1. **Теория**

Кнопка POWER кратковременно замыкает контакты на системной плате (motherboard).

Первым делом запускается тестирование питания системного блока (проверка напряжения на линиях питания). Если с питанием все “ок”, посылается сигнал Power Good. На вход ЦП (центральный процессор, CPU) подается команда на сброс. Процессор обнуляет содержимое памяти и начинает работу.

Далее из BIOS (basic input/output system) считываются инструкции по тестированию, инициализации и диагностике устройств POST (Power-On Self Test). Программа POST выполняет тестирование процессора, оперативной памяти, чипсета, видеосистемы, накопителей, системы управления питанием, клавиатуры, портов и подключенных к ним устройств и других компонентов компьютера.

При успешном завершении POST начинается выполнение кода с указанного в настройках BIOS загрузочного раздела диска. В этом разделе содержатся инструкции по загрузке/развертыванию (bootstrap) операционной системы. В оперативную память загружается ядро операционной системы и загружается на исполнение в ЦП.

Далее по ядру Linux:

Ядро монтирует файловую систему

Выполняет программу /sbin/init

Init смотрит на уровень выполнения

Выполняются некоторые файлы в соответствии с этим уровнем выполнения.

В результате вышеописанных действий буду загружены определенные программы, в том числе можно прописать загрузку nginx (например в /etc/init.d или /etc/rc.local).

1. Сотрудникам так и не дали доступ, установил виртуалку на свою машину
2. Скомпилировал программу нв C:

#include <stdio.h>

int main(void) {

printf("Hello, world!\n");

return 0;

}

Системные вызовы согласно strace:

**read** - чтение из файла (все есть файл)

**write** - запись в файл

**open** - открытие файла

**close** - закрытие файла

**fstat** - статус файла

**mmap** - maps files or devices into memory

**mprotect** - is used to control memory protection (memory access)

**munmap** - unmap or devices from memory

**brk** - устанавливает конец сегмента данных в значение, указанное в аргументе end\_data\_segment, когда это значение является приемлимым, система симулирует нехватку памяти и процесс не достигает своего максимально возможного размера сегмента данных

**access** - проверяет имеет ли процесс доступ к файлу

**execve** - выполняет программу

**arch\_prctl** - sets architecture-specific process or thread state

**4. \***

Скомпилировал HelloWorld на Swift:

print("Hello, swift!")

К сожалению из коробки Swift бинарник запускается только на MacOS, так что собирал системные вызовы на macOS с помощью аналога strace - утилиты dtruss

Системные вызовы согласно dtruss:

**access**

**bsdthread\_register**

**exit** - нормальное завершение выполнения

**fstat64**

**getpid** - returns the process ID of the current process

**getrlimit** - get and set resource limits respectively

**getuid** - returns the real user ID of the calling process.

**issetugid** - returns 1 if the process environment or memory address space is considered ``tainted'', and returns 0 otherwise.

**madvise** - give advice or directions to the kernel about the address range beginning at address addr and with size length bytes

**thread\_selfid**

**write\_nocancel**

**csops**

**getegid**

**gettid**

**ioctl** - манипулирует базовыми параметрами устройств, представленных в виде специальных файлов.

**read**

**stat64**

**sysctl** - is used to modify kernel parameters at runtime

**close**

**geteuid**

**open**

**readlink**

**mprotect**

**getattrlist**

**5. \*\***

Простой Hello world

ldd Hello

linux-vdso.so.1 => (0x00007ffd197e0000)

libc.so.6 => /lib64/libc.so.6 (0x00007f439c8f8000) - стандартная библиотека c

/lib64/ld-linux-x86-64.so.2 (0x00007f439ccc5000)

ldd /usr/bin/mv

linux-vdso.so.1 => (0x00007ffeda5b1000)

libselinux.so.1 => /lib64/libselinux.so.1 (0x00007fd95714b000)

libacl.so.1 => /lib64/libacl.so.1 (0x00007fd956f42000)

libattr.so.1 => /lib64/libattr.so.1 (0x00007fd956d3d000)

libc.so.6 => /lib64/libc.so.6 (0x00007fd956970000)

libpcre.so.1 => /lib64/libpcre.so.1 (0x00007fd95670e000)

libdl.so.2 => /lib64/libdl.so.2 (0x00007fd95650a000)

/lib64/ld-linux-x86-64.so.2 (0x00007fd957372000)

libpthread.so.0 => /lib64/libpthread.so.0 (0x00007fd9562ee000)