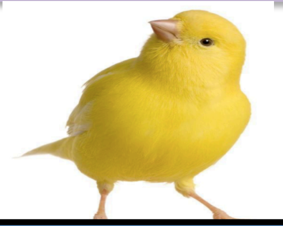
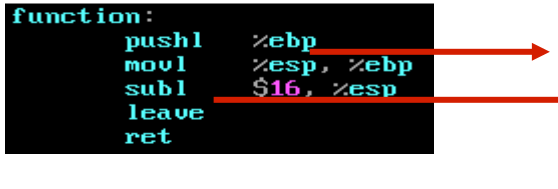
Preventing Buffer Overflows with Canaries and W^X

Canaries



- Known (pseudo random) values placed on stack to monitor buffer overflows.
- A change in the value of the canary indicates a buffer overflow.
- Will cause a 'stack smashing' to be detected



```
function:
        pushl   %ebp
        movl    %esp, %ebp
        subl    $16, %esp
        leave
        ret
```

Insert a canary here

check if the canary value has got modified

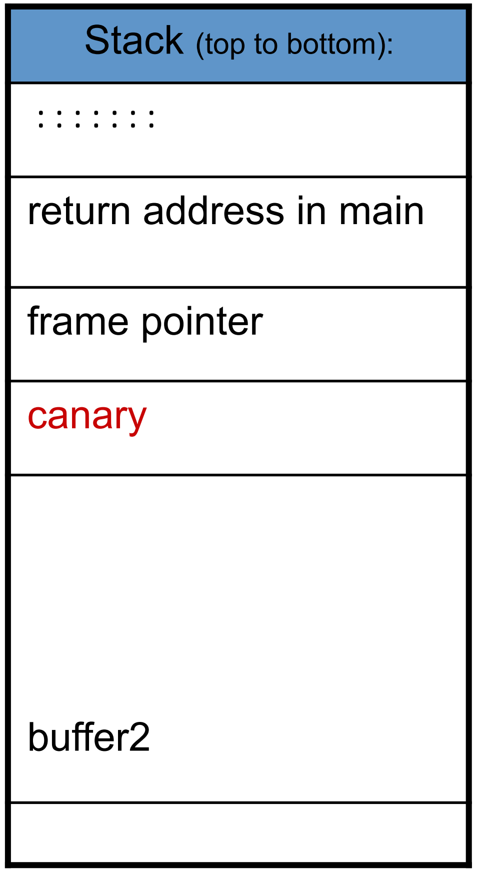
Stack (top to bottom):
stored data
Function parameters
return address
Frame pointer(%ebp)
Insert canary here
buffer1
buffer2

Canaries and gcc

- As on gcc 4.4.5, canaries are not added to functions by default
  - Could cause overheads as they are executed for every function that gets executed
- Canaries can be added into the code by ***–fstack-protector*** option
  - If ***-fstack-protector*** is specified, canaries will get added based on a gcc heuristic
    - For example, buffer of size at-least 8 bytes is allocated
    - Use of string operations such as strcpy, scanf, etc

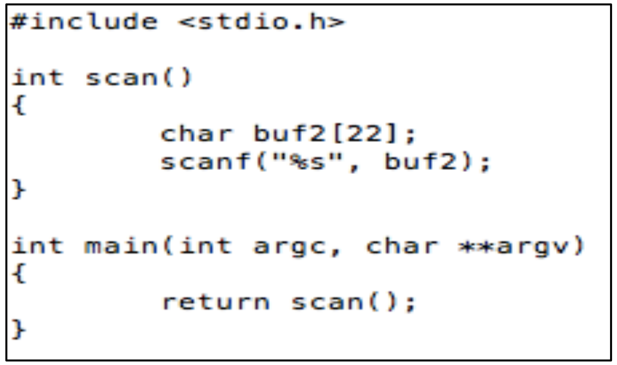
Canaries Example

```
#include <stdio.h>

int scan()
{
        char buf2[22];
        scanf("%s", buf2);
}

int main(int argc, char **argv)
{
        return scan();
}
```

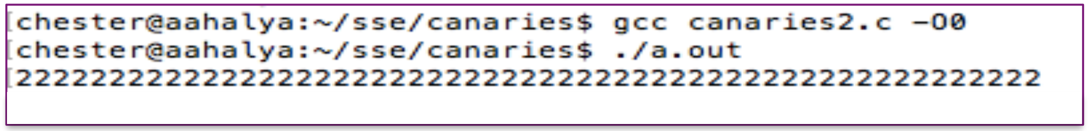


Canaries Example

With canaries, the program gets aborted due to stack smashing.



```
#include <stdio.h>

int scan()
{
        char buf2[22];
        scanf("%s", buf2);
}

int main(int argc, char **argv)
{
        return scan();
}
```

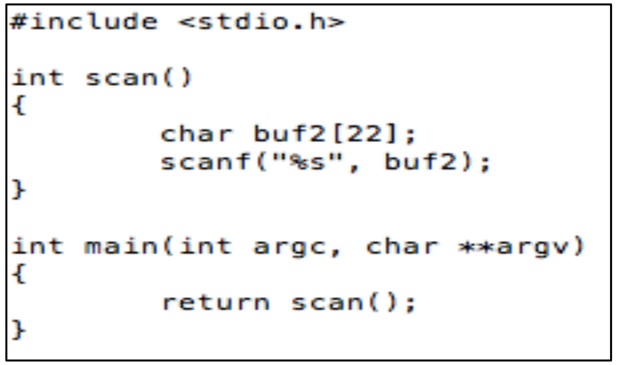


```
chester@aahalya:~/sse/canaries$ gcc canaries2.c -O0
chester@aahalya:~/sse/canaries$ ./a.out
2222222222222222222222222222222222222222222222222222222
```

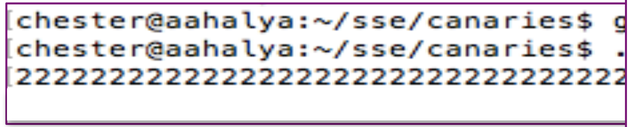
Stack (top to bottom):
: : : : : : :
32323232
32323232
32323232
32323232 32323232 32323232 32323232 32323232

Canaries Example

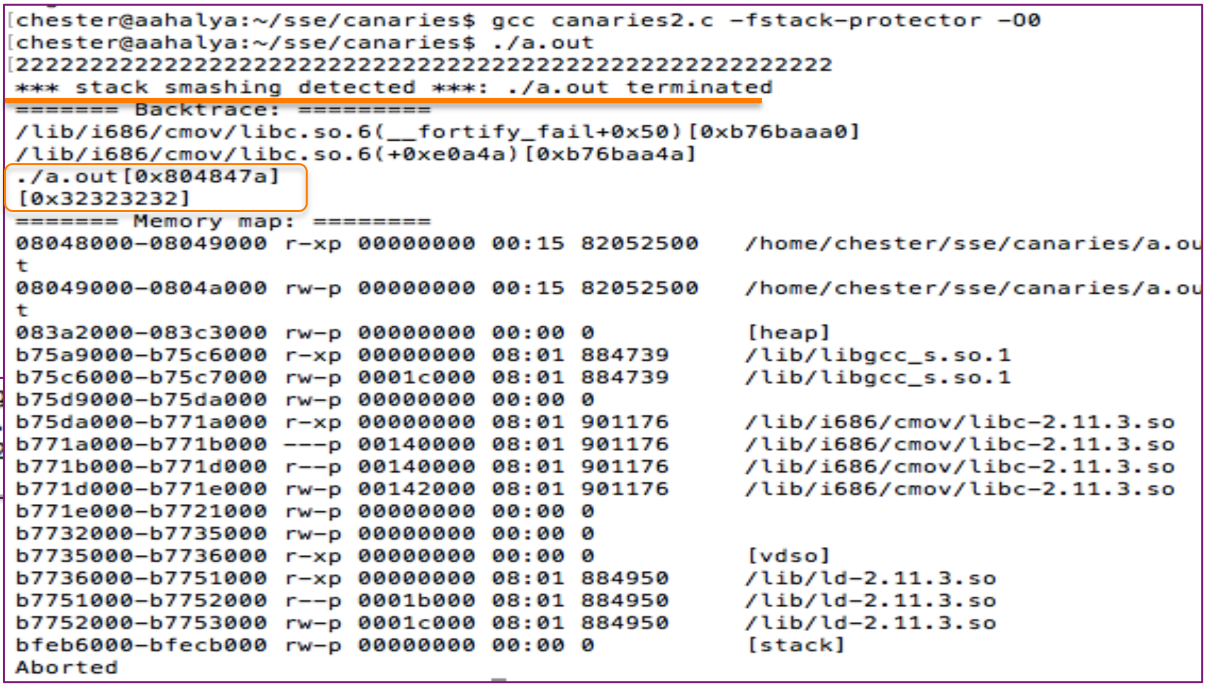
With canaries, the program gets aborted due to stack smashing.



```
#include <stdio.h>

int scan()
{
        char buf2[22];
        scanf("%s", buf2);
}

int main(int argc, char **argv)
{
        return scan();
}
```

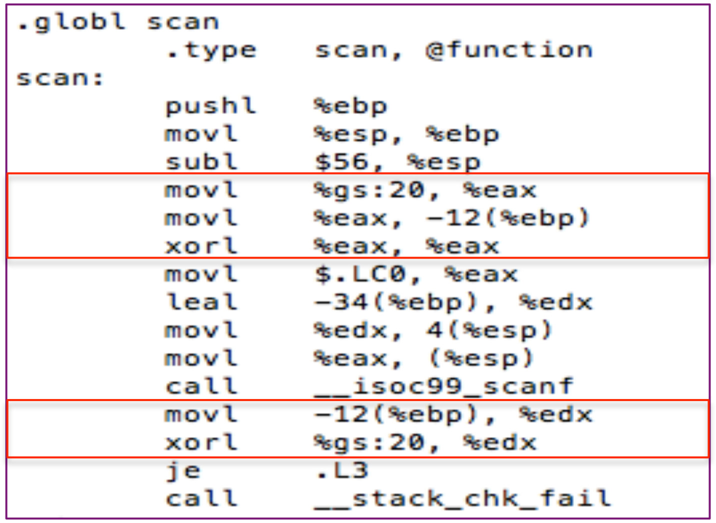


```
chester@aahalya:~/sse/canaries$ g
chester@aahalya:~/sse/canaries$ .
222222222222222222222222222222222
```



```
chester@aahalya:~/sse/canaries$ gcc canaries2.c -fstack-protector -O0
chester@aahalya:~/sse/canaries$ ./a.out
2222222222222222222222222222222222222222222222222222222
*** stack smashing detected ***: ./a.out terminated
======= Backtrace: =========
/lib/i686/cmov/libc.so.6(__fortify_fail+0x50)[0xb76baaa0]
/lib/i686/cmov/libc.so.6(+0xe0a4a)[0xb76baa4a]
./a.out[0x804847a]
[0x32323232]
======= Memory map: ========
08048000-08049000 r-xp 00000000 00:15 82052500   /home/chester/sse/canaries/a.ou
t
08049000-0804a000 rw-p 00000000 00:15 82052500   /home/chester/sse/canaries/a.ou
t
083a2000-083c3000 rw-p 00000000 00:00 0          [heap]
b75a9000-b75c6000 r-xp 00000000 08:01 884739     /lib/libgcc_s.so.1
b75c6000-b75c7000 rw-p 0001c000 08:01 884739     /lib/libgcc_s.so.1
b75d9000-b75da000 rw-p 00000000 00:00 0
b75da000-b771a000 r-xp 00000000 08:01 901176     /lib/i686/cmov/libc-2.11.3.so
b771a000-b771b000 ---p 00140000 08:01 901176     /lib/i686/cmov/libc-2.11.3.so
b771b000-b771d000 r--p 00140000 08:01 901176     /lib/i686/cmov/libc-2.11.3.so
b771d000-b771e000 rw-p 00142000 08:01 901176     /lib/i686/cmov/libc-2.11.3.so
b771e000-b7721000 rw-p 00000000 00:00 0
b7732000-b7735000 rw-p 00000000 00:00 0
b7735000-b7736000 r-xp 00000000 00:00 0          [vdso]
b7736000-b7751000 r-xp 00000000 08:01 884950     /lib/ld-2.11.3.so
b7751000-b7752000 r--p 0001b000 08:01 884950     /lib/ld-2.11.3.so
b7752000-b7753000 rw-p 0001c000 08:01 884950     /lib/ld-2.11.3.so
bfeb6000-bfecb000 rw-p 00000000 00:00 0          [stack]
Aborted
```

Canary Internals

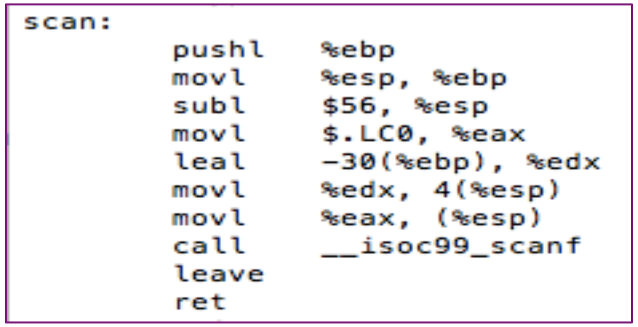


```
.globl scan
        .type   scan, @function
scan:
        pushl   %ebp
        movl    %esp, %ebp
        subl    $56, %esp
        movl    %gs:20, %eax
        movl    %eax, -12(%ebp)
        xorl    %eax, %eax
        movl    $.LC0, %eax
        leal    -34(%ebp), %edx
        movl    %edx, 4(%esp)
        movl    %eax, (%esp)
        call    __isoc99_scanf
        movl    -12(%ebp), %edx
        xorl    %gs:20, %edx
        je      .L3
        call    __stack_chk_fail
```

Store canary onto stack

Verify if the canary has changed

With canaries



```
scan:
        pushl   %ebp
        movl    %esp, %ebp
        subl    $56, %esp
        movl    $.LC0, %eax
        leal    -30(%ebp), %edx
        movl    %edx, 4(%esp)
        movl    %eax, (%esp)
        call    __isoc99_scanf
        leave
        ret
```

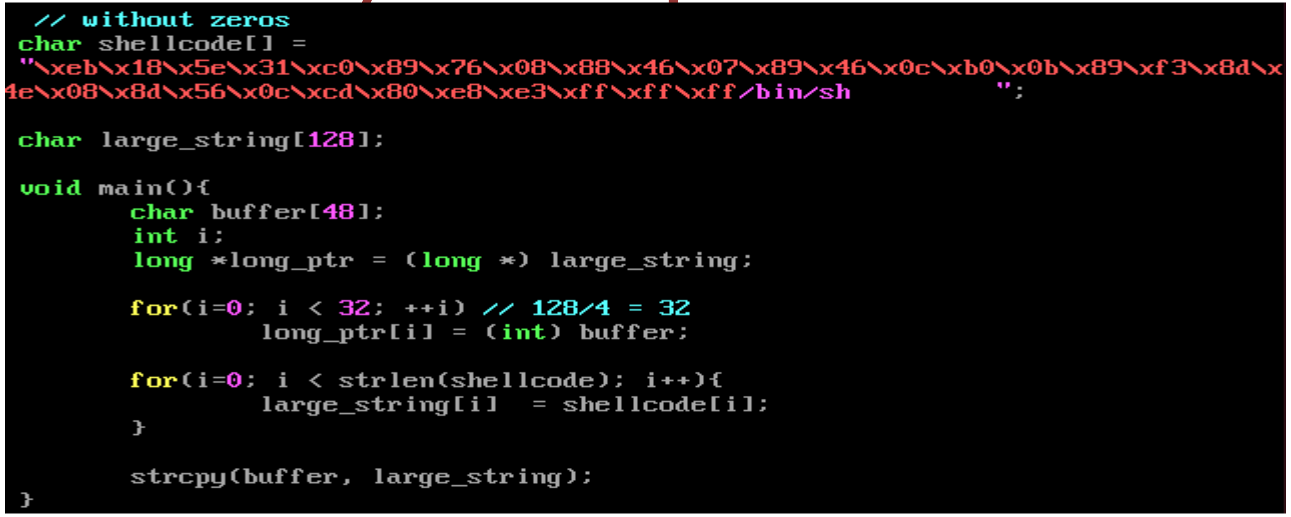
Without canaries

gs is a segment that shows thread local data; in this case it is used for picking out canaries

Non Executable Stacks (W^X)

- In Intel/AMD processors, ND/NX bit present to mark non code regions as non-executable.
  - Exception raised when code in a page marked W^X executes
- Works for most programs
  - Supported by Linux kernel from 2004
  - Supported by Windows XP service pack 1 and Windows Server 2003
    - Called DEP – Data Execution Prevention
- Does not work for some programs that NEED to execute from the stack.
  - Eg. JIT Compiler, constructs assembly code from external data and then executes it.
    (Need to disable the W^X bit, to get this to work)

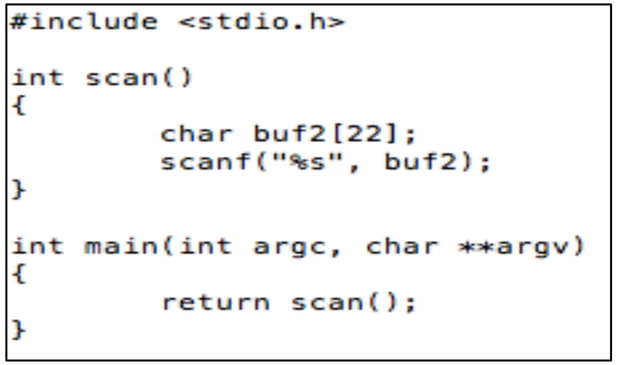
Some Defense Mechanisms already Incorporated



```c
 // without zeros
char shellcode[] =
"\xeb\x18\x5e\x31\xc0\x89\x76\x08\x88\x46\x07\x89\x46\x0c\xb0\x0b\x89\xf3\x8d\x
4e\x08\x8d\x56\x0c\xcd\x80\xe8\xe3\xff\xff\xff/bin/sh          ";

char large_string[128];

void main(){
        char buffer[48];
        int i;
        long *long_ptr = (long *) large_string;

        for(i=0; i < 32; ++i) // 128/4 = 32
                long_ptr[i] = (int) buffer;

        for(i=0; i < strlen(shellcode); i++){
                large_string[i]  = shellcode[i];
        }

        strcpy(buffer, large_string);
}
```

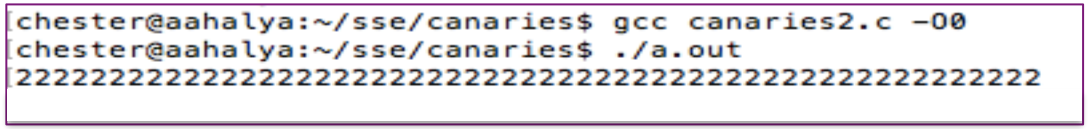
```
bash$ gcc -m32 -fno-stack-protector -z execstack overflow1.c
bash$ ./a.out
$                                (shell created successfully)
```

Points to Ponder



```
#include <stdio.h>

int scan()
{
        char buf2[22];
        scanf("%s", buf2);
}

int main(int argc, char **argv)
{
        return scan();
}
```

What happens to the execution when canaries are not enabled for this program and given the same input below?



```
chester@aahalya:~/sse/canaries$ gcc canaries2.c -O0
chester@aahalya:~/sse/canaries$ ./a.out
2222222222222222222222222222222222222222222222222222222
```