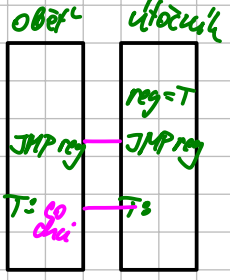
Spectre (2017-2018)

① obejití kontroly mezí

```
int f(uint i)
{ if (i<n)
    return x[i];
  else
    error();
}
y[64*f(i)];
```

`char x[...];` pro $n=2^k$

- funguje jen uvnitř adresového prostoru
  ↳ problém třeba s JS
- ∃ na všech CPU s predikcí skoků



② nepřímé skoky
- zneužití z jiného procesu
- ∃ prakticky všude

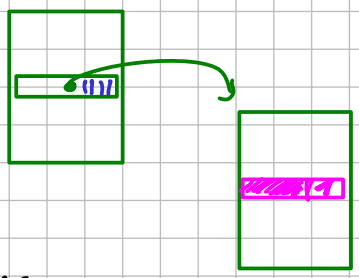
obrana: retpoline

```
call tmp
loop: (pause)
      jmp loop
tmp: movq reg,%(rsp)
     ret
```

L1TF (Level 1 Terminal Fault) 2018 (foreshadow)

- lze přečíst (spekulativně) data uložená v L1 na adrese určené neplatnou položkou MMU stromu
  ↳ virtualizace!

virtuální adresa uvnitř VM → MMU1 (ovládá VM) → fyz. adresa uvnitř VM → MMU2 (ovládá hostitel) → fyz. adresa hostitele



- jen Intel

↳ lze se bránit flushem L1 cache při přepínání VM
  ↳ oops: HT

MDS (Fallout, RIDL, Zombieload) 2019

- útoky na buffery

Store buffer
- forwardování dat
- porovnává jen část adresy (dolních 12 bitů)
- 56 položek (v HT rozdělen napůl)

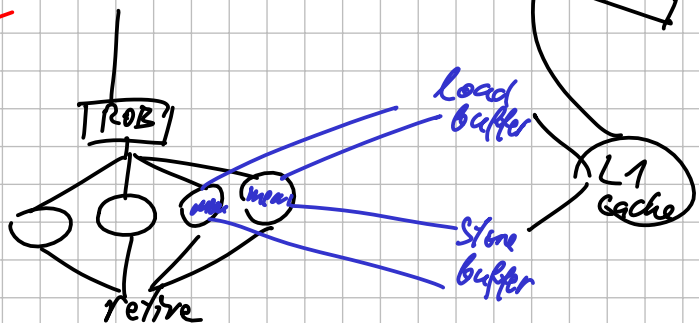
Load buffery

Fill buffery

útoky jen v rámci vlákna

cross-thread útoky

scénář:
① naplním store buffer
② zavolám kernel, který zapíše do nepřístupné paměti
③ čtu z adresy, kam mám přístup a která má stejných dolních 12 bitů



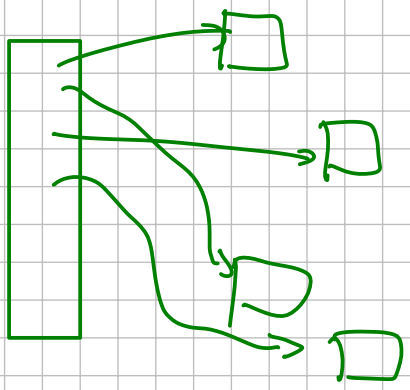
řešení: bariéry kolem syscallů a přerušení a uspání/probuzení vlákna

Třídění

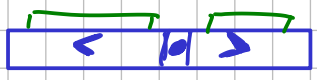
16 B položky ∋ 4B klíč

pro srovnání: qsort z knihovny
- adresa, # položek, velikost položky, komparátor

indirekce:



QuickSort



- na místě
- pro malé úseky přepnu na Insertsort
- vlastní zásobník
- na zásobník odkládám větší úsek
  - ↳ hloubka zásobníku ≤ log n