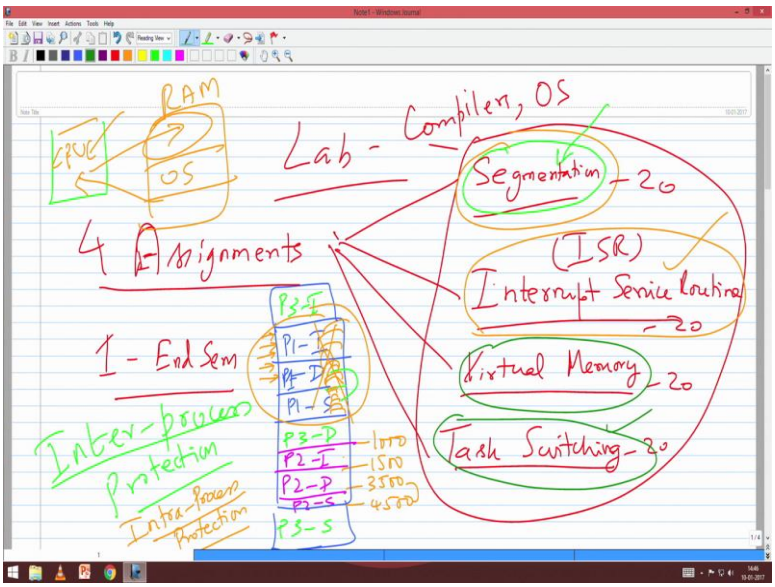
Computer Organization and Architecture
Prof. V. Kamakoti
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture – 03
Lab 1: Introduction

(Refer Slide Time: 00:16)



Actually, we will do slightly different. The second would be interrupt service routines. Next would be virtual memory. And the last will be on what we call as task switching. These 4 are very important concepts which you need to understand to finally, appreciate why certain things are done by the operating system and the compilers right. So, if you want have a good understanding of that then understanding these 4 concepts in detail is a must. So, each will carry 20 marks, we will try to see again, like in the previous semester course, I want you to understand that it first. So, just getting a working code we will not give you marks, but will have a very strict viva voce.

So, we will we will ask you lot of conceptual questions there and we want you to answer those questions. So, this 20 marks would be directly proportional to how well you handle the viva voce. So, understand and do the code I do not even mind you know I will not use the word copying I will be diplomatic discussing with yourself and doing the code. So, I do not really mind that. When it comes to viva voce it is going to be one on one and I want you to answer those questions very clearly, that we will check your thoroughness

there. So, this is not more of skill, but it is more of an understanding that we want to develop through this lab.

So, when we teach you these things like we will cover some background material for each one of these 4, and that will happen in the first 45 minutes of each lab course. So, everywhere there will be I will talk influence of compilers on segmentation and am influence of operating system on segmentation right, similarly influence of compilers on ISRs and OS on ISRs - ISR essentially stands for interrupt service routine. The way you can show yourself as psi funda the easiest way of showing yourself as psi funda is to have lot of these abbreviations memorized. So, ISRs, you can talk an entire talk you can give just with an abbreviations also virtual memory and then task switching.

So, what I will do today is I will now talk for another half an hour and I will tell you what each assignment what it means to complete this assignment what all the objectives of this assignment why this 4 things, that I have put here. When a program wants to get executed the program essentially will have 3 blocks. So, your a dot out if you go and see how it is getting executed, it needs 3 different parts of memory right. So, let us just start when we when we want to execute a program. So, each instruction will undergo 5 different each execution of an instruction will be in 5 phases correct, what are those phases I will fetch the instruction and increment the program counter next.

Student: Decode.

Decode the instruction, fetch the data, execute the instruction and store back the results. So, at least 3 of this right fetching the instruction, fetching the data, storing back the result at least 3 stages I may be touching the memory right. So, a program essentially has 3 components, one is the instruction that we are trying to execute. The second is the data that I am going to use. And the third is; obviously, the stack why do we need the stack again we will explain in more detail we have we have done a quiet a bit of stack work of in our third semester course right, lot of stack you have seen both in your data structure plus in the foundation to computer system design. We will see more of the stack here. So, there are 3 different segments that are necessary when you want to execute the program right. And 2 important things come up. One is that when I look at a mail server for example, there are 2 processes that are executing together right.

So, one process will execute half way in it is execution it may be pulled out another process will go and start executing. Then only it can give a feeling that it is serving all of you right if 4 of you log into a Gmail server, Google server right then you start typing the email till you finish your email the other fellow has to wait right then you get really frustrated there will be one million users. So, the entire CPU is scheduled. So, that you get some time then you are pulled out then he gets some times. So, everybody has a feeling that the server is giving some service to it right.

So, when more than one process one what is the process program in execution right. So, I have hello world dot c that is a c program I compile it I get a dot out that is an executable program. So, I press dot slash a dot out and press enter then that executable program starts executing then it becomes a process. So, a process is nothing, but a program in execution. So, now, there will be several programs in execution right. So, each program each process will have it is own code, it is own data and it is own stack right. So, when I look at the memory there will be p 1s instruction, p 1s, data p 1s stack then there will be p 2 instruction p 2 data p 2 stack.

Now, I will have p 3 instruction, p 3 data p 3 stack when p 3 came and did not have contiguous memory location. So, right it need not be contiguous.

So, there are 3 segments that is why we call it as a segmentation. There are 3 different segments that are associated with each program right each process. Now I need to protect these segments correct. For example, p 1 should not go and look into p 2s instruction data or stack or p 3s instruction data or stack are you getting this p 1 then what will happen it may go and you know can change something in p 2s data or p 2s instruction and then the correctness of execution will not be there when I when I compile a program and I know it is correct I expected to execute correctly no external factors could come and change the execution of my program. That is what I guarantee to the user right.

So, since you have coded a program, and you are syntactically and semantically correct, syntax means the grammar semantics means is the meaning, the logic you are syntactically and semantically correct, and you start executing in my machine the operating system should ensure that your program will execute in the way you want. If some other program comes and right you know changes your execution or changes your data, then you will not get the desired result. So, one of the responsibility of the

operating system is to see that p 1 should go and access p 2s instruction or p 2s data or stack p 1 should not have access to the other thing right. So, this is what we call as inter process protection. I should I call this as inter process protection. One process data stack and code is protected or isolated from the other process instruction data and stack.

Now how do we achieve this? We achieve this through what we call as segmentation, one of the ways of achieving this through segmentation. The next important requirement from the operating system end is that myself. So, what is a stack right. So, what are the 2 operations on the stack, one at a time, push and pop. So, so let us say this is stack and this is data right for p 1, now I am keeping pushing into the stack. Push, push, push, push, push, push, push. Then what happens? I go and push into the data of that fellow right into myself I am pushing, and I go and overwrite my data push, push, push, push, push, push, push, everything is gone. Are you able to follow what I am saying right?

So, I start pushing into my stack. I have access to my data my code and my stack correct. So, I start pushing and I keep overwriting into my data I start overwriting into my instruction right. Then what happen my entire program is execution is completely gone right. Similarly, I start executing program count, what is the program counter program counter points to the next instruction that we need to execute. Now what will happen I keep on executing one after another after another after and I start executing something from the data I start considering my data as a instruction. So, I keep executing and I pass the boundary and starts some mistake happened I start executing from data.

Now, these things your operating system should contain if it has given me this much amount of stack if I overshoot the stack it should say stack over flow. If I start executing outside my segment, then it should say segmentation fault seg fault core dump all these things you have seen right yes or not yes. So, core dump all these things should come now who is giving this operating system should give.

Now, let us go into one fundamental question. I think one of you asked I do not know who asked this question last time. So, core dump that comment is coming right, who will give that core dump comment who is printing that comment core dump segmentation fault who is printing it.

Student: Compiler.

Compiler.

Student: Operating system.

Some program in execution only print it. So, operating system is printing it, but when it is printing it when your program executes and you do some error the operating system comes and prints for you. So, suppose I have only one CPU, right this was happening even when I did programming right where there was only one CPU in that CPU right in that CPU my program is running the program that is going to do a seg fault right that is going to do that do that mistaken that program is running and it did that mistake right how will the operating system come and print it is also a software it is also it should also a execute to go and print something right I should also execute to print something.

So, operating system is a software which is on your memory ram my program is currently executing here I did a mistake, but now you go and say that mistake is the mistake that you have done that mistake what that mistake is being printed by the operating system, how you are understanding this operating system is also a software it has to execute to go and print an error message correct now I am executing there is only one CPU operating system. Obviously, cannot execute there I am executing, I did a mistake and then there is a statement segmentation violation or core dump whatever, but you say that the operating system comes and prints there. How can it do are you getting my question yes or no yes now you tell me how.

Student: Program provides some of callbacks.

Now, how will I know that I have done a mistake, I mean instruction that much I do by divided by 0. So, I will know that I have done a mistake? You are not writing right you are not writing callback routines and all inside you just write a program which will do whatever you wanted to do and there is some mistakes has happened right. The point is I need the hardware support for it. So, when I do a mistake, the hardware will catch and then transfer control a, hey you have done a mistake stop. It will throw you out of the CPU bringing the operating system, and tell operating system hey this guy has done some mistake go and attend to it. Then the operating system will find out what mistake you have done and it will go and print it. So, there is a switch from your program to operating system, you are understanding this correct. So, when you do a mistake when you do a divided by 0.

So, a division instruction comes to the arithmetic logic unit ALU, with the denominator as 0 then immediately it will it will stop your execution you are not fit to be executed you do not know how to even divide. So, it will throw you out it will go and call the operating system and say this fellow is trying to do something, why do not you attend to it. So, the operating system will be loaded. So, your program will go out to the ram and the operating system will get loaded and then operating system will start executing, and then it will find out what error it is and then print it right. So, for me to, so the thing that now we are talking of here if you call that the earlier one has inter process protection, what is this intra process protection right intra process protection. I tried to write I tried to overgrow my stack, I have been given some space, but I try to overgrow that. I overgrow I overwrite it into my data I overwrite it into my instruction memory correct.

Now, somebody has to stop me. So, how will the architecture stop you and how will the control go to the operating system. So, what is the, this is a nice you know this, this entire you know handling of this is done jointly by the hardware and the operating system. Now what you need to understand is what is done by the hardware and what is done by the operating system. You need to have a very clear idea correct. Then you will appreciate how these things are going happen. So before that some way I should know that I am overshooting my stack or I am overshooting my execution right. My instruction is only 2000 bytes I am trying to execute more than that 2000 bytes or my stack is only 500 bytes I am going more than 500 bytes right. My data it will start from here and end here, but I am now shooting overshooting it, somewhere I should deduct that I am overshooting it right and those things that deduction is basically done using segmentation right.

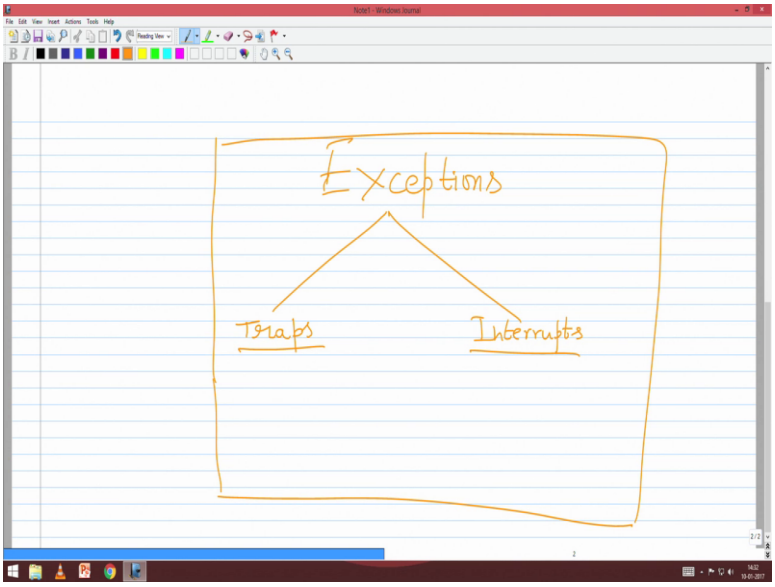
So, segment to sum up segmentation does 2 things one is inter process protection you the process one you are process 2, you cannot see his code and stack and you cannot see his code data and stack you cannot execute or look into it. Perfect isolation like putting one big barrier wall interval that is number one that is inter process protection. The next one is intra process protection where, you know I as accept I have access to my instruction data stack, but I cannot overgrow my stack I cannot keep executing beyond my instructions segment I cannot you know access data beyond.

So, for example, let us take this for p 2 this is 1000 sorry this is thousand this is 1500 this is 3500 this is 4500 for example, So, I cannot store more than 1000 bytes in my stack. I

cannot access data which is before 1000 or after 3500 all my data should be there and all the instructions I execute as p 2 right. It should be in the range between 1000 and 1499 right if I do something if I start executing something before 1000 or after 1499 hardware should catch. How will it catch segmentation will give a methodology for that? So, this is what we will learn in the segmentation.

The next thing that we will learn is the interrupt service routine will do interrupts ISR next right nothing. So, we will do interrupt service routine.

(Refer Slide Time: 18:57)



So, interrupts. So, this interrupt has come in the literature, but I will try to change that. I will call it as exceptions. Exceptions are of 2 types, traps and interrupts. So, there are n different books and n different books call it n different ways, but let us stick to this terminology. And this is much more neater than what I have seen on outside. So, exception what you mean by exceptional scenario, today is an exceptional day means what. Something that is not normal has happened right something abnormal has happened.

So when you start executing a program normally it should execute, but something different than what we expect happen then it is an exceptional scenario. This exception can have happened due to 2 things one is called trap another is called interrupt. Trap I am trapped I am interrupted. So, what is a difference between I am trapped and I am interrupted. Somebody is trapped somebody when do you when you are getting trapped.

So, you do something and you fall into some problem that is called trapping right, like divided by 0 is my problem it is not the operating systems problem, I start executing and very I put a 0 in the denominator.

So, it is not operating systems problem my problem corrects. So, I am this is a trap I start overgrowing my stack right. So, you had given me only say 1000 locations I put 1001 byte into that. So, that is again a trap because it is my thing it is my mistake as a process right. I go and start seeing his code, I want to copy his code I go and see start seeing his code and it is again my mistake correct. So, I am not supposed to do some things and start doing it then it is a trap and the hardware my architecture should find out that I am trying to do something which I am not supposed to do catch me and go and take me to the operating system and say this fellow is doing some fraud you are getting this right. So, that is what we call as a trap. I am doing all correctly, some keyboard is trying to give an interrupt, some disk is trying to give an interrupt there is an urgent thing somebody is pressing control c somebody has gone sat on the top of the keyboard.

So, this is not my mistake. So, the interrupt is coming from some external sources some timer is giving interrupt get out. So, this is all I am this is my problem this is coming from external and that is called an interrupt. The moment there is a trap or there is an interrupt immediately what will happen the hardware, will shut down this process and bring the operating system into existence. The operating system will go and service your trap or service your interrupt, are you getting this right. What you mean my servicing your trap or servicing interrupt if you are doing divided by 0 it will say divide division by 0 error check throw you out right. If it is a stack overflow it is a stack overflow if it has segmentation seg fault core dump it can give you and go off so, but there are cases which we will see where when there is a there is a trap, then it will say no, no wait it will go and do something and allow you to continue right. There are there are scenarios we will see that.

So, all exceptional scenarios that you land up because of your mistake as a process you call it as a trap. All exceptional scenarios due to external sources that you call it as interrupt. I am I clear now whenever there is trap whenever there is an interrupt you need to service the operating system has to come in and service. Who will deduct the interrupt or the trap, the hardware will deduct the interrupt or the trap and it is all the hardware is also responsible for transferring control to a part of the operating system, that part will

start executing and it will do the service. This whole thing is what you will learn in your second assignment namely interrupt service routine clear.

Now, the third is virtual memory when you write a program do you really care about how much memory is there. When you write a program you do not care about lot of things. You do not care about how much memory is there, what is the compiler what everything right one of the important thing is you do not care about what the memory is right. Suppose you are given a 32-bit architecture what is the maximum addressable memory 4 GB 2 power 32 bytes. So, that it will be byte addressable architecture. What you mean by byte addressable architecture, every byte can be individually addressed. So, if I have a 32-bit byte addressable architecture; that means, I could un store up to 32 bytes. So, 4 GB ram I can have. Within that 4 GB some will be occupied by some process some will be occupied by some other process. So many things will be occupied by. So, many processes correct and your process will occupy only some part of the memory, but you can still write a program which is as large as 4 GB, in terms of it can handle huge data it can handle huge stack and some code.

I can have a very big program and still get it executed on a machine which has physical memory of even 2 GB correct. I could still have a 4 GB size program execute on a ram of size 2 GB. So, as for as the programmer is concerned, the operating system says hey there is 4 GB available to you. So, the programmer assumes that there is 4 GB and he writes a program. The compiler assumes that it is 4 GB and it compiles the program correct.

Now, the operating system take it has only say 2 GB ram in which it can give probably say some 512 KB for or 512 MB for this program, within that 512 MB it will somehow execute that 4 GB program and give back. So, as for as how does how does it happen that is what we are going to learn in virtual memory right. So, for as the programmer is concerned he is not bothered about how much physical ram is there he is given the operating system presents a view of 4 GB. Both for the programmer and for the compiler assumes that has 4 GB compiler assumes that their programmer assumes that there is 4 GB. And he writes a program the programs get executed right the operating system will take care of executing a 4 GB ram 4 GB program on say a 5 12 MB ram.

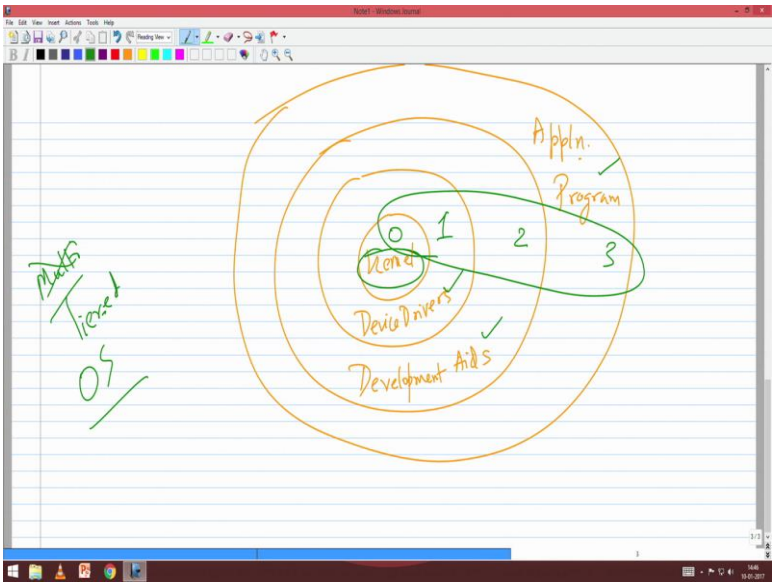
How does it happen this cannot happen again with just the operating system support right? There is a hardware support that is needed and then operating system support that is needed. So, we will understand what the hardware does and what the operating system does, with respect to virtual memory, and the last thing is about what we call as task switching. So, what is task switching? So, if you look at Linux right there is something called supervisor mode road route and there is some normal user route is sort of dada, today all of you do every action with your mobile phone right, those days when all programs have to be written and you forget your password in you know in your d c f right. Route is god night you should call him, hey come reset yaar, I forgot my password he will abuse you, all this you should take what beautiful words yaar common.

Now, you do not need all you have personal laptop and first your fathers are rich to give you laptop, second thing is means that you have use your fathers money right. So, use it properly. Second thing that you need not to now the notion of a d c f is gone right. At last semester assignment you did in your laptop correct. So, lab in the lab is in the lap right so. So, now, this semester also you can do it here you need not actually go to a lab to do a computer science lab fine. So, but those days' route is and even today when you look at major infrastructures today like you know your mail servers right your CSE mail server CSE is a small thing, but if you look at bank mail server right or a bank you know database.

So, the admin is a very powerful guy right. So, there is certain things that you can do as an administrator. There has certain things you can do as a normal user. So, the entire system should have 2 parts. One is the supervisor mode or 2 modes, I should say one is a supervisor mode or another is user mode. So, all the programs normal user program run in the user mode while all some very important things like accessing devices etcetera will happen in the supervisor mode. So, as a program, right as a process I can belong to the operating system, I can also belong to a normal user. So, the process can be classified as a user process as a OS process. The OS process runs in a different mode; the user process runs in some different mode right.

So, each process has the privilege. If I am a OS process, I am little more dada than the normal user process correct. So, every process has something called a privilege level right.

(Refer Slide Time: 30:07)



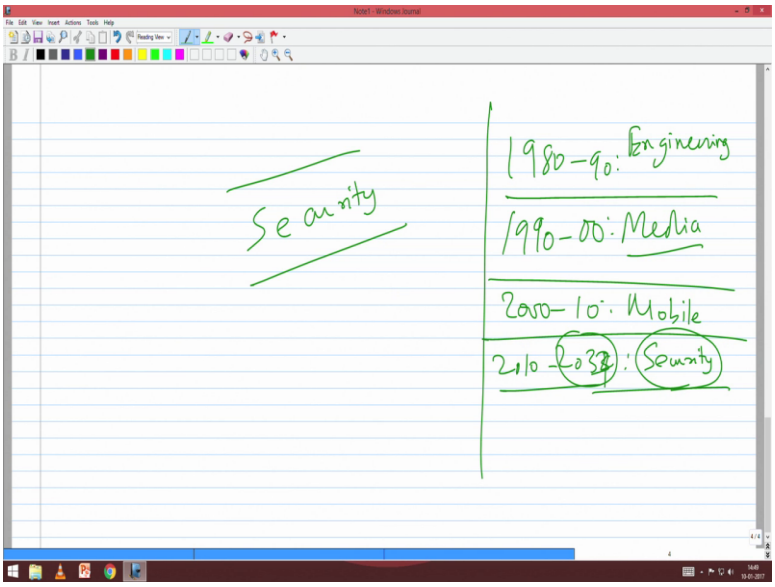
So, in a typical 4 tier operating system, there is something called kernel that is inside after that there would be some system programs like device drivers etcetera. Then there will be some middleware like your run time environments the virtual machine you saw last time right. So, those type of things or your compilers etcetera. So, these are some development layer, development aids right or we can call it as middleware and above this will sit your application program right. So, there is privilege level privilege level 0, privilege level one, privilege level 2 privilege level 3.

The most powerful is the kernel followed by device drivers followed by development aids followed by application programs. So, this is what you call as a tiered operating system. Not tiered tired or multi tired operating system. Tier means it multi-tiered operating fine. So, the architecture what is the support of the architecture for you know maintaining this privilege levels. Every process should be associated with one of these privilege level. I am a process means am I a process of privilege level 3 or 2 or 1 or 0, somewhere I have to maintain that information. And when I want to switch from one privilege level to another privilege level I should do it with some amount of care. I got arbitrarily switched then what happens I become an if I could arbitrarily switch from privilege level 3 to 2 privilege level 0 then I become the super user and change all your passwords correct.

So, there should be some amount of protection that is necessary. So, all these things basically, so how to maintain these privilege levels, and how to switch from one privilege level to another privilege level these are the things that we will learn in the fourth assignment which is task switching.

Now, as you see here we have been more or less I have been talking about one point very clearly right.

(Refer Slide Time: 32:46)



All these 4 assignments finally, has one flavor which is what we call as the security right. So, I thing I have told in the previous course, if you look at 1980 to 90 what was the thrust in computing engineering computations. How do I solve Runge Kutta method, first how will I compute cos hyperbolic fast, how will I do atmospheric weather calculations fast, these are all the big things? So, they are talk about how many million instructions I can do for second 1990 to whatever 2000 the whole story was media right. Jurassic park how will I shoot movies; how will I do all multimedia all these things.

So, it for us consume how to attract the consumer towards computing. A non-computing consumer to computing, right. So, to buy a computer basically media players all these things consumer electronics came up in a big way where computer got this. 2000 to 2010 this is all mobile correct now 2010 to 2033 here I am going to retire.

This is going to be security correct. Actually 34, because now you have start using your withdrawal the cash loss and all these things coming in place, I think that is way the economy is going to grow that is way you know world going to move. So, it is going towards. So, you are going do lot of ecommerce, your health is going to get maintained by your mobile phone. So, already very close to your heart right. More closer than your girlfriend or boyfriend right or wife no problem.

So, now, when you have started using these things in the great way then, that is lot things that we need to do. So, that you know your confidential data that things are not compromised is very important and so, security is something like you know it is like a habit, which should suppose you have been sleeping in every c o class on the last class I say keep awake know it will be very difficult. So, in the initial days itself you should get used to habit not at the end.

Now, unfortunately last 30 40 years' computers have been in existence, nobody gave a damp for security. Suddenly one fine you say I want to make this secure, I have one 10 million lines of operating system and middleware code and executing on one million transistors, now what why would they talk about security right. I want to retrofit security into this security is a habit and at the age of 50 you try to for a computer you are trying to say now you inculcate this habit in a very big way it is not going to happen so easily right. So, first thing is first we should start talking about security in the courses right. That itself we have not talked right. Today can you buy a CPU with a single core no. Right CPU the single core the 8085 something is now 4000 dollars because it is ethnic value. So, the last 50 left out and they will sell it.

Now, we have multiple course, now how many if you just take sigma of across the world university curriculum, including all our own courses how many courses are teaching parallel algorithms. You learn data structure you learn compiler even what I taught you did I tell about parallel how to do this in parallel no we are only still talking about sequential program correct. This multi course started somewhere in 2004, 12 years' curriculums are not matured to talk about concurrent programming at all. Whatever be that reason there are many reasons we can talk about that philosophy later.

But now something which is decayed old we have not still adopted now we are going to talk about and that is much simpler to handle as a concept, as a technology as a process

then what we are talking for security correct. Because there everything is at least open for us to understand in security we do not know what is inside, we do not know what is thus anyone know one million lines of Unix code, even the original originator of Unix may not know, is not that what a code does what that what it does when it is interacting with other parts of codes. So, the security issue is not about just one piece of code, but multiple components trying to do.

Suppose there is a breach in say you know in credit card swiping, that breach could happen because of 20 different vendors today, understand. 20 different people are involved to from the moment you swipe a card to the point it goes to your bank account and it says know you app you get a approval right. 20 different components are there. So, if I talk about a breach somebody says know for a breach you should be responsible somebody says it is, not we can not assign responsibility today there are 2 different fellows who are involved and the fault can be on multiple people, the fault can be of no one right I have done correctly he as also done correctly, but the way we both interact has some fault.

Now whom do we give the responsibility of this interaction right correct. Right, some major companies say that they will keep the server in the sea, in some international waters then they have not bound by the tax rules of the any of the nation's right. So, if I keep it in international water then I it is not India it is not China it is not Pakistan anybody. So, I will not pay tax to anyone you got my point. So, if keep my server and do business in international waters, then no country can claim tax know to which country will I pay and so, these type of this is philosophically the question that we are trying to answer.

So, today security is becoming a very big issue. And for us to learn security we have to learn from the grass root level that is that is my call here. So, that is why one of the lectures that I had sent the YouTube videos and other things, which basically talks about all the 4 assignments that I have put here. Those are the beginning points of you know making your system, secure and to 5 to 10 years later the biggest jobs will not be in your Uber or your Morgan Stanley or Goldman sacks, it will be in security even if you go there it will be in security. So, let us try and give you a vision about that and that is what we will be covering here.

So, my whole thing will have a thrust on security because that is what I feel is the next gen hard topic, when you all go out into the market. You do cloud computing you do data analytics you do everywhere there will be a security there will be a privacy issue there is some NDA that you need to sign right. So, you should understand those languages. I think I have to train you and that is what we will be doing in this course.