A case study of sandboxing base systems with Capsicum

Mariusz Zaborski

<m.zaborski@wheelsystems.com>
<oshogbo@FreeBSD.org>

Outline

- Capsicum
- Is Capsicumizing hard?
- Debugging infrastructure
- Casper
- Future

Capsicum

Capsicum

kernel infrastructure that provides:

- tight sandboxing

```
int cap_enter(void);
```

Capsicum vs. namespace

- Process IDs
- File paths
- NFS file handle
- Filesystems IDs
- Sysctl MIB
- System V IPC
- POSIX IPC
- System clocks
- Jails
- CPU sets
- Protocol address
- Routing tables

Capsicum

kernel infrastructure that provides:

- tight sandboxing

```
int cap_enter(void);
```

- capability rights

```
int cap_rights_limit(int fd, const cap_rights_t *rights);
```

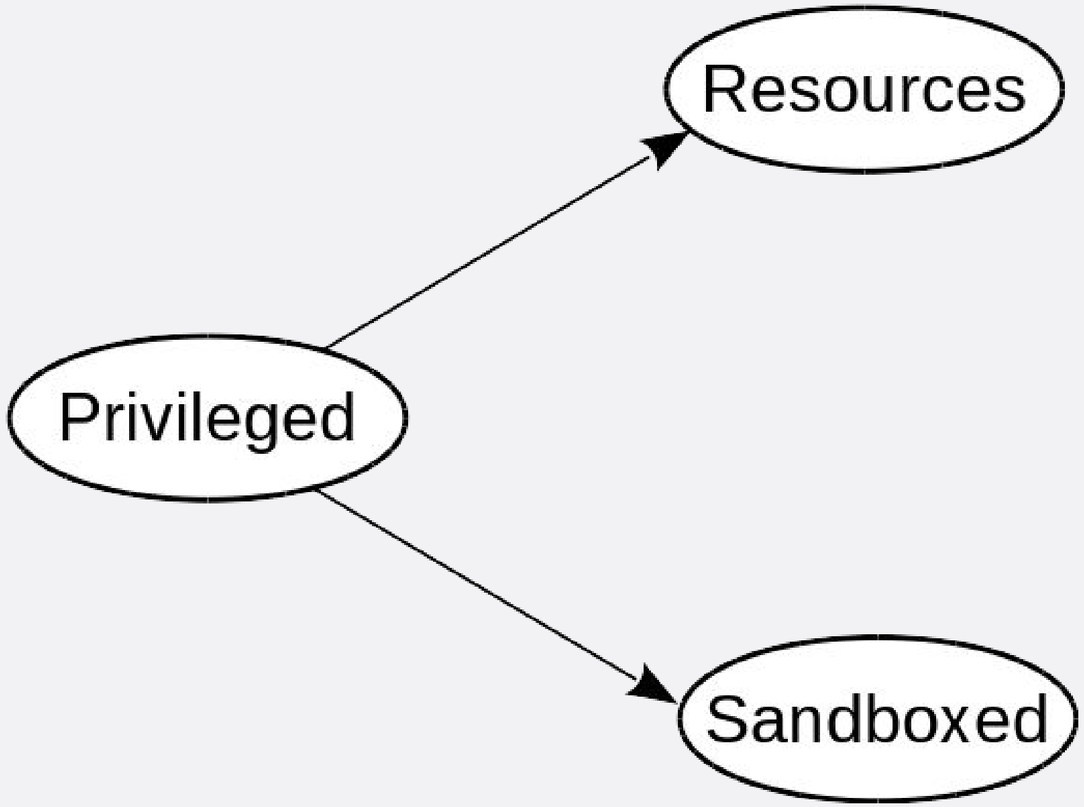
Capsicum rights

- CAP_READ
- CAP_WRITE
- CAP_APPEND
- CAP_ACCEPT
- CAP_FCHMOD
- CAP_CREATE
- CAP_UNLINKAT
- CAP_IOCTL
- CAP_RECV
- CAP_LISTEN
- ...

Capsicum

Two ways to obtain more capabilities:

- the initialization phase
- delegation



Is Capsicumizing hard?

No for new code. What about existing one?

Capsicum 2015

- dhclient(8)
- hastd(8), hastctl(8)
- rhowd(8), rwho(1)
- uniq(1)
- auditdistd(8)
- sshd(8)
- tcpdump(8)
- kdump(1)
- ping(8)

Capsicum 2016

- basename(1)
- cmp(1)
- col(1)
- elfdump(1)
- dc(1)
- dd(1)
- dirname(1)
- dma-mbox-create
- echo(1)
- factor(6)
- fold(1)
- getopt(1)
- hexdump(1)
- indent(1)
- jot(1)
- ktrdump(8)
- last(1)
- locate(1)
- logname(1)
- md5(1)
- ministat(1)
- printenv(1)
- sleep(1)
- soelim(1)
- tee(1)
- traceroute(1)
- bhyve(1)
- decryptcore(1)
- ktrdump(8)
- procstat(1)
- pkg
- irssi

bspatch(1)

- FreeBSD-SA-16:25

  The implementation of bspatch does not check for a **negative value** on numbers of bytes read from the diff and extra streams, allowing an attacker who can control the patch file to write at arbitrary locations in the heap.

- FreeBSD-SA-16:29

  The implementation of bspatch is susceptible to **integer overflows** with carefully crafted input, potentially allowing an attacker who can control the patch file to write at arbitrary locations in the heap. This issue was partially addressed in FreeBSD-SA-16:25.bspatch, but some possible integer overflows remained.

bspatch(1) - Step 0: read the code

```
if ((f = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);

if (fread(header, 1, 32, f) < 32) {
        if (feof(f))
                errx(1, "Corrupt patch\n");
        err(1, "fread(%s)", argv[3]);
}

if (memcmp(header, "BSDIFF40", 8) != 0)
        errx(1, "Corrupt patch\n");

bzctrllen = offtin(header + 8);
bzdatalen = offtin(header + 16);
newsize = offtin(header + 24);
if ((bzctrllen < 0) || (bzdatalen < 0) || (newsize < 0))
        errx(1, "Corrupt patch\n");

if (fclose(f))
        err(1, "fclose(%s)", argv[3]);
if ((cpf = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);
if (fseeko(cpf, 32, SEEK_SET))
        err(1, "fseeko(%s, %lld)", argv[3],
            (long long)32);
```

```
if ((cpfbz2 = BZ2_bzReadOpen(&cbz2err, cpf, 0, 0, NULL, 0)) == NULL)
        errx(1, "BZ2_bzReadOpen, bz2err = %d", cbz2err);
if ((dpf = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);
if (fseeko(dpf, 32 + bzctrllen, SEEK_SET))
        err(1, "fseeko(%s, %lld)", argv[3],
            (long long)(32 + bzctrllen));
if ((dpfbz2 = BZ2_bzReadOpen(&dbz2err, dpf, 0, 0, NULL, 0)) == NULL)
        errx(1, "BZ2_bzReadOpen, bz2err = %d", dbz2err);
if ((epf = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);
if (fseeko(epf, 32 + bzctrllen + bzdatalen, SEEK_SET))
        err(1, "fseeko(%s, %lld)", argv[3],
            (long long)(32 + bzctrllen + bzdatalen));
if ((epfbz2 = BZ2_bzReadOpen(&ebz2err, epf, 0, 0, NULL, 0)) == NULL)
        errx(1, "BZ2_bzReadOpen, bz2err = %d", ebz2err);
```

bspatch(1) - Step 0: read the code

```
if ((f = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);

if (fread(header, 1, 32, f) < 32) {
        if (feof(f))
                errx(1, "Corrupt patch\n");
        err(1, "fread(%s)", argv[3]);
}

if (memcmp(header, "BSDIFF40", 8) != 0)
        errx(1, "Corrupt patch\n");

bzctrllen = offtin(header + 8);
bzdatalen = offtin(header + 16);
newsize = offtin(header + 24);
if ((bzctrllen < 0) || (bzdatalen < 0) || (newsize < 0))
        errx(1, "Corrupt patch\n");

if (fclose(f))
        err(1, "fclose(%s)", argv[3]);
if ((cpf = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);
if (fseeko(cpf, 32, SEEK_SET))
        err(1, "fseeko(%s, %lld)", argv[3],
            (long long)32);
```

```
if ((cpfbz2 = BZ2_bzReadOpen(&cbz2err, cpf, 0, 0, NULL, 0)) == NULL)
        errx(1, "BZ2_bzReadOpen, bz2err = %d", cbz2err);
if ((dpf = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);
if (fseeko(dpf, 32 + bzctrllen, SEEK_SET))
        err(1, "fseeko(%s, %lld)", argv[3],
            (long long)(32 + bzctrllen));
if ((dpfbz2 = BZ2_bzReadOpen(&dbz2err, dpf, 0, 0, NULL, 0)) == NULL)
        errx(1, "BZ2_bzReadOpen, bz2err = %d", dbz2err);
if ((epf = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);
if (fseeko(epf, 32 + bzctrllen + bzdatalen, SEEK_SET))
        err(1, "fseeko(%s, %lld)", argv[3],
            (long long)(32 + bzctrllen + bzdatalen));
if ((epfbz2 = BZ2_bzReadOpen(&ebz2err, epf, 0, 0, NULL, 0)) == NULL)
        errx(1, "BZ2_bzReadOpen, bz2err = %d", ebz2err);
```

bspatch(1) - Step 1: code reorganization

```
@@ -89,0 +90,11 @@ int main(int argc, char *argv[])
+       if ((cpf = fopen(argv[3], "rb")) == NULL)
+               err(1, "fopen(%s)", argv[3]);
+       if ((dpf = fopen(argv[3], "rb")) == NULL)
+               err(1, "fopen(%s)", argv[3]);
+       if ((epf = fopen(argv[3], "rb")) == NULL)
+               err(1, "fopen(%s)", argv[3]);
+       if ((oldfd = open(argv[1], O_RDONLY | O_BINARY, 0)) < 0)
+               err(1, "open(%s)", argv[1]);
+       if ((newfd = open(argv[2], O_CREAT | O_TRUNC | O_WRONLY | O_BINARY,
+           0666)) < 0)
+               err(1, "open(%s)", argv[2]);
@@ -126,2 +177,0 @@ int main(int argc, char *argv[])
-       if ((cpf = fopen(argv[3], "rb")) == NULL)
-               err(1, "fopen(%s)", argv[3]);
@@ -133,2 +182,0 @@ int main(int argc, char *argv[])
-       if ((dpf = fopen(argv[3], "rb")) == NULL)
-               err(1, "fopen(%s)", argv[3]);
@@ -140,2 +187,0 @@ int main(int argc, char *argv[])
-       if ((epf = fopen(argv[3], "rb")) == NULL)
-               err(1, "fopen(%s)", argv[3]);
@@ -148,3 +193,0 @@ int main(int argc, char *argv[])
-       oldfd = open(argv[1], O_RDONLY | O_BINARY, 0);
-       if (oldfd < 0)
-               err(1, "%s", argv[1]);
@@ -218,3 +260,0 @@ int main(int argc, char *argv[])
-       newfd = open(argv[2], O_CREAT | O_TRUNC | O_WRONLY | O_BINARY, 0666);
-       if (newfd < 0)
-               err(1, "%s", argv[2]);
```

bspatch(1) - Capsicumize???

```
@@ -89,0 +90,11 @@ int main(int argc, char *argv[])
+       if ((cpf = fopen(argv[3], "rb")) == NULL)
+               err(1, "fopen(%s)", argv[3]);
+       if ((dpf = fopen(argv[3], "rb")) == NULL)
+               err(1, "fopen(%s)", argv[3]);
+       if ((epf = fopen(argv[3], "rb")) == NULL)
+               err(1, "fopen(%s)", argv[3]);
+       if ((oldfd = open(argv[1], O_RDONLY | O_BINARY, 0)) < 0)
+               err(1, "open(%s)", argv[1]);
+       if ((newfd = open(argv[2], O_CREAT | O_TRUNC | O_WRONLY | O_BINARY,
+           0666)) < 0)
+               err(1, "open(%s)", argv[2]);
@@ -126,2 +177,0 @@ int main(int argc, char *argv[])
-       if ((cpf = fopen(argv[3], "rb")) == NULL)
-               err(1, "fopen(%s)", argv[3]);
@@ -133,2 +182,0 @@ int main(int argc, char *argv[])
-       if ((dpf = fopen(argv[3], "rb")) == NULL)
-               err(1, "fopen(%s)", argv[3]);
@@ -140,2 +187,0 @@ int main(int argc, char *argv[])
-       if ((epf = fopen(argv[3], "rb")) == NULL)
-               err(1, "fopen(%s)", argv[3]);
@@ -148,3 +193,0 @@ int main(int argc, char *argv[])
-       oldfd = open(argv[1], O_RDONLY | O_BINARY, 0);
-       if (oldfd < 0)
-               err(1, "%s", argv[1]);
@@ -218,3 +260,0 @@ int main(int argc, char *argv[])
-       newfd = open(argv[2], O_CREAT | O_TRUNC | O_WRONLY | O_BINARY, 0666);
-       if (newfd < 0)
-               err(1, "%s", argv[2]);
```

cap_enter()

bspatch(1) - Step 2: read more code

```
if ((f = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);

if (fread(header, 1, 32, f) < 32) {
        if (feof(f))
                errx(1, "Corrupt patch\n");
        err(1, "fread(%s)", argv[3]);
}

if (memcmp(header, "BSDIFF40", 8) != 0)
        errx(1, "Corrupt patch\n");

bzctrllen = offtin(header + 8);
bzdatalen = offtin(header + 16);
newsize = offtin(header + 24);
if ((bzctrllen < 0) || (bzdatalen < 0) || (newsize < 0))
        errx(1, "Corrupt patch\n");

if (fclose(f))
        err(1, "fclose(%s)", argv[3]);
if ((cpf = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);
if (fseeko(cpf, 32, SEEK_SET))
        err(1, "fseeko(%s, %lld)", argv[3],
            (long long)32);
```

```
if ((cpfbz2 = BZ2_bzReadOpen(&cbz2err, cpf, 0, 0, NULL, 0)) == NULL)
        errx(1, "BZ2_bzReadOpen, bz2err = %d", cbz2err);
if ((dpf = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);
if (fseeko(dpf, 32 + bzctrllen, SEEK_SET))
        err(1, "fseeko(%s, %lld)", argv[3],
            (long long)(32 + bzctrllen));
if ((dpfbz2 = BZ2_bzReadOpen(&dbz2err, dpf, 0, 0, NULL, 0)) == NULL)
        errx(1, "BZ2_bzReadOpen, bz2err = %d", dbz2err);
if ((epf = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);
if (fseeko(epf, 32 + bzctrllen + bzdatalen, SEEK_SET))
        err(1, "fseeko(%s, %lld)", argv[3],
            (long long)(32 + bzctrllen + bzdatalen));
if ((epfbz2 = BZ2_bzReadOpen(&ebz2err, epf, 0, 0, NULL, 0)) == NULL)
        errx(1, "BZ2_bzReadOpen, bz2err = %d", ebz2err);
```

bspatch(1) - Step 2: read more code

```
if ((f = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);

if (fread(header, 1, 32, f) < 32) {
        if (feof(f))
                errx(1, "Corrupt patch\n");
        err(1, "fread(%s)", argv[3]);
}

if (memcmp(header, "BSDIFF40", 8) != 0)
        errx(1, "Corrupt patch\n");

bzctrllen = offtin(header + 8);
bzdatalen = offtin(header + 16);
newsize = offtin(header + 24);
if ((bzctrllen < 0) || (bzdatalen < 0) || (newsize < 0))
        errx(1, "Corrupt patch\n");

if (fclose(f))
        err(1, "fclose(%s)", argv[3]);
if ((cpf = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);
if (fseeko(cpf, 32, SEEK_SET))
        err(1, "fseeko(%s, %lld)", argv[3],
            (long long)32);
```

```
if ((cpfbz2 = BZ2_bzReadOpen(&cbz2err, cpf, 0, 0, NULL, 0)) == NULL)
        errx(1, "BZ2_bzReadOpen, bz2err = %d", cbz2err);
if ((dpf = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);
if (fseeko(dpf, 32 + bzctrllen, SEEK_SET))
        err(1, "fseeko(%s, %lld)", argv[3],
            (long long)(32 + bzctrllen));
if ((dpfbz2 = BZ2_bzReadOpen(&dbz2err, dpf, 0, 0, NULL, 0)) == NULL)
        errx(1, "BZ2_bzReadOpen, bz2err = %d", dbz2err);
if ((epf = fopen(argv[3], "rb")) == NULL)
        err(1, "fopen(%s)", argv[3]);
if (fseeko(epf, 32 + bzctrllen + bzdatalen, SEEK_SET))
        err(1, "fseeko(%s, %lld)", argv[3],
            (long long)(32 + bzctrllen + bzdatalen));
if ((epfbz2 = BZ2_bzReadOpen(&ebz2err, epf, 0, 0, NULL, 0)) == NULL)
        errx(1, "BZ2_bzReadOpen, bz2err = %d", ebz2err);
```

bspatch(1) - Step 3: Capsicumize

```
@@ -82,0 +95,3 @@ int main(int argc, char *argv[])
+#ifdef HAVE_CAPSICUM
+	cap_rights_t rights_ro, rights_wr;
+#endif
@@ -90,0 +105,17 @@ int main(int argc, char *argv[])
+#ifdef HAVE_CAPSICUM
+	if (cap_enter() < 0 &&errno != ENOSYS) {
+		err(1, "failed to enter security sandbox");
+	} else {
+		cap_rights_init(&rights_ro, CAP_READ, CAP_FSTAT, CAP_SEEK);
+		cap_rights_init(&rights_wr, CAP_WRITE);
+
+		if (cap_rights_limit(fileno(f), &rights_ro) < 0 ||
+		    cap_rights_limit(fileno(cpf), &rights_ro) < 0 ||
+		    cap_rights_limit(fileno(dpf), &rights_ro) < 0 ||
+		    cap_rights_limit(fileno(epf), &rights_ro) < 0 ||
+		    cap_rights_limit(oldfd, &rights_ro) < 0 ||
+		    cap_rights_limit(newfd, &rights_wr) < 0)
+			err(1, "cap_rights_limit() failed, could not restrict"
+			    " capabilities");
+	}
+#endif
```

bspatch(1) - Step 3: Capsicumize

```
@@ -82,0 +95,3 @@ int main(int argc, char *argv[])
+#ifdef HAVE_CAPSICUM
+	cap_rights_t rights_ro, rights_wr;
+#endif
@@ -90,0 +105,17 @@ int main(int argc, char *argv[])
+#ifdef HAVE_CAPSICUM
+	if (cap_enter() < 0 &&errno != ENOSYS) {
+		err(1, "failed to enter security sandbox");
+	} else {
+		cap_rights_init(&rights_ro, CAP_READ, CAP_FSTAT, CAP_SEEK);
+		cap_rights_init(&rights_wr, CAP_WRITE);
+
+		if (cap_rights_limit(fileno(f), &rights_ro) < 0 ||
+		    cap_rights_limit(fileno(cpf), &rights_ro) < 0 ||
+		    cap_rights_limit(fileno(dpf), &rights_ro) < 0 ||
+		    cap_rights_limit(fileno(epf), &rights_ro) < 0 ||
+		    cap_rights_limit(oldfd, &rights_ro) < 0 ||
+		    cap_rights_limit(newfd, &rights_wr) < 0)
+			err(1, "cap_rights_limit() failed, could not restrict"
+			    " capabilities");
+	}
+#endif
```

cmp(1) - deduplicate code

```
@@ -148,2 +154,33 @@ main(int argc, char *argv[])
+       cap_rights_init(&rights, CAP_FCNTL, CAP_FSTAT, CAP_MMAP_R);
+       if (cap_rights_limit(fd1, &rights) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit rights for %s", file1);
+       if (cap_rights_limit(fd2, &rights) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit rights for %s", file2);
+
+       /* Required for fdopen(3). */
+       fcntls = CAP_FCNTL_GETFL;
+       if (cap_fcntls_limit(fd1, fcntls) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit fcntls for %s", file1);
+       if (cap_fcntls_limit(fd2, fcntls) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit fcntls for %s", file2);
+
+       cap_rights_init(&rights, CAP_FSTAT, CAP_WRITE, CAP_IOCTL);
+       if (cap_rights_limit(STDOUT_FILENO, &rights) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit rights for stdout");
+
+       /* Required for printf(3) via isatty(3). */
+       cmd = TIOCGETA;
+       if (cap_ioctls_limit(STDOUT_FILENO, &cmd, 1) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit ioctls for stdout");
+
+       /*
+        * Cache NLS data, for strerror, for err(3), before entering capability
+        * mode.
+        */
+       (void)catopen("libc", NL_CAT_LOCALE);
```

cmp(1) - deduplicate code

```
@@ -148,2 +154,33 @@ main(int argc, char *argv[])
+       cap_rights_init(&rights, CAP_FCNTL, CAP_FSTAT, CAP_MMAP_R);
+       if (cap_rights_limit(fd1, &rights) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit rights for %s", file1);
+       if (cap_rights_limit(fd2, &rights) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit rights for %s", file2);
+
+       /* Required for fdopen(3). */
+       fcntls = CAP_FCNTL_GETFL;
+       if (cap_fcntls_limit(fd1, fcntls) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit fcntls for %s", file1);
+       if (cap_fcntls_limit(fd2, fcntls) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit fcntls for %s", file2);
+
+       cap_rights_init(&rights, CAP_FSTAT, CAP_WRITE, CAP_IOCTL);
+       if (cap_rights_limit(STDOUT_FILENO, &rights) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit rights for stdout");
+
+       /* Required for printf(3) via isatty(3). */
+       cmd = TIOCGETA;
+       if (cap_ioctls_limit(STDOUT_FILENO, &cmd, 1) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit ioctls for stdout");
+
+       /*
+        * Cache NLS data, for strerror, for err(3), before entering capability
+        * mode.
+        */
+       (void)catopen("libc", NL_CAT_LOCALE);
```

cmp(1) - deduplicate code

```
@@ -148,2 +154,33 @@ main(int argc, char *argv[])
+       cap_rights_init(&rights, CAP_FCNTL, CAP_FSTAT, CAP_MMAP_R);
+       if (cap_rights_limit(fd1, &rights) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit rights for %s", file1);
+       if (cap_rights_limit(fd2, &rights) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit rights for %s", file2);
+
+       /* Required for fdopen(3). */
+       fcntls = CAP_FCNTL_GETFL;
+       if (cap_fcntls_limit(fd1, fcntls) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit fcntls for %s", file1);
+       if (cap_fcntls_limit(fd2, fcntls) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit fcntls for %s", file2);
+
+       cap_rights_init(&rights, CAP_FSTAT, CAP_WRITE, CAP_IOCTL);
+       if (cap_rights_limit(STDOUT_FILENO, &rights) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit rights for stdout");
+
+       /* Required for printf(3) via isatty(3). */
+       cmd = TIOCGETA;
+       if (cap_ioctls_limit(STDOUT_FILENO, &cmd, 1) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit ioctls for stdout");
+
+       /*
+        * Cache NLS data, for strerror, for err(3), before entering capability
+        * mode.
+        */
+       (void)catopen("libc", NL_CAT_LOCALE);
```

Capsicum helpers

- capsicum_helpers.h
- Inline functions:
  - caph_limit_stream()
  - caph_limit_stdout()
  - caph_limit_stdin()
  - caph_limit_stderr()

libc is not your friend

- err(3)
- localtime(3)
- syslog

- Modify virtual dynamic shared object (vdso) to not open device
- More capsicum helpers:
  - caph_cache_catpages()
  - caph_cache_tzdata()

cmp(1) - deduplicate code

```
@@ -148,2 +154,17 @@ main(int argc, char *argv[])
+       cap_rights_init(&rights, CAP_FCNTL, CAP_FSTAT, CAP_MMAP_R);
+       if (cap_rights_limit(fd1, &rights) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit rights for %s", file1);
+       if (cap_rights_limit(fd2, &rights) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit rights for %s", file2);
+
+       /* Required for fdopen(3). */
+       fcntls = CAP_FCNTL_GETFL;
+       if (cap_fcntls_limit(fd1, fcntls) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit fcntls for %s", file1);
+       if (cap_fcntls_limit(fd2, fcntls) < 0 && errno != ENOSYS)
+               err(ERR_EXIT, "unable to limit fcntls for %s", file2);
+
+       if (caph_limit_stdout() == -1)
+               err(ERR_EXIT, "unable to limit stdio");
+
+       caph_cache_catpages();
```

Debugging infrastructure

Debugging - ktrace

- ktrace/kdump
- Getting only trace

- Very easy to miss something
- Hard to cover all paths

Debugging - ktrace

```
802 random  CALL  cap_enter
802 random  RET   cap_enter 0
802 random  CALL  openat(AT_FDCWD,0x400877,0<O_RDONLY>)
802 random  CAP   restricted VFS lookup
802 random  RET   openat -1 errno 94 Not permitted in capability mode
802 random  CALL  sigprocmask(SIG_BLOCK,0x8008209c8,0x7ffffffe640)
802 random  RET   sigprocmask 0
802 random  CALL  sigprocmask(SIG_SETMASK,0x8008209dc,0)
802 random  RET   sigprocmask 0
802 random  CALL  sigprocmask(SIG_BLOCK,0x8008209c8,0x7ffffffe1b0)
```

Debugging - enotcap

- kern.trap_enotcap
- procctl(PROC_TRAPCAP_CTL)
- Getting core dump

- Hard to miss something
- Hard to cover all paths

Debugging - enotcap

```
Program received signal SIGTRAP, Trace/breakpoint trap.
0x0000000080090b34a in _openat () from /lib/libc.so.7
Current language:  auto; currently minimal
Breakpoint 1 at 0x80090b34a
(gdb) bt
#0  0x0000000080090b34a in _openat () from /lib/libc.so.7
#1  0x0000000080086e457 in open (path=<value optimized out>,
flags=<value optimized out>)
    at /usr/src/lib/libc/sys/open.c:57
#2  0x0000000000400a18 in main () at a.c:24
```

libCasper

Casper

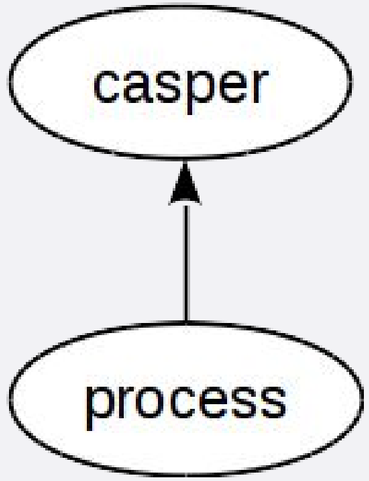
- Provides functionality not available in capability mode through convenient APIs making Capsicum more practical

- Make easier to separate process
- Done before entering Capability mode
- Creating zygote
- Set of dynamic libraries

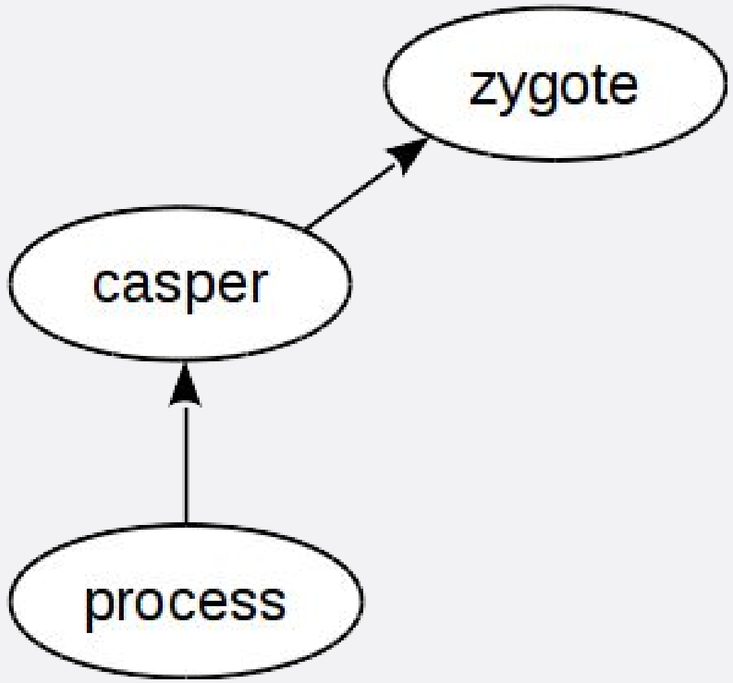
Casper - how its works?

Casper - how its works?

- cap_init()
- cap_service_open()
- cap_close()

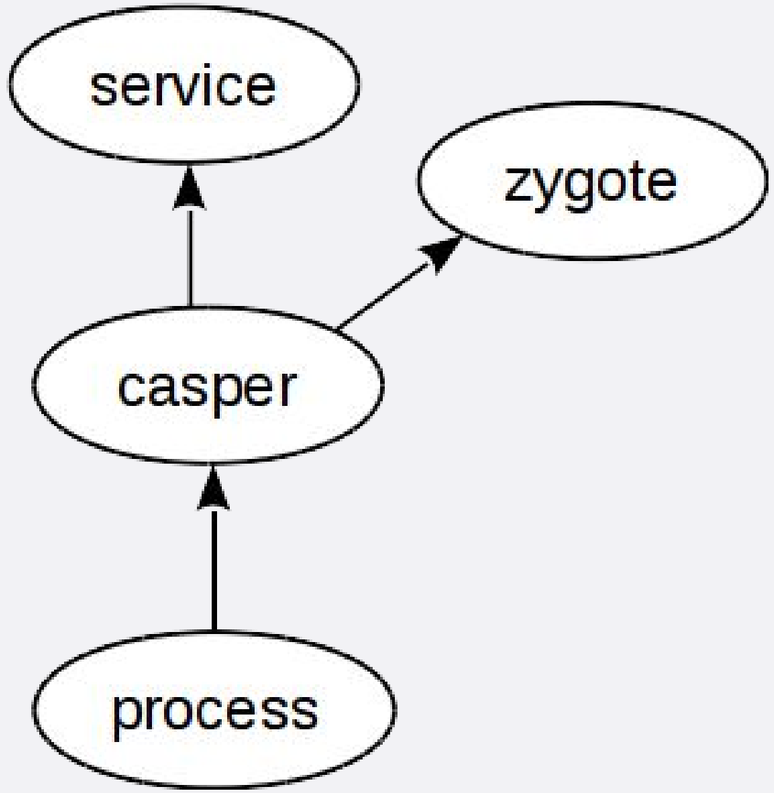


Casper - how its works?



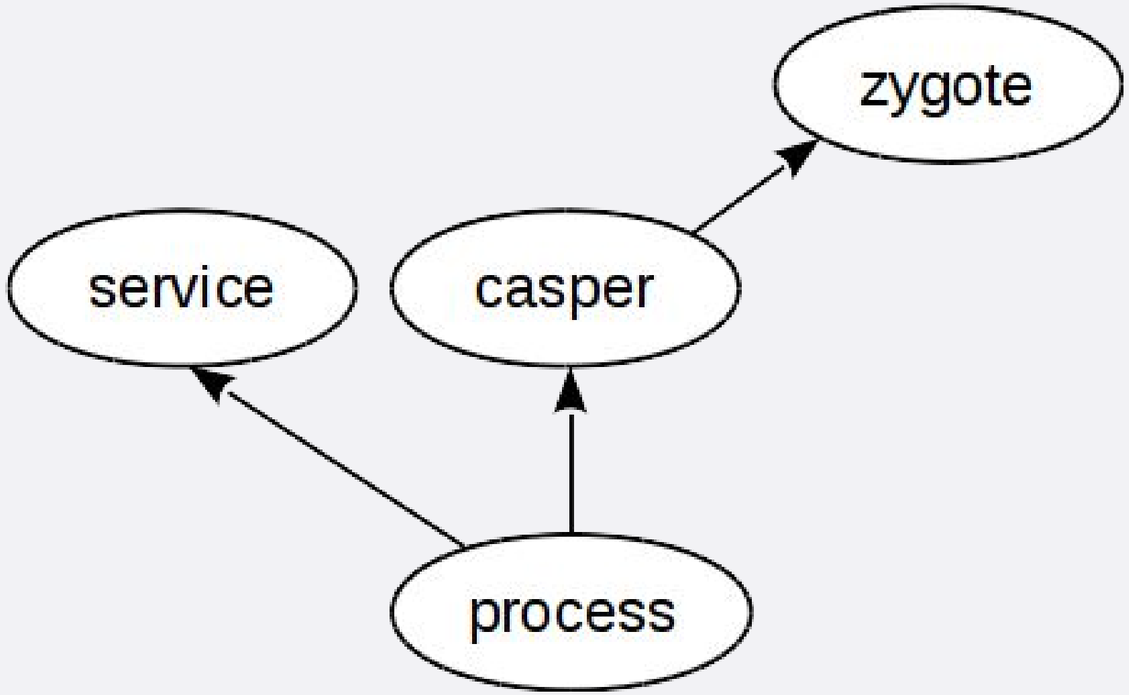
- cap_init()
- cap_service_open()
- cap_close()

Casper - how its works?



- cap_init()
- cap_service_open()
- cap_close()

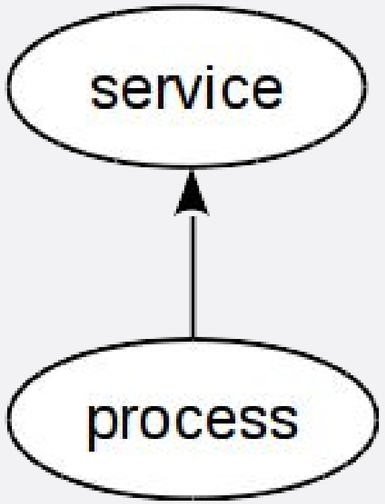
Casper - how its works?



- cap_init()
- cap_service_open()
- cap_close()

Casper - how its works?

- cap_init()
- cap_service_open()
- cap_close()



Casper

- system.dns
- system.grp
- system.pwd
- system.random
- system.sysctl

Traceroute - Capsicumize with Casper

```
+#ifdef HAVE_LIBCASPER
+       const char *types[] = { "NAME", "ADDR" };
+       casper = cap_init();
+       if (casper == NULL)
+               errx(1, "unable to create casper process");
+       capdns = cap_service_open(casper, "system.dns");
+       if (capdns == NULL)
+               errx(1, "unable to open system.dns service");
+       if (cap_dns_type_limit(capdns, types, 2) < 0)
+               errx(1, "unable to limit access to system.dns service");
+       families[0] = AF_INET;
+       if (cap_dns_family_limit(capdns, families, 1) < 0)
+               errx(1, "unable to limit access to system.dns service");
+       cap_close(casper);
+#endif /* HAVE_LIBCASPER */
```

Traceroute - Capsicumize with Casper

```
+#ifdef HAVE_LIBCASPER
+       cansandbox = true;
+#else
+       if (nflag)
+               cansandbox = true;
+       else
+               cansandbox = false;
+#endif
+       if (cansandbox && cap_enter() < 0) {
+               Fprintf(stderr, "%s: cap_enter: %s\n", prog, strerror(errno));
+               exit(1);
+       }
+
+       cap_rights_init(&rights, CAP_SEND, CAP_SETSOCKOPT);
+       if (cansandbox && cap_rights_limit(sndsock, &rights) < 0) {
+               Fprintf(stderr, "%s: cap_rights_limit sndsock: %s\n", prog,
+                   strerror(errno));
+               exit(1);
+       }
+
+       cap_rights_init(&rights, CAP_RECV, CAP_EVENT);
+       if (cansandbox && cap_rights_limit(s, &rights) < 0) {
+               Fprintf(stderr, "%s: cap_rights_limit s: %s\n", prog,
+                   strerror(errno));
+               exit(1);
+       }
```

Traceroute - Capsicumize with Casper

```
@@ -1770,7 +1843,12 @@ inetname(struct in_addr in)
                else {
                        cp = strchr(domain, '.');
                        if (cp == NULL) {
-                               hp = gethostbyname(domain);
+#ifdef HAVE_LIBCASPER
+                               if (capdns != NULL)
+                                       hp = cap_gethostbyname(capdns, domain);
+                               else
+#endif
+                                       hp = gethostbyname(domain);
                                if (hp != NULL)
                                        cp = strchr(hp->h_name, '.');
                        }
```

Casper - mocks

- Reduce amount of ifdefs in code
- Hide ifdefs in library itself
- Use inline/defines to create mocks.

```
#ifdef WITH_CASPER
struct hostent *cap_gethostbyname(cap_channel_t *chan, const char *name);
#else
#define cap_gethostbyname(chan, name)            gethostbyname(name)
#endif
```

Future!

Casper - next next generation!?

- Integration with libc?
  - Make libc more plugable
  - Start casper in _start

- Sandbox services
  - Services run with user privileges
  - Reduce TCB

Casper services

- system.dns
- system.grp
- system.pwd
- system.random
- system.sysctl
- system.filesystem
- system.syslog
- system.login
- system.tls
- system.socket
- system.configuration

Casper - dhclient(8)

```
Starting devd.
Starting dhclient.
pid 336 (dhclient), uid (65):  Path `/var/crash/dhclient.65.0.core' failed on initial open
test, error = 2
pid 336 (dhclient), uid 65: exited on signal 5
Trace/BPT trap
/etc/rc.d/dhclient: WARNING: failed to start dhclient
add host 127.0.0.1: gateway lo0 fib 0: route already in table
Script /etc/rc.d/defaultroute interrupted
Creating and/or trimming log files.
Starting syslogd.
```

Casper - dhclient(8)

```
Starting devd.
Starting dhclient.
pid 336 (dhclient), uid (65):  Path `/var/crash/dhclient.65.0.core' failed on initial open
test, error = 2
pid 336 (dhclient), uid 65: exited on signal 5
Trace/BPT trap
/etc/rc.d/dhclient: WARNING: failed to start dhclient
add host 127.0.0.1: gateway lo0 fib 0: route already in table
Script /etc/rc.d/defaultroute interrupted
Creating and/or trimming log files.
Starting syslogd.
```

Casper - dhclient(8)

```
Starting program: /sbin/dhclient vtnet1

Program received signal SIGTRAP, Trace/breakpoint trap.
0x0000000800bbdd1a in _connect () from /lib/libc.so.7
Current language:  auto; currently minimal
(gdb) bt
#0  0x0000000800bbdd1a in _connect () from /lib/libc.so.7
#1  0x0000000800bb0499 in connectlog ()
    at /usr/home/oshogbo/git/freebsd/lib/libc/gen/syslog.c:379
#2  0x0000000800bb0090 in vsyslog (pri=<value optimized out>,
    fmt=<value optimized out>, ap=0x7fffffffe9c0)
    at /usr/home/oshogbo/git/freebsd/lib/libc/gen/syslog.c:254
#3  0x0000000800bafcdd in syslog (pri=<value optimized out>,
    fmt=<value optimized out>)
    at /usr/home/oshogbo/git/freebsd/lib/libc/gen/syslog.c:128
#4  0x000000000040cf7b in note (fmt=0x41056d "")
    at /usr/home/oshogbo/git/freebsd/sbin/dhclient/errwarn.c:132
#5  0x0000000000405178 in send_discover (ipp=0x80066a000)
    at /usr/home/oshogbo/git/freebsd/sbin/dhclient/dhclient.c:1285
#6  0x00000000004037a2 in main (argc=<value optimized out>, argv=<value optimized out>)
```

Casper - dhclient(8)

```
Starting program: /sbin/dhclient vtnet1

Program received signal SIGTRAP, Trace/breakpoint trap.
0x0000000800bbdd1a in _connect () from /lib/libc.so.7
Current language:  auto; currently minimal
(gdb) bt
#0  0x0000000800bbdd1a in _connect () from /lib/libc.so.7
#1  0x0000000800bb0499 in connectlog ()
    at /usr/home/oshogbo/git/freebsd/lib/libc/gen/syslog.c:379
#2  0x0000000800bb0090 in vsyslog (pri=<value optimized out>,
    fmt=<value optimized out>, ap=0x7fffffffe9c0)
    at /usr/home/oshogbo/git/freebsd/lib/libc/gen/syslog.c:254
#3  0x0000000800bafcdd in syslog (pri=<value optimized out>,
    fmt=<value optimized out>)
    at /usr/home/oshogbo/git/freebsd/lib/libc/gen/syslog.c:128
#4  0x000000000040cf7b in note (fmt=0x41056d "")
    at /usr/home/oshogbo/git/freebsd/sbin/dhclient/errwarn.c:132
#5  0x0000000000405178 in send_discover (ipp=0x80066a000)
    at /usr/home/oshogbo/git/freebsd/sbin/dhclient/dhclient.c:1285
#6  0x00000000004037a2 in main (argc=<value optimized out>, argv=<value optimized out>)
```

Casper - dhclient(8)

```
359         /* Initially, log errors to stderr as well as to syslogd. */
360         openlog(__progname, LOG_PID | LOG_NDELAY, DHCPD_LOG_FACILITY);
361         setlogmask(LOG_UPTO(LOG_DEBUG));

521         if (cap_enter() < 0 && errno != ENOSYS)
522                 error("can't enter capability mode: %m");
```

Casper - dhclient(8)

```
359         /* Initially, log errors to stderr as well as to syslogd. */
360         openlog(__progname, LOG_PID | LOG_NDELAY, DHCPD_LOG_FACILITY);
361         setlogmask(LOG_UPTO(LOG_DEBUG));

521         if (cap_enter() < 0 && errno != ENOSYS)
522                 error("can't enter capability mode: %m");
```

```
void openlog(const char *ident, int logopt, int facility);
```

The routines closelog(), openlog(), syslog() and vsyslog() return no value.

Casper - dhclient(8)

```
Starting devd.
Starting dhclient.
pid 336 (dhclient), uid (65):  Path `/var/crash/dhclient.65.0.core' failed on initial open
test, error = 2
pid 336 (dhclient), uid 65: exited on signal 5
Trace/BPT trap
/etc/rc.d/dhclient: WARNING: failed to start dhclient
add host 127.0.0.1: gateway lo0 fib 0: route already in table
Script /etc/rc.d/defaultroute interrupted
Creating and/or trimming log files.
Starting syslogd.
```

Casper - dhclient(8)

```
Starting devd.
Starting dhclient.
pid 336 (dhclient), uid (65):  Path `/var/crash/dhclient.65.0.core' failed on initial open
test, error = 2
pid 336 (dhclient), uid 65: exited on signal 5
Trace/BPT trap
/etc/rc.d/dhclient: WARNING: failed to start dhclient
add host 127.0.0.1: gateway lo0 fib 0: route already in table
Script /etc/rc.d/defaultroute interrupted
Creating and/or trimming log files.
Starting syslogd.
```

Casper - system.syslog

- Change the order?
- Casper service:
  - https://reviews.freebsd.org/D12824
- Fixed version of dhclient
  - https://reviews.freebsd.org/D12825

Casper - sshd(8)

```
$ ssh 192.168.0.151
Connection to 192.168.0.151 closed by remote host.
Connection to 192.168.0.151 closed.


# dmesg
pid 99355 (sshd), uid (22):  Path `/var/crash/sshd.22.0.core' failed on initial open test,
error = 2
pid 99355 (sshd), uid 22: exited on signal 5
```

Casper - sshd(8)

Casper - sshd(8)

```
#ifdef HAVE_LOGIN_CAP
        if (authctxt->pw != NULL &&
            (lc = login_getpwclass(authctxt->pw)) != NULL) {
                logit("user %s login class %s", authctxt->pw->pw_name,
                    authctxt->pw->pw_class);
                from_host = auth_get_canonical_hostname(ssh, options.use_dns);
                from_ip = ssh_remote_ipaddr(ssh);
                if (!auth_hostok(lc, from_host, from_ip)) {
                        logit("Denied connection for %.200s from %.200s [%.200s].",
                            authctxt->pw->pw_name, from_host, from_ip);
                        packet_disconnect("Sorry, you are not allowed to connect.");
                }
                if (!auth_timeok(lc, time(NULL))) {
                        logit("LOGIN %.200s REFUSED (TIME) FROM %.200s",
                            authctxt->pw->pw_name, from_host);
                        packet_disconnect("Logins not available right now.");
                }
                login_close(lc);
        }
#endif  /* HAVE_LOGIN_CAP */
```

Casper - sshd(8)

```
#ifdef HAVE_LOGIN_CAP
        if (authctxt->pw != NULL &&
            (lc = login_getpwclass(authctxt->pw)) != NULL) {
                logit("user %s login class %s", authctxt->pw->pw_name,
                    authctxt->pw->pw_class);
                from_host = auth_get_canonical_hostname(ssh, options.use_dns);
                from_ip = ssh_remote_ipaddr(ssh);
                if (!auth_hostok(lc, from_host, from_ip)) {
                        logit("Denied connection for %.200s from %.200s [%.200s].",
                            authctxt->pw->pw_name, from_host, from_ip);
                        packet_disconnect("Sorry, you are not allowed to connect.");
                }
                if (!auth_timeok(lc, time(NULL))) {
                        logit("LOGIN %.200s REFUSED (TIME) FROM %.200s",
                            authctxt->pw->pw_name, from_host);
                        packet_disconnect("Logins not available right now.");
                }
                login_close(lc);
        }
#endif  /* HAVE_LOGIN_CAP */
```

Casper - sshd(8)

- This code exists only in FreeBSD
- Opens two files $HOME/login.conf and /etc/login.conf
- system.login

Thanks!

- Allan Jude
- Baptiste Daroussin
- Conrad Mayer
- Ed Maste
- Konstantin Belousov

Thank you!