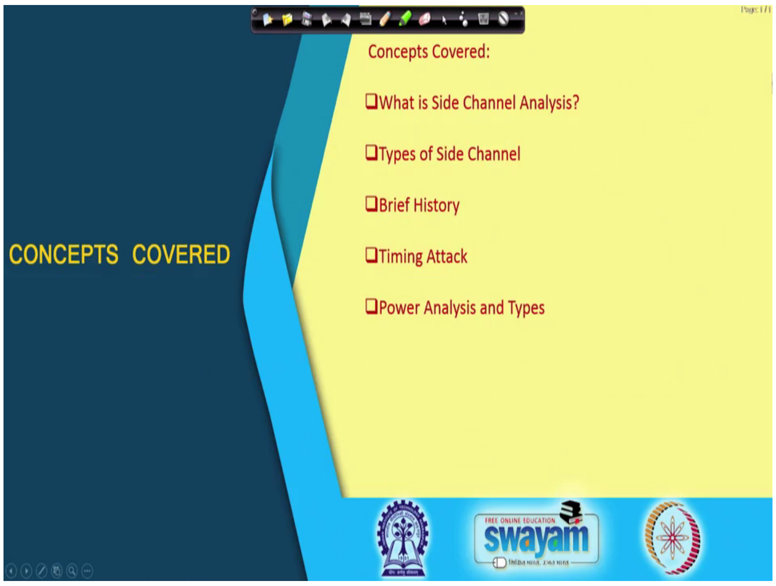
Hardware Security

Prof. Debdeep Mukhopadhyay
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 24
Introduction to Side Channel Analysis

So, welcome to this class on Hardware Security. So, today we shall be starting a new topic which is side channel attacks or Side Channel Analysis.

(Refer Slide Time: 00:25)



CONCEPTS COVERED

Concepts Covered:

❑What is Side Channel Analysis?

❑Types of Side Channel

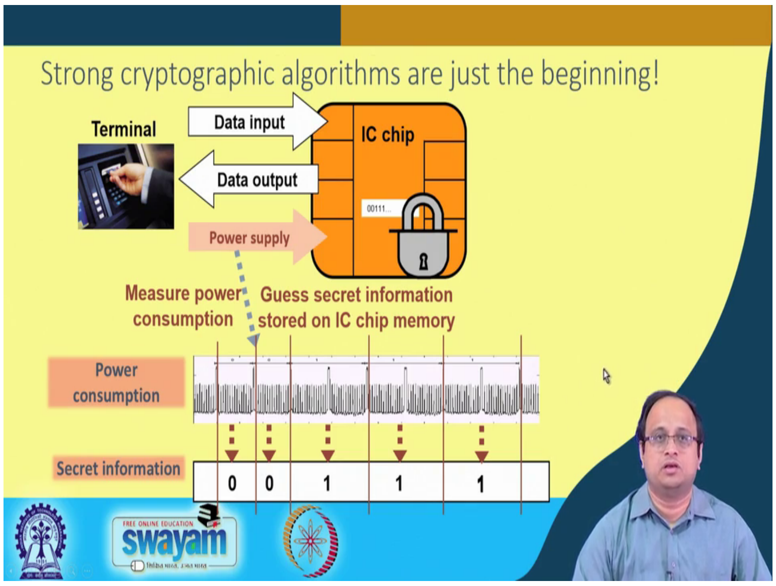
❑Brief History

❑Timing Attack

❑Power Analysis and Types

So, in particular we shall be trying to kind of understand what side channel analysis means and try to understand the certain types of side channels we shall take a brief look at the history of side channels how it started and how it kind of (Refer Time: 03:39). And finally, we shall be taking the taking a look into preliminary side channel analysis which were reported way back in mid 90s. So, the first one is called timing attacks due to Paul Kocher and finally, we shall be taking a quick look into power analysis and the types of power analysis that are available.

(Refer Slide Time: 00:59)



Strong cryptographic algorithms are just the beginning!

So, the message is that we have already been starting strong cryptographic algorithms like AES RSA ECC and we have been seeing on their hardware implementation, but the point is right strong cryptographic algorithms and there implementations are just the beginning.

So, imagine that you have got your smart card which as which is basically fortified with some kind of cryptographic algorithm and you take it to a card accepting device and you do a communication. So, it is fine because it is protected by a very strong algorithm. So, you are happy because you know that when the input and outputs are exchanged even if they fall into the hands of adversary the adversary should not have any easy way to get the key to get the secret key.

However, in practical life there are many other things which also happens which are called as a side channels for example, it can leak the power consumption of the device and that power consumption of the device essentially can be often used to trivially obtain the secret key ok. So, it can happen that, if you observe the power profile very minutely for example, this is the power profile which has been taken from a microcontroller which is typically used to realise several of our products.

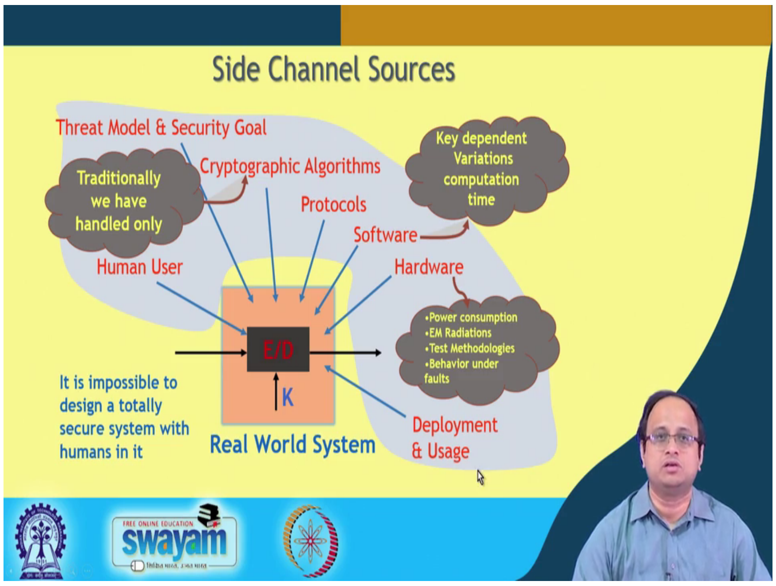
So, you can see that spikes which are available right in this power presses they are not of the same time ok. So, these power this is can be sometimes you can see the spike sometimes can be narrow and sometimes can be quite kind of quite can be broader so there district and idea here is that typically right if you take an RSA kind of algorithm as we have discussed that the basic operation right in the such kind of mathematical function right it is essentially suppose the square and multiply algorithm.

So the square and multiply algorithm essentially is nothing, but an efficient way of calculating the exponentiation. So, the idea is the if the secret bit is 1 then suppose we have very naive implementation where you do this like you always do a squaring and do a conditional multiplication; that means, if the key bit or the exponent bit is 1 then you do a conditional multiplication.

So, that implies that there is an extra power consumption and that power consumption can get reflected by a different kind of spike. And if you can and if the attacker can understand that and can understand that this pattern is distinct from the other pattern then it can very well distinguish a 0 from a 1 in the secret key and that can lead to a trivial leakage of the secret key although the mathematics is quite nice.

So, therefore, right these kind of attacks or side channel attacks typically do not rather their many focus right is the implementation. So, they basically target the implementation and although if we have very strong mathematical background because of implementation is not proof against side channel attacks it still remains vulnerable.

(Refer Slide Time: 03:51)



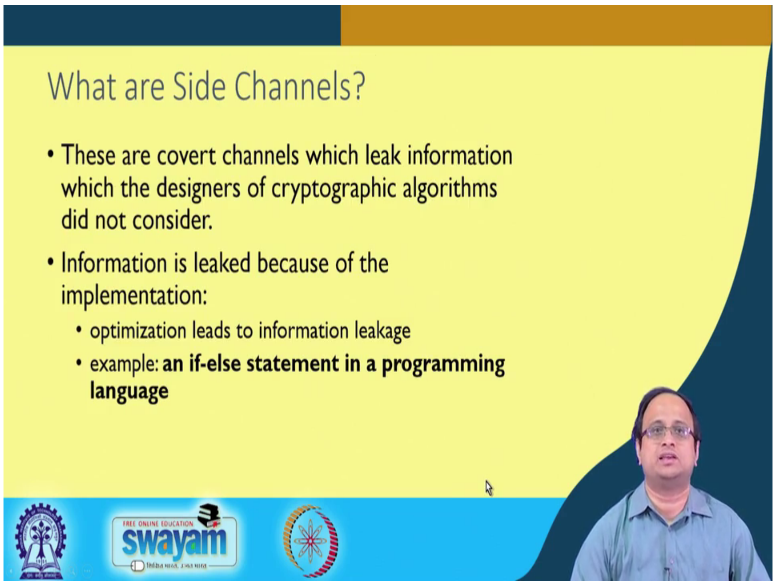
So, there are different side channel sources in the real world for example, if you take an encryption or decryption device and kind of port it could be either in the form of a software or it could be a in the form of a hardware, but if you take it right there are several types of thinks which can happen. So, traditionally we have been handling cryptographic algorithms ok, but there are different like if you want an end to end security there are several other concerns that we need to take care of.

For example, the threat model and security goal in which you are putting the device is an important concern the protocols which is essentially employee this kind of devices, the way you are implementing them. So, it could be in the form of software or it could be in the form of hardware and also finally, how you use them that is also very important and finally, right you have the human which is often the you know weakest link in the security.

So, it is probably impossible you know to design a totally secure system with humans in it, but any way we are not talking about humans in this course. So, we are mainly talking about other things like which we can manage. And if you see overall right then traditionally we have been handling cryptographic algorithms, but on the contrary right there are software hardware implementations of the ciphers which often can lead to key dependent variations which can have the variations in computation times which can lead to protocol as timing attacks if you consider a hardware often you have got power consumption the electromagnetic radiations.

The test methodologies which we employee because testing is a very important aspect right because when you are deploying you would like to know that or ensure that your chip or design that is working as intended. But often it turns out that those test methodologies can be in turn used for attacks because, they also open up you know like avenues for the adversary to look what is inside and that is essentially catastrophe. So, likewise right we will also see that behaviour under falls can also be a way of you like the how the fault essentially propagates in a cipher can also be utilised to develop attacks which are called as fault attacks.

(Refer Slide Time: 05:55)



What are Side Channels?

- These are covert channels which leak information which the designers of cryptographic algorithms did not consider.
- Information is leaked because of the implementation:
  - optimization leads to information leakage
  - example: **an if-else statement in a programming language**

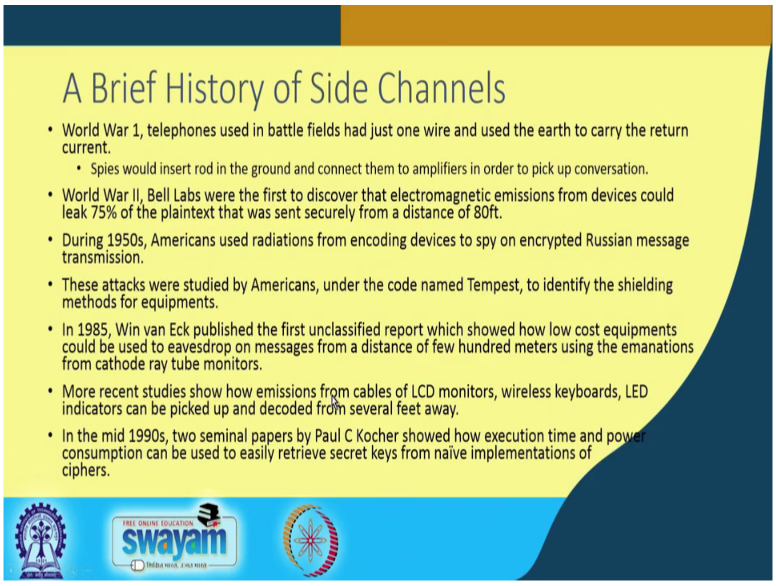
So, therefore, right I mean the whole point is right that side channels therefore, are essentially kind of secret channels which leak information which the designers of the cryptographic algorithms did not consider. Because traditionally that the attack model which you know like which is covered in at least traditional theoretical cryptographic algorithm constructions are essentially what are called as you know like you got the cipher text only model you have got the chosen, you have got the known plaintext model, you have got that you know like the chosen plaintext models, you have got the chosen cipher text models, but they do not take care of this kind information leakages ok. So, which are called as side channels.

So, information is leaked because of the implementation and therefore, the target or the main you know like discussions about when we talk about side channel protections are based on the implementation they often happen because of optimisation. So, therefore, right for example, if you have got optimisations and the optimisations are essentially concerning only performance right often you will find out that they can also leak in terms of side channels.

For example, if you have a code for example, or a program and if you are using an if else statement right then it may turn out that if else statement is in turn dependent upon a secret key then it may happen that I mean then it can be utilised there is a vulnerability in your program which you need to kind of protect against ok.

Something that is often not stressed you know like when we go through our beginning beginners programming language courses right because at that point our main objective is to know how to code. So, that the functionality is realised.

(Refer Slide Time: 07:35)



A Brief History of Side Channels

- World War 1, telephones used in battle fields had just one wire and used the earth to carry the return current.
  - Spies would insert rod in the ground and connect them to amplifiers in order to pick up conversation.
- World War II, Bell Labs were the first to discover that electromagnetic emissions from devices could leak 75% of the plaintext that was sent securely from a distance of 80ft.
- During 1950s, Americans used radiations from encoding devices to spy on encrypted Russian message transmission.
- These attacks were studied by Americans, under the code named Tempest, to identify the shielding methods for equipments.
- In 1985, Win van Eck published the first unclassified report which showed how low cost equipments could be used to eavesdrop on messages from a distance of few hundred meters using the emanations from cathode ray tube monitors.
- More recent studies show how emissions from cables of LCD monitors, wireless keyboards, LED indicators can be picked up and decoded from several feet away.
- In the mid 1990s, two seminal papers by Paul C Kocher showed how execution time and power consumption can be used to easily retrieve secret keys from naïve implementations of ciphers.

So, therefore, right I mean here is a brief history of side channel ok. So, how side channel started? So, it is not that side channels are very new although you know like in academics is probably very new. But if you go in to classified documents right for example, World War 1, then telephones used in the battle fields had just one wire and you know like the earth was actually used to carry the return current ok. And therefore, what spies did is there you know like they took or the inserted rods in the ground and connect them to amplifiers in order to pick up conversation ok. So, this an example of side channel ok.

Likewise in World War II, bell labs essentially first discovered that electromagnetic radiations from devices can be used or can be utilised to leak 75 percent of the plaintext that are that was sent securely from distance of something like 80 feat ok. So, this an classical example of side channel attacks ok. So, likewise right during 1950 or around 1950 Americans used radiations from encoding devices to spy on encrypted Russian messages and they essentially did it kind of on the various equipments which were essentially prevalent ok.

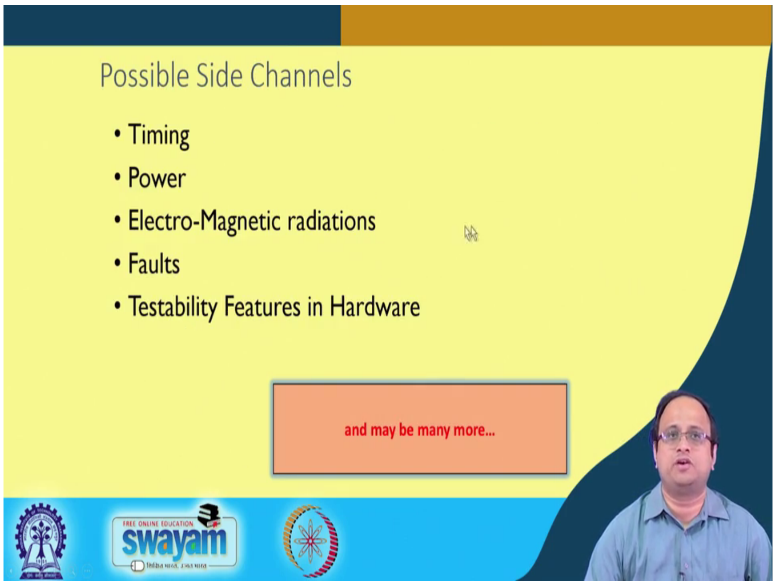
So, likewise you know like this attacks where therefore, studied in depth by Americans and there was code name given to this project which is famously call as the tempest and this was used to identify what kind of shielding methods should be essentially protect kind of available so, that these attacks essentially can be circumvented or prevented.

So, in 1985, there was a very interesting experiment which was done by Win van Eck which essentially showed or probably the first unclassified report which showed how low cast equipments could be used eavesdrop on messages from a distance something like even 100 meters and they were used essentially they were using the emanations from the CRT or the cathode ray tubes.

And although it was kind of believe till a point that the CRT monitors are probably responsible for this kind of leakages more recently I think around 2003 actually it was showed that even LCD monitors, wireless keyboards or LED indicators can be picked up and also can be decoded from a feet apart or feet away ok.

So, but, but it turns out that in academics right I mean there are two very important seminar papers which was written in the mid 90s around 1996 90s 1999 around that time frame, when they were two seminar papers which was written by Paul C Kocher which showed how execution time and power consumption can be systematically utilised to retrieve the keys from implementations ok. And this essentially will be the basis you know like of what we essentially today study as the topic of side channels.

(Refer Slide Time: 10:43)



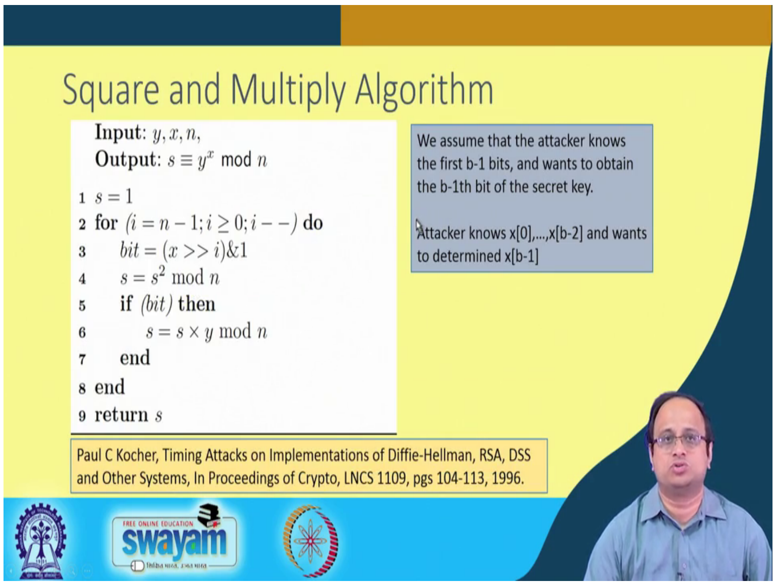
Possible Side Channels

- Timing
- Power
- Electro-Magnetic radiations
- Faults
- Testability Features in Hardware

and may be many more...

So, therefore, right I mean there are different types of side channels which has over you know over the years grown and it is still growing probably. For example, some of the classic ones are like which are based on timing, power, electromagnetic radiations, faults, testability features in hardware. So, you can just add on to the list for example, it could be even sound ok.

(Refer Slide Time: 11:03)



Square and Multiply Algorithm

```
Input: $y, x, n,$
Output: $s \equiv y^x$ mod $n$
1 $s = 1$
2 for $(i = n - 1; i \geq 0; i - -)$ do
3     $bit = (x >> i)\&1$
4     $s = s^2$ mod $n$
5     if $(bit)$ then
6         $s = s \times y$ mod $n$
7     end
8 end
9 return $s$
```

We assume that the attacker knows the first b-1 bits, and wants to obtain the b-1th bit of the secret key.

Attacker knows x[0],...,x[b-2] and wants to determined x[b-1]

Paul C Kocher, Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS and Other Systems, In Proceedings of Crypto, LNCS 1109, pgs 104-113, 1996.

So, so, there could be many more. So, let me now you know just to kind of hit the point take a specific example which was published by Paul Kocher in crypto paper of 1996 which showed how a timing attack works.

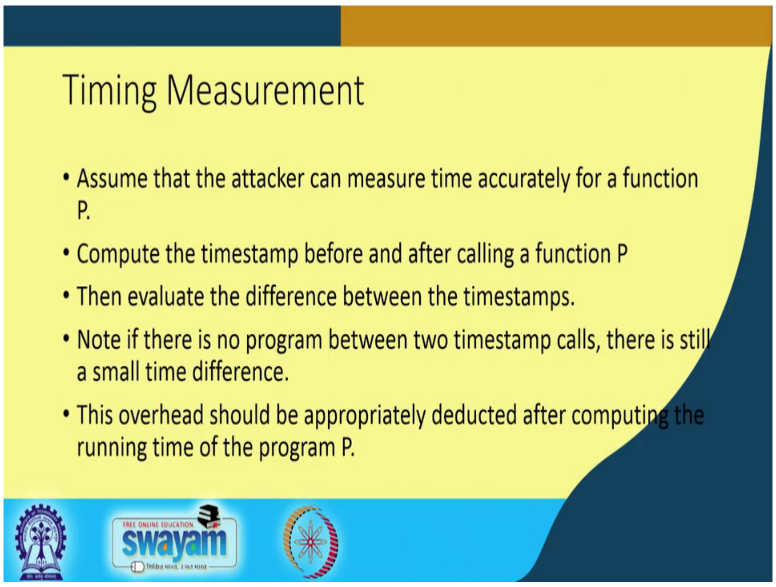
So, this essentially is nothing, but a program or a code which essentially performs the square and multiply algorithm. So, if you read right then essentially it starts from the MSB of the exponent. So, you are basically trying to calculate y power of x mod of n. So, this is traditionally what we do in an RSA description for example, so, you can imagine that x essentially stand for the secret.

So, therefore, right if I the simple code would be that I will start with i equal to n minus 1 and I will go down and I will essentially you know like get the bit and I will always do a squaring operation and if I see the bit and to see that the bit is 1 then I do a conditional multiplication operation. So, this is a very nice implementation of the exponentiation routine modulo exponentiation routine. So, here is an attack settings so, we assume that the attacker knows the first say b minus 1 bit. So, this is an iterative attack ok. So, it kind of retrieves the x values 1 by 1.

So, let me assume that it knows the first x b minus 1 bits and wants to obtain the second the next bit ok. So, it wants to obtain the you know like. So, basically what I mean to say is that suppose the attacker knows till form x 0 to x b minus 2 ok. So, it wants to determine the next bits so, basically wants to determine x b minus 1 ok. So, so basically right you can say. So, may maybe you can just correct this should be the bth bit which I essentially want to obtain as an attacker.

So, the idea is the attacker wants to know the next bit. So, if I do not give you any side channel information imagine that the secret key is just a random bit string right. What is the probability of guessing x b minus 1 it should be half because it could be 0; it could be 1, with an equal probability. But what we will see is that using timing information you can actually understand x b minus 1 with a larger probability and essentially that gives you the avenue for doing a side channel attack to recover the entired exponent of x.

(Refer Slide Time: 13:21)



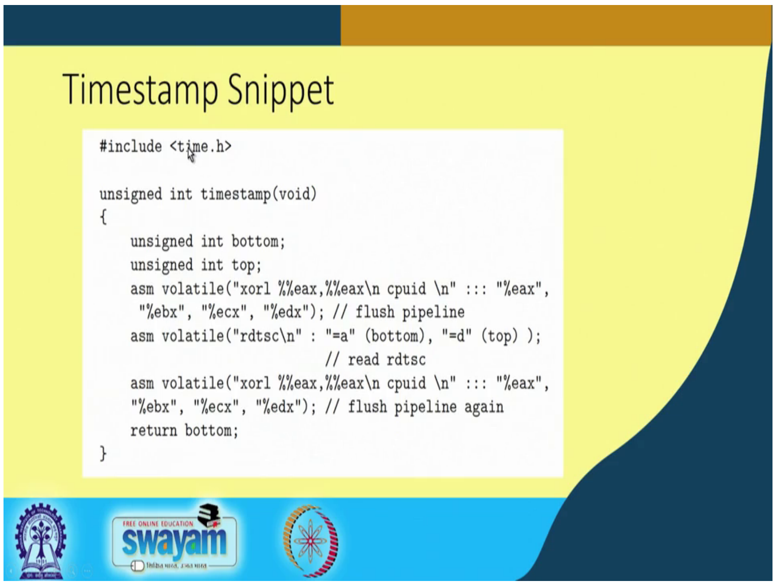
Timing Measurement

- Assume that the attacker can measure time accurately for a function P.
- Compute the timestamp before and after calling a function P
- Then evaluate the difference between the timestamps.
- Note if there is no program between two timestamp calls, there is still a small time difference.
- This overhead should be appropriately deducted after computing the running time of the program P.

So, the description of the attack is as follows you basically relies upon timing measurement. So, it is called as timing attacks. So, we will assume that the attacker can measure timing accurately for a function P ok. So, it can measure timing very accurately. So, the idea is that we will compute the timestamp before and after calling the function P. So, that mean before I call a function P. So, it could be the squarer multiply routine I will take a timestamp and after that I will again take a timestamp and then I will take the difference of this timestamps.

So, note that even if there is no program between these two timestamps still you will get a non 0 value, because there will a small timing difference and which is due to the overheads ok. So, there should be a measurement for this overhead and this overhead should be eventually deducted from the time to get a more accurate estimate of what time essentially has been required to do a perform to do a essentially performer execute a program P.

(Refer Slide Time: 14:17)



Timestamp Snippet

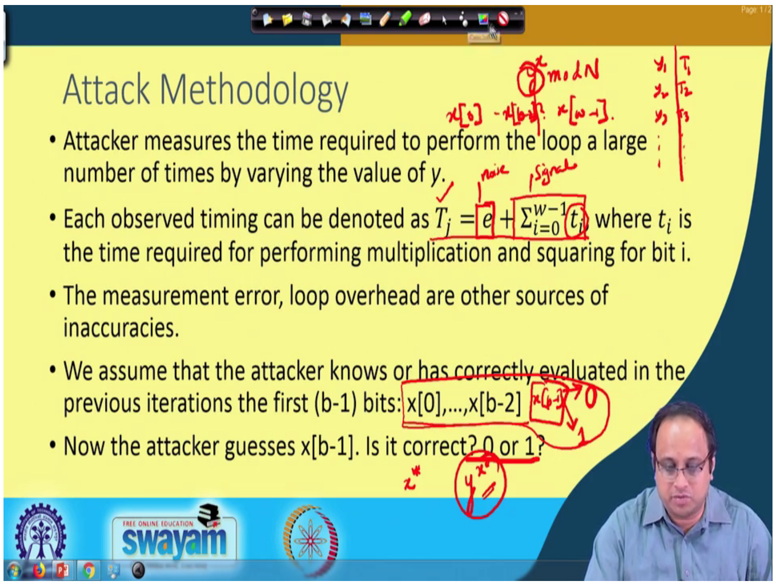
```
#include <time.h>

unsigned int timestamp(void)
{
    unsigned int bottom;
    unsigned int top;
    asm volatile("xorl %%eax,%%eax\n cpuid \n" ::: "%eax",
     "%ebx", "%ecx", "%edx"); // flush pipeline
    asm volatile("rdtsc\n" : "=a" (bottom), "=d" (top) );
                                  // read rdtsc
    asm volatile("xorl %%eax,%%eax\n cpuid \n" ::: "%eax",
    "%ebx", "%ecx", "%edx"); // flush pipeline again
    return bottom;
}
```

So, here is a timestamp snippet which you can use ok. So, I have just given the code for doing a timestamp execution and you can see that there are certain important you know like a s m volatile routines which has been called as a part of this function ok. So, for example, there is a c p u i d instruction which has been used the objective of the c p u i d instruction is to flash the pipeline ok. So, that you essentially get more accurate estimate and the register which is being used to kind of monitor the time is essentially and r d t s c timestamp counter. So, this is a specific register which we read to get an accurate measurement of time so this should be much more accurate than doing something like get time of day for example.

And then we do I get a c p u i d instruction; that means, flush the pipeline again and finally, right this essentially is returned this bottom value essentially is return which gives you in measurement of the time which is or the timestamp. So, you basically take a timestamp before a program execute before a function call and you take another timestamp before after the function call and you take a difference of these two timestamps. So, that is how you measure it.

(Refer Slide Time: 15:29)



Attack Methodology

- Attacker measures the time required to perform the loop a large number of times by varying the value of *y*.
- Each observed timing can be denoted as $T_j = e + \Sigma_{i=0}^{w-1} t_i$ where t_i is the time required for performing multiplication and squaring for bit i.
- The measurement error, loop overhead are other sources of inaccuracies.
- We assume that the attacker knows or has correctly evaluated in the previous iterations the first (b-1) bits: x[0],…,x[b-2]
- Now the attacker guesses x[b-1]. Is it correct? 0 or 1?

So, assume that you can measure timing pretty accurately; this is how Kocher attack works. So, the attacker measures the time required to perform the loop; that means, you have a essentially the loop for doing the square multiply number of times by varying the value of y. So, basically what you are doing is your computing y power of x.

So, let me you know like choose several values of y and the x is constant because the x is fixed to a secret key which you do not know or at least you do not know beyond b minus 1 bit. So, then you start accumulating several timing information ok. So, you basically give in the y 1 you get t 1 you get y 2 you get t 2 these are all timing information which you start accumulating.

So, each observe timing can be denoted as set T j which is equal to e plus or small e plus sigma i equal to 0 to w minus 1 t i. So, t i is the time required for performing an multiplication and squaring for bit i, that is a specific iteration in the loop actually. And then imagine that your secret right which you have; that means, the secret key is w bit value it is a w bit value. So, you basically observe the timing for everyone every step is t i and you kind of some over from i equal to 0 2 w minus 1 and that essentially is this component of the timing requirement. And small e some kind of you know overheads you could be due to the measurement error it could be the other sources of inaccuracies.

So, what we assume as I said that the attacker know or has correctly retrieved or evaluated the first iterations. So, it as basically got from x 0 to x b minus two. So, this the attacker knows what the attacker wants to now know is the x bit x b minus 1. So, it wants to know that whether x b minus 1 is 0 or whether it is 1 ok. So, how can we use this timing information to essentially understand what is the next bit. The idea is that or the central idea is that whenever you have got a bit which is 1 right then you are doing a multiplication. So, therefore, there is an increase in time requirement that is the basic sort of you know like you basic kind of observation, but you need to exploit it.

So so, therefore, right what the idea the attack technique is as follows. So, you basically take you basically do this you. So, let me describe next the attack methodology. So, now, we will assume that the attacker has been able to correctly monitor time. So, now what we want is we want to explode that time or timing information. So, the basic observation is that whenever you have got a secret bit which is 1, you are essentially doing an addition and multiplication operation and because of this there is an extra time that you are required that every loop is requiring. So, therefore, the loop which has a multiplication because of the secret bit b 1 should be slightly more than essentially when you do not have the multiplication operation.

So, therefore, right what the attacker does is as follows. So, you know that your target is your basically your basically calculating you are basically calculating y y power of x. So, you already have y and you are raising y power of x. So, this is what you are doing in your in during a computation right. So, what the attacker does is basically starts varying y so, its starts to take different values of y. So, it basically takes it basically takes y 1 y 2 y 3 and so on and also observes a corresponding timing like T 1 T 2 T 3 and so on. So, it basically observe this statistics ok.

So, the statistics T j for example, any jth value here essentially is written over here. So, this essentially has got two components, the 1st one e is basically nothing, but a measurement error. So, it is basically due to several source of inaccuracies which can come up because of system and other things because of parallel process being executing, but there all noise and then you have got another component which is essentially the signal of your measurement. So, this is the noise essentially. So, you have got a noise component and you have got a signal component which essentially is an important information.

So, typically your exponent x is as I said is the w bit string for example. So, you have got x 0 to say x w minus 1. So, these are the w bits of x and you know till x b minus 2. So, you know till this point, but you do not know from this point ok. So, therefore, right what this part essentially. So, this is essentially the execution time for the entire operation that is when you are processing the entire exponent from 0 x 0 2 x w minus 1.

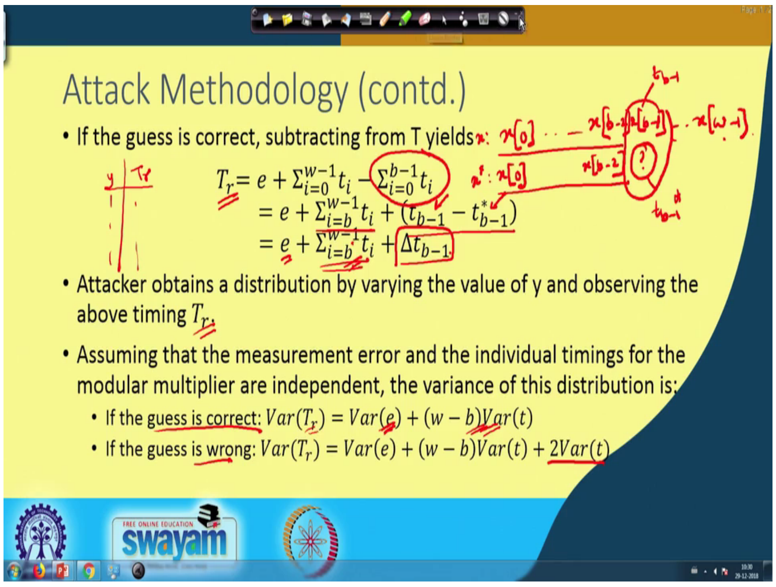
So, each iteration of the loop right suppose it takes you t i amount of time ok. So, the first loop says takes t 0 or the second loop takes t 1 and so on. So, if I some of some them of from 0 to w minus 1 then get I get an estimation of the rural timing which is required that is T j.

So, now the question is right how. So, we have got this statistics in this way and we know that this is the formulation for that. So, now, is you see there the attacker knows form x 0 to x b minus 2, but it does not know what is your x b minus 1. So, this is kind of unknown to the attacker ok. So, the attacker if we does not have a side channel attack it has got a half probability of guessing.

So, now we will see that how we can distinguish whether it 0 is correct or whether it 1 is correct guess ok. So, therefore, let me say that suppose I make a 0 guess for x b minus 1 or I make a 1 guess for x b minus 1. So, it could be anything which I can guess right and then I create this exponent. So, this exponent essentially is nothing, but suppose denoted by x star this is comprises of you know like from 0 to b minus 2 what I have already retrieved and the b minus 1 bit which I guess ok. So, it can be correct it can be wrong if for example, is 0 is the correct bit and if I guess 0 then I get essentially the correct string otherwise I get a wrong guess for x star.

So, therefore, now what I do is I do this computation y x star y to the power of x star. So, remember that x star is not a w bit stream, but essentially is a partial stream and therefore, right if I do this computation and again I do this y power of x star and you know like a I do this y power of x star for again certain number of observations I will again get a timing statistics, I will get again get a timing profile. So, let me right that timing profile I mean see the timing profile and see that.

(Refer Slide Time: 22:11)



Attack Methodology (contd.)

- If the guess is correct, subtracting from T yields:

$$T_r = e + \Sigma_{i=0}^{w-1} t_i - \Sigma_{i=0}^{b-1} t_i$$
$$= e + \Sigma_{i=b}^{w-1} t_i + (t_{b-1} - t_{b-1}^*)$$
$$= e + \Sigma_{i=b}^{w-1} t_i + \Delta t_{b-1}$$

- Attacker obtains a distribution by varying the value of y and observing the above timing T_r.
- Assuming that the measurement error and the individual timings for the modular multiplier are independent, the variance of this distribution is:
  - If the guess is correct: $Var(T_r) = Var(e) + (w-b)Var(t)$
  - If the guess is wrong: $Var(T_r) = Var(e) + (w-b)Var(t) + 2Var(t)$

So, therefore, the timing profile essentially is nothing, but this right is essentially this part of the term. So, you see that it is this part because essentially it has got and you see that it is accurate because here you essentially can accurately you exactly know the system you are essentially you are doing in your system.

So, therefore, right if you take a difference of these two components then you will see that the difference that is essentially, if I take the difference of this timing with the previous timing then you will have the difference as shown here; that means, this is one part of the. So, essentially nothing, but you have got from b b to w minus 1 which essentially is there, but remember that the previous things we will get cancelled out because you know even for x 0 till x b minus 2 and x b minus 1 till x w minus 1 this is your target exponent and in your x star right. So, this is your x and this is your x star. So, you have got from x 0 till x b minus 2 which are all correct only this is something which you are guessing ok.

So, therefore, if you take the difference right then this from this part to this part they will get cancelled out ok. So, therefore, the sigma terms will get cancelled out from 0 to b minus 2. So, here there will be a difference which essentially shown by these two timings ok. So, you have this t minus t b minus 1 minus t b minus 1 star because, the timing required here is say t b minus 1 and the timing required is say t b minus 1 star ok.

So, the difference of timing between these two things essentially can be written as essential this part that is you have got from and the remaining part is from b to w minus 1 and this is the extra timing that is require from b to w minus 1. So, therefore, this T r essentially is nothing, but the difference of the two timings; that means, you have got T j timing and you have got a timing requirement for x star which essentially you have manufactured by your previous knowledge from 0 to b minus 2 and your guess of b minus 1 and you take the difference ok.

So, now the attacker obtains the distribution by varying the value of y and observing this above timing ok. So, if you can see that if I can just vary the value of time right if I just vary for example, different values of y I will get different values of T r. So, I will get a statistics in this fashion.

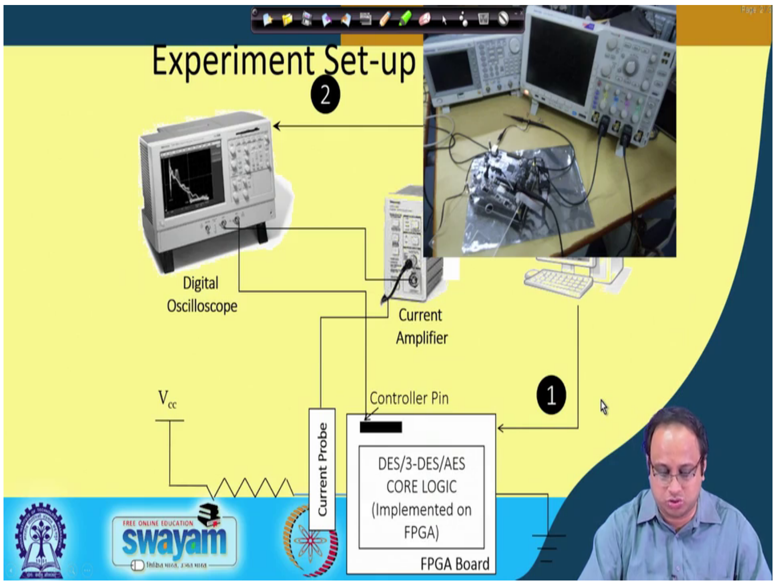
So, now there is an interesting observation the observation is that, if I take this statistics of timing and if I concern compute the variance of this timing distribution. Then the variance you can write by assuming that the noise is independent and every loop is independent, you can write them that is basically neglecting the covariance you can write the variance of T r as variance of e plus w minus b variance of t why in a because this is your also I am assuming that every you know the variance is constant across every loop.

So, therefore, right this is this is a part. So, you will get for this you will get the variance of e and for this part you will get w minus b variance of t ok, but when your guess is correct then this will be 0 ok, but if your guess is wrong then this essentially will be reflected in to two times variance t because, this the variance you do this and the variance you do this will get added up because the essentially they are different.

And therefore, right you will get a two times variance of t which shows that the variance you will get increased when you are when you are making a wrong guess so; that means, you can distinguish I mean whether this guess is correct or whether this guess is wrong by just simply trying to measure the variance of T r and if the variance of T r is more than you assume that or then you can say that my guess was correct and if the variance was less then your guess was probably wrong ok.

So, this way you can distinguish a correct guess from a wrong guess and then you know like if you can retrieve the bth bit or b minus 1th bit you can do that iteratively starting from 0 to w minus 1. So, you can iteratively an in order n; that means, in time proportional to the length of the secret you can retrieve the secret and therefore, you can have extremely efficient attacks which essentially can be utilized you know like for side channel attacks.

(Refer Slide Time: 26:49)



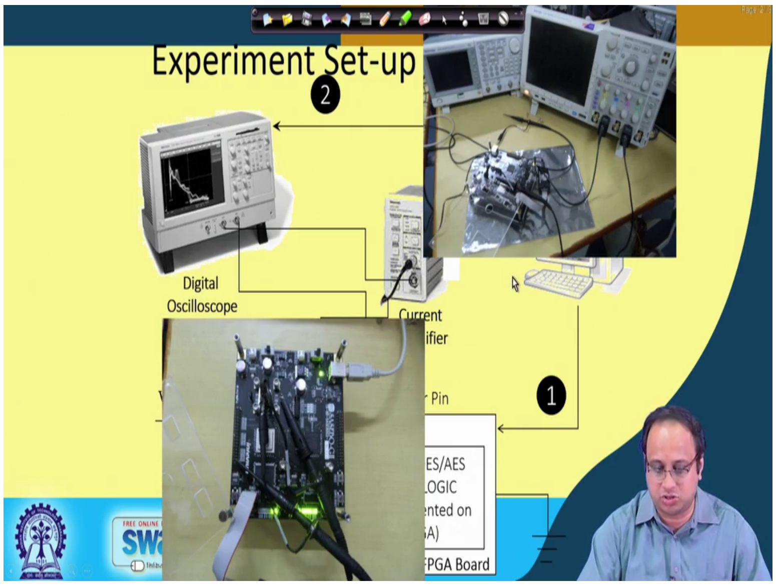
Experiment Set-up

So, therefore, right I mean. So, this is one way of doing the attacks and there are other ways of doing analysis of side channel attacks also for example, you can also do power attacks ok. So, here is an experimental setup for doing power attacks. So, you can see that there are different components in our setup.

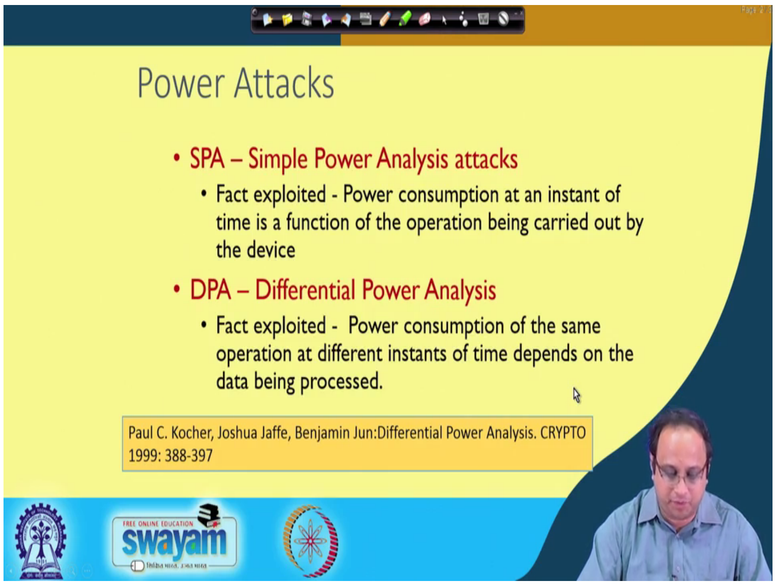
So, on one hand you have got the target device on which you are you know like implementing your hardware for encryption like it to be DES, 3 DES, AES or some other things. And then you have gotten varies good oscilloscope through which you basically measure the power consumption of your device and there is a current amplifier typically which takes the current and essential is gets amplified and it gets kind of stored in the oscilloscope there is a central P c which essentially controls the entire manually.

So, what it does it basically sense a plain text data to the hardware, the hardware takes the ciphertext and again stores it back into the P C and in synchronisation it also captures the power profile by the oscilloscope and is basically also send back to the P C the P C therefore, stores the plain text, the ciphertext and also along the comparing power trace ok. Now with this kind of in also here I am snapshots to show you how it looks like. So, this is the target board and these are the oscilloscope and the companying hardware ok.

(Refer Slide Time: 27:57)



(Refer Slide Time: 28:03)



Power Attacks

- SPA – Simple Power Analysis attacks
  - Fact exploited - Power consumption at an instant of time is a function of the operation being carried out by the device
- DPA – Differential Power Analysis
  - Fact exploited - Power consumption of the same operation at different instants of time depends on the data being processed.

Paul C. Kocher, Joshua Jaffe, Benjamin Jun:Differential Power Analysis. CRYPTO 1999: 388-397

So, now when you know like have captured capture power profiles like that there are two types of broad category of categories of power attacks which you can do, one categories what is called a Simple Power Attacks or SPA and a second one is called DPA or Differential Power Analysis. So, essentially in Poul C Kocher prepare in crypto 99, it was shown that these kinds of power measurements can be very powerful to do attacks ok.

For example, in simple power attacks as we have already kind of pointed out is that the fact which is exploited they are is that the power consumption typically depends upon the operation or the underlying operation. For example, the when you are talking about the squarer multiply algorithm the power consumed for doing a squaring is different from the power consume that is different from that is required for doing a multiplication ok.

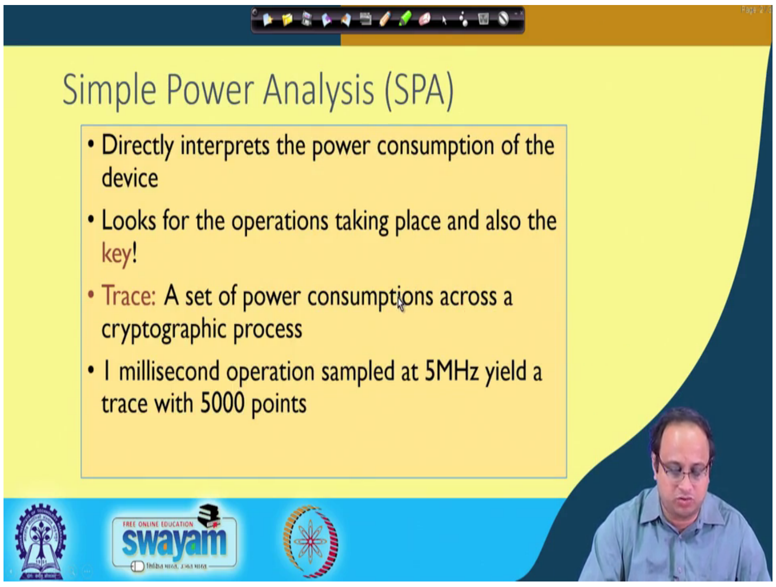
Now, this can be protected there are different variations of the squarer multiply algorithm for example, one way of protecting this would be like let me do a dummy multiplication and every time step its not very efficient, but still you should be able to protect against simple power attacks. But then there is a more powerful form of adversary which is called as differential power analysis and we will study them in more details in our future classes the fact that is exploited here is that power consumption of the same operation at different instants of times depends upon the data that is being processed and that essentially is more fundamental and therefore, right can due to even more differentiating attacks ok.

So, an implementation which is protected again simple power attacks like just now as I said that if you are you know doing a damming operation can still be attacked using a DPA attack or differential power attack because, it the basic cause is even more fundamental; that means, it basically kind of capitalize on the fact that power consumption depends upon the underline data which is being process.

Then therefore, right what happens is that when you are measuring power and measuring power very accurately you get an information about the underline data and that essentially is dangerous you can imagine intuitively why it. So, for example, in AES as we have started right what the attack what the designer once you is to get information after say ten rounds of encryption.

But suppose if you are measuring power and that gives you information about data after say first round or second round or so on; that means, right you are essentially not adhering to the specifications or the theoretical specifications of the of the algorithm and therefore, it can lead to attacks.

(Refer Slide Time: 30:35)

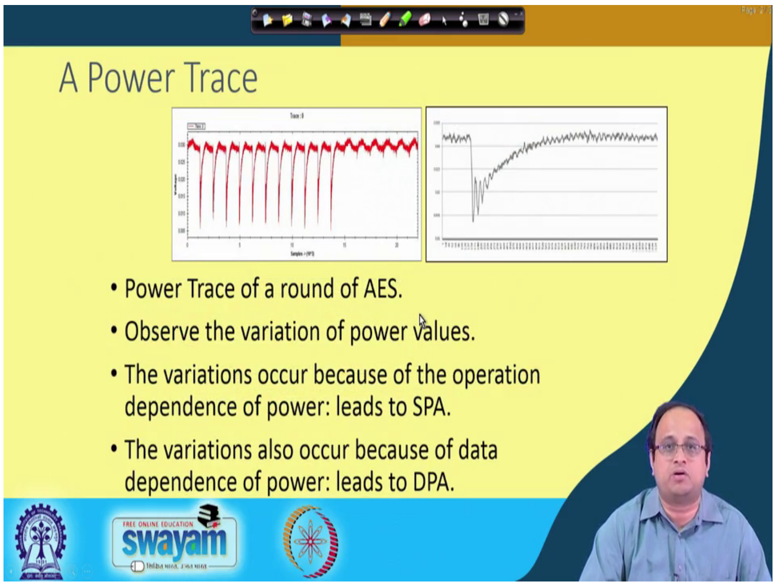


Simple Power Analysis (SPA)

- Directly interprets the power consumption of the device
- Looks for the operations taking place and also the key!
- Trace: A set of power consumptions across a cryptographic process
- 1 millisecond operation sampled at 5MHz yield a trace with 5000 points

So, therefore, this is again a very powerful form of adversary and therefore, we need to kind of consider them. So, let so, let me kind of stop by you know like showing you an example power trace of AES.

(Refer Slide Time: 30:45)



A Power Trace

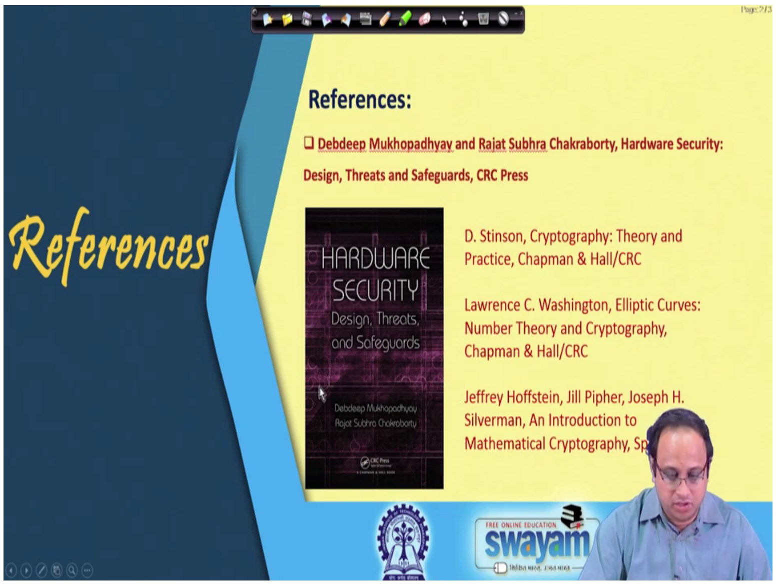
- Power Trace of a round of AES.
- Observe the variation of power values.
- The variations occur because of the operation dependence of power: leads to SPA.
- The variations also occur because of data dependence of power: leads to DPA.

So, you can see that when you are observing the power trace of AES there are certain information which you can easily see like all of us studied right till now that AES has AES 120 has got 10 rounds and you see that here is a power profile which we have captured in our laboratory setup and you can easily read out the 10 round you can see that there is a distinct feature of the rounds.

Likewise if you zoom in into one of the rounds you can also see the internal power consumption for example, you can see that the beginning part is kind of distinct from the later on part and typically what we have understood in the by now is that, this probably relates move to the x boxes and the you know like the and the following step.

So, therefore, right I mean although you cannot read out the keep or say right from this profiles, but definitely right this profiles leak some information ok. So, you can see that there are variations for example, the variations occur because of the operation dependence of power and likewise variations also occur because of data dependence of power ok. So, the message here is that the operation dependence of power leads to simple power attacks whereas, the data dependence of power leads to differential power attacks.

(Refer Slide Time: 31:53)



References

References:

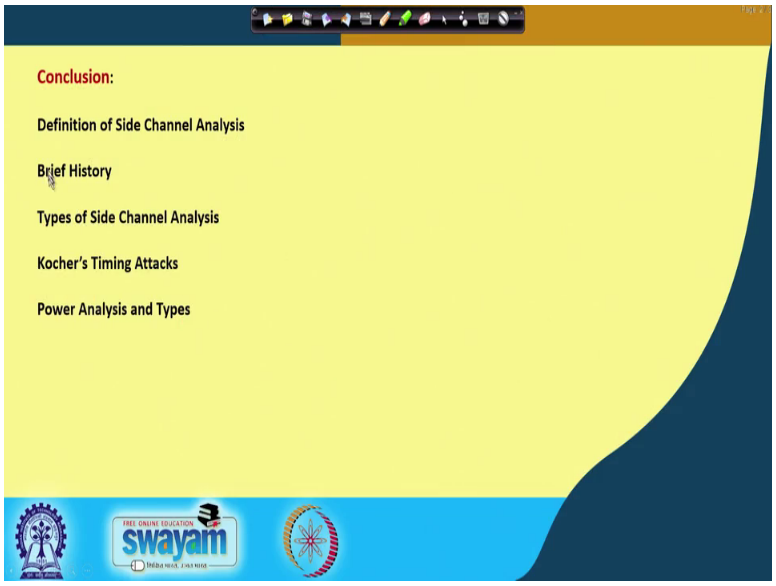
❑ Debdeep Mukhopadhyay and Rajat Subhra Chakraborty, Hardware Security: Design, Threats and Safeguards, CRC Press

D. Stinson, Cryptography: Theory and Practice, Chapman & Hall/CRC

Lawrence C. Washington, Elliptic Curves: Number Theory and Cryptography, Chapman & Hall/CRC

Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman, An Introduction to Mathematical Cryptography, Sp

(Refer Slide Time: 31:57)


Conclusion:

Definition of Side Channel Analysis

Brief History

Types of Side Channel Analysis

Kocher's Timing Attacks

Power Analysis and Types

So, here as some references that we have followed and just to conclude definitions of what we have discussed is definitions of side channel analysis your discussed about brief history of side channel attacks. We have discussed several types of side channel analysis in particular we have studied Kocher's timing attacks, it is interesting and fun to actually implement Kocher's timing attack and see how it works. And we have also discussed about the types of power analysis in today's discussion.

So, with this would like to thank you for your attention.