Remote Memory-Deduplication Attacks

Martin Schwarzl, Erik Kraft, Moritz Lipp, Daniel Gruss

Graz University of Technology

Motivation

- More and more services hosted in the cloud

Motivation

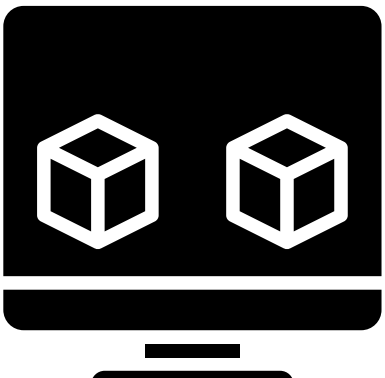
- More and more services hosted in the cloud
- Providers try to isolate tenants

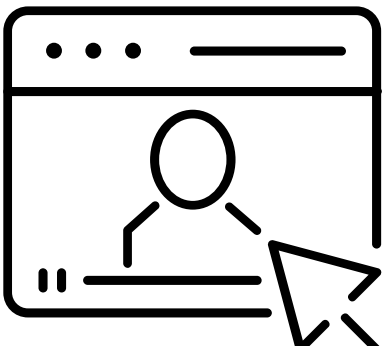
Motivation

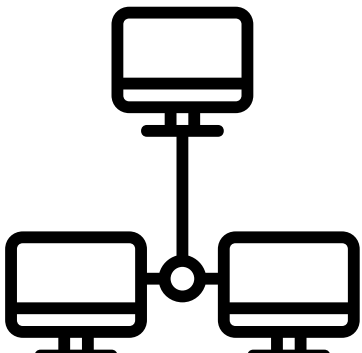
- More and more services hosted in the cloud
- Providers try to isolate tenants
- Need to consider side-channel attacks in both soft- and hardware

Motivation

- More and more services hosted in the cloud
- Providers try to isolate tenants
- Need to consider side-channel attacks in both soft- and hardware
- Network throughput is increasing







Motivation

- Memory deduplication got re-enabled on Windows and Linux

Motivation

- Memory deduplication got re-enabled on Windows and Linux
- Is used in virtual machines in the cloud

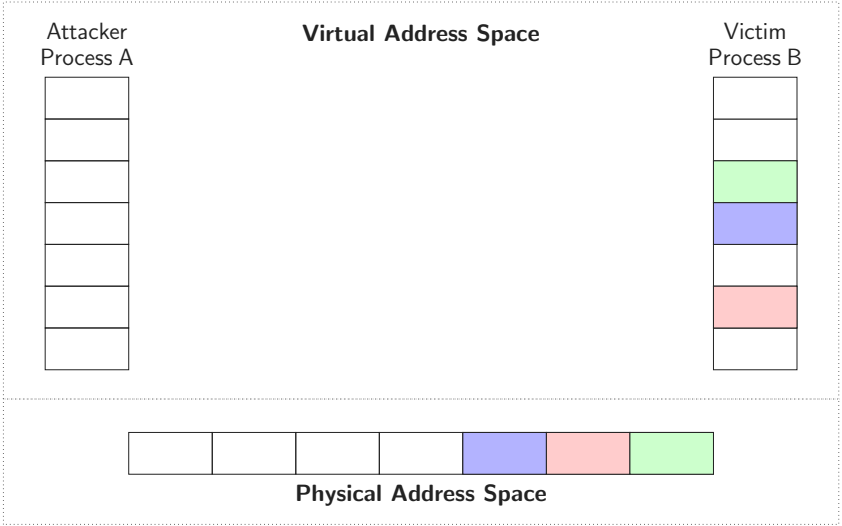
Motivation

- Memory deduplication got re-enabled on Windows and Linux
- Is used in virtual machines in the cloud
- Current active mitigations try to prevent cross-security-domain attacks

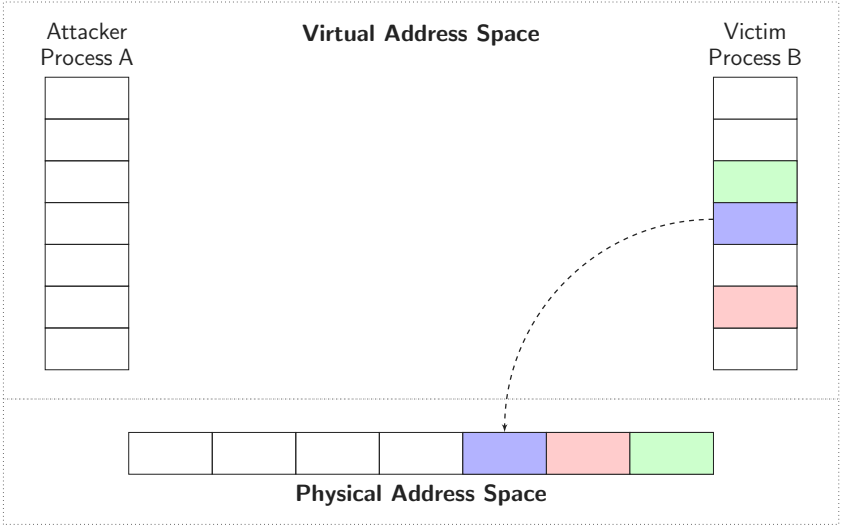
Motivation

- Memory deduplication got re-enabled on Windows and Linux
- Is used in virtual machines in the cloud
- Current active mitigations try to prevent cross-security-domain attacks
- Can memory deduplication attacks be performed on the same security domain across the internet?

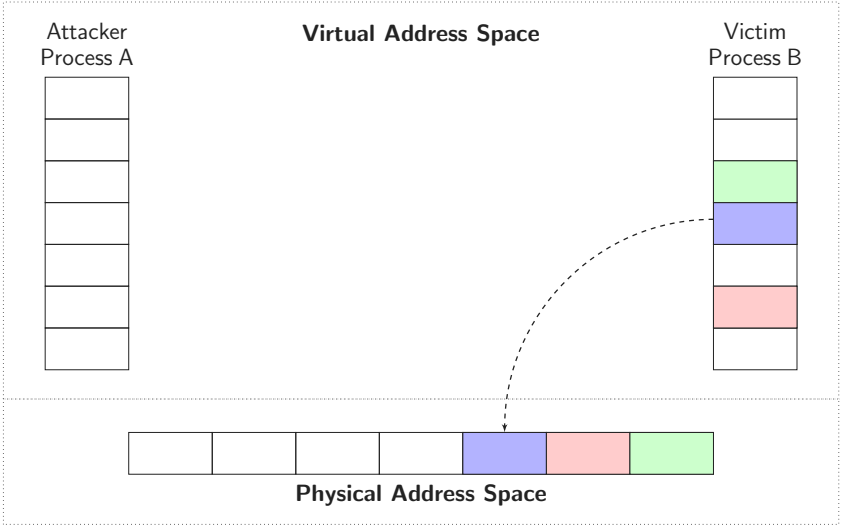
Memory Deduplication



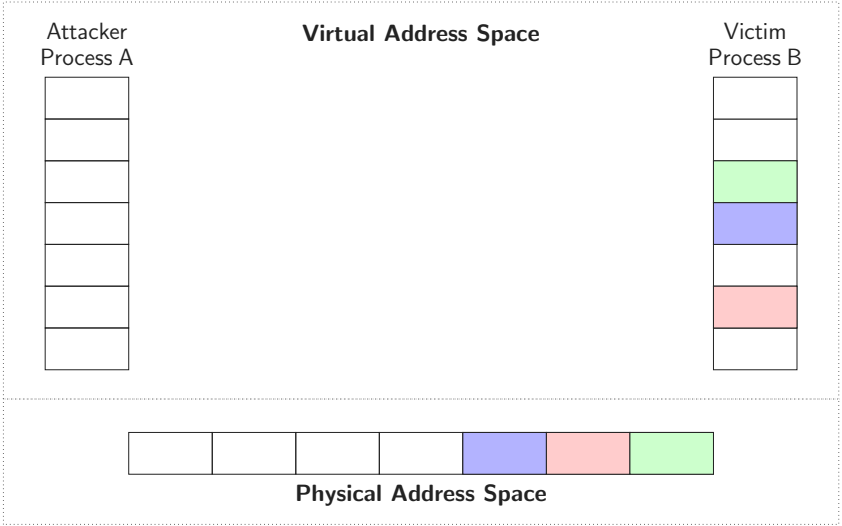
Memory Deduplication



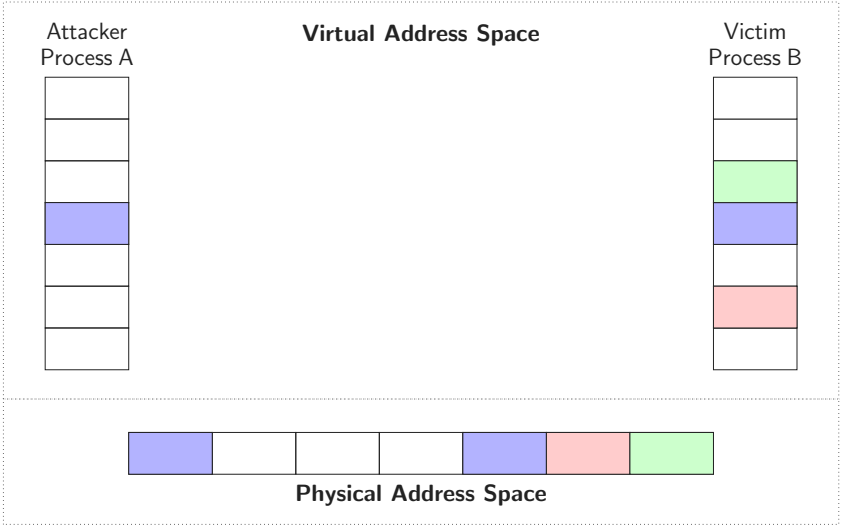
Memory Deduplication



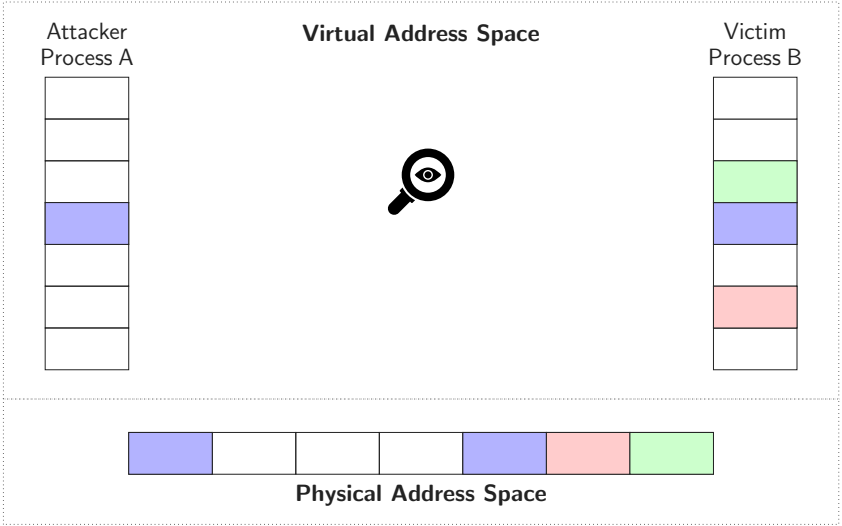
Memory Deduplication



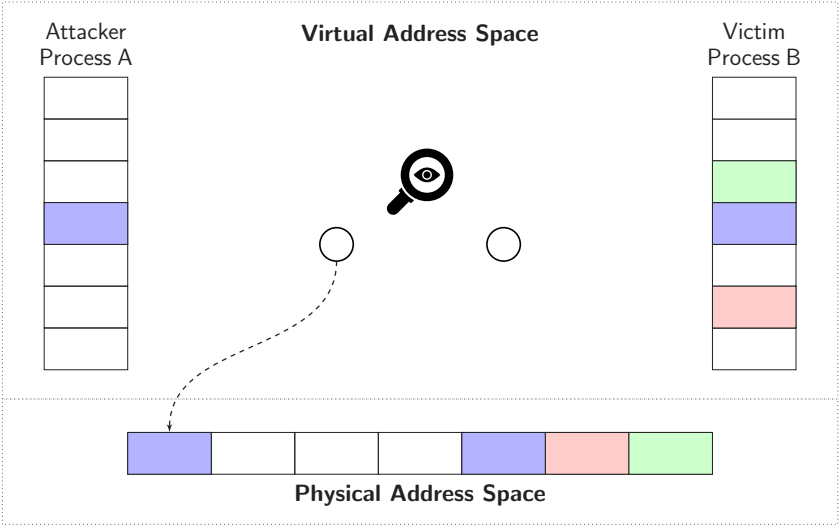
Memory Deduplication



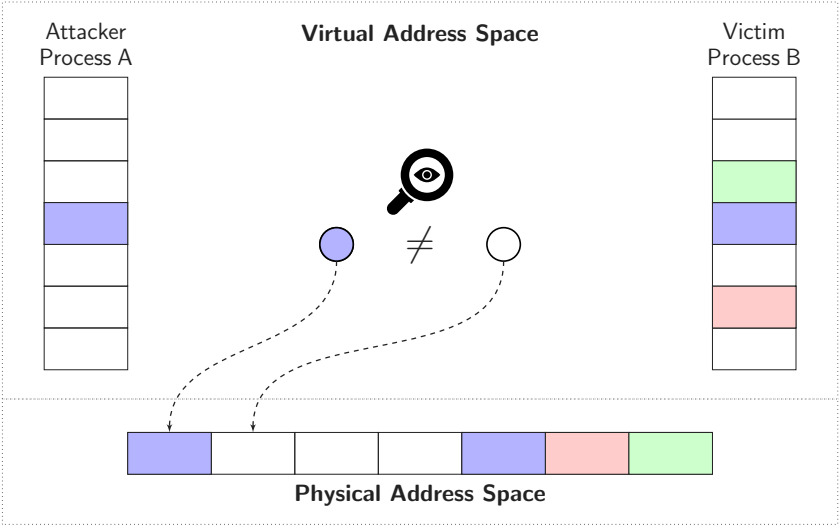
Memory Deduplication



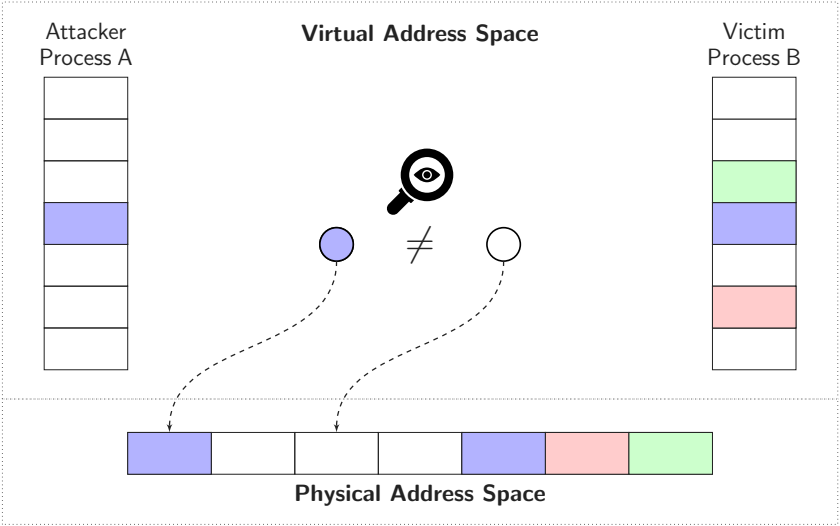
Memory Deduplication



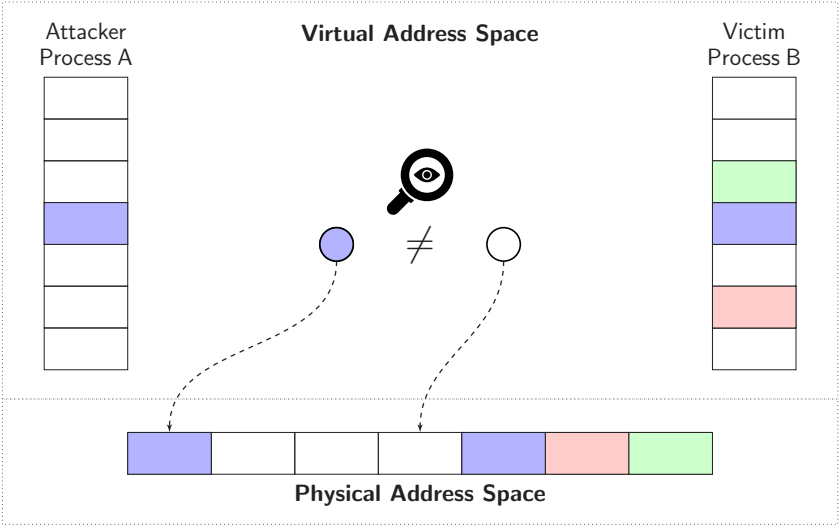
Memory Deduplication



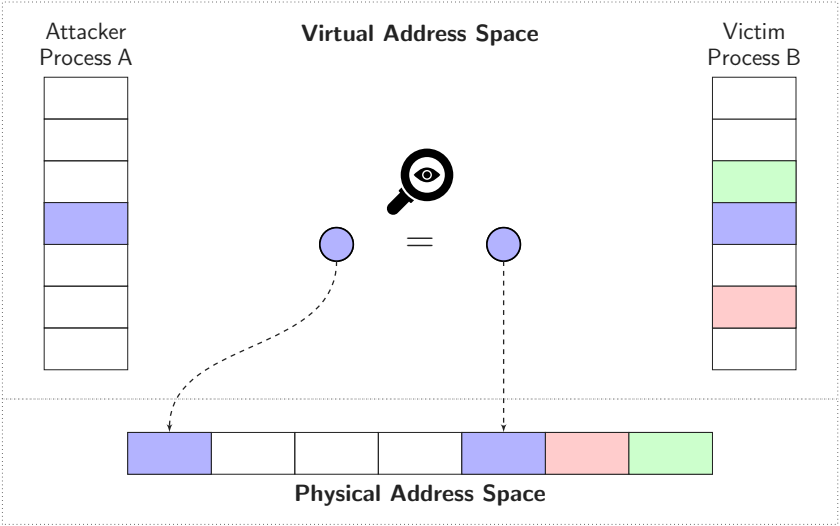
Memory Deduplication



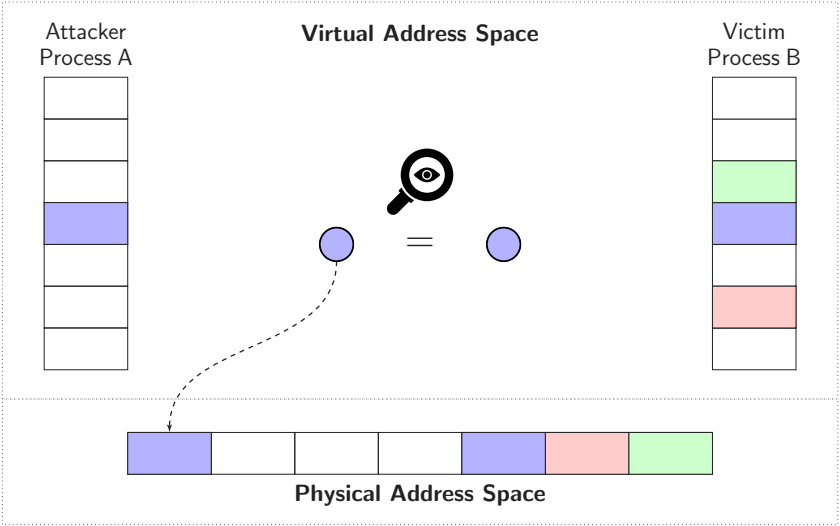
Memory Deduplication



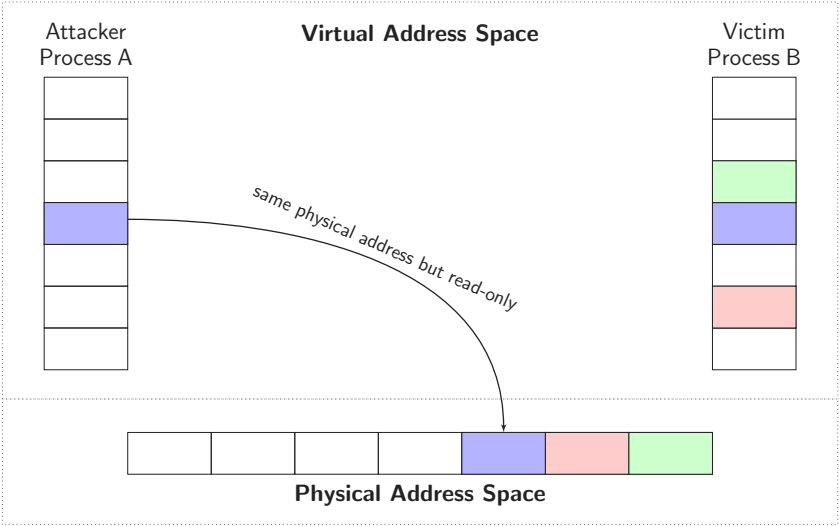
Memory Deduplication



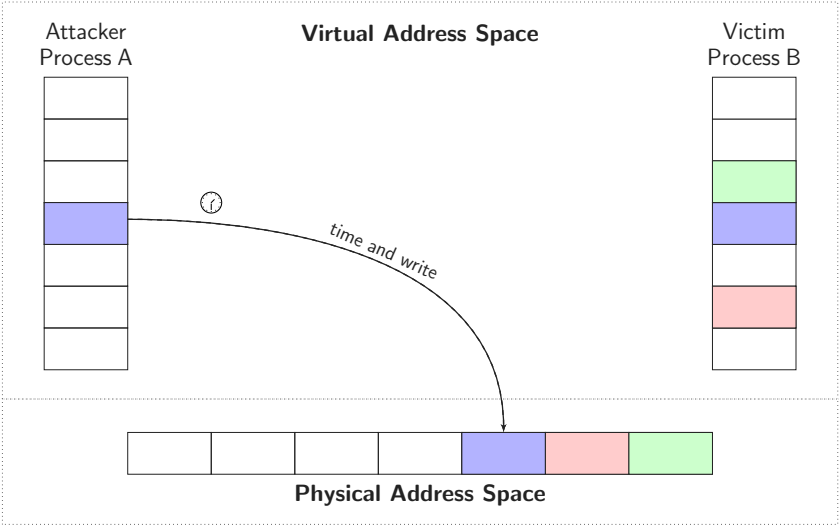
Memory Deduplication



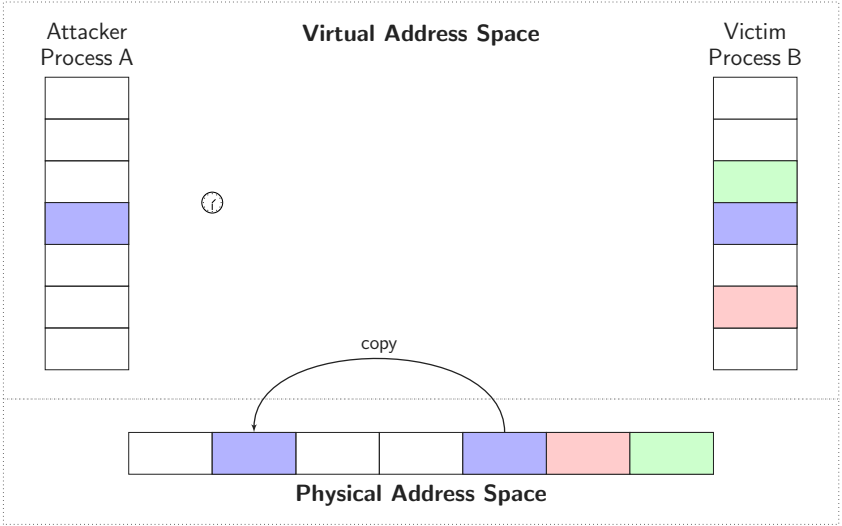
Memory Deduplication



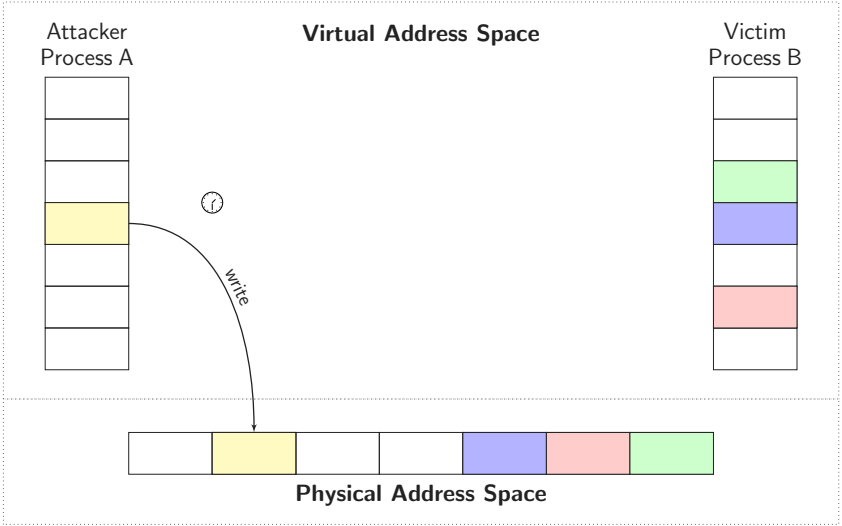
Memory Deduplication



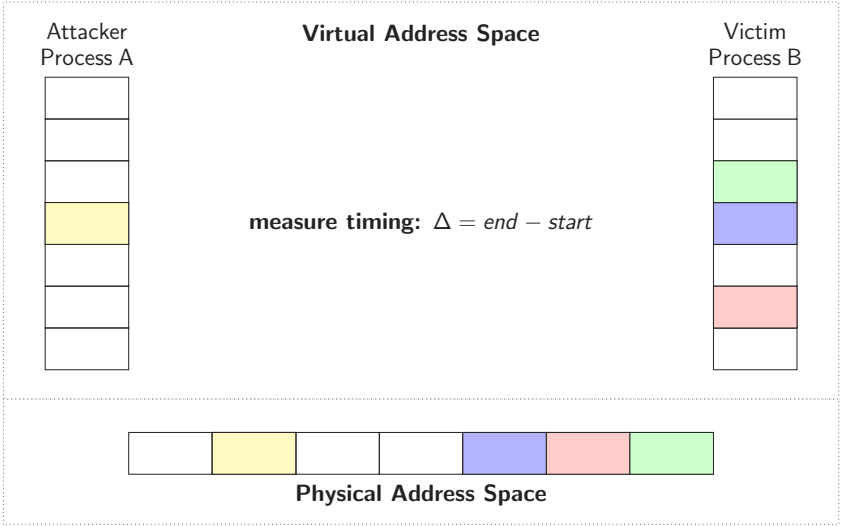
Memory Deduplication



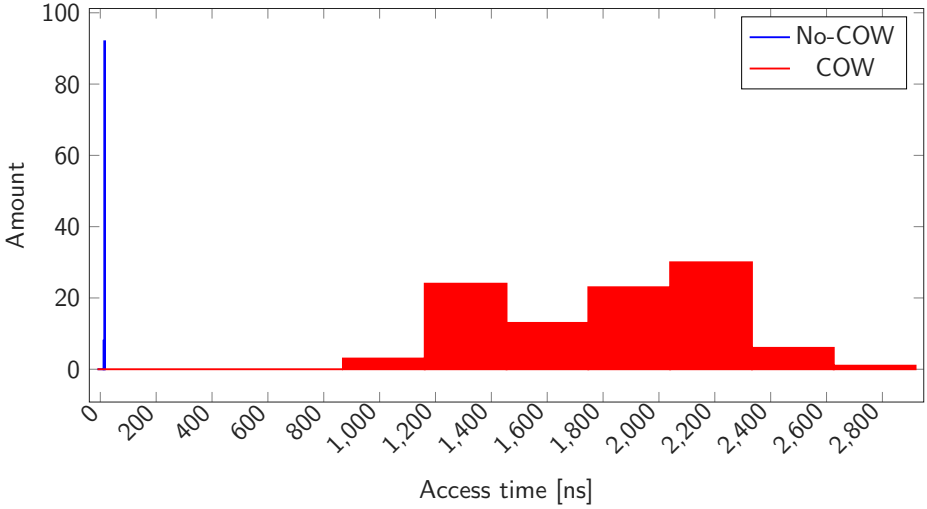
Memory Deduplication



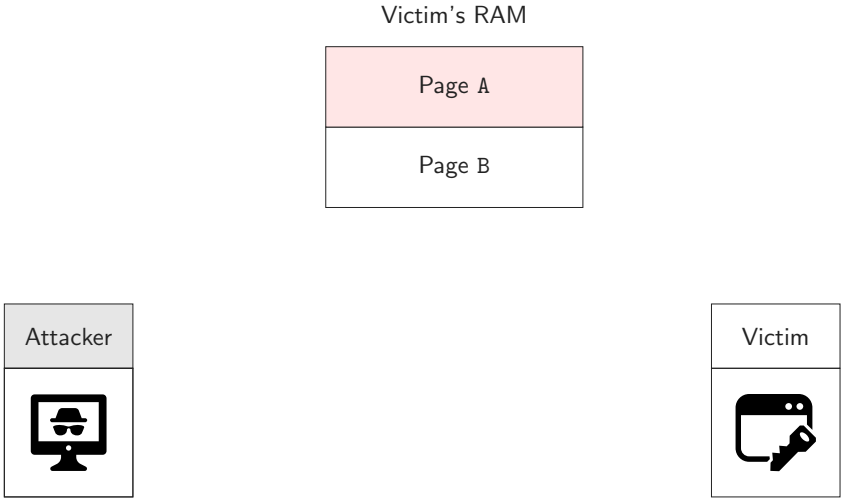
Memory Deduplication



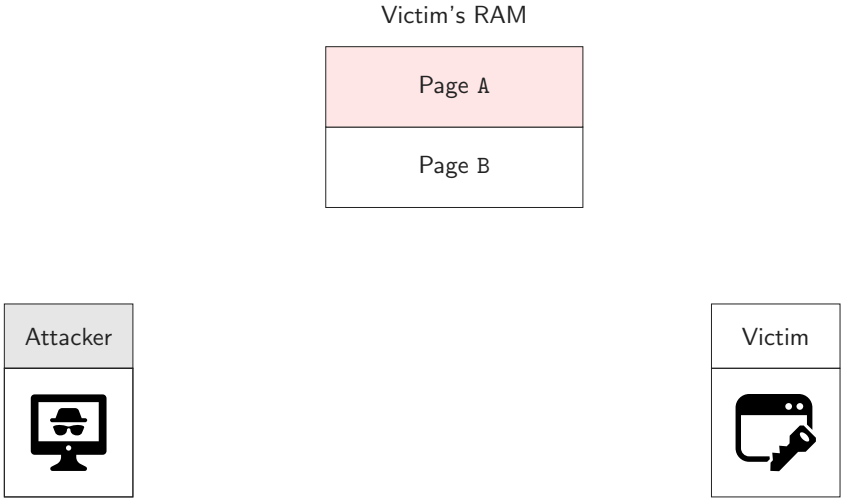
Timing Difference of COW-PF vs. Non-COW



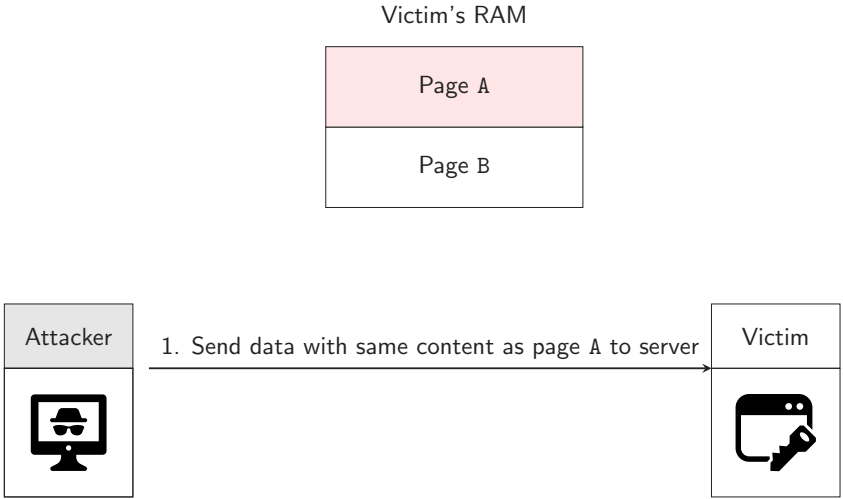
Attack Idea



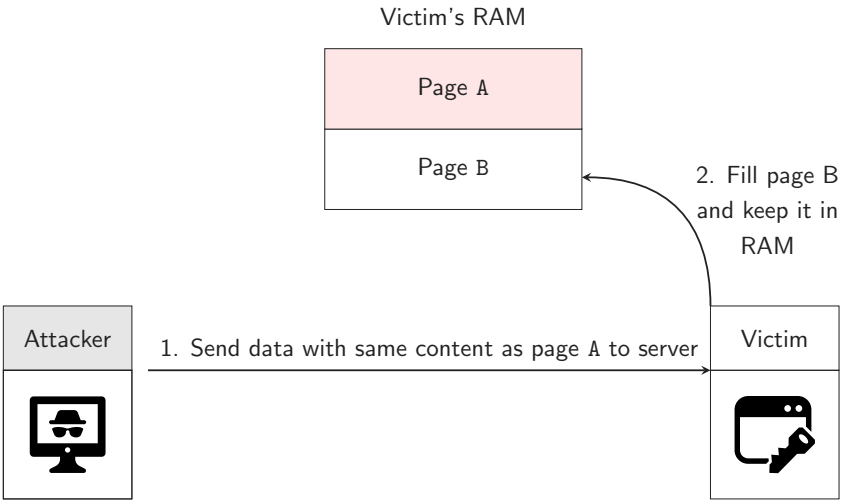
Attack Idea



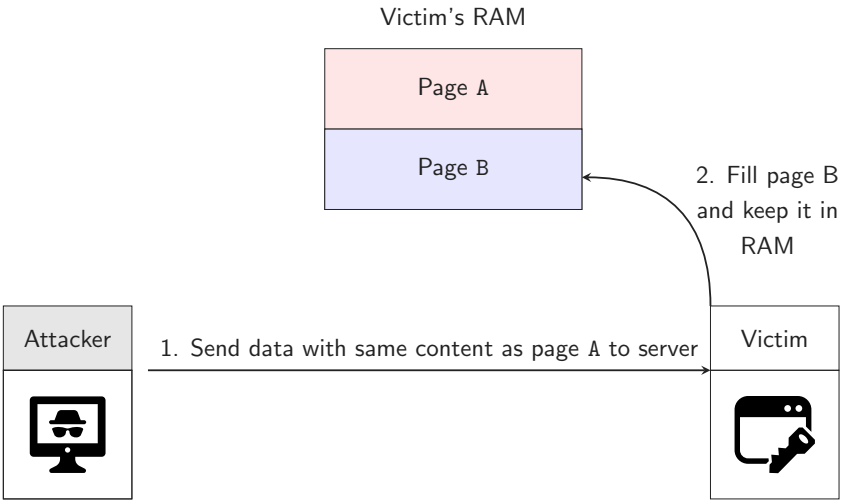
Attack Idea


Victim's RAM
Page A
Page B
Attacker
1. Send data with same content as page A to server
Victim

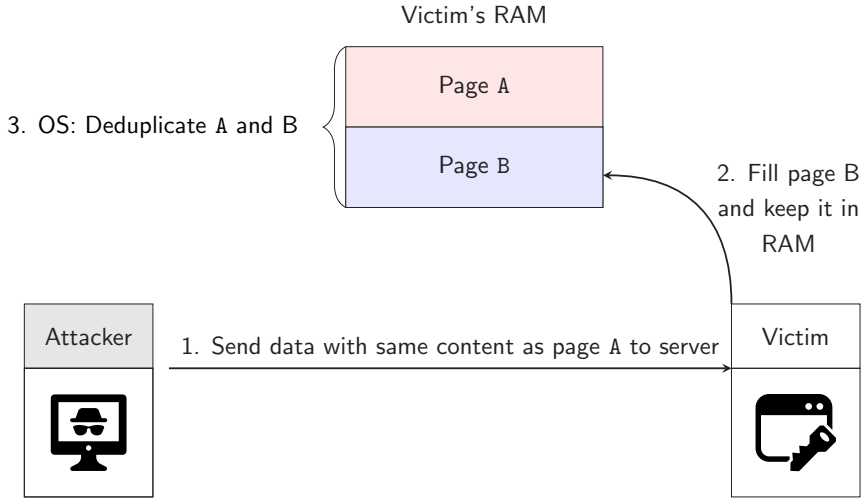
Attack Idea


Victim's RAM
Page A
Page B
2. Fill page B and keep it in RAM
Attacker
1. Send data with same content as page A to server
Victim

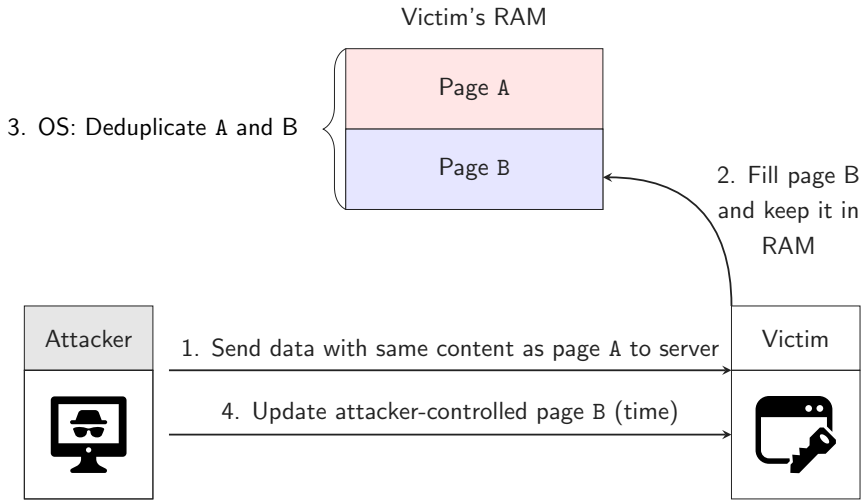
Attack Idea


Victim's RAM
Page A
Page B
2. Fill page B and keep it in RAM
Attacker
1. Send data with same content as page A to server
Victim

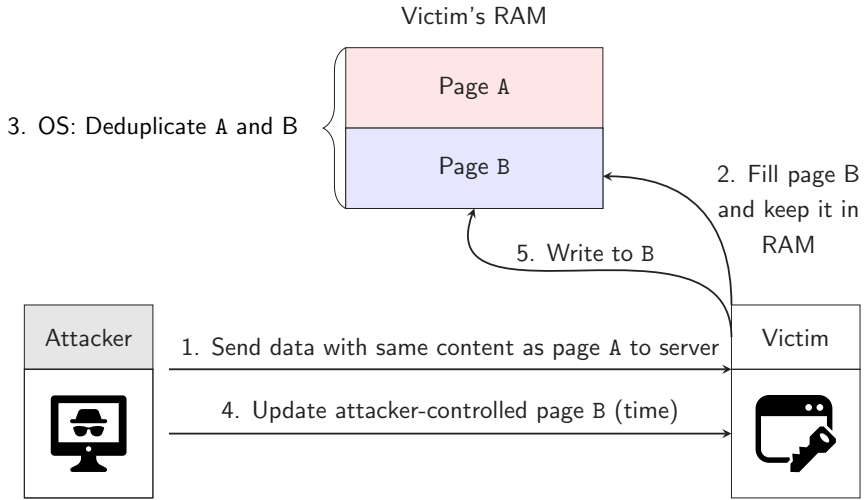
Attack Idea


Victim's RAM
Page A
Page B
3. OS: Deduplicate A and B
2. Fill page B and keep it in RAM
Attacker
1. Send data with same content as page A to server
Victim

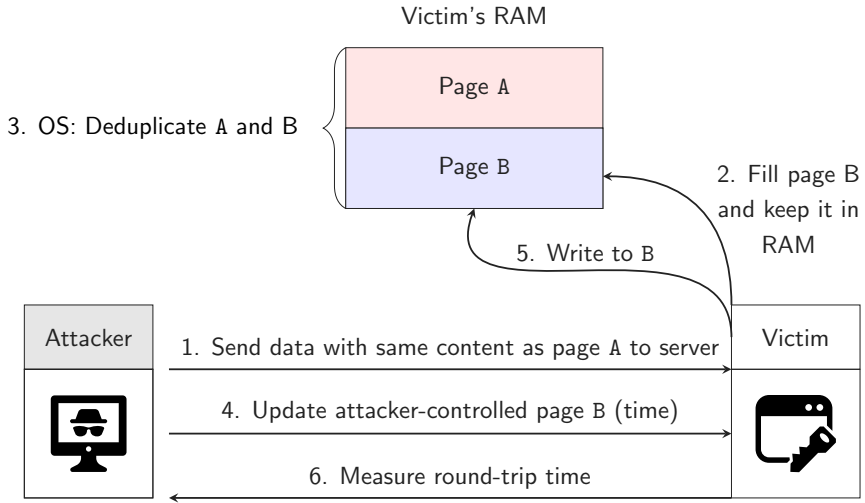
Attack Idea


Victim's RAM
Page A
Page B
3. OS: Deduplicate A and B
2. Fill page B and keep it in RAM
Attacker
Victim
1. Send data with same content as page A to server
4. Update attacker-controlled page B (time)

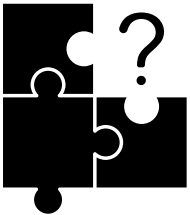
Attack Idea


Victim's RAM
Page A
Page B
3. OS: Deduplicate A and B
2. Fill page B and keep it in RAM
5. Write to B
Attacker
Victim
1. Send data with same content as page A to server
4. Update attacker-controlled page B (time)

Attack Idea

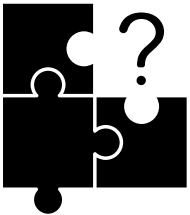

Victim's RAM
Page A
Page B
3. OS: Deduplicate A and B
2. Fill page B and keep it in RAM
5. Write to B
Attacker
Victim
1. Send data with same content as page A to server
4. Update attacker-controlled page B (time)
6. Measure round-trip time

Attack Setup



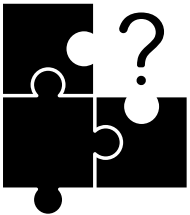
- Remote Server 14 hops → high-latency

Attack Setup



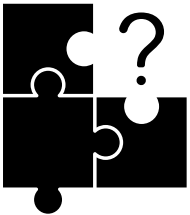
- Remote Server 14 hops → high-latency
- KVM with Ubuntu VM

Attack Setup



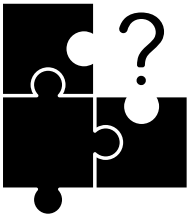
- Remote Server 14 hops → high-latency
- KVM with Ubuntu VM
- Nginx with PHP, Memcached and MySQL installed

Attack Setup



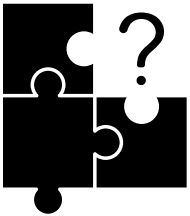
- Remote Server 14 hops → high-latency
- KVM with Ubuntu VM
- Nginx with PHP, Memcached and MySQL installed
- Use pyshark to capture web requests

Remote Covert Channel



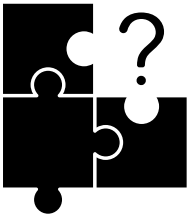
- Use amplification across the internet

Remote Covert Channel



- Use amplification across the internet
- Transmit multiple bits at once

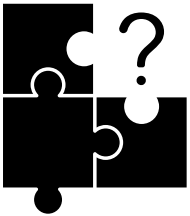
Remote Covert Channel



- Use amplification across the internet
- Transmit multiple bits at once
- Use asyncio
- Covert channel across internet is 34.41 B/h

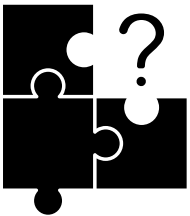
Attacks	Location	Environment	Local	Type	Attack Type	Performance
Suzaki	Co-located	Cross-VM	Yes	Native	Fingerprinting	-
Owens	Co-located	Cross-VM	Yes	Native	Fingerprinting	-
Gruss	Remote	Browser/Cross-VM	Yes	JS	Fingerprinting	-
Barresi	Remote	Cross-VM	Yes	Native	ASLR break	8.7 days
Bosman	Remote	Browser	Yes	JS	Bytewise leakage, ASLR break, Rowhammer	2.75 h
Lindemann	Co-located	Cross-VM	Yes	Native	Fingerprinting	1.8 h
Kim	Co-located	Cross-VM	Yes	Native	KASLR break	12 min
Our work	Remote	**Internet/LAN**	**No**	**None**	Bytewise leakage, KASLR break, Fingerprinting	1.5 B/h (LAN) / 4 min / 166.51 s

Challenges



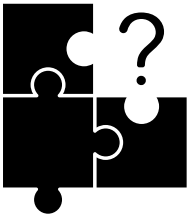
- C1: Remotely amplify latencies for non-repeatable events.

Challenges



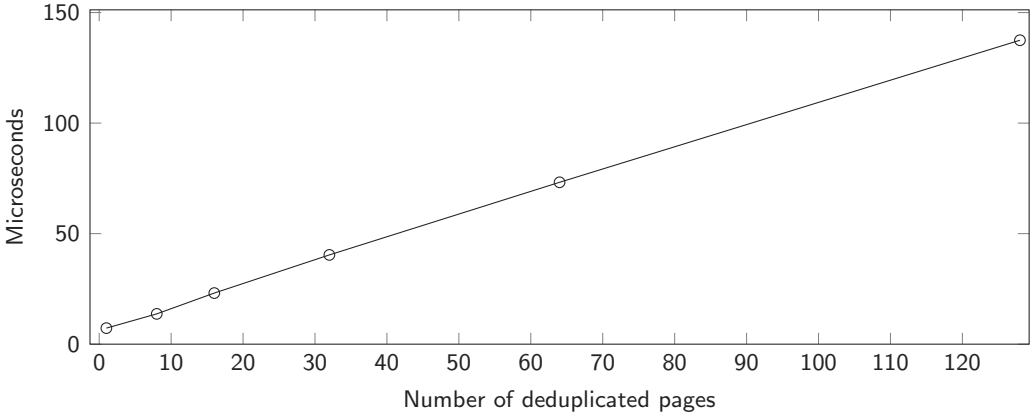
- C1: Remotely amplify latencies for non-repeatable events.
- C2: Trigger and observe COW-pagefaults in a victim domain that shares no memory with any attacker domain.

Challenges

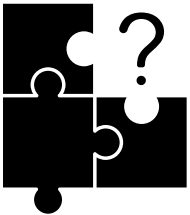


- C1: Remotely amplify latencies for non-repeatable events.
- C2: Trigger and observe COW-pagefaults in a victim domain that shares no memory with any attacker domain.
- C3: Find remote request paths that do not only keep attacker-controlled data in memory but also provide the attacker with control over alignment and in-memory representation.

C1: Amplification

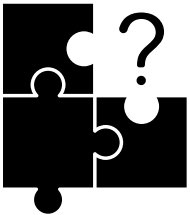


C2: Trigger-COW pagefaults without shared memory



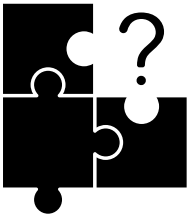
- A web application provides a file-upload

C2: Trigger-COW pagefaults without shared memory



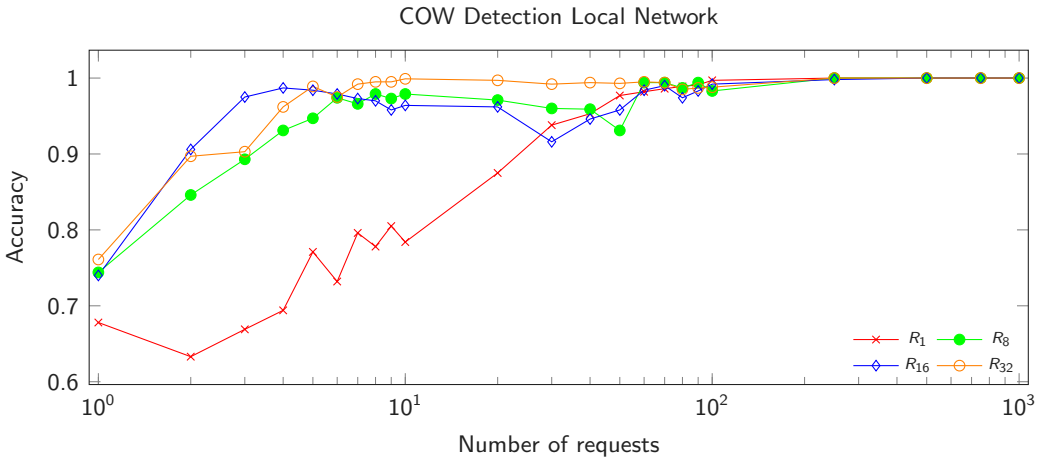
- A web application provides a file-upload
- Data is cached in RAM e.g., Memcached

C2: Trigger-COW pagefaults without shared memory

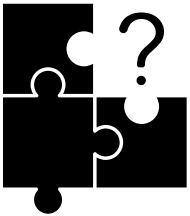


- A web application provides a file-upload
- Data is cached in RAM e.g., Memcached
- The attacker can update/overwrite the uploaded data → trigger pagefaults

C2: Trigger-COW pagefaults without shared memory

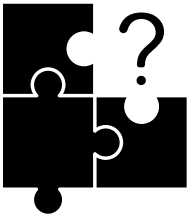


C2: Fingerprint a system library



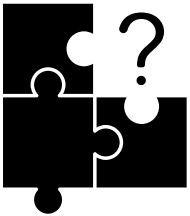
- Fingerprint a system by uploading memory

C2: Fingerprint a system library



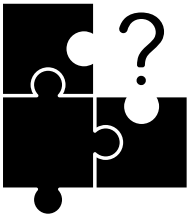
- Fingerprint a system by uploading memory
- Use Memcached to store and replace

C2: Fingerprint a system library



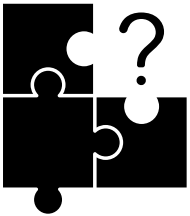
- Fingerprint a system by uploading memory
- Use Memcached to store and replace
- Page-alignment unknown therefore we guess all possible offsets

C2: Fingerprint a system library



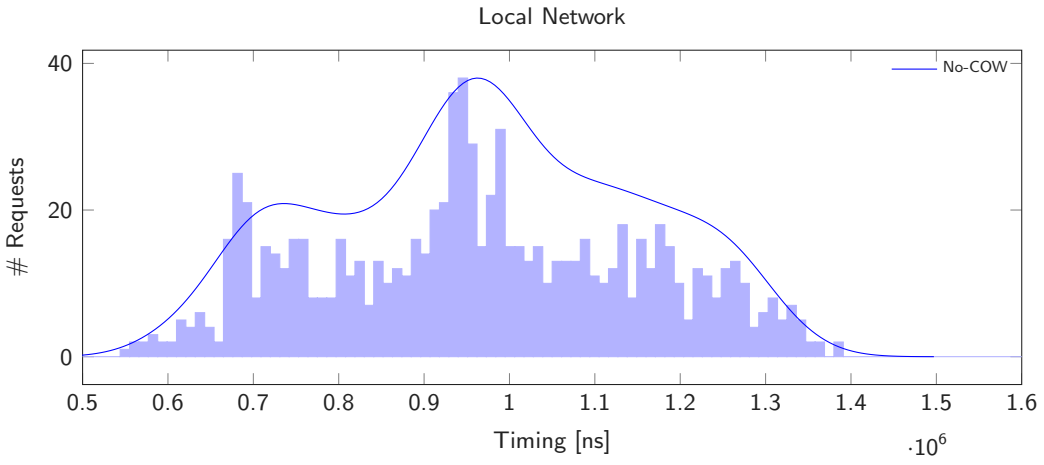
- Fingerprint a system by uploading memory
- Use Memcached to store and replace
- Page-alignment unknown therefore we guess all possible offsets
- Race with other users via re-allocation on free-list

C2: Fingerprint a system library

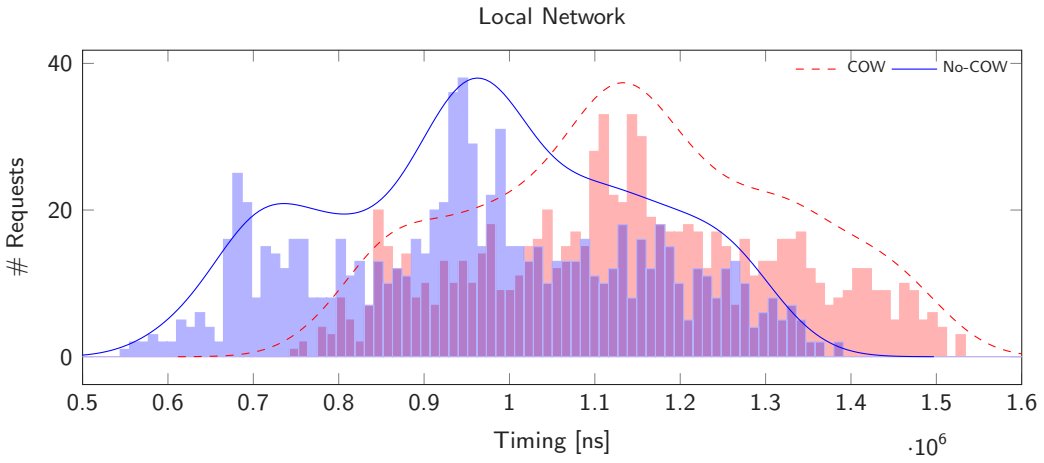


- Fingerprint a system by uploading memory
- Use Memcached to store and replace
- Page-alignment unknown therefore we guess all possible offsets
- Race with other users via re-allocation on free-list
- If re-assigned overwrite page and trigger COW-pagefault

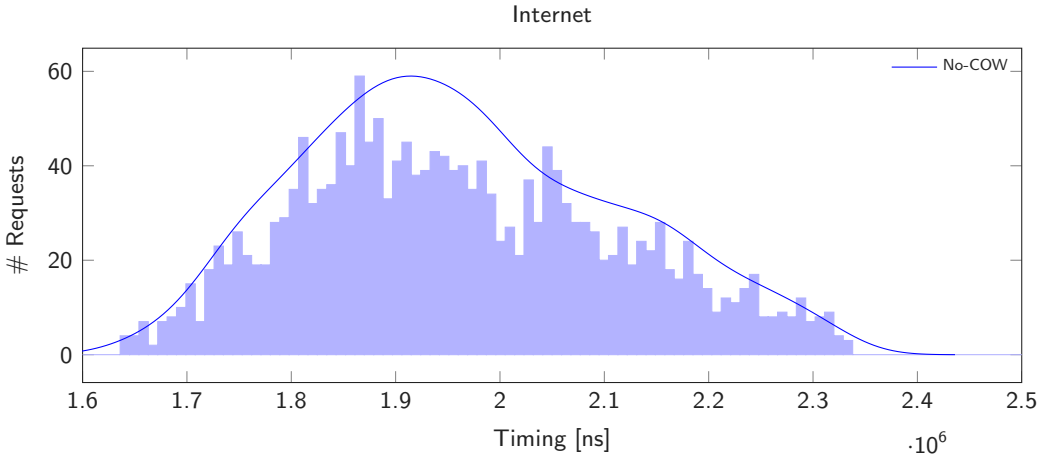
C2: Fingerprinting (LAN)



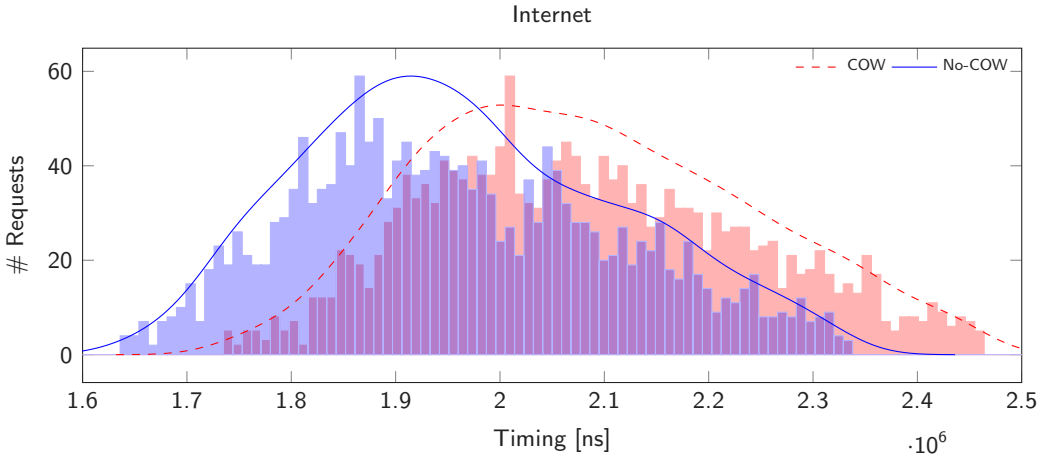
C2: Fingerprinting (LAN)



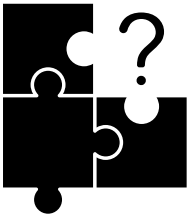
C2: Fingerprinting (Internet)



C2: Fingerprinting (Internet)

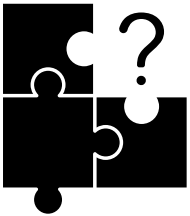


C2: Break KASLR



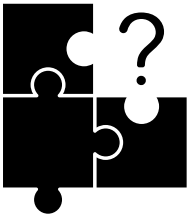
- Break KASLR in remote VMs

C2: Break KASLR



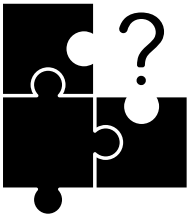
- Break KASLR in remote VMs
- Sample low-entropy pages offline pointing to kernel text

C2: Break KASLR



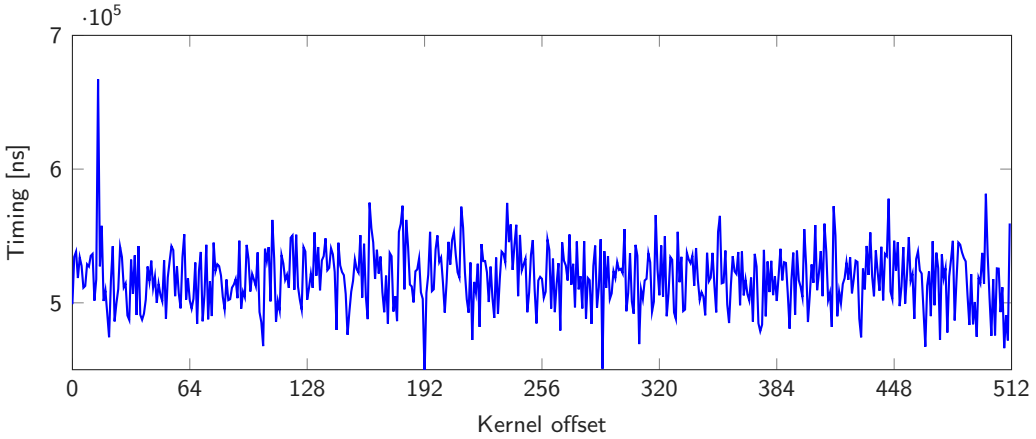
- Break KASLR in remote VMs
- Sample low-entropy pages offline pointing to kernel text
- Try all 512 different offsets

C2: Break KASLR

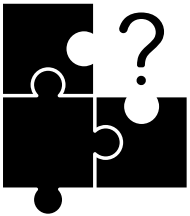


- Break KASLR in remote VMs
- Sample low-entropy pages offline pointing to kernel text
- Try all 512 different offsets
- Attacker uploads blob and triggers pagefaults

Break KASLR

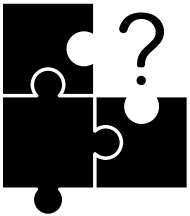

·10^5
7
6
5
Timing [ns]
0
64
128
192
256
320
384
448
512
Kernel offset

C3: Control over alignment and in-memory representation



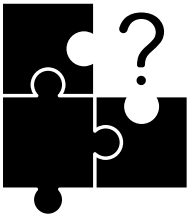
- InnoDB is a memory cache for DBMS (e.g., MySQL/MariaDB)

C3: Control over alignment and in-memory representation



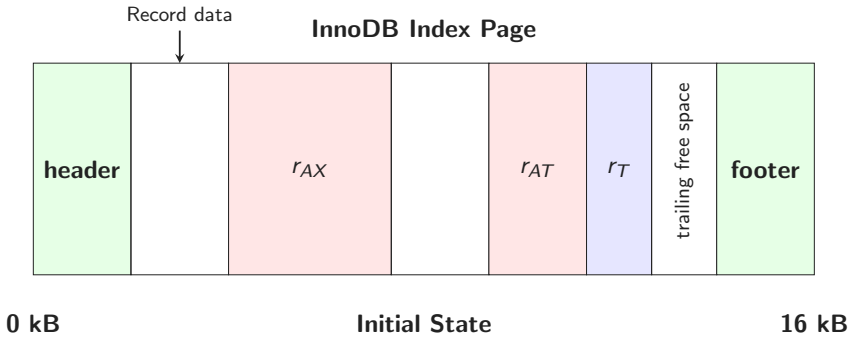
- InnoDB is a memory cache for DBMS (e.g., MySQL/MariaDB)
- Reorganization optimization in index page enables bytewise leakage

C3: Control over alignment and in-memory representation

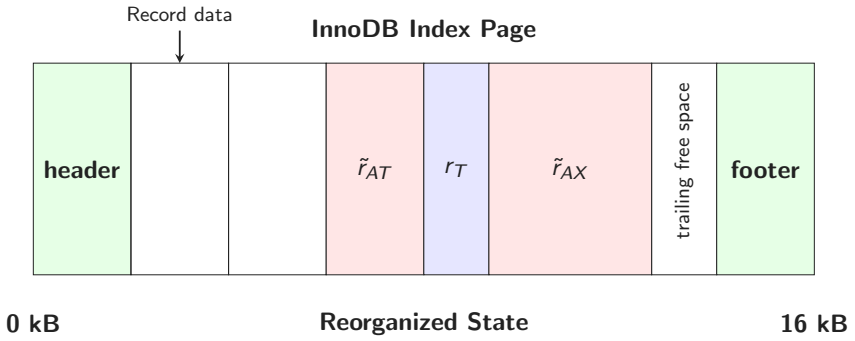


- InnoDB is a memory cache for DBMS (e.g., MySQL/MariaDB)
- Reorganization optimization in index page enables bytewise leakage
- Use Memcached as leakage primitive to leak InnoDB records

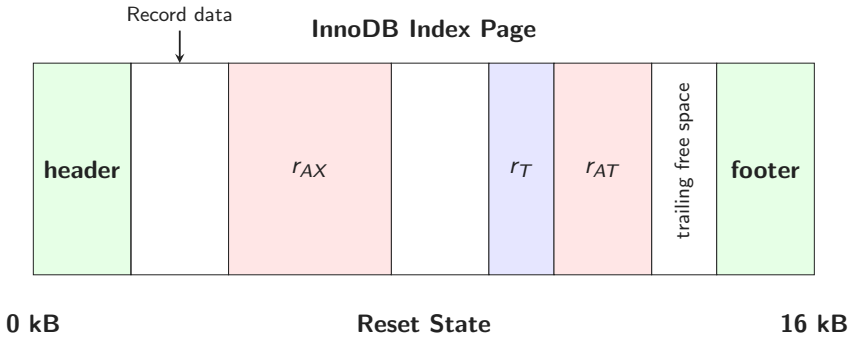
Inno-DB record



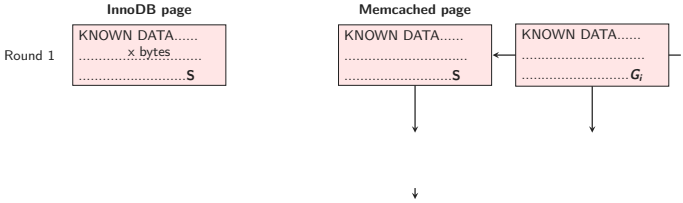
Inno-DB Reorganization



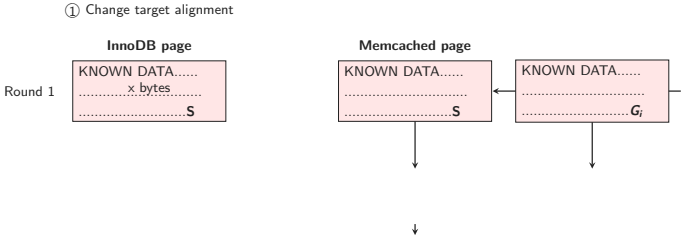
Inno-DB Reset



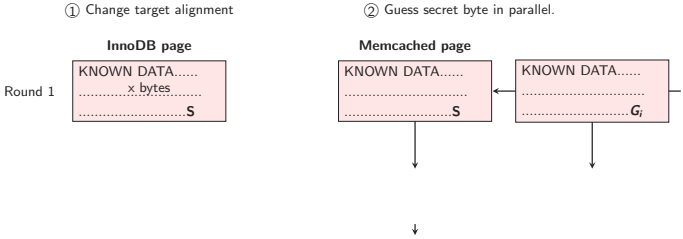
Inno-DB Leaking



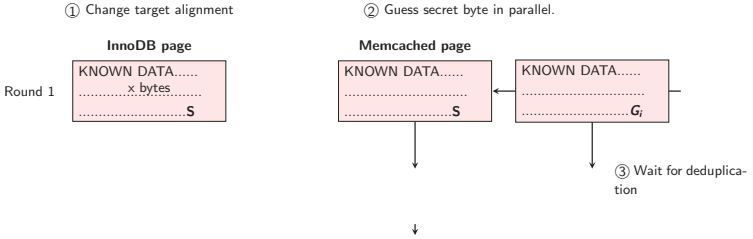
Inno-DB Leaking



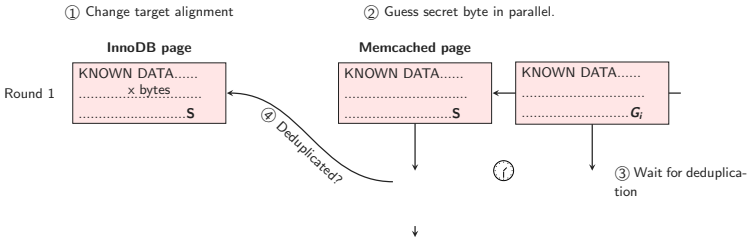
Inno-DB Leaking



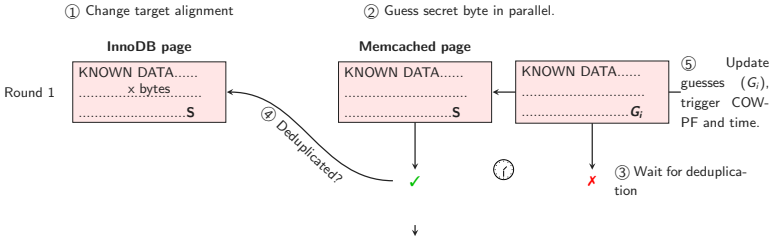
Inno-DB Leaking



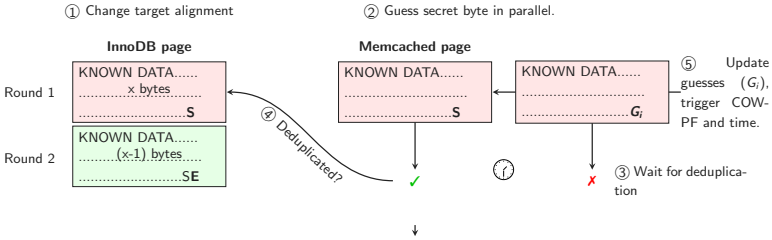
Inno-DB Leaking



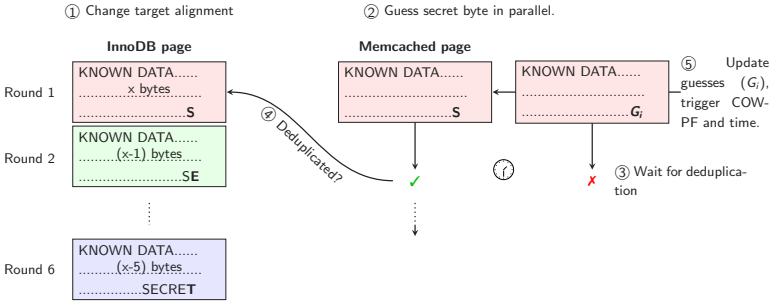
Inno-DB Leaking



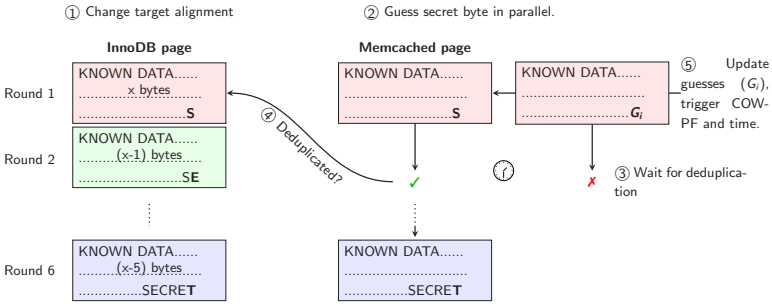
Inno-DB Leaking



Inno-DB Leaking



Inno-DB Leaking



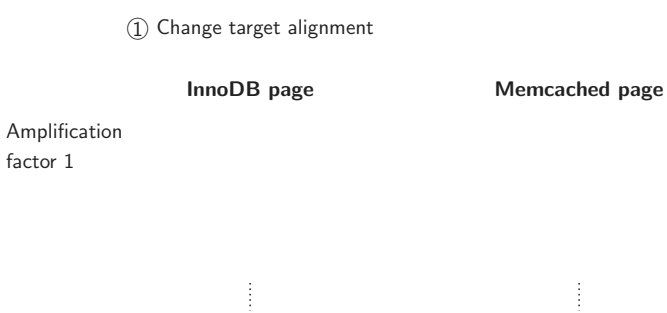
Amplification via Memcached

InnoDB page

Memcached page

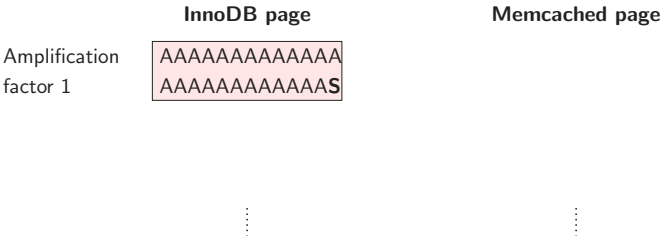
Amplification via Memcached

① Change target alignment



Amplification via Memcached

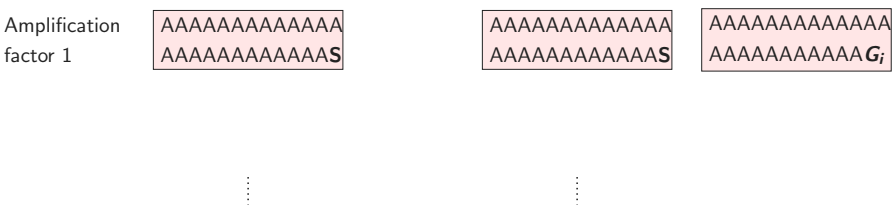
① Change target alignment



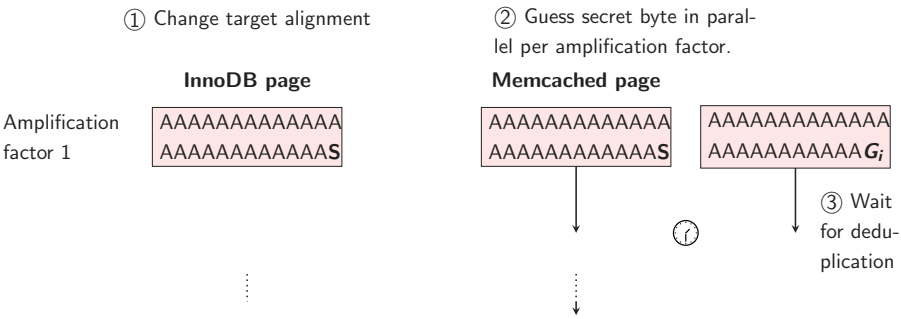
Amplification via Memcached

① Change target alignment

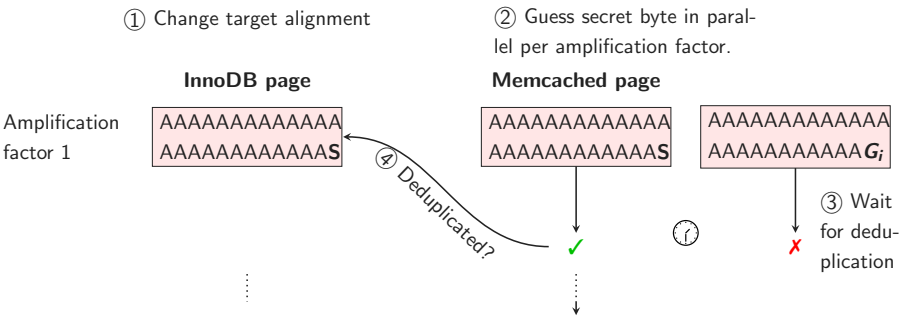
② Guess secret byte in parallel per amplification factor.



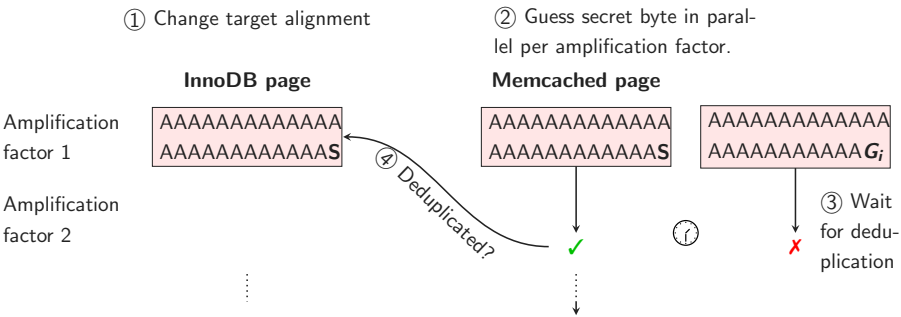
Amplification via Memcached



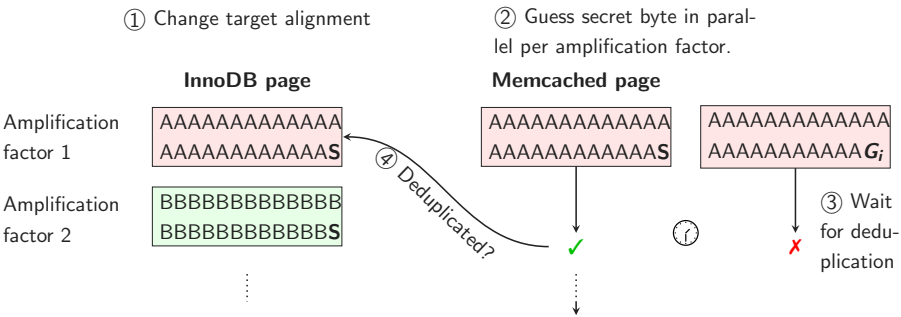
Amplification via Memcached



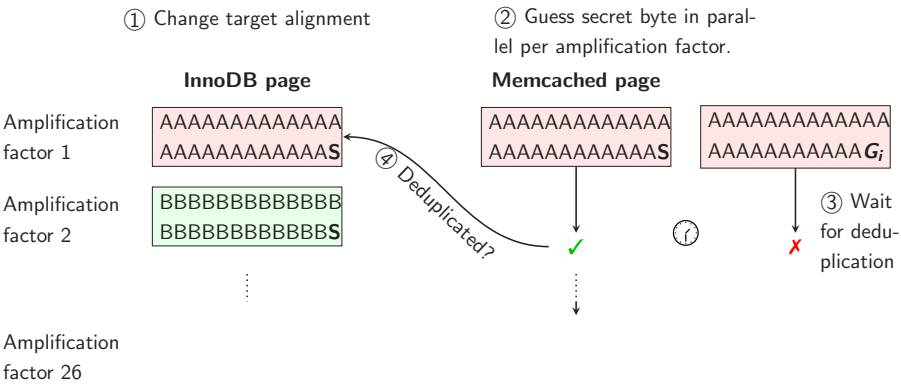
Amplification via Memcached



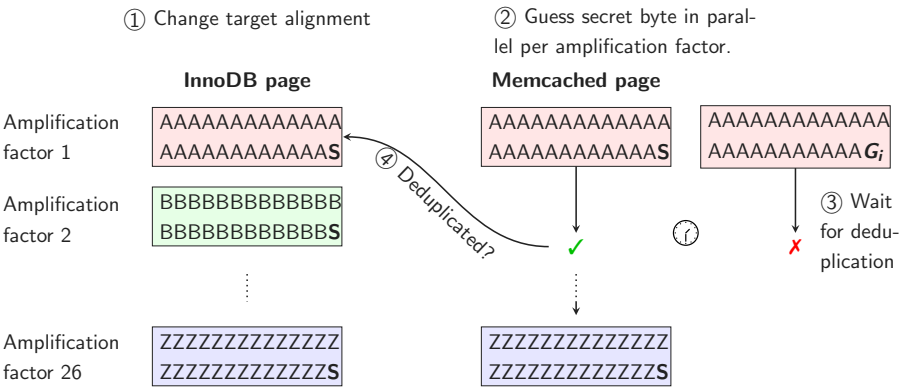
Amplification via Memcached



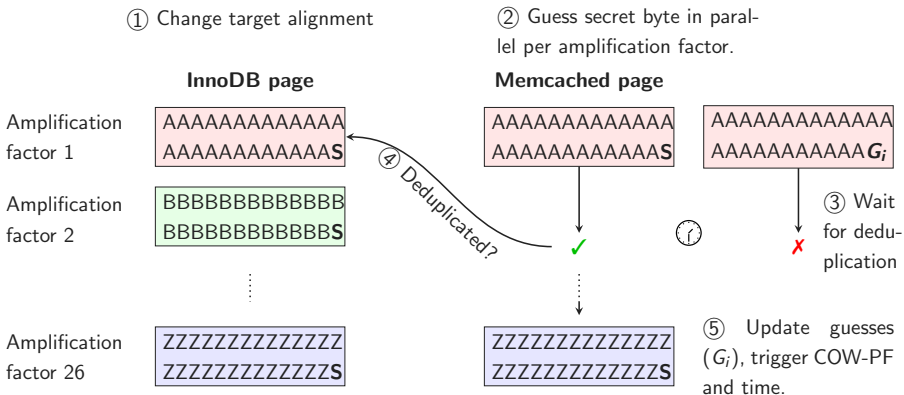
Amplification via Memcached



Amplification via Memcached



Amplification via Memcached



Mitigations

- Disable memory deduplication

Mitigations

- Disable memory deduplication
- Apply same behaviour for every memory write (VUsion)

Mitigations

- Disable memory deduplication
- Apply same behaviour for every memory write (VUsion)
- Only deduplicate zero pages

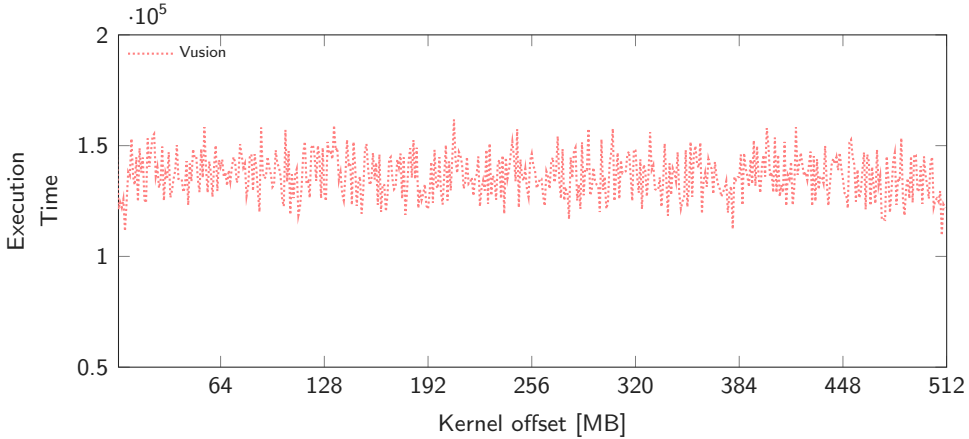
Mitigations

- Disable memory deduplication
- Apply same behaviour for every memory write (VUsion)
- Only deduplicate zero pages
- Detect attack on network layer with packet inspection

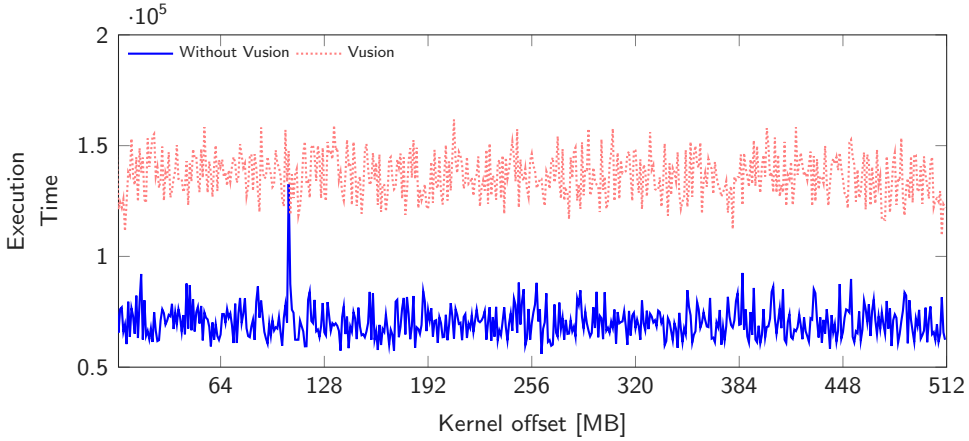
Mitigations

- Disable memory deduplication
- Apply same behaviour for every memory write (VUsion)
- Only deduplicate zero pages
- Detect attack on network layer with packet inspection
- Encode pages with different random salts

VUsion


·10^5
2
1.5
1
0.5
Vusion
Execution Time
64
128
192
256
320
384
448
512
Kernel offset [MB]

VUsion


·10^5
2
1.5
1
0.5
Without Vusion
Vusion
Execution Time
64
128
192
256
320
384
448
512
Kernel offset [MB]

Conclusion

- Remote Attack was assigned CVE-2021-3714

Conclusion

- Remote Attack was assigned CVE-2021-3714
- Remotely fingerprinting of libraries

Conclusion

- Remote Attack was assigned CVE-2021-3714
- Remotely fingerprinting of libraries
- Break KASLR in $\leq$ 4 minutes across the internet

Conclusion

- Remote Attack was assigned CVE-2021-3714
- Remotely fingerprinting of libraries
- Break KASLR in $\leq$ 4 minutes across the internet
- Leak database records via InnoDB reorganization

Conclusion

- Remote Attack was assigned CVE-2021-3714
- Remotely fingerprinting of libraries
- Break KASLR in $\leq$ 4 minutes across the internet
- Leak database records via InnoDB reorganization
- Red Hat developed a probabilistic mitigation as opt-in for Linux kernel

Remote Memory-Deduplication Attacks

Martin Schwarzl, Erik Kraft, Moritz Lipp, Daniel Gruss

Graz University of Technology