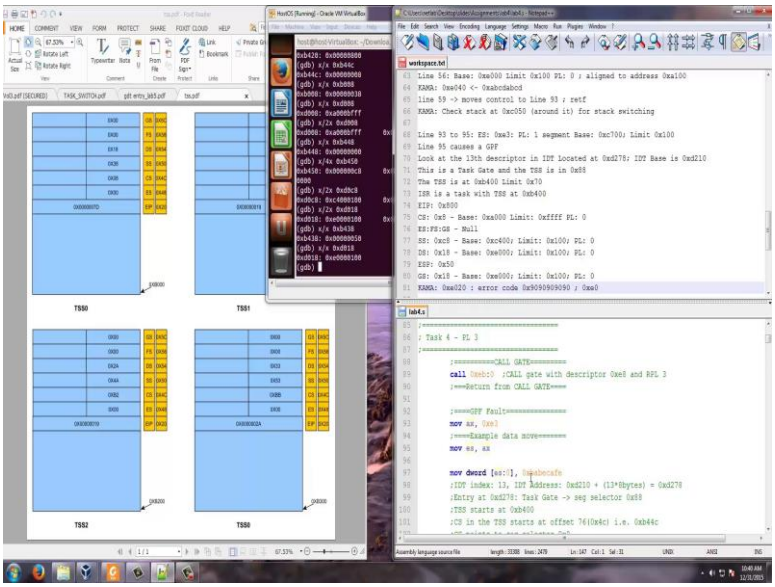
**Information Security - II**
**Prof. V. Kamakoti**
**Department of Computer Science and Engineering**
**Indian Institute of Technology, Madras**

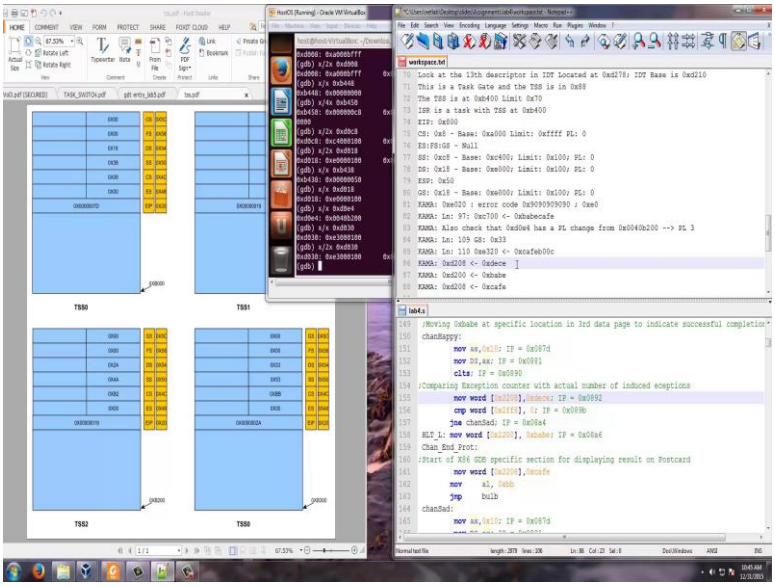
**Lecture - 34**
**Lab4 Part 4 - Week 6**

(Refer Slide Time: 00:09)



And I do a iret. When I do an iret you start executing back at this point. And now your ES colon 0 should work correctly because it should not moves ES comma AX should work correctly because now it as become previous level 3.

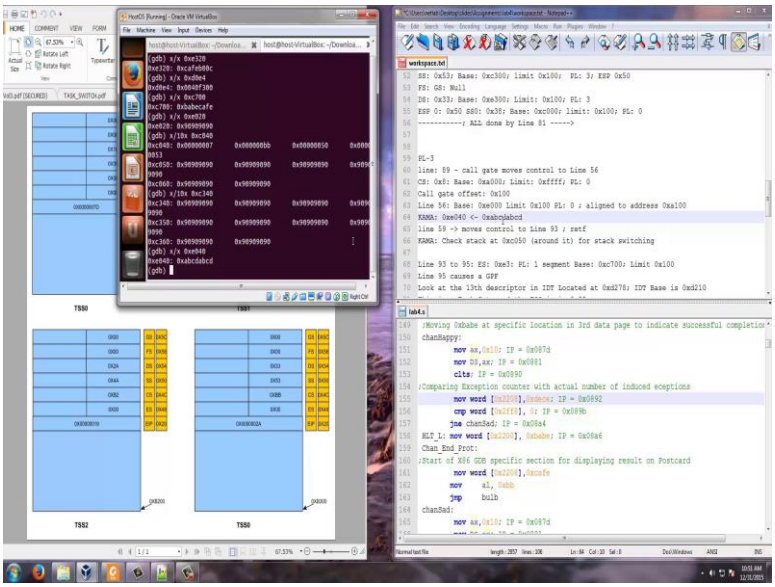
(Refer Slide Time: 00:34)



Now, we got to ES colon 0 in line 97 what I do, in line number 97. I am moving ES colon 0. What ES? Actually, ES as E3 now, E3 is what? So E3 is a PL now its a PL3 segment so is c1700. In a 0xc1700 I need 0x, also check that 0xd0e4 as PL (Refre Time: 01:46) Now, what is d0e4 currently has? d0e4 currently as 40 p200 and that will become 40 f f 0 f200, so this should change so also check will change from 0x040b2002 PL3, currently it is PL1. Now when this happens, let us now go down so we have finished this. Now, we move AX comma 0x33, so let us see what is there is in 0xb0b3d030. This is a e300 segment, it is a previously 3 segment e300 with base 0, so e300 I am moving line number 110 1 naught 9. Your GS is 0x33 comma e300 line number 110, your 0xe300 should actually become 0xcaf e b 000c. And then I do and iret. When I do an iret, this basically goes to the fellow who called me, who called me? This is the fellow who called me. So I go to the ins instruction next to iret. This iret will take me to next to the fellow who called me, so that is this line number 71 so I go to this iret. This iret will take me to the fellow, who call me which is this and from there I jmp to chan. This is the entire code and then we do all the other things from chan . So, from there when you see here we see that your d208 should become 0xdc, your d200 should become 0xd and your d208 should essentially become 0xcac. So, this is annulled because d208 is and then that is it. This is the entire execution of your think. So let us just go to the front intency. This d200 is now changed so you will not see def, so this should go out and this d208 should also

change. Let us see quickly what are all the things that are going to happen and then we will also see what are the things that could happen here.

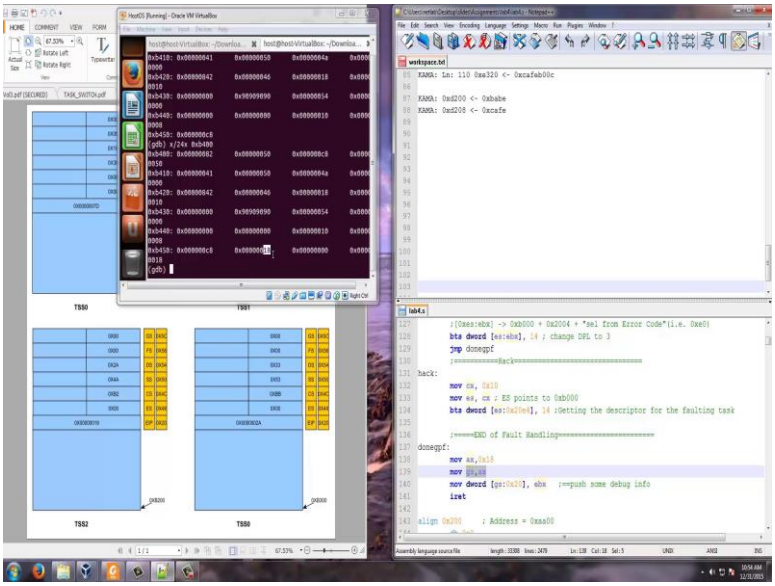
(Refer Slide Time: 06:51)



So, let us go and execute this. These error do come we can just try with reduced board rate instead of 38400 you could have 19200 and some times it will work much faster. This is done now. So, let us go and check from the back. Your d208, yes it has cafe and this as babe and let us see I am just looking at the sentence. Let us see what is happening to your e300. This is be 00c yes this is working and d0e4 your privilege level has to have change. Note that it was40b2 now, it as become 40f3, f is the privilege level 3 and access bit as become 1. This f is privilege 3 I have changed it to privilege 3, then only that particular instruction would have work. Now, let us see what is there in c700 (Refer Time: 08:48) and than what is there in e020. This is a latest version of virtual block, so it as 90 it as only the null code this particular thing it not puss it correctly. Similarly, the stack around the segment should have worked here, so c450 let us see x slash 10x0xc040. As you see here there as been some pushing here 53, 50 b b 7 etcetera. So, these are the stack corresponding to the 0 stack. When the interrupt service routine was the task sets 0, please note on your right hand side, I am showing the 0x68 stack was used though the process which create that interrupt was a PL3 process, but this is actually using a PL0 stack. And then let us go forward and see. Now again when we did

that col gate, the stack around this c050 was actually doing that. So it is using that stack a which is to be used always here, though it was the original stack was in c300. If you see the PL3 stack as you see was c350, so let us say x slash 10xc340 the stack is never touched still null, 90 is null. When I did a call gate please not that I have touch the stack which is around the c050 stack which is the PL0 stack because the call gate, the called function works at PL0 and so all my data that I passed to the call function I will put it in the PL0 stack and not the PL3 stacks. This is very very important from both a security and protection prospective.

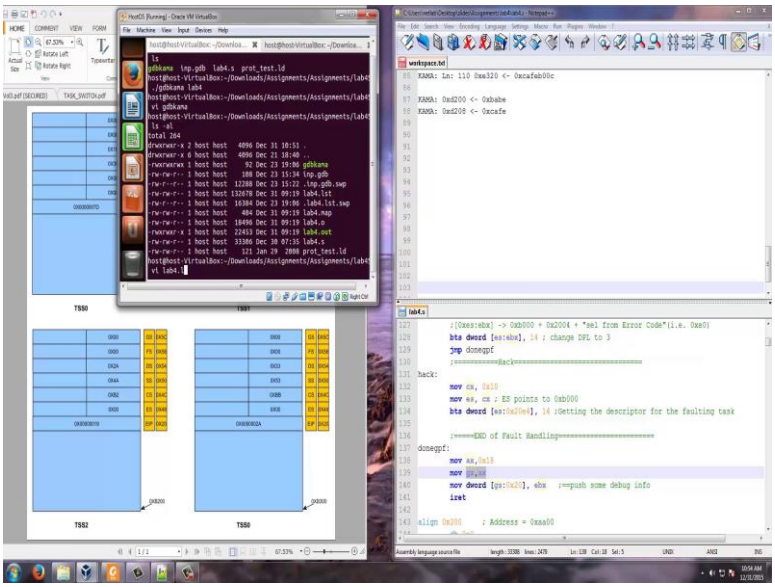
(Refer Slide Time: 11:26)



Now let us go and see also that what is there in line 0x is e040, and there you see abcd, abcd done. E220 that is cafe yes done, e120 that is face p00c done. So, all the code as executed as per what you have seen here. Before we windup let us go and have a look at the different segment descriptors. So let us go and see what is there in x slash, say some 60x x60 x slash 70x0x not 70 so the each x will give me 4 bytes, so 104 by 4 is 21. So 21x slash 21x0x b400 this is the interrupt service routine task gate. Please note that your EIP which was b420 is now 842 because we started executing at 8000 and this is 842, your 48b440 (Refer Time: 13:46) 48 is 10 where ES is 10, your CS is 8 and your SS is c8 and your DS is 18 sorry 8. SS is c8, DS is 18 your FS and GS, GS has become 18. Please

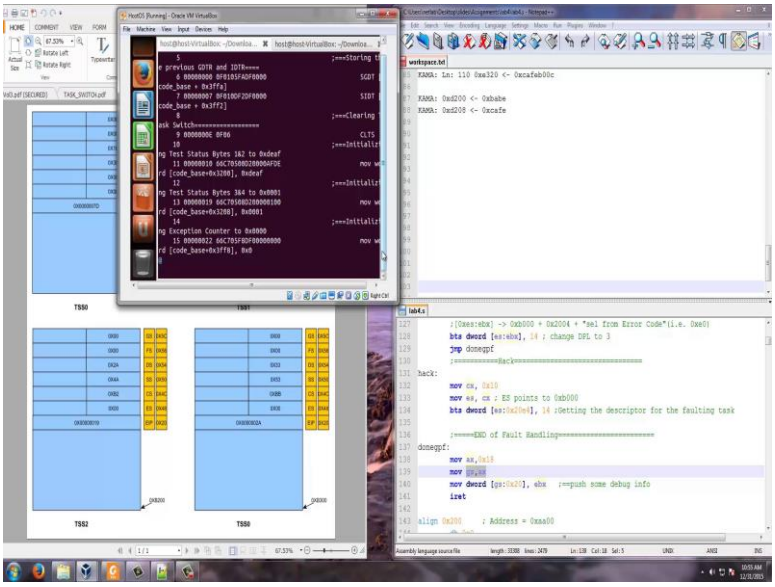
note that you made GS 18 as a part of your interrupt service routine note here in line 139 you made GS 18 that is (Refer Time: 14:37) and 842 is this line.

(Refer Slide Time: 14:42)



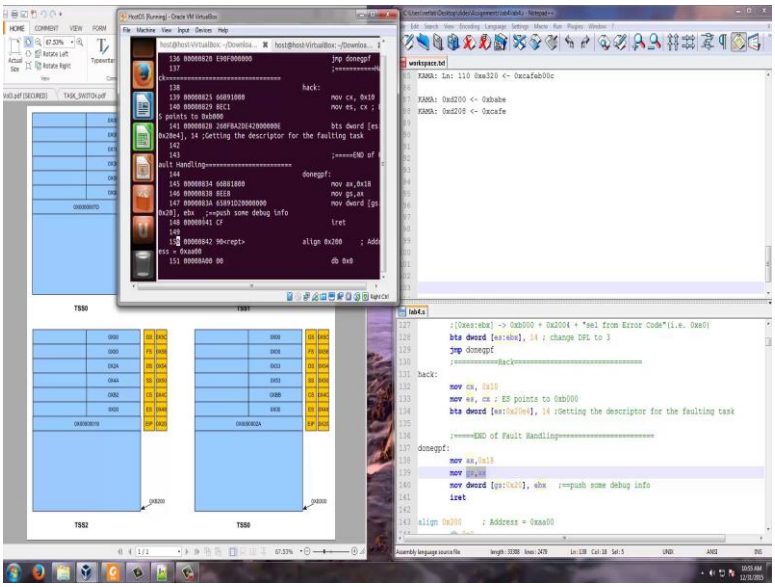
So, one small thing that we can also do when we are doing this we can also say minus L dollar 1 dot LST.

(Refer Slide Time: 14:58)



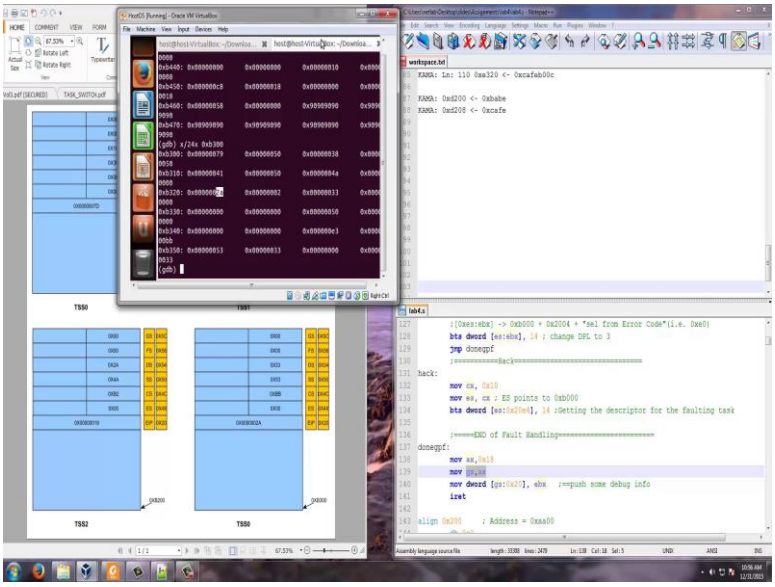
So, when you compile this you also get this LST file, lab 4 dot LST. Let us see what is there in lab 4 dot LST. Now, this will also give you the actual code. So just note this gives you the IP address sorry the EIP address, the Instructing Pointer Address. This is the instruction pointer and this is the actual compiled machine code for SGDT. So these very utility.

(Refer Slide Time: 15:46)



Now, you see there if you go. So the interrupt service routine, this is at 10 GPF. Your iret is at 841 please note here, line number.

(Refer Slide Time: 16:17)



So that is why; when, you went back your written address was 0x842. So written address what you see at b320, b420 is we see here is 842. Similarly, let us see what has happened to the thing at b300. It essentially says that your written address will be b320 to saying 2a. 2a is for your TSS 3. Now, let us see where is TSS 3 ending. Your TSS 3 was ending at here and your starting address was a600 and the offset was 2a. So 29 it ended, so 2a is what is log their. This is basically to tell you that how that TSS gets updated, similarly for your b200 can also, see some specific values being put here and similarly you can do it for your b1100. So, this is how it is their. And this is b000 for your current stack. So, this how task switching has happened and we are gone from privilege level 1, 2, 3, 0 to 1 to 2 to 3 and then there we have done here a task call gate and came back, we dot got a GPF and we handle the GPF and we came back and than we came back from 3 to 2 to 1 to 0. And there we also demonstrator the most important aspect of task switching and also all the privilege check that are happening. When I move from one to another. If I have an authentication to move from PL3 to PL0 through a call gate. I am able to a successfully do it. What if I do not have authentication? I cannot go and check a lower privilege object namely in this case data segment where I caught by a general protection fault. So, all these things put together should serve as an very, very important input for you to understand the architectural aid that is provided for security purposes. If the

operating system which ever the operating system we utilizes the these features in a proper way then there is a chance that we will could build highly.