Technische Universität München
Lehrstuhl Informatik VIII
Prof. Dr.-Ing. Georg Carle
Dipl.-Ing. Stephan Günther, M.Sc.
Johannes Naab, M.Sc.

Programmieraufgaben zur Vorlesung Grundlagen Rechnernetze und Verteilte Systeme

Programmieraufgabe 1 – Wake on LAN (2 Punkte)
(28. April – 10. Mai 2015)

Abgabe via SVN bis Sonntag, 10. Mai 2015, 23:59:59.99 Uhr (MESZ)

1 Übersicht

Wake-On-LAN (WoL) Wake-On-Lan ist ein Netzwerkstandard der es ermöglicht ausgeschaltete oder im Ruhezustand befindliche Geräte durch Senden eines speziellen Rahmens einzuschalten. Ziel dieser Programmieraufgabe ist es, ein Wake-On-LAN-Paket manuell zu konstruieren und zu versenden. Dabei werden Sie Erfahrung im Umgang mit den VMs sammeln, die Ihnen die Bearbeitung der nachfolgenden Programmieraufgaben erheblich erleichtern wird.
Es werden Rahmenprogramme für C und Java bereitgestellt über das SVN Repository unter *pub/assignment1/* zur Verfügung gestellt. Es bietet sich an, die Rahmenprogramme als Grundlage für die Abgabe zu verwenden. Stellen, an denen die Programme modifiziert werden müssen, sind mit einem “TODO” gekennzeichnet. Um Ihnen eine Rückmeldung zu Ihrer Abgabe vor der Deadline zu ermöglichen, existiert ein Test-Framework, das die aktuelle Programmversion auf ihre Funktionsfähigkeit testet.
Der Aufbau eines WoL-Pakets ist in Abbildung 1 dargestellt. Eine detaillierte Anleitung zur Benutzung von Makefiles und Testumgebung finden sich in den Abschnitten 2.1 und 2.3. Falls Sie darauf verzichten wollen, die Rahmenprogramme zu verwenden, finden Sie wichtige Hinweise in 2.4.

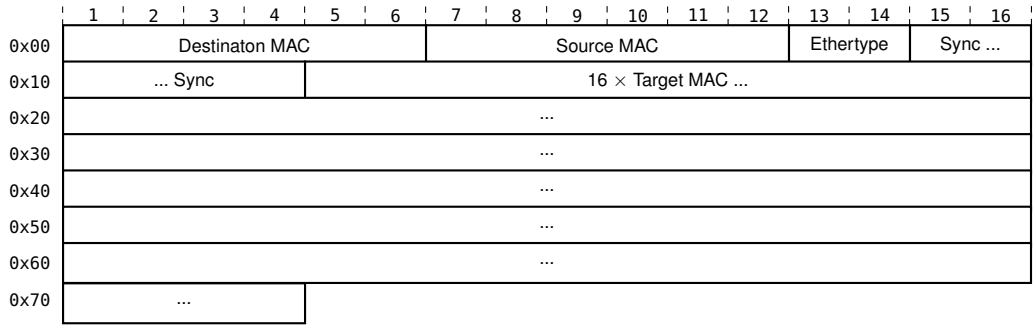

Abbildung 1: WoL-Rahmenformat. *Destination MAC* ist die L2-Broadcast-Adresse. *Source MAC* ist die MAC-Adresse des Interfaces, von dem aus der Rahmen versendet wird. *Ethertype* gibt die Art L2-SDU an und ist im Falle von WoL `0x0842` (auf Byte-Order achten!). Im Anschluss folgt die WoL-Payload, die aus einer 6 B langen Synchronisationssequenz (`0xff`) gefolgt von 16 Wiederholungen der MAC-Adresse des Zielrechners.

2 Benutzung der Programmierumgebung

Nach dem Auschecken ihres SVN-Repositories erhalten Sie nun die folgende Ordnerstruktur.

```
assignment1
|-- C
|   |-- Makefile
|   |-- libraw
|   |   |-- Makefile
|   |   |-- include
|   |   |   |-- hexdump.h
|   |   |   `-- raw.h
|   |   `-- src
|   |       |-- hexdump.c
|   |       |-- raw.c
|   |       `-- timespec.h
|   `-- src
|       |-- assignment1.c
|       |-- wol.c
|       `-- wol.h
`-- java
    |-- Makefile
    |-- deps
    ...
```

```
...
|   |-- GRNVS_RAW.c
|   |-- GRNVS_RAW.h
|   |-- libraw
|   |   |-- Makefile
|   |   |-- include
|   |   |   |-- hexdump.h
|   |   |   `-- raw.h
|   |   `-- src
|   |       |-- hexdump.c
|   |       |-- raw.c
|   |       `-- timespec.h
|   |-- manifest.txt
|   `-- run
`-- src
    |-- Arguments.java
    |-- Assignment1.java
    |-- GRNVS_RAW.java
    `-- Timeout.java
```

In den Unterordnern `C` und `Java` finden sich die Umgebungen, die Sie für die Bearbeitung der Aufgabe benötigen. Entscheiden Sie sich für *eine* Programmiersprache und kopieren Sie ***den Inhalt eines dieser Ordner*** in ihr Arbeitsverzeichnis:

`/users/<LRZ-Kennung>/assignment1/abgabe/`

Es sollten sich danach die Ordner `src, deps` und die Datei `Makefile` in `abgabe` befinden. Nachdem das gewünschte Rahmenprogramm nach `abgabe` kopiert wurde, muss dort nun `src/assignment1.c` resp. `src/Assignment1.java` bearbeitet werden.

Wichtig: Die Verzeichnisstruktur muss exakt eingehalten werden. Lediglich Quelltext im Abgabeverzeichnis wird bewertet.

2.1 Makefiles

Große Programmierprojekte setzen sich häufig aus vielen Quellcode-Dateien zusammen. Mithilfe einer *Makefile* können alle notwendigen Optionen und Parameter für den Compiler zusammengefasst und der Aufruf zum Kompilieren vereinfacht werden. Nachdem Sie Änderungen an den Rahmenprogrammen vorgenommen haben, kompilieren Sie ihr Projekt, indem Sie indem Sie im Ordner `abgabe` den Befehl `make` ausführen. Nach dem Ausführen werden Ihnen etwaige Fehler in ihrem Quelltext angezeigt. Falls der Quellcode fehlerfrei übersetzt werden konnte, befindet sich nun Ihr erzeugtes Programm mit dem Namen *wol* im Ordner `abgabe` und kann ausgeführt werden. Mit dem Befehl `make clean` entfernen Sie Dateien, die beim Kompilieren erstellt wurden. Checken Sie keine Binaries in Ihr SVN Repository ein!

2.2 Ausführung

Nachdem `make` ausgeführt wurde finden Sie nun die ausführbare Binary *wol* in Ihrem Verzeichnis. Diese kann im Ordner `abgabe` mit dem Befehl `./wol -i <interface> <target mac>` ausgeführt werden. Da es bei Wake-On-Lan keine Antworten versendet werden, steht unter `wol.net.in.tum.de` ein Server mit Webinterface zum eigenen Testen der Implementierung bereit. Der Server befindet im selben Layer 2 Segment wie die bereitgestellten virtuellen Maschinen und kann über der MAC Adresse `00:16:3e:b4:ac:14` angesprochen werden. Das Webinterface unter `http://wol.net.in.tum.de` zeigt den Empfang eines validen WoL-Pakets über den aktuellen Timestamp zur Absender MAC Adresse an. Die MAC Adresse Ihrer VM können sie z. B. durch den Befehl `ip link show` unter `link/ether` erfahren:

```
root@wol.net.in.tum.de:~# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode DEFAULT group default qlen 1000
    link/ether 00:16:3e:b4:ac:14 brd ff:ff:ff:ff:ff:ff
```

Hier hat das Interface `eth0`, von dem aus das WoL-Paket versendet wird, die MAC Adresse `00:16:3e:b4:ac:14`[1].
Ohne zusätzliche Parameter wird durch die Rahmenprogramme das Netzwerkinterface `eth0` verwendet. Soll das WoL-Paket über ein anderes Interface versendet werden, so kann dieses über den Switch `-i` angegeben werden. Für das Interface `enp3s0` müsste dann `./wol -i enp3s0 <target mac>` ausgeführt werden. Die Tests gegen `wol.net.in.tum.de` können von Ihnen selbst ausgeführt werden und sind nicht in Zeit oder Anzahl beschränkt.

2.3 Abgabetests

Wenn Sie der Meinung sind, dass Ihr Programm die Anforderungen erfüllt, können sie die Funktionalität durch die bereitgestellte Testumgebung überprüfen lassen. Um die korrekte Ausführung des Testers zu gewährleisten, dürfen die Ordnerstrukturen sowie die Makefile nicht verändert werden. Im Ordner `assignment1` befindet sich seit dem letzten SVN Update die Datei `testbitte.txt`. Um einen Test anzufordern, verändern Sie diese Datei und committen sie in das SVN Repository.

[1] Es handelt sich hierbei um `wol.net.in.tum.de`

Beachten Sie:[2] Sie können nur einen Test alle 6 Stunden ausführen lassen. Falls Sie vor Ablauf der 6 Stunden einen zweiten Test beantragen, wird dieser verzögert, bis die 6 Stunden vergangen sind. Die Testumgebung wird nach dem Commit die Ergebnisse des Tests in den Ordner `results/<revision nr>` schreiben, welchen Sie nach einem `svn update` erhalten.

2.4 Hinweise falls Sie nicht die Rahmenprogramme verwenden

Es ist möglich, die Aufgaben in einer Sprache Ihrer Wahl zu lösen. Es gibt ein paar Dinge, die dabei aber beachtet werden müssen:

- Der Arbeitsaufwand muss vergleichbar sein mit dem der Rahmenprogramme (magic.sendwol()) währe **nicht** ok. Falls Sie sich unsicher sind, fragen Sie die Übungsleitung.
- Die Ausführbare Datei **muss** *wol* heißen.
- Die Ausführbare Datei **muss** die Parameter der Rahmenprogramme unterstützen. (`-i <interface> <destination-mac>`)
- Es **muss** eine Makefile existieren. Falls Ihre Lösung nicht kompiliert werden muss, da Sie z.B. eine Scriptsprache verwendet haben, muss dennoch eine Dummy-Makefile vorhanden sein, welche ggf. einfach nichts tut.
- Um Pakete vor dem Test zu installieren müssen die Paketnamen in einer Datei *deps.txt* in dem Ordner *abgabe* stehen.
- Pakete dürfen nur aus Debian Jessie *main* und *contrib* installiert werden.

[2] Aktuell werden die Test noch nicht vollautomatisch aus dem SVN ausgeführt. Deshalb kommt es noch zu Verzögerungen bei der Testausführung.