Группа

Имя студента

Номер лабы

Сама лаба

https://drive.google.com/drive/folders/1-zAMKks3xeqXtg90uQBAG0oUhvhe11Lx?usp=share\_link

Лаб1

Forkbomb

1. Написать программу forkbomb для Linux, Windows

2. Составить график числа процессов в ОС

3. Как ОС реагирует на форкбомбу

Лаб 2

Membomb

1. Написать программу выделения памяти и заполнения ее нулями с

шагом, равным размеру страницы памяти (mmap, VirtualAlloc)

2. Составить график свободной памяти

3. Ознакомиться с работой демона OOM Killer в Linux

4. Достичь сообщения о невозможности выделить память в Windows

swapoff -a

<https://www.informit.com/articles/article.aspx?p=101760&seqNum=2> - старый планировщик

<https://developer.ibm.com/tutorials/l-completely-fair-scheduler/> - новый планировщик

<https://csc.sibsutis.ru/sites/csc.sibsutis.ru/files/courses/os/Lecture_06.pdf>

<https://www.ibm.com/developerworks/ru/library/l-scheduler/index.html>

<https://www.tux.in.ua/articles/527>

<https://www.kernel.org/doc/html/latest/scheduler/sched-design-CFS.html>

<https://serverfault.com/questions/948401/debugging-and-fine-tuning-the-linux-process-scheduler>

<https://developer.ibm.com/tutorials/l-completely-fair-scheduler/>

<http://www.brendangregg.com/perf.html#SchedulerAnalysis>

<https://doc.opensuse.org/documentation/leap/archive/42.1/tuning/html/book.sle.tuning/cha.tuning.taskscheduler.html>

<https://man7.org/linux/man-pages/man7/sched.7.html>

Лаб3

https://github.com/ereyes01/linpack

Простой вариант

Все на одной ОС

Найти и скомпилировать программу linpack для оценки производительности компьютера (Flops) и протестировать ее при различных режимах работы ОС:

1. С различными приоритетами задачи в планировщике
2. С наличием и отсутствием привязки к процессору
3. Провести несколько тестов, сравнить результаты по 3 сигма или другим статистическим критериям

Усиленный вариант

То же самое, плюс изменить параметры на уровне ядра (выбрать одно):

1. Запретить выполнение всех потоков кроме того, который тестируется (путем запрета прерываний) (cli sti)
2. Найти другие планировщики процессов для Linux и сравнить результаты работы вычислительной задачи на них
3. Повлиять на настройки имеющегося планировщика
4. Вмешаться в работу планировщика на уровне ядра

ФС

<http://samag.ru/archive/article/245> <http://pascal.net.ru/book/fat.pdf> - fat

<http://samag.ru/archive/article/203> - ext2

Лаб 4. Планировщик

Провести тестирование и найти лучший планировщик ввода-вывода среди других.

Усложнение

Модифицировать существующий планировщик на уровне ядра

Планирование IO

<https://habr.com/ru/post/81504/>

DISC="sda"; \

cat /sys/block/$DISC/queue/scheduler; \

for T in kyber bfq none; do \

echo $T > /sys/block/$DISC/queue/scheduler; \

cat /sys/block/$DISC/queue/scheduler; \

sync && /sbin/hdparm -tT /dev/$DISC && echo "----"; \

sleep 15; \

done

<https://xakep.ru/2014/05/11/input-out-linux-planning/>

<https://habr.com/ru/company/selectel/blog/346844/>

<https://habrahabr.ru/post/337102/>

<https://www.opennet.ru/base/sys/linux_shedulers.txt.html>

<http://books.gigatux.nl/mirror/kerneldevelopment/0672327201/ch13lev1sec5.html>

https://www.kernel.org/doc/Documentation/block/cfq-iosched.txt

<https://web.archive.org/web/20080606005055/http://www.dcs.ed.ac.uk/home/stg/pub/D/disk.html>

https://www.kernel.org/doc/html/latest/block/bfq-iosched.html

Лаб 5. Тестирование ФС.

Выбрать 3 (или больше) файловых систем, выбрать методику проверки и найти лучшую из них.

Усложненный вариант

Экзотические фс

или

Экзотические методики проверки

Задания на паре

1. С помощью dd создать диск

dd if=/dev/zero of=123.bin bs=1M count=64

2. С помощью mkfs создать файловую систему mkfs -t ntfs -F 123.bin

3. Смонтировать фс в папку

mount 123.bin mymount/

echo dfg > mymount/123.txt

umount 123.bin

4. losetup /dev/loop0 disk.img

5. fdisk disk.img

создать 2 диска

6. kpartx -a 123.bin

создать фс на обоих дисках, смонтировать, создать файлы, размонтировать,

kpartx -d 123.bin

7. lvm тома

создать диск из двух файлов

dd if=/dev/zero of=./d01 count=1 bs=1M # забить файл данными

losetup -f --show ./d01 # смонтировать файл как устройство

pvcreate /dev/… # сделать доступным

Pvmove # освободить от данных диск

Группа дисков

vgscan # какие группы есть

vgcreate –s 32M [name] disks # создать группу дисков

vgextend, vgreduce # добавить и удалить из группы

Логический диск

lvcreate -n first -L 2G vg # создать логический том

lvresize -L 40G vg1/lv2 # изменить размер тома

Lvremove # удаление тома

Memory management

<https://www.kernel.org/doc/gorman/html/understand/understand009.html>

<https://blog.holbertonschool.com/hack-the-virtual-memory-malloc-the-heap-the-program-break/>

<http://goog-perftools.sourceforge.net/doc/tcmalloc.html>

<https://codearcana.com/posts/2016/07/11/arena-leak-in-glibc.html> - статистика по памяти

Лаб 6.

Протестировать функцию malloc/free и построить график зависимости времени выделения от размера запрашиваемой памяти.

Либо винда, либо линукс

Сложный (или)

1. Сравнить с другими малоками

2. Тестировать на живом процессе

sudo perf record -ag -e syscalls:sys\_enter\_mmap -- sleep 30

sudo perf script --header

Лаб 7.

Перечислите все известные вам способы обнаружения работы в виртуальной машине.

(>=5)

Сложный вариант (или)

1. Привести способ выхода из виртуальной машины
2. На ассемблере

Практика

1. Установить docker https://docs.docker.com/engine/install
2. Запустить ubuntu, пробросить порт 8000, запустить python -mSimpleHTTPServer, пробросить внутрь каталог, предоставить доступ к каталогу

Полезные команды в docker

1) sudo docker run -ti ubuntu /bin/bash - запустить контейнер и перейти в интерактивный режим

2) CTRL+P+Q - отсоединиться от контейнера, оставив его запущенным

3) docker ps -a - посмотреть список запущенных контейнеров

4) docker commit <id> <name> - сохранить контейнер id как образ name

5) docker images - получить список имеющихся образов

docker search - найти image по имени

docker pull <ИМЯ Image> - загрузить себе образ

6) docker attach <id> - соединиться с контейнером

7) docker run <Разные опции> -ti ubuntu /bin/bash - запустить контейнер и пробросить его сетевой интерфейс на интерфейс хоста - все открытые порты виртуалки будут открыты в реале (CTRL+P+Q)

docker ps -a - показать все контейнеры

docker attach <id контейнера> - приаттачиться к контейнеру

docker exec -ti <id контейнера> /bin/bash - исполнить команду в контейнере

docker run --net="host" -ti ubuntu /bin/bash - стартовать с нативным адаптером сети

-p <HOST PORT>:<GUEST PORT>

8) docker run -v host\_path:container\_path - пробросить в контейнер папку

9) docker export <id> <name> - экспортировать контейнер

10) cat name.tar | docker import - <name> - импортировать контейнер

Dockerfile - конфиг со способом создания Image

FROM ubuntu

RUN apt update

RUN apt install -y python3

WORKDIR /home/test

ENTRYPOINT python3 -m http.server 2000

docker build -t mycont .

docker-compose.yml - какие контейнеры создать и какие порты и папки в них пробросит

version: "2.0"

services:

my\_cointainer:

build: .

volumes:

- /tmp/test/shared:/home/test

ports:

- 8011:2000

docker-compose up --build -d

PAM

<https://www.opennet.ru/base/dev/pam_linux.txt.html>

SElinux

<https://habr.com/ru/company/kingservers/blog/209644/>

<https://defcon.ru/os-security/1264/>

Apparmor

<https://help.ubuntu.ru/wiki/%D1%80%D1%83%D0%BA%D0%BE%D0%B2%D0%BE%D0%B4%D1%81%D1%82%D0%B2%D0%BE_%D0%BF%D0%BE_ubuntu_server/%D0%B1%D0%B5%D0%B7%D0%BE%D0%BF%D0%B0%D1%81%D0%BD%D0%BE%D1%81%D1%82%D1%8C/apparmor>

<https://losst.ru/nastrojka-apparmor-v-ubuntu-16-04>

<https://manpages.ubuntu.com/manpages/bionic/en/man5/apparmor.d.5.html>

<https://defcon.ru/os-security/1544/> - история

<https://ru.bmstu.wiki/AppArmor> - правила

LSM

<https://habr.com/ru/company/pt/blog/144014/>

Windows: Руссинович, Внутреннее устройство Windows, стр 510

Маркер доступа - <https://ru.bmstu.wiki/Access_Token>

Дескриптор защиты - <https://intuit.ru/studies/courses/10471/1078/lecture/16581?page=2>

<https://docs.microsoft.com/en-us/windows/win32/api/winnt/ns-winnt-security_descriptor?redirectedfrom=MSDN>

Про подписывание драйверов <https://docs.microsoft.com/en-us/windows-hardware/drivers/install/kernel-mode-code-signing-policy--windows-vista-and-later->

ELAM <https://docs.microsoft.com/en-us/windows-hardware/drivers/install/early-launch-antimalware>

Kernel Patch Protector <https://xakep.ru/2019/11/25/byepg/>

EWT - https://docs.microsoft.com/en-us/windows/win32/etw/event-tracing-portal

HVCI - <https://docs.microsoft.com/en-us/windows/security/threat-protection/device-guard/enable-virtualization-based-protection-of-code-integrity>

Про UAC https://docs.microsoft.com/en-us/windows/security/identity-protection/user-account-control/how-user-account-control-works

Лаб 8.

Обе

1. Настроить Apparmor для мониторинга сложного приложения и продемонстрировать его работу при ограниченных правах (оконное приложение или веб-сервер)
2. Настроить selinux в режиме мандатного доступа (CentOS и др.) и продемонстрировать работу в двухуровневой модели.

Усиленный вариант (или)

1. Придумать и написать свой LSM-модуль (сложная авторизация действий)

2. Придумать и написать свой PAM-модуль (сложная авторизация действий)

3\*. Свой сложный Seccomp-фильтр

Задание

apparmor-profiles

apparmor-utils

aa-status

aa-complain - режим мониторинга

aa-enforce - режим защиты

aa-autodep - новый профиль

aa-genprof - обновить профиль

сгенерировать свой профиль для ncat

Сети

Лаб 9.

Одно из

1. Написать фильтр сетевых пакетов на основе nfqueue и iptables и протестировать скорость работы
2. Протестировать работу сокетов tcp при различных настройках setsockopt

Сложный вариант (одно из)

1. Написать фильтр пакетов на основе интерфейса netfilter (ядро)
2. Реализовать rpc-программу для linux с поддержкой аутентификации (rpcinfo,rpcbind)

<https://it.wikireading.ru/7079> - опции сокетов

<https://www.ibm.com/developerworks/ru/library/l-hisock/index.html> - оптимизация tcp

<https://mathcs.clarku.edu/~jbreecher/cs280/UNIX%20Network%20Programming(Volume1,3rd).pdf>

<http://tharikasblogs.blogspot.com/p/how-to-write-simple-rpc-programme.html> - создание rpc-программ для linux

ftp://[ftp.mao.kiev.ua/pub/docs\_books/Linux/%D0%A3%D0%BE%D0%BB%D1%82%D0%BE%D0%BD%20%D0%A1%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5%20%D1%81%D0%B5%D1%82%D0%B5%D0%B2%D1%8B%D1%85%20%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B9%20%D0%B2%20%D1%81%D1%80%D0%B5%D0%B4%D0%B5%20linux.pdf](http://ftp.mao.kiev.ua/pub/docs_books/Linux/%D0%A3%D0%BE%D0%BB%D1%82%D0%BE%D0%BD%20%D0%A1%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5%20%D1%81%D0%B5%D1%82%D0%B5%D0%B2%D1%8B%D1%85%20%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B9%20%D0%B2%20%D1%81%D1%80%D0%B5%D0%B4%D0%B5%20linux.pdf)

Практика

1. Реализовать клиент-сервер на языке си по протоколу TCP (сервер и клиент).

10-15 минут 12-00

1. Реализовать обмен данными, не блокирующий ввод с сокета и клавиатуры через select