OSY rychle

• Značení

♣Zeleně – je možnost více správných odpovědí
Tučně – správná odpověď

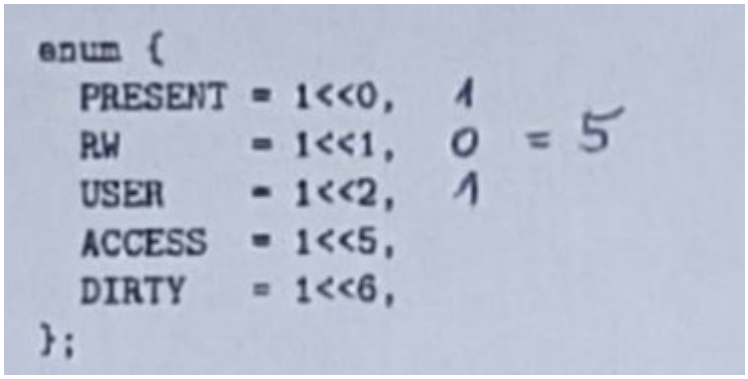
• Copyleft je:

A. Myšlenkový směr požadující dostupnost veškerého duševního vlastnictví všem zdarma

B. Strategie využívající autorského zákona k šíření uživatelských svobod

C. Strategie, jak obejít autorský zákon

• Co musí být nastaveno v tabulce stránek pokud virtuální adresa: 0xAAAAAAAA je součástí kódu uživatelského programu a uživatel z ní může číst, ale nesmí jí modifikovat. Dále požadujeme data z této virtuální adresy byla v paměti RAM na adrese 0x44444AAA



Správné nastavení odpovídající položky v tabulce stránek je:

A. 0x44444005
B. 0XAAAAA005
C. 0x11111AAA
D. 0XAAA44444
E. 0X44444AAA

• ♣Která následující tvrzení jsou pravdivá?

A. Semafor lze vždy použít místo mutexu
B. Semafor na rozdíl od mutexu čeká pomocí aktivního čekání (tzv. busy waiting)
C. V moderních OS semafor i mutex obsahuje frontu čekajících vláken
D. Mutex lze vždy použít místo semaforu
E. Mutexy a semafory není možné používat současně v jednom programu
F. Nic z výše uvedeného není správně

• Mějme následující program

```
int main() {
  int f=fork();
  if (f==0) {
    execl("./prg", "prg", NULL);
    print("Konec\n");
  } else {
    wait(NULL);
  }
  return 0;
}
```

Kdy se vytiskne hlášení “Konec”

G. Pokud program “prg” ukončí běh chybou, návratová hodnota programu “prg” bude rozdílná od 0

H. Vždy po ukončení programu “prg”

I. Pokud příkaz execl skončí chybou, neboť nelze spustit program “prg“

J. Pokud program “prg“ ukončí běh bez chyby s návratovou hodnotou programu “prg“ 0

- Mějme následující program:

```
int main() {
  int f=fork();
  if (f==0) {
    int a=open("a.txt", O_WRONLY | O_CREAT);
    printf("Start\n");
    dup2(a,1);
    printf("Exec\n");
    execl("./prg", "./prg", NULL);
  } else {
    wait(NULL);
  }
  return 0;
}
```

Pokud proběhne program bez chyb, pak:

A. hláška "Start" se vytiskne na standardní vystup, hláška "Exec" bude zapsána do souboru "a.txt“ a program "prg" bude mít přesměrován výstup do souboru "a.txt"

B. hláška "Start" a "Exec" se vytiskne na standardní výstup, a program "prg" bude mít přesměrován výstup do souboru "a.txt"

C. hláška "Start" a "Exec" se vytiskne do souboru "a.txt" a program "prg" bude mít přesměrován výstup do souboru "a.txt"

D. hláška "Start" se vytiskne na standardní výstup, hláška "Exec" bude zapsána do souboru "a.txt" a program "prg" bude mít výstup přesměrován na standardní výstup rodiče

E. hláška "Start" a "Exec" se vytiskne na standardní výstup a program 'prg' bude mít vstup také ze standardního vstupu

F. hláška "Start" se vytiskne na standardní výstup, hláška "Exec" se nevytiskne a program "prg" bude mít výstup do souboru "a.txt"

- Na čem závisí velikost stránek a rámců
  A. Na architektuře procesoru
  B. Na velikosti paměti
  C. Na velikosti disku
  D. Na velikosti adresové sběrnice paměti

- ♣ Inodový souborový systém
  A. Umožňuje vzdálený přístup do sítě
  B. Typicky dělí disk na superblok, bitmapy, tabulku inodů a datové bloky
  C. Umožňuje spočítat umístění daného inode na disku na základě jeho čísla (indexu)
  D. Neumožňuje používat tzv. extents, které implementují moderní souborové systémy
  E. Ukladá informace umístění souborů v datové struktuře zvané inode
  F. Nic z výše uvedeného není správně

- Systémové volání poll:
  A. Umožňuje současné čekání na události z více file desriptorů
  B. Synchronizuje běh více vláken tak, aby bylo možné v jednom okamžiku hlasovat ohledně výsledku paralelně běžící operace
  C. Implementuje smyčku událostí pro grafické aplikace
  D. Slouží k alokaci většího množství paměti (tzv. memory pool)

- Přepínání vláken v rámci jednoho procesu jádro OS řeší pomocí
  A. Vykonání instrukce ctxsw s povolenými přerušeními
  B. Uložení registrů CPU do kontextu původního vlákna a načtení jejich nové hodnoty z kontextu jiného vlákna
  C. Změny adresy v registru CR3 tak, aby ukazovala na adresář stránek nového vlákna
  D. Ošetření výpadku stránky (pagefault), který vzniká při přístupu nového vlákna k jeho registrům
  E. Vykonání instrukce ctxsw se zakázanými přerušeními

- ♣Útoku metodou return oriented programming lze zabránit
  A. Kontrolou integrity zásobníku před návratem z funkce
  B. Nastavením nespustitelného zásobníku
  C. Randomizací umístění zásobníku
  D. Randomizací rozložení kódu programu
  E. Nic z výše uvedeného není správně

- ♣Která tvrzení jsou pravdivá
  A. Program který nepoužívá sdílené prostředky může obsahovat deadlock
  B. Program obsahující deadlock skončí vzájemným čekáním a nikdy nedokončí svojí práci
  C. Pouze program, který používá sdílené prostředky může obsahovat deadlock

D. Program obsahující deadlock může někdy dokončit svoji práci
E. Nic z výše uvedeného není správně

• Jaký bude obsah proměnných A a B po provedení následujícího skriptu:

```
A=1
B=1
A=$((A+1))
B=${B}+1
```

A. A je 2, B je 2
B. A je 1+1, B je 1+1
C. A je 2, B je 1+1
D. A je 1+1, B je 2

• ♣Hardwarově asistovaná virtualizace
A. Zrychluje běh virtuálního uživatelského režimu
B. Zrychluje vykonávání některých privilegovaných instrukcí
C. Používá mechanismus trap and emulate k emulaci hardware
D. Funguje jedině na systému s podporou UEFI firmwarem
E. Přidává další vrstvu stránkovacích tabulek
F. Nic z výše uvedeného není správně

• Jaká bude hodnota proměnné A po provedení následujícího skriptu:

```
X='\home\usr\readme.txt'
A=${X//\\/\/}
```

X='\home\usr\readme.txt' A=${X//\\/\/}
A. /home/usr/readme.txt
B. \/home\/usr\/readme.txt
C. /\/home/\/usr/\/readme.txt
D. /\home/\usr/\readme.txt

• Aplikace psané v jazyce C/C++ nejčastěji volají služby OS pomocí
A. Pomocných funkcí ve standardní knihovně jazyka C (libc)
B. Pomocí instrukce call na adresu uvnitř jádra
C. Vzdáleného volání procedur (RPC)
D. Služeb systémového procesu systemd (Linux) či svchost.exe (windows)

• ♣Běžná implementace funkce malloc (např. v Glibc):
A. Nevyužívá paměť uvolněnou dříve funkcí free
B. Vždy posune konec haldy o velikost požadované paměti
C. Se snaží optimálně využít dostupnou paměť
D. Umožňuje alokovat část paměti na haldě (heap)
E. Používá se pro definici statických proměnných
F. Nic z výše uvedeného není správné

• Rozdíl mezi permisivními a recipročními licencemi je:

A. Reciproční licence chrání uživatele před patentovými hrozbami, kdežto permisivní ne
B. Software pod permisivní licencí můžu šířit bez zdrojových kódů, kdežto software pod reciproční licencí ne
C. Reciproční licence požadují uvedení původního autora, kdežto permisivní licence ne

• ♣Označte pravdivé výroky: Cyklické plánování – round robin
A. Přidělí každému procesu stejné časové kvantum
B. Preferuje krátké procesy
C. Umožňuje rychlé reakce na události v závislosti na předchozí délce běhu programu
D. Neumožňuje stárnutí procesů
E. Nic z výše uvedeného není správně

• Ukazatel rámce (frame pointer) je:
A. Ukazatel na začátek (konec) lokálních proměnných právě vykonávané funkce
B. Ukazatel na začátek kódového segmentu aktuálního programu
C. Ukazatel na začátek stránky paměti, do které uživatelský program právě zapisuje

• UNIXový příkaz strace
A. Vypisuje, které funkce daný program volá
B. Vypisuje všechny řetězce tvořené tisknutelnými znaky nalezené v daném souboru
C. Vypisuje systémová volání provádění daným programem
D. Odstraňuje ze spustitelného souboru ladící informace

• Jak ovladač sdělí hardwaru, že data v paměti může začít zpracovávat (odesílat na síť/zpracovat na disk)?
A. Zasláním přerušení do HW
B. Zápisem do registru daného HW
C. Systémovým voláním ioctl (nebo ekvivalentem v neunixových os)
D. Nijak, HW sám časem zjistí že se v paměti objevila nová data k odeslání

• Souborový systém
A. Je standard definující název a obsah jednotlivých složek (např. program files, /dev)
B. Vždy alokuje data jednoho souboru do souvislé oblasti na pevném disku
C. Je způsob organizování dat na pevném disku
D. Nemůže garantovat konzistenci uložených dat při náhlém vypnutí či pádu počítače
E. Je typ cloudové služby umožňující ukládaní dat (např. amazon S3

• Kritická sekce je část programu:
A. Která se musí vykonávat co nejrychleji
B. Kde se využívají sdílené zdroje
C. Kde dochází ke komunikaci s periférierni
D. Kde může dojít k výjimce

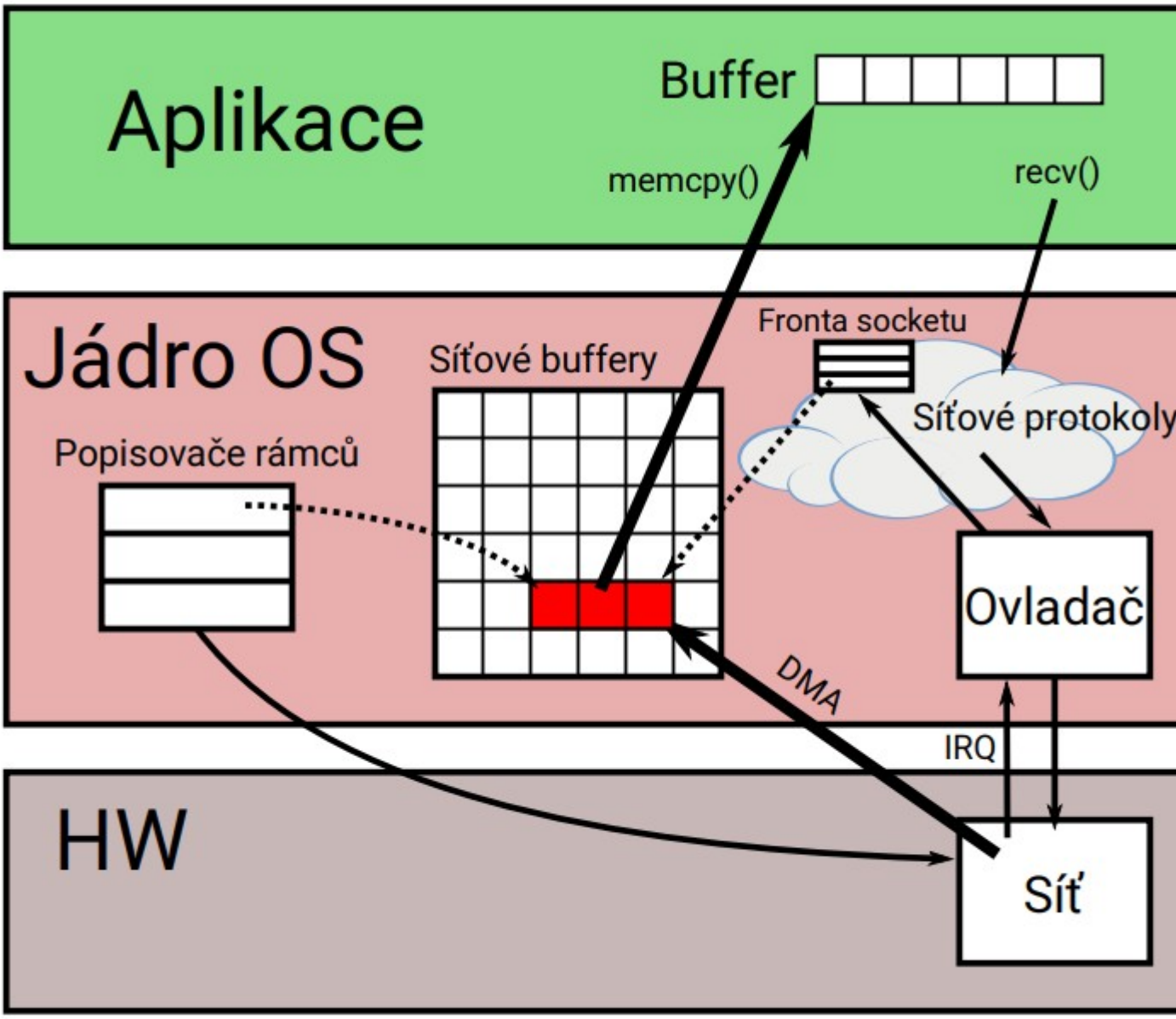
• ♣Mechanismus trap-and-emulate se používá
A. K virtualizaci vstupu a výstupu (např. síťové rozhraní)
B. Jen v systémech bez podpory hardwarové akcelerace virtualizace
C. Pokud virtualizovaný systém provede citlivou instrukci
D. K virtualizaci jednotky správy paměti a stránkovacích tabulek

E. Nic výše uvedeného není správně

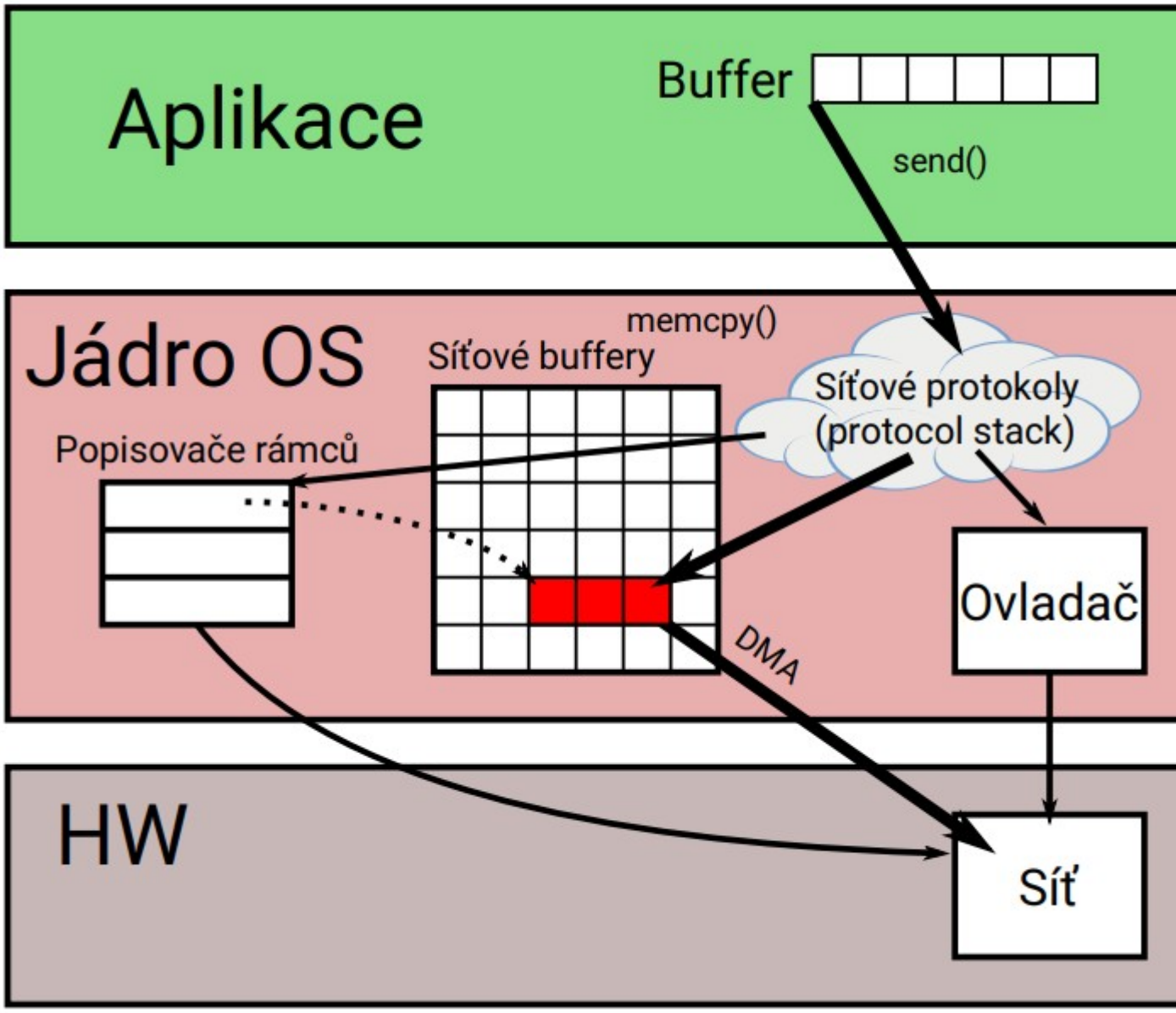
• Jaké způsoby/principy návrhu softwaru vedou ke zvýšení bezpečnosti?
A. Potvrzování všech prováděných operací uživatelem
B. Utajení informací o vnitřním logování programu před uživateli depends
C. Co nejjednodušší návrh
D. Důsledná kontrola vstupů od potenciálně nedůvěryhodných uživatelů
E. Nic z výše uvedeného není správně

Pozor v testu se vyskytuje docela často! (P9_S24&S26):

• Popište, jak probíhá příjem dat ze sítě a jaká je jejich cesta skrze jádro OS až do aplikace:



• Popište, jak probíhá odesílání dat do sítě a jaká je jejich cesta skrze jádro OS až do aplikace?



• Jak probíhá ukládání dat z aplikace na pevný disk. Co se s daty děje a jakou roli v tom hraje JOS?

- Zápis na disk:
  1. Aplikace zavolá write()
  2. JOS data zkopíruje z aplikace do page cache a write() se vrátí.
  3. Čas od času JOS zapisuje „špinavé stránky“ na disk. V Linuxu označováno jako „writeback“.
     - Zápis se dá vynutit systémovým voláním fsync() (Linux)

Pozn.: *fsync()* vs. *fflush()*
fsync() ukládá data z page cache na disk, *fflush()* posílá data z bufferu aplikace (schovaný v libc) do jádra (page cache v případě práce se soubory soubory).

• Jak probíhá načítání dat do aplikace na pevný disk. Co se s daty děje a jakou roli v tom hraje JOS?

Čtení z disku:

1. Aplikace zavolá read()
2. Jádro OS (JOS) přepošle požadavek správnému ovladači
3. Ovladač disku pošle příkaz pro načtení dat a uložení do page cache
4. Disk sám o sobě posílá data do paměti (DMA) a o dokončení informuje přerušením (interrupt request, IRQ)
5. V reakci na IRQ, JOS zkopíruje data z page cache do paměti aplikace

• Co je to Trusted Computing Base?
  A. Softwarové komponenty kritická pro bezpečnost celého systému
  B. Základní operace boolovské algebry umožňující OS kontrolu přístupových práv
  C. Pravidla pro návrh bezpečeného softwaru, zejména OS

• Deadlock – vzájené uváznutí – může vzniknout
  A. Pokud jedno vlákno sdílí více společných zdrojů
  B. Pokud alespoň 2 vlákna sdílejí 1 společný zdroj
  C. Za náhodných okolností, pokud alespoň 2 vlákna sdílejí 2 společné zdroje
  D. Vždy, pokud vlákna sdílejí 4 společné zdroje

• ♣Žurnálování
  A. Zajišťuje konzistenci dat (či metadat) na disku pro případ pádu systému
  B. Zrychluje přístupu k disku pomocí ukládaní dat do vyrovnávací paměti
  C. Využívá souvislou oblast disku k záznamu diskových transakcí
  D. Je záznam informací o dění v OS pro účely bezpečnostního auditu
  E. Nic z výše uvedeného není správně

•♣ Které z následujících tvrzení je pravdivé

A. Semafor lze použít pro synchronizaci paralelně běžících vláken
B. Semafor nemusí pracovat korektně v multiprocesových systémech
C. Semafor je jediný způsob jak zajistit vstup do kritické sekce
D. Semafor řadí čekající vlákna do fronty
E. Semafor zaručuje, že nenastane deadlock – vzájemně uváznutí
F. Nic z výše uvedeného není správně

• ♣Neblokující file desriptor (v unixu s nastaveným příznakem O_NONBLOCK)

A. Způsobuje, že pokud nejsou k dispozici žádná data ke čtení, systémové volání read vrátí chybu
B. Se často používá se systémovými voláními, poll, select či epoll
C. Se používá výhradně ve výkonných síťových serverech
D. Způsobuje, že systémová volání read a write vždy vrátí chybu
E. Nic z výše uvedeného není správně

• Vlákna mezi sebou sdílejí

A. Globální proměnné, dynamicky alokovaná data
B. Globální proměnné, dynamicky alokovaná data a lokální proměnné
C. lokální proměnné a dynamicky alokovaná data
D. lokální proměnné

• Co musí být nastaveno v tabulce stránek, pokud virtuální adresa: 0x22222222 je součástí kódu uživatelského programu a uživatel z ní může číst i jí modifikovat. Dále požadujeme data z této virtuální adresy byla v paměti RAM na adrese 0x55555222

```
enum {
  PRESENT = 1<<0,
  RW      = 1<<1,
  USER    = 1<<2,
  ACCESS  = 1<<5,
  DIRTY   = 1<<6,
};
```

Správné nastavení odpovídající položky v tabulce stránek je:

A. 0x55555003
B. 0x55555222
C. 0x22255555
D. 0x11111222
E. 0x55555007

• Co provádí následující příkaz? tr -d ’/n’

A. Odstraní znaky n ze standardního vstupu
B. Zkopíruje standarní vstup beze změny
C. Odstraní konce řádek ze standardního vstupu
D. Změní všechny znaky d na znak n

• Co provádí následující příkaz? tr -d 't'
A. Změní všechny znaky d na znak t
B. Odstraní znaky t ze standardního vstupu
C. Zkopíruje standardní vstup beze změny
D. Odstraní tabulátory ze standardního vstupu

• ♣Přístup k periferiím v uživatelském prostoru:
A. Není nikdy možný
B. Je možný s využitím služby jádra
C. Je možný s využitím mapování do paměti procesoru
D. Je možný s využitím instrukce test
E. Nic výše uvedeného není správně

• Takzvané init skripty jsou
A. Skripty spouštěné po zalogování uživatele, které nastavují například rozložení klávesnice
B. Skripty spouštěné některými init systémy pro spouštění a ukončování systémových služeb
C. Skripty, které zjednodušují spouštění komplexních aplikací (většinou v jazyce java, např. Eclipse, matlab apod.)

•♣ Která následující tvrzení jsou pravdivá
A. Semafor lze použít pro synchronizaci paralelně běžících vláken
B. Při použití semaforu vždy hrozí deadlock
C. Semafor lze použít pro správný přístup ke sdíleným datům
D. Semafor zaručuje, že nenastane deadlock – vzájemné uváznutí
E. Semafor zaručuje spravedlivé čekání na sdílený zdroj
F. Nic z výše uvedeného je pravdivé

• Jaké jsou vlastnosti ovladačů zařízení implementovaných v uživatelském prostoru?
A. Nelze je používat v systémech s monolitickými jádry
B. Typicky jsou výrazně pomalejší než ovladače v privilegovaném režimu
C. Při chybě v ovladači nehrozí pád celého systému
D. Při aktualizaci takového ovladače je potřeba restartovat celý systém

•♣ Útoku na systém pomocí přetečení zásobníku se dá zabránit
A. Použitím programovacího jazyk, který přetečení automaticky kontroluje
B. Odpojením počítače ze sítě
C. Randomizaci adresního prostoru (ASLR)
D. Použitím antivirového programu
E. Zakázáním vykonávání kódu v paměti, kde je zásobník, pomocí atributů stránek
F. Nic z výše uvedeného není správně

•♣ Která tvrzení jsou pravdivá?
A. Deadlocku se není možné vyhnout vhodným plánováním
B. Deadlock je možné eliminovat, pokud se nebudou používat semafory
C. Deadlocku je možné se vyhnout vhodným plánováním, pokud jsou známy

požadavky vláken dopředu

D. Deadlocku není možné zabránit a může se vyskytnout, kdykoliv se používají sdílené zdroje

E. Deadlock se může detekovat, pokud se bude monitorovat čekání na vstup do kritické sekce

F. Nic z výše uvedeného není správné

Otázka 9 Uvažujte následující příkaz BASHe:

```
l=`ls -l`
```

Který z následujících příkazů vytiskne proměnnou l v původním formátu příkazu ls?

- ☐ echo ${l}
- ☐ echo ($l)
- ☐ echo $l
- ☐ echo '$l'
- ☒ echo "$l"

Otázka 10 Mějme následující program:

```
int main() {
  int f=fork();
  if (f==0) {
    int a=open("a.txt", O_WRONLY | O_CREAT);
    printf("Start\n");
    dup2(a,1);
    printf("Exec\n");
    execl("./prg", "./prg", NULL);
  } else {
    wait(NULL);
  }
  return 0;
}
```

Pokud proběhne program bez chyb pak:

- ☐ hláška "Start" a "Exec" se vytiskne na standardní výstup a program "prg" bude mít vstup také ze standardního vstupu
- ☐ hláška "Start" se vytiskne na standardní výstup, hláška "Exec" bude zapsána do souboru "a.txt" a program "prg" bude mít výstup přesměrován na standardní výstup rodiče
- ☐ hláška "Start" a "Exec" se vytiskne na standardní výstup, a program "prg" bude mít přesměrován výstup do souboru "a.txt"
- ☐ hláška "Start" a "Exec" se vytiskne do souboru "a.txt" a program "prg" bude mít přesměrován výstup do souboru "a.txt"
- ☒ hláška "Start" se vytiskne na standardní výstup, hláška "Exec" bude zapsána do souboru "a.txt" a program "prg" bude mít přesměrován výstup do souboru "a.txt"
- ☐ hláška "Start" se vytiskne na standardní výstup, hláška "Exec" se nevytiskne a program "prg" bude mít výstup do souboru "a.txt"

• ♣Licence svobodného software dovolují každému uživateli:

A. Změnit licence software podle potřeby

B. Prodávat software za peníze
C. Software spouštět, studovat, šířit a vylepšovat
D. Dělat si se software cokoliv bez jakýchkoliv omezení
E. Nic z výše uvedeného není správně

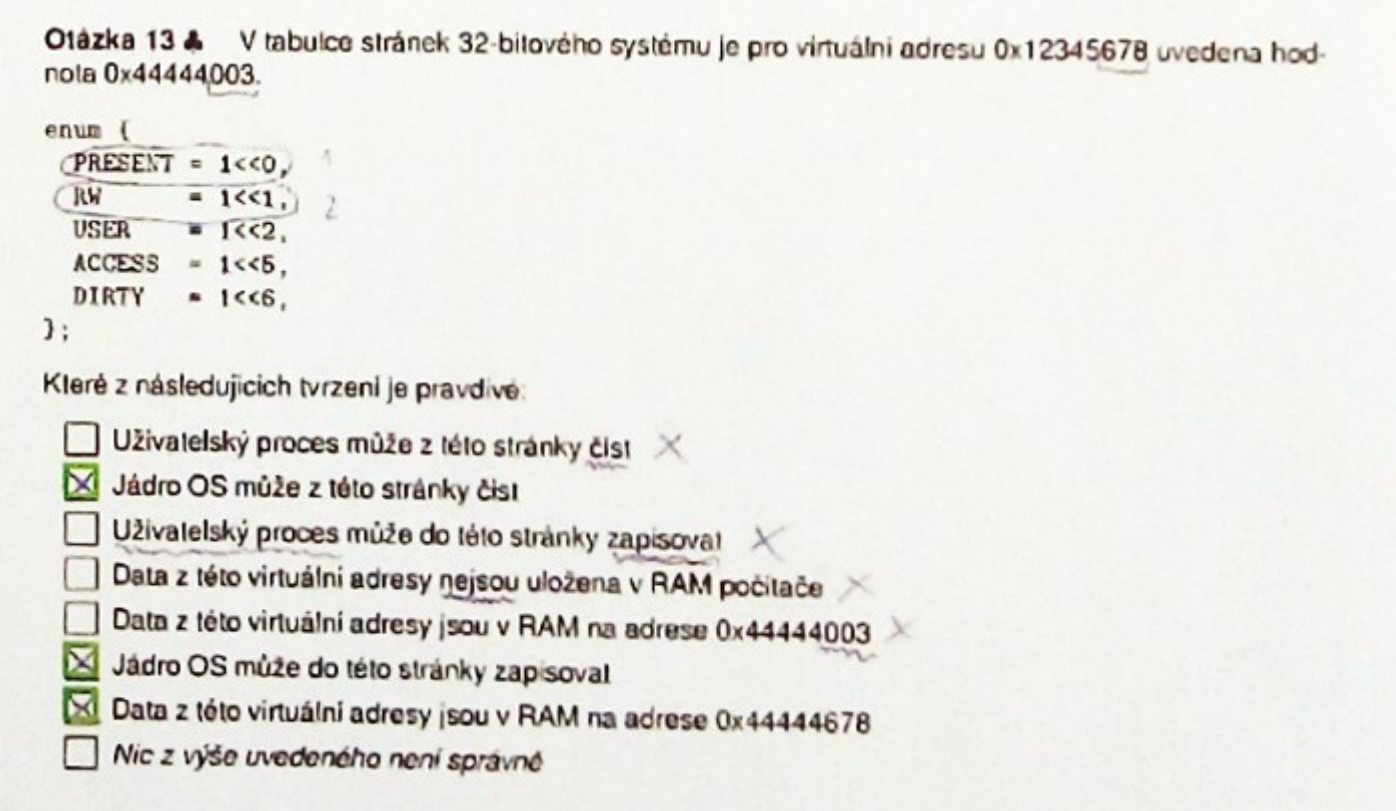
• ♣K zajištění konzistence souborového systému po náhlém vypnutí či pádu systému
A. Je potřeba používat disková pole, kde jsou data uložena minimálně na dvou místech
B. Lze použít metodu, kdy se popis potřebných modifikací nejprve uloží do speciální oblasti na disku
C. Je vždy potřeba kompletní kontrola souborového systému po následném zapnutí počítače
D. Je potřeba vybavit počítač záložním zdrojem (UPS)
E. Je potřeba provádět všechny související modifikace souborového systému v definovaném pořadí
F. Nic z výše uvedeného není správně

Otázka 6 ♣ V tabulce stránek 32bitového sy
hodnota 0xCCCCC005.

```
enum {
   PRESENT = 1<<0,
   RW      = 1<<1,
   USER    = 1<<2,
   ACCESS  = 1<<5,
   DIRTY   = 1<<6,
};
```

Které z následujících tvrzení je pravdivé:

- [x] Jádro OS může z této stránky číst
- [ ] Uživatelský proces může do této stránky
- [ ] Data z této virtuální adresy jsou v RAM n
- [x] Data z této virtuální adresy jsou v RAM n
- [x] Uživatelský proces může z této stránky č
- [ ] Data z této virtuální adresy jsou v RAM n
- [ ] *Nic z výše uvedeného není správně*



Otázka 13 ♣ V tabulce stránek 32-bitového systému je pro virtuální adresu 0x12345678 uvedena hodnota 0x44444003.

```
enum {
  PRESENT = 1<<0,
  RW      = 1<<1,
  USER    = 1<<2,
  ACCESS  = 1<<5,
  DIRTY   = 1<<6,
};
```

Které z následujících tvrzení je pravdivé:

- ☐ Uživatelský proces může z této stránky číst
- ☒ Jádro OS může z této stránky číst
- ☐ Uživatelský proces může do této stránky zapisovat
- ☐ Data z této virtuální adresy nejsou uložena v RAM počítače
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0x44444003
- ☒ Jádro OS může do této stránky zapisovat
- ☒ Data z této virtuální adresy jsou v RAM na adrese 0x44444678
- ☐ *Nic z výše uvedeného není správně*

Otázka 11 Uvažujte následující příkaz BASHe:

```
l=`ps`
```

Který z následujících příkazů vytiskne na každou řádku jeden p

☐ echo ${l} ☐ echo ($l) ☐ echo "$l"

• ♣Co je to výpadek stránky (page fault)

A. Kritická chyba způsobující havárii jádra OS
B. Výjimka CPU, podobně jako například dělení nulou
C. Ztráta dat v paměti, programy si musí ztracená data obnovit ze zálohy
D. Reakce CPU na situaci, kdy se nějaká instrukce pokouší přistoupit k paměti kde nemá přístup
E. Nic z výše uvedeného není správně

• ♣Neblokující I/O

A. Dovoluje vláknu dělat něco užitečného, když nejsou k dispozici data, na která čeká
B. Garantuje, že se vlákno nikdy nezablokuje a výkon cpu se tak využije na 100%
C. Je možné používat jen u počítačů se SSD diskem

D. Může zabránit dlouhému blokování hlavního vlákna v grafických aplikacích
E. Nic z výše uvedeného není správně

•♣ Software šířený pod licencí GPLv2:
A. není možné prodávat za peníze
B. nesmí se používat s komerčními OS (např. MS Windows)
C. musí být šířen společně se zdrojovými kódy
D. je možné použít v proprietárním SW, ale pouze po odstranění všech hlášek ”copyright”
E. Nic z výše uvedeného není správně

•♣ Která následující tvrzení jsou pravdivá?
A. Při použití mutexu může vždy nastat deadlock – vzájemné uváznutí
B. Mutex lze použít pro správný přístup ke sdíleným datům
C. Mutex zaručuje, že nenastane deadlock – vzájemné uváznutí
D. Mutex lze použít pro synchronizaci paralelně běžících vláken
E. Nic z výše uvedeného není správně

• NEBUDE! Bylo jen pro rok 2018. ♣V roce 2018 zveřejněná zranitelnost CPU zvaná Meltdown:
A. způsobí, že při vykonání škodlivého kódu se procesor přehřeje a dojde k jeho zničení
B. je to chyba HW a OS s tím nemůže nic dělat
C. umožňuje číst data i ze stránek, ke kterým nemá uživatelský proces přístup
D. je jen další mediální bublinou
E. vyskytuje se pouze u procesorů od Intelu
F. Nic z výše uvedeného není správně

•♣ Stárnutí (starvation) je problémem, který hrozí plánovacím algoritmům. Které algoritmy ohrožuje stárnutí:
A. cyklické plánování – round robin
B. prioritní plánování
C. First Come First Served – podle pořadí spuštění
D. nejkratší proces první – shortest process next
E. Nic z výše uvedeného není správně

Otázka 4 Mějme následující program:

```
int a[2];
pthread_mutex_t mutex[2];

void *fce1(void *n) {
    int num=*(int*)n;
    for (int i = 0; i < 150; i++) {
        pthread_mutex_lock(&mutex[0]);
        pthread_mutex_lock(&mutex[1]);
        a[num] += a[1-num];
        pthread_mutex_unlock(&mutex[1]);
        pthread_mutex_unlock(&mutex[0]);
    }
    pthread_exit(NULL);
}

void *fce2(void *n) {
    int num=*(int*)n;
    for (int i = 0; i < 150; i++) {
        pthread_mutex_lock(&mutex[1]);
        pthread_mutex_lock(&mutex[0]);
        a[num] += a[1-num];
        pthread_mutex_unlock(&mutex[0]);
        pthread_mutex_unlock(&mutex[1]);
    }
    pthread_exit(NULL);
}

int main()
{
    pthread_t tid[2];
    a[0]=0; a[1]=1;
    pthread_mutex_init(&mutex[0], NULL);
    pthread_mutex_init(&mutex[1], NULL);
    pthread_create(&tid[0], NULL, fce1, NULL);
    pthread_create(&tid[1], NULL, fce2, NULL);
    pthread_join(tid[0], NULL);
    pthread_join(tid[1], NULL);
    return 0;
}
```

- [ ] program vždy skončí chybou
- [ ] program někdy skončí, někdy uvázne v deadlocku
- [ ] program vždy uvázne v deadlocku
- [ ] program vždy bez problémů skončí
- [ ] program neskončí, obsahuje nekonečnou smyčku

Otázka 7 Předpokládejte, že nedojde k chybě při spuštění r

```
int main() {
  int f=fork();
  f=fork();
  f=fork();
  printf("%d\n", f);
  return 0;
}
```

Jaký bude výstup tohoto programu?

- ☐ osm nenulových čísel
- ☐ šest nenulových čísel a dvě 0
- ☐ dvě nenulová čísla a dvě 0
- ☐ čtyři nenulová čísla a čtyři 0
- ☐ čtyři
- ☐ tři n
- ☐ osm

• ♣Hardwarově asistovaná virtualizace (např. VT-x)

A. zrychluje běh virtuálního uživatelského režimu.

B. odstraňuje nutnost emulovat hardware (disk, síťová karta, ...) mechanismem trap-and-emulate.

C. funguje jedině na 32bitovém systému.

D. zrychluje vykonávání některých privilegovaných instrukcí.

E. přidává další vrstvu stránkovacích tabulek.

F. Nic z výše uvedeného není správně

•♣ Jaký je vztah pojmů „page fault“ (výpadek stránky) a „segmentation fault“

A. Segmentation fault běžně nastává během běhu většiny programů

B. Page fault běžně nastává během běhu většiny programů

C. Segmentation fault je vždy důsledek page fault

D. Dva různé pojmy pro stejnou věc

E. Page fault je vždy důsledek segmentation fault

F. Page fault se používá k implementaci copy-on-write

G. Nic z výše uvedeného není správně

/pokud nastal seg fault, tak musel nastat page fault/

• Vlákna na jedno-procesorovém počítači (bez hyper-threadingu):
A. mohou běžet paralelně pokud nikdy nepoužívají sdílené proměnné
B. nemohou běžet paralelně nikdy
C. mohou běžet paralelně pouze pokud použijí mutex
D. mohou běžet paralelně

• Provedení funkce z jádra operačního systému na architektuře x86 lze z uživatelského programu vyvolat:
A. instrukcí int s registrem obsahujícím číslo služby jádra
B. instrukcí nop s registrem obsahujícím číslo služby jádra
C. instrukcí call na adresu služby jádra
D. instrukcí jmp na adresu služby jádra

• Return oriented programming je
A. Druh útoku na http protokol, který zneužívá hlavičky vrácené serverem
B. Způsob, jak spustit kód na napadaném systému i když nemá spustitelný zásobník
C. Bezpečný způsob programování bez použití příkazu goto

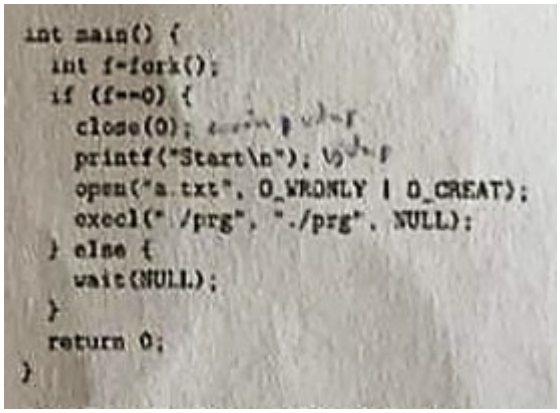
•♣ Smyčka událostí
A. Může pro obsluhu události vytvořit nové vlákno
B. Používá se pouze v aplikacích s jedním vláknem
C. Je typicky součástí grafických a serverových aplikací
D. Se používá pouze v grafických aplikacích psaných ve vyšších programovacích jazycích (java,Csharp)
E. Slouží v aplikacích typu kalendář k upozorňování uživatelů na blížící se události
F. Nic z výše uvedeného není správně

• Službu jádra lze na architektuře x86 spustit
A. Instrukcí out s registrem obsahujícím číslo služby jádra
B. Instrukcí int s registrem obsahujícím číslo služby jádra
C. Instrukcí call na adresu služby jádra
D. Instrukcí jmp na adresu služby jádra

• ♣Souborové systémy, které používají tzv. extents
A. Nejsou vhodné pro nasazení na serverech
B. Potřebují méně místa k uložení informací o umístění dat daného souboru
C. Jsou velmi efektivní, pokud data souborů nejsou fragmentovaná
D. Potřebují používat nepřímé bloky odkazů na data
E. Nesmí obsahovat soubory větší než 2GiB
F. Nic z výše uvedeného není správně

•♣ Která následující tvrzení jsou pravdivá
A. Při použití semaforu vždy hrozí deadlock
B. Semafor lze použít pro zajištění správného přístupu ke sdíleným datům
C. Semafor zaručuje, že všechna čekající vlákna postupně získají sdílený zdroj (=vlákna se nepředbíhají)
D. Semafor zaručuje, že nenastane deadlock – vzájemné uváznutí

E. Semafor lze použít pro synchronizaci paralelně běžících vláken
F. Nic z výše uvedeného je pravdivé



```
int main() {
  int f=fork();
  if (f==0) {
    close(0);
    printf("Start\n");
    open("a.txt", O_WRONLY | O_CREAT);
    execl("./prg", "./prg", NULL);
  } else {
    wait(NULL);
  }
  return 0;
}
```

Pokud proběhne program bez chyb pak:

- [ ] hláška "Start" se vytiskne na standardní výstup a program "prg" bude mít vstup také ze standardního vstupu
- [ ] hláška "Start" se vytiskne do souboru "a.txt" a program "prg" bude mít přesměrován výstup do souboru "a.txt"
- [ ] hláška "Start" se nevytiskne a program "prg" bude mít vstup ze souboru "a.txt"
- [x] hláška "Start" se vytiskne na standardní výstup a program "prg" bude mít přesměrován vstup ze souboru "a.txt"
- [ ] hláška "Start" se vytiskne do souboru "a.txt" a program "prg" bude vystupovat na standardní výstup
- [ ] hláška "Start" se nevytiskne a program "prg" bude mít přesměrován výstup do souboru "a.txt"

• Uvažujte nasledující príkaz BASHe: P=”HELLO WORLD!”

Který příkaz vypíše proměnou P s 3 mezerami mezi slovami HELLO a WORLD!

A. echo ($P)
B. echo $(P)
C. echo “$P”
D. echo ‘$P’
E. echo $P

• Předpokládejte, že nedojde k chybě při spouštění následujícího programu

```
int main(){
      int f=fork();
      f=fork();
      printf(“%d\n”,f);
      return 0;
}
```

Jaký bude výstup tohoto programu?

A. jedno nenulové číslo a dvě 0
B. tři nenulová čísla a jedna 0
C. jedno nenulové číslo
D. dvě nenulová čísla a dvě 0

• Zneužití chyby přetečení zásobníku se dá zabránit tím, že kompilátor vygeneruje kód, který

A. Nahradí všechny návraty z funkcí systémového voláním “ret“, které útočník nemůže ovlivnit

B. Po vstupu do funkce uloží návratovou hodnotu do záložní proměnné

C. Vždy před návratem z funkce zkontroluje integritu návratové adresy

• Seznamy pro řízení přístupu (tzv. ACL)

A. Se používají pro určení, zda má být danému uživateli umožněn nebo zamítnut přístup k souboru

B. Slouží jako vstupní data při vytváření stránkovacích tabulek nových procesů

C. Obsahují jména a hesla uživatelů, kteří se smí připojit k danému počítači

• Služby jádra se na architektuře x86 rozlišují

A. Řetězcem obsahujícím název služby

B. Adresou, na které je služba umístěna v paměti

C. Návratovou hodnotou uloženou na zásobníku

D. Číslem definujícím typ služby

• Proč se nesmí volat některé funkce (např. printf) v obslužné rutině signálu (handleru)?

A. Protože handler může být volán uprostřed již probíhající funkce printf a její opětovné volání z handleru by pak změnilo hodnoty statických proměnných touto funkcí používaných, což by vedlo k pádu programu.

B. Protože funkce jako printf používají signály interně ke komunikaci s jádrem OS a jejich volání v handleru by tak vedlo k nekonečné rekurzi a následnému pádu programu kvůli nedostatku paměti.

C. Protože handlery signálů jsou vykonávány v kontextu speciálního procesu, který nemá přístup ke standardnímu vstupu a výstupu původního procesu.

• ♣Která tvrzení jsou pravdivá?

A. Podmínkové proměnné umožňují vláknům čekat na změnu sdílených dat

B. Čekání na podmínkovou proměnnou lze i bez předchozího zamknutí mutexu

C. Při použití podmínkové proměnné vždy hrozí deadlock

D. Při čekání na podmínkovou proměnnou se uvolní propojený mutex

E. Podmínková proměnná nemá žádný vztah k mutexu

F. Nic z výše uvedeného není správně

• ♣Virtualizace celého systému

A. Vyžaduje CPU s podporou dvouvrstvého stránkování

B. Je možná, jen pokud všechny citlivé instrukce použitého CPU způsobují výjimku

C. Vykonává veškerý kód jádra virtuálního stroje v uživatelském režimu

D. Je možná, jen pokud jsou všechny instrukce použitého CPU privilegované

E. Nic z výše uvedeného není správně

Otázka 1 Níže je vypsána část informací o určitém souborovém systému. Zjistěte, ko
možné na souborový systém ještě uložit a kolik je možné na něm vytvořit souborů či adresá

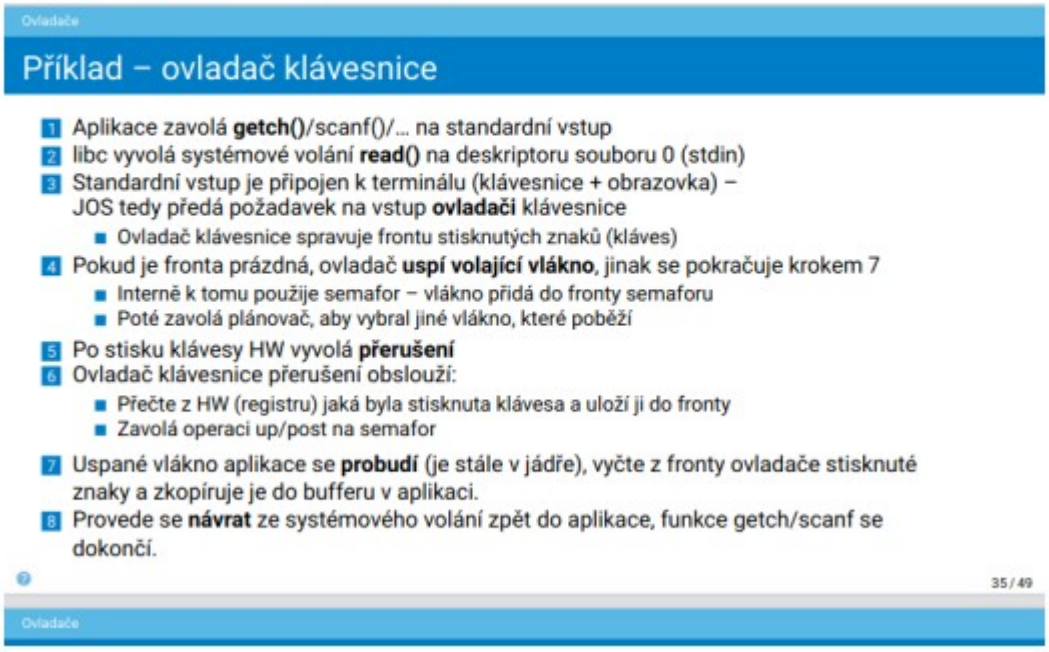
```
Filesystem UUID:          f1020f7d-e2af-41df-a5b9-5fcbf824f92f
Filesystem magic number:  0xEF53
Filesystem revision #:    1 (dynamic)
Filesystem features:      has_journal ext_attr dir_index needs_recovery ext
Default mount options:    user_xattr acl
Filesystem state:         clean
Errors behavior:          Continue
Filesystem OS type:       Linux
Inode count:              5242880
Block count:              20971520
Reserved block count:     1048576
Free blocks:              5226593
Free inodes:              5161664
First block:              0
Block size:               4096
Fragment size:            4096
Blocks per group:         32768
Fragments per group:      32768
Inodes per group:         8192
Inode blocks per group:   512
Mount count:              44
First inode:              11
Inode size:               256
Journal inode:            8
Default directory hash:   half_md4
Journal size:             128M
Journal length:           32768
Journal sequence:         0x000002ba
Journal start:            0


Group 0: (Blocks 0-32767) csum 0x0143 [ITABLE_ZEROED]
  Block bitmap at 1025 (+1025)
  Inode bitmap at 1041 (+1041)
  Inode table at 1057-1568 (+1057)
  23513 free blocks, 8181 free inodes, 2 directories, 8181 unused inodes
  Free blocks: 9255-32767
  Free inodes: 12-8192
Group 1: (Blocks 32768-65535) csum 0x7f9a [INODE_UNINIT, ITABLE_ZEROED]
  Block bitmap at 1026 (bg #0 + 1026)
  Inode bitmap at 1042 (bg #0 + 1042)
```

- Vyberte pravdivé tvrzení:
  - A. Různá vlákna sdílejí registry procesoru
  - B. Vlákna sdílejí návratové hodnoty z funkcí.
  - C. Vlákna sdílejí parametry funkcí.
  - D. Vlákna sdílejí proměnné deklarované s příznakem static.
- Jak do proměnné A přiřadíte výstup příkazu ls -l
  - A. A=$(ls -l)
  - B. A=(ls -l)
  - C. A="ls -l"
  - D. A=ls -l
  - E. A=bash ls -l
- ♣Které z následujících licencí označujeme jako permisivní?
  - A. MPL
  - B. GPL
  - C. MIT
  - D. Apache
  - E. LGPL
  - F. BSD
- ♣ Které z následujících licencí označujeme jako reciproční s omezeným působením?
  - A. MPL
  - B. GPL
  - C. MIT
  - D. Apache
  - E. LGPL
  - F. BSD
- Jak program získá data z klávesnice?

Další otázky OSY:

1. Jak do proměnné A přiřadíte výstup příkazu: ls -l
A. A=$(ls -l)
B. A=(ls -l)
C. A="ls -l"
D. A=ls -l
E. A=bash ls -l

2. Které z následujících licencí označujeme jako permisivní
A. MPL
B. GPL
C. MIT
D. Apache
E. LGPL
F. BSD

3. Které z následujících licencí označujeme jako reciproční s omezeným působením
A. MPL
B. GPL
C. MIT
D. Apache
E. LGPL
F. BSD

4. Jak program získá data z klávesnice?