genua.

OpenBSD Security Mitigations

Alexander Bluhm

September 2023

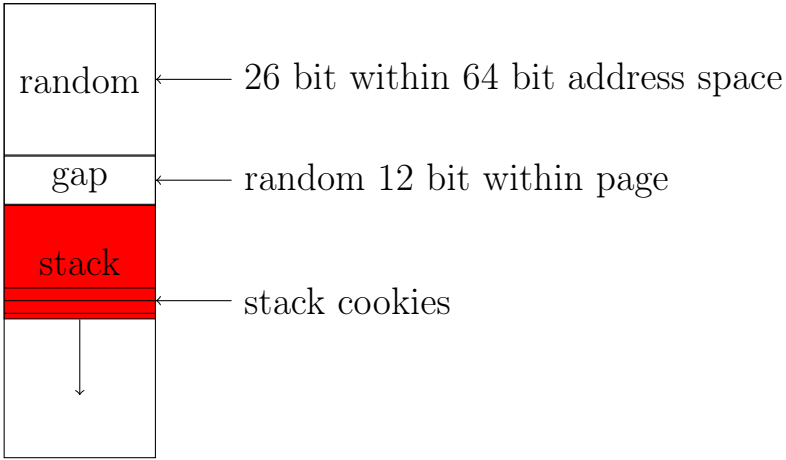
genua.

Stack Overflow

1. random gap
2. non executable
3. stack protector
4. retguard
5. immutable
6. random location

genua.

Stack Layout



genua.

Heap Overflow

- `malloc(3)` options
- guard page
- random chunk order
- canaries
- meta data `mprotect(2)` and `mimmutable(2)`

genua.

Heap Use-After-Free

- unmap freed
- junk at alloc
- junk at free
- SSH use-after-free
- limit chunk reuse

genua.

Return Oriented Programming

- map stack
- retguard
- avoid 0xc3 due to %rbx
- fork+exec
- `sigreturn(2)` cookie
- `setjmp(3)`, `longjmp(3)` cookie
- ARM BTI & Intel IBT endbr64
- random relinking

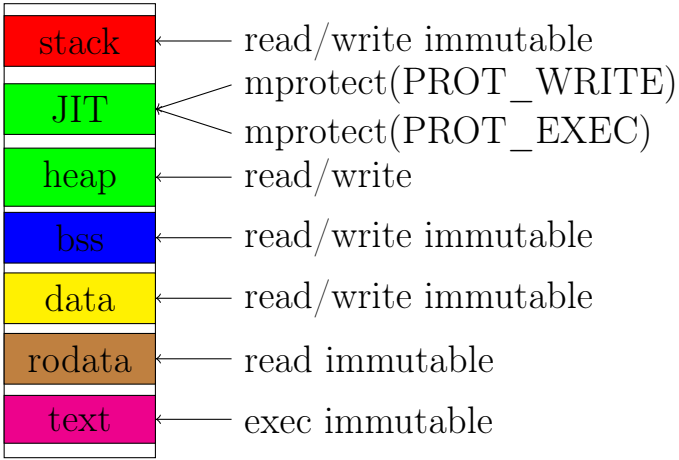
genua.

Mapping Protection

- `mprotect(2)`
  - W^X
  - xonly
- `mimmutable(2)`
- `mmap(2)`
  - MAP_STACK
  - MAP_CONCEAL

genua.

Address Space Protection



genua.

Hardware xonly

- arm64, riscv64 have RWX
- new amd64 have PKU

genua.

Opportunistic xonly

```
addr = mmap(NULL, 4096, PROT_NONE,
    MAP_PRIVATE | MAP_ANON, -1, 0);
mprotect(addr, 4096, PROT_READ);
printf("%p: %08x\n", addr, *(int *)addr);
mprotect(addr, 4096, PROT_EXEC);
printf("%p: %08x\n", addr, *(int *)addr);

0x73d15433000: 00000000
0x73d15433000: 00000000
```

genua.

Trap xonly

```
addr = mmap(NULL, 4096, PROT_NONE,
    MAP_PRIVATE | MAP_ANON, -1, 0);
mprotect(addr, 4096, PROT_READ);
mprotect(addr, 4096, PROT_EXEC);
printf("%p: %08x\n", addr, *(int *)addr);

Segmentation fault (core dumped)
```

genua.

Random Everywhere

- `arc4random(3)`
- `arc4random_uniform(3)`
- `srand_deterministic(seed)`
- `getentropy(2)`
- ELF header `.openbsd.randomdata`
- boot /etc/random.seed

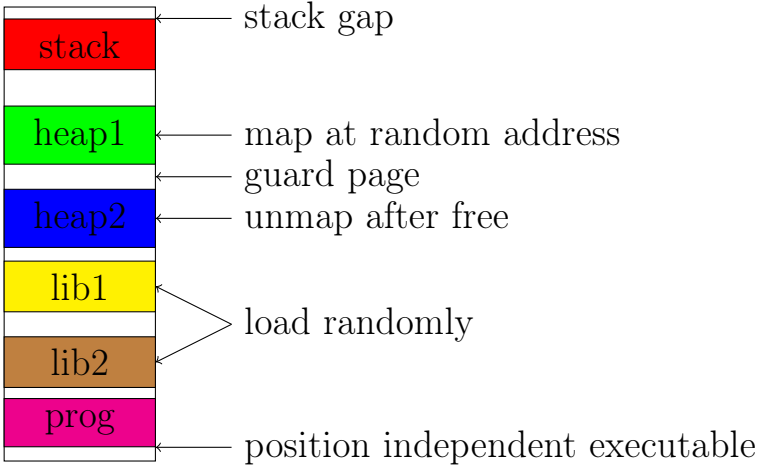
genua.

Address Space Layout Randomization

1. shared library mapping
2. heap `mmap(2)`
3. stack gap
4. PIE program text
5. relink libc
6. relink `sshd(8)`

genua.

Address Space Layout



genua.

Process Mapping

```
sleep 1 & procmap $!
```

- `Start End` random
- `rwxSeIpc` xonly, stack, system call entry, immutable
  - `--x-eIp+` text
  - `r----Ip+` read only
  - `rw---Ip-` data
  - `rw----p-` heap
  - `rw-S-Ip-` stack

genua.

Kernel Exec

- check argc > 0
- random stack
- random signal cookie
- exec pledge
- setuid open stdin, stdout, stderr

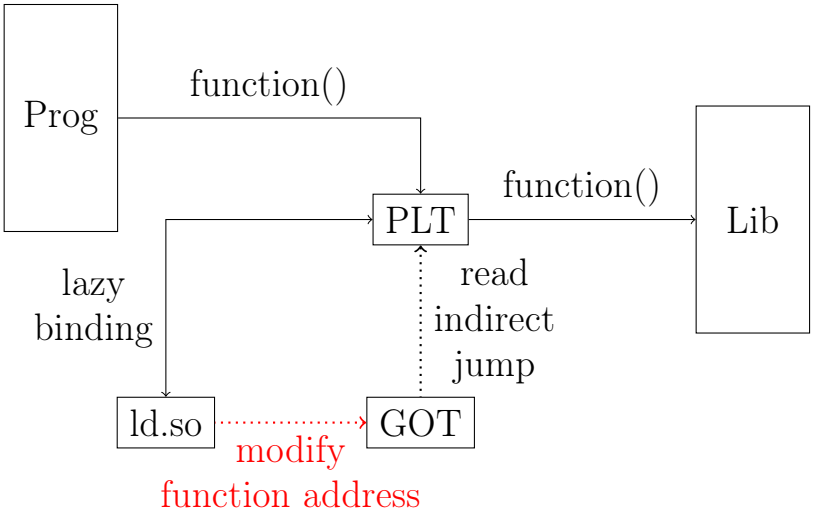
genua.

Dynamic Loader

- init random `getentropy(2)`
- fix permissions `mimmutable(2)`
- random layout `mmap(2)`
- syscall area `msyscall(2)`
- exec area `pinsyscall(2)`
- lazy-binding `kbind(2)`

genua.

Lazy Binding kbind(2)



genua.

Pledge and Unveil

- ps -ax -o stat | grep p ⇒ 86% pledged
- ps -ax -o stat | grep U ⇒ 39% unveiled
- easy to use
- structures design securely
- daily violation mail
- `lastcomm(1)`, `acct(5)`, `accton(8)`

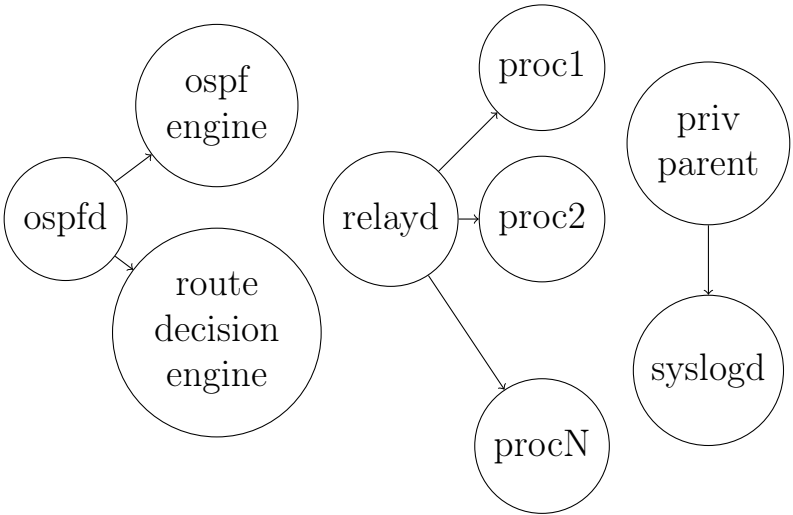
genua.

Privilege Separation in Processes

- identify isolated tasks with high risk
- `socketpair(2)`
- `fork(2)`
- `chroot(2)`
- `setuid(2)`
- `pledge(2)`
- `imsg_init(3)`
- file descriptor passing

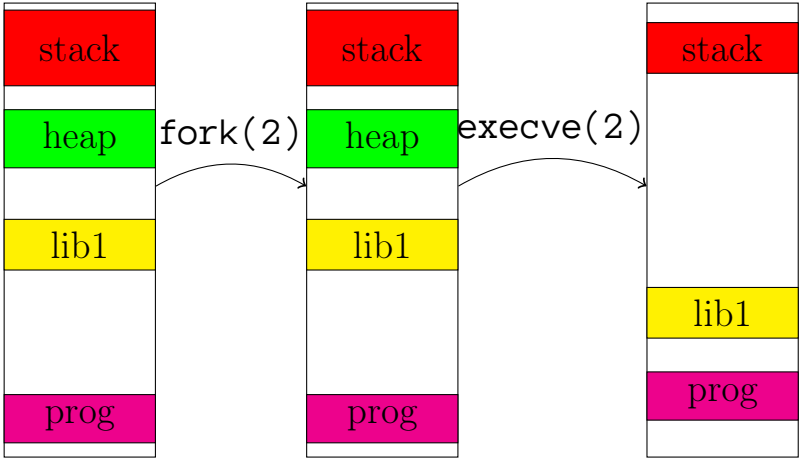
genua.

Programs with Privsep



genua.

fork+exec



genua.

API Changes

- `strlcpy(3)`, `strlcat(3)`
- `reallocarray(3)`
- `explicit_bzero(3)`
- `freezero(3)`
- `malloc_conceal(3)`
- `printf(3)` without %n

genua.

File Descriptors

- `getdtablecount(2)`
- `sendsyslog(2)`
- `socket(2)` type SOCK_DNS

genua.

Ports Pain

- BTI and IBT
- xonly
- map stack
- W^X
- otto@ malloc

genua.

Default and Force

- control: kernel, libc, base
- adapt: ports
- last resort: mount and program flags
  - mount -o wxallowed
  - ld -z wxneeded
  - ld -z nobtcfi
  - NOT execstack, force-bti, force-ibt

genua.

SSH Agent Library Exploit

`dlopen(3)` 4 libraries:

- ELF header executable stack
- init signal handler SIGSEGV
- ELF flag NODELETE
- on load segmentation fault

genua.

Not Yet

- ARM pointer authentication
- shadow stack
- remove `syscall(2)`

genua.

Conclusion

- powerful combination
- cheap and effective
- always on
- works with ports

genua.

Questions

?