Bomb Analysis

Overview

Sam Hu (khu04) and David Chen (zchen18)

Bomb number: 31

Help and collaboration:

x86 reference sheet for interpreting assembly code

Hours spent: approximately 8 hours

Defuse

The following six lines of code defuses the bomb (does not activate nor defuse the secret phase, which requires Igor_Straminsky after line 4 and a number at the 7th line, respectively):

```
Klinger, how dare you wear that hat while in uniform?
13 13 13 13 13 13
5 -830
4
232323
709
```

Phases Explanation

Phase 1 reads a line and compares it with the string “Klinger, how dare you wear that hat while in uniform?”, explodes if not the same and defuse otherwise.

Phase 2 reads 6 signed integers (“%d %d %d %d %d %d”) from the input line by the function read_six_numbers using sscanf. The bomb explodes if read was not successful. The numbers are then compared with its preceding (not the first one) number, and bomb explodes if they are not equal.

Phase 3 takes two integers by sscanf and explodes if read was unsuccessful. The first number also has to be within the range [0-5] (there were two checks, one checking for <= 7 and other for <= 5). The code then performs a switch table operation, where the first value is treated as an index to select how the target value is generated. The target value is then compared with the second number and bomb explodes if not equal.

Phase 4 takes 1 integer, runs it through func4, which is a factorial function, and compares the result with 24. bomb explodes if not equal.

Code

----------PHASE_5----------

```
/*****************************************************************************
* phase5.c                                                                   *
* David Chen (zchen18)                                                       *
* Sam Hu (khu04)                                                             *
*                                                                            *
* HW5: bomb                                                                  *
*                                                                            *
* Summary:                                                                   *
* Provides functionality of the assembly function  "phase_5"  written in c   *
*                                                                            *
*                                                                            *
* Notes:                                                                     *
*                                                                            *
*****************************************************************************/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

const int ARR[16] = {2,10,6,1,12,16,9,3,4,7,14,5,11,8,15,13};
const int LENGTH = 6, VAL = 21;

Extern void explode_bomb();

/*************** phase_5 ***************
*
* hash 6 characters by 16 and use as index to add number to accumulator
*
* Parameters:
*      char* input: contains the input string
```

```
*
* Return:
*       none
*
* Notes:
*       Bomb explodes if string is not length 6 or value for accumulator does
*       not match
*
************************/
void phase_5(char* input)
{
        if(strlen(input) != 6){
                explode_bomb();
                exit(1);
        }

        int val = 0;
        for (int i = 0; i < LENGTH; i++){
                int idx = (int)input[i] & 0xf; /*hash with last 4 bits*/
                val += ARR[idx];/*add to accumulator*/
        }

        if (val != VAL ) {
                explode_bomb();
        }
}
```

----------PHASE_6----------

```c
/*****************************************************************************
* phase6.c                                                                   *
* David Chen (zchen18)                                                       *
* Sam Hu (khu04)                                                             *
*                                                                            *
* HW5: bomb                                                                  *
*                                                                            *
* Summary:                                                                   *
* Provides functionality of the assembly function "func6" and "phase 6"      *
* written in c                                                               *
*                                                                            *
*                                                                            *
* Notes:                                                                     *
*                                                                            *
*****************************************************************************/

#include <stdio.h>
#include <stdlib.h>
#include <assert.h>

typedef struct List_Node {
	long val;
	struct List_Node *next;
} *List_Node;

const int LEN = 9;

const int list[9] = {800, 597, 338, 976, 709, 72, 677, 424, 694};

void explode_bomb();


/*************** construct_List ***************
*
* Construct a linked list of size LEN and return the head node
*
* Parameters:
*	NONE
*
* Return:
*	Returns the head node of the empty linked list
```

```
*
* Notes:
*       - CRE if memory allocation fails
*
***********************/
List_Node construct_List()
{
        List_Node head, prev;
        for (int i = 0; i < LEN; i++) {
                List_Node cur = malloc(sizeof(*cur));
                assert(cur != NULL);
                cur->val = list[i];

                if (i == 0) {
                        head = cur;
                } else {
                        prev->next = cur;
                }
                prev = cur;
        }
        return head;

}

/*************** fun6 ***************
*
* Sort a linked list in non-increasing order
*
* Parameters:
*       a List_Node object representing the head of the linked list
*
* Return:
*       None, sorting in place
*
* Notes:
*       - will terminate if the linked list is empty or of length 1
*
***********************/
void fun6(List_Node head)
{
        if (head == NULL || head->next == NULL) {
                return;
```

```
        }

        /* Left stores the sorted list of insertion */
        List_Node left = NULL;
        List_Node cur = head;
        List_Node nextNode = NULL;

        while (cur != NULL) {
                nextNode = cur->next;
                cur->next = NULL;
                /* Add the current node to the left of the sorted list */
                if (left == NULL || left->val <= cur->val) {
                        cur->next = left;
                        left = cur;
                /* Add the current node to its position in the sorted list */
                } else {
                        List_Node temp = left;
                        while (temp->next != NULL
                               && temp->next->val > cur->val) {
                                temp = temp->next;
                        }
                        cur->next = temp->next;
                        temp->next = cur;
                }
                cur = nextNode;
        }
        head = left;
}

/*************** phase6 ***************
*
* Check if the input integer (in the form of a char pointer) is
* equal to the third largest value of the linked list; explodes the
* bomb if it is not equal
*
* Parameters:
*       a char pointer input representing the target long integer
*
* Return:
*       None
*
************************/
```

```c
void phase6(char *input)
{
	/* Convert input string to long */
	long input_long = strtol(input, NULL, 10);

	/* Construct the linked list and sort it */
	List_Node list_head = construct_List();
	fun6(list_head);

	/* Get the third biggest value and compare with input */
	list_head = list_head->next;
	list_head = list_head->next;
	if (list_head->val != input_long) {
		explode_bomb();
	}
}
```