Side-Channel Attacks on Optane Persistent Memory

Sihang Liu, *University of Virginia;* Suraaj Kanniwadi, *Cornell University;*
Martin Schwarzl, Andreas Kogler, and Daniel Gruss, *Graz University of Technology;*
Samira Khan, *University of Virginia*

https://www.usenix.org/conference/usenixsecurity23/presentation/liu-sihang

This artifact appendix is included in the Artifact Appendices to the Proceedings of the 32nd USENIX Security Symposium and appends to the paper of the same name that appears in the Proceedings of the 32nd USENIX Security Symposium.

August 9–11, 2023 • Anaheim, CA, USA

978-1-939133-37-3

USENIX'23 Artifact Appendix: Side-Channel Attacks on Optane Persistent Memory

Sihang Liu
University of Virginia

Suraaj Kanniwadi*
Cornell University

Martin Schwarzl
Graz University of Technology

Andreas Kogler
Graz University of Technology

Daniel Gruss
Graz University of Technology

Samira Khan
University of Virginia

A Artifact Appendix

A.1 Abstract

This paper presents reverse-engineering on Intel Optane persistent memory and demonstrates four attacks. This artifact has included all the source code and scripts for reverse-engineering Optane's internal caches and the following attacks: (1) a local covert channel based on Optane's internal caches, (2) a keystroke side-channel attack on a remote user via an Optane-backed key-value store, `pmemkv`, (3) a fully remote covert channel where the sender stealthily sends a message through textttpmemkv. and (4) Note Board attack where the sender leaves a message on Optane and the receiver can retrieve the message after a long time.

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

Our demonstrated attacks are all based on unmodified systems. The evaluation system does not disable or modify any security-related features in the system. Therefore, there is no risk for evaluators.

A.2.2 How to access

The source code of our artifact is available at https://github.com/Systems-ShiftLab/optane_sec23_ae/

A.2.3 Hardware dependencies

The experiments require a hardware platform as listed below:

- CPU: Intel Cascade Lake.
- Persistent Memory: 1st generation Optane DCPMM. If the server contains multiple Optane modules, they must be running under non-interleaved mode.
- Network: one-hop Ethernet connection between the two Optane servers (for remote covert channels and side-channel attacks).

A.2.4 Software dependencies

The experiments require a software environment as listed below:

- Operating system: Ubuntu 18.04, kernel v5.4.
- Compiler: gcc and g++-7.5.
- Libraries and tools: PMDK v1.9, ndctl v68, ipmctl v02.00.00.3852, and websocket-client.
- Optane mode: The Optane memory must be running in *App Direct* mode using ipmctl.
- File system for Optane: The Optane device must be mounted in DAX mode.

A.2.5 Benchmarks

This artifact includes the following workloads:

- pmemkv server: an Optane-optimized key-value store server program. The covert channels and side-channel attacks are performed via this program.
- weServer: a WebSocket server library. This library enables the typer to send keystroke inputs to the pmemkv-based storage server.
- Keystroke inputs: a dataset from http://personal.ie.cuhk.edu.hk/~ccloy/files/datasets/keystrokes.zip. This input dataset is used to generate keystorkes with realistic inter-keystroke timings.

A.3 Set-up

A.3.1 Installation

To compile our microbenchmarks and run our experiments, we require the following prerequisite software:

*Suraaj Kanniwadi contributed to this work during his internship at the University of Virginia.

- ndctl/daxctl v68
- pmdk 1.9.2
- pmemkv 1.3
- libuv 1.18
- websocket-client (pip package)

A.3.2 Basic Test

A small Reverse Engineering microbenchmark can be used as a self test. One can run `reverse/wear-level/script-ae.sh` as an example. If this fails, this can be due to two reasons:

- **Build failure**: The required libraries are missing.
- **Runtime failure**: It is not the case that the system has persistent memory mounted in dax mode with read/write permissions.

A.4 Evaluation workflow

A.4.1 Major Claims

(C1): Our *Reverse Engineering Framework* runs in user-space and helps unveil interesting internal components of Optane, such as:

- Size, associativity and replacement policies of Optane internal buffers.
- Wear-levelling behaviour exhibited by Optane when performing repeated writes to the same location
- Effect of concurrent writes on read performance

Experiments involved: E1, E2, E3, E4, E5

(C2): *Local Covert Channels* using 3 techniques (RMW, AIT and Read/Write contention) to transmit data covertly.
Experiments involved: E6

(C3): A *Keystroke Attack* which enables us to detect inter-keystroke interval between typists.
Experiments involved: E7

(C4): A *Remote Covert Channel* which allows us to achieve a bandwidth of 10 bps over the network.
Experiments involved: E8

(C5): The **Noteboard Attack**, which is an asynchronous, persistent covert channel.
Experiments involved: E9

A.4.2 Experiments

For ease of artefact evaluation, we provide a single script `runall-ae.sh` which builds our microbenchmarks, runs all experiments, collects data, and finally generates a concise report (`report/report.pdf`) with all the results. `runall-all.sh` is simply a wrapper script which does the following:

- **Run all experiments**: Each experiment directory has a `script-ae.sh` script. `runall-ae.sh` finds all these scripts and runs them all.
- **Copy all results**: Each experiment directory stores results in a `results-ae/` directory. The script copies them to a common `report/` directory.
- **Compiles the report**: The `report/` directory contains latex files which plot all the results.

Information about each experiment is listed below:

(E1): *[Reverse Engineering Heirarchy] [25 compute-minutes + 64GB pmem disk]*
How to: Run `reverse/user_lens/script-ae.sh`
Description: Runs the experiment with depicts the overall structure of Optane.

(E2): *[Reverse Engineering Bitmask Pointer Chasing] [3.5 compute-hours + 64 GB pmem disk]*
How to: Run `reverse/bit_pc/script-ae.sh`
Description: Runs experiments which depict Optane's internal buffer associativity.

(E3): *[Reverse Engineering Replacement Policy] [10 compute-minutes + 1 GB pmem disk]*
How to: Run `reverse/replacement/script-ae.sh`
Description: Runs experiments which depict Optane buffers' replacement policy.

(E4): *[Reverse Engineering Wearlevelling Policy] [15 compute-seconds + 1 GB pmem disk]*
How to: Run `reverse/wear-level/script-ae.sh`
Description: Runs experiments which depict Optane's wear-levelling behaviour.

(E5): *[Reverse Engineering Read-Write Contention] [1 compute-minute + 1 GB pmem disk]*
How to: Run `reverse/read_write_cont/script-ae.sh`
Description: Runs experiments which depict read-write contention in Optane.

(E6): *[Local Covert Channel] [45 compute-minutes + 2 GB pmem disk]*
How to: Run `local_covert/script-ae.sh`
Description: Runs 2 processes (a sender and receiver) on the same machine, and facilitates covert communication, and measures the achievable bandwidth and accuracy.

(E7): *[Keystroke Side Channel] [1 compute-hour + 1 GB pmem disk]*
How to: Run `keystroke/script-ae.sh`
Description: Runs the keystroke attack as follows:

- Run a kv_server + prober on server A.
- Then, a client connects to server A from server B and sends keystrokes.
- These keystrokes can be detected via the prober on server A.

(E8): *[Remote Covert Channel] [35 compute-seconds + 2 GB pmem disk]*

How to: Run `remote_covert/script-ae.sh`
Description: Runs 2 processes (a sender and receiver) on machines connected across the network, and measures achievable accuracy and bandwidth.

(E9): *[Noteboard Attack] [1 compute-hour + 1 GB pmem disk]*
How to: Run `noteboard/script-ae.sh`
Description: Runs the noteboard attack as follows:

- Run a kv_server server A.
- Then, a client connects to server A from server B and encodes a message in wear-levelling metadata.
- Then another client connects to server A and retrieves this encoded message.

A.5 Version

Based on the LaTeX template for Artifact Evaluation V20220926. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at https://secartifacts.github.io/usenixsec2023/.