Welcome back to CCCCCC!

Week C

Ed meme recap:

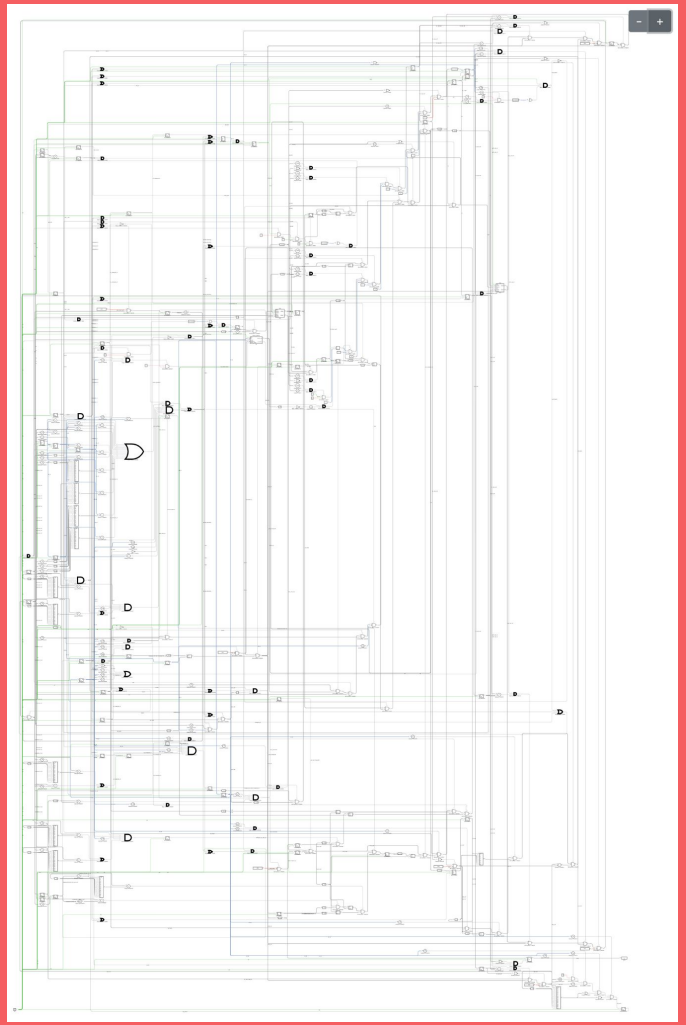
NAND flash #638

Anonymous
2 days ago in General

PIN STAR WATCH 125 VIEWS

9

Comment Edit Delete Endors



Questions on lecture content?
Or about cats?

Quiz everyone say MEOW!

Poll

```
0x57687920
0x64696420
0x796F7520
0x6465636F
0x64652074
0x68697300
```

How was the quiz?

A. easy
B. mostly fine
C. mostly fine, but not enough time
D. too hard, but finished mostly in time
E. too hard and not enough time
F. too hard regardless of time

Stress

- 429H is not an easy class
  - Lots of new materials
  - Unfamiliar programming environments
  - Fast, often relentless pace
- Struggling in this course is normal
  - There will be times you won't know the answer of the solution
  - This is expected—we want we everyone to succeed, but the only way we can help is if you ask for it
- If you find yourself overly overwhelmed or spending more time on this class than you think you should be, please reach out to Dr. Gheith or the TAs
  - We can help out as far as the class goes
  - We can provide other resources where we are not able to help

Mental health resources available at UT

Do your course evaluations!

- It would make us happy
- Pizza party on May 1st if all of the instructors get at least 50/55 responses?

Thank you everyone for your submitted quiz questions!

- We appreciate the effort that went into creating quiz questions :)

Final Project

Poll

How’s your status on Final Project?

A. What’s Final Project?
B. I have a group, but we have not started yet
C. My group has started planning
D. My group has started writing code
E. My group has made significant progress
F. Final Project any% speedrun

Final Project Presentations are Next Week!!

- It seems like everyone has signed up for a presentation time
  - Good job self organizing :)
- You will have some time after the presentation to finalize and submit your project (likely due Monday 4/29)

What we are looking for in presentation

- Be prepared!!
  - Have a backup plan if your live demo doesn't work
- Explain your work
  - Provide background that is appropriate for CS429H students
  - Ideally people will learn something about architecture from your presentation!
- Demonstrate what you did
  - Show screenshots of results, live demos, whatever is appropriate for your project

Things we may consider while grading

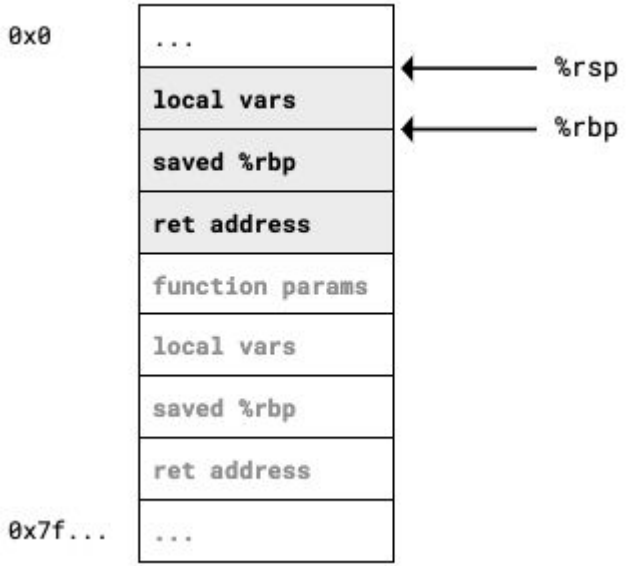
- Difficulty of project
- Accomplishments
- How much the class learned

so we spent this semester learning how to build a computer…

how do we break it?

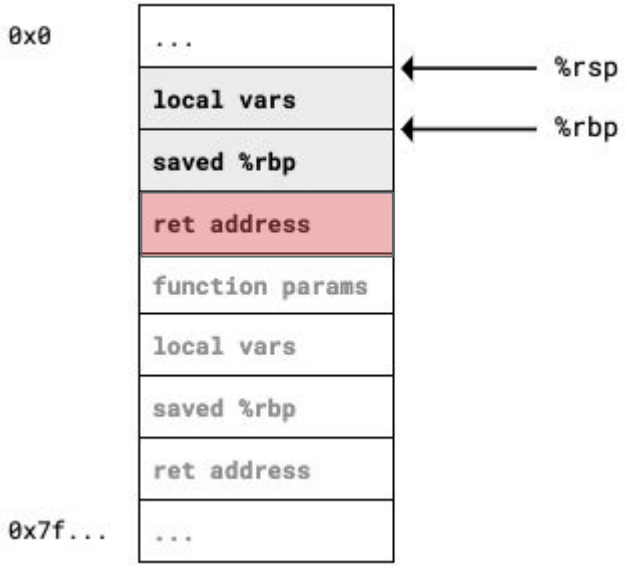
What is a stack?

- What is the layout of the stack?
- What happens on a function call?
- What potential problems could we run into?

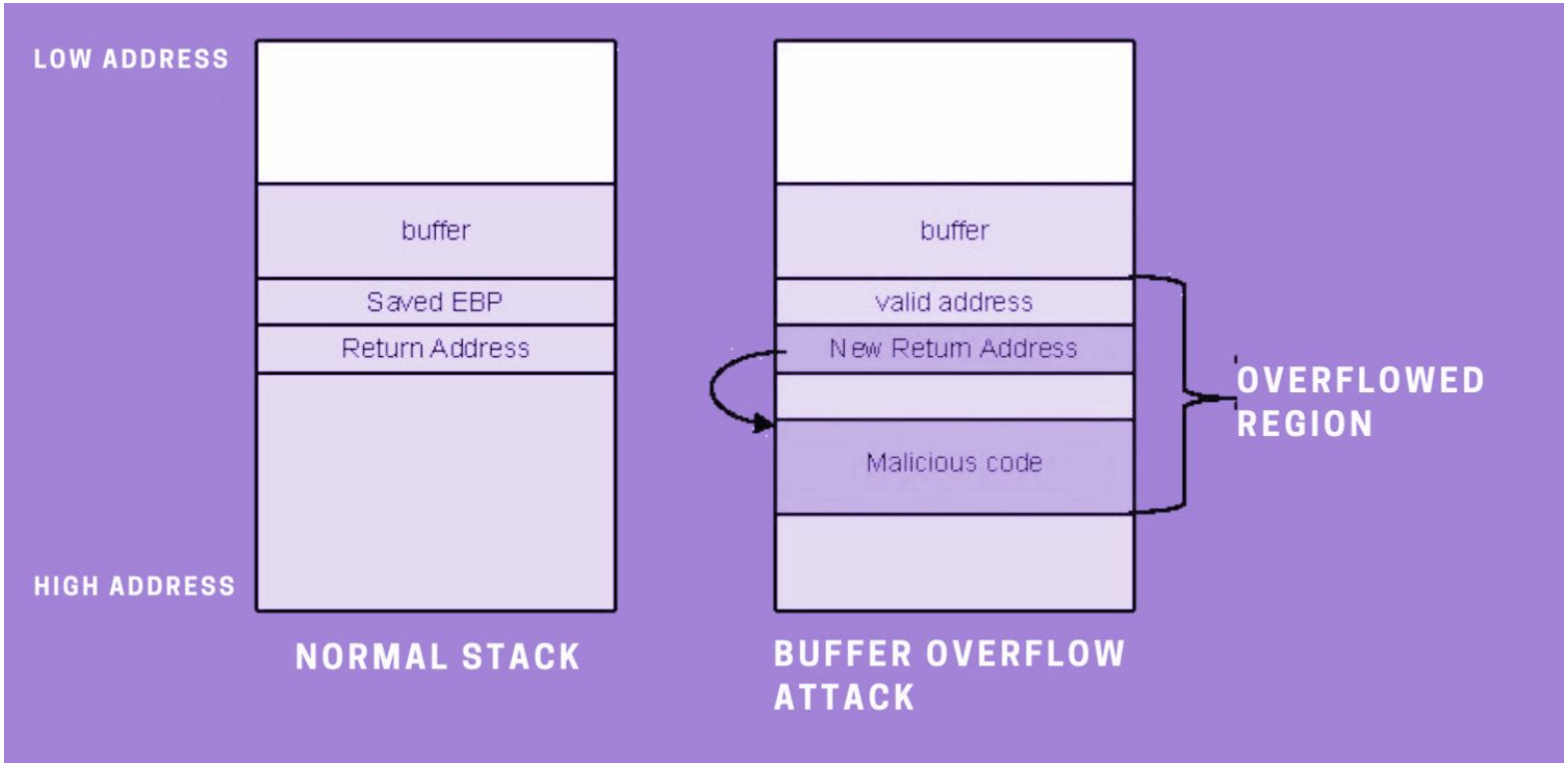


What is a stack?

- Could we change the return address to something else?
- If we were an attacker, what could we do to execute code we want?



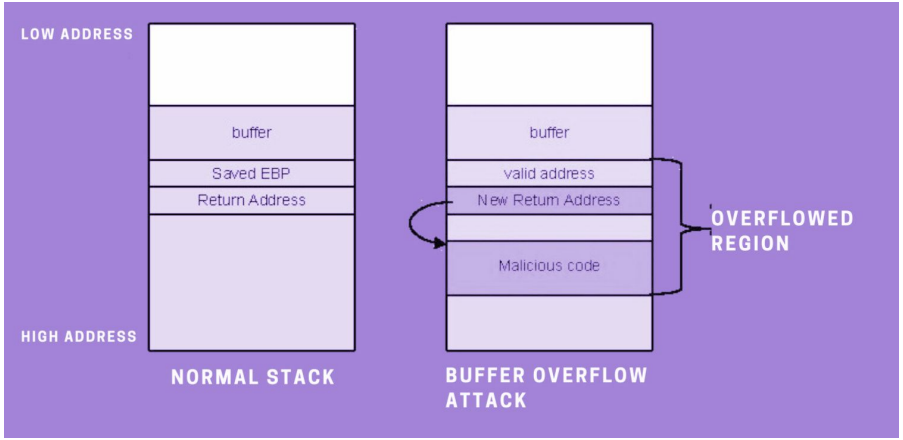
Buffer Overflow Attack



Buffer Overflow Attack

```
void f(char * str) {
    char buffer[16];
    strcpy(buffer, str);
}
```

Can you describe the string you could pass in to `f` that would execute the buffer overflow attack shown on the right?

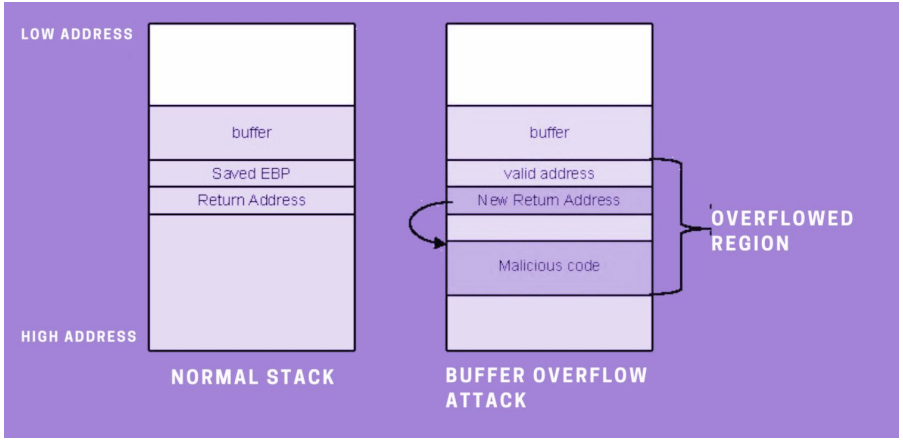


Buffer Overflow Attack

```
void f(char * str) {
    char buffer[16];
    strcpy(buffer, str);
}
```

we need bytecode that we can execute

is there anything special we need to consider? why?



Any ideas how we can fix this?

Any ideas how we can fix this?

- No parts of memory can be **both** writable and executable
- Control flow integrity (CFI)
- Use safer string functions (e.g. `strncpy` rather than `strcpy`)

Any ideas how we can fix this?

- No parts of memory can be **both** writable and executable
- Control flow integrity (CFI)
- Use safer string functions (e.g. `strncpy` rather than `strcpy`)

How can we implement and/or break these?

Any ideas how we can fix this?

- No parts of memory can be **both** writable and executable
- Control flow integrity (CFI)
- Use safer string functions (e.g. `strncpy` rather than `strcpy`)

How can we implement and/or break these?

- return oriented programming
- page table permission bits
- control flow graph
- stack canaries
- address space layout randomization (ASLR)
- compiler warnings / static analysis

spectre… aka exploiting all our optimizations :)

Spectre

Consider the following pseudocode residing in a victim's code and suppose that x is based on some input that an attacker has access to.

```
if x in bounds:
    y = arr[x]
```

How might speculative execution affect how this code runs?

Spectre

Consider the following pseudocode residing in a victim's code and suppose that x is based on some input that an attacker has access to.

```
if x in bounds:
    y = arr[x]
```

We may speculatively execute this line of code even when x is out of bounds.

In fact, an attacker can trick it into running (how?).

Spectre

Consider the following pseudocode residing in a victim's code and suppose that x is based on some input that an attacker has access to.

```
if x in bounds:
    y = arr[x]
```

We may speculatively execute this line of code even when x is out of bounds.

In fact, an attacker can trick it into running by invoking this code with valid x's a bunch of times.

Spectre

But who cares? Do we care? Does it matter?

Spectre

But who cares?

What happens when there is a misprediction?

- We flush and nothing has been committed in writeback so we are good!
- It’s like nothing happened :).
- ~~no one needs to know we messed up~~

Right?

Spectre

But who cares?

What happens when there is a misprediction?

- We flush and nothing has been committed in writeback so we are good!
- It's like nothing happened :).
- ~~no one needs to know we messed up~~

Wellll… the caches are affected. When we access memory, it gets pulled into the cache and we don't undo this.

Spectre

That’s ok though! Who cares what is in the cache? We can’t access it. Sensitive information is in the cache all the time.

So it’s fine?

Spectre

That's ok though! Who cares what is in the cache? We can't access it. Sensitive information is in the cache all the time.

So it's fine?

Ok so there is a timing difference when accessing memory, depending on whether it's in the cache or not (that's the whole point). So let's design an attack.

Spectre

Consider the following set up:

The attacker create an array the size of the cache and accesses the elements. This pulls our array into the cache. (priming)

Then we make the victims code run, and its secret gets placed into the cache.

We can then access elements in our array and see if there is a timing difference. (probing)

But what does this do?

Spectre

Consider the following set up:

The attacker create an array the size of the cache and accesses the elements. This pulls our array into the cache. (priming)

Then we make the victims code run, and its secret gets placed into the cache.

We can then access elements in our array and see if there is a timing difference. (probing)

We can tell where in the cache, and consequently in memory, the secret is based on the timing difference. That's not very interesting… What can we do now?

Spectre

Consider the following modified version of our earlier code:

```
if x in bounds:
    y = arr[arr[x]]
```

If we run the attack now, what do we get?

Spectre

Consider the following modified version of our earlier code:

```
if x in bounds:
    y = arr[arr[x]]
```

If we run the attack now, what do we get?

Well now we can get the location of y, which is arr[x]. We can now access anything in our victims address space! Yay!

Spectre

Consider the following modified version of our earlier code:

```
if x in bounds:
    y = arr[arr[x]]
```

If we run the attack now, what do we get?

Well now we can get the location of y, which is arr[x]. We can now access anything in our victims address space! Yay!

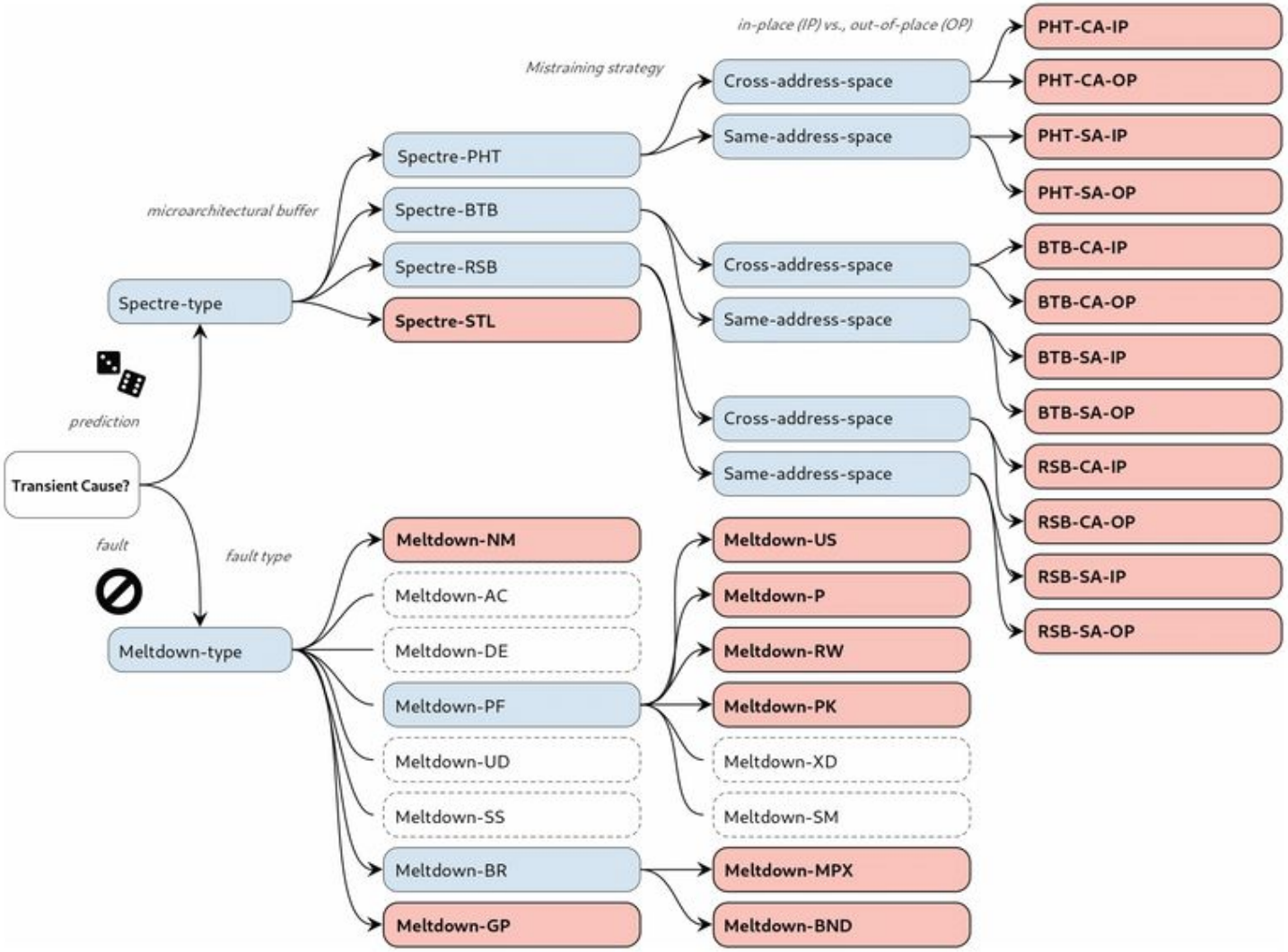
(small asides: technically, we would want scale our accesses, so that different values of arr[x] will result in memory getting written to different cache lines. also we may have to run this multiple times, since other processes may kick stuff out of the cache too.)

Mitigation?

Mitigation?

- Don’t do speculative execution.
  - Really slow
  - New hardware (Intel 2019+) mitigations
  - Can’t “fix” existing hardware
  - lfence around conditional branches - lots of libraries need to be patched :) also still slow
- Limit data extraction from covert channels
  - Inaccurate timers
    - Kinda sucks
    - Used for web applications
    - Ways to get around this (convert it into a question about how filled is the cache?)

Demo


in-place (IP) vs., out-of-place (OP)
Mistraining strategy
microarchitectural buffer
prediction
fault
fault type
Transient Cause?
Spectre-type
Meltdown-type
Spectre-PHT
Spectre-BTB
Spectre-RSB
Spectre-STL
Meltdown-NM
Meltdown-AC
Meltdown-DE
Meltdown-PF
Meltdown-UD
Meltdown-SS
Meltdown-BR
Meltdown-GP
Cross-address-space
Same-address-space
Cross-address-space
Same-address-space
Cross-address-space
Same-address-space
Meltdown-US
Meltdown-P
Meltdown-RW
Meltdown-PK
Meltdown-XD
Meltdown-SM
Meltdown-MPX
Meltdown-BND
PHT-CA-IP
PHT-CA-OP
PHT-SA-IP
PHT-SA-OP
BTB-CA-IP
BTB-CA-OP
BTB-SA-IP
BTB-SA-OP
RSB-CA-IP
RSB-CA-OP
RSB-SA-IP
RSB-SA-OP

Questions?

Thanks for listening to us talk for a whole semester <3