Afdeling Informatica	**Bedrijfssystemen**
Vrije Universiteit	**28.03.2007**

Dit examen beslaat twee pagina's

1a MINIX3 volgt het **client-server model** voor haar structuur. Leg uit wat dit model inhoudt en ga daarbij vooral in op de rol van de kernel. *5pt*

1b Drivers in MINIX3 worden geëxecuteerd als gewone *user-space* processen. In welk opzicht legt dit beperkingen op om met hardware controllers to communiceren, en hoe wordt dit opgelost? *5pt*

1c Schets de *control flow* wanneer een *user-space* proces een bibliotheekfunctie read(fd,buffer,bytes) in een monolithisch besturingssysteem aanroept. Hint: geef het eea weer in een diagram. *5pt*

1d In veel gevallen biedt de hardware ondersteuning voor meerdere ringen van protectie voor programma's. Hoe kunnen we gebruik maken van deze faciliteit voor het inrichten van een besturingssysteem? *5pt*

2a De semantiek van de *atomaire* **swap** machine instructie is als volgt gedefinieerd. Laat zien hoe deze instructie gebruikt kan worden om een kritieke sectie te beschermen. *5pt*

swap(**inout** boolean a, **inout** boolean b){ temp = a; a = b; b = temp;}

2b Beschouw het programma op pagina 2, dat als een apart *user-space* MINIX3 **lock manager** proces geëxecuteerd dient te worden. De kern van het programma wordt gevormd door de functies do_down() en do_up() die de standaardoperaties op semaforen implementeren. Gegeven lock_manager(), geef een pseudo-code implementatie van de functie do_down(sema). *5pt*

2c Geef ook een schets van de implementatie van do_up(). *5pt*

2d Het retourneren van SUSPEND door do_down() heeft tot resultaat dat de executie van een proces opgeschort wordt. Welk proces is dat en hoe wordt deze blokkering feitelijk tot stand gebracht? *5pt*

3a Leg uit hoe MINIX3 (en veel andere besturingssystemen) meerdere *timers* simuleren gebruikmakend van een enkele klok. Teken een figuur om je antwoord uit te leggen. *5pt*

3b Leg het verschil uit tussen **character devices** en **block devices**, en waarom het maken van een dergelijk onderscheid nuttig kan zijn voor het verbeteren van I/O prestaties. *Hint: denk eens aan het schrijven van een serie bytes naar disk.* *5pt*

4a Leg het principe uit van de fork() system call. *5pt*

4b **Copy-on-write** is een techniek waarbij een blok geheugen pas gevuld wordt met data van een specifieke bron op het moment dat er voor het eerst naar geschreven wordt. Hoe kan deze techniek gebruikt worden voor het optimaliseren van de implementatie van fork()? *Wees precies!* *5pt*

5a Leg uit wat de mount() system call doet door het geven van een voorbeeld. *Leg je voorbeeld uit!* *5pt*

5b Mount() verandert velden in inodes en de kopiën van superblocks zoals opgeslagen in hoofdgeheugen. Leg uit wat er verandert wordt. *5pt*

5c Beschouw de volgende operaties die uitgevoerd worden op een geformatteerde maar verder lege USB stick. Geef het resultaat wanneer de inhoud van de folder test weergegeven wordt (d.m.v. de laatste operatie ls). *5pt*

```
mount /dev/sdb1 /usbstick   Mount the USB stick
cd /usbstick                Enter the directory
mkdir test                  Create a subdirectory ''test.''
touch test/x                Create a file ''x'' in ''test.''
mount /dev/sdb1 test        Mount the USB stick again
ls test                     List the directory contents
```

5d Leg nu precies uit wat er gebeurt is met de *superblock* tabel en de *inode* tabel als de twee mount operaties uit het vorige voorbeeld uitgevoerd zijn. *5pt*

6a Wat wordt bedoeld met een protectie domein? *5pt*

6b Geef een practisch voorbeeld van hoe je van éé n protectie domein naar een ander domein kunt omschakelen, en leg uit hoe een dergelijke transitie door een besturingssysteem geïmplementeerd kan worden. *5pt*

```
01 PUBLIC int lock_manager(){
02   int result, s, proc_nr;
03   struct mproc *rmp;
04   while (TRUE) {
05     receive(ANY, &msg_in);
06     who = msg_in.m_source;          /* who sent the message          */
07     sema = msg_in.m5_l1;            /* which semaphore is this?      */
08     call_request = msg_in.m5_i1;    /* which operation is requested? */
09     mp = &mproc[who];
10     switch(call_request){
11       DOWN: result = do_down(sema); break;
12       UP:   result = do_up(sema); break;
13     }
14
15     /* Send the results back to the user to indicate completion. */
16     if (result != SUSPEND) setreply(who, result); /* Prepare  reply message */
17     /* Send out all pending reply messages, including the answer to
18      * the call just made above.
19      */
21     for (proc_nr = 0, rmp = mproc; proc_nr < NR_PROCS; proc_nr++, rmp++) {
22       if ((rmp->mp_flags & REPLY) == REPLY ){
23         send(proc_nr, &rmp->mp_reply);
24         rmp->mp_flags &= ~REPLY;
25       }
26     }
27   }
28   return(OK);
29 }
```

Cijferbepaling: *Het cijfer wordt bepaald door de punten die per onderdeel behaald zijn, bij elkaar op te tellen (totaal maximaal 90 punten), en vervolgens daar 10 extra punten bij te tellen. Er zijn dus totaal 100 punten te behalen.*