

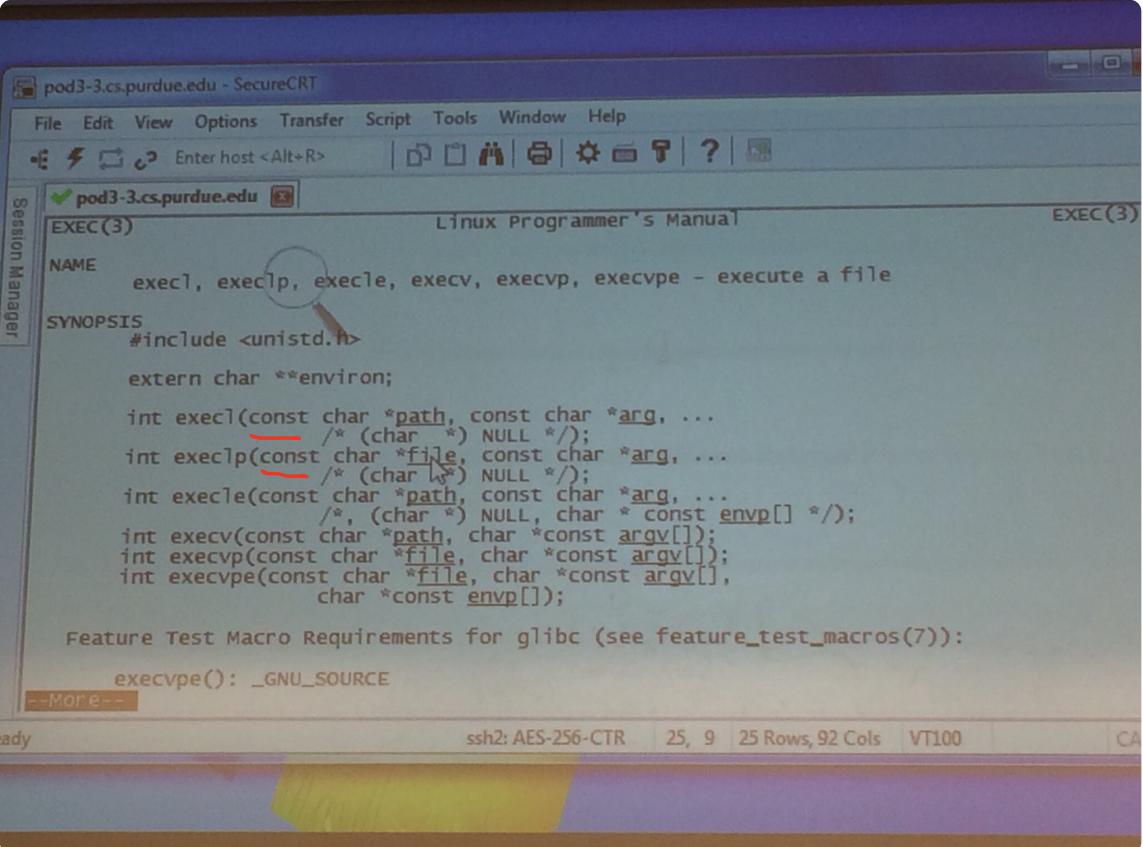
```

main() {
    int a[100];
    abc(a, );
}

```

\downarrow
 const int *

}



pod3-3.cs.purdue.edu - SecureCRT

File Edit View Options Transfer Script Tools Window Help

Session Manager

EXEC(3) Linux Programmer's Manual EXEC(3)

NAME exec, execvp, execle, execv, execvp, execvpe - execute a file

SYNOPSIS

```
#include <unistd.h>
extern char **environ;
```

```
int execl(const char *path, const char *arg, ...
          /* (char *) NULL */);
int execlp(const char *file, const char *arg, ...
          /* (char *) NULL */);
int execle(const char *path, const char *arg, ...
          /* (char *) NULL, char * const argv[] */);
int execv(const char *path, char *const argv[]);
int execvp(const char *file, char *const argv[]);
int execvpe(const char *file, char *const argv[],
            char *const envp[]);
```

Feature Test Macro Requirements for glibc (see `feature_test_macros(7)`):

```
execvpe(): _GNU_SOURCE
```

--More--

ady ssh2: AES-256-CTR 25, 9 25 Rows, 92 Cols VT100 CA

Const

```

File Edit View Options Transfer Script Tools Window
pod3-3.cs.purdue.edu Enter host <Alt+R>
// Illustration of const
#include <stdio.h>
#include <stdlib.h>

main()
{
    const int x = 2;
    int y;
    x = 9;           ← errors
    /*
        printf("%d\n", x);
        scanf("%d\n", &x);
        printf("%d\n", x);
    */
}
~
~
~
~ "main2.c" 20L, 184C
ssh2: AES-256-CBC

```



```

File Edit View Options Transfer Script Tools Window
pod3-3.cs.purdue.edu Enter host <Alt+R>
// Illustration of const
#include <stdio.h>
#include <stdlib.h>

main()
{
    const int x = 2;
    int y;
    // x = 9; ← comment out
    /*
        printf("%d\n", x);
        scanf("%d\n", &x);
        printf("%d\n", x);
    */
}
~
~
~
~ ready
ssh2

```

But when running
 it might cause
 undeclared case,
 Bugs might happen
 might not happen
 depends on compiler.

```
execvp ("ls", "ls" "-a" "-l" NULL)
```

```
printf(" ", 0, 0)
```

$\text{mysum}(2, 2, 2) = 6$

$\text{mysum}(7, 3) = 10$

variable number of arguments

Must have at least one argument ~~one~~

The image shows two terminal sessions on a Linux system (pod3-3.cs.purdue.edu). The left session contains the following code:

```
// example to illustrate different number of arguments
#include <stdio.h>
#include <stdarg.h>

int mysum(int, ...);
int main()
{
    int res;
    res = mysum(4, 2, 2, 1, 5);
    printf("result = %d\n", res);
}

int mysum(int a, ...)
{
    int sum = 0;
    int i;
    va_list arglist;
    va_start(arglist, a);
    for (i=0; i<a; i++)
        sum = sum + va_arg(arglist, int);
    va_end(arglist);
    return sum;
}
```

The right session contains the following code:

```
int res;
res = mysum(4, 2, 2, 1, 5);
printf("result = %d\n", res);

int mysum(int a, ...)
{
    int sum = 0;
    int i;
    va_list arglist;
    va_start(arglist, a);
    for (i=0; i<a; i++)
        sum = sum + va_arg(arglist, int);
    va_end(arglist);
    return sum;
}
```

Red annotations explain the differences:

- A magnifying glass over the 'main' function call in the left session is annotated with "can be char, float, anything".
- An arrow from the 'arglist' parameter in the 'mysum' declaration to the 'arglist' parameter in the 'va_start' call is annotated with "hot using malloc".
- An annotation next to the 'arglist' parameter in the 'mysum' declaration says "compiler allocate mem space in compile time/hard not run time code".
- A large bracket under the 'va_start' and 'va_end' calls is annotated with "Should be between va_start() and va_end()".

calling vararg first time return
the first argument after a .
which is 2 (the second argument)

calling ... second time return

. . . second . . . after a

the third argument , which is 2 .

calll third time return

third . . . after a

the fourth argument which is 1.

. . . .

pod3-3.cs.purdue.edu - SecureCRT

```
// example to illustrate different number of arguments
// a string as in printf()

#include <stdio.h>
#include <stdarg.h>
#include <string.h>

main()
{
    int mysum(char *, ...);
    int res;
    res = mysum("jizdzd", 2, 2, 1, 10, 5, 5);
    printf("result = %d\n", res);
}

int mysum(char *a, ...)
{
    int sum = 0, i;
    int cnt;
    va_list arglist;
    "main4.c" 36L, 530C written
    main4.c:8:1: warning: return type defaults to 'int' [-Wimplicit-int]
    main()
    A
    pod3-3 37 % a.out
    string length: 6
    result = 25
    pod3-3 38 %
    va_end(arglist);

    return sum;
}

```

ssh2: AES-256-CTR 25, 13 25 Rows, 92 Cols

pod3-3.cs.purdue.edu - SecureCRT

```
int res;
res = mysum("jizdzd", 2, 2, 1, 10, 5, 5);
printf("result = %d\n", res);

int mysum(char *a, ...)
{
    int sum = 0, i;
    int cnt;
    va_list arglist;
    va_start(arglist, a);
    cnt = strlen(a);
    printf("string length: %d\n", cnt);
    for (i=0; i<cnt; i++)
        sum += va_arg(arglist, int);
    va_end(arglist);
    return sum;
}
```

ssh2: AES-256-CTR 25, 13 25 Rows, 92 Cols

```
pod3-3.cs.purdue.edu - SecureCRT
File Edit View Options Transfer Script Tools Window Help
Enter host <Alt+R> | [Icons]
Session Manager
Session: pod3-3.cs.purdue.edu
{
    int mysum(char *, ...);
    int res;
    res = mysum("abc", 2, "2", 1);
    printf("result = %d\n", res);
}

int mysum(char *a, ...)
{
    int sum = 0, i;
    int cnt;
    char *s;
    va_list arglist;

    va_start(arglist, a);
    cnt = strlen(a);

    for (i=0; i<cnt; i++) {
        if (i == 1) {
            s = va_arg(arglist, char *);
            sum += atoi(s);
        }
        else
            sum += va_arg(arglist, int);
    }

    va_end(arglist);
}
```

Ready ssh2: AES-256-C

pod3-3.cs.purdue.edu - SecureCRT
File Edit View Options Transfer Script Tools Window Help
Enter host <Alt+R> | [Icons]
Session Manager
Session: pod3-3.cs.purdue.edu
}
int mysum(char *a, ...)
{
 int sum = 0, i;
 int cnt;
 char *s;
 va_list arglist;

 va_start(arglist, a);
 cnt = strlen(a);
 for (i=0; i<cnt; i++) {
 if (i == 1) {
 s = va_arg(arglist, char *);
 sum += atoi(s);
 }
 else
 sum += va_arg(arglist, int);
 }

 va_end(arglist);
}

Ready ssh2: AES-256-C

We know
the third
argument
is a
String

myprint f

pod3-3.cs.purdue.edu - SecureCRT

File Edit View Options Transfer Script Tools Window Help

Enter host <Alt+R>

Session Manager Job

✓ pod3-3.cs.purdue.edu [x]

// example to illustrate different number of arguments
// a string as in printf() and counts the number

```
#include <stdio.h>
#include <stdarg.h>
#include <string.h>
#include <stdlib.h>

main()
{
    int myprintf(char *, ...);
    myprintf("%d %d %d", 2, 2, 5);
}

int myprintf(char *a, ...)
{
    int i, count = 0;
    int s;
    va_list arglist;
    va_start(arglist, a);
    "main6.c" 41L, 593C
```

Ready ssh2: AES-256-CBC

(Note: The code has been annotated with handwritten notes explaining the printf format string "%d %d %d". A red arrow points from the word "count" to the first "%d", and another red arrow points from "the number" to the third "%d".)

pod3-3.cs.purdue.edu - SecureCRT

File Edit View Options Transfer Script Tools Window Help

Enter host <Alt+R>

Session Manager

pod3-3.cs.purdue.edu

```
int myprintf(char *a, ...)  
{  
    int i, count = 0;  
    int s;  
    va_list arglist;  
  
    va_start(arglist, a);  
  
    while(*a != '\0') {  
        if (*a == '%')  
            count++;  
        a++;  
    }  
  
    printf("count = %d\n", count);  
  
    for (i=0; i<count; i++) {  
        s = va_arg(arglist, int);  
        printf("%d\n", s);  
    }  
    va_end(arglist);  
}
```

write()

Ready

ssh2: AES-256-CTR

pod3-3.cs.purdue.edu - SecureCRT

File Edit View Options Transfer Script Tools Window Help

Enter host <Alt+R>

Session Manager

Session pod3-3.cs.purdue.edu

```
// example to illustrate different number of arguments: First argument is
// a string as in printf(), counts the number of occurrences of %, and
// checks for type of input: %d, %s, %d, %c

#include <stdio.h>
#include <stdarg.h>
#include <string.h>
#include <stdlib.h>

main()
{
char r = 'y';
int h = 2;
int myprintf(char *, ...);

myprintf("%c %d %s %d %f %c", r, h, "hello", 5, 7.2, 'y');

}

int myprintf(char *a, ...)
{
int x;
char *y;
main.c" 60L, 962C
1,1
```

File Edit View Options Transfer Script Tools Window Help

Enter host <Alt+R>

Session Manager

Session pod3-3.cs.purdue.edu

```
int myprintf(char *a, ...)
{
int x;
char *y;
double z;
char m;
va_list arglist;

va_start(arglist, a);

while(*a != '\0') {
    if (*a == '%') {
        switch (*(a+1)) {
            case 'd':
                x = va_arg(arglist, int);
                printf("%d\n", x);
                break;
            case 's':
                y = va_arg(arglist, char *);
                printf("%s\n", y);
                break;
            case 'f':
                z = va_arg(arglist, double);
                break;
        }
    }
}
```

43,1-8

Ready

ssh2: AES-256-CTR 24, 8 25 Rows, 92 Cols VT100

Session Manager

```
pod3-3.cs.purdue.edu
```

```
va_start(arglist, a);
while(*a != '\0') {
    if (*a == '%') {
        switch (*(a+1)) {
            case 'd':
                x = va_arg(arglist, int);
                printf("%d\n", x);
                break;
            case 's':
                y = va_arg(arglist, char *);
                printf("%s\n", y);
                break;
            case 'f':
                z = va_arg(arglist, double);
                printf("%f\n", z); cheating with printf
                break;
            case 'c':
                m = va_arg(arglist, int);
                printf("%c\n", m);
                write(1, &m, 1); using write()
                break;
            default:
                continue; true ↗
        }
    }
}
//
```

Ready ssh2: AES-256-CTR 24, 8 25 Rows, 92 Cols VT100 52,1-

Session Manager

```
pod3-3.cs.purdue.edu - SecureCRT
```

```
File Edit View Options Transfer Script Tools Window Help
Enter host <Alt+R> | ☰ 🌐 🔍 ⚙️ 🛡️ 🔍 🔒 ? |
```

```
#include <stdio.h>
```

```
main() {
    int x = 2, y = 3;
    printf("%d %d\n", x, y, x);
}
```

↓ count number of %'s

2
print 2 arguments

warning but still ↗

not error!

"main8.c" 8L, 81C

Ready ssh2: AES-256-CTR 1, 1 25 Rows, 92 Cols VT100

A screenshot of a SecureCRT terminal window titled "pod3-3.cs.purdue.edu - SecureCRT". The window shows a C program named "main8a.c" with the following code:

```
#include <stdio.h>
main() {
    int x = 2, y = 3;
    printf("%d %d\n", x);
}
```

The terminal output shows the following:

```
"main8a.c" 8L, 75C
Ready      ssh2: AES-256-CFR 1, 1 25 Rows, 92 Cols VT100
1,1
```

Handwritten annotations in red are present in the terminal window:

- A magnifying glass icon is placed over the first argument of the printf call.
- The text "Count 2 , [Warning !!!]" is written next to the printf line, with "not error" written above it.
- The text "But only can print / correctly" is written below the printf line.

when call va-arg() the
second time, will print
junk!