1. There are two files, namely file1 and file2.

Contents:

File1:

This is version 1.0 of file1.

This should be version 2.0 of file1.

File2:

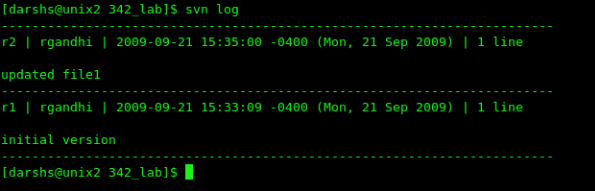
This if version 1.0 of file2

1. ‘rgandhi’ committed it.

Command used: svn log

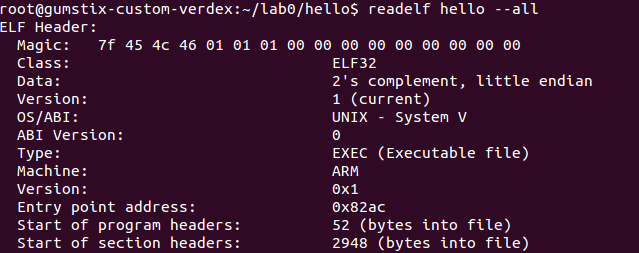
1. There were 2 commits:

The first commit was done at 15:33 on 09/21 and the second was done at 15:35 on 09/21



3.4.3

1. Entry point address is : 0x82ac



2. we went to hellod and starting from 0x82ac, we analyzed the instruction further till we encountered the first branch instruction. We found it branches to 0x8288. Then using “readelf -r hello | grep 8288” we found that the first function it branches to is **\_\_libc\_start\_main**.

3. objdump –d displays the assembler mnemonics for the machine instrctions from objfile. This option only disassembles those sections which are expected to contain instructions. The –D disassembles the contents of **all sections, not just those expected to contain instructions.**

If the target is an ARM architecture, this switch also has the effect of forcing the disassembler to decode places of data found in code sections as if they were instructions.

1. No, the interpretation is not correct since it converts data into instructions. As objdump –D disassembles all sections as explained previously, this is incorrect.

Code explanation:

Src=”theinitialstring\n” Srclen=16 bytes

Dst=”PENGUINS” Dstlen=8 bytes

After saving R6, we first check if destination length is 0 and if so we just exit. If not, we subtract the destination length by 1 since a count-down loop is used. So now the value of R3 will be 7. Now we load the character from the destination string into R12 and keep a copy of it in R6 also. Hence R6 and R12 will be ‘S’ in this case. Then we subtract 23 from R12 until it is less than or equal to 22. Now compare R12 will the length of source string and if greater, just continue to the next character in the destination string. If not, replace the corresponding character in the source string with this character in the key. Therefore, we are replacing n with S in the source string. This loop is continued until the value of R3 becomes negative (i.e.) all the characters in the destination string are traversed through. Finally, we restore R6 before exiting.