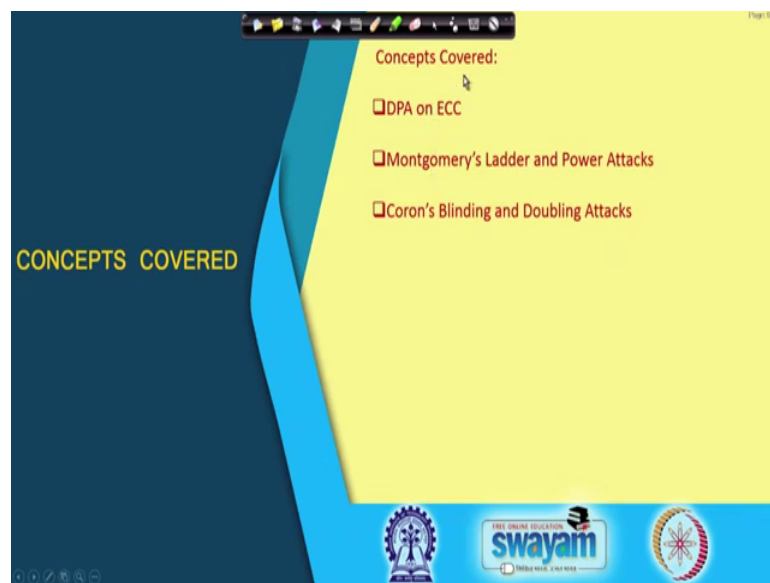


Hardware Security
Prof. Debdeep Mukhopadhyay
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 37
Power Analysis – XIII

So, welcome to this next lecture on power attacks. So, we shall be continuing our discussions on power attacks on elliptic curve hardware. So, in the last class we were discussing about power attacks or simple power attacks on elliptic curves. In today's we shall be discussing that those designs are still all remains against DPA and therefore, we shall try to see how we can protect them and we should be considering some more improvements in the attack.

(Refer Slide Time: 00:39)



So, to start with right I mean essentially today, we shall be covering this concept. So, we shall be looking into DPA on E C C, we shall be talking about a specific way to implement elliptic curve hardware, which is called as Montgomery's ladder and we shall be trying to look at it in the context of power attacks.

We shall be trying to look at it interesting technique of protecting in this D P A, which is called as Coron's blinding technique and we shall be also reflecting on an attack, which is called as doubling attacks, which is relevant on the Montgomery ladder and also maybe relevant on the Coron's blinding technique.

(Refer Slide Time: 01:07)

Montgomery Ladder

Algorithm 2.4: The Montgomery ladder for elliptic curve scalar multiplication.

Input: An integer $d \geq 1$ and a point P on elliptic curve. $d = \sum_{i=0}^{k-1} 2^i d_i$.

Output: dP .

$Q_1 \leftarrow P$ and $Q_2 \leftarrow 2P$.

for i from $k-2$ down to 0 do

 if $d_i = 1$ then

$Q_1 \leftarrow Q_1 + Q_2$ and $Q_2 \leftarrow 2Q_2$;

 end

 else

$Q_2 \leftarrow Q_1 + Q_2$ and $Q_1 \leftarrow 2Q_1$;

 end

end

return Q_1 .

$k=9=(1001)$

$Q_1=P, Q_2=2P$

$k_2=0 \Rightarrow Q_2=3P,$

$Q_1=2P$

$k_1=0 \Rightarrow Q_2=5P,$

$Q_1=4P$

$k_0=1 \Rightarrow Q_1=9P,$

$Q_2=8P.$

Result is in Q_1 .

So, let us see, you know like this is essentially this; what is called as a Montgomery ladder. So, in Montgomery's ladder the idea is that you again this is an type of algorithm, where you do both doubling and addition and the interesting thing is that I mean there are essentially right.

You can as we, you know like you can actually implement it in a very efficient manner ok. You can essentially like if you are using projective coordinates for elliptic hardware, then you can we can as well do this without the y coordinates.

Therefore, you can actually remove one of the coordinates and you can still do it you can still implement it ok, but today we shall be discussing the at this at more from the context of power attacks, but first let us have a quick look to understand that we really understand how the algorithm works. So, therefore this is your scalar that is d is your scalar which is d is equal to this a binary.

So, as we have done right we have basically taken d and we have written it in the binary format. So, therefore, suppose $d_{k-1} d_{k-2} \dots d_0$ is your scalar that you have and d_{k-1} is 1 though therefore, we are assuming that d_{k-1} is 1. So, we will be starting the processing from $k-2$. So, what we do is basically initialize Q_1 . So, now we have got two registers Q_1 and Q_2 we initialize Q_1 to P and Q_2 to $2P$ and the idea is that if d_i is equal to 1; then we are basically adding in Q_1 and we are doing doubling in Q_2 .

Whereas, if you are doing if you know like your d_i is 0 then you can you have basically adding in Q_2 and you are doubling in Q_1 . So, this form of representation is very efficient, because you know you can actually implement this loop also in a slightly different way where you can remove this, if statement you can actually write this in a way such that the suffix of the registers Q_1 and Q_2 .

You can use it as the key bit so; that means, you can write something like Q_b where b stands for the secret key bit. So, this I am not elaborating at this point and I kind of leave it as an exercise to think how we can write this Montgomery's ladder without the if statement.

So, but rather what we will we looking at in today's class is that if you observe that the reason, why the S P A was ordinarily working is, because there was a conditional statement where you are doing an conditional you know where you are doing the addition depends upon the secret bit, but here you can observe that irrespective of whether the key bit is 1 or whether the key bit is 0.

You are always doing an addition operation, but in one case you are doing addition in Q_1 in the other case you are doing addition in Q_2 , but you are always doing an addition operation and therefore, you do not get that clear power signature to distinguish between and understand what is the secret key bit, it works correctly. So, let us take an example suppose your key is 91001 so; that means, we will processing from this 0. So, our initialization is Q_1 is equal to P and Q_2 is equal to $2P$ and then since k_2 is equal to 0; that means, is your 0.

The secret scalar bit is 0 then what I do is, I basically if you see this branch, this one. So, therefore, I come to this branch and therefore, I add in Q_2 . So, I add in Q_2 , which means that Q_2 becomes P plus $2P$ that is $3P$ and Q_1 becomes the double of Q_1 . So, Q_1 becomes $2P$ then I get again 0, the next scalar is 0.

So, therefore, I do essentially the same operation; that means, I essentially add in Q_2 . So, Q_2 becomes $5P$ and Q_1 becomes $4P$ then I get a 1. So, now, what I will do is I will add in Q_1 . So, therefore, Q_1 becomes $5P$ plus $4P$, which is $9P$ and Q_2 is $8P$, but this, we can use for if the, if the, if the scalar was longer, but then the result that we return is essentially Q_1 , because Q_1 is $9P$ that you want to compute.

So, therefore, functionality wise this is correct and also it gives us an enhanced protection against simple power attacks, because there is no conditional statement and therefore, this seems to be more secure.

(Refer Slide Time: 05:23)

DPA Recapitulation

Introduced by P. Kocher and colleagues

More powerful and more difficult to prevent than SPA

Different power consumption for different state (0 or 1)

Data collection phase and data analysis phase

Procedure

- Gather many power consumption curves
- Assume a key value
- Divide data into two groups (0 and 1 for chosen bit)
- Calculate mean value curve of each group
- Correct key assumption → not negligible difference

Handwritten notes and diagrams:

- $R = 2R$
- $\rightarrow \text{if } (d_i = 1) \left. \begin{array}{l} R = R + P \end{array} \right\}$
- Target → [Diagram of a box labeled 'Target'] → Hypothesis
- 0-bin 1-bin [Diagram of two boxes labeled '0-bin' and '1-bin']

So, now let us see you know like. So, basically now let us look at it, at the with the context of DPA ok. So, now, again let us recapitulate DPA little bit. So, as we have already discuss the DPA is essentially a more fundamental form of this of attack, because here we try to kind of exploit the fact that the power consumed is actually dependent upon data.

So, what we did right essentially is kind of shown here in kind of informally as a procedure. So, we basically gather many power consumption curves as oppose to simple power attacks, we do on few power observations, we assume a secret ok. So, the secret could be in the case of block ciphers and n bit value or in the case of a stream cipher could be a single bit. In the case of DPA also we will do something like what we are doing for stream ciphers that means we will do the attack in iterative fashion.

So, we will assume that we know the till i bits and then we will be trying to guess the i plus 1 th bit. So, therefore, the guess is either 0 or 1 and we will basically trying to use this DPA or distinguisher to understand whether my guess is correct or not.

So, basically what I do, if I guess this 0 and guess this 0 this the key bit to be either 0 or 1 and then we basically have a target register or target computation. So, now, the thing is that if my guess was correct then my target is correctly computed and therefore, right I will, I if correlate it if I, if I extract out a hypothetical power from that target register and I correlate with my real power trace then I should get a very maxima I should get a maximal correlation.

On the other hand, if my guess was wrong then that correlation will be much lesser and therefore, I can distinguish and understand whether my key guess was correct or whether my key guess was wrong. So, what I mean is essentially if I just take the, you know simpler scalar multiply algorithm, I mean or the double and add algorithm.

So, what we are doing in the normal double and add algorithm is that you essentially are doing a double operation. So, therefore, if you have got a point P you are doing at $2P$ operation always. If your scalar is 1; that means, if your d_i is equal to equal to 1 then you are basically adding right. So, basically you are doing. So, this suppose, I am you know like having a register and I am writing R equal to $2R$, if this key bit is 1 then I am writing R equal to R plus P ok. So, basically I am trying to calculate λP or maybe dP in this way.

Right, I am trying to calculate dP in this fashion. So, therefore, right it means that I have got the scalar d and I want to kind of add P d times. So, therefore, right here what I will do is that, I will now guess this d_i and I will target this register R . This the target, this register R is constantly being updated, if my key bit was in the i th state was 0 then I would have got $2R$.

But now, instead of that I am getting R plus P ; that means, this $2R$ is happening and then I am adding P to it. So, the idea is that if my guess is 0 then I will be basically you know like trying to kind of estimate what would be the value of R and from there so therefore, my target register is my target. In this case is the register R and from there I will be trying to predict, what is my hypothetical power value.

So, in case of CPA you basically find out the hypothetical power value in case of DOM you probably target one of the LSBs of this R suppose this LSB is 0 or the LSB is 1. You take all the power traces, you basically make a 0 bin, you make a 1 bin and then you basically, you know kind of take the traces and put them into the 0 bins and the 1

bins. You take the average, take the difference of means, the idea is that if your guess was correct then difference of mean is quite high; that means, there is a high correlation.

In case of CPA, you basically take the entire value of R, we predict the hypothetical power value by some kind of power model and then you try to correlate that power value with your real power trace and the idea is that you employ the calculate the Pearson's correlation and you find out that the Pearson's correlation should be maximized at some time instance, which is your point of interest. So, this is how you can adopt DPA to even elliptic curves and essentially, the adoc should fundamentally work.

(Refer Slide Time: 09:39)

DPA of Montgomery Ladder

Algorithm 2.4: The Montgomery ladder for elliptic curve scalar multiplication.
Input: An integer $d \geq 1$ and a point P on elliptic curve. $d = \sum_{i=0}^{k-1} 2^i d_i$.
Output: dP .
 $Q_1 \leftarrow P$ and $Q_2 \leftarrow 2P$.
 for i from $k-2$ down to 0 do
 if $d_i = 1$ then
 $Q_1 \leftarrow Q_1 + Q_2$ and $Q_2 \leftarrow 2Q_2$;
 end
 else
 $Q_2 \leftarrow Q_1 + Q_2$ and $Q_1 \leftarrow 2Q_1$;
 end
 end
 return Q_1 .

Table 4.11: Computed points in Algo. 2.4 for first three bits of d .

$d_{i-3} = 0$		$d_{i-3} = 1$	
$Q_2 = 3P, Q_1 = 2P$	$Q_2 = 6P, Q_1 = 5P$	$Q_2 = 7P, Q_1 = 6P$	$Q_2 = 8P, Q_1 = 7P$
$d_{i-4} = 0$	$d_{i-4} = 1$	$d_{i-4} = 0$	$d_{i-4} = 1$
$Q_2 = 5P, Q_1 = 4P$	$Q_2 = 11P, Q_1 = 12P$	$Q_2 = 13P, Q_1 = 14P$	$Q_2 = 15P, Q_1 = 16P$
$Q_2 = 9P, Q_1 = 10P$	$Q_2 = 17P, Q_1 = 18P$	$Q_2 = 19P, Q_1 = 20P$	$Q_2 = 21P, Q_1 = 22P$
$Q_2 = 8P, Q_1 = 9P$	$Q_2 = 10P, Q_1 = 11P$	$Q_2 = 12P, Q_1 = 13P$	$Q_2 = 14P, Q_1 = 15P$
$d_{i-3} = 1$		$d_{i-3} = 0$	
$Q_2 = 4P, Q_1 = 3P$	$Q_2 = 8P, Q_1 = 7P$	$Q_2 = 7P, Q_1 = 6P$	$Q_2 = 6P, Q_1 = 5P$
$d_{i-4} = 0$	$d_{i-4} = 1$	$d_{i-4} = 0$	$d_{i-4} = 1$
$Q_2 = 13P, Q_1 = 14P$	$Q_2 = 15P, Q_1 = 16P$	$Q_2 = 17P, Q_1 = 18P$	$Q_2 = 19P, Q_1 = 20P$
$Q_2 = 12P, Q_1 = 13P$	$Q_2 = 14P, Q_1 = 15P$	$Q_2 = 16P, Q_1 = 17P$	$Q_2 = 18P, Q_1 = 19P$

Handwritten annotations: $d_{k-2} \rightarrow 0$, $d_{k-2} \rightarrow 1$, $2P \rightarrow 0$, $4P \rightarrow 1$.

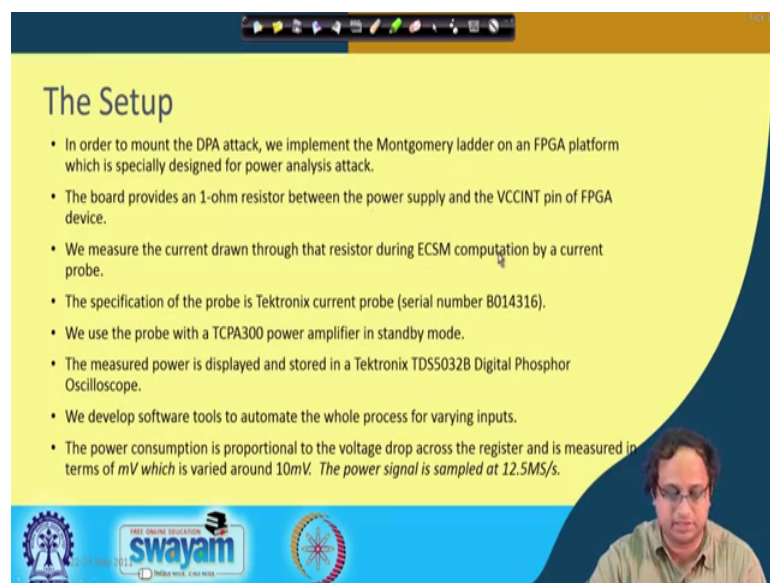
So, let us see how it will work. So, what we do is that if I want to do an DPA on the Montgomery ladder for example, then essentially I again look back at the Montgomery structure and I try to kind of find out how the you know like the Montgomery evolves the or the Montgomery ladder evolves depending upon my key bits.

So, basically I start with my key d_{k-2} , because that is the first key that I need to find out. So, this key can, this key bit can be either 0 can be either 1, if I am not given any side channel error trace, then I have got no clue to understand whether it is a 0 or whether it is a 1, because the algorithm is nicely shielding this fact in a mathematical fashion, but then what I observed is that if d_{k-2} is equal to 0 then there are two registers like Q_2 and Q_1 . Q_1 for example, you can see is equal to $2P$.

On the other hand, if this bit is 1 then Q_2 , you see then we can see that this $2P$ right, never occurs like you will see that if this is 1 then Q_2 is $4P$ and Q_1 is $3P$. So, therefore, right, if you observe then rather than you know like targeting or correlating with the value of $3P$, it would be an intelligent right or it should be more say and to basically correlate it with the value of $2P$, because if the value, if the key bit is 0 then $2P$ occurs in one case whereas, $2P$ does not occur if the key bit is 1. You can see $3P$ occurs in both cases, this is $3P$ and this is a $3P$.

So, rather what I will try to do therefore, is that I will be trying to kind of correlate it with $2P$, if it correlates with $2P$ then the idea would be that that probably the guess is 0. I will return 0 as my guess, if it correlates with $4P$ then I will return 1 as my guess. So, I can also confirm whether my guess is correct in this way, both of them you know like should confirm ideally. So, this is a simple strategy that you can take and you can try to kind of adopt your differential power attacks for elliptic curves even if that is a Montgomery ladder implementation underlined.

(Refer Slide Time: 11:57)



The Setup

- In order to mount the DPA attack, we implement the Montgomery ladder on an FPGA platform which is specially designed for power analysis attack.
- The board provides an 1-ohm resistor between the power supply and the VCCINT pin of FPGA device.
- We measure the current drawn through that resistor during ECSM computation by a current probe.
- The specification of the probe is Tektronix current probe (serial number B014316).
- We use the probe with a TCPA300 power amplifier in standby mode.
- The measured power is displayed and stored in a Tektronix TDS5032B Digital Phosphor Oscilloscope.
- We develop software tools to automate the whole process for varying inputs.
- The power consumption is proportional to the voltage drop across the register and is measured in terms of mV which is varied around $10mV$. The power signal is sampled at $12.5MS/s$.

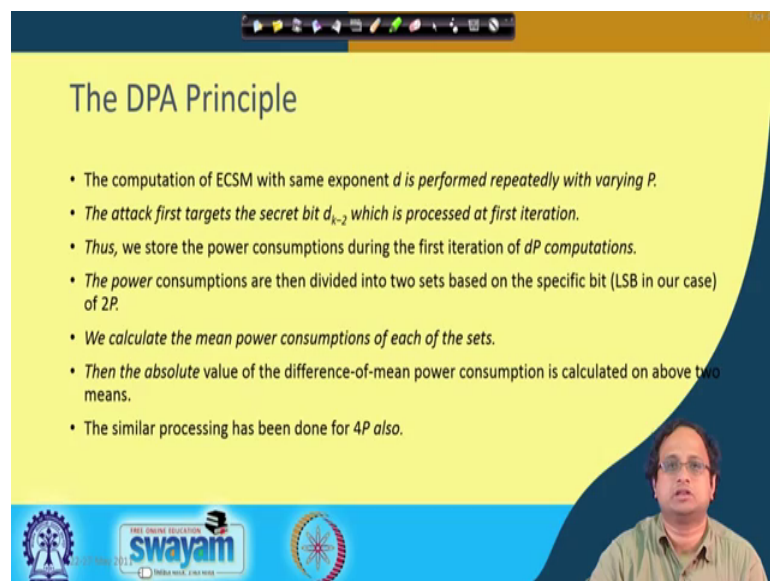
So, so, therefore, right I mean with this we let us try to see again the set up. So, in this case again we try to do a DPA with a real with real power traces. So, in order to , mount the DPA attack you implement the Montgomery ladder on an F P G A platform, which is specifically designed for power analysis attacks just like the one that we saw in some of the previous classes.

So, the board basically provides a 1 ohm resistor between the power supply and the V C C and pin of the F P G A. So, we measure the current which is drawn through that resistor during the elliptic curve scalar multiplication operation by a current probe. So, we use as we discussed current probes of proper specifications.

So, there is a high S N R of your power traces ok. So, we know what is an S N R. So, the we would like that the power traces high signal to noise ratio and then right we basically, try to measure the power in a Tektronix oscilloscope or for that matter any oscilloscope which has got high bandwidth. So, this is the oscilloscope that we use in our setup and then we develop software tools to automate the whole process for varying inputs.

Now, the power consumption is proportional to the voltage drop across the register and is measured in terms of millivolts, which is varied around 10 millivolts and the power signal is also sample at a high sampling rate of something like 12.5 mega samples per seconds, in our current set up, we essentially sample it at even a higher sampling rate.

(Refer Slide Time: 13:21)



The DPA Principle

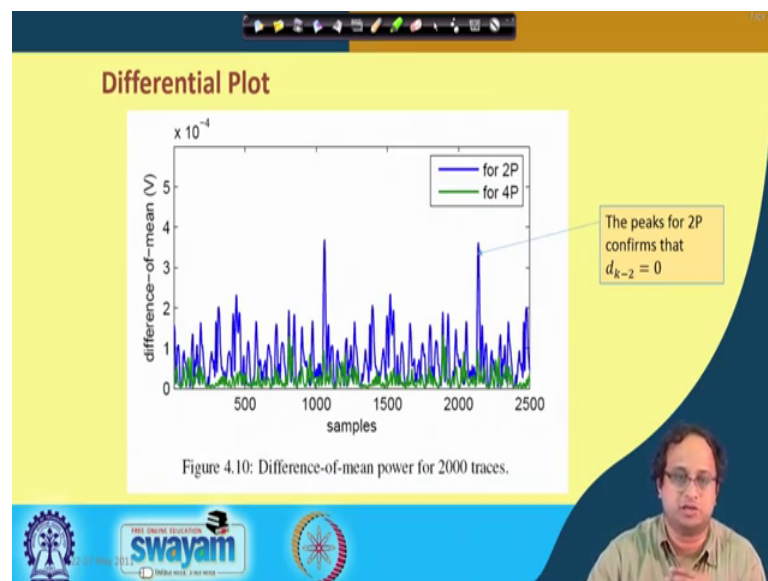
- The computation of ECSM with same exponent d is performed repeatedly with varying P .
- The attack first targets the secret bit d_{i-2} which is processed at first iteration.
- Thus, we store the power consumptions during the first iteration of dP computations.
- The power consumptions are then divided into two sets based on the specific bit (LSB in our case) of $2P$.
- We calculate the mean power consumptions of each of the sets.
- Then the absolute value of the difference-of-mean power consumption is calculated on above two means.
- The similar processing has been done for $4P$ also.

The slide features a yellow background with a blue wave-like shape on the right side. At the bottom, there are logos for 'swayam' and other educational institutions, along with a small video inset of a man speaking.

So, therefore, right the ones we have that then we apply our DPA principle. So, DPA principle is as we have discussed that with the same exponent d , you again you perform repeated operations with different P values ok, because the scalar is which you want to basically obtain is kept constant and you are basically, giving different P values and you are computing different times you are computing dP basically like multiplying P with d .

So, the attack first targets the secret bit d_{k-2} , which is processed at first iteration and thus, we store the power consumptions during the first iteration of d_P computations. Now, the power consumptions are then divided into two sets based on the specific bit or L S B in our case of 2_P , because we are trying to correlate with the value 2_P ok. We calculate the mean power consumption of each of the sets and then the absolute value of the difference of mean power consumption is calculated, as a difference of mean and the similar processing also is done for 4_P .

(Refer Slide Time: 14:23)



So, here we observe an result. So, we basically show the differential plot when you are trying to correlate with both 2_P as 4_P . So, you can see that we obtain a correlation with 2_P . So, 2_P is the color which is shown in blue and you get a correlation high correlation that implies that we will be returning 0 and that turns out with the correct key. So, therefore, on the other hand right, there is no correlation with 4_P , which means like that a key bit is not 1 ok. So, basically you can do both the tests and can confirm your guess, you can confirm your estimation.

(Refer Slide Time: 14:55)

Counter-measure

- **Coron's Point blinding method:**
 - Choose a random secret value R and compute, $S=dR$.
 - Blind the point P, by computing $P+R$
 - Evaluate $d(P+R)$ and subtract S.
 - The computations are not correlated to the actual point P, and does not leak the k under conventional DPA.
 - The random points R and S are refreshed at the starting of a new execution by the relation $R' = (-1)^b 2R, S' = (-1)^b 2S$.

$dP = d(P+R) - S$

$S' = dR', dP = d(P+R) - S'$

$b \in_R \{0,1\}$

\downarrow

So, therefore, right I mean we understand that the Montgomery's ladder is vulnerable against differential power attacks, although it is resistant against simple power attacks, but there is a simple technique or there is a very nice technique, which we can apply, which is called as Coron's point blinding method. So, here what you do is you choose a random secret value R and compute S equal to d R ok. So, note that what I just do is I calculate, I basically choose a secret value which is R so, designed internally does that.

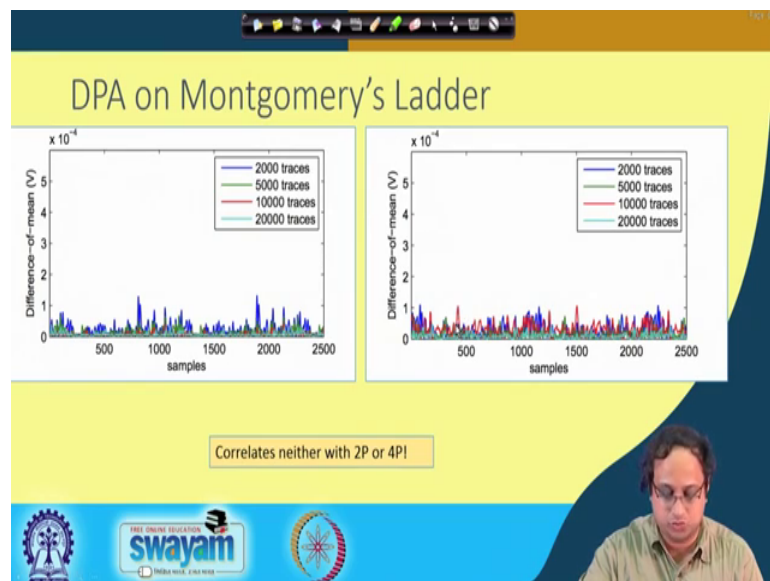
Every time you are doing an encryption ideally, you should use a different R and then you basically calculate S equal to d R and then what you do is basically, you blind the point P by computing P plus R, because P plus R essentially makes it kind of, you know you know independent of P, because you are basically choosing a random point and you are blinding that point and then you multiply d into P P plus R that is you calculate d into P plus R and you subtract out S.

So, you can easily see that this is nothing, but equal to d P, because d into R minus S, because of this equation gets cancelled out and therefore, you basically are calculator in d P, but at every step you are basically, doing this computation or you are doing this computation both of them are essentially right, ensuring that you have a random component which is coming into play. And the idea is that you basically, would like to ensure that as I say that every time you are doing repetition of this, you should have a fresh random values ok.

So, therefore, one trick that essentially you can do is you can use this equation to update your blinders or masks. So, what I do is basically, I take a coin like a like a (Refer Time: 16:36) essentially stands is basically randomly sampled from 0 or 1 and then I basically do this operation that is minus 1 to the power of b into 2 R and S dash minus 1 to the power of b into 2 S. So, you can note that even with R dash S dash, this relationship that is S dash is equal to d of R dash is maintained right and therefore, right you can use these two blinders also in the next step, where you basically calculate maybe d into P 1, where your you know like trying to calculate d P 1.

So, d P 1 you can calculate as d into P 1 plus R dash and subtract out S dash from this right. So, therefore, right the idea is that every time you should do this operation then this would be naturally resistant against a differential power attack, because every time, because of this randomization you are ensuring that you cannot, the attacker cannot pick up this correlations. So, therefore, right I mean this is quite nice and we basically right can probably expect that this will be more resistant against power attacks.

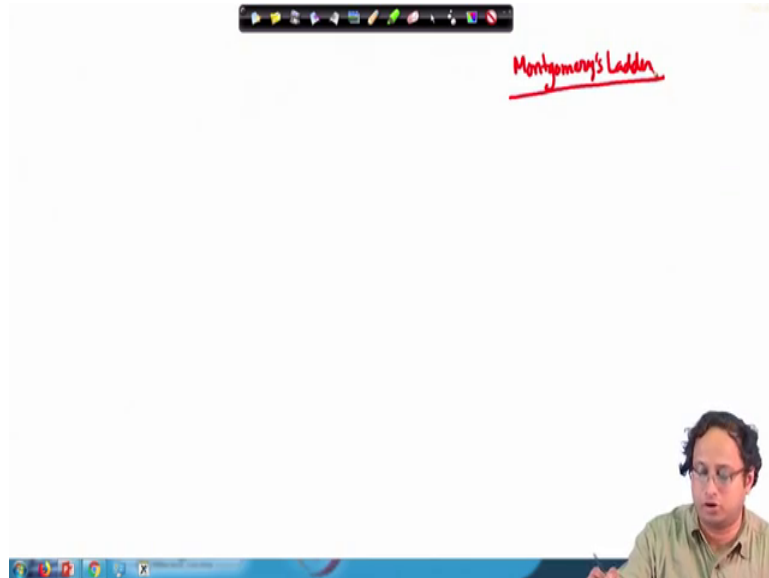
(Refer Slide Time: 17:35)



So, we can see for example, that in the same case when we adopt the Coronas technique then there is you know like this, there the correlations that we were getting to distinguish between the key bit being 1 or the key bit being 0 are now gone ok. We then when quite large number of traces like around 20000, there is no correlation with 2 P or 4 P and therefore, Coronas blinding technique works quite nicely.

So, there is another class of attack, which I would like to consider here and essentially, that essentially is what is called as is called as doubling attacks ok. So, we will try to see how doubling works or doubling attack works.

(Refer Slide Time: 18:19)



So, just remember that in a Montgomery's ladder, we like when you essentially have the original Montgomery's ladder so, and this is more you know like useful for Montgomery's ladder is that.

(Refer Slide Time: 18:31)

	M	2M	
→ 1			1: $\beta_1 \leftarrow \beta_1 + \beta_1, \beta_2 = 2\beta_2$
→ 1	$\beta_1 = M + 2M$ $\beta_2 = 2(2M)$	$\beta_1 = 2M + 4M$ $\beta_2 = 2(4M)$	0: $\beta_2 \leftarrow \beta_1 + \beta_2, \beta_1 = 2\beta_1$
→ 0	$\beta_2 = 3M + 4M$ $\beta_1 = 2(3M)$	$\beta_2 = 6M + 9M$ $\beta_1 = 2(6M)$	
0	$\beta_2 = 13M$ $\beta_1 = 12M$	$\beta_2 = 12M + 14M$ $\beta_1 = 2(12M)$	
1	$\beta_1 = 12M + 13M$ $\beta_2 = 2(13M)$	$\beta_1 = 24M + 26M$ $\beta_2 = 2(26M)$	
0	$\beta_2 = 25M + 26M$ $\beta_1 = 2(25M)$	$\beta_2 = 50M + 52M$ $\beta_1 = 2(50M)$	
1	$\beta_1 = 50M + 51M$ $\beta_2 = 2(51M)$	$\beta_1 = 100M + 102M$ $\beta_2 = 2(102M)$	
1	$\beta_1 = 101M + 102M$ $\beta_2 = 2(102M)$	$\beta_1 = 202M + 204M$ $\beta_2 = 2(204M)$	

If $d_{i-1} = d_i$, then
 same doubling operation
 is executed in $(i-1)^{th}$
 iteration of $d(2M)$ and
 in operation of $d(M)$.

$R + P \rightarrow R' + 2P$
 $R' = (-1)^b 2R$
 with probs $\frac{1}{2}$, $R' = 2R$
 $2R + 2P = 2(R+P)$

If your key bit is 1, you basically add in Q 1. So, Q 1 is equal to Q 1 plus Q 2 and you double in Q 2. So, you do Q 2 equal to 2 into Q 2, if the key bit is 0 then you add in Q 2. So, Q 2 is equal to Q 1 plus Q 2 and you doubling Q 1.

So, now let us try to see and see how the evolution works, when there are two inputs specifically, which were processing in the Montgomery's ladder ok. In so, in one case I am processing m that is P equal to M in the other case I am processing P equal to 2 M and the scalar that I considered is 1 1 ok. So, 1 1 followed by 0 0 1 0 1 and 1. So, therefore, this is my corresponding scalar which I process. So, in the first case I say that this is one. So, I will leave this, I will start from this one ok. So, because this is my leading 1. So, I can just leave it right this point.

So, therefore, Q 1 here is equal to M plus 2 M and Q 2 is equal to 2 into 2 M likewise right. In the next case I get a 0 here. So, if it is so, so, if it is a 0 then I will be adding in Q 2. So, you can see that I add in Q 2. So, Q 2 is equal to so in this case you have got 3 M and you have got 4 M. So, I just get 3 M plus 4 M and Q 1 is equal to the double of 3 M, because this gets doubled. So, I get 2 into 3 M and then I have got 0.

So, again I have got 0, which means I will again be adding in Q 2. So, this turns out to be 13 M and Q 1 is equal to 12 M, because I am again doing double of 2 into 3, which is 6 M. Now, if this is 1, the next bit is 1. So, I get Q 1 as 12 M plus 13 M. So, I am adding in Q 1 and I am doing doubled in Q 2. So, this becomes 2 into 13 M again I get 0. So, therefore, I add in Q 2. So, Q 2 becomes 25 M plus 26 M and Q 1 is equal to 2 into 25 M right then you have got 1. So, you have got Q 1, which is equal to 50 M plus 51 M and Q 2 is equal 2 into 51 M and next again you get a 1.

So, basically you do Q 1 equal to 101 M plus 102 M and Q 2 is equal to 2 into 102 M. So, therefore right you can verify that Q 1 is equal to 203 M, which is what we want to calculate. So, this scalar should stand for that corresponding result. Now, the interesting thing is that let us see the evolution what happens is when you are processing 2 M basically. So, again we will so a similar thing.

So, we will start here with Q 1. So, Q 1 is 2 M plus 4 M and Q 2 is equal to 2 into 4 M ok. So, that is my first computation, which is being done. In the second case, I get a 0 here. So, therefore, I do Q 2 which is equal to 6 M plus 8 M right 6 M plus 8 M 4 M is 2 M plus 4 M is 6 M and then this 8 M.

So, I add in Q 2, because it is 0 and Q 1 is equal to the double of 2 into 6 M, which is 12 M ok. So, that again is my this processing then I get again a 0. So, therefore, I can add in Q 2. So, Q 2 equal to 12 M plus 14 M and Q 1 is equal to 2 into 12 M right. So, that ends my computation for this stage then I get again a 1. So, because of this 1, I will add in Q 1. So, Q 1 becomes equal to 24 M plus 26 M and Q 2 is equal to 2 into 26 M Q; Q 2 gets doubled here and that is my end this evolution.

So, then I get a 0. So, Q 2 is equal to now 50 M plus 52 M and Q 1 is equal to 2 into 50 M ok. So, that ends this evolution and then you have got 1. So, because of this 1 you will have Q 1, which is equal 100 M plus 102 M and Q 2 is equal to 2 into 102 M and that is obtained at this point and then finally, right because of this 1 you have got Q 1 which is equal to 202 M plus 204 M and Q 2 is equal to 2 into 204 M and that essentially means this is your result and that essentially is nothing, but the result which you want ok, which is the double of 203 M ok.

Now, interestingly you observe the; if the two successive key bits are 0 and if you target the register Q 1, you will see that both of them compute the same value ok. Like in this case it is 12 M, this is also 12 M here, also if these are 2, these are this 2 are ones and if you target this Q 2; that means, whenever there is a doubling operation, you will see that the doubling right essentially has got the same result so; that means, right.

We can write from here is this fact that if d^{i-1} is equal to d^i then same doubling operations then same doubling operation is executed is executed in the $i-1$ iteration of d into of d into $2^i M$ and the i th operation of d into M ok.

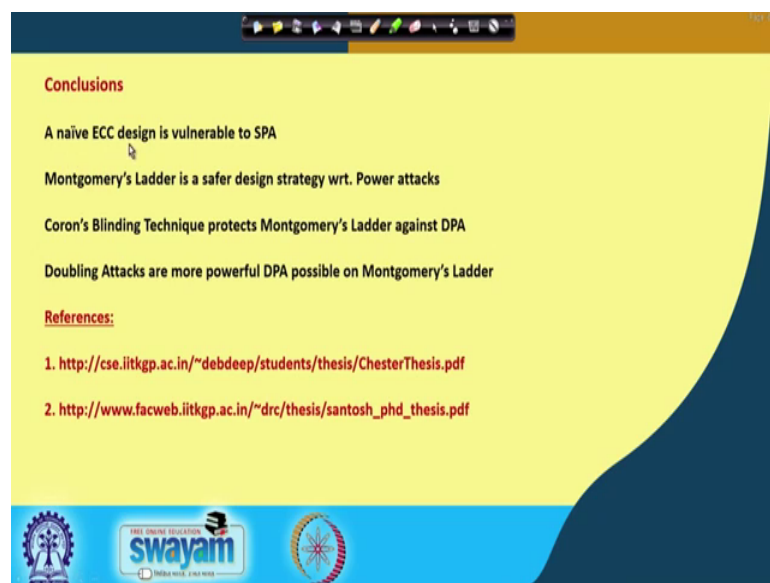
So, this is an attack which you can still try on the Montgomery's ladder. Now, interestingly observe that the way we were upgrading the mask in we are using Coronas technique is that we had this $R + P$. So, this we were modifying to $R - P$ plus $2P$. So, imagine that if you have got I do around with P and then I do around with $2P$ then this is what will happen right.

I will I will add $R + P$ and here I will add some $R - P$ plus $2P$, but the relation between $R - P$ and R is given by this relation $(-1)^{b-1} R$. So, with a probability of 0.5 with a probability of 0.5 you still have that $R - P$ equal to $2R$ and; that means, you are basically processing 2 into R plus 2 into P , which is 2 into R

plus P. So, here you are processing with R plus P with a probability of 0.5 you are, you are processing 2 into R plus P.

So, therefore, right you can still try to fancy attack even when you have Coronas blinding protection and therefore, you probably need to do something better to update your blinding techniques ok. So, we will basically stop here and, we basically you know like we will just conclude what we discussed and. So, basically the; what we discussed is that the naïve E C C design is vulnerable to S P A.

(Refer Slide Time: 26:39)



Conclusions

- A naïve ECC design is vulnerable to SPA
- Montgomery's Ladder is a safer design strategy wrt. Power attacks
- Coron's Blinding Technique protects Montgomery's Ladder against DPA
- Doubling Attacks are more powerful DPA possible on Montgomery's Ladder

References:

1. <http://cse.iitkgp.ac.in/~debdeep/students/thesis/ChesterThesis.pdf>
2. http://www.facweb.iitkgp.ac.in/~drc/thesis/santosh_phd_thesis.pdf

The footer of the slide features three logos: the Swamyam logo with the tagline 'THE ONLINE SOLUTION', and two circular institutional logos.

The Montgomery's ladder is a safer design strategy with respect to power attacks, because it prevents against simple power attacks. Coronas blinding technique is an interesting way to protect Montgomery's ladder against DPA and doubling attacks are more powerful D P A, which are possible on Montgomery's ladder and even right we still can fancy our chances on even if there is a coronas blinding protection, which is updated in the fashion that we described.

So, the two references for this discussion are you know like the M S thesis of my former M S student. So, we can see this the URL for that and the PhD thesis for my one of my former PhD students ok. So, you can go and see this references for more elaborations on this topics ok.

Thank you for your attention.