Fork!

A fork() in the road

cs380l

Indictments against fork()

- Not thread-safe
- Inefficient and unscalable
- Security concerns
  - From a security perspective, the inherit-by-default behaviour of fork violates the principle of least privilege.
- Conflates process with address space
  - A process using, say, DPDK with a kernel-bypass NIC, or OpenCL with a GPU, cannot safely fork since the OS cannot duplicate the process state on the NIC/GPU
- Hostile to user-space implementation of OS functionality:
  - Breaks buffered I/O
  - Breaks kernel-bypass networking
  - Does not compose
  - Everything must support it

Historical Context

- Hack: 27 lines of PDP-7 assembly
- Fork was simple
  - Cf CreateProcessEx
- Fork eased concurrency

Modern Context

- Not simple
  - 25 POSIX special cases (e.g. file locks, timers, async IO, tracing)
  - Bleeds into other syscalls
    - e.g. madvise: MADV_DONTFORK/DOFORK,WIPEONFORK
    - O_CLOEXEC,FD_CLOEXEC, pthread_atfork()
- Not thread-safe
  - How to keep multiple forks from getting different versions of parent AS?
- Insecure
  - Inherit everything → violates least privilege principle
- Incompatible with single AS (e.g. unikernels)
- Incompatible with heterogeneous HW (e.g. GPUs)
  - (Maybe this isn't fork's fault)

Fork perfor(k)mance

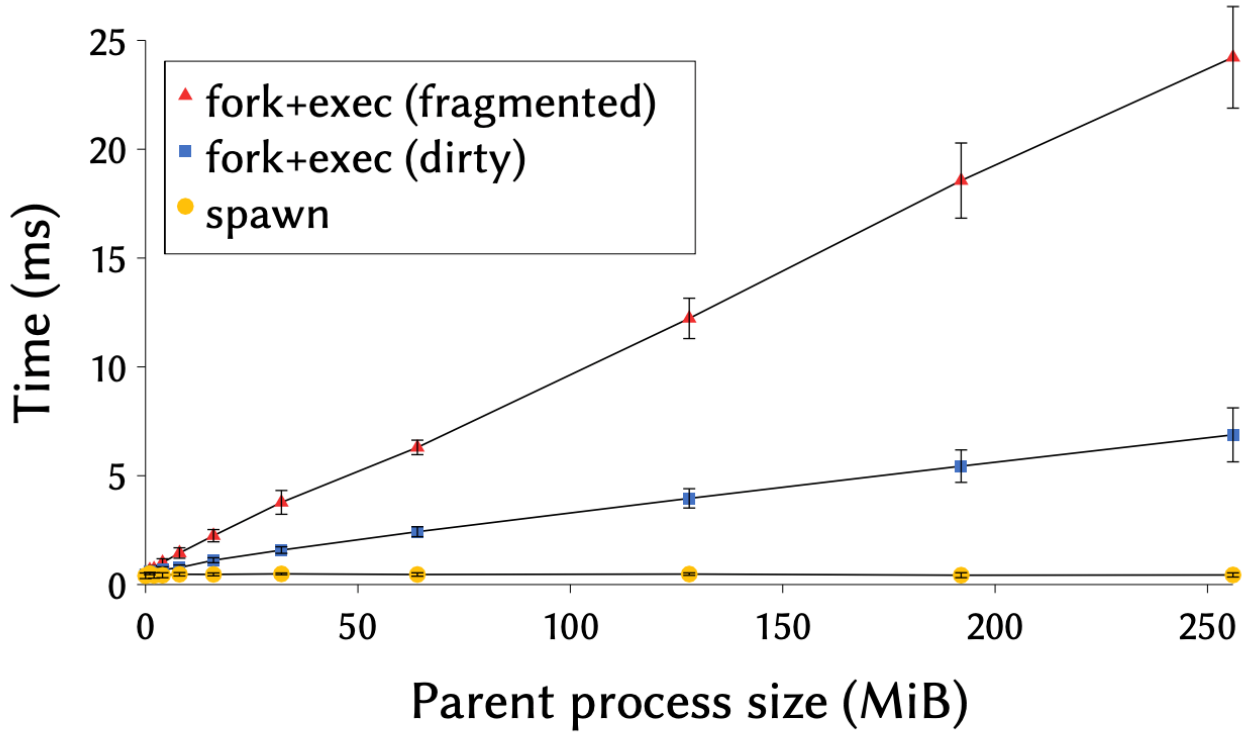

Figure 1: Cost of `fork()` + `exec()` vs. `posix_spawn()`

Replacements

- posix_spawn()
  - Gets close
  - Incomplete
    - doesn’t support terminal attributes, isolated namespaces
    - Errors reported asynchronously
- vfork()
  - Child shares parent’s address space *until* calling exec()
  - Parent is suspended while child executes
  - Difficult to use safely (child and parent share address space)

CreateProcess

```
BOOL CreateProcessA(
  [in, optional]      LPCSTR                lpApplicationName,
  [in, out, optional] LPSTR                 lpCommandLine,
  [in, optional]      LPSECURITY_ATTRIBUTES lpProcessAttributes,
  [in, optional]      LPSECURITY_ATTRIBUTES lpThreadAttributes,
  [in]                BOOL                  bInheritHandles,
  [in]                DWORD                 dwCreationFlags,
  [in, optional]      LPVOID                lpEnvironment,
  [in, optional]      LPCSTR                lpCurrentDirectory,
  [in]                LPSTARTUPINFOA        lpStartupInfo,
  [out]               LPPROCESS_INFORMATION lpProcessInformation
);
```

posix_spawn()

```
int posix_spawn(
        pid_t *restrict pid,
        const char *restrict path,
        const posix_spawn_file_actions_t *restrict file_actions,
        const posix_spawnattr_t *restrict attrp,
        char *const argv[restrict],
        char *const envp[restrict]);
```

Discuss

- Pick one thing fork does badly, explain it and explain a better alternative
- Explain if you are more convinced by fork's downsides or the potential upsides of a spawn interface