Defeating Transient Execution Attacks by Limiting Secret Reachability through

REGISTER HIDING & SHADOWCFI

Daniël Trujillo, Jagadish Kotra, David Kaplan, Mengjia Yan

Teaser

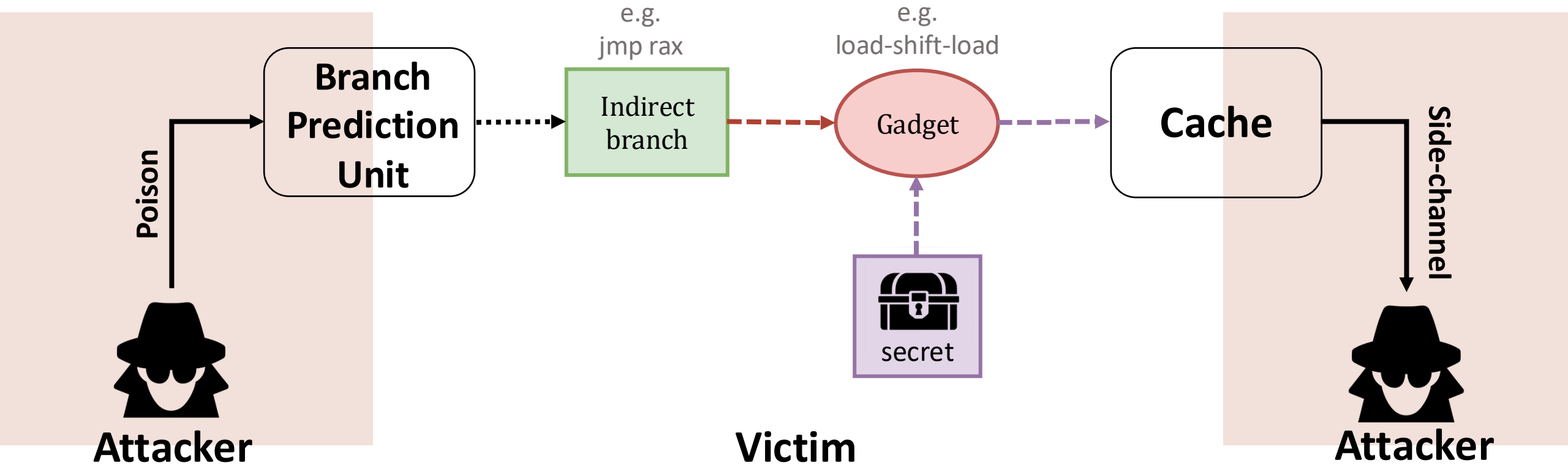
Spectre v2 attacks continue to be found,
requiring new mitigations constantly

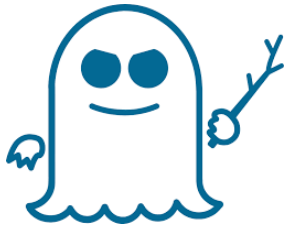
Previous defenses overlook
one attack ingredient: **secret reachability**

REGISTER HIDING & **SHADOWCFI**
hide register states during misprediction

Applying **RH** & **SCFI** to the Linux kernel **yields better performance than default mitigation** on AMD Zen 4

Spectre v2 Remains a Big Threat



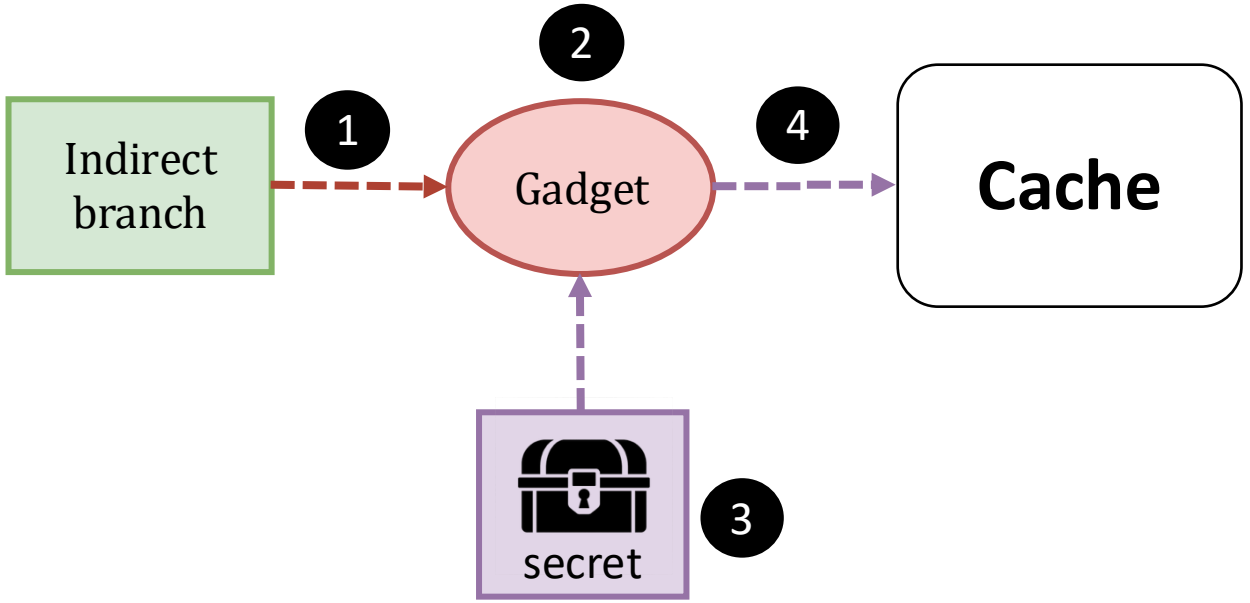


Original Spectre v2 (2018)
Retbleed (2022)
Branch History Injection (2022)

Phantom (2023)
Inception (2023)
InSpectre Gadget(2024)

Breaking the Barrier (2025)
Training Solo (2025)
Branch Privilege Injection (2025)

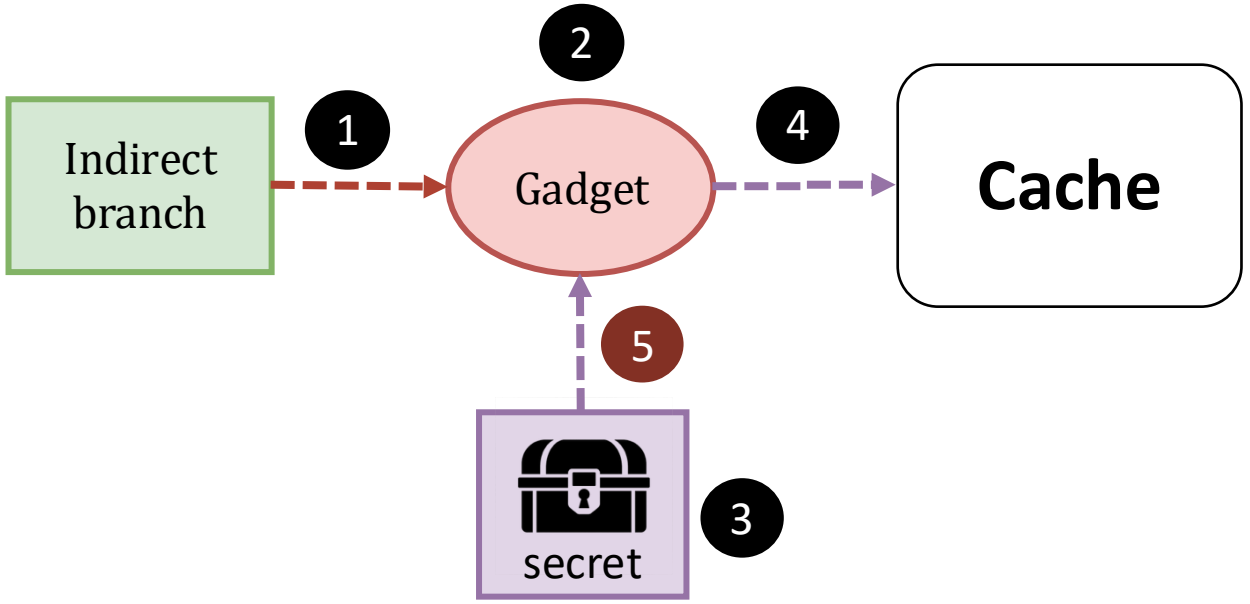
Proposed Spectre v2 Mitigations



1. **Blocking mispredictions**
2. **Removing gadgets**
3. **Removing secrets**
4. **Blocking secret transmission**

Each of these approaches has limitations

Proposed Spectre v2 Mitigations



1 **Blocking mispredictions**

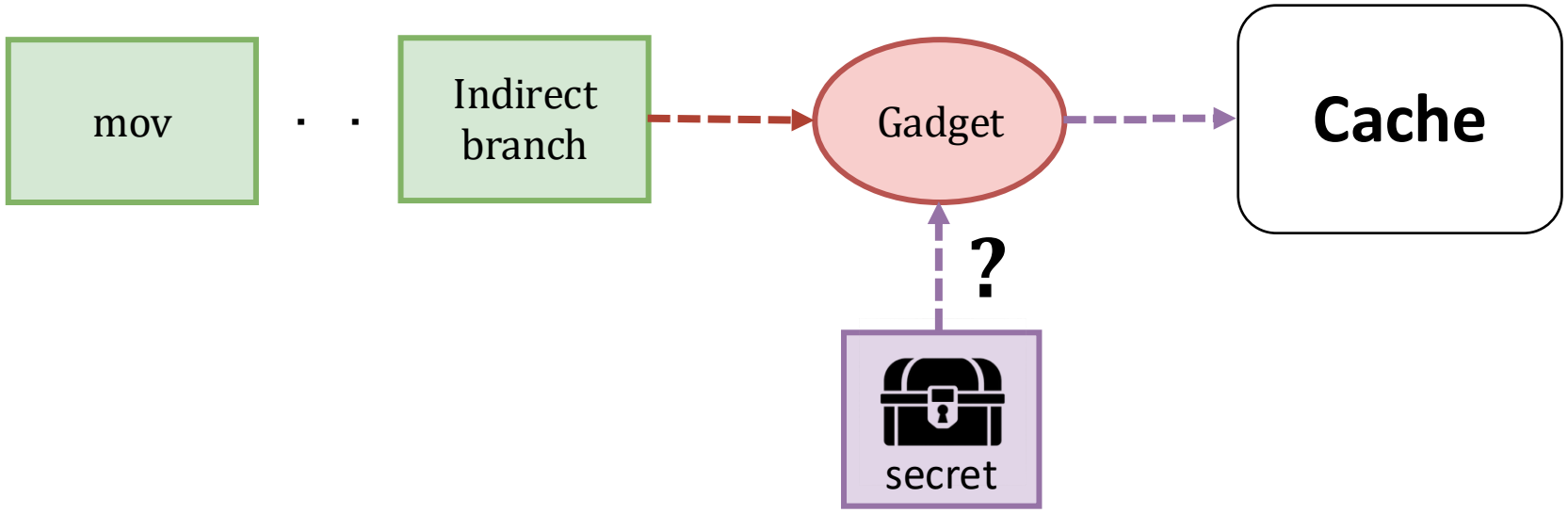
2 **Removing gadgets**

3 **Removing secrets**

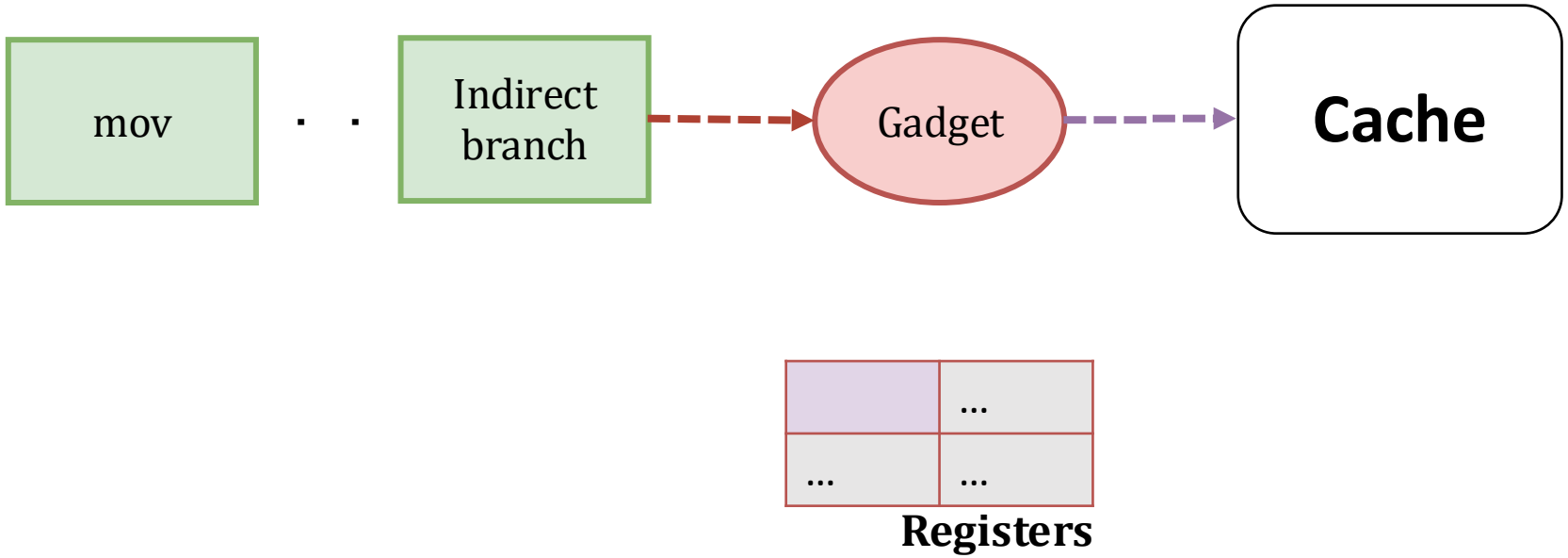
4 **Blocking secret transmission**

What about 5 secret reachability?

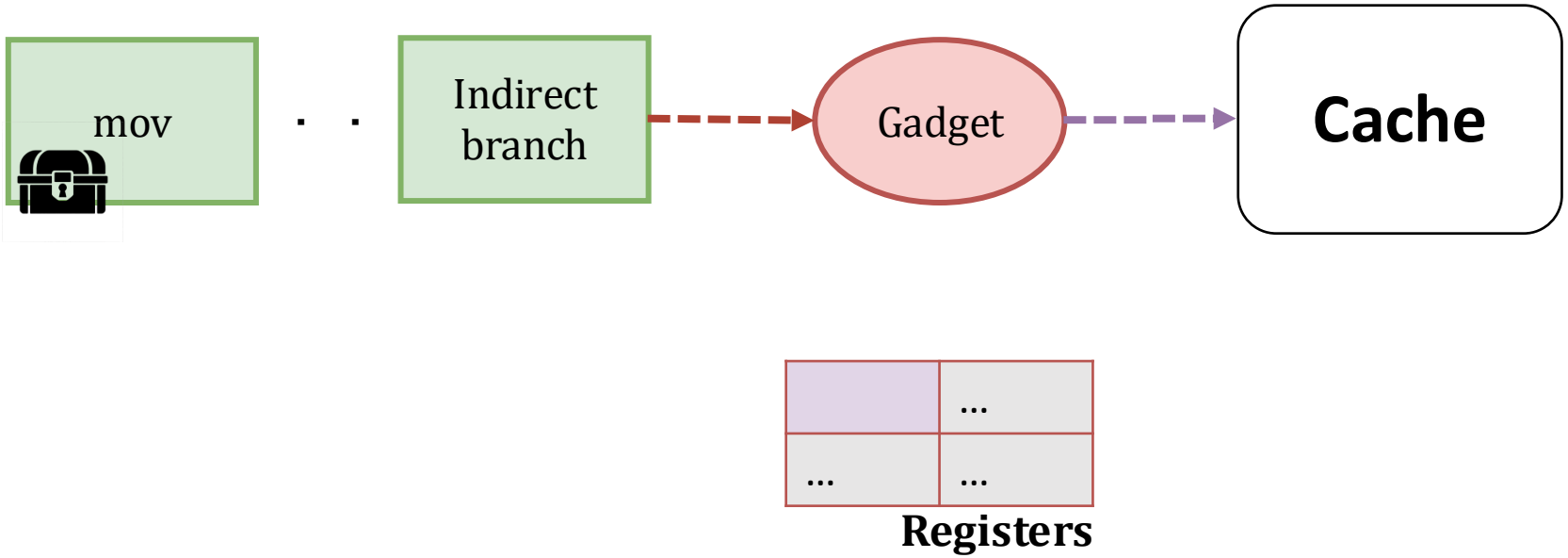
Secret reachability



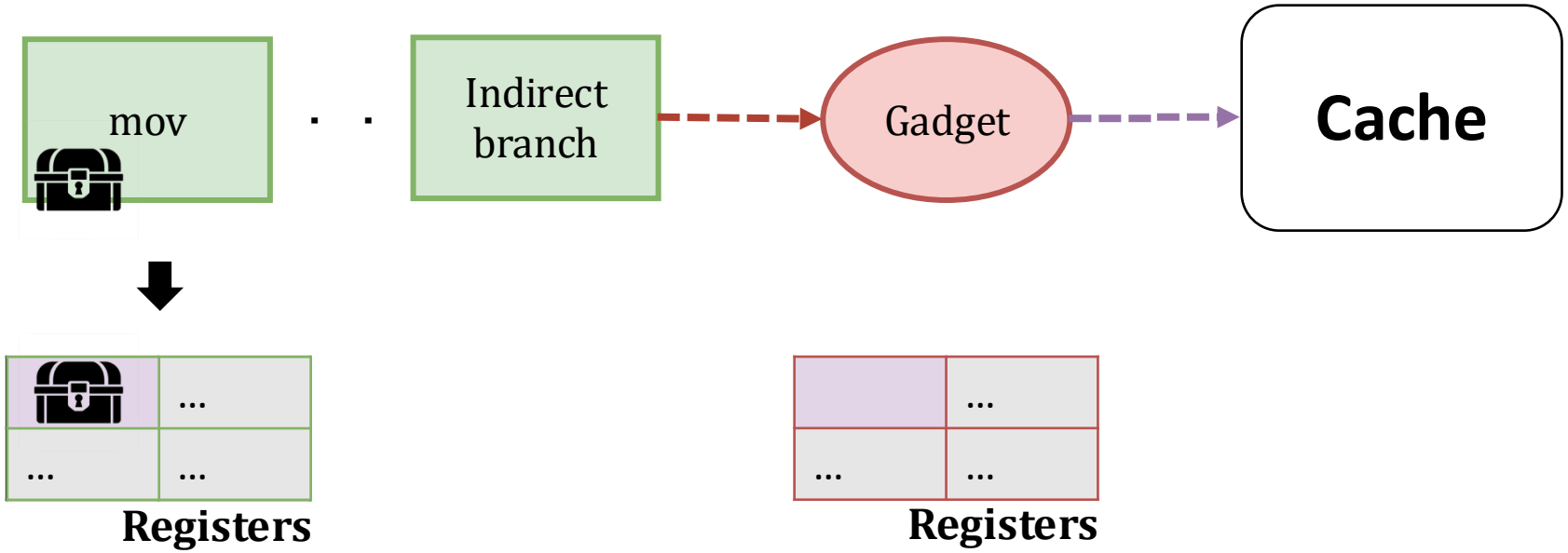
Secret reachability



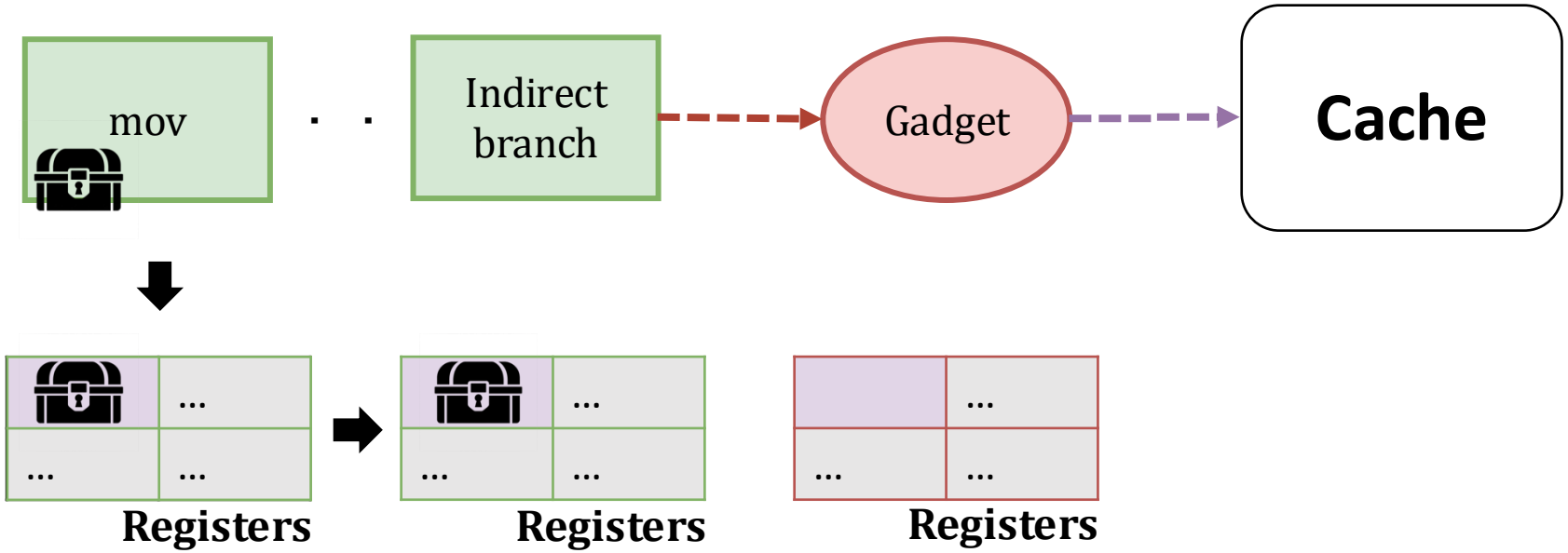
Secret reachability



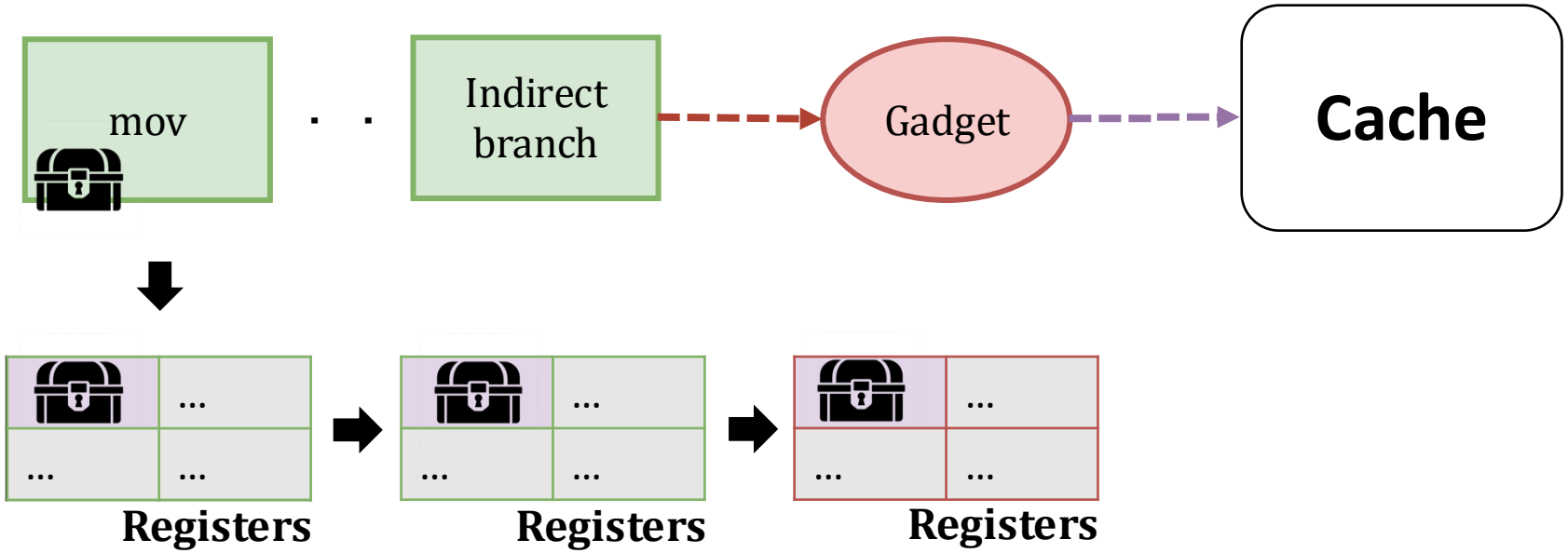
Secret reachability



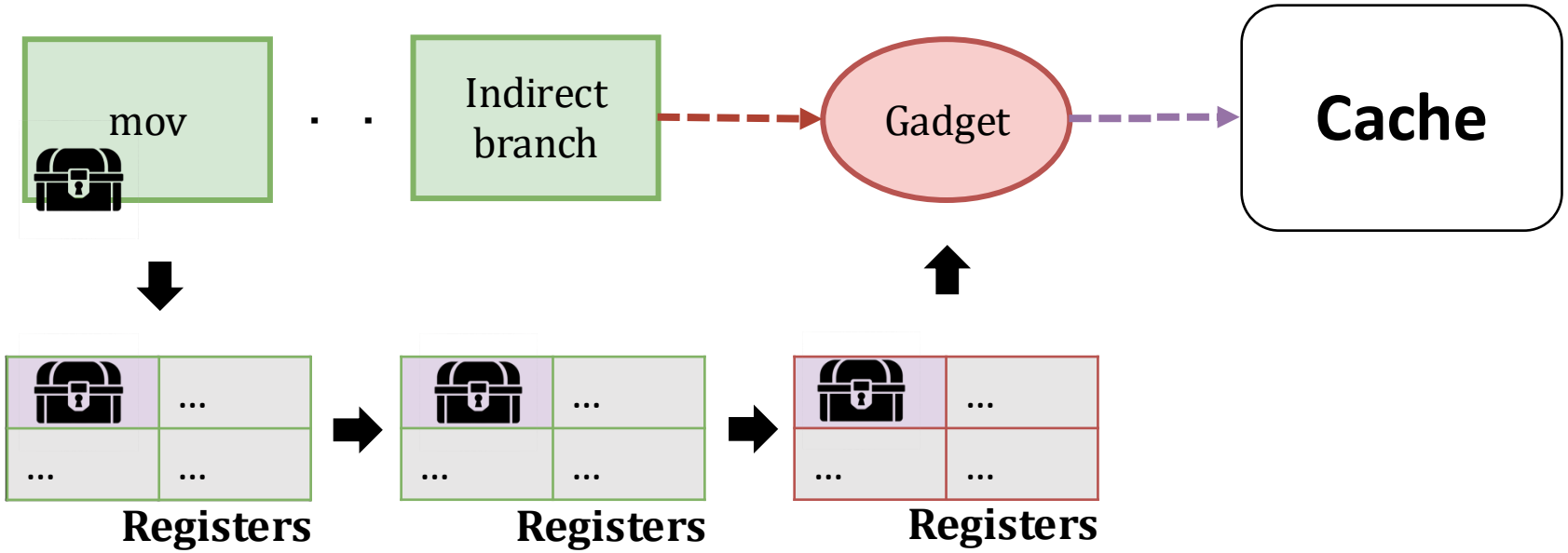
Secret reachability



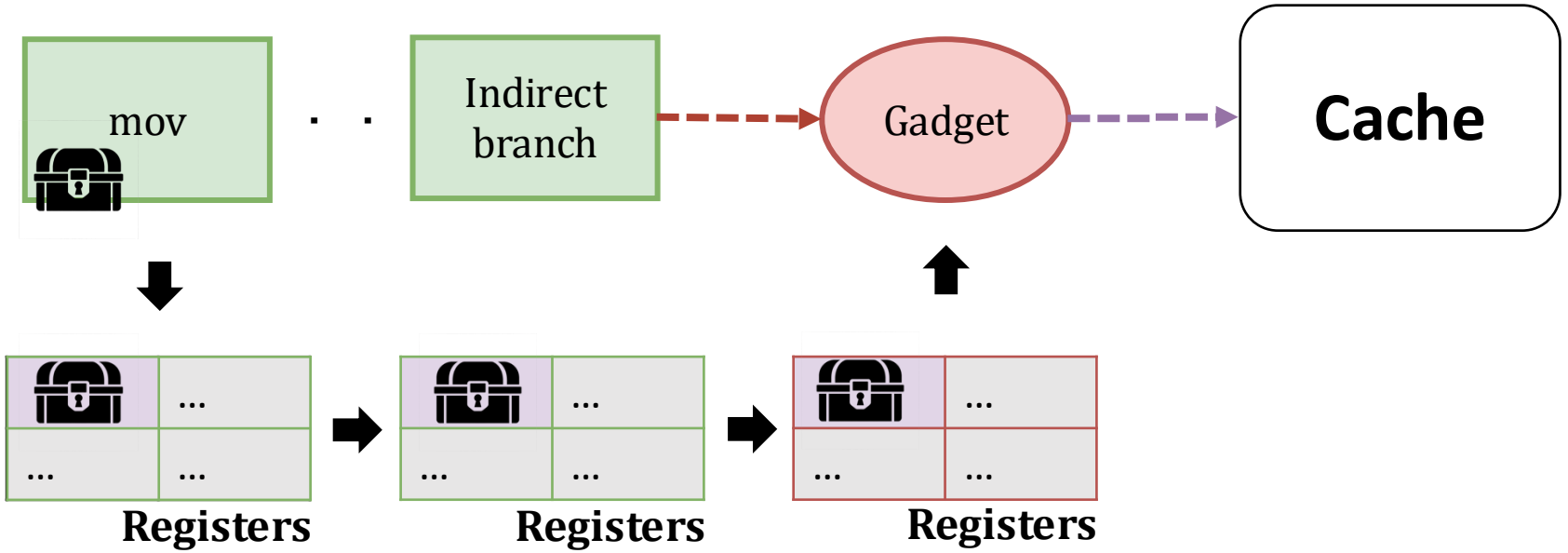
Secret reachability



Secret reachability



Secret reachability



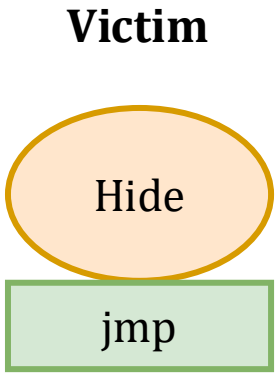
Access to secrets relies on **register values created before misprediction**

REGISTER HIDING

Victim

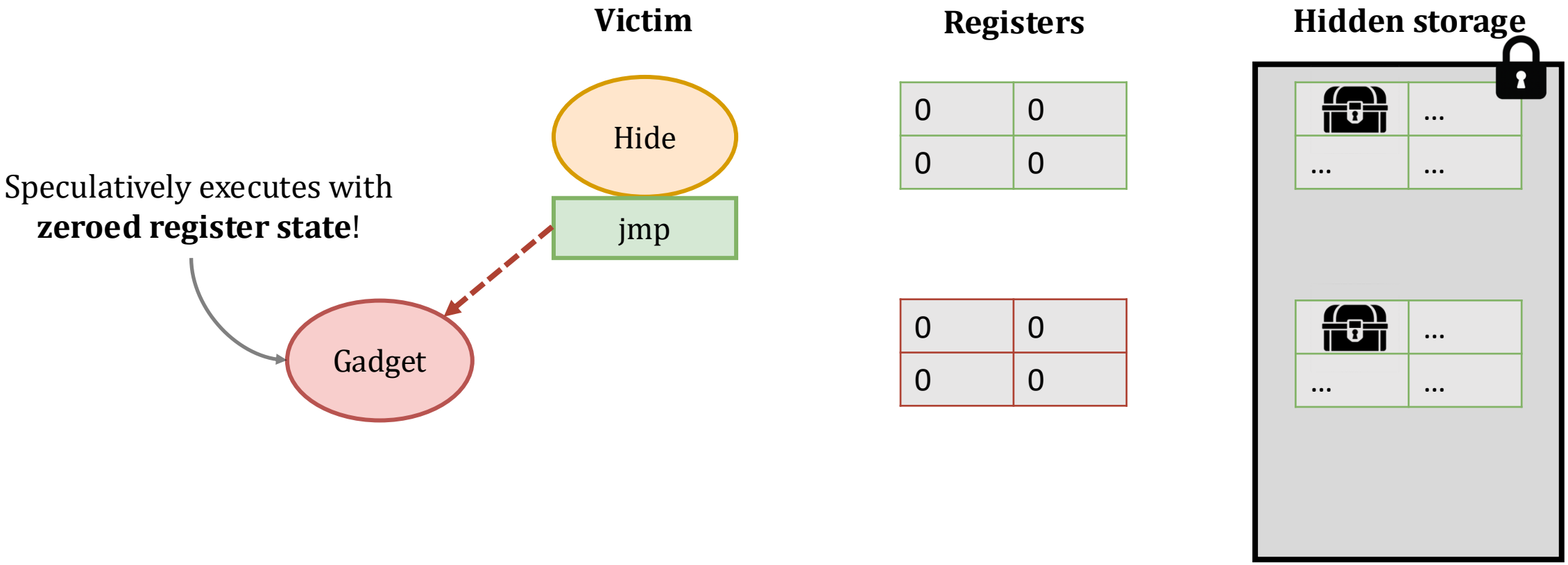
jmp

REGISTER HIDING

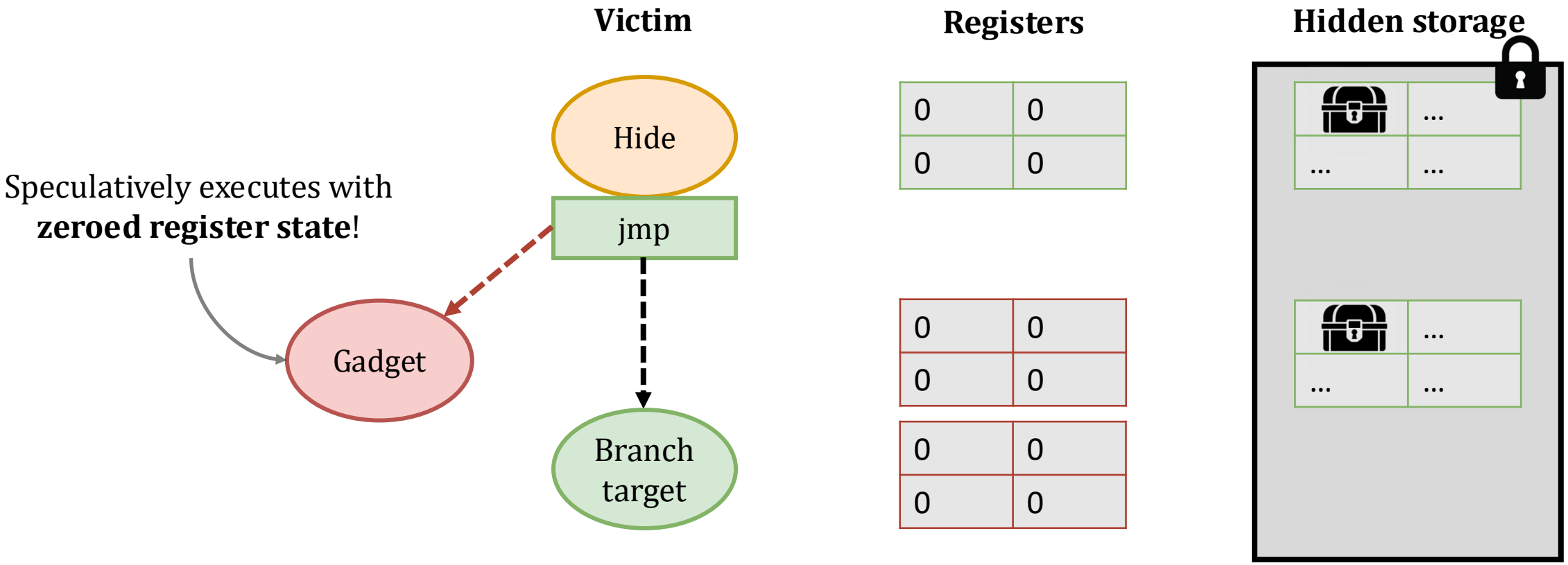


REGISTER HIDING

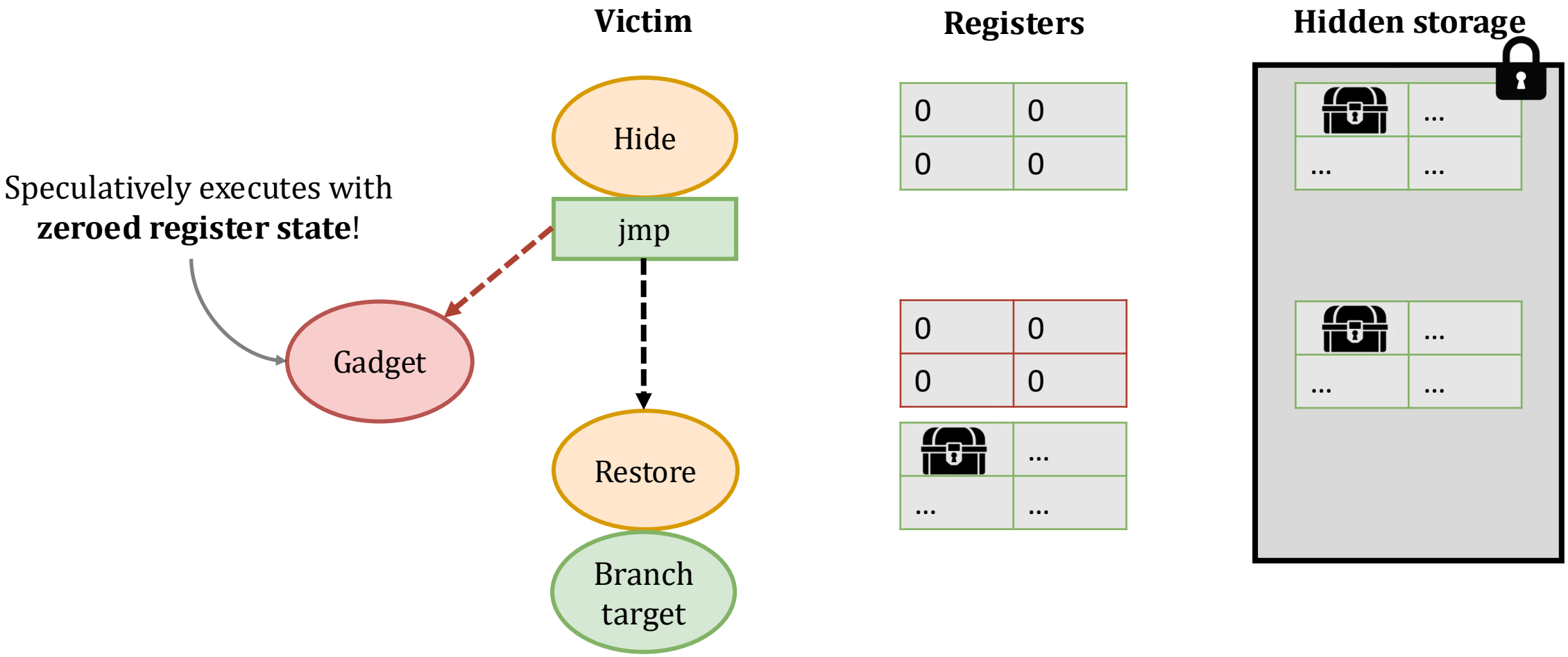
REGISTER HIDING



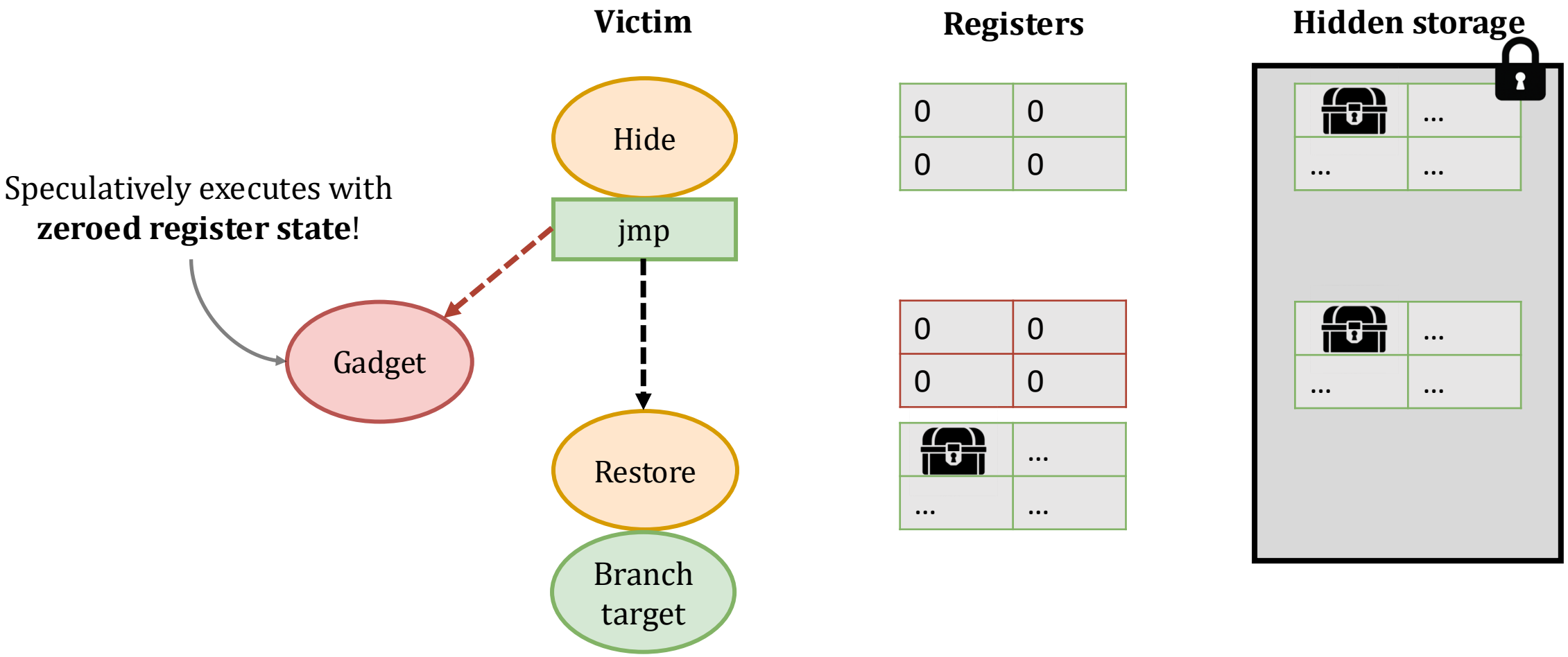
REGISTER HIDING



REGISTER HIDING



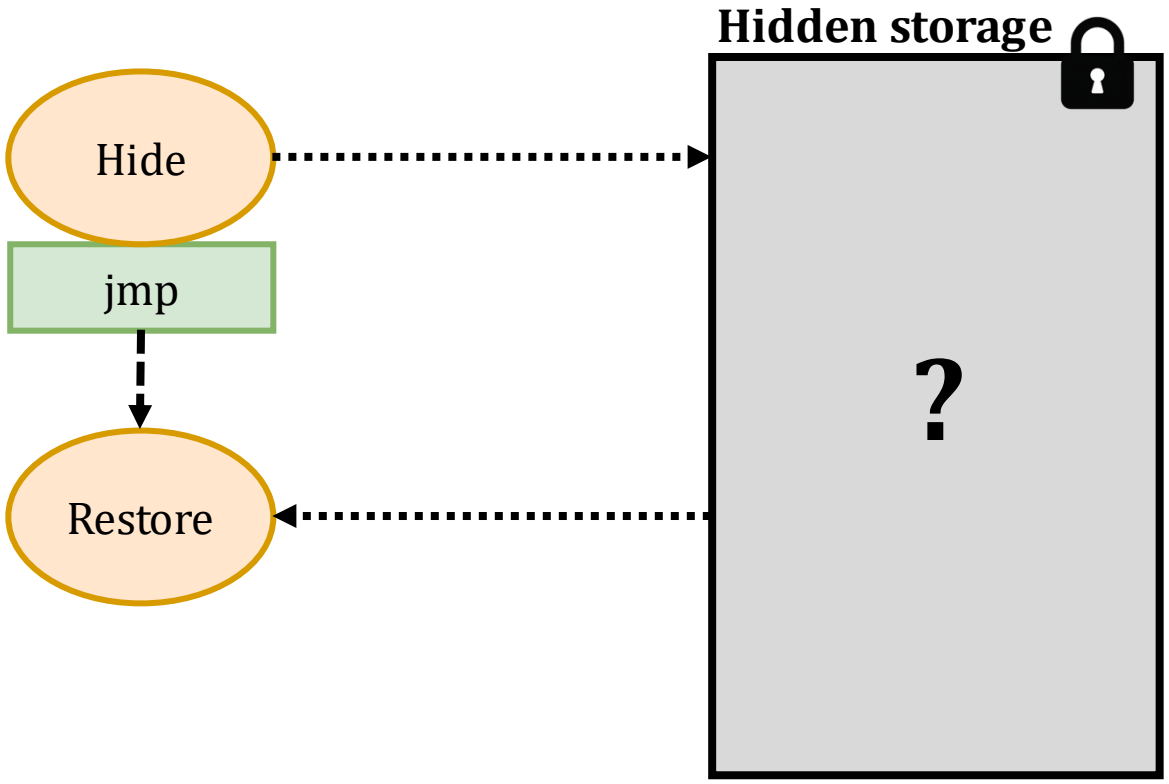
REGISTER HIDING



REGISTER HIDING limits secret reachability by **hiding registers** during misprediction

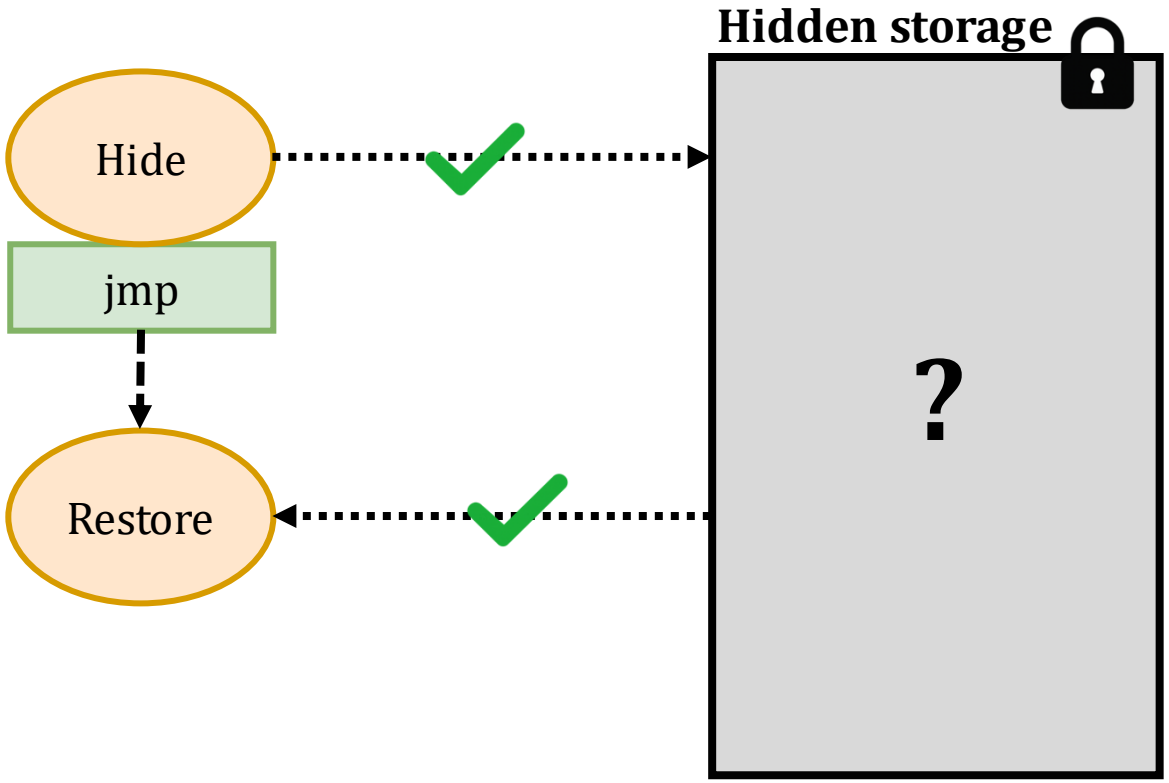
REGISTER HIDING

Goal: accessible only by **Hide** and **Restore** instructions



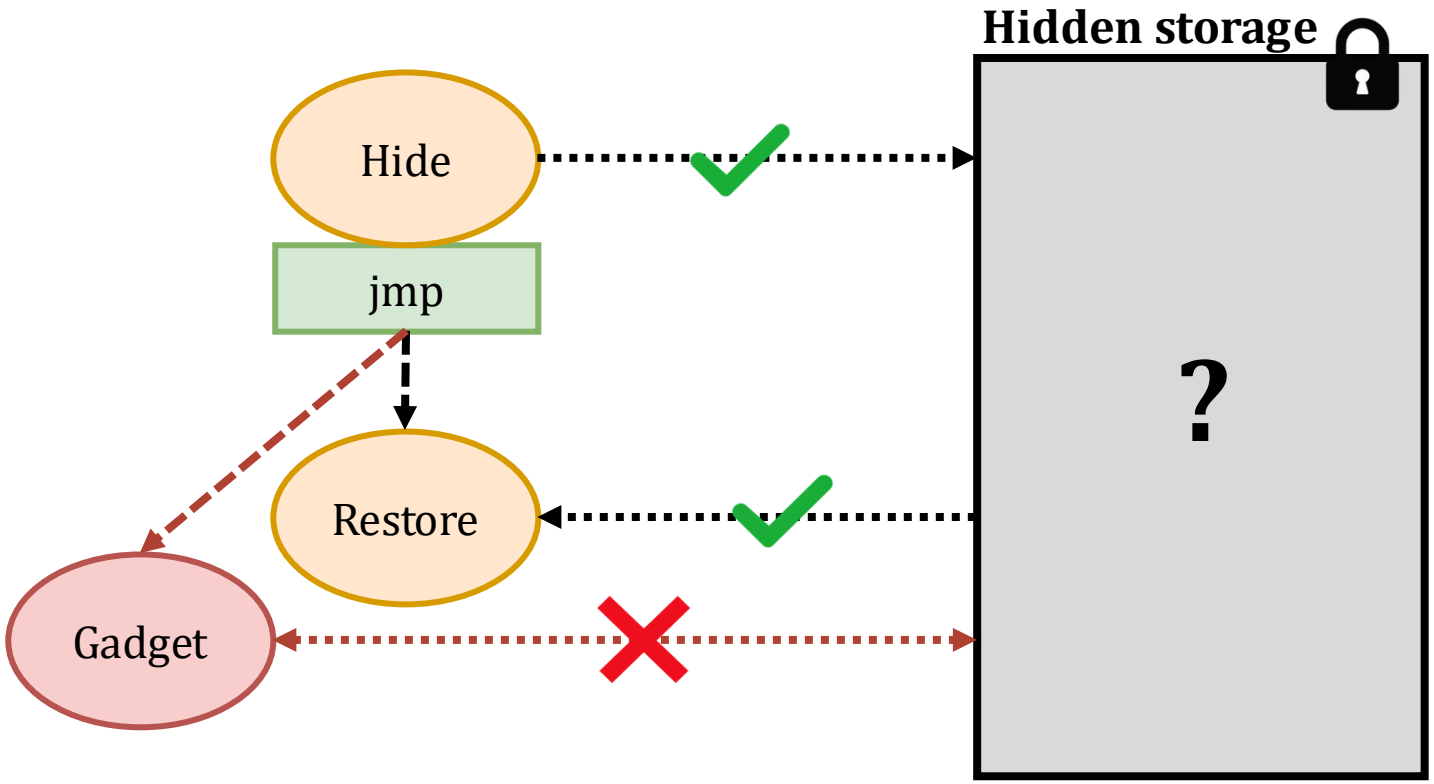
REGISTER HIDING

Goal: accessible only by **Hide** and **Restore** instructions



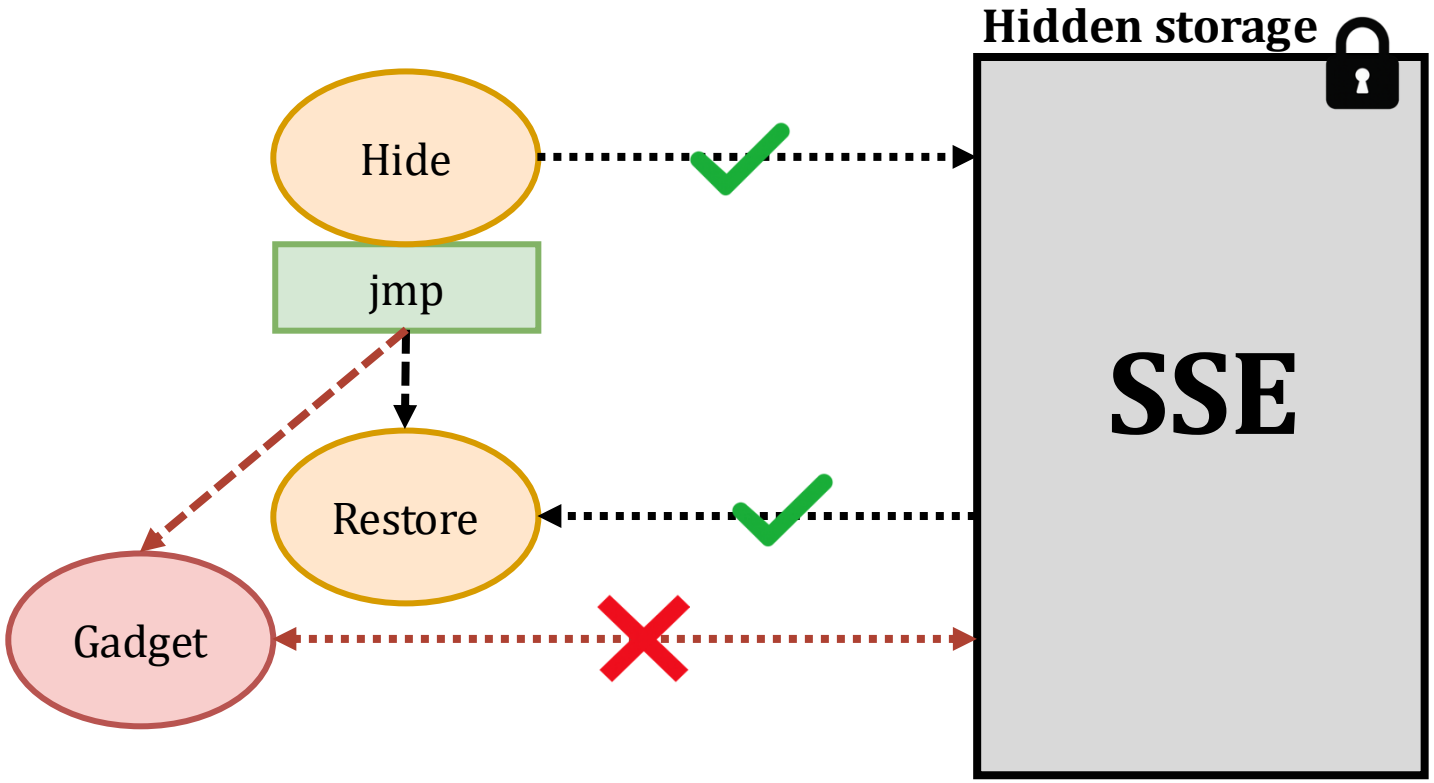
REGISTER HIDING

Goal: accessible only by **Hide** and **Restore** instructions



REGISTER HIDING

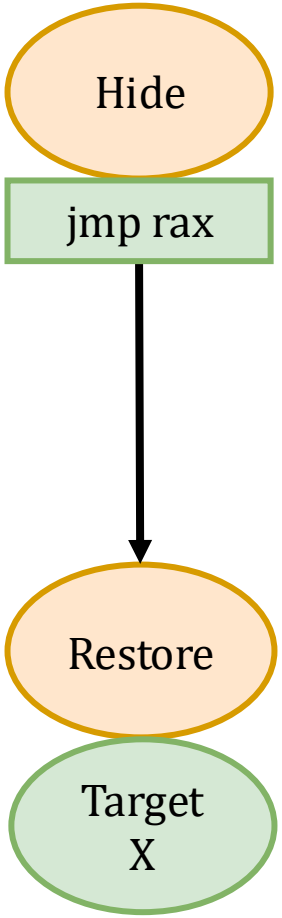
Goal: accessible only by **Hide** and **Restore** instructions



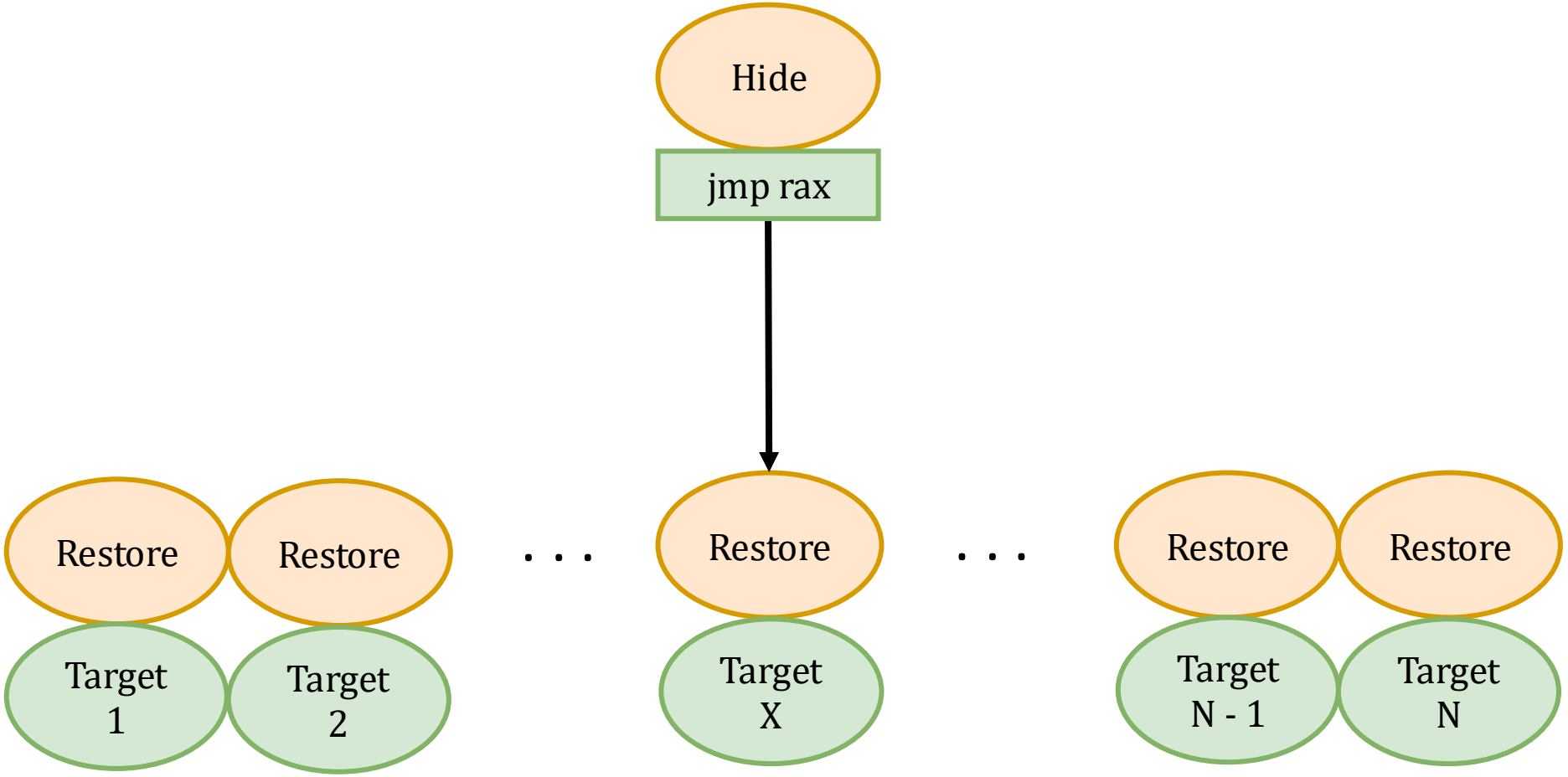
Solution: **SSE registers**

- **Limited instructions** can access SSE registers, **allowing verification of goal**
- **Unused** by many programs, including Linux kernel

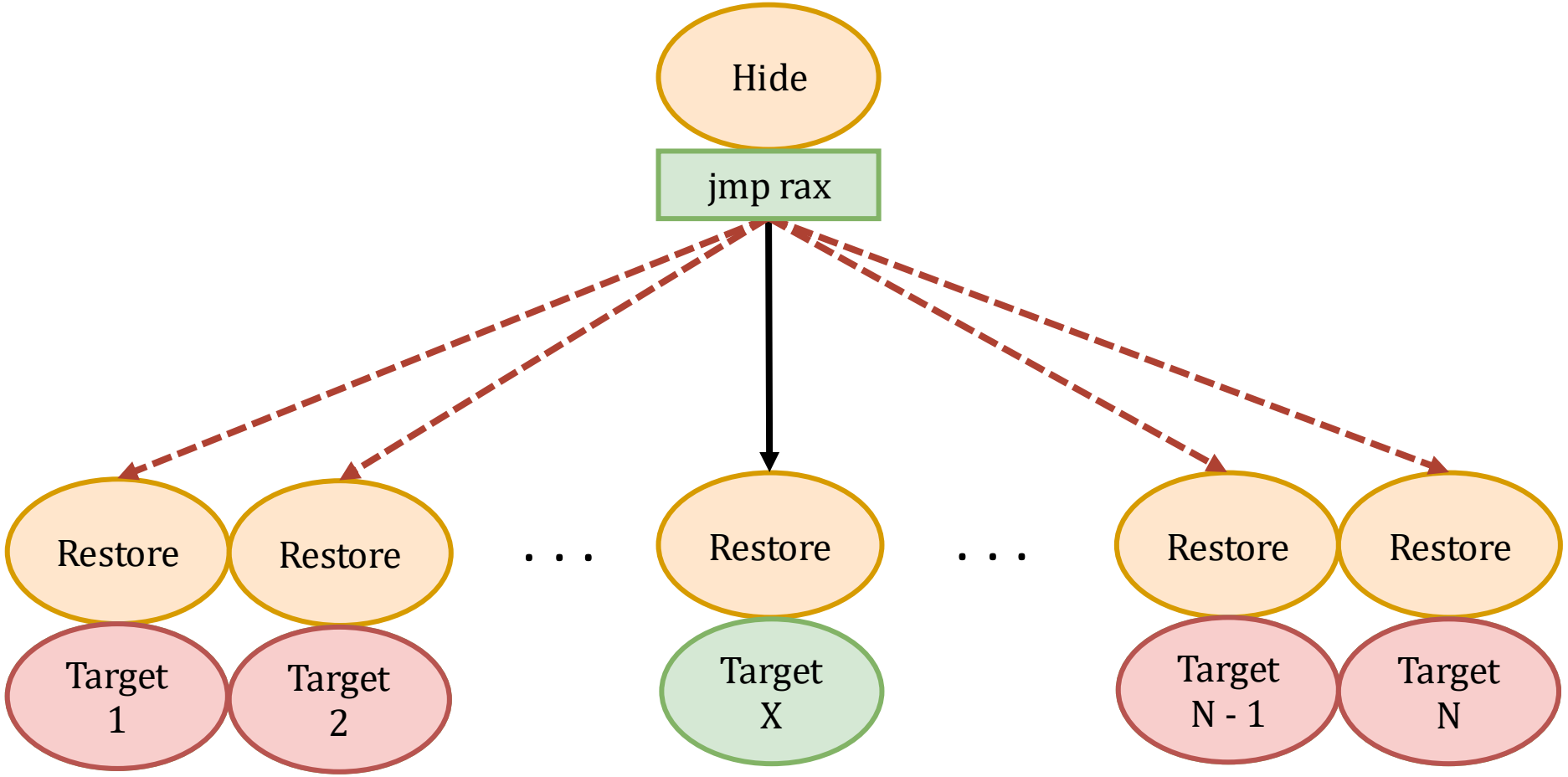
REGISTER HIDING: Remaining Attack Surface



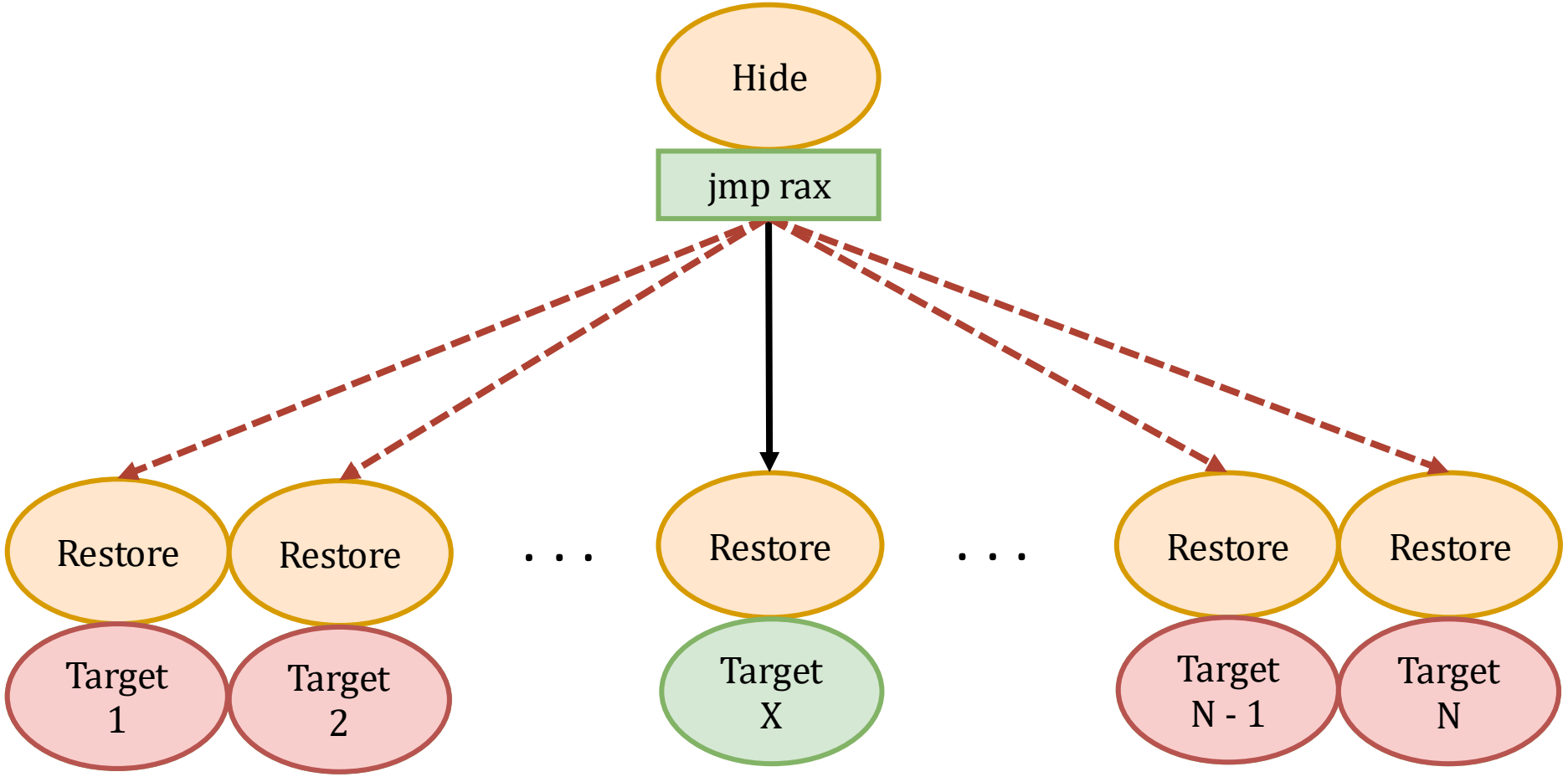
REGISTER HIDING: Remaining Attack Surface



REGISTER HIDING: Remaining Attack Surface



REGISTER HIDING: Remaining Attack Surface



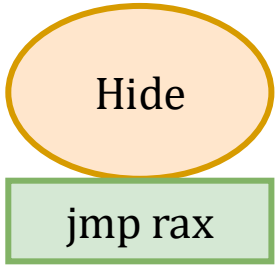
Misprediction to **any** **Restore** gives access to register state!

SHADOWCFI

Compare **non-speculative target** against **speculative instruction pointer**

SHADOWCFI

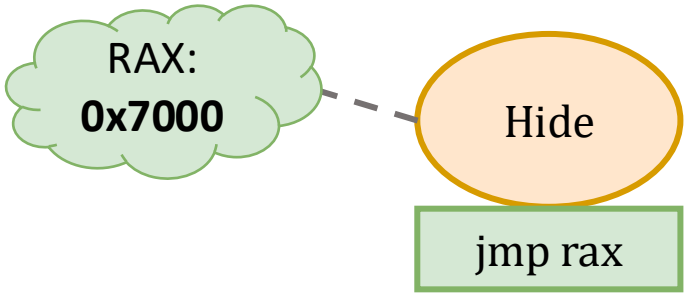
Compare **non-speculative target** against **speculative instruction pointer**



NON-SPECULATIVE

SHADOWCFI

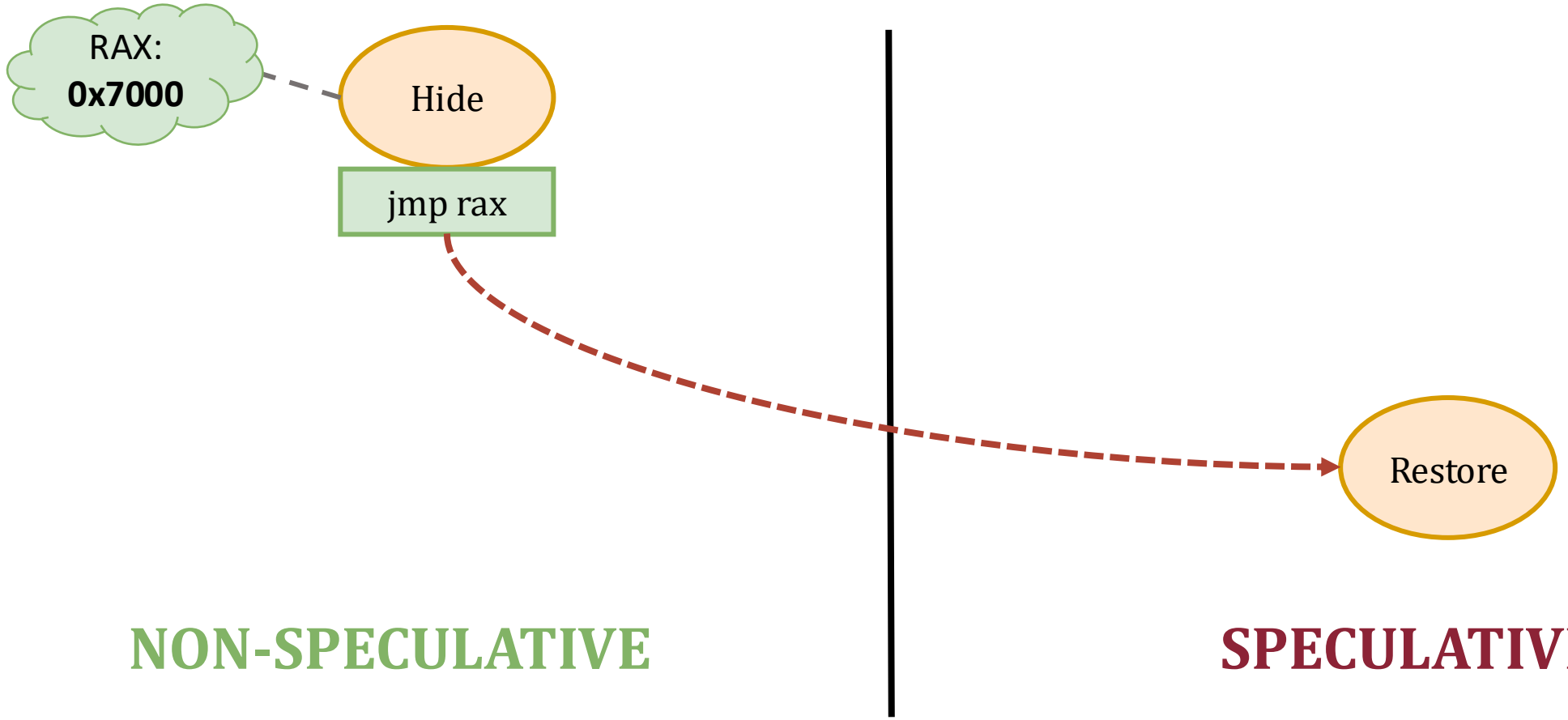
Compare **non-speculative target** against **speculative instruction pointer**



NON-SPECULATIVE

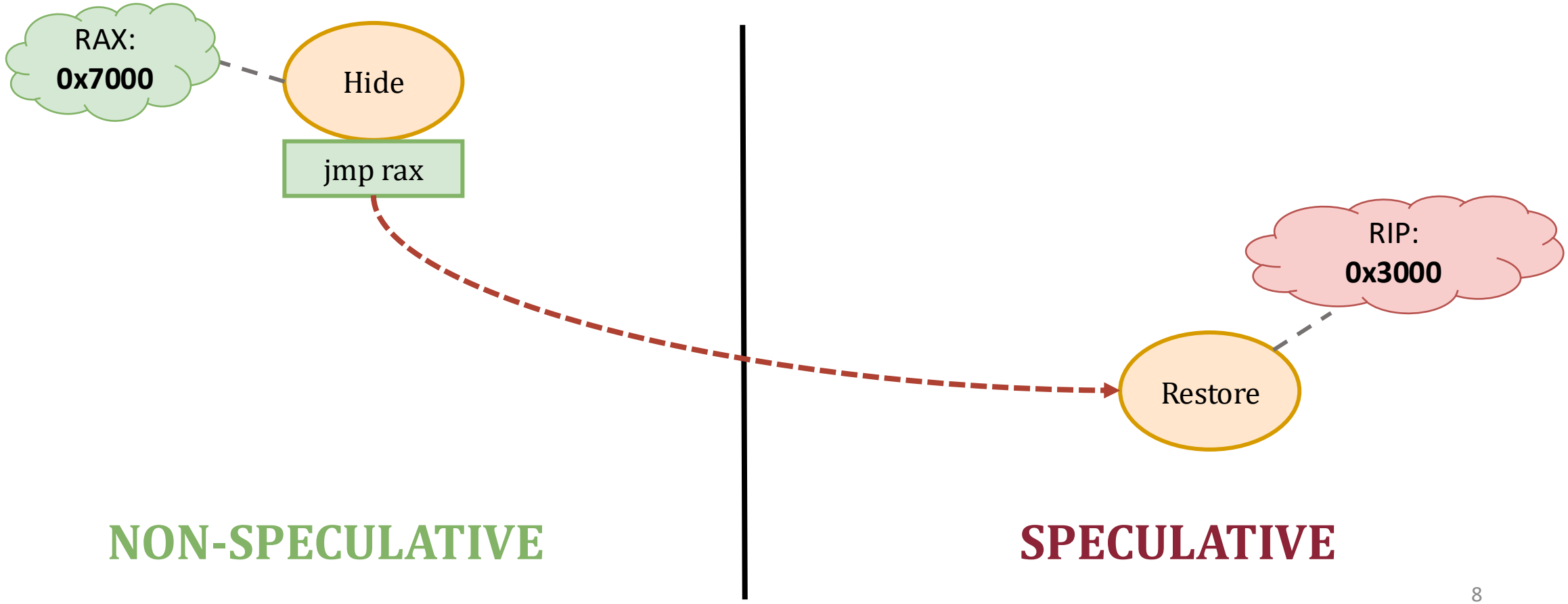
SHADOWCFI

Compare **non-speculative target** against **speculative instruction pointer**



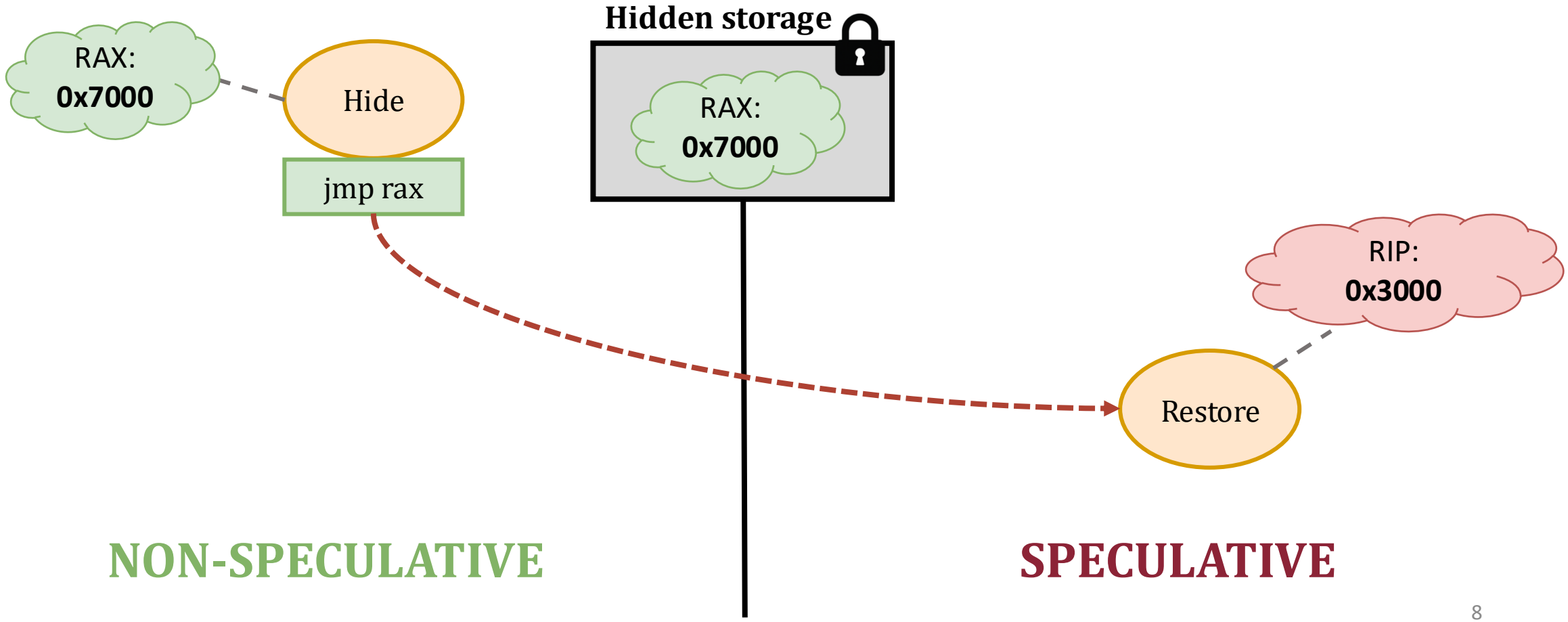
SHADOWCFI

Compare **non-speculative target** against **speculative instruction pointer**



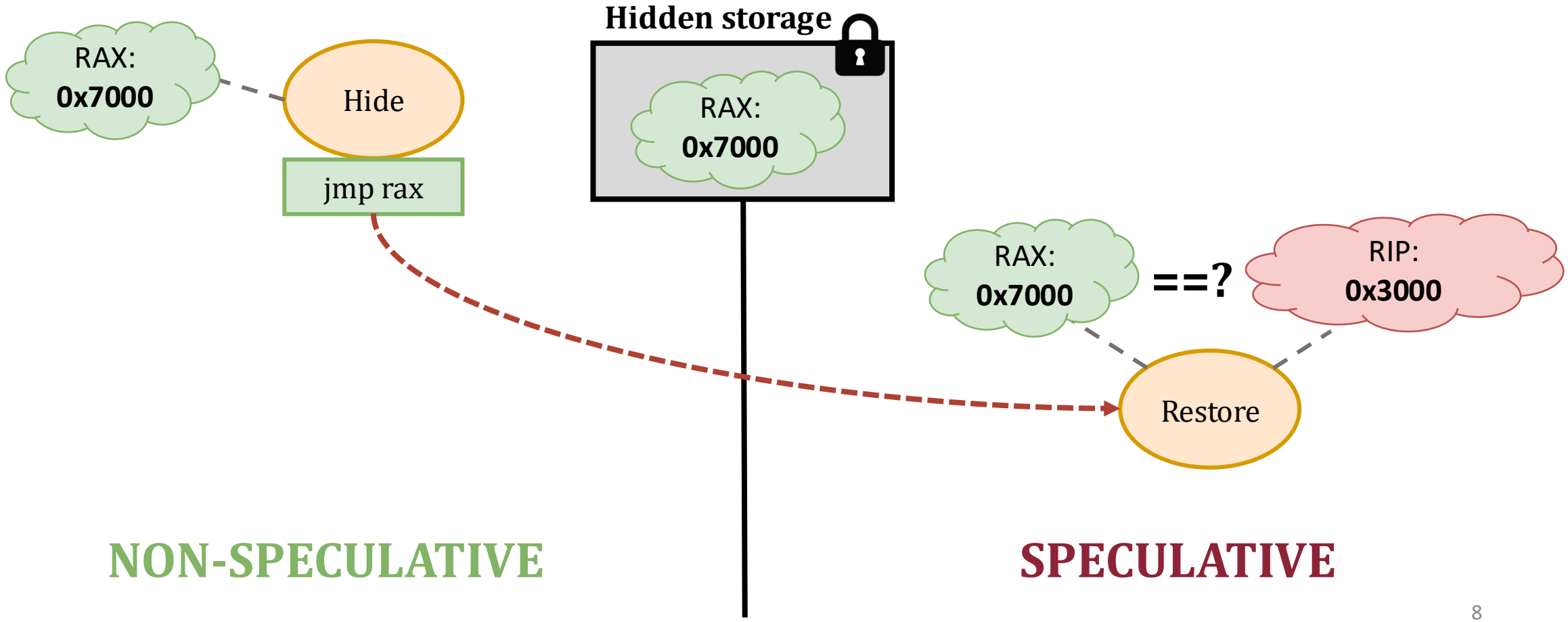
SHADOWCFI

Compare **non-speculative target** against **speculative instruction pointer**

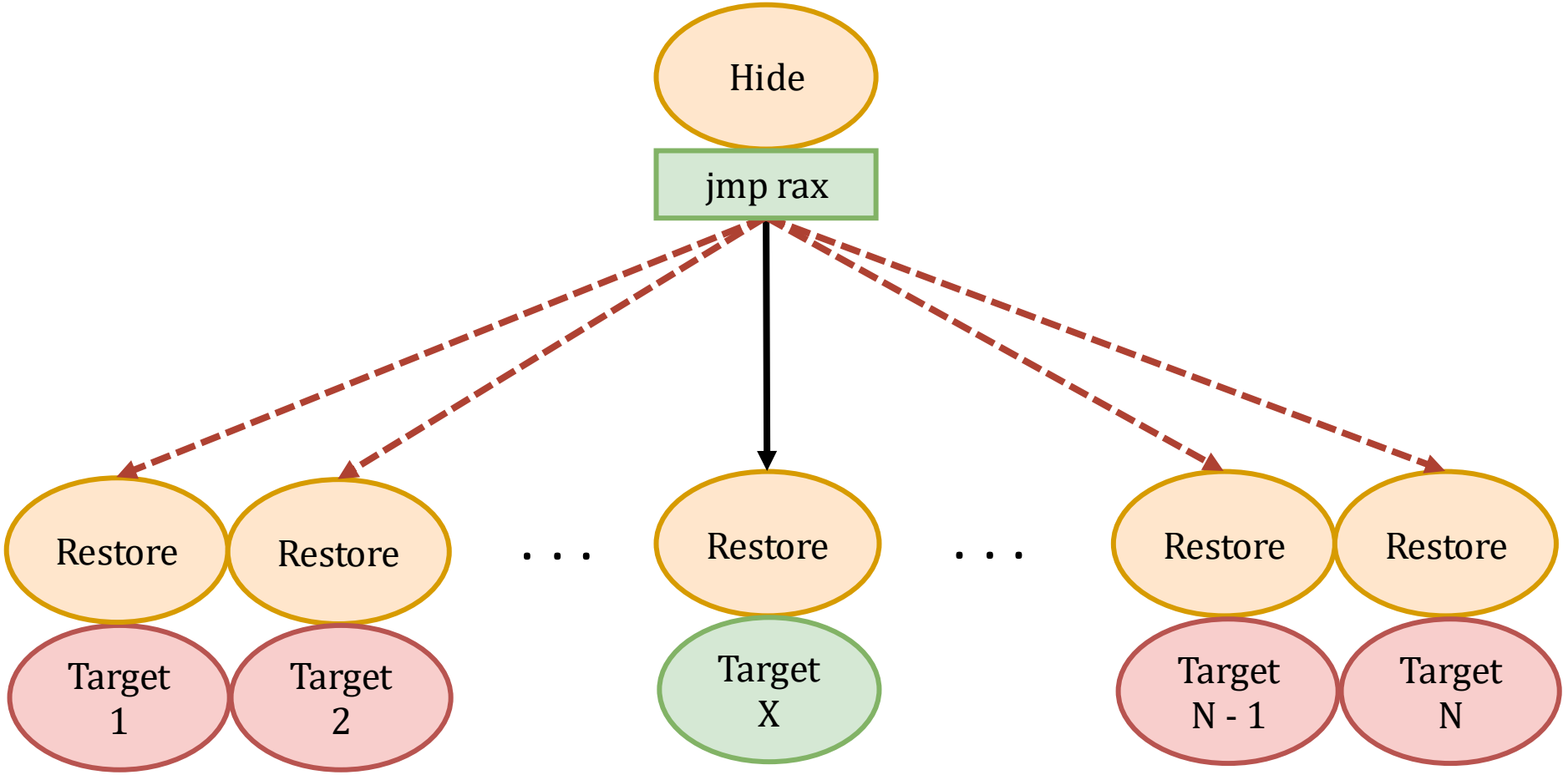


SHADOWCFI

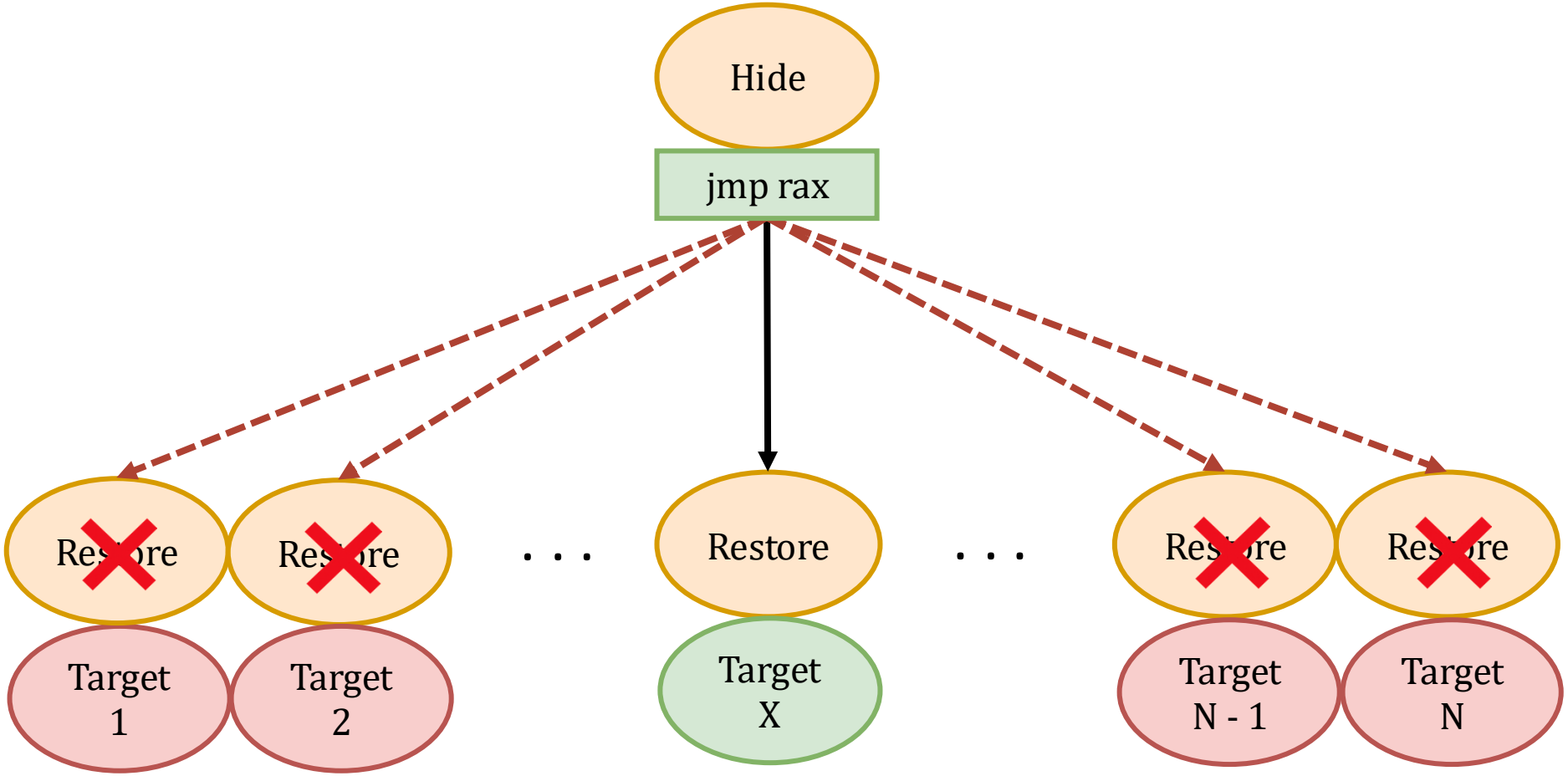
Compare **non-speculative target** against **speculative instruction pointer**



SHADOWCFI


Hide
jmp rax
Restore
Restore
. . .
Restore
. . .
Restore
Restore
Target 1
Target 2
Target X
Target N - 1
Target N

SHADOWCFI



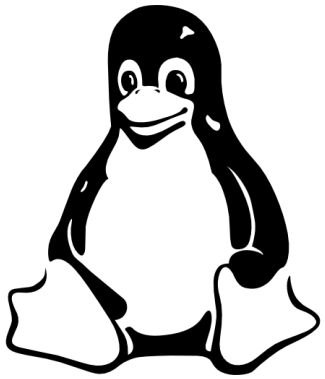
SHADOWCFI restricts register access to **correct branch target**

Implementation for the Linux kernel

GCC 14.0.0

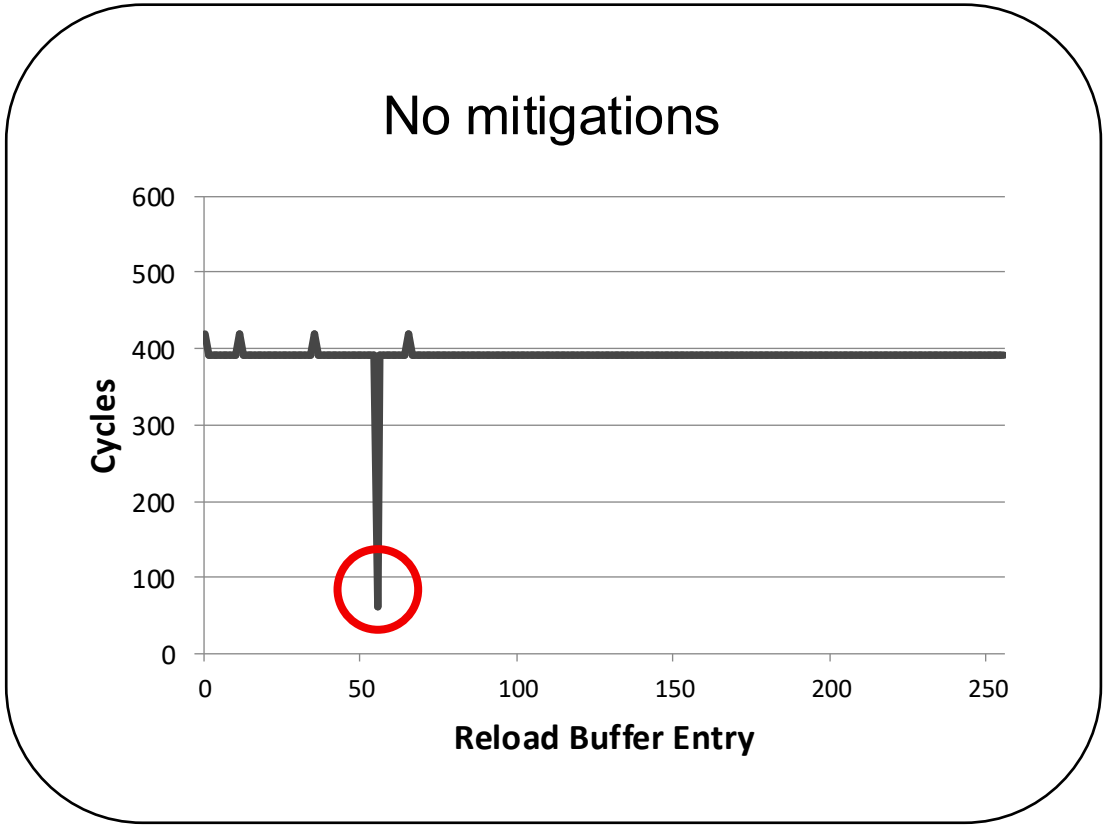
- Insert **Hide** before indirect branch
- Insert **Restore** at all branch targets

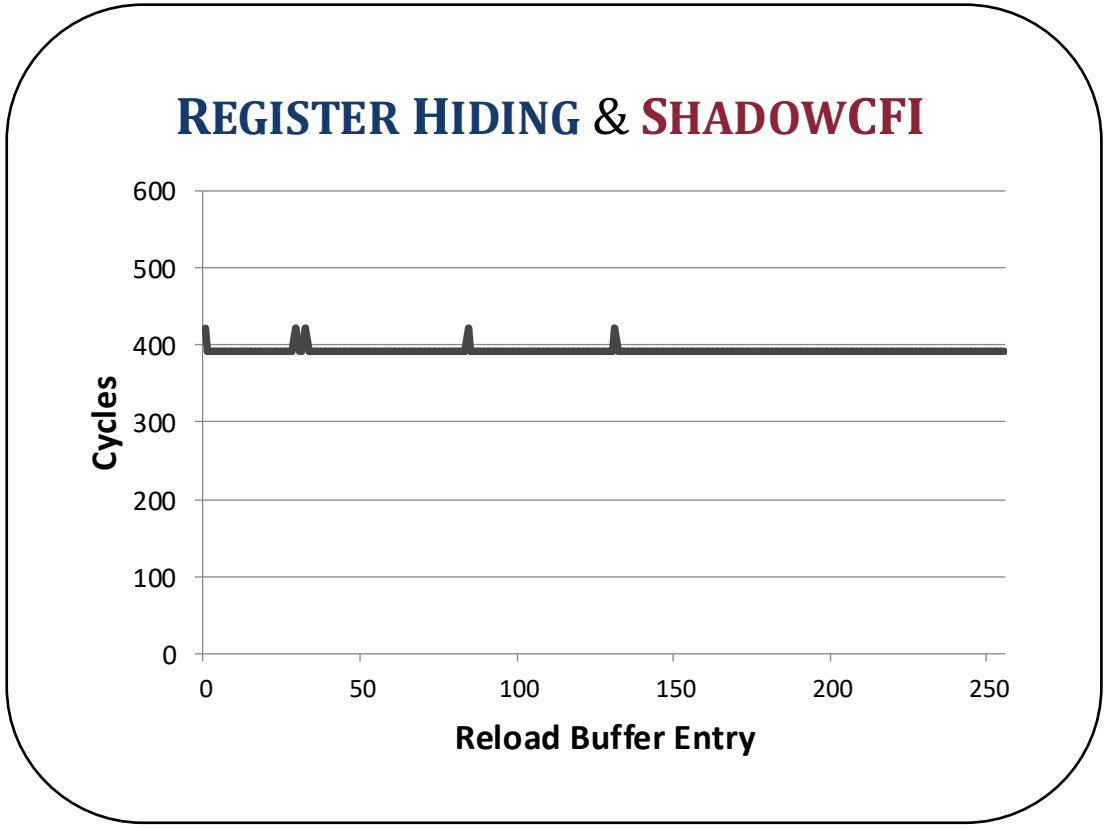
Linux 6.8.0



- **Preserve** user **SSE state**
- **Manual patching** for BPF, KVM etc.

Evaluation: Security on AMD Zen 4





Spectre attack is **successfully blocked** by REGISTER HIDING & SHADOWCFI

Evaluation: Performance on AMD Zen 4

Default mitigations

SafeRET

Re-trains predictor to mispredict to benign location

+

AutoIBRS

Disables speculative execution of indirect jumps and calls in the kernel

Evaluation: Performance on AMD Zen 4

Default mitigations

SafeRET

Re-trains predictor to mispredict to benign location

+

AutoIBRS

Disables speculative execution of indirect jumps and calls in the kernel

Mitigation	Hardware assumptions
SafeRET + AutoIBRS	High
Our paper	Low

Evaluation: Performance on AMD Zen 4

Default mitigations

SafeRET

Re-trains predictor to mispredict to benign location

+

AutoIBRS

Disables speculative execution of indirect jumps and calls in the kernel

Mitigation	Hardware assumptions	LEBench	Server workloads
SafeRET + AutoIBRS	High	114.1%	33.4%
Our paper	Low	75.9%	25.8%

REGISTER HIDING & **SHADOWCFI** are **more general and performant** than mitigations deployed today

Conclusion

- Previously overlooked Spectre v2 ingredient: **secret reachability**
- We propose **REGISTER HIDING** and **SHADOWCFI** to prevent transient access to the register state established prior misprediction
- We build a fully functional patch for Linux kernel 6.8.0
- **REGISTER HIDING & SHADOWCFI** perform better than default mitigations on AMD Zen 4

Thank you! Questions?


danieltr@mit.edu


https://github.com/MATCHA-MIT/
register-hiding-and-shadowcfi