IChannels

Exploiting Current Management Mechanisms to Create Covert Channels in Modern Processors

Jawad Haj-Yahya

Jeremie S. Kim *A. Giray Yağlıkçı* *Ivan Puddu* *Lois Orosa*

Juan Gómez Luna *Mohammed Alser* *Onur Mutlu*

SAFARI ETH zürich

Executive Summary

Problem: Current management mechanisms throttle instruction execution and adjust voltage/frequency to accommodate power-hungry instructions (PHIs).
These mechanisms may compromise a system's confidentiality guarantees

Goal:
1. Understand the throttling side-effects of current management mechanisms
2. Build high-capacity covert channels between otherwise isolated execution contexts
3. Practically and effectively mitigate each covert channel

Characterization: Variable execution times and frequency changes due to running PHIs
We observe five different levels of throttling in real Intel systems

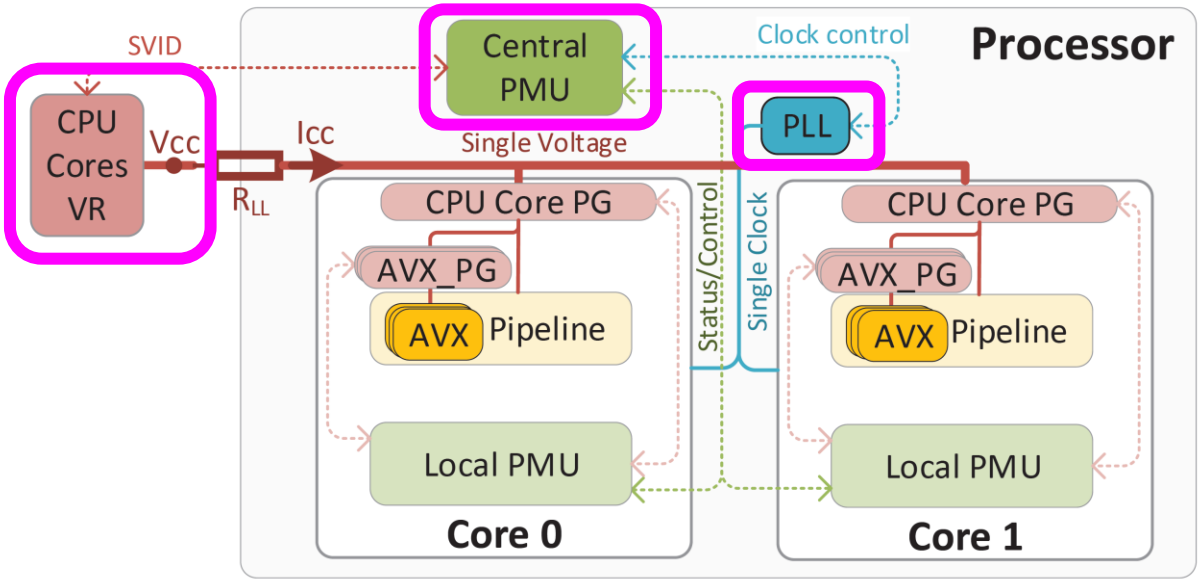
IChannels: New covert channels that exploit side-effects of current management mechanisms
- On the same hardware thread
- Across co-located Simultaneous Multi-Threading (SMT) threads
- Across different physical cores

Evaluation: On three generations of Intel processors, IChannels provides a channel capacity
- 2× that of PHIs' variable latency-based covert channels
- 24× that of power management-based covert channels

Presentation Outline

1. **Overview of Client Processor Architectures**
2. Motivation and Goal
3. Throttling Characterization
4. IChannels Covert Channels
5. Evaluation
6. Conclusion

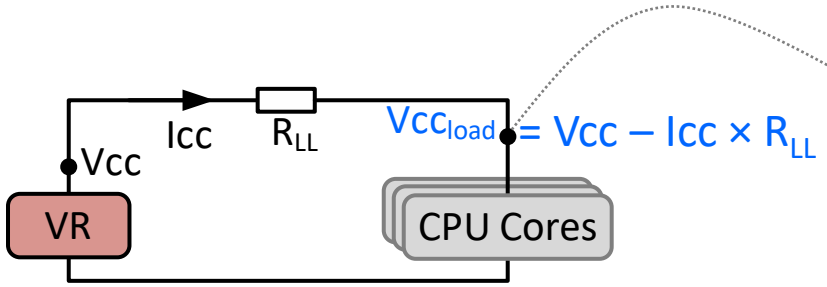
Overview of Client Processor Architectures

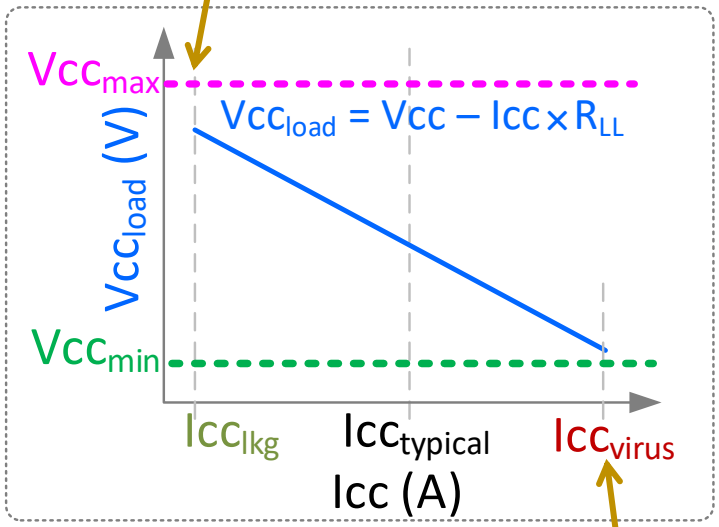


- In many recent processors (e.g., Intel Coffee Lake, Cannon Lake), CPU cores:
  - Share the same voltage regulator (VR) and clock domain
  - Controlled by a central power management unit (PMU)

Load Voltage and Voltage Guardband

Below the maximum operational voltage (Vcc_{max}) under the lightest load (leakage, Icc_{lkg})





Above the minimum functional voltage (Vcc_{min}) under the most intensive load (power-virus, Icc_{virus})

- The relationship between load voltage (Vcc_{load}), supply voltage (Vcc) and current (Icc) under a given system impedance (R_{LL}) is : Vcc_{load} = Vcc – Icc × R_{LL}
- The PMU adds voltage guardband to Vcc to a level that keeps Vcc_{load} within limits

Presentation Outline

1. Overview of Client Processor Architectures
2. **Motivation and Goal**
3. Throttling Characterization
4. IChannels Covert Channels
5. Evaluation
6. Conclusion

Motivation and Goal

- Prior works propose covert channels that exploit some of the throttling side-effects of executing power-hungry instructions (PHIs)
  - These works are limited and use inaccurate observations

- Our goal in this work is to:

1. Experimentally understand the throttling side-effects of current management mechanisms in modern processors to gain several deep insights into how these mechanisms can be abused by attackers

2. Build high-capacity covert channels, IChannels, between otherwise isolated execution contexts

3. Practically and effectively mitigate covert channels caused by current management mechanisms

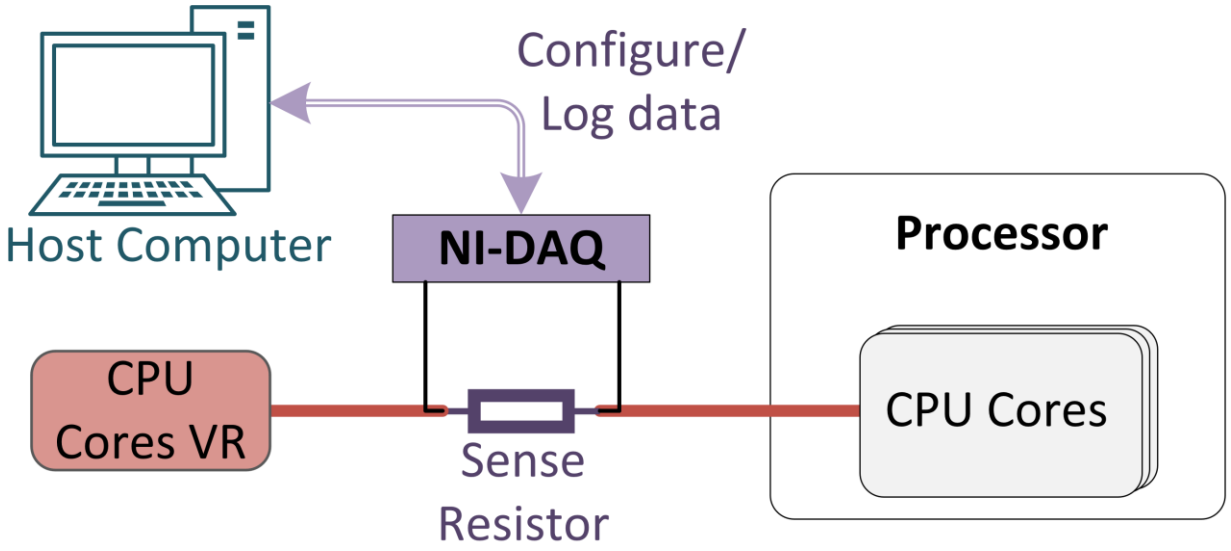
Presentation Outline

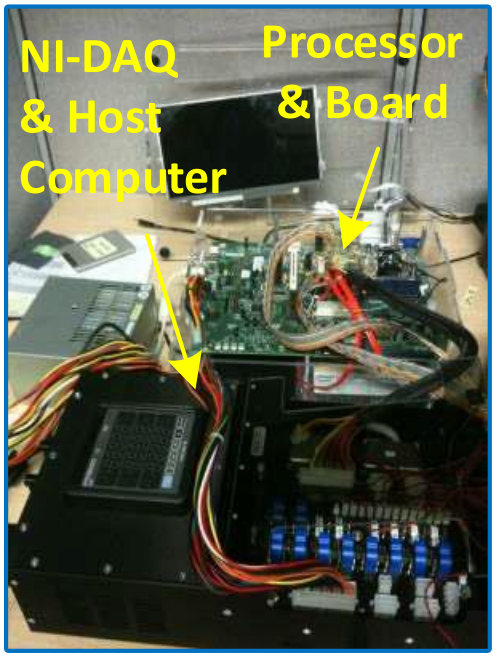
1. Overview of Client Processor Architectures
2. Motivation and Goal
3. **Throttling Characterization**
4. IChannels Covert Channels
5. Evaluation
6. Conclusion

Experimental Methodology

- We experimentally study three modern Intel processors
  - Haswell, Coffee Lake, and Cannon Lake

- We measure voltage and current using a Data Acquisition card (NI-DAQ)





Characterization Insights

- Our rigorous characterization or real system provide deep insights on current management mechanisms

- We find that
  1. The core frequency reduction that directly follows the execution of PHIs at the Turbo frequency is due to maximum instantaneous current limit (Icc_{max}) and maximum voltage limit (Vcc_{max}) protection mechanisms
  2. Power-gating AVX execution units accounts for only ~0.1% of the total throttling time observed when executing PHIs. Most of the throttling time is due to voltage transitions
  3. Current management mechanisms result in a multi-level throttling period depending on the computational intensity of the PHIs
  4. The 4× core IPC reduction that directly follows the execution of PHIs because the core blocks the front-end to back-end uop delivery during 75% of the time for both threads in an SMT Core

Presentation Outline

1. Overview of Client Processor Architectures
2. Motivation and Goal
3. Throttling Characterization
4. **IChannels Covert Channels**
5. Evaluation
6. Conclusion

IChannels Covert Channels

- Threat model consists of two malicious user-level attacker applications, sender and receiver, which cannot communicate through overt channels
- We build three high-throughput covert channels between sender and receiver that exploit throttling side-effects of current management mechanisms
  - On the same hardware thread
  - Across SMT threads, and
  - Across cores
- Each covert channel sends 2 bits from Sender to Receiver in every transaction
  - Each covert channel should wait for reset-time (~650us) before starting a new transaction
  - We demonstrate the covert channels on real Intel Coffee Lake and Cannon Lake systems

Sender

```
case (send_bits[i+1:i])
  00: 128b_Heavy loop()   //L4
  01: 256b_Light_loop()   //L3
  10: 256b_Heavy_loop()   //L2
  11: 512b_Heavy_loop()   //L1
```

Receiver

```
start = rdtsc
if (same-thread) 512b_Heavy_loop()
if (across-SMT)   64b_loop()
if (across-cores) 128b_Heavy_loop()
TP = rdtsc - start
case ( TP )
   L4_range: received_bits[1:0] = 00
   L3_range: received_bits[1:0] = 01
   L2_range: received_bits[1:0] = 10
   L1_range: received_bits[1:0] = 11
```

Across Cores Covert Channel: IccCoresCovert (1/2)

- **IccCoresCovert** covert channel exploits the **Multi-Throttling-Cores** side effect
- **Multi-Throttling-Cores:** when two cores execute PHIs at similar times, the throttling periods (TP) are exacerbated proportionally to the computational intensity of each PHI executed in each core
  - This increase in the TP is because the power management unit (PMU) waits until the voltage transition for core A to complete before starting the voltage transition for core B

T1 TP depends on the computational intensity of Inst0, which determines Vcc level to which the PMU needs to increase the supply voltage before handling T1 voltage transition

T0 and T1 loops are throttled (IPC=1/4)

T1 continues to be throttled since the PMU will not handle T1 voltage transition until T0 voltage target is reached

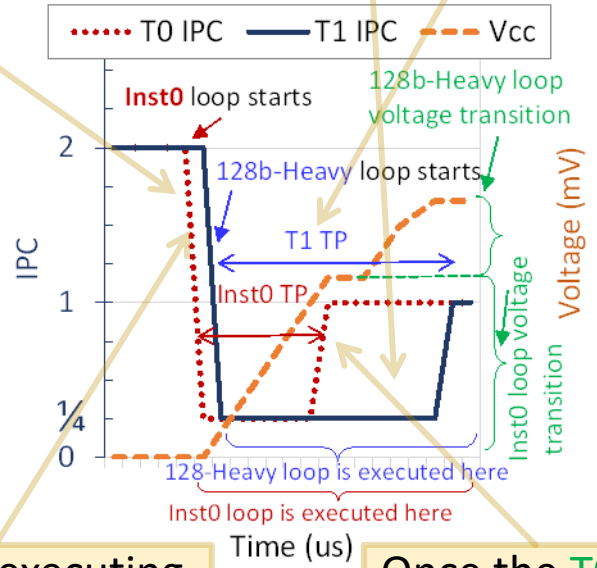
T0 (Core0):

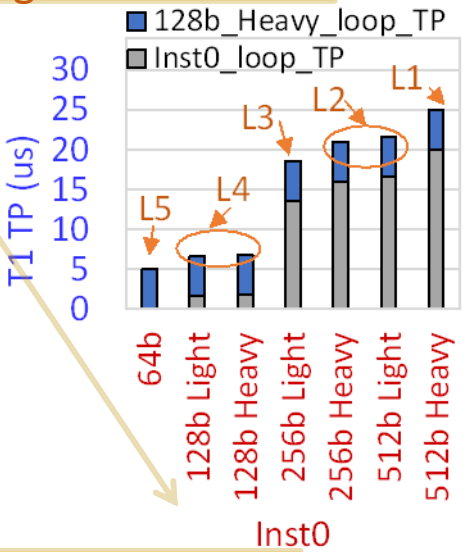
...

T1 (Core1):

...

...





T0/T1 in core0/1 starts executing Inst0/128b-Heavy loop with IPC=1

Once the T0 target Vcc is reached, T0 throttling is stopped (IPC = 1)

Across Cores Covert Channel: IccCoresCovert (2/2)

Sender

```
case (send_bits[i+1:i])
  00: 128b_Heavy loop()   //L4
  01: 256b_Light_loop()   //L3
  10: 256b_Heavy_loop()   //L2
  11: 512b_Heavy_loop()   //L1
```

Receiver

```
start = rdtsc
if (same-thread) 512b_Heavy_loop()
if (across-SMT)  64b_loop()
if (across-cores) 128b_Heavy_loop()
TP = rdtsc - start
case ( TP )
  L4_range: received_bits[1:0] = 00
  L3_range: received_bits[1:0] = 01
  L2_range: received_bits[1:0] = 10
  L1_range: received_bits[1:0] = 11
```

- **IccCoresCovert** exploits the **Multi-Throttling-Cores** side-effect to build a covert channel between Sender and Receiver:
- The Sender executes a PHI loop with a computational intensity level (L1–L4) depending on the values of two secret bits it wants to send
- The Receiver can infer the two bits sent by the Sender based on the measured TP of the 128b_Heavy loop
  - The higher the power required by the PHI loop executed by the Sender, the higher the TP experienced by the Receiver will be

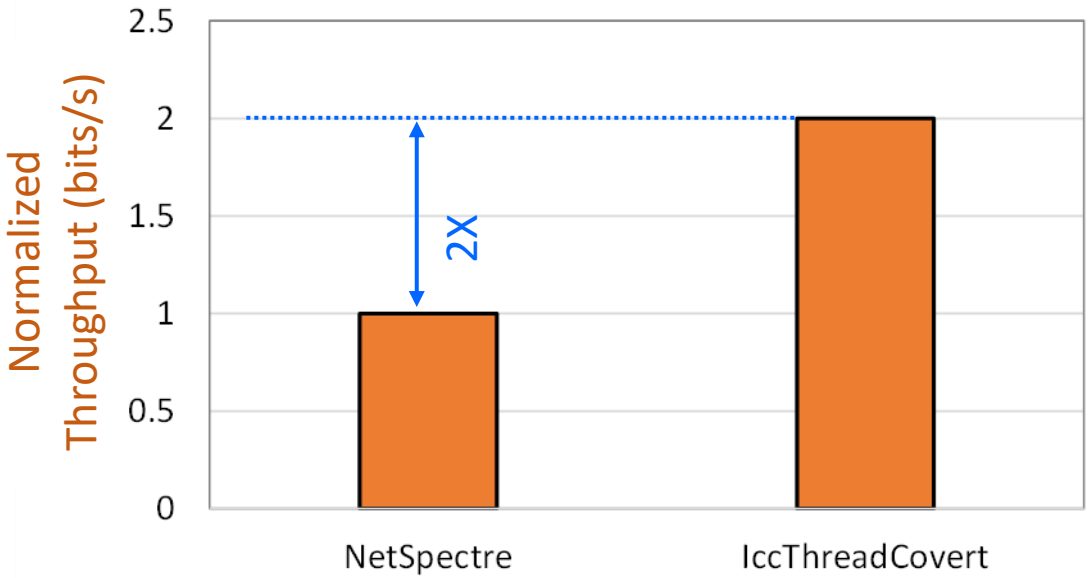
Presentation Outline

1. Overview of Client Processor Architectures
2. Motivation and Goal
3. Throttling Characterization
4. IChannels Covert Channels
5. **Evaluation**
6. Conclusion

Methodology

- **Framework:** We evaluate IChannels on Coffee Lake and Cannon Lake
- **Workloads**: Proof-of-concept codes of the covert channels
- **Comparison Points**: We compare IChannels to four recent works
- **Mitigation Mechanisms**: We propose three mitigation mechanisms

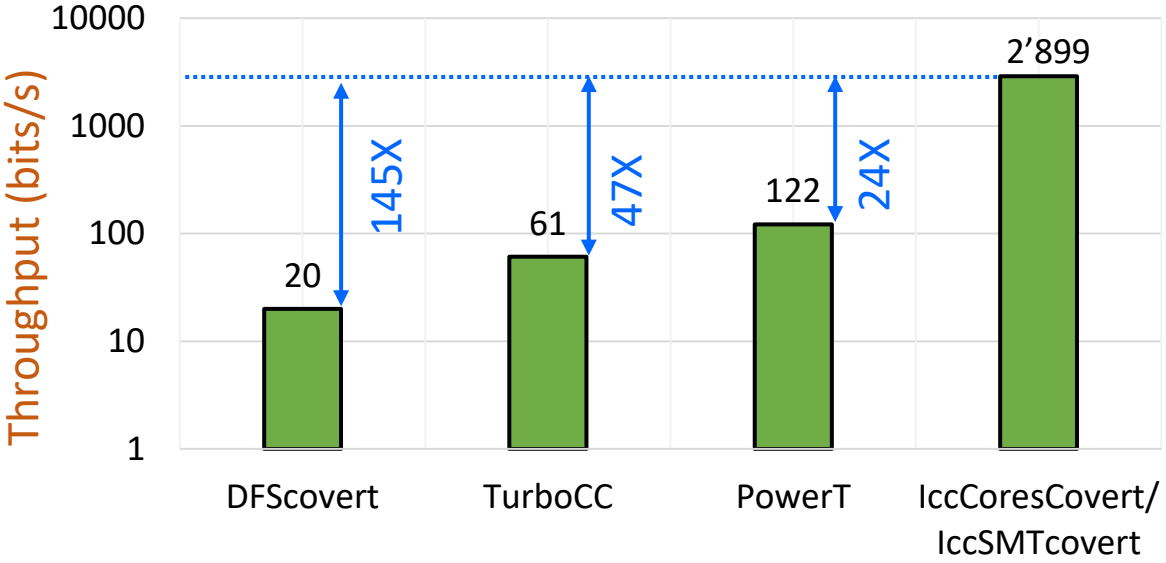
Results – IccThreadCovert



- We compare IccThreadCovert against NetSpectre
  - The state-of-the-art work that exploits the variable latency of PHIs to create a covert channel between two execution contexts running on the same hardware thread

- The NetSpectre covert channel can send one bit per transaction
  - IccThreadCovert covert channel can send two bits per transaction

Results – IccSMTcovert & IccCoresCovert



- We compare IccSMTcovert and IccCoresCovert against DFScovert, TurboCC and PowerT
  - The state-of-the-art works that exploit different power management mechanisms of modern processors to build covert channels across cores and SMT threads
- IccSMTcovert/IccCoresCovert throughput is 145×, 47×, and 24×
  - The throughput of DFScovert, TurboCC, and PowerT, respectively
- The three works exploit slow mechanisms (e.g., frequency/thermal changes)
  - Compared to the current management side-effects that our IChannels exploits

Presentation Outline

1. Overview of Client Processor Architectures
2. Motivation and Goal
3. Throttling Characterization
4. IChannels Covert Channels
5. Evaluation
6. **Conclusion**

Conclusion

- We introduced IChannels, new covert channels we can exploit based on current management mechanisms in modern processors
- We showed that the multi-level throttling effects of current management mechanisms can be exploited for malicious information leakage on real systems (Intel Cannon Lake & Intel Coffee Lake)
- IChannels is 2× and 24× higher throughput than existing state-of-the-art covert channels
- We propose multiple practical mitigations to protect against IChannels in modern processors
- We hope our work paves the way for eliminating the confidentiality breaches such current management mechanisms lead to

IChannels

Exploiting Current Management Mechanisms to Create Covert Channels in Modern Processors

Jawad Haj-Yahya

Jeremie S. Kim *A. Giray Yağlıkçı* *Ivan Puddu* *Lois Orosa*

Juan Gómez Luna *Mohammed Alser* *Onur Mutlu*

SAFARI ETH zürich