Mariusz Zaborski: Case studies of sandboxing base system with Capsicum

Outline

- capsicum
- is capsicumizing hard?
- debugging infra
- casper
- future

Capsicum

- echo: can read write all data
- capability model
- kernel inra
    - tight sandboxing
    - cap_enter()

Capsicum vs namespaces

- PIDs
- file paths
- NFS handles
- fs ids
- sysctl mib
- sysv ipc
- posix ipc
- clocks
- jails
- CPU sets
- protocol addrs
- routing tables

Capsicum

- int cap_enter(void);
- int cap_rights_limit(int fd, const cap_rights_t *rights);
- caps as file descriptors

rights

- CAP_READ, WRITE, APPEND, ACCEPT, FCHMOD, CREATE, UNLINKAT< IOCTL, RECV, LISTEN

Capsicum

- two ways to obtain more caps
    - initialization phase
    - delegation
- Privileged -> Resources
    - -> Sandboxed (send resource by fd passing)

Is capsicumizing hard?

- Not for new code. Existing code?

2015 sandboxing effort

- dhclient, ...

2016 sandboxing effort

- much more

bspatch(1)

- SA-16:25: negative value
- SA-16:29: integer overflows

bspatch(1) - Step 0: read the code

- (... C code …)

Step 1: code reorg

- every single open is done during initialization phase, i.e. move open before cap_enter()

Step 2: read more code

- code attempts to read from fds, seeking on fds

Step 3: limit operations on fds

- CAP_READ, CAP_SEEK

cmp(1) - deduplicate code

- read file, determine if same or not

capsicum helpers

- capsicum_helpers.h
- inline functions
    - caph_limit_stream(), limit_stdout(), limit_stdin(), limit_stderr()

libc is not your friend

- err(3)
- localtime(3)
- syslog
- modify vdso to not open device
- more helpers
    - caph_cache_catpages(): NLS support
    - caph_cache_tzdata()

debugging infrastructure

- ktrace/kdump
    - getting only trace
- very easy to miss something
- hard to cover all paths

debugging - krace

- (example output)

debugging - enotcap

- kern.trap_enotcap
- procctl(PROC_TRAPCAP_CTL)
- get core dump
- hard to miss something
- hard to cover all code paths

debugging - enotcap

- (example enotcap SIGTRAP)

libCasper

- 2nd way to get more caps: delegation

Casper

- provides functionality not avail in cap mode through convenient APIs
- easier process separation
- done before entering Capability mode
- Creating zygote
- set of dynamic libs

Casper: how?

- process (cap_init()) -> casper -> zygote
- casper -> service (cap_service_open())
- cap_close(): leave only process->service

Casper

- system.dns
- system.grp

- system.pwd: password files
- system.random
- system.sysctl

Traceroute - capsicumize with casper

(.. C code …)

casper - mocks

- reduce amount of ifdefs in code
- hide ifdefs in lib itself
- use inline/defines to create mocks
- e.g.
  - cap_gethostbyname ifdef'ed to gethostbyname in header if not using casper

Future!

Casper - next next generation

- integration with libc?
  - make libc more pluggable, e.g. multiple gethostbyname functions (scott: nss?)
  - start casper in _start
- sandbox services
  - services run with user privileges
  - reduce TCB

Casper services

- system.filesystem
- system.syslog
- system.login
- system.tls
- system.socket
- system.configuration

Casper - dhclient(8)

(SIGTRAP core dump example)

dhclient started before syslog in initscripts

Casper - system.syslog

- change the order? (not viable)
- casper service D12824
- fixed dhclient D12825

casper - sshd(8)

(sshd core dump example)

- login_getpwclass() was failing
- opens $HOME/login.conf and /etc/login.conf
- patch specific to FreeBSD

Acknowledgements

Q&A

- per thread caps? A: caps represented by descriptors, so no
- much more fine grained than OpenBSD pledge? A: yes
- Is this a bit like Android permissions? A: no, fd represent a much more precise thing. More granular.