

Learner's Space

Hardware Security

Mission_3

Report: Occupancy-based Covert Channel Attack

-Dikshit Singla
-23b3974

Mission Objective

Design and implement a **covert communication channel** between two processes (sender and receiver) without shared memory, using **Last-Level Cache (LLC) contention**. The sender encodes a secret message (from `msg.txt`) as CPU cache activity; the receiver decodes it by measuring memory access latency.

Core Idea: Occupancy-based Attack

This mission simulates a **side-channel communication technique**. The sender and receiver communicate by **intentionally generating or avoiding cache activity**:

Bit Sent	Sender Behavior	Receiver Observation
1	LLC cache thrashing	High latency (less cache available)
0	Idle (minimal cache use)	Low latency (more cache available)

The receiver records loop iteration counts during fixed-length time windows. Lower iteration counts suggest cache contention, indicating a 1.

Implementation Details

Sender (**sender.c**)

- For bit **1**: Thrashes the LLC using a **buffer** of size 8 MB.
- For bit **0**: Performs lightweight **nop** operations.
- Uses a fixed **BIT_DURATION = 250 million CPU cycles** per bit.
- Repeats each bit **three times** to enable error correction on the receiver side.
- Sends a **preamble** of 3 bytes of **0xFF (11111111)** also repeated 3 times each to indicate start of message.

Receiver (**receiver.c**)

- Uses a buffer of 8 MB to probe LLC load by counting loop iterations during each **BIT_DURATION**.
- First runs a **calibration phase** to measure average counter when cache is idle → determines a threshold.
- Records **NUM_BITS = 6144** bits (3x for 2048 bits = 256 bytes).
- Detects preamble (72 consecutive 1s) to locate message start.
- Uses **majority voting** on every 3-bit group to decode each real bit (forward error correction).
- Converts clean bits to bytes and compares with **msg.txt** using **check_accuracy()**.



Optimization Techniques Applied

Technique	Purpose
Increased <code>BIT_DURATION</code>	Reduced overlap and timing drift
LLC buffer stride = 64B	Matches cache line size for better eviction
Bit repetition (3x)	Added resilience to timing noise
Physical core pinning (<code>taskset</code>)	Avoided interference from SMT (hyperthreading)
Soft preamble match	Allowed a few bit errors during sync
Threshold = $0.75 \times \text{calibration}$	Balanced detection of 1s vs. 0s



Results

Parameter	Value
Bits transmitted	2048

Total bits sent 6144 (w/ 3x repeat)

Accuracy Achieved ~47% (after error correction)

Accuracy improved from ~17% to 47% through carefully layered fixes.

```
Bit 1012: Counter = 18232928 => 0
Bit 1013: Counter = 17301504 => 0
Bit 1014: Counter = 19398656 => 0
Bit 1015: Counter = 17694720 => 0
Bit 1016: Counter = 18612224 => 0
Bit 1017: Counter = 18612224 => 0
Bit 1018: Counter = 18087936 => 0
Bit 1019: Counter = 17694720 => 0
Bit 1020: Counter = 18219008 => 0
Bit 1021: Counter = 18743296 => 0
Bit 1022: Counter = 18481152 => 0
Bit 1023: Counter = 18219008 => 0

Received Message:
CKk;{+cc7^bw7'V?6BwFw2FR6GRGFVvV鎧W"17?vv,
nm nL.
Accuracy: 16.70%
^C
[1]+ Done taskset -c 0 ./receiver
madhuri@ENG7:~/Mission_3$ taskset -c 0 ./receiver &
sleep 0.5
taskset -c 2 ./sender
[1] 16317
Receiver: Threshold = 14352384
Receiver: Listening for 1024 bits...
```

```
[1]+ Done taskset -c 0 ./receiver
madhuri@ENG7:~/Mission_3$ make clean
rm -f sender receiver *.o
madhuri@ENG7:~/Mission_3$ make
gcc -Wall -c sender.c -o sender.o
gcc -Wall -c utils.c -o utils.o
gcc -Wall -o sender sender.o utils.o
gcc -Wall -c receiver.c -o receiver.o
gcc -Wall -o receiver receiver.o utils.o
madhuri@ENG7:~/Mission_3$ ./receiver
Calibrated threshold = 9673113
Received message:
4<6;{g-gftrwg once it lview |hc\4@XPPAAAgUec;
/[[a)C
!o+o~^fW2F76:F0R1LWwnm-/n
Accuracy (%): 35.793963
```

```
80 | int received_msg_size = (non_bits - start_idx) / 8;
|
make: *** [Makefile:25: receiver.o] Error 1
madhuri@ENG7:~/Mission3$ make
gcc -Wall -c receiver.c -o receiver.o
gcc -Wall -o receiver receiver.o utils.o
madhuri@ENG7:~/Mission3$ ./receiver
Calibrated threshold = 16751001
Receiver starting to collect 6144 bits...
Received Message:
y'8>xuxuupxsA
#?~8$~?e~x{=B+A
p8D~!C?g?ep Cxm{;
Accuracy: 47.83%
madhuri@ENG7:~/Mission3$ make clean
rm -f sender receiver *.o
```

