C Programming Warm-up

Liang Yan

September 18, 2014

1 Violation identify

(a) Dynamic allocation for an array is in units that does not correspond to the array type.

```
L58     nameList=malloc(40*sizeof(char));
/* Allocate memory for the array of names*/
```

Fix:

```
nameList=malloc(40*sizeof(char*));
/* Allocate memory for the array of names*/
```

(b) It is possible to write beyond the allocation for array nameList (can occur in seven different statements).

```
L58     nameList=malloc(40*sizeof(char));
```

Since there are only 40 entries for nameList, it could be easily be written beyond when there are over 40 files in a directory.

Fix: I use a threshold variable, let the Dynamic array increased automatically, when reaching threshold, make the capacity double.

(c) It is possible to write beyond the allocation for array fullName (can occur in any of three different statements).

```
L71                 fullName=malloc(100*sizeof(char));
          strcpy(fullName,dirName);
          strcat(fullName,"/");
          strcat(fullName,entry.d_name);
```

There could be overflow here since no size check for strcppy and strcat. Fix: set the fullName size equals the sum of dirName slash, and entry.dname;Also use strncpy and strcat instead of strcpy and strcat.

(d) Two variables are declared but never used.

```
L48     int             i;              // Loop counter
L54     struct stat     entryStats2;    // File info given by lstat
```

Fix: Just delete them.

(e) A variable is used before it is initialized.

```
L56     int    ptr;          // Next file name goes at this index in array
```

As no initialization, it could be any value, makes the pointer nameList[ptr] very dangerous.

Fix: add ptr = 0;

(f) A pointer to dynamically allocated memory may be lost before the memory can be freed. (This means that it may not be possible to free the memory. It does not mean that the memory is not freed.)

```
opendir
L65 L90 readdir_r
L71 malloc during the while Loop
L79 lstat
L92 closedir
```

Each function, if they could not return successfully, there is a possibility that it let the typelist return to main without the pointer address of nameList, then, the memory is lost.

Also, even the typelist returns successfully, we should free them on the main function.

Besides that, the typelist forget to free the fullName that not equal to typeNum.

Fix
Add return code handling part, when exit abnormally for typelist, we need to free the malloc part fist.
Also, even everything works well, we will need to free the unused variables.

(g) The value from the typeList subroutine may not always have been set to reflect an error.

```
L64       if (dirPtr==NULL)return;
```

Fix

```
    if (dirPtr==NULL){ // if could not open the directory
      perror("open dir error");
      free(nameList);
      nameList = NULL;
      return nameList;
    }
```

(h) Return codes are not checked after each subroutine call

```
opendir
readdir_r
malloc during the while Loop
lstat
closedir
```

Same as question f.

(i) There is a path through the code in which a pointer variable may not have been successfully initialized before it is used (that is, write to memory before successful allocation).

```
L59       bzero(nameList, sizeof(nameList));
/* All array elementsinitially zero */
L71            fullName=malloc(100*sizeof(char));
```

L59, nameList only presents a size of a pointer which is 8 in 64 bits system, here we use it as an array, so we need to make a full initialization.

```
nameList= malloc(40*sizeof(char* ));          /* Allocate memory for
the array of names */
```

L71, since we use strcpy and strcat here, there is a potential overflow Violation here, we need to initialize fullName, and use strncat and strncpy instead of strcat and strcpy.

```
  bzero(fullName, length*sizeof(*fullName));          /* All array
  elements initially zero */

strncpy(fullName, dirName, length);
  strncat(fullName,"/",1);
  strncat(fullName, entry.d_name, strlen(entry.d_name));
```

2 other problem

1. int main function needs a return value here.

2. default state from the switch part, it looks like no path to here.

3. L59 bzero should after L60 malloc check.