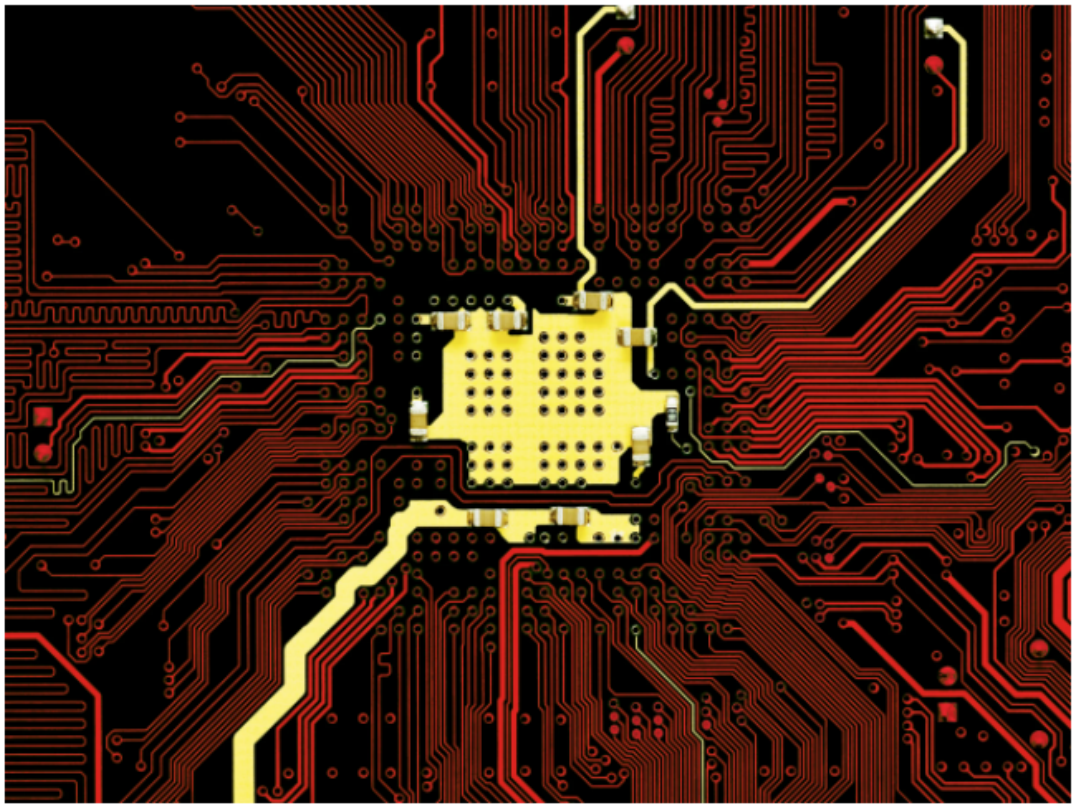


One year after, Headlines in 2019

LILY HAY NEWMAN SECURITY 01.03.2019 12:33 PM

The Elite Intel Team Still Fighting Meltdown and Spectre

One year after a pair of devastating processor vulnerabilities were first disclosed, Intel's still dealing with the fallout.



CASEY CHIN; GETTY IMAGES

New variants continue to appear at a regular cadence.

ACM SIGARCH

JOIN BENEFIT CONTRIBUTE DISCOVER BLOG

Computer Architecture Today

Informing the broad computing community about current activities, advances and future directions in computer architecture.

Let's Keep it to Ourselves: Don't Disclose Vulnerabilities

by Gus Uht on Jan 31, 2019 | Tags: Opinion, Security



CONTRIBUTE

Editor: Alvin R. Lebeck

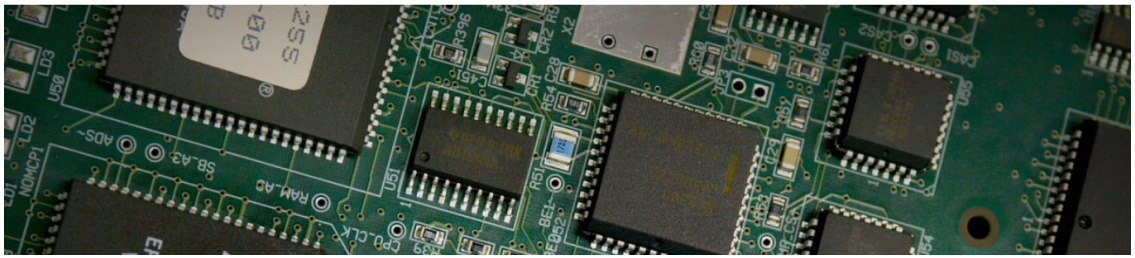
Associate Editor: Vijay Janapa Reddi

Contribute to Computer Architecture Today

New variants on the headlines in 2019

New secret-spilling flaw affects almost every Intel chip since 2011

Zack Whittaker @zackwhittaker / 12:00 pm CDT • May 14, 2019

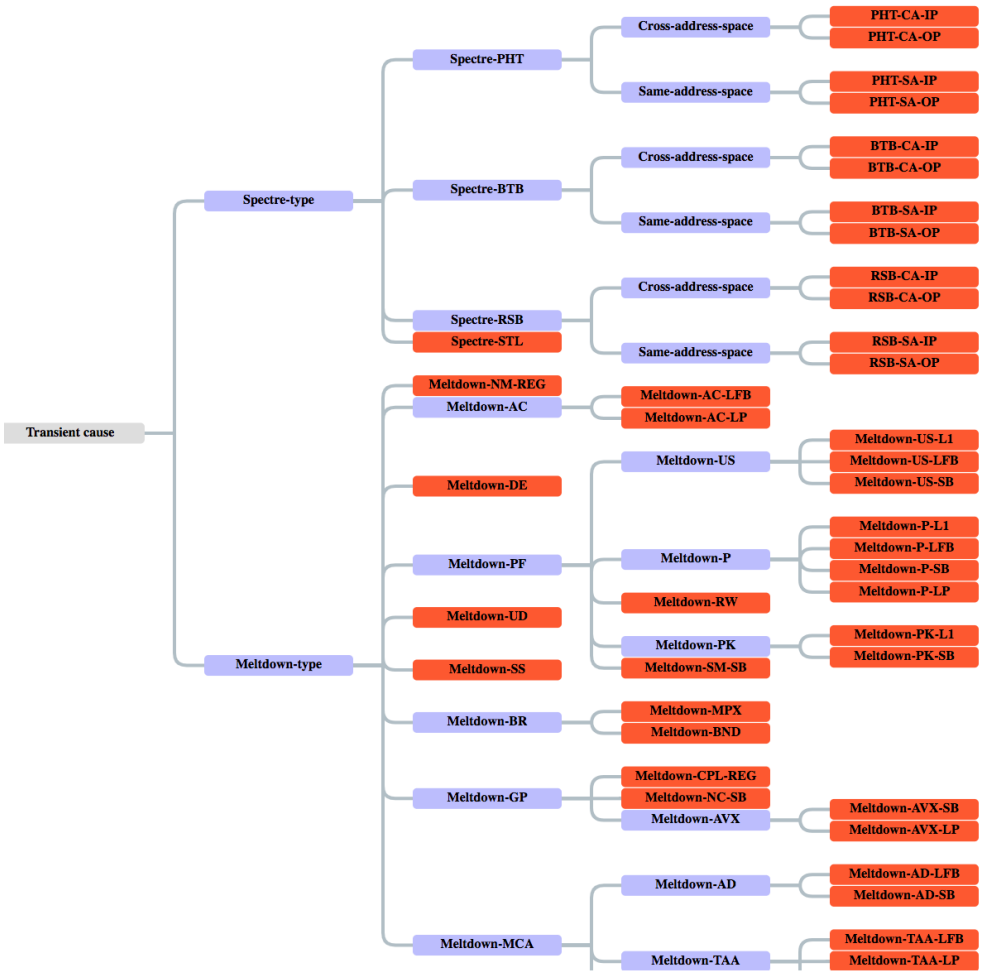


Computing

Intel CPUs attacked again as ZombieLoad V2 exploit rises f Spectre's grave

By Jon Martindale
November 14, 2019

Variations of Spectre and Meltdown attacks using different components of pipeline:



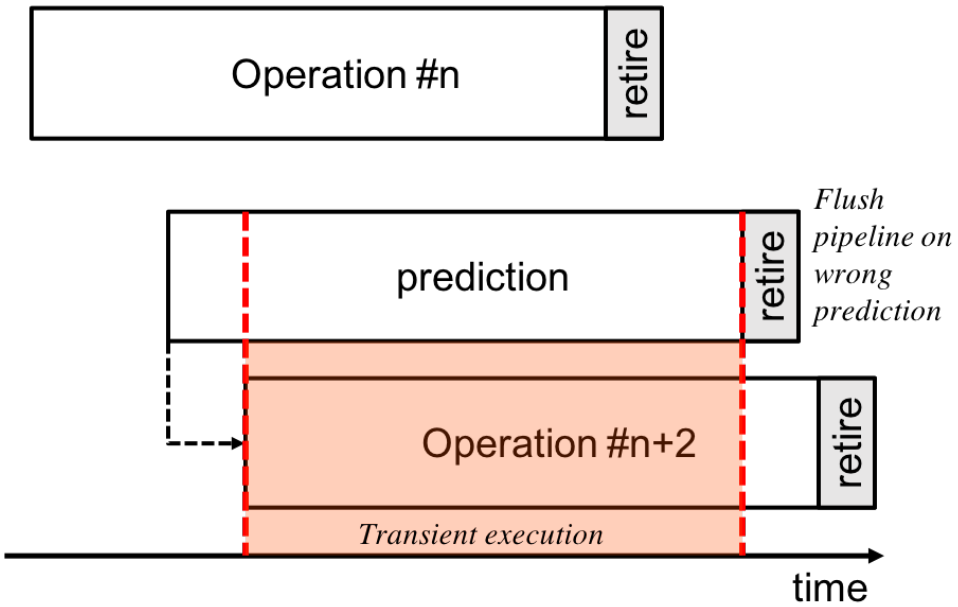
Source: A Systematic Evaluation of Transient Execution Attacks and

Which foundation of optimizations leaks information

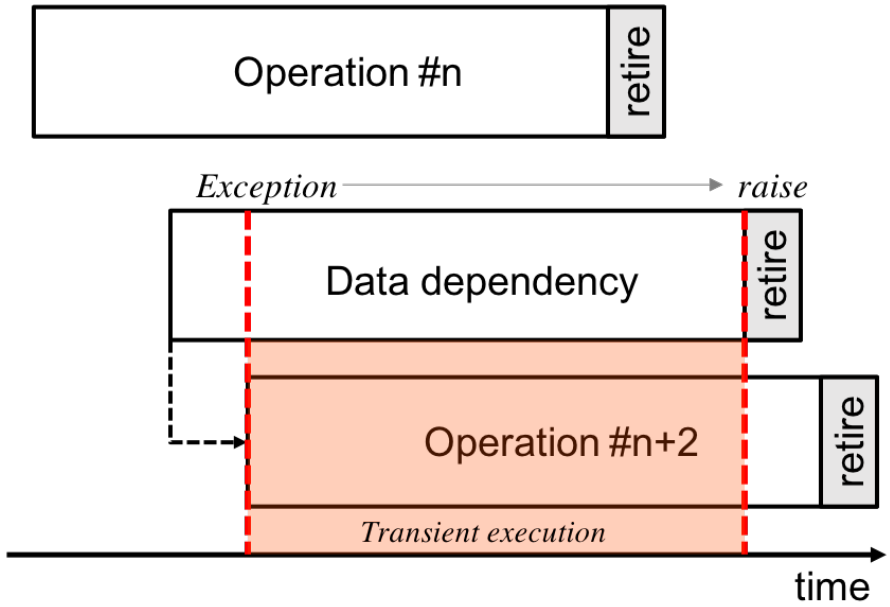
- For many years we had a free ride from Dennard scaling
  - Scale key transistor features down
  - Speed, density, energy efficiency improved
  - Driving force behind Moore's Law
- End of Moore's Law
- No more easy gains from low-level physics
  - Increase clock rates mostly maxed out (3.8 GHz P4 in 2004)
  - Improve switching speed come at the cost of increased power
  - Improve memory speeds DRAM latency huge, not improving much
- Industry focus on pipelining and parallelism:
  - Reducing memory delays $\rightarrow$ caches
  - Working during delays $\rightarrow$ speculative execution

Mircroarchitectural incontinence in modern processors

- Timing difference
  - e.g cache hit and cache miss are perfect distinguishers.
- Speculative and out-of-order execution leave microarchitectural traces
  - We can see them for example through the cache



Spectre-type attack



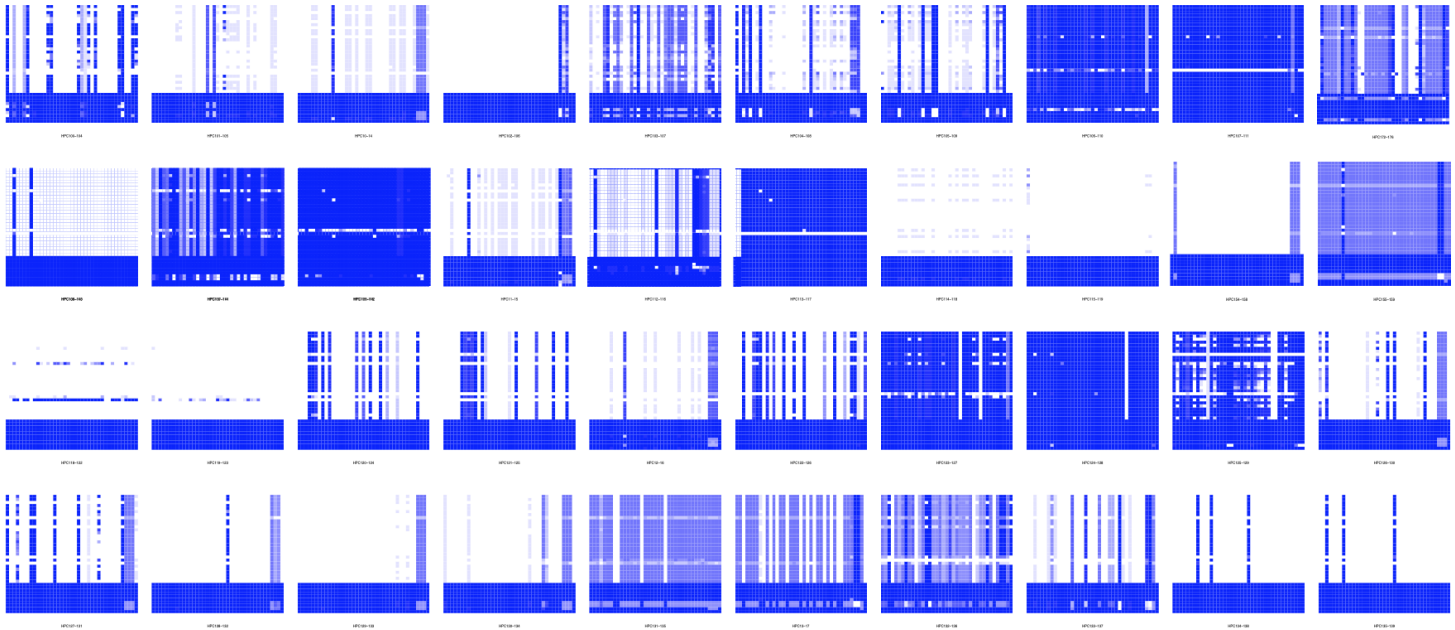
Meltdown-type attack

Detection in software

- The number of events available to performance counters is limited
  - E.g. recent Intel processors, can only monitor up to 4 at a time.
  - Requires counter multiplexing to monitor reach feature space
- Counter multiplexing
  - Degrades accuracy,
  - Provides opportunities for evasion,
  - Missing opportunities to detect an attack.
- An attacker may be able to access performance counters in software and modulate its attack to evade detection.
- The sampling rate of performance counters is low
  - Allows an attacker to adjust the bandwidth of an attack to fit within the sampling interval of a software based detector.

Software detectors can be evaded easily

No combination of four performance counters was found that can detect new variations.



White pixels in diagonal and white horizontal or vertical lines indicate lack of generalization.

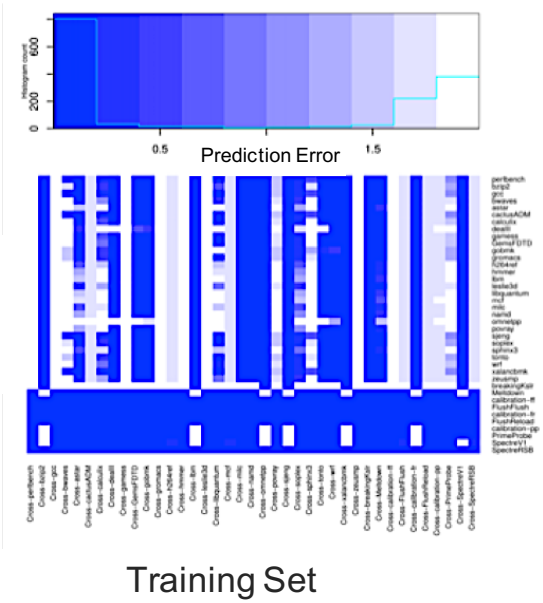
Moving the detection to hardware

- Allows the predictor to efficiently use a large set of features
  - Not limited to the commit state, including speculative instructions.
  - Increase the accuracy
- Security guarantees are stronger
  - The detector cannot be disabled by software even if the kernel is compromised,
- Operates outside of the processors critical path, offering security without added computation delay.
- High sampling frequency is possible in hardware
  - Makes it impossible for an attacker to time its attack to the sampling interval

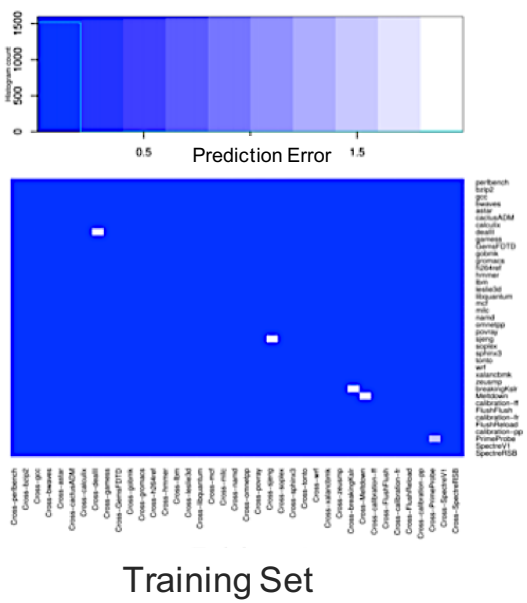
We will discuss: Detecting different variations of attacks is similar to identifying a rotated or moved target in Object Recognition. Monitoring the entire input space (having *Replicated Detectors* in all components of pipeline), is imperative and is only possible in hardware.

Relying on highly correlated features is not sufficient;
using all the features cause *overfitting*;
we need a solution to the *domain hopping* problem.

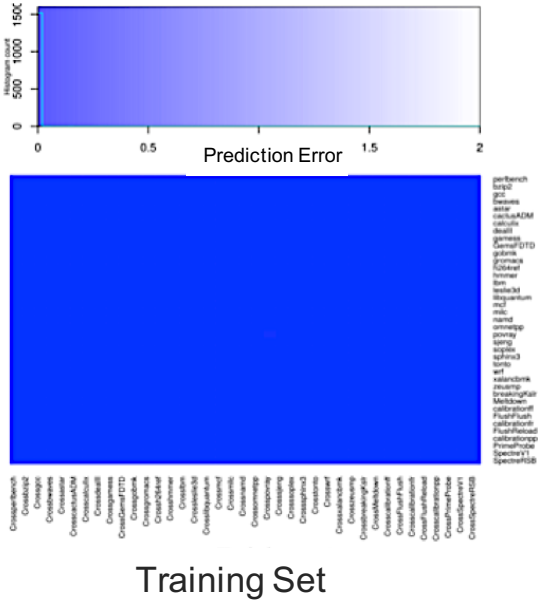
Previous features



Detection in hardware

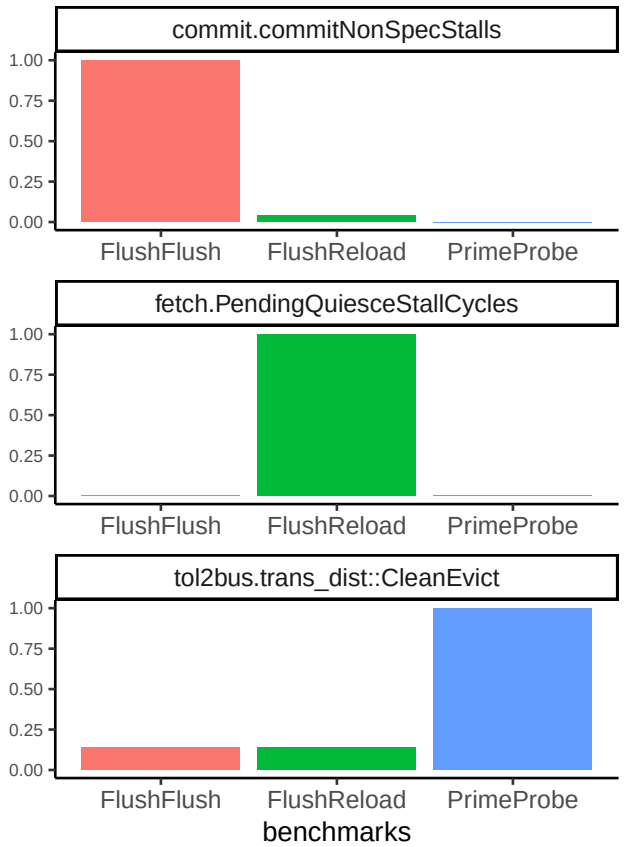


PerSpectron



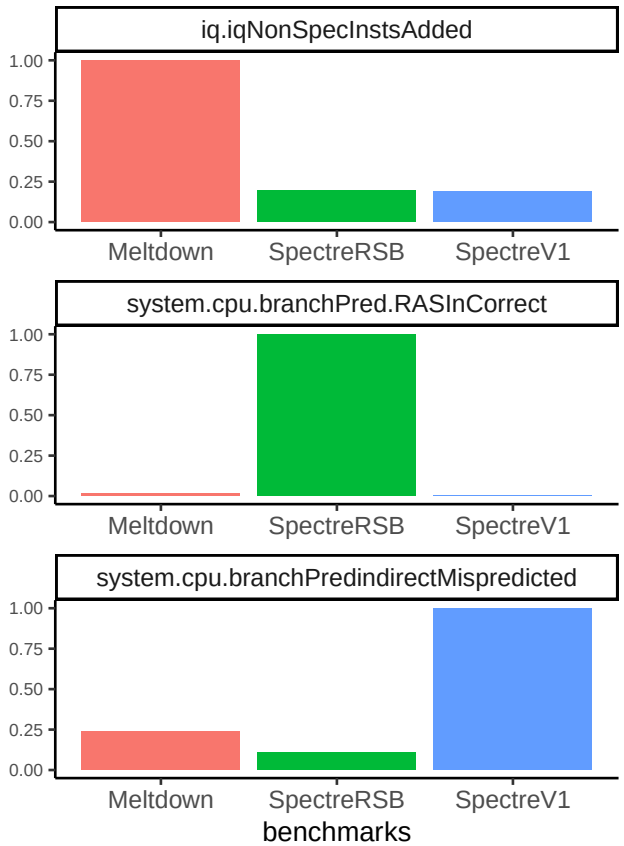
The main challenge is **generalization**: Recognizing an attack of new variation and under polymorphic or low bandwidth transformation.

Different variants of attacks present a domain hopping problem similar to the *viewpoint* problem of image recognition.



- While Flush+Flush does not cause high *PendingQuiesceStallCycles*, it causes abnormal *commitNonSpecStalls*.
- Also, while Flush+Reload does not cause abnormal *commitNonSpecStalls*, it causes an abnormal number of *PendingQuiesceStallCycles*.
- While Prime+Probe does not cause abnormally low *tol2bus.transdist.CleanEvict* or *PendingQuiesceStallCycles* values, it causes abnormal *tol2bus.transdist.CleanEvict*.

Different variants of attacks present a domain hopping problem similar to the *viewpoint* problem of image recognition



- While SpectreRSB does not have a high *branchPredindirectMispredicted* value, it has a high *RAS In-corrects* value to the IQ.
- Meltdown cannot be predicted with a high SquashedLoads value, but it causes high *NonSpecInstrsAdded* to the IQ.

ch

- *Reads from Write Buffer* significantly increases all the variations! This in an invariant feature indicative of either of those attacks.

This observation was also confirmed later when CacheOut attack was disclosed which directly targets write buffer.

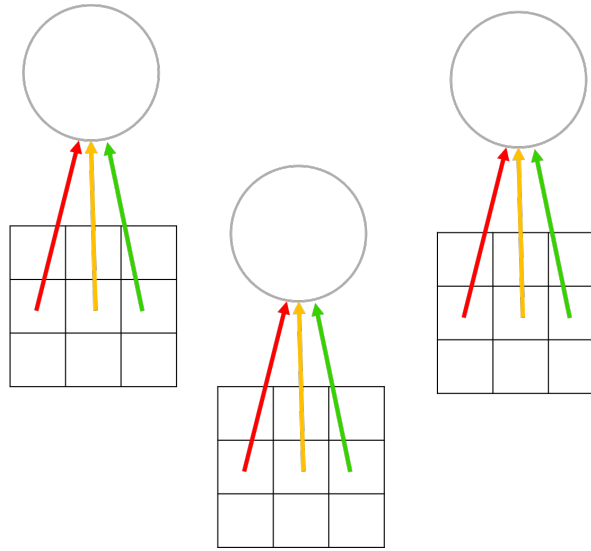
PerSpectron: General approach

- If a feature vector was useful in detecting one target, it is likely that a similar feature detector with different positions in the input space can detect the replicated information.
  - Replicated features approach
- Using the entire space of possible parameters in the training set allowed the use of a simpler architecture
  - No need to represent the relationships between features directly because other features capture them.

No hidden layer was necessary—the mapping from the features' instantiation parameters to the attacks class became linear.

Building the replicated perceptron detectors

- Create a correlation matrix.
- Group features $f_1 \ldots f_m$ that are closely correlated into group $g_1 \ldots g_c$, $c = 17$ is number of pipeline components. (unsupervised procedure)
- Decorrelate the features within a component, still keep one from every group.
- Sort features in each component by the mutual information about the class *benign* or *suspicious*.(supervised procedure)
- Select a bank of n features from each component of pipeline.



This process seeks to maximize the information about the class contained in each component and minimize the correlation within a component and ensures replicated detectors.

Invariant feature example 1

- CommittedMaps in Rename
  - Polymorphic evasions that uses contention on the Register File were detected preleakage after including CommittedMaps features even by the simplest models i.e (DT-CART) which shows this feature is highly informative.
- The Committed Map table holds the speculative mappings from ISA registers to physical registers.
- If enabled, this allows a single-cycle reset of the pipeline during flushes and exceptions of any kind.
  - E.g on every branch, the Rename Map tables are used to allow single-cycle recovery on a branch misprediction.

This functionality explains why CommittedMaps is invariant to different versions of transient execution attacks.

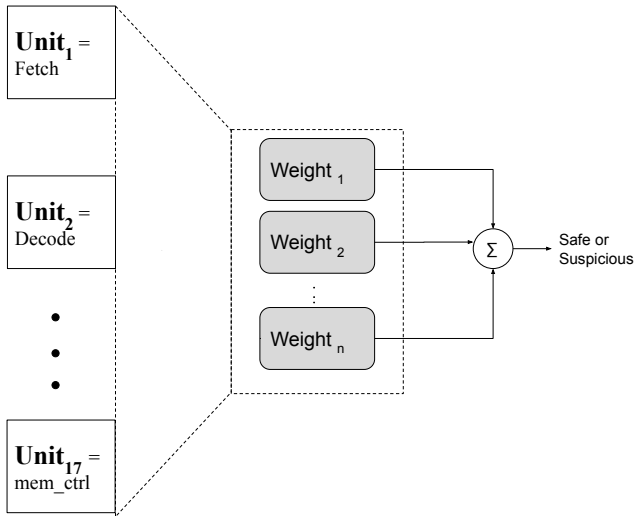
Invariant feature example 2

- fetch.MiscStallCycle
  - It is the effect of misses and stalls in the Fetch stage.
  - The squashed cycles in each stage, all the ROB, IQ, and Register full events, undone maps in the Rename stage, and memory order violation in the IEW stage, propagate back to the Fetch stage.
- The relationship between these events' is not a simple cumulative function in an out-of-order processor.
- But by including overlapping and redundant features, one feature tells us how other features are related.

This solves the domain-hopping problem caused by different variants of microarchitectural attacks, eliminates the need for hidden layers
and
makes our detector more difficult to evade!

Hardware design

- PerSpectron feature learning algorithm streamlines 106 features from 1159 microarchitectural features
- The features correlate with a range of microarchitectural attack
  - SpectreV1, SpectreV2, SpectreRSB, Meltdown, breakingKSLR, Flush+Flush, CacheOut, Flush+Reload, Prime+Probe, and Cache attack Calibration.
- The selected features are used as inputs to a perceptron-based hardware predictor for detection and classification of attacks.



Hardware-friendly representation and computation

- Regardless of the order and the position of each feature, the k-sparse representation are distinctive, and only contain 0/1 values
- Computing the perceptron output
  - Multiplication is not necessary to compute the dot product.
  - Simply add the weight when the input bit is 1 and subtract when bit is -1.
  - Only the sign bit of the result is needed to make a prediction.
- Prediction happens in hardware and in parallel to the execution.
- No additional overhead for hardware counters
- Simpler than perceptron-based branch predictors used in mobile CPUs (see e.g. the Samsung Exynos paper in ISCA2020).

$$\textit{suspicious} :< 0,1,0,0 >$$

$$\textit{suspicious} :< 1,0,1,0 >$$

$$\textit{suspicious} :< 0,0,0,1 >$$

$$\textit{safe} :< 1,1,0,0 >$$

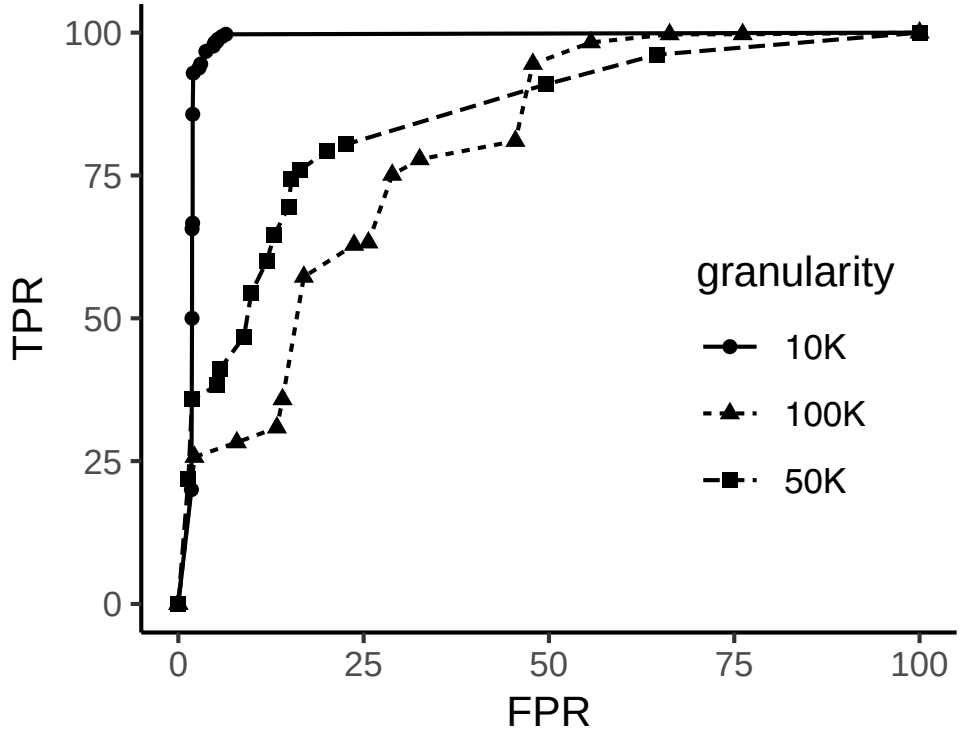
Predictive performance on future (unseen) attacks

- Standard K-fold cross-validation (CV)
  - The unbiased estimate of prediction error.
  - The expected loss on future instances.
- Splitting our dataset into three parts:
  - A training set for model fitting, a validation set for model selection,
  - A test set for the final evaluation of the selected model.
- At every fold, we remove all samples belonging to attacks and benign programs in the test set from the training and validation data.
- We use different cache channels for the attacks in the test set and the training set of each CV fold.

k'th test fold	D_k	D_{-k}
1	spectreRSB, spectreV2, cacheOut, breakingKSLR, prime+probe	meltdown, spectreV1, flush+flush, flush+reload
2	spectreV1, spectreV2, cacheOut, flush+reload	meltdown, spectreRSB, breakingKSLR, flush+flush, prime+probe
3	spectreV2, cacheOut, meltdown, breakingKSLR, flush+flush	spectreV1, spectreRSB, flush+reload, prime+probe

Sensitivity and specificity

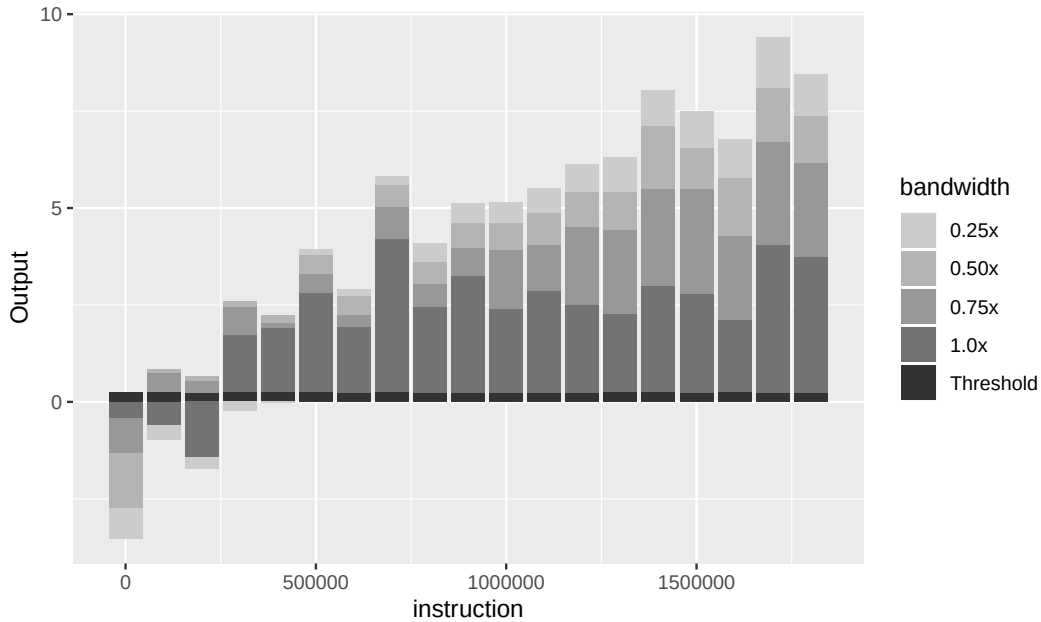
- We run our detector for a set of thresholds and sampling intervals
- 10K interval is better than the 50K and 100K.
- Threshold of 0.25 is the best with an area under the curve.



- PerSpectron has high sensitivity and specificity (AUC of 0.9949)

Resiliency to Bandwidth Reduction Mimicry

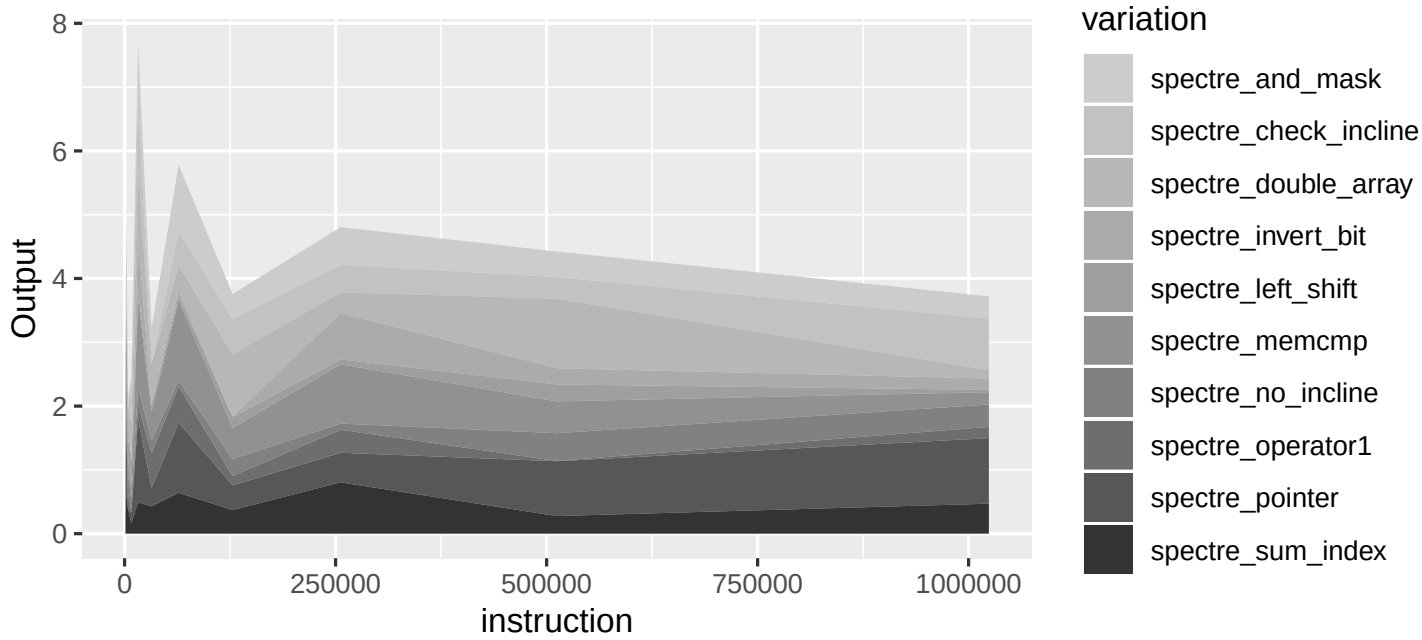
- Based on timing the completion of all attack atomic tasks to fit within the sampling interval.
- It is a problem with the low sampling frequency in software detectors.



- All lower bandwidth versions stay above the cutoff threshold
- Our sampling interval is 3 μs, which gives 20 sampling intervals within the 61 μs run time it takes to complete all atomic tasks

Resiliency to Polymorphic Evasion

- Attacker attempts to produce different binaries implementing the attack.
- This is a typical strategy used by malware to evade signature based detectors



- All variations were detected and triggered the perceptron output as suspicious at the same sampling interval.

Conclusion

- A robust detection system for broad variations of microarchitectural attacks
- Operates in hardware and outside the processor's critical paths
  - offers security without added computation delay.
- Detection before leakage
- Resilient to known bandwidth and polymorphic evasion methods
- PerSpectron mapped a non-linearly separable problem to linear:
  - Separable by perceptron and is implementable in hardware.
  - Competitive with a more complex deepNN that is not easily implementable in hardware.
- PerSpectron neural weights can be updated
  - Using a vendor distributed patch reflecting training with the most recent known classes of attacks if needed.
- Ready to be deployed:
  - Uses presently available hardware performance counters
  - Uses Perceptron circuit which is already built, verified and is successfully used in modern processors.
- In the future: security of PerSpectron against adversarial ML attacks.