



UNIVERZITA
PAVLA JOZEFA ŠAFÁRIKA
V KOŠICIACH



Financované
Európskou úniou
NextGenerationEU

PLÁN [OBNOVY]



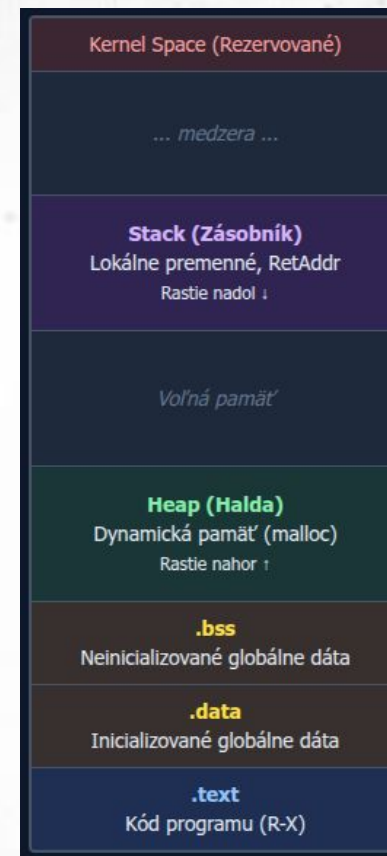
MINISTERSTVO
INVESTÍCIÍ, REGIONÁLNEHO ROZVOJA
A INFORMATIZÁCIE
SLOVENSKEJ REPUBLIKY

2. Analýza architektúry pamäte a exploitácia

Virtuálny adresný priestor (VAS)

Zobrazenie rozdelenia (Split):

- **Kernel Space (Vyššie adresy - 0xFFFF...):** Obsahuje kód jadra, mapovania zariadení, stránkovacie tabuľky. Nedostupné pre bežný proces (Ring 3).
- **User Space medzera (Canonical Gap):** Nevyužitá oblasť (non-canonical addresses), prístup sem vyvolá #GP (General Protection Fault).
- **User Space (Nižšie adresy - 0x0000...):**
 - **Stack (Zásobník):** Rastie smerom nadol (k nižším adresám). Obsahuje lokálne premenné, návratové adresy, stack frames.
 - **Memory Mapping Segment (mmap):** Kde sa nahrávajú zdieľané knižnice (.so súbory) a veľké alokácie pamäte.
 - **Heap (Halda):** Rastie smerom nahor (k vyšším adresám). Spravovaná cez `brk()`/`sbrk()` (hlavná aréna, menšie bloky) a `mmap()` (veľké bloky nad mmap threshold a arény vlákien).
 - **BSS / Data / Text:** Statické segmenty binárky.



Segmenty a oprávnenia

Segment	Obsah	Oprávnenia (RWX)	Typické útoky
.text	Strojový kód programu	R-X (Read, Exec)	Code Reuse (ROP)
.rodata	Konštanty, reťazce	R-- (Read Only)	Info Leak
.data	Inicializované globálne prem.	RW- (Read, Write)	Data Corruption
.bss	Neinicializované globálne prem.	RW- (Read, Write)	Data Corruption
Heap/Stack	Dynamické dáta	RW- (Read, Write)*	Overflow, UAF, ROP
GOT/PLT	Dynamické linkovanie	RW- (Partial/No RELRO) R-- (Full RELRO)	GOT Overwrite



Financované
Európskou úniou
NextGenerationEU

PLÁN [OBNOVY]



MINISTERSTVO
INVESTÍCIÍ, REGIONÁLNEHO ROZVOJA
A INFORMATIZÁCIE
SLOVENSKEJ REPUBLIKY



Aritmetika ukazovateľov

- **Základné pravidlo:** $\text{ptr} + n$ sa preloží na adresu $\text{ptr} + (n * \text{sizeof}(*\text{ptr}))$.
- **Zarovnanie (Alignment):** Prečo procesory preferujú čítanie z adries deliteľných 4 alebo 8? (Performance penalties, bus errors na iných architektúrach ako x86).
- **Nebezpečenstvo:** Ak programátor nesprávne pretypuje pointer a posunie ho, môže skončiť "uprostred" dátovej štruktúry, čím číta skomolené dáta alebo prepisuje kritické premenné.

```
struct P {  
    int id;          // 4 bajty  
    char name[8];    // 8 bajtov  
    // padding?     // 0-4 bajty (zarovnanie)  
};  
struct P pole[2];  
struct P *ptr = pole;  
  
// Klasická aritmetika  
int *i_ptr = (int*)ptr;  
i_ptr++;        // +4 bajty  
  
ptr++;          // +sizeof(struct P) (napr. 12 alebo 16 bajtov)
```



Undefined Behavior (UB) a Optimalizácie

- **Filozofia C:** Programátor vie, čo robí. Kompilátor (GCC/Clang) optimalizuje agresívne.
- **Strict Aliasing Rule:** Ďalší zdroj chýb. Kompilátor predpokladá, že pointery rôznych typov neukazujú na tú istú pamäť.
- **Dôsledok:** Bezpečnostné kontroly písané v C môžu v binárke úplne zmiznúť. Útočníci to zneužívajú na vyvolanie Integer Overflow, ktorý vedie k Heap Overflow.

```
void vuln(char *ptr, int size) {  
    char *end = ptr + size;  
    if (end < ptr) { // Kontrola pretečenia (Wraparound)  
        return; // Error  
    }  
    // ... zápis ...  
}
```



Úniky pamäte

Schéma typov únikov:

- **Definitely Lost:** Žiadny pointer neexistuje. (Memory Black Hole).
- **Indirectly Lost:** Pointer na rodiča stratený, deti sú stratené tiež (typické pre stromy/listy).
- **Still Reachable:** Pamäť, ktorú OS uprace po ukončení procesu (často globálne pooly), nie vždy chyba.

Prečo to vadí? Dlhodobé bežiacie služby (daemons) vyčerpajú RAM -> OOM Killer (Out Of Memory) zabije proces -> DoS (Denial of Service).

Bezpečnostný dopad: Memory Leaks (DoS): Zapríčinia vyčerpanie pamäte a pád služby. Heap Grooming / Feng Shui: Cílená manipulácia s alokáciami a voľnými blokmi (často legálnymi) s cieľom usporiadať haldu do predvídateľného stavu pred samotnou exploitáciou.

Nástroje: Valgrind je pomalý (emulácia CPU), ASan je rýchly (compile-time inštrumentácia).



Financované
Európskou úniou
NextGenerationEU

PLÁN [OBNOVY]



MINISTERSTVO
INVESTÍCIÍ, REGIONÁLNEHO ROZVOJA
A INFORMATIZÁCIE
SLOVENSKEJ REPUBLIKY



Anatómia Chunku (Malloc Chunk)



- **In-band Metadata:** Toto je Achillova päta ptmallocu. Metadáta, ktoré riadia štruktúru haldy, sú uložené hneď vedľa dát, ktoré ovláda útočník.
- **Heap Overflow:** Stačí prepísať **size** pole nasledujúceho chunku, aby sme zmenili jeho veľkosť, alebo zmazať **P** bit, aby si alokátor myslel, že predchádzajúci chunk je voľný (vyvolanie **unlink** makra).

- Size Flags (AMP): Spodné 3 bity veľkosti.
 - P (Prev_Inuse) : 0 = predchádzajúci chunk je voľný (umožňuje spájanie).
 - M (Mmap) : Alokované cez mmap.
 - A (Non-Main Arena) : 0 = Main Arena, 1 = Thread Arena.
- Zraniteľnosť spočíva v prepísaní metadát susedného chunku (Heap Overflow).



Algoritmus alokátora a Bins (Koše)

Tcache (Thread Local Cache)

Najrýchlejší. LIFO. Bez zamykania (per-thread). Limitovaná veľkosť (7 prvkov). Historicky veľmi zraniteľný (bez double-free kontrol).

Fast Bins

Malé bloky. LIFO. Nikdy sa nespájajú (no coalescing). Zdieľané (zamykanie).

Unsorted Bin

"Medzistanica". Dvojsmerný zoznam. Ak sa po free() chunk nezместí do Tcache/Fast, ide sem. Dáva "druhú šancu" pre rýchle znovupoužitie.

Small & Large Bins

Klasické úložisko. FIFO pre Small. Zoradené podľa veľkosti pre Large (skip-list).

Vývojový diagram alokácie:

1. Požiadavka `malloc(size)`.
2. Sedí do Tcache? -> Áno -> Vráť okamžite (LIFO).
3. Sedí do Fastbin? -> Áno -> Vráť (LIFO).
4. Je v Unsorted Bin? -> Prejdi zoznam, ak nájdeš presnú zhodu, vráť. Ak nie, roztrieď do Small/Large bins.
5. Hľadaj v Small/Large bins.
6. Top Chunk (odkroj z konca haldy).



Financované
Európskou úniou
NextGenerationEU

PLÁN [OBNOVY]



MINISTERSTVO
INVESTÍCIÍ, REGIONÁLNEHO ROZVOJA
A INFORMATIZÁCIE
SLOVENSKEJ REPUBLIKY

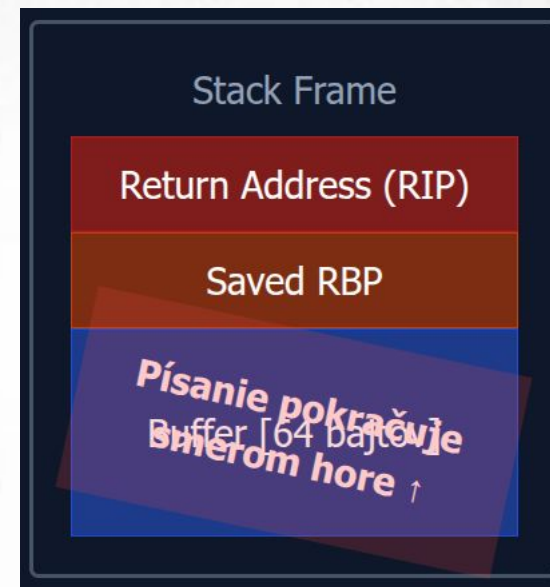


Stack Buffer Overflow

Stack Frame (alokácia):

1. Zavolanie funkcie: `call func` -> push RIP.
2. Prológ: `push rbp, mov rbp, rsp, sub rsp, 0x40` (alokácia buffera).
3. **Vstup:** `strcpy(buf, user_input)`.
4. **Pretečenie:** Dáta `AAAA...` zaplnia buffer, prepíšu Saved RBP a nakoniec Saved RIP.
5. Epilóg: `leave` (obnoví stack), `ret` (pop RIP -> skok na `0x41414141` -> SegFault).

```
void vuln(char *input) { char  
buf[64]; // Bez kontroly dĺžky!  
strcpy(buf, input); }
```



Ak vstup presiahne 64 bajtov + veľkosť RBP (8B), prepíšeme Return Address. Pri inštrukcii `ret` procesor skočí na nami zvolenú adresu (Shellcode alebo ROP).



Financované
Európskou úniou
NextGenerationEU

PLÁN [OBNOVY]



MINISTERSTVO
INVESTÍCIÍ, REGIONÁLNEHO ROZVOJA
A INFORMATIZÁCIE
SLOVENSKEJ REPUBLIKY



Off-by-One & Stack Pivoting

- Čo ak môžeme prepísať len jeden bajt za hranicou buffera?
- Tento bajt je zvyčajne najmenej významný bajt (LSB) uloženého Saved RBP.

Pôvodný RBP: `0x7fffffffef4b0`

Po prepise: `0x7fffffffef400` (LSB zmenené na 00)

1. Funkcia končí sekvenciou `leave; ret`.
2. `leave = mov rsp, rbp; pop rbp`.
3. Stack Pointer (RSP) sa presunie na falošnú adresu (posun o x bajtov nižšie).
4. Zásobník je "otočený" (pivoted) do oblasti, ktorú kontroluje útočník.



Financované
Európskou úniou
NextGenerationEU

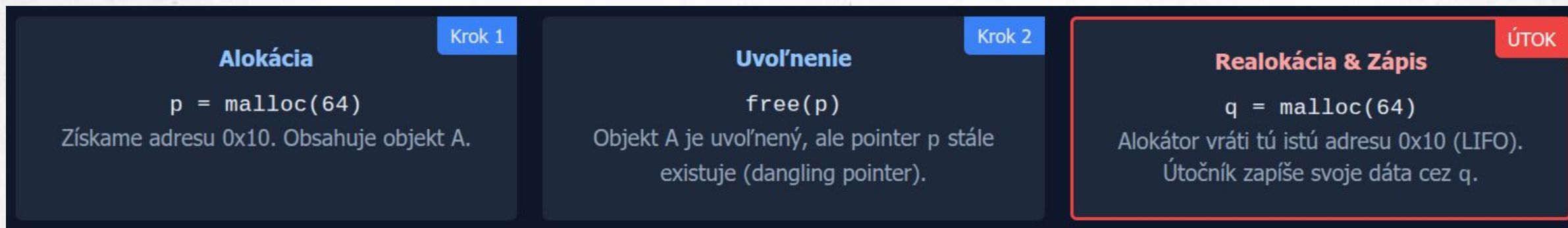
PLÁN [OBNOVY]



MINISTERSTVO
INVESTÍCIÍ, REGIONÁLNEHO ROZVOJA
A INFORMATIZÁCIE
SLOVENSKEJ REPUBLIKY



Use-After-Free (UAF)



Ak teraz program použije pôvodný pointer p (napr. zavolá funkciu cez pointer v štruktúre), vykoná kód útočníka, ktorý tam bol zapísaný v kroku 3.



Financované
Európskou úniou
NextGenerationEU

PLÁN [OBNOVY]



MINISTERSTVO
INVESTÍCIÍ, REGIONÁLNEHO ROZVOJA
A INFORMATIZÁCIE
SLOVENSKEJ REPUBLIKY



Tcache Poisoning & Double Free

- Manipulácia zretazovaného zoznamu voľných blokov.

```
// Double Free free(ptr); free(ptr);  
// V glibc 2.26-2.28 neexistovala kontrola dvojitého uvoľnenia v Tcache.  
// Od glibc 2.29 bola pridaná kontrola pomocou poľa key.  
// ptr pridaný do zoznamu 2x.  
// Zoznam: Head -> ptr -> ptr -> ...
```

- Následný malloc vráti ptr. Útočník do neho zapíše adresu cieľa (napr. &stack_var). Keďže ptr je stále v zozname ako "next", prepíšeme fd pointer.

Head -> ptr -> [Cieľ Útočníka]

- Ďalší malloc vráti cieľovú adresu -> Arbitrary Write Primitive.



Financované
Európskou úniou
NextGenerationEU

PLÁN [OBNOVY]



MINISTERSTVO
INVESTÍCIÍ, REGIONÁLNEHO ROZVOJA
A INFORMATIZÁCIE
SLOVENSKEJ REPUBLIKY



Ochranné mechanizmy (Mitigácie)

ASLR

Randomizácia adries. Vyžaduje **Information Leak** na zistenie bázy knižnice alebo haldy.

NX / DEP

No-Execute. Zásobník a halda nie sú spustiteľné. Zabraňuje klasickému shellcodu. Riešenie: **ROP**.

Stack Canaries

Náhodná hodnota pred RBP. Pri zmene program spadne. Riešenie: Leak kanárika.

Safe-Linking (GLIBC 2.32+)

Ochrana f d pointera v Tcache/Fastbins.

```
Stored_Ptr = (Addr >> 12) ^ Ptr
```

Útočník musí poznať adresu haldy, aby správne podvrhol pointer.



Financované
Európskou úniou
NextGenerationEU

PLÁN [OBNOVY]



MINISTERSTVO
INVESTÍCIÍ, REGIONÁLNEHO ROZVOJA
A INFORMATIZÁCIE
SLOVENSKEJ REPUBLIKY



Return Oriented Programming (ROP)

Ako obísť NX? Použijeme kód, ktorý už v pamäti je (gadgets).

Cieľ: `system("/bin/sh")`

V 64-bit Linuxe sa 1. argument dáva do registra RDI.

Adresa Gadgetu: `pop rdi; ret`

Adresa reťazca: `"/bin/sh"` (pôjde do RDI)

Adresa funkcie: `system()`

Útočník vyskladá tento "falošný zásobník" a po inštrukcii `ret` začne procesor vykonávať gadgets.

*Poznámka: Praktický problém (64-bit):
Volať funkciu `system()` vyžaduje zarovnanie zásobníka (RSP) na 16 bajtov kvôli inštrukcii `movaps`. Ak zásobník nie je zarovnaný, exploit skončí Segfaultom. Riešením je pridať do chainu jeden extra 'ret' gadget na posunutie RSP.*



Financované
Európskou úniou
NextGenerationEU

PLÁN [OBNOVY]



MINISTERSTVO
INVESTÍCIÍ, REGIONÁLNEHO ROZVOJA
A INFORMATIZÁCIE
SLOVENSKEJ REPUBLIKY



Záver a trendy

Boj medzi útočníkmi a obrancami je neustály. Zatiaľ čo jednoduché pretečenia zásobníka miznú, exploitácia haldy sa stáva dominantnou, hoci zložitejšou (séria útokov "House of X" ako House of Force/Spirit, Tcache poisoning).

- Ofenzíva: Automatizované hľadanie gadgetov, útoky na logiku alokátora.
- Defenzíva: Hardware mitigácie (Intel CET), prechod na memory-safe jazyky (Rust).



Financované
Európskou úniou
NextGenerationEU

PLÁN [OBNOVY]



MINISTERSTVO
INVESTÍCIÍ, REGIONÁLNEHO ROZVOJA
A INFORMATIZÁCIE
SLOVENSKEJ REPUBLIKY





UNIVERZITA
PAVLA JOZEFA ŠAFÁRIKA
V KOŠICIACH



Financované
Európskou úniou
NextGenerationEU

PLÁN [OBNOVY]



MINISTERSTVO
INVESTÍCIÍ, REGIONÁLNEHO ROZVOJA
A INFORMATIZÁCIE
SLOVENSKEJ REPUBLIKY

Ďakujem za pozornosť.