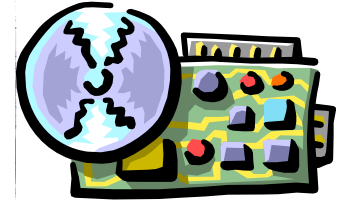




TI III: Operating and Communication Systems



WS 2007/08
Übungsblatt Nr. 6

Georg Wittenburg, M.Sc., AG Technische Informatik, Freie Universität Berlin

Ausgabe am 11.1.2008 — Abgabe spätestens 25.1.2008, 10:00 Uhr

Bitte bei der Abgabe beide Namen/Matr.Nr. der Mitglieder einer Gruppe, NUMMER DER ÜBUNG/TEILAUFGABE und DATUM auf den Lösungsblättern **nicht vergessen!** Darauf achten, dass die Lösungen beim richtigen Tutor/der richtigen Tutorin abgegeben werden.
Achten Sie bei Programmieraufgaben außerdem darauf, dass diese im Linuxpool kompilierbar sind.
Zu spät abgegebene Lösungen werden nicht mehr berücksichtigt!

1. Aufgabe: Begriffe (8 Punkte)

Beschreiben Sie jeden der folgenden Begriffe durch maximal zwei Sätze: Polling, Star Topology, Open Shortest Path First, Backward Learning, Bridge, Gateway, VLAN, ARP

2. Aufgabe: Dienste (4 Punkte)

Geben Sie vier Dienste der Sicherungsschicht (Schicht 2) an, und erläutern Sie in wenigen Stichpunkten deren Aufgabe und Funktionsweise.

3. Aufgabe: Sliding Window Algorithmus (6 Punkte)

Der Sliding-Window-Algorithmus wird für die Flusssteuerung und sichere Datenübertragung eingesetzt. Bei der Kommunikation verwaltet der Sender eine Senderfenstergröße (Send Window Size, SWS) und der Empfänger ein Empfängerfenstergröße (Receive Window Size, RWS).

Zeichnen Sie ein Zeitstrahldiagramm für den Sliding-Window-Algorithmus mit $SWS=RWS=3$ Frames für die beiden folgenden Situationen. Verwenden Sie ein Timeout-Intervall von $2 \times RTT$ (Round-Trip-Time).

- a) Es wird 6 Frames gesendet und Frame 4 geht verloren.
- b) Es wird 7 Frames gesendet und Frames 4 bis 6 gehen verloren.

Kennzeichnen Sie im Diagramm, an welchen Stellen die verloren gegangenen Frames erneuert gesendet werden.

4. Aufgabe: Fifty/Fifty (8 Punkte)

Entscheiden Sie, ob folgenden Aussagen zutreffend oder nicht zutreffend sind, und begründen Sie Ihre Entscheidung:

- Wenn ein IP-Paket beim Router des Heimnetzes des Empfängers ankommt, so sendet der Router dieses mit einem Broadcast ins LAN, wo es alle angeschlossene Rechner auf die zutreffende IP-Adresse hin untersuchen.
- Ob in einem konkreten Protokoll Vorwärtsfehlerkorrektur zum Einsatz kommt, hängt auch vom verwendeten physikalischen Medium auf der unteren Schicht ab.
- Gerade bei niedriger Auslastung kann auf den unnötigen Overhead der Prüfsumme im Paketkopf verzichtet werden.
- Bursty Traffic kann über die vielen Kanäle des Multiplexing gut entgegengewirkt werden.

5. Aufgabe: Scheduler II (14 Punkte)

Verwenden Sie den CPU-Emulator QEMU (<http://fabrice.bellard.free.fr/qemu/>, bzw. für Windows <http://www.h7.dion.ne.jp/~qemu-win/>), um das auf der Homepage der Veranstaltung verlinkte, in einer Zipdatei hinterlegte Festplatten-Image (ca. 400 MB) zu starten. Melden Sie sich im emulierten System mit dem Benutzernamen „**root**“ und dem Passwort „**fuberlin**“ an.

Wechseln Sie in das Verzeichnis, das den Quellcode des Linux-Kernels enthält:

```
# cd /usr/src/linux-source-2.6.18
```

Öffnen Sie die Datei `kernel/sched.c` mit dem Editor **nano** (Dokumentation unter <http://www.nano-editor.org/>):

```
# nano kernel/sched.c
```

Passen Sie den Scheduler des Linux-Kernels so an, dass immer, wenn das aktive und das abgelaufene Prioritätsarray zum 100 Mal vertauscht werden, mit der Funktion **printk()** eine Nachricht auf der Konsole ausgegeben wird.

Kompilieren Sie Ihren modifizierten Kernel (ggf. auf Fehlermeldungen achten und Implementierung korrigieren):

```
# make
```

Installieren Sie den neuen Kernel im emulierten System:

```
# make install
# make modules_install
# update-initramfs -k 2.6.18 -c
# ln -s /boot/initrd.img-2.6.18 /boot/initrd.img
# update-grub
```

Starten Sie nun das System neu:

```
# reboot
```

Achten Sie im Boot-Menü darauf, Ihren neu kompilierten Kernel auszuwählen. Booten Sie mit dem mitgelieferten Kernel, falls es zu Problemen kommt und beheben Sie diese. Dokumentieren Sie die Funktionsweise des modifizierten Kernels. Fahren Sie abschließend das System herunter:

```
# halt
```

6. Aufgabe: Webserver II (10 Punkte)

Testen Sie Ihre Implementierung eines einfachen Webserver aus dem vorhergehenden Übungsblatt auf sicherheitsrelevante Schwachstellen:

- Variieren Sie die Formatierung der HTTP GET-Anfrage, und fügen Sie geeignete ASCII-Kontrollzeichen in die Anfrage ein.
- Fragen Sie nicht vorhandene Seiten an, und versuchen Sie auf sonstige Dateien auf dem Server-Rechner zuzureifen (z.B. durch die Verwendung der Zeichenkette „**../dateiname**“ für das nächst höhere Verzeichnis).
- Versuchen Sie, die in der Implementierung des Servers allozierten Arrays durch zu lange Zeichenketten in Ihrer HTTP GET-Anfrage zum Überlauf zu bringen.

Dokumentieren Sie Ihre Anfragen (mindestens drei unterschiedliche pro Stichpunkt) und die Reaktion Ihres Webserver. Finden und verbessern Sie die Schwachstellen in Ihrer Implementierung und dokumentieren Sie, welche typischen Fehlerquellen vorlagen. Sollte Ihre Implementierung weniger als drei Fehler enthalten, dann erläutern Sie am Quelltext, wie Sie Ihre Implementierung bezüglich der drei oben genannten Stichpunkte abgesichert haben.