Exploiting Correcting Codes: On the Effectiveness of ECC Memory Against Rowhammer Attacks

Lucian Cojocar, Kaveh Razavi, Cristiano Giuffrida, Herbert Bos

VU UNIVERSITY AMSTERDAM

VUSec
https://vusec.net
Grad students who have no life. Which is most of them I think. Get enough grad students, and you can parallelize some of the gathering of data.
Rowhammer (RH) causes bits to flip

- Exploit to escalate privilege [Seaborn '15]
- Exploit to escape sandboxes [Seaborn '15, Gruss '18]
- Exploit to compromise confidentiality [Razavi '16]
- Exploit different targets:
  - Desktop computers (browser, local shell etc.)
  - On phones [van der Veen '17], on GPUs [Frigo '18]
  - Over the network [Tatar '18, Lipp '18]
Previous RH attacks are on non-server memory
Previous RH attacks are on non-server memory

ECCploit, RH on server (ECC) memory
Overview

1) Challenges for RH on ECC memory
2) Single-bit flips on ECC memory
   1) Causing them
   2) Observing them
3) Reverse engineering of ECC functions
4) Performance of Rowhammer on ECC memory
What makes the exploitation of ECC memory difficult?
IF AT FIRST YOU DON'T SUCCEED

USE MORE C4
It is hard (and dangerous) to get 3 bit flips

1 bit flipped

2 bits flipped

3 bits flipped
It is hard (and dangerous) to get 3 bit flips

1 bit flipped

2 bits flipped

3 bits flipped

Corrected by ECC
It is hard (and dangerous) to get 3 bit flips

1 bit flipped → Corrected by ECC
2 bits flipped → Potentially uncorrectable machine crash
3 bits flipped
It is hard (and dangerous) to get 3 bit flips

1 bit flipped ➞ Corrected by ECC

2 bits flipped ➞ Potentially uncorrectable machine crash

3 bits flipped ➞ Potentially uncorrectable potentially undetectable
It is hard (and dangerous) to get 3 bit flips

1 bit flipped → Corrected by ECC
2 bits flipped → Potentially uncorrectable machine crash
3 bits flipped → Potentially uncorrectable potentially undetectable

Kind of useless for Rowhammer
It is hard (and dangerous) to get 3 bit flips

Rowhammer on ECC memory is a mere DoS attack!
It is hard (and dangerous) to get 3 bit flips

ECCploit is an upgrade from the DoS attack. ECCploit only causes undetectable bit flips
Q: How to get from one bit flip to three bit flips without hitting two bit flips?
A: Templating bit flips on ECC memory
(ECCploit)

1. Get single bit flips

2. Combine them to cause silent corruptions (same ECC)
Challenge: causing a single bit to flip
Challenge: causing a single bit to flip
Challenge: causing a single bit to flip
Challenge: causing a single bit to flip

A: 1 1 1 1 1 1 ... 1
V: 0 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
V: 1 0 1 1 1 1 ... 1
V: 1 1 0 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
V: 1 1 1 0 1 1 ... 1
V: 1 1 1 0 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
V: 1 1 1 1 0 1 ... 1
V: 1 1 1 1 0 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
A: 1 1 1 1 1 1 ... 1
Challenge: observing a single bit flip
Challenge: observing a single bit flip
ECC correction is observable
A: Templating bit flips on ECC memory (ECCploit)

1. Get single bit flips

2. Combine them to cause silent corruptions (same ECC)
Challenge: finding a suitable 3 bit flip that cause silent corruptions
Challenge: finding a suitable 3 bit flip that cause silent corruptions
Challenge: finding a suitable 3 bit flip that cause silent corruptions

Reverse engineering the ECC implementation
ECC errors reveal the ECC function

Fault injection on the memory bus

Cold-boot attack
ECC errors reveal the ECC function

Fault injection on the memory bus

Cold-boot attack
CPU writes data and control bits

```plaintext
*ptr = data;  ControlBits = ECC(data);
```

ECC bits are stored next to data
CPU writes data and control bits

*ptr = data;  ControlBits = ECC(data);

ECC bits are stored next to data
CPU reads data and checks control bits

```
data = *ptr;
CB_exp = ECC(data);
if (CB_read != CB_exp)
    Error(DataForRAS);
```

ECC bits are stored next to data
We can reconstruct the ECC function by observing ECC errors.

```
data = *ptr;
CB_exp = ECC(data);
if (CB_read != CB_exp)
    Error(DataForRAS);
```

ECC bits are stored next to data.
We can reconstruct the ECC function by observing ECC errors

```c
data = *ptr;
if (CB_read != CB_exp)
    Error(DataForRAS);
```

ECC bits are stored next to data.
We can reconstruct the ECC function by observing ECC errors.

```c
data = *ptr;
CB_exp = ECC(data);
if (CB_read != CB_exp)
    Error(DataThatWeUseForRE);
```

ECC bits are stored next to data.
ECCploit attack

1) Recover the ECC function (offline)

2) Template the memory
   1) Avoid crashes by triggering only single-bit flips
   2) Knowing the ECC function, combine single bit flips in undetectable bit flips

3) Massage the memory

4) Run the Exploit
How long it takes to template ECC memory for Rowhammer?*

*On our setup
How long it takes to template ECC memory for Rowhammer?

- If a perfect side channel (bit granularity) it takes:
  - 32 minutes for PTE or code change
  - 2 hours for the RSA key attack

*On our setup
How long it takes to template ECC memory for Rowhammer?*

- If a perfect side channel (bit granularity) it takes:
  - 32 minutes for PTE or code change
  - 2 hours for the RSA key attack

- If a typical side channel (word granularity) it takes:
  - 19 hours for PTE or code change
  - 3 days for RSA key attack

*On our setup
Error Correcting Codes: Only Slow Down Rowhammer Attacks

https://vusec.net/projects/eccploit

https://vusec.net