**深 圳 大 学 实 验 报 告**

**课程名称： 计算机系统(2)**

**实验项目名称： 逆向工程实验**

**学院： 计算机与软件学院**

**专业： 计算机科学与技术**

**指导教师： 罗秋明**

**报告人： 刘俊楠 学号： 2017303010 班级： 01**

**实验时间： 2020年04月22日**

**实验报告提交时间： 2020年05月13日**

**教务处制**

|  |
| --- |
| **一、 实验目标与要求：**   1. 理解程序（控制语句、函数、返回值、堆栈结构）是如何运行的 2. 掌握GDB调试工具和objdump反汇编工具 |
| **二、实验环境：**   1. 计算机（Intel CPU） 2. Linux64位操作系统（Ubuntu 17） 3. GDB调试工具 4. objdump反汇编工具 |
| **三、实验方法与步骤：**  本实验设计为一个黑客拆解二进制炸弹的游戏。我们仅给黑客（同学）提供一个二进制可执行文件bomb\_64和主函数所在的源程序bomb\_64.c，不提供每个关卡的源代码。程序运行中有6个关卡（6个phase），每个关卡需要用户输入正确的字符串或数字才能通关，否则会引爆炸弹（打印出一条错误信息，并导致评分下降）！  要求同学运用**GDB调试工具和objdump反汇编工具**，通过分析汇编代码**，**找到在每个phase程序段中，引导程序跳转到“explode\_bomb”程序段的地方，并分析其成功跳转的条件，以此为突破口寻找应该在命令行输入何种字符串来通关。  6个关卡，难度随关卡升级而提升；  通过解读汇编代码来推断其对应的函数结构（推断过程不唯一），某些关卡答案不唯一；  尽力而为，能通几关就几关。  提示：   1. 第一关（知识点：string，函数调用，栈） 2. 第二关（知识点：循环语句，数组） 3. 第三关（知识点： switch语句） 4. 第四关（知识点：递归） 5. 第五关（知识点：字串变换，ascii转换，寻址） 6. 第六关（知识点：寻址）   （第1-5关，各15分。第6关10分。实验总结15分。） |
| **四、实验过程及内容：**   1. 将bomb.txt文件的后缀改为exe，并利用objdump反汇编，将结果输出在1.txt中（图4-1-1）。     图4-1-1   1. GDB调试（图4-1-2）     图4-1-2  <Phase 1>  分析1.txt里的phase\_1代码段（图4-2-1）得知，此题为判断字符串相等，故找到对应储存字符串的地址，再用gdb查看即可知道答案。    图4-2-1  【思路】 通过阅读汇编代码可知，%rsp标志爆炸状态的变量，分析得知，后面有一个strings\_not\_equal的函数，故我们推测它进行字符串比较。将%esi储存的字符串与后来输入的字符串比较，%eax应该是检测输入是否为空的作用，随后若输入字符串与$0x401af8地址对应字符串相等,则跳过400e82的炸弹爆炸环节。故程序自身应有一个字符串，经代码检测，关键点在0x401af8。故下面进入gdb调试，然后查看0x401af8该地址对应的数据（图4-2-2）。    图4-2-2  由上图可知，0x401af8地址存储了一个字符串“Science isn’t about why, it’s about why not?”，很可能就是本题答案，所以尝试该答案（图4-2-3）：    图4-2-3  由上图可知，本题答案正是“Science isn’t about why, it’s about why not?”  第一关通过。  <Phase 2>    图4-3-1  （如图4-3-1）从400ea7中的<read\_six\_numbers>可以看出phase\_2需要读入6个数。然后在400eba-400ed4的循环中比较%rbp和%rbp+0xc，%rbp+0x4和%rbp+0x10，%rbp+0x8和%rbp+0x14。即第一个数与第四个数、第二个数与第五个数、第三个数与第六个数分别比较。如果出现不相等则爆炸，如果都相等，到400ed6中判断%r12d是否为0，若为0则爆炸，否则通关。所以1 2 3 1 2 3满足以上条件，进行测试，结果如下（图4-3-2）：    图4-3-2  第二关通过。  <Phase 3>  代码分析如下图所示（图4-4-1）：    图4-4-1  首先看400f07，发现0x401ebe的值被当作参数传入，所以查看其内容（图4-4-2）：    图4-4-2  发现是需要读两个int类型数。由代码分析（图4-4-1）可知第一个数不能超过7，否则直接爆炸。然后来到jmpq跳转指令，根据输入的第一个数跳到一个地址对%eax赋值。通过gdb调试的x指令显示16进制查看跳转的地址（图4-4-3）：    图4-4-3  建立第一个输入和跳转到的地址想要给%eax赋的值之间的映射，如下（图4-4-4）：    图4-4-4  最后判断第二个输入和%eax是否相等，相等则通关。  所以，输入2 214就应该可以通关（图4-4-5）：    图4-4-5  第三关通过！  <Phase 4>  首先看400fca（图4-5-1），发现0x401ec1的值被当作参数传入，所以查看其内容：    图4-5-1  发现是需要读一个int类型的数。故对源代码进行分析（图4-5-2）。    **且输入大于1**  图4-5-2    图4-5-3  由上图分析可知，我们需要输入一个数，将这个数传进func4的话会返回55；而对于递归函数func4，func4(1)=1，func4(x)=func4(x-1)+func4(x-2)(x>1)，即求斐波那契数列的第n项的值。所以问题转化成：斐波那契数列数组中，第几项为55？答案是9（图4-5-4）。    图4-5-4  第四关通过。  <Phase 5>  首先看401010，发现0x401ebe的值被当作参数传入（图4-6-1），所以查看其内容：    图4-6-1  发现需要读两个int类型的数。    图4-6-2  由上图的分析可知需要知道0x401ba0开始预存的数组，用gdb查看：    图4-6-3  从图中可知数组中元素内容为（下标为0开始，15结束）：  {a，2，e，7，8，c，f，b，0，4，1，d，3，9，6}（16进制）  即：  {10，2，14，7，8，12，15，11，0，4，1，13，3，9，6}（10进制）  从15回溯12次就可以得到第一个数的下标（表4-6-1）。   |  |  |  | | --- | --- | --- | | 回溯次数 | 值 | 下标 | | 12 | 15 | 6 | | 11 | 6 | 14 | | 10 | 14 | 2 | | 9 | 2 | 1 | | 8 | 1 | 10 | | 7 | 10 | 0 | | 6 | 0 | 8 | | 5 | 8 | 4 | | 4 | 4 | 9 | | 3 | 9 | 13 | | 2 | 13 | 11 | | 1 | 11 | 7 |   表4-6-1  程序为自底向上，故第一个参数为回溯开始时数组下标：7，又因为求和数组15+6+14+2+1+10+0+8+4+9+13+11=93 ，所以第二个输入：93，结果如下（如图4-6-4）：.    图4-6-4  第五关通过！  <Phase 6>    图4-7-1  现有注释node0（图4-7-1），说明可能含有链表结构，在设置链表处设置断点b \*0x4010ec,设置用x/20xw 0x602780查看初始链表值（图4-7-2）：    图4-7-2  果然是链表，且有十个值，分别指向是：0->374->826->370->782->488->673->286->600->529->NULL.  下面再看看fun6做了什么（图4-7-3）：    图4-7-3  观察每个节点的next指针变化（图4-7-3），容易发现链表的进行了一次由大到小的排序，（826,782,673,600,529,488,374,370,286,6）我们输入的数是一直保存在第一个结点之中，此操作将第一个结点的0改写。    图4-7-4  观察这段代码可以发现%rax指向的是第四个结点（node4），而该结点的值必须等于输入，也就是说我们输入的数必须排在原来的链表中第四的位置（图4-7-4），而从该链表中原有的数字（826,782,673,600,529,488,374,370,286,0）可以看出，输入的数字必须在600-673之间，刚好这样插进第四大的位置被%rax指着,否则%rax的值要么>673（输入太大）要么就是<600（输入太小），不会等于输入值。数字600符合分析（图4-7-5），结果如下：    图4-7-5  第六关通过。  <Secret Phase>   1. 发现代码仍然有fun7，于是猜测有隐藏题目，在全局搜索关键字可以发现在phase\_defused这个函数里调用了phase\_defused，查看该关卡可以发现有跳转到Secret Phase的代码（图4-8-1）。     图4-8-1  设置断点调用gdb查看内存0x 401ec4中传的什么参数（图4-8-2）    图4-8-2  可看出是要输一个整数和一个字符串。查看0x603030中的值，结果如下，值为9（图4-8-3），所以可能是在第四关输入‘9’之后再输入另一个字符串就能触发Secret Phase。查看内存0x401eca中的值如下图所示（图4-8-4），所以尝试在第四关输入正确答案之后再输入该字符串（图4-8-5）    图4-8-3    图4-8-4    图4-8-5  通过对多个点的调试与查看值，一个一个试，最后试出107为正确答案（图4-8-6 图4-8-7 图4-8-8）。    图4-8-6    图4-8-7    图4-8-8  3、测试答案（图4-8-9）    图4-8-9 |
| 1. **实验结论：**   第一关答案是：Science isn't about why, it's about why not?  第二关答案是满足前三位和后三位对应相等且前三位数字和不为0的6个数，如：1 2 3 1 2 3  第三关答案通过gdb建立映射，选取合适的答案，如：2 214  第四关答案是：9 austinpowers(隐藏答案)  第五关答案是：7 93  第六关答案是在[600,673]范围的整数 如：600  Secret Phase答案是107 |
| **六、心得体会：**  通过这次实验，我体会到了反汇编的用法，以及在汇编语言下实现gdb调试的能力，明白了汇编语言是如何进行递归、判断、调用、回溯……的功能。了解了计算机系统是如何在内部执行一条指令，对栈与堆的递归有了更加深刻地理解。深刻意识到c++ ，c，java等高级编程语言的重要性。 |

|  |
| --- |
| 指导教师批阅意见：  成绩评定：  指导教师签字：  2021年 月 日 |
| 备注： |