Ethical Hacking
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

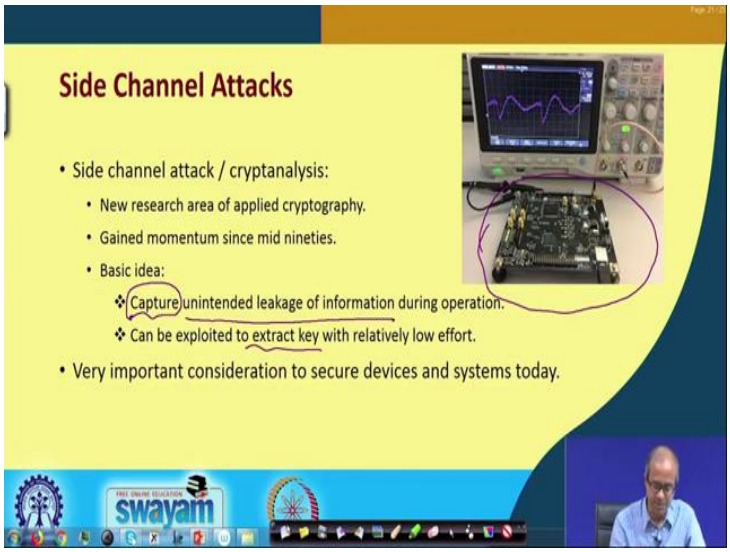
Lecture - 47
Side Channel Attacks (Part I)

So, we continue with the discussion on hardware based attacks. In this lecture we shall be starting our discussion on Side Channel Attacks which I told you is a new kind of an attack which is very interesting in some respect, and it is quite informative to go into a little detail about how this actually works.

(Refer Slide Time: 00:46)

So, in this lecture which is titled Side Channel Attacks Part I, here we shall be talking about firstly, the general idea what is side channel attack and then we shall be looking into a little bit detail about one kind of side channel attack that is called timing analysis attack, ok.

(Refer Slide Time: 01:03)



So, let us see, what is a side channel attack? You see in conventional cryptography, we talk about something called cryptanalysis as the art of trying to break a code; someone has encrypted something; you are trying to break it; you are trying to decode it; but these are done entirely using software mechanism. See over the communication channel some communication is going on; someone has captured it and you are trying to decode it; but side channel attack is different.

Here what is saying that well let us say I have a device like this; I know that something is going on inside this device; there is some encryption or decryption that is going on inside this device and somehow I have captured this device. Now, I take my device to my lab; this device to my lab and in the lab I carry out some kind of experimentation on this device, so that I am trying to find out what is going on inside.

So, this is also a kind of cryptanalysis, but the difference is that the device where encryption decryption is going on that is in my hand; I have access to the device ok; this is the main difference. So, this is a relatively new area of research; because traditional crypt analysis is here for many many decades and in fact, this was proposed in the mid nineties since then the importance was appreciated by the researchers and this has gained momentum. The basic idea is like this; that you see this picture; in this diagram here we show some kind of a circuit board; some kind of a circuit board is here which is doing

something and you see we here we are showing some kind of an oscilloscope, some kind of an instrument which using a probe is connected to this board.

So, while this board is carrying out some operation I am observing what is going on; I am observing some kind of wave forms in the axis of time. So, the basic idea is something like this; we are trying to capture unintended leakage of information; this is the keyword unintended leakage of information during operation, ok. And this information that you are capturing can be used to extract even the secret key that you are using for encryption/decryption with relatively very less effort, ok.

And understanding side channel attack is therefore very important; because unless you are aware of this, you may be building a piece of hardware which can be very easily attacked in this way and the secret key can be retrieved relatively easily. Therefore, in future generation devices this kind of consideration has to be incorporated in a very big way and all security devices and systems must have this kind of consideration in place, fine.

(Refer Slide Time: 04:34)

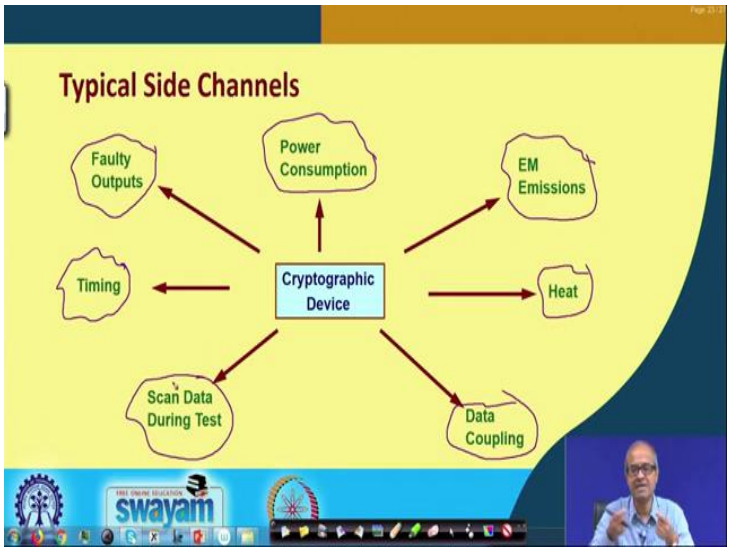
So, side channel attack is important; because the main point is the developer of a secure product; suppose, I am developing some product related to security, ok. So, mean if I say that well my work is only to look at the software; I have to ensure that a software is secure I do not care about the hardware; that is not the right attitude, I have to ensure the security of my whole device; it includes both software as well as hardware. So, the

developer of the product has to defend the product against all possible attack paths; it can be software based, it can also be hardware based.

Then side channel attack, the point to notice that we are not saying that the cryptographic algorithms that are running inside the devices are bad or weak we are not saying that. We are saying that they are mathematically very sound, very stable we all know that; but when you are doing a hardware implementation of this algorithm, somewhere you have become a little careless and because of that carelessness some information is getting leaked out while the algorithm is getting executed in hardware that is the main idea.

So, it is the implementation of the algorithms that is not proper and some information is getting leaked out during the execution of the algorithm in hardware, this is the basic concept, ok.

(Refer Slide Time: 06:17)



So, the typical side channels people talk about, see the side channel as I said is nothing, but some unintended leakage of information; but what do you mean by information? In this picture we have showed some of the most common side channels or information that have been exploited by researchers; like you see timing; how much time it is required to execute an algorithm; this can be one information.

How much power is being consumed during computation of something; you observe the variation of power with time that can give you some information; electromagnetic

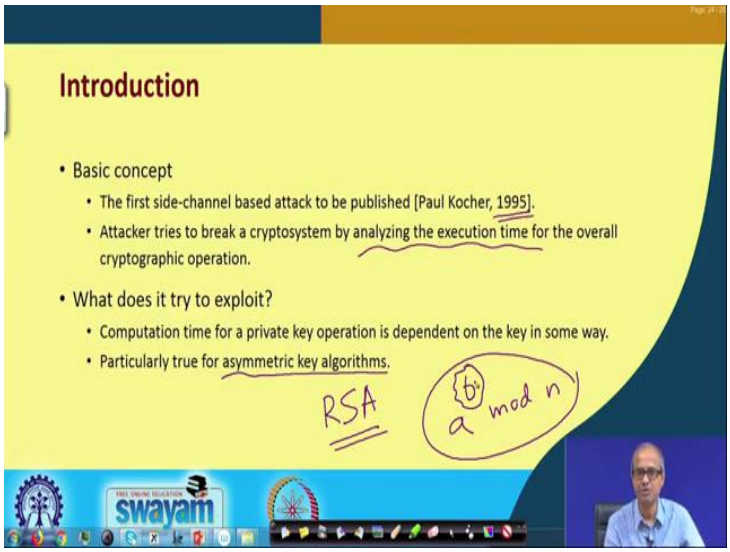
emissions, if you have an electromagnetic sensor sitting on top of your device, so, when something is going on inside, you can directly sense the electromagnetic radiations and observe the variations that are taking place, ok. So, this power consumption and electromagnetic emissions variations will be very similar in nature. Heat, if you have some kind of a heat sensor, you can see how the variation of heat is taking place with time; when the chip is becoming hotter, when just becoming cooler.

Similarly, faulty outputs, some of the outputs, you can inject some fault; these are called fault analysis attacks. You can inject some faults deliberately by changing supply voltages making some disturbance in some of the channels that way you can inject some errors and because of that some important information might come out which may be helpful for you in breaking the system. Data coupling, well, if there are some coupled data that are going; you see some means I am just giving a simple example; suppose you are watching your TV in one of the rooms and your door and windows are everything closed.

So, you are, I means so many, you are assuming that no one is knowing what you are watching, but the video cable that is connected to a TV let us say that is going and is getting connected to an antenna which is on your roof; someone outside your house can connect some kind of a probe on that cable and can capture that data that is flowing through that cable that is what I mean by data coupling; that you can see; that is data coupling, And that data if I connect to another TV, I can see exactly what you are watching; I can replicate that; if I amplify it suitably and I plug into another TV I can see what you have been watching inside, ok.

And scan data during test, this is another thing, there are some additional signal pins that are kept in an IC chip, that are helpful for testing the chip; during testing those are required, but when you are not testing even during that time some data are being coming out of those pins. So, if you measure those signals actually you are getting some information what is there, ok. So, all these are typical side channels which people have explored and there are lot of research papers on these topics.

(Refer Slide Time: 09:57)



So, specifically we talked about timing analysis at times, because it is easier to understand and appreciate. Now, this kind of an attack was first proposed by Paul Kocher in the year 1995, in the mid 90's.

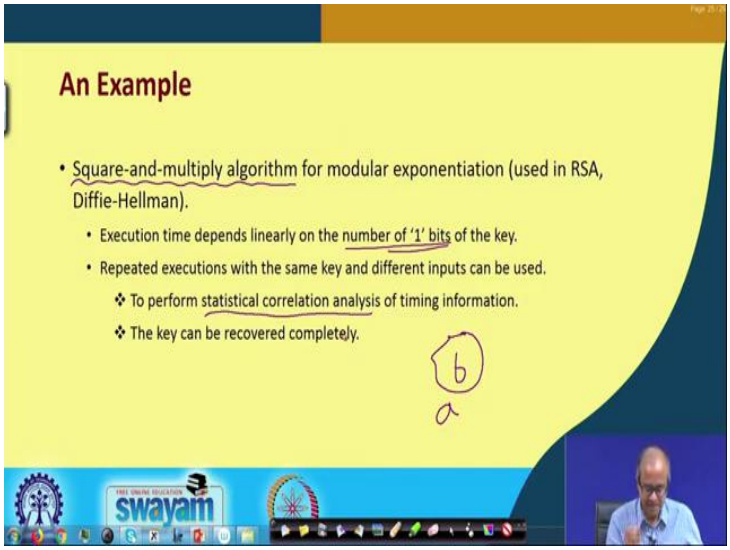
So, the idea is very simple; you are trying to measure the time that is taken for some particular operation to complete. Attacker tries to break a crypto system by analyzing the execution time. Suppose, I somehow can tell that well it is now an encryption process is starting and here it is ending. So, how much time it is taking; if I can find that out and if I can measure it, that time can give you some very important information, well.

Here I will give an example and in this process what does it try to exploit; what are you trying to exploit? You see, if you look at a symmetric key algorithm; you have already briefly seen how public key cryptography algorithm works; we have seen that RSA; we have mentioned RSA is the most widely used public key algorithm that is being used today, ok.

In RSA how you do encryption decryption; basically, we are raising a number to the power of another number modulo some other number. This a, b, n are very large numbers; this is basically how this encryption and decryption both take place. Now if I can measure the time, we will see that we can get lot of more information about this number b, this b is actually the key, the power; either the public or private key whatever,

b is the key. So, if I can measure the time, I can get lot of information about this b and let us see how?

(Refer Slide Time: 12:08)

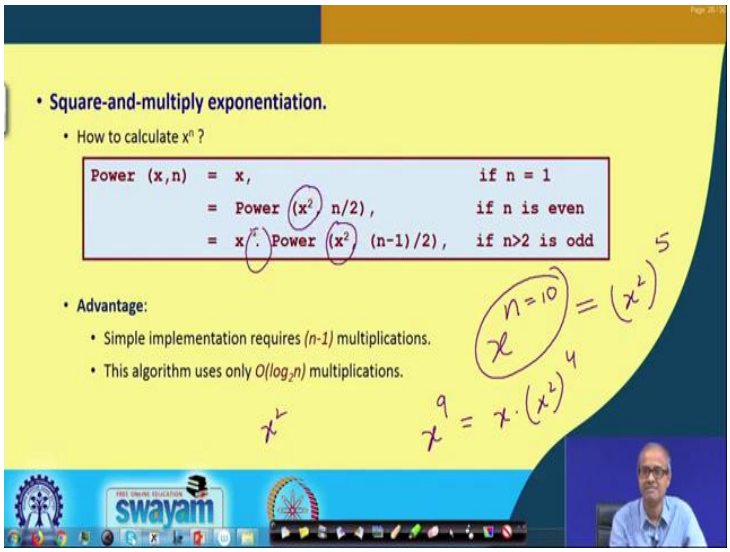


The basic way we compute this kind of modular exponential, $a^b mod\ n$; this is called modular exponentiation algorithm. And for modular exponentiation one of the most commonly used algorithm is square and multiply; this I will explain very briefly; square and multiply is one of the most efficient, you see when you compute a^b, you do not multiply a to itself $b-1$ times that is very inefficient. If b is a very large number, you need very large number of multiplications to compute a^b.

So, this square and multiply algorithm as you shall see, that the execution time will depend when we do a^b. If you treat this b as a binary number, it will consist of a string of 0's and 1's, execution time will directly depend on number of 1's in this b; it will be proportional to this number of 1's actually in fact.

So, here you can use repeated execution with the same key different inputs you can try out various ways; you can do various statistical correlation analysis and with multiple experimentation you can try to recover the key in a complete way; this is the basic idea, ok. Let us try to understand how it works without going into too much mathematical detail.

(Refer Slide Time: 13:46)

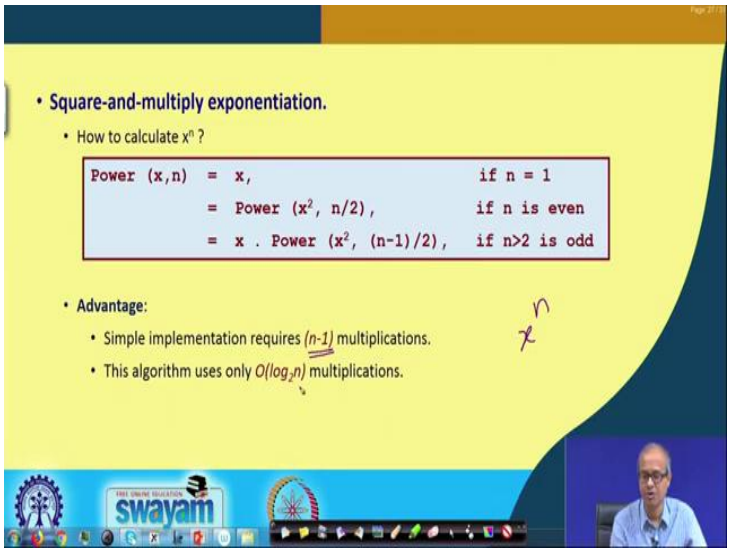


This square and multiply algorithm works like this. So, I am stating the algorithm here and I shall be explaining with the help of an example in the next slide; just remember this slide here.

$Power\ (x, n)$ means I am trying to compute x^n. So, I am writing it like this; power as a function of x and n. If $n = 1$ the result is x, obviously; if n is even, suppose if n is let us say 10, if n is 10 then I say that x^{10} is the same as $(x^2)^5$, right. So, $(x^2)^{\frac{n}{2}}$, right; similarly, if I have x^9, I am saying it is same as $x \times (x^2)^4$. So, if x is odd if not x; say if n is odd, then it will be $x \times (x^2)^{\frac{n-1}{2}}$ which is 4, $\frac{9-1}{2}$.

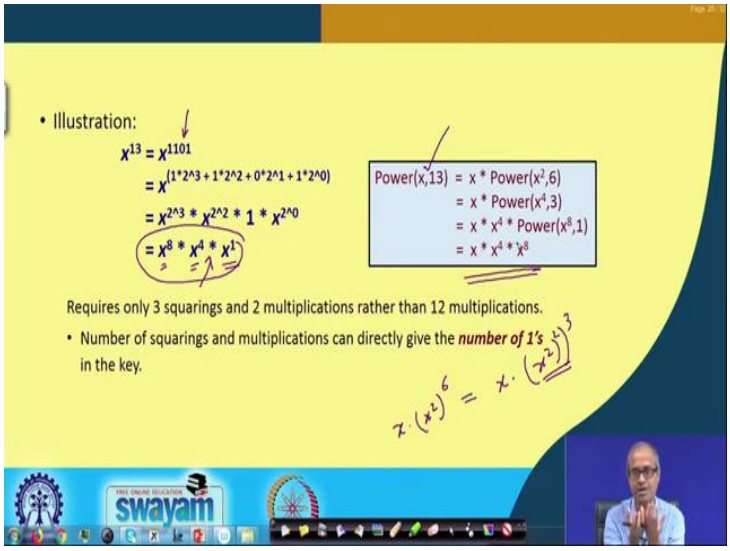
So, if you repeatedly apply this formula then the number of operations can be drastically reduced. You see here we are carrying out only two kinds of operation; one is squaring, other is multiplication; we are being squaring here or here and multiplication we are doing here, ok.

(Refer Slide Time: 15:34)



So, if you are not using this efficient method and if you are calculating x^n, then we need $n-1$ multiplications; but, if we use this technique then the number gets drastically reduced to approximately of $O(\log n)$ multiplication, $\log_2 n$. So, it becomes much faster, ok.

(Refer Slide Time: 15:55)



Let us work out a simple example; x^{13}, 13 in binary is 1101. You see these, each of these will indicate the different powers of x; this least significant bit means x^1; this 0 will be

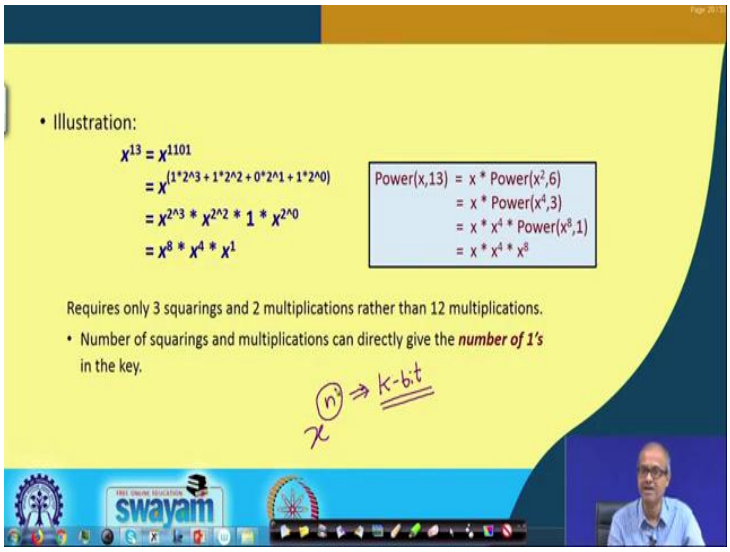
x^2; this 1 is x^4; this 1 is x^8. Now because there is a 0 here; there is a 0 out here, that is why this x^2 is missing, is not there.

So, depending on how many 1's are there, that many terms will be there in the final product so many multiplication operations will be carried out. So, for 13 if you just work out that previous algorithm the previous steps, step by step powered x^{13} is nothing, but $x \times (x^2)^{\frac{n-1}{2}}$, $\frac{13-1}{2}$ is 6.

So, in the first step we write, it is $x \times (x^2)^6$. This $(x^2)^6$ we write again as since now in the 6 is even, $(x^2)^2$. So, this we write as $x \times ((x^2)^2)^3$ which means $(x^4)^3$, like this you proceed. This 3 is now again odd so, this will be $x^4 \times (x^8)^1$, $\frac{3-1}{2}$; this $((x^4)^2)^1$ and finally, this is 1 so, this is the only extra point.

So, whatever we have got here, we have obtained the same result here. So, we need 3 squarings from x, we need to compute x^2, then x^4, then x^8 and we need 2 multiplications because there are 3 terms to be multiplied, 2 multiplications; so, 3 squarings and 2 multiplications. So, the point is the number of squarings will always be the same.

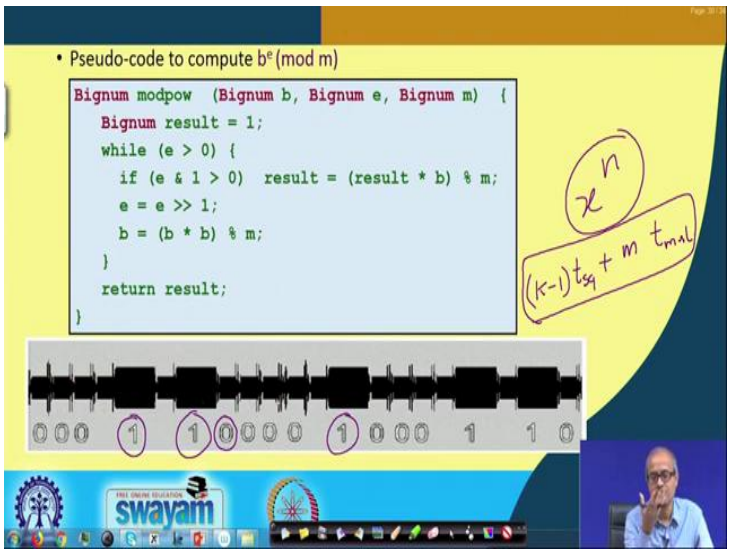
(Refer Slide Time: 18:29)



If this x^n, let us say if I write x^n, if n is a k bit number then I will always require $k-1$ squarings that is fixed, but number of multiplication will be equal to how many 1's are there in these k bits.

So, the total time will depend on the how many multiplications are there; you see the number of squaring is fixed; number of multiplication is variable. So, if I measure the time, I can get a very fair idea about how many multiplications are carried out. So, I can know how many 1's are there in this power n. So, I can get lot of information about the key.

(Refer Slide Time: 19:17)



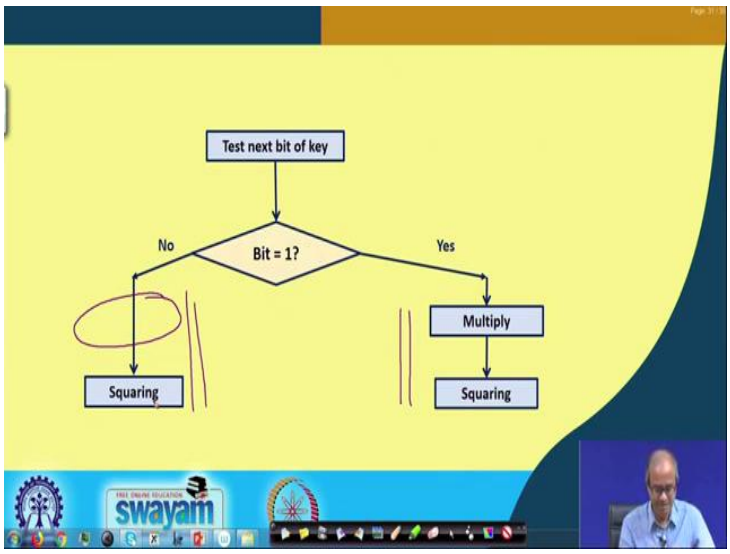
Now, this is a pseudo code in C like language that implements that same algorithm; because the numbers are very large number these are not integers, I am using a special data type called big number called Bignum. If you look at this algorithm, this is a C program, it exactly computes whatever was mentioned here. Now, the point is that suppose I am computing this kind of x^n for some value of n and during this calculation; well, here I am, the time is all right, but if I look at the total time you see this pictorially I am showing something. So, whenever there is a 1 in the bit you need a squaring also multiplication.

So, you need more time; there is another 1 you need more time, another 1 you need more time, but when there is a 0 then you need only a squaring, no multiplication. So, your time is less. So, you can say, if there are k number of bits, so, there will be $(k-1)t_{sq}$

plus, how many 1's are that I do not know; if m number of 1s are there in this n, then $m \times t_m$.

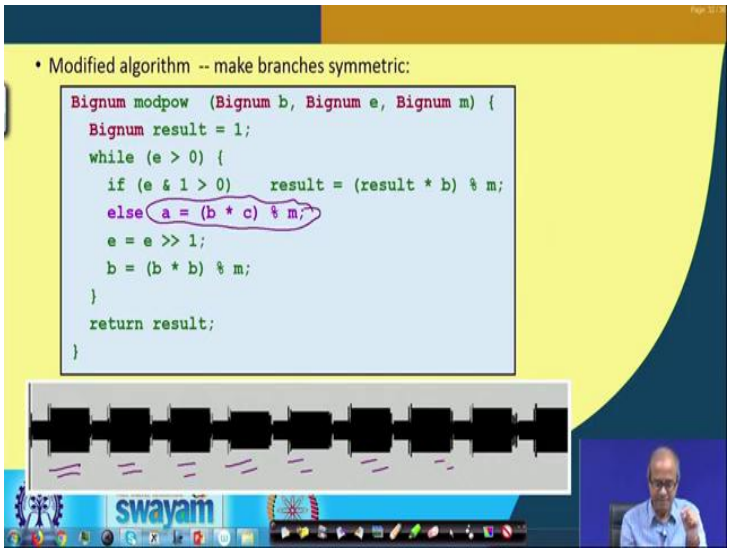
So, if I measure this time the first part is constant, second part I can directly get the value of m; I can know how many multiplications. And if I observe some kind of current waveform on the oscilloscope I can see a waveform like this, I shall see. I shall again come back to it later; you can directly see that which are 0's and which are 1's; if you observe the way from visually on the oscilloscope directly also, you can see and by measuring time also you can directly get how many 1's are there in the power, ok.

(Refer Slide Time: 21:36)



So, the algorithm basically works like this; you are checking the bits one by one; if the bit is 1, you do both multiply and squaring; if the next bit is 0, you do only squaring, no multiplication; that is how the difference in the time is coming. So, naturally the question arises if I want to stop anyone from doing this kind of an attack, if I can make these two symmetrical like, if I also add a dummy multiplication here so that both the branches of this if statement take same time, multiply, squaring, here also multiply, squaring, then the times will become same and this timing analysis cannot be mounted.

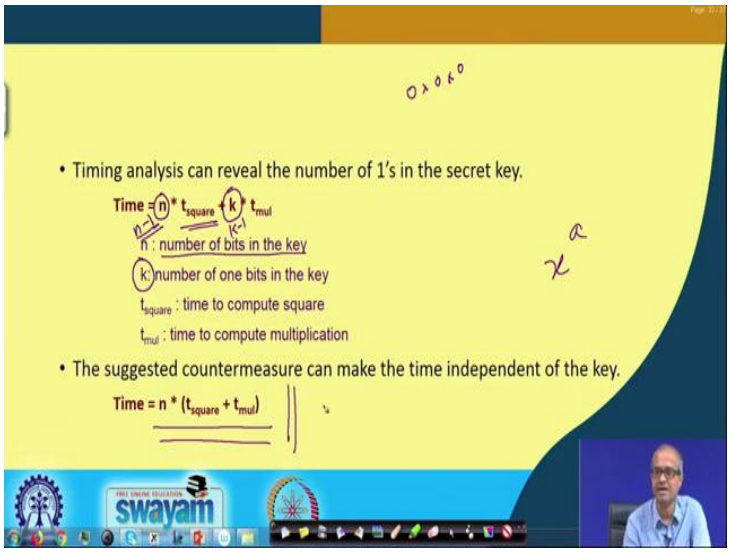
(Refer Slide Time: 22:25)



So, what I mean to say is that, if I modify this algorithm that same one and add a dummy multiplication here, this does not do anything just a dummy multiplication; I am just adding in between here, but what it results in is that, all the times now become same. So, you cannot distinguish anything just by measuring the time. So, a design will become resistant to timing analysis attack. This is how; this is why I was saying that this kind of an attack relies on carelessness in the implementation, not the strength of the algorithm; algorithm is good, but because of this feature this kind of difference was coming.

So, if I insert a dummy statement in one of the branch conditions, both becomes symmetrical; they will take same time. So, overall I cannot say how many 1's were there in the power x^n ok, this is how it basically works.

(Refer Slide Time: 23:32)

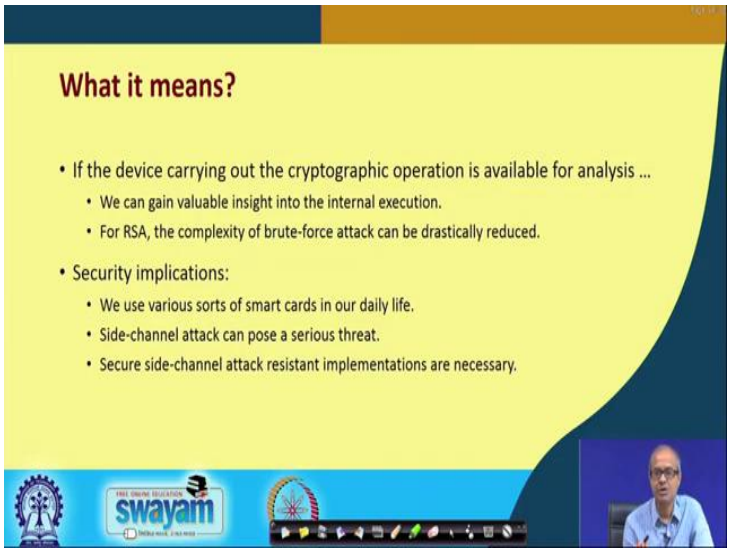


So, here is a little bit of math exactly what I was trying to say, if n is the total number of bits in the key, let us say n denotes the number of bits. So, initially I was saying and how many x^n. Let us say x^a, n is the number of bits and k denotes that how many 1's are there in the key.

Then in the normal case, in the previous case the total time was square; well actually this will be $n-1$ not n; this will be $n-1$, 1 less and k multiplied by number of so many multiplication operations. Similarly, this will actually 1 less actually this, I have just showed here for just in less this way $k-1$, because if the three once you are multiplying three things. So, you will be needing two multiplications, right and t_{square} is the time to compute a square, t_{mul} is the time to compute multiplication.

But, if we include that dummy multiplication step, your time will become something like this, this will be independent of k; this is what you want; we want that the time should not depend on k; this is what we have achieved by introducing that counter measure, ok.

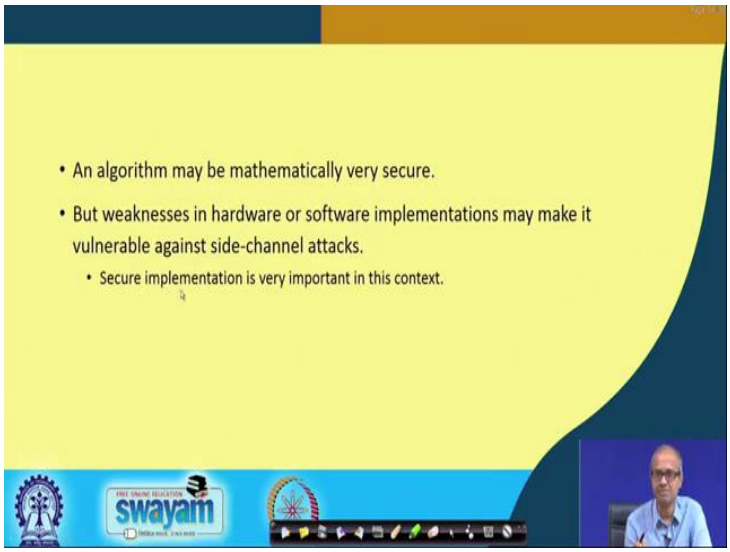
(Refer Slide Time: 25:04)



So, what it actually means is that, if the device as I have said that is carrying out the operation is available with me, it is in my hand; it is available for analysis, then I can take it in the lab. We can gain valuable insight during execution process and for example, for algorithms like RSA which relies on modular exponentiation. The complexity of brute-force data can be drastically reduced from $O(n)$; I bring it down to $O(\log n)$.

It is a drastic reduction, not even log and much less than that; I can directly tell you how many 1's are there in the k, that is a great saving. And, security implications is that as I have already said, we use this kind of device every day; ATM that is the device we use, we have so much secret information. If someone mounts our side channel attack setup on the ATM machine then whenever you swipe a card, your information will be stolen. So, these are all implications of side channel attack or side channel analysis. If someone has the device at his or her disposal, then this kind of attack can be mounted and as I had said secured side channel attack resistant implementations become the order of the day, becomes very important, ok.

(Refer Slide Time: 26:36)



So, as soon as I am repeating; the algorithm that is being implemented maybe mathematically very secure no one is doubting that, but in terms of hardware or software implementation there has been some weakness. That is why the implementation is becoming vulnerable to side channel attacks. Therefore, secure implementation becomes that much more important, right.

So, with this we come to the end of this lecture where we have tried to give you a brief idea about what is side channel attacks, side channel analysis and we talked about one kind of attack, timing analysis attack. In the next lecture we shall be also talking about another kind of an attack that is called power analysis attack, this we shall be discussing in the next lecture.

Thank you.