# Side Channels through Lower-level Caches

PIs: Dmitry Ponomarev (SUNY Binghamton) and Nael Abu-Ghazaleh (University of California Riverside. Project URL: http://www.cs.binghamton.edu/~secarch

## PROJECT OBJECTIVES

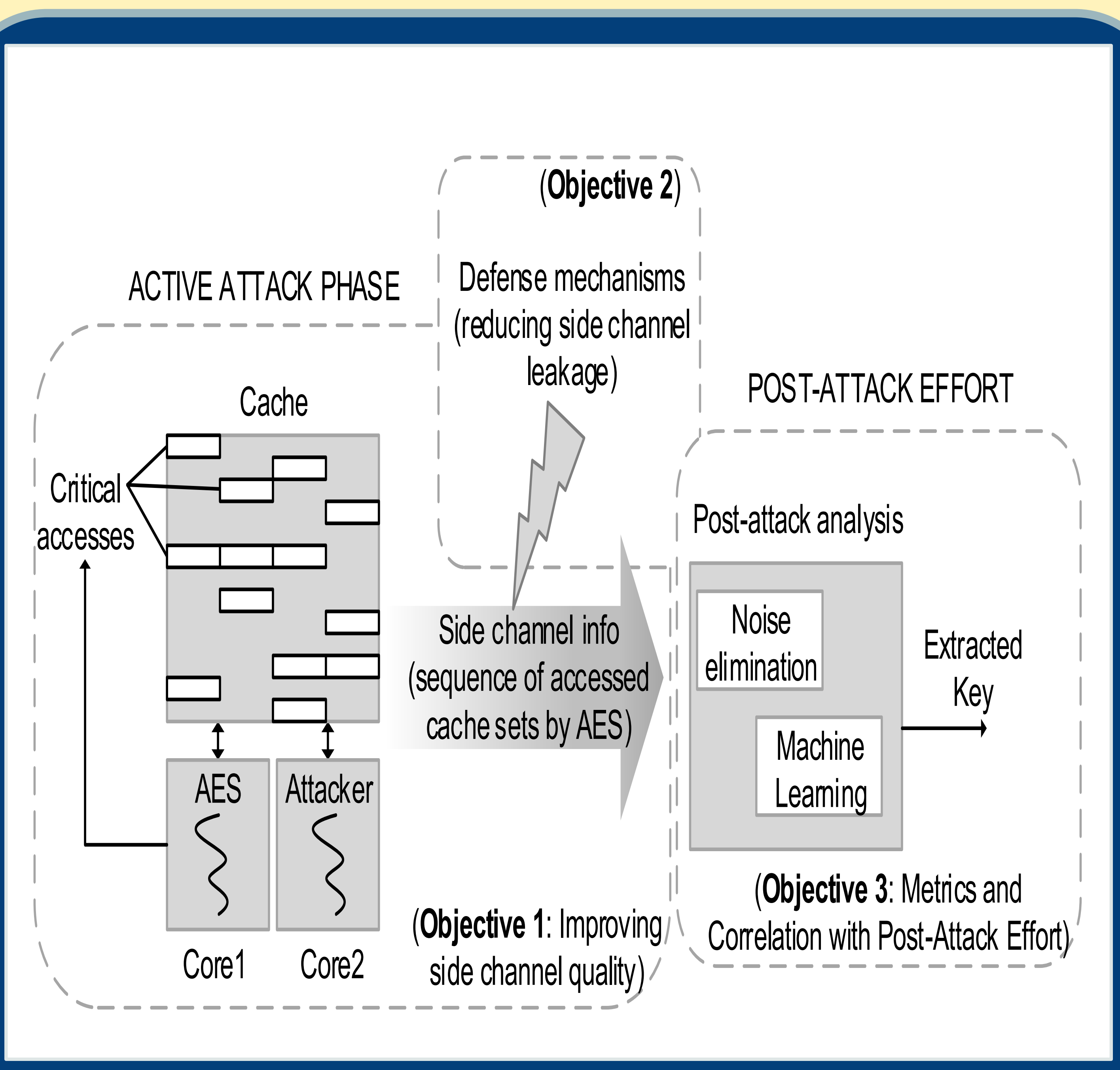
The objectives of these project include development of high-resolution side-channel attacks on Last-Level Cache (LLC), investigation of efficient defense mechanisms, development of security metrics and consideration of timing channels through other processor resources.

### LLC Attack Summary

New PRIME+PROBE attack that works with regular pages and does not rely on the presence of large pages. Reverse-engineered cache operation on real Sandy Bridge processor (Kayaalp et al, DAC'16, **best paper nominee)**

### LLC Defense Summary

Relaxed Inclusion Caches (RIC): Relax inclusion property when it is safe to do so, specifically for read-only data and thread-private data. Result: critical data stays in local caches and is not seen through LLC. Paper is currently **under review**.


(Objective 2)
ACTIVE ATTACK PHASE
Defense mechanisms (reducing side channel leakage)
POST-ATTACK EFFORT
Cache
Critical accesses
Post-attack analysis
Noise elimination
Extracted Key
Side channel info (sequence of accessed cache sets by AES)
Machine Learning
AES
Attacker
Core1
Core2
(Objective 1: Improving side channel quality)
(Objective 3: Metrics and Correlation with Post-Attack Effort)

## Approach and Evaluation

### LLC attacks and defenses

- Use real processors to implement attacks, reverse-engineer cache operation and measure the leakage through cache side channel in realistic scenario
- Use cycle-accurate processor simulator to model the impact of hardware-based defenses.

### Other CPU Timing Channels

- Analyze vulnerability of shared CPU hardware resources to timing attacks
- Implement either side-channel attacks or end-to-end covert channels using a variety of resources on real CPUs
- Model hardware defenses using simulators and implement software defenses on real machines

### Other Activities: Side-Channels through Branch Predictors

Shared branch predictor hardware leaks branch-specific information across software layers:

- Directional predictor can leak the direction of a particular branch (taken or not)
  - Can compromise system if outcome of branch instruction depends on secret information, e.g. bits of secret key (**ongoing work**)
- BTB can leak the address of a branch instruction
  - Can be used to bypass Address Space Layout Randomization (ASLR) schemes, including kernel and hypervisor ASLR (**Evtyushkin et al., MICRO'16**)

### Other activities: Detecting and Mitigating Covert Channels

Shared hardware in modern CPUs can be used to build fast and reliable covert channels:

- Hardware Random Number Generator (RNG)
  - Recent Intel CPUs provide RDSEED instruction to enable direct access to RNG with explicit notification of random data availability
  - Slow random bit regeneration rate allows to create fast and practically error free covert channel (**Evtyushkin and Ponomarev, CCS'16**)
- Branch predictor unit
  - Novel Residual State Channel (RSC) encodes data by placing branch predictor entries into of of the "strong" states (**HASP'15, TACO'16**)
- Covert channel through GPGPU (**CAL 2016**)

Interested in meeting the PIs? Attach post-it note below!

NSF Secure and Trustworthy Cyberspace Principal Investigator Meeting
Jan. 9 -11th 2017

National Science Foundation
WHERE DISCOVERIES BEGIN

BINGHAMTON UNIVERSITY STATE UNIVERSITY OF NEW YORK

UNIVERSITY OF CALIFORNIA UC RIVERSIDE