CARAS
Arch & Sec

RowHammer: A Summary

Presenter: Haowen Liu

Time: 21st Dec. 2020

Outline

> Risks in Comfort Zone

> DRAM Background
> Disturbance Error Phenomenon -> RowHammer

> Characterization
> Exploitation
> Mitigation

> Conclusion

Risks in Comfort Zone

Risks in Comfort Zone

- Classical Security issues most focus on software-level security.
  - Authority control, process isolation, digital signature, cryptosystems, …
  - Popular tools and methodologies can address most of security problems.

- Most of these techniques fit well with their "all-round" threat model.
  - There is nothing new to be addressed in security now. All that remains is to reuse our threat models to address more and more new-born software vulnerabilities.

Risks in Comfort Zone

- But how about the underlying hardware architecture?
  - All your threat models are based on the assumption that the underlying architecture is secure enough to support these upper-structure security techniques.

- Well, now it’s intuitively secure enough though, with many years’ demonstration in universal implementations. So there's no need to be paranoid.

Risks in Comfort Zone

Risks in Comfort Zone

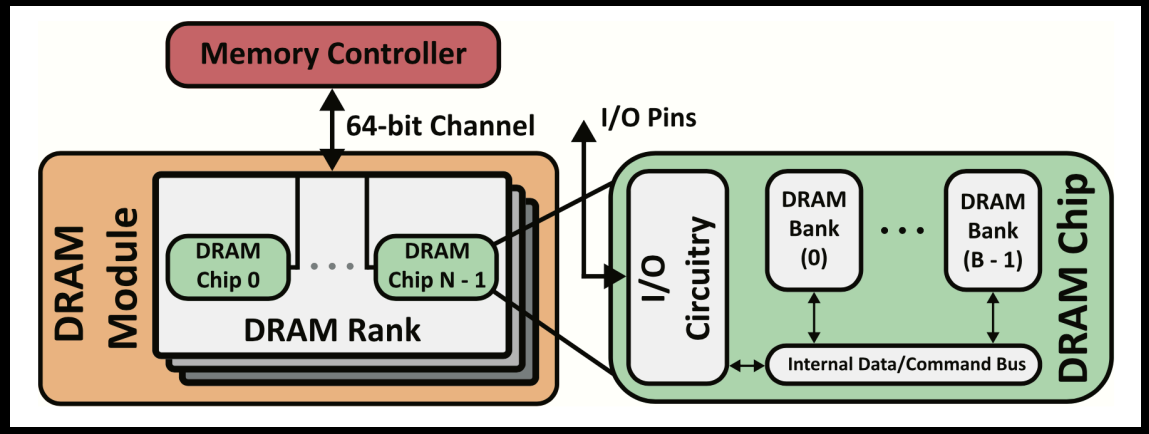
- RowHammer can break through most of existing software-level security mechanisms, for it attacks the basis of all of them: DRAM.
  - Authority control: flip authority control bits to obtain authority.
  - Privilege control: flip instruction bits to escalate privilege.
  - Process isolation and sandbox: corrupt processes in adjacent rows by bit flips.
  - Digital signature and cryptosystems: flip bits in public keys to factorize them.

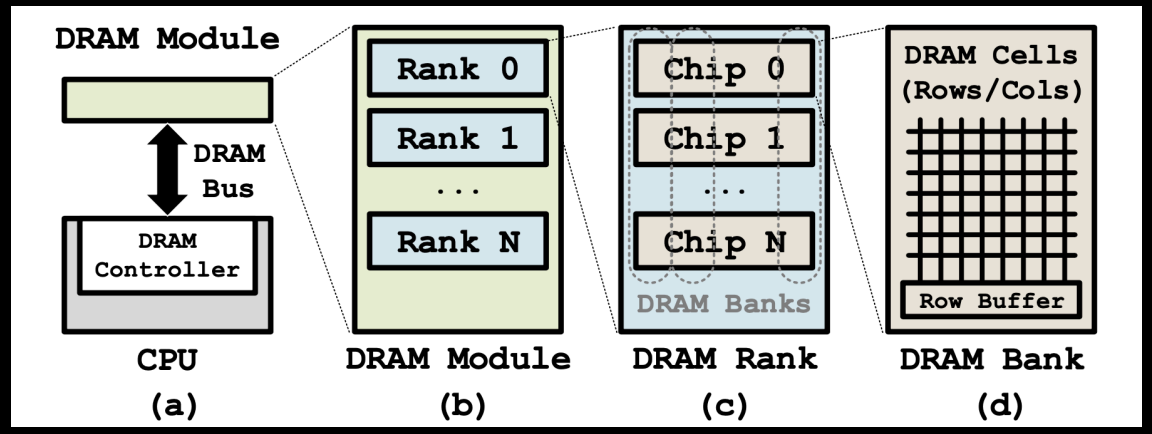
- No comfort zone now, so get up and go to work :-)

DRAM Background

DRAM Background: Organization

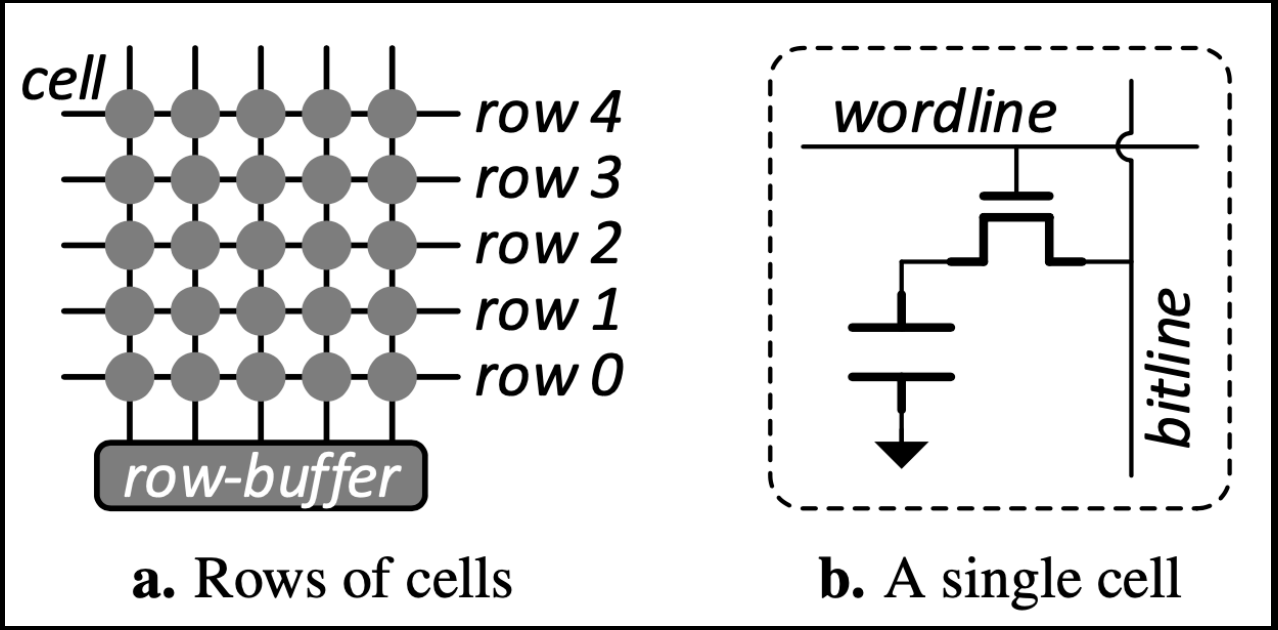
> How to describe a DRAM address of a bit?
  - a 6-tuple of the form <channel, DIMM, rank, bank, row, column>





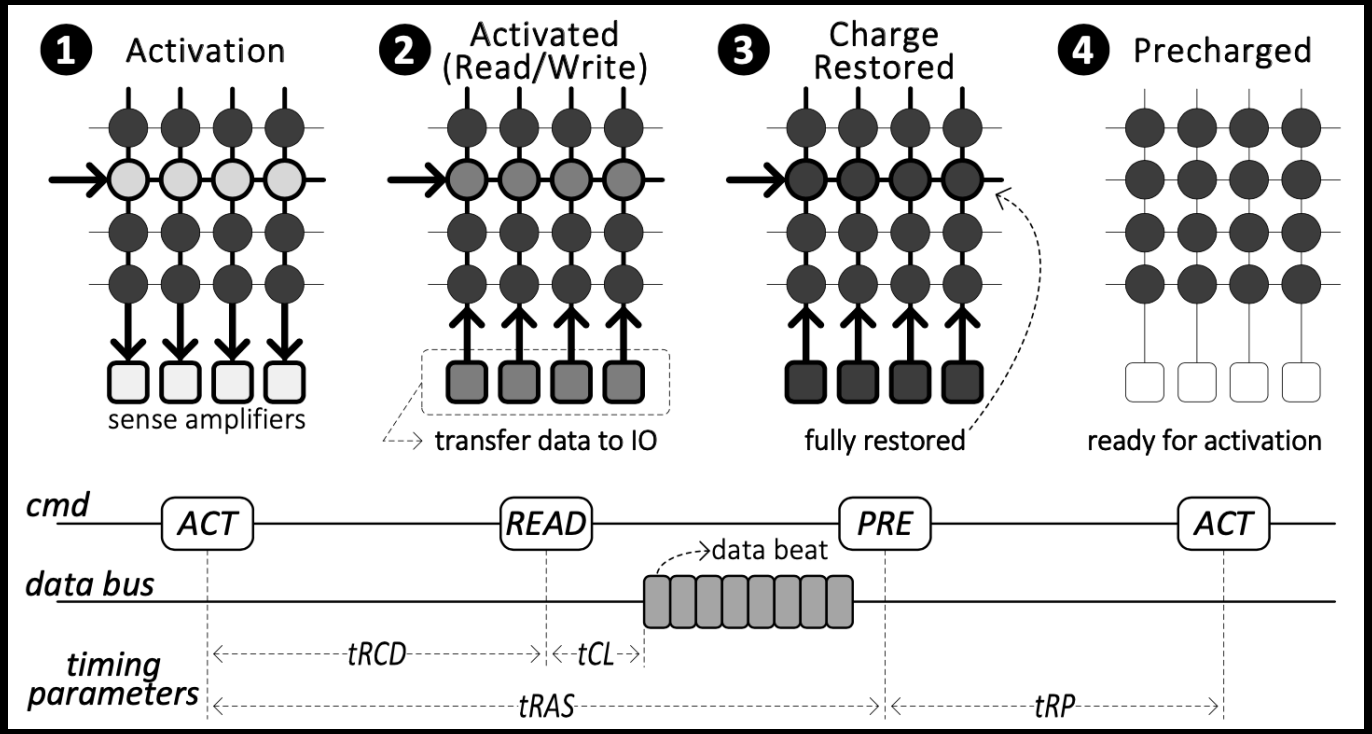
DRAM Background: Organization

> Take a closer look to the cells.



DRAM Background: Operation

> Activation, restoration and precharge.



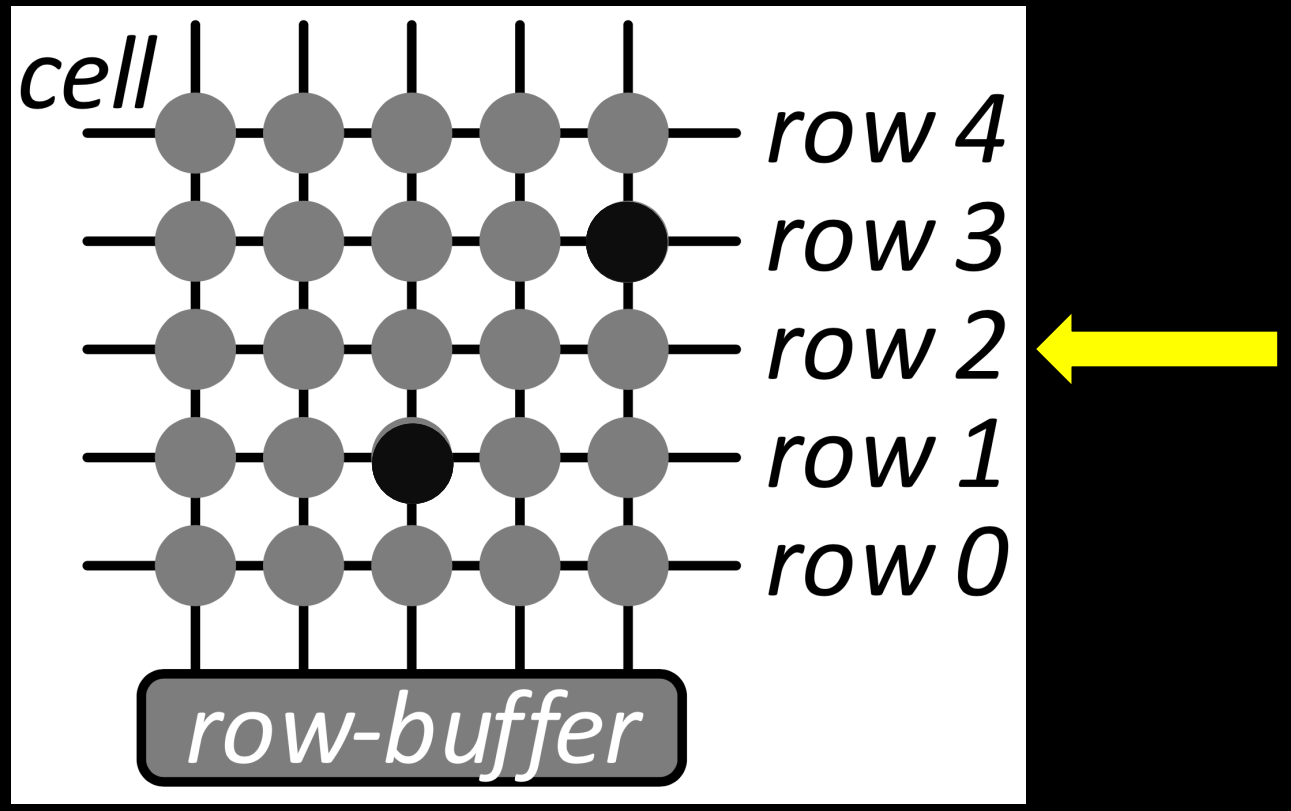
Disturbance Error Phenomenon -> RowHammer

Disturbance Error Phenomenon

> DRAM cells store data in capacitor. But capacitor leaks charge all the time, so should be refreshed in a certain interval.

> Repeated toggling of a DRAM row's wordline stresses inter-cell coupling effects that **accelerate charge leakage** from nearby rows. When such a cell loses too much charge in a refresh interval, it experiences a disturbance error.

Disturbance Error Phenomenon



RowHammer

> Malicious attackers can exploit disturbance error phenomenon to break the ***memory isolation security principle*** to cast great threat on computer systems.

> In later works, the exploitation of disturbance error phenomenon is called *RowHammer*, meaning repeatedly "hammering" a row induces bit flips into adjacent rows.

Characterization

Characterization: Circuit Level

> DRAM disturbance errors are caused by the repeated opening/closing of a row, not by column reads or writes.

> Disturbance Errors are widespread across DRAM types and vendors.

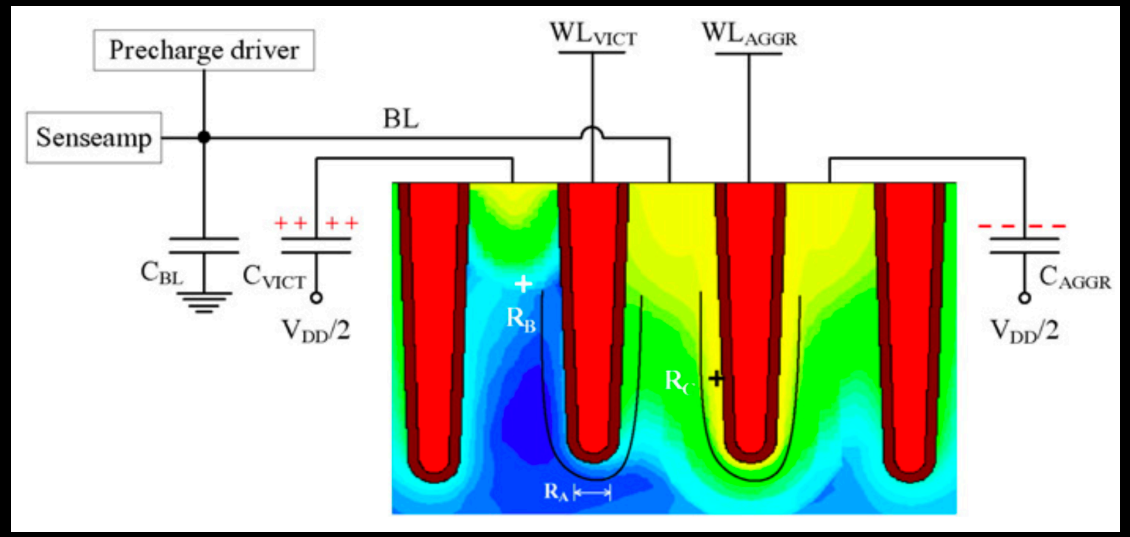
> Sensitivity Results: errors are mostly repeatable; victim cells ≠ weak cells; not strongly affected by temperature (but affected).

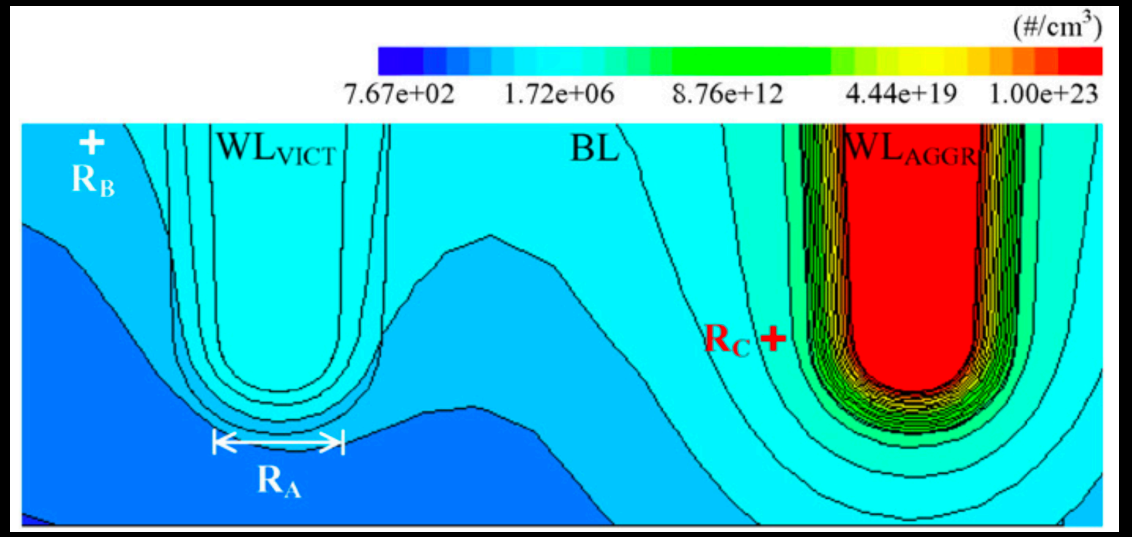
Characterization: Circuit Level

- **Access Pattern Dependence**: the more activations in a refresh interval, the more bit-flip errors.
- **Address Correlation**: an aggressor row and its victim rows are likely to have consecutive physical row-addresses, i.e., physically adjacent.
- **Data Pattern Dependence**: despite rare exceptions, every other victim cell (cell experiencing bit flip) had an error in just a single preferred direction (leakage direction). For example, one cell can only be flipped from 1 to 0 but cannot reverse.

Characterization: Semiconductor Level

> RowHammer fault is primarily caused by the **charge recombination** (electrons and charges combine to electrical neutrality) of the victim cell with the charges in the channel of the neighborhood cell.





Exploitation

Exploitation

> Exploitation of RowHammer covers almost every field of computer security, generates lots of innovative attack vectors and methodologies and force many mechanisms to be disabled or look for substitutes.

Exploitation

- Security techniques broken through by RowHammer:
  - Privilege/authority control: [18], Drammer [7], GuardION [19], Dedup Est Machina [20]
  - Process isolation, sandbox and virtual machine: Project Zero, Dedup Est Machina [20], Flip Feng Shui [6], ECCploit [10]
  - Digital signature and cryptosystems: Flip Feng Shui [6], ECCploit [10]
  - Denial-of-service: Locking down the processor [21]
  - Remote system/network: Throwhammer [22]

Exploitation

- Three classical/representative exploitations used in many works to evaluate attack feasibility:
  - exploit targeting PTEs (Page Table Entries) to obtain kernel privileges
    - representative: Drammer
  - corrupts RSA public keys to gain access to a co-hosted VM
    - representative: Flip Feng Shui
  - flip bits on the sudo binary opcode to bypass permission checks
    - representative: Another Flip

Exploitation

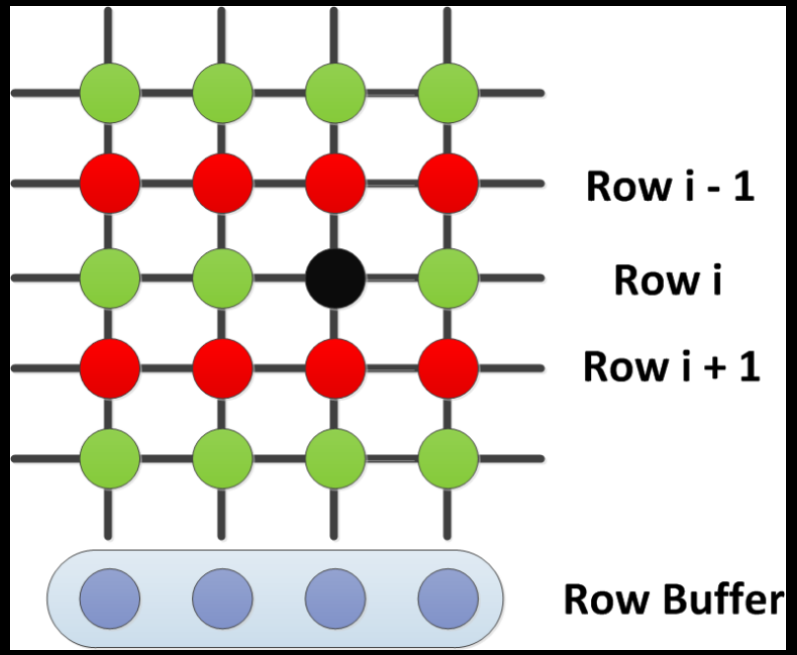
> The common 3-step attack methodology (firstly proposed by FFS):

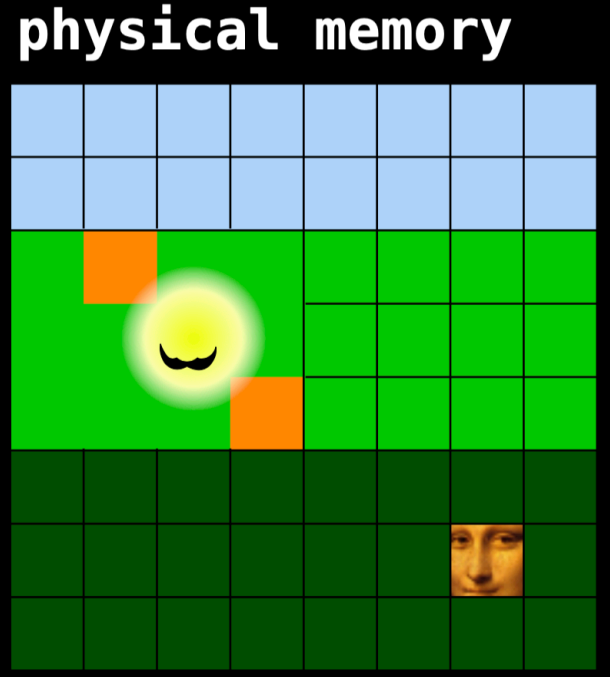
- **Memory Templating**: find the RowHammer templates in memory, such as row addresses, access patterns, bit-flip offsets and directions and data patterns.
- **Memory Massaging**: steer targeted sensitive data towards the vulnerable physical memory locations found in memory templating step.
- **Exploitation**: exploit RowHammer bit flips to compromise target systems.

Using FFS an example

Exploitation: Example - Memory Templating

> FFS conducts memory templating to get the knowledge of usable bit-flip patterns (bit-flip direction, bit-flip offset, and data pattern of the page) using a variant of double-sided RowHammer.

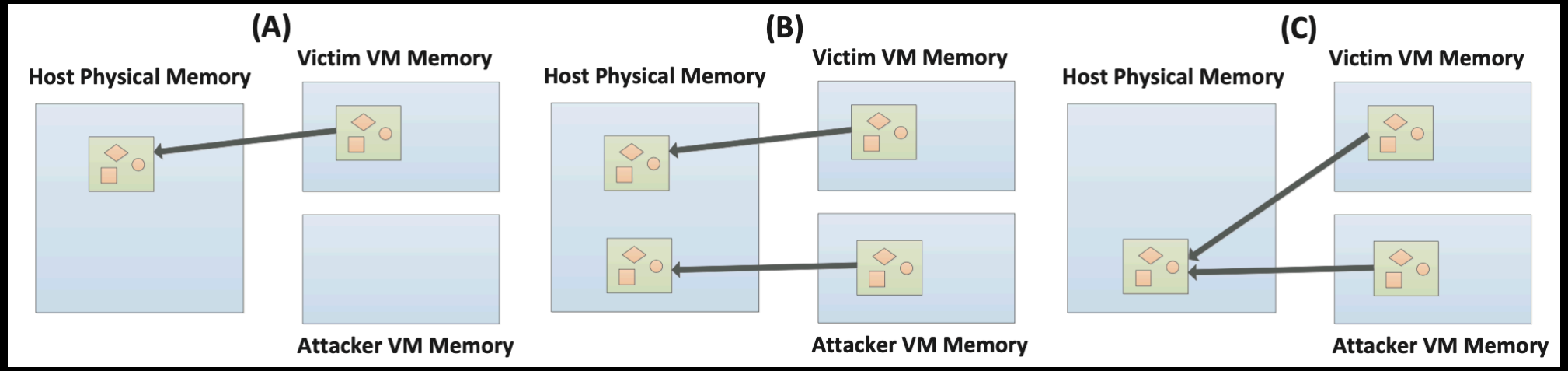




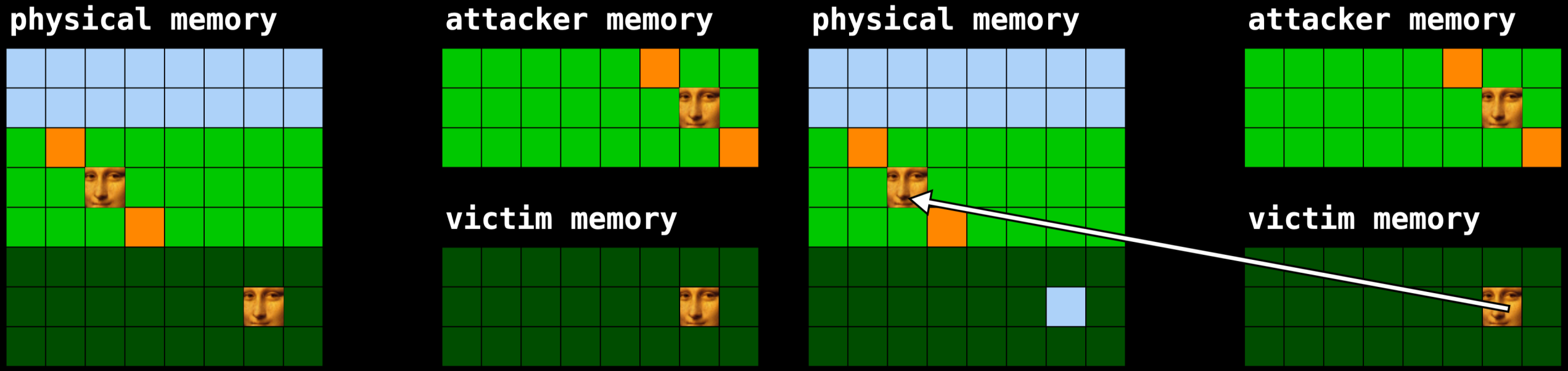
Exploitation: Example - Memory Massaging

> FFS exploits memory deduplication to steer targeted sensitive data towards the vulnerable physical memory locations found in memory templating step.

Memory deduplication can provide an attacker control over the layout of physical memory (merges pages having same contents and backs the first registered physical page).



Exploitation: Example - Memory Massaging



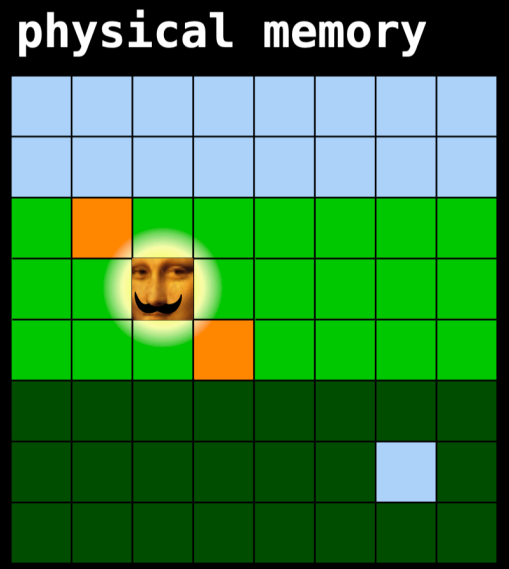
Exploitation: Example - Exploitation

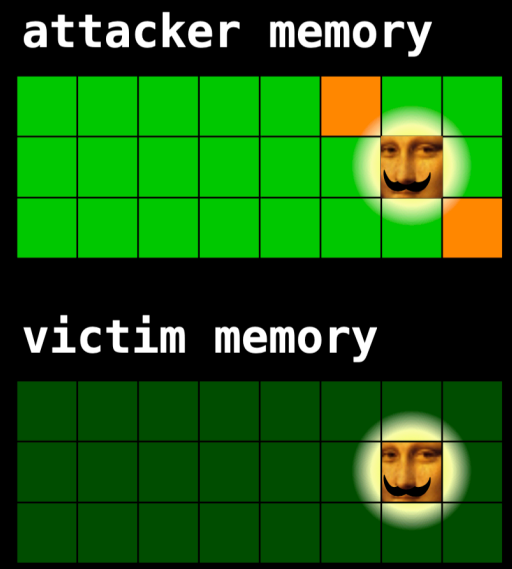
> Finally, FFS uses RowHammer to induce bit flips in RSA public keys in co-hosted VM's memory, factorize them and generate private keys, successfully compromising OpenSSH and apt package distribution system in a co-hosted cloud VMs.

OpenSSH ~/.ssh/authorized_keys

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC52/Uk84iUmmic
el7ESr+/D/PWZ6Ljkhlu8yv35bEEoTwXm9eGxJyzV+1s68tRyzpD
3VQvwSHiKqDnCg+OtaAo0KvCqZcoBQFB9XawIfJI5dSeGtcUBuok
Uv+TlmAZ+D9MNNAxjuSBBH0ShbaiH65imlauISfR3VZWFE7uy6sB
26j52LhWG5BRwSkMnMRN2E2fqHaP96J9ROFlHuykw8jwUXJwl4kJ
8vRo1uhX0SVu8Z9wGrKR5b+GQWJ3Ph7vjoMVU/KoAbWnNnYKR8IT
BnkPD0LrEyAKRygEfi7gwcixOvQR79by8lL6ypJ4kM5eyobSBsNC
jmghxQj8RRzGUtd1 victim@laptop
```

Exponent Modulus (p' * q' * r' ...)





Exploitation: Optimization

> Use accurate DRAM addressing: RAMSES [9].

> Bypass ECC: ECCploit [10].

> Bypass TRR: TRRespass [11].

> Use near-optimal activation rate: An End-to-End Methodology for Cloud Providers [12].

Mitigation

Mitigation: Hardware-Level

> Summary of hardware-level mitigation methodologies.
  - Double refresh rate
  - Error correction: ECC
  - Counter-Based structure: TRR, TWiCe [17], MRLoc [16]
  - Probabilistic Adjacent Row Activation (PARA)
  - RowHammer manufacturing profiling

Mitigation: Software-Level

> Summary of software-level mitigation methodologies.
  - Guard rows (isolation): CATT [14], GuardION [19], ZebRAM [15].
  - Counter-Based: ANVIL [23], ProHIT [24].
  - RowHammer online profiling

Mitigation: Problems

- Increase refresh rate cannot prevent RowHammer bit flips, but induces much overhead.
- ECC and TRR (counter-based method) were demonstrated to be bypassed.
- Guard rows (isolation) methods induce prohibitive overhead.
- RowHammer profiling costs too much and cannot be applied widely.

Conclusion

Conclusion

> RowHammer breaks the ***memory isolation security principle*** to cause exploitations.

> Mitigations focus on two directions, corresponding to the two root causes of RowHammer: 1) reduce leakage by refresh; 2) reduce inter-cell coupling effects by increasing distance between rows.

> General-purpose hardware is fallible (in a very widespread manner) and its problems are actually exploitable. So step outside our comfort zone.

References

[1] Kim, Yoongu, et al., "Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors," in ISCA, 2014.

[2] Kim, Jeremie S., et al., "Revisiting RowHammer: An Experimental Analysis of Modern DRAM Devices and Mitigation Techniques," in ISCA, 2020.

[3] Mutlu, Onur, and Jeremie S. Kim, "RowHammer: A retrospective," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2019.

[4] T. Yang et al., "Trap-Assisted DRAM Row Hammer Effect," EDL, 2019.

[5] K. Park et al., "Experiments and Root Cause Analysis for Active-Precharge Hammering Fault in DDR3 SDRAM under 3×nm Technology," MR, 2016.

[6] K. Razavi et al., "Flip Feng Shui: Hammering a Needle in the Software Stack," in USENIX Security, 2016.

[7] V. Van Der Veen et al., "Drammer: Deterministic Rowhammer Attacks on Mobile Platforms," in CCS, 2016.

[8] D. Gruss et al., "Another Flip in the Wall of RowHammer Defenses," in SP, 2018.

[9] A. Tatar et al., "Defeating Software Mitigations against Rowhammer: A Surgical Precision Hammer," in RAID, 2018.

[10] L. Cojocar et al., "Exploiting Correcting Codes: On the Effectiveness of ECC Memory Against RowHammer Attacks," in SP, 2019.

[11] P. Frigo et al., "TRRespass: Exploiting the Many Sides of Target Row Refresh," in SP, 2020.

References

[12] L. Cojocar et al. "Are We Susceptible to Rowhammer? An End-to-End Methodology for Cloud Provider," in SP, 2020.

[13] Z. B. Aweke et al., "ANVIL: Software-Based Protection Against Next-Generation Rowhammer Attacks," in ASPLOS, 2016.

[14] F. Brasser et al., "Can't Touch This: Practical and Generic Software-only Defenses Against RowHammer Attacks," USENIX Security, 2017.

[15] R. K. Konoth et al., "ZebRAM: Comprehensive and Compatible Software Protection Against Rowhammer Attacks," in OSDI, 2018.

[16] J. M. You et al., "MRLoc: Mitigating Row-Hammering Based on Memory Locality," in DAC, 2019.

[17] E. Lee et al., "TWiCe: Preventing Row-Hammering by Exploiting Time Window Counters," in ISCA, 2019.

[18] Seaborn, M., Dullien, T., "Exploiting the DRAM Rowhammer Bug to Gain Kernel Privileges," in BHUS, 2015.

[19] V. van der Veen et al., "GuardION: Practical mitigation of DMA-based rowhammer attacks on ARM," in DIMVA, 2018.

[20] E. Bosman et al., "Dedup Est Machina: Memory Deduplication as an Advanced Exploitation Vector," in S&P, 2016.

[21] Y. Jang et al., "SGX-Bomb: Locking Down the Processor via RowHammer Attack," in SysTEX, 2017.

[22] A. Tatar et al., "Throwhammer: Rowhammer Attacks over the Network and Defenses," in USENIX ATC, 2018.

[23] Z. B. Aweke et al., "ANVIL: Software-Based Protection Against Next-Generation Rowhammer Attacks," in ASPLOS, 2016.

[24] M. Son et al., "Making DRAM Stronger Against Row Hammering," in DAC, 2017.