Modern Userland Exploitation

From gets() to pwned.

Preamble

The vulnerabilities are not real world in any way. The focus is on specific exploitation techniques. Think of it as an overly simplified model so that we may focus on other things. With this in mind, the information leak is given to us for free and an obvious stack buffer overflow is present.

Overview

- The vulnerability
- The mitigations
- The plan
- The exploit
- Lessons
- Questions

The Vulnerability

```
...
    gets(buf);
...
```

The Vulnerability

```
$ man 3 gets
...
gets() reads a line from stdin into
the buffer pointed to by s until
either a terminating newline or EOF,
which it replaces with a null byte
('\0').
...
```

The Mitigations

- ASLR - randomizes the base address of loaded libraries, heap, stack, and executable

The Mitigations

Image stolen from:
http://wasntnate.com/2012/05/how-address-space-layout-randomization-aslr-was-beaten/

The Mitigations

- DEP/NX/W^X - Pages in memory can have write permissions or execute permissions

The Mitigations

Getting around ASLR

- Information disclosure
- Using non-randomized structures
- bruteforce

Getting around DEP

- full ROP payload
- modify protections with mprotect()
- create a new region with mmap()

The Plan

- find the base address of a large library
- overwrite %rip
- return to mprotect() to make the stack rwx
- jmp to our, now executable, shellcode

The Exploit

Ingredients:

- info disclosure (bypassing ASLR)
- rop chain (bypassing DEP)
- shellcode (bypassing not executing attacker code)

The Exploit (ASLR)

Info Disclosure (for free)

```
...
   fp = fopen("/tmp/<pid>", "w");
    ...
   fprintf(fp, "%p", dlsym(libc, "send"));
...
```

The Exploit (DEP)

Getting around DEP with gadgets

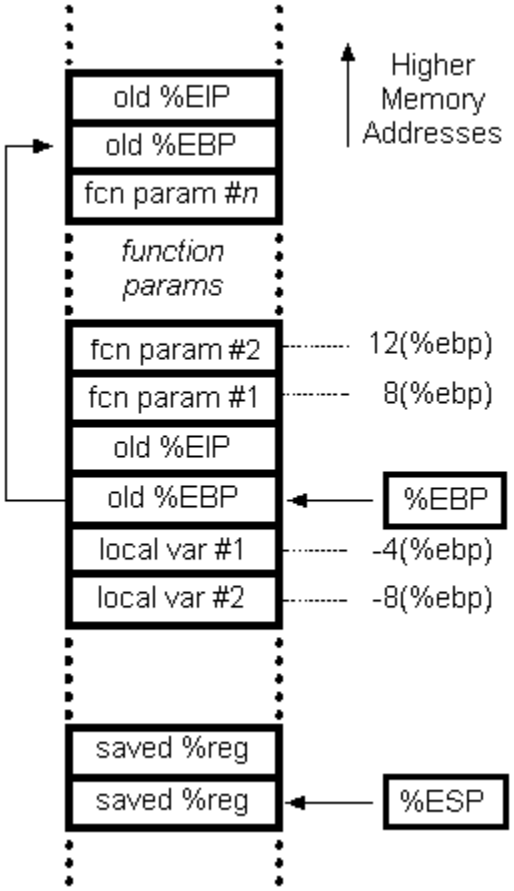
```
$ rp -r 4 -f /lib/libc.so.6
Trying to open '/lib/libc.so.6'..
Loading ELF information..
FileFormat: Elf, Arch: Ia64
Using the Nasm syntax..

Wait a few seconds, rp++ is looking for gadgets..
in PHDR
0 found.

in LOAD
107156 found.
...
```

The Exploit (DEP)

Before we get into using our gadgets, let's take a step back...

Here's a 32-bit call-stack:

The Exploit (DEP)



The Exploit (DEP)

In an exploit:

- We overwrite old %eip
- The function "returns" to old %eip (which we specified)
- We have execution control

The Exploit (DEP)

What if we can't return to shellcode?

The Exploit (DEP)

Why not just use a library (i.e., ret2libc)?

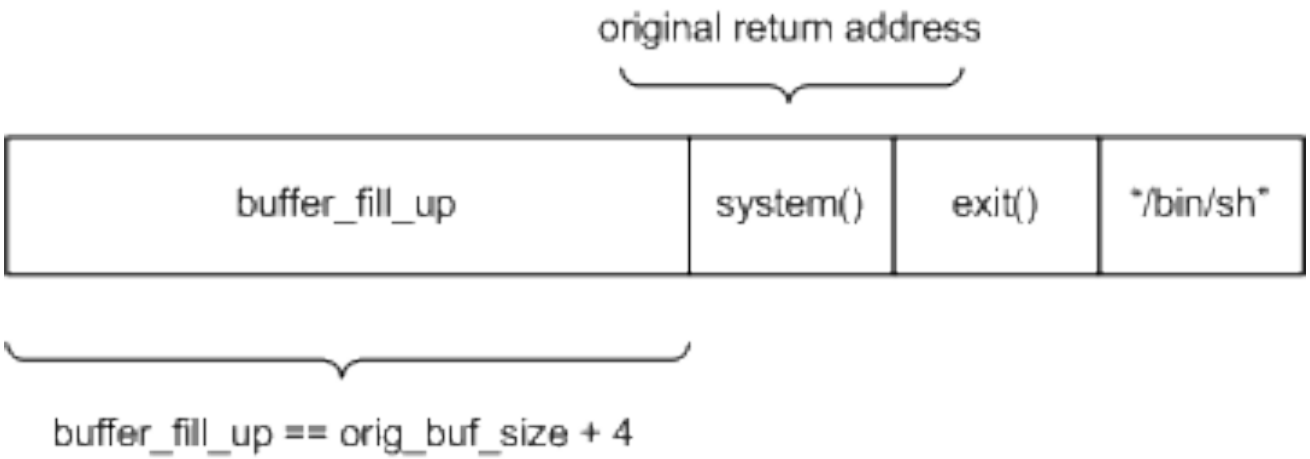


image stolen from: http://www.w00zl3.net/

The Exploit (DEP)

What if there is no library function available that will perform our task?

The Exploit (DEP)

What if there is no library function available that will perform our task?

Do we even HAVE to return to functions?

The Exploit (DEP)

Since we control the stack, we can "return" (jmp) to any location we desire.

The Exploit (DEP)

Since we control the stack, we can "return" (jmp) to any location we desire.

What if we return to a "ret" instruction?

The Exploit (DEP)

Now what if we return to a sequence of instructions ending in ret?

The Exploit (DEP)

Now what if we return to a sequence of instructions ending in ret?

What if we do it again?

The Exploit (DEP)

Now what if we return to a sequence of instructions ending in ret?

What if we do it again?

What if we continue indefinitely?

The Exploit (DEP)

GADGETS!

The Exploit (DEP)

GADGETS!

- Think of %rsp as a makeshift instruction pointer
- intermingles code and data
- makes programming this "weird machine"* interesting

*shout out to pope

The Exploit (DEP)

int mprotect(void *addr, size_t len, int prot)

- void *addr
  - The starting address of the page we wish to modify.
- size_t len
  - How many bytes are we modifying?
- int prot
  - What are we setting the protections to?

The Exploit (DEP)

The calling convention is:

- argument 1: %rdi
- argument 2: %rsi
- argument 3: %rdx

...

The Exploit (DEP)

Our ROP chain must:

- set %rdi to our shellcode's address
- set %rsi to (&shellcode - %rdi) + len(shellcode)
- set %rdx to +rwx (i.e., 7)
- return to &mprotect
- return/call/jmp to our shellcode

The Exploit (shellcode)

- gets() is annoying to our file descriptor
- to get around this we'll use a bindshell

The Exploit (shellcode)

```
sock = socket(AF_INET, SOCK_STREAM, 0);
bind(sock, &servaddr, sizeof(servaddr));
listen(sock, 1);
conn = accept(sock, NULL, NULL);
for (i = 0; i < 3; ++i)
    dup2(conn, i);
execve("/bin/sh", {"sh", "-i", NULL}, NULL);
```

The Exploit (shellcode)

- Now that we know what we're doing, we can talk with the kernel directly instead of going through a library wrapper
- To do this, we need to lookup the syscall numbers

The Exploit (shellcode)

```
$ vi arch/x86/syscalls/syscall_64.tbl
...
41 common  socket          sys_socket
49 common  bind            sys_bind
50 common  listen          sys_listen
43 common  accept          sys_accept
33 common  dup2            sys_dup2
59 64      execve          stub_execve
```

The Exploit

demo

Lessons

- Memory corruption is involved
- Modern mitigations require multiple bugs to gain code execution
- amd64 ABI doesn't require any pop-pop-rets but would be bad news if you lacked register control
- Manually writing ROP is either fun or the worst experience of your life

Questions

Ask away!

References

- http://www.phrack.org/issues.html?issue=49&id=14
- http://www.phrack.org/issues.html?issue=58&id=4
- http://www.phrack.org/issues.html?issue=59&id=9
- http://cseweb.ucsd.edu/~hovav/papers/s07.html
- http://uninformed.org/index.cgi?v=9&a=4
- http://trailofbits.files.wordpress.com/2010/04/practical-rop.pdf
- http://www.suse.de/~krahmer/no-nx.pdf
- http://www.phrack.org/issues.html?issue=67&id=9

References

- http://www.phrack.org/issues.html?issue=57&id=15
- http://www.phrack.org/issues.html?issue=57&id=5
- http://esec-lab.sogeti.com/post/2011/07/05/Linux-syscall-ABI
- http://msdn.microsoft.com/en-us/library/windows/hardware/ff561499(v=vs.85).aspx

Images (don't sue me)

- http://wasntnate.com/2012/05/how-address-space-layout-randomization-aslr-was-beaten/
- http://www.w00zl3.net/

End

all files: http://security.cs.rpi.edu/~candej2

email: napum@demigods.org (may or may not be down)

Notes:

- Hack the planet!
- dancetillyanosebleeds