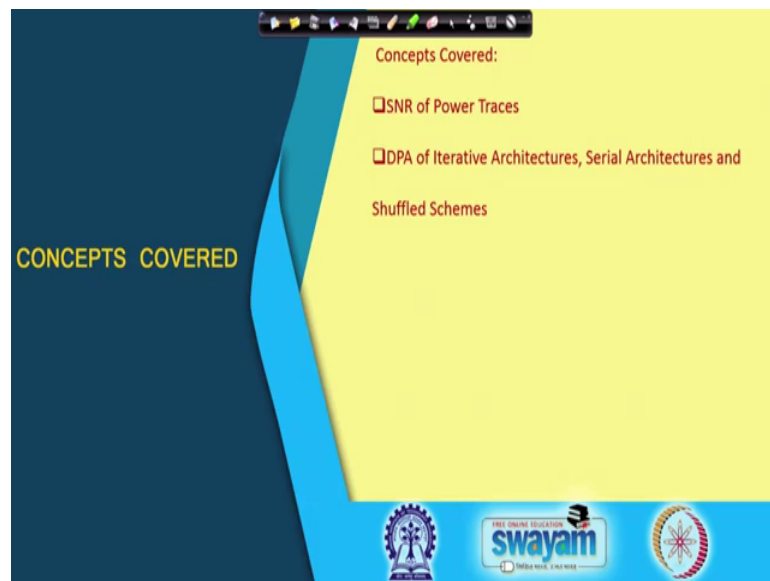


**Hardware Security**  
**Prof. Debdeep Mukhopadhyay**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 33**  
**Power Analysis – IX**

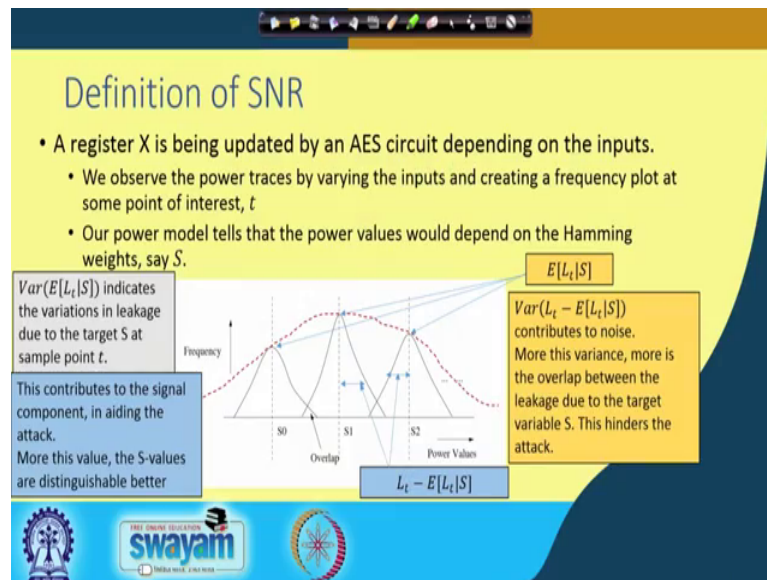
So, welcome back. So, we shall be continuing our discussions on power attack.

(Refer Slide Time: 00:20)



In particular, we shall be discussing about Signal to Noise Ratio or SNR of power traces and we shall be using it as a guideline to compare between a different architectures as we will see in this today's class.

(Refer Slide Time: 00:29)



So, to start with, as we you know as we have been discussing that in the context of AES. So, again I will be taking the example of an AES hardware and try to explain what is meant by SNR or signal to noise ratio of a power trace. So, consider that there is a register  $x$  which is being updated by an AES circuit depending on the inputs and we observe the corresponding power traces by varying the inputs and creating a histogram or a frequency point or a frequency plot at some point of interest it is a  $t$ .

So, therefore, right I mean as we know that, there could be different values that the state register takes ok. For example, if you are modelling by say some hamming weight classes, then it can take several hamming values. So, our power model tells us that the power values would depend on the hamming weight say  $S$ . So, here is a kind of a pictographic representation of the normal distribution and if you observe carefully, what we have kind of observe in  $x$  axis is the power value. So, in the  $y$  axis is the frequency as the a normal frequency plot. But you see that I have drawn, so, in like some vertical lines to indicate the possible hamming values ok.

Like the hamming value  $S$  for example, can take the value  $S_0$ , can take the value  $S_1$ , can take the value  $S_2$ . So, what is the goal of an attacker? The goal of the attacker right is by using or observing this leakage this is the overall leakage is to be able to distinguish between these hamming classes right. As we have seen that it basically tries to map from the key space to say the hamming class. So, what essentially is here is that if you see for

example, say  $S_1$  right which is probably the most commonly occurring hamming class. So, it could be typically right if you are talking an  $n$  bit class then it is typically the  $n/2$  class which occurs with the maximum frequency.

So, you will find that there is a nice Gaussian distribution across that vertical line. Likewise right, there is a you know like internal normal distribution across all possible hamming classes ok. So, now, we basically try to find out in this particular diagram or in this particular distribution, we try to find out the signal component and the noise component because as we know that the SNR is nothing but the ratio between the signal component to the normal component to the noise component. So, we denote this particular statistic which is variance of  $E L_t$  given  $S$  as the signal component ok.

So, what is  $E L_t$  given  $S$ ? So, you see that this is the leakage,  $L_t$  is essentially nothing but the random variable which is denoting the leakage at a time instance  $t$ . So, therefore, right this leakage can again be conditioned by the corresponding value which these state register takes like the state register can take  $S_0$ , can take  $S_1$ , can take  $S_2$  whatever values. So, therefore, right all the leaked or all the traces which are basically go in say to the  $S_1$  bin ok. So, is a basically if I partition them depending upon you know like what is the hamming classes, then the average value over there is essentially nothing but  $E L_t$  given  $S$ . It is the expectation of the random variable  $L_t$  given  $S$  ok.

And that essentially is the mean value. So, you can probably think is as a hypothetically right this if this is the normal distribution around  $S_1$ , then this essentially stands for  $E L_t$  given  $S_1$  ok, likewise this transfer say  $E L_t$  given  $S_2$ . These are the means of the distributions. So, then what is variance of  $E L_t$  given  $S$ ? So, basically right the variance of  $E L_t$  given  $S$  would kind of indicate how far or how separated these means are from each other. And you can understand that if these means are separated which means if these vertical lines are separated out then they are more distinguishable and that is why they you know they contribute to determining the signal component of your power trace ok.

Likewise, you see that this is the overall leakage right this is the leakage that essentially has been shown here and this is your  $E L_t$  given  $S$  as I have just said. So, if I take that subtract, if I subtract out and get  $L_t$  minus  $E L_t$  given  $S$ , that is not can we stand for this part ok. So, that stands for the fact that if I you know like if I take this parameter  $L_t$

minus  $E[L_t]$  given  $S$  and  $I$  you know like make the variance of these statistic more, then that would imply in a sense that there is more overlap between these frequency plots ok. For example, if I take this frequency plot and if I take this frequency plot, then this essentially stands for  $L_t$  minus  $E[L_t]$  given  $S_1$  and this stands for  $L_t$  minus  $E[L_t]$  given  $S_2$  ok.

And if I make them more spread; that means, if I make this normal distributions more spread, then this overlap region will increase ok. Likewise you can see here if I take this and if I spread it out more; if I spread it out more, then this overlap region will increase. So, there for this variance that is variance of  $L_t$  minus  $E[L_t]$  given  $S$  essentially contributes to noise is essentially a measure of noise ok. More this variance, more is a overlap between the leakage due to the target variable  $S$  and therefore, this will hinder the attack.

(Refer Slide Time: 06:09)

**Definition of SNR**

$$SNR_t = \frac{\text{var}(E[L_t|S])}{\text{var}(L_t - E[L_t|S])}$$

- The SNR is a very useful metric for assessing the threat of a power attack on a given implementation.
- How can it be computed in experiments?
  - Say we collect 10,000 traces.
  - We target a byte, say denoted as  $S$ .
  - Depending on the Hamming Weight (say) we split the traces into 9 bins.
  - At a time  $t$ , thus the  $i^{th}$ -bin consists of power values,  $P_i^0, \dots, P_i^{n_i-1}$ 
    - The average of this gives a point in the distribution  $E[L_t|S]$ . We calculate the variance for the signal.
  - From each of the power values in the  $i^{th}$ -bin we subtract the average of the bin
    - The variance of this gives the noise component.

swamyam

So, therefore, I mean it is very intuitive that now we can actually define SNR as nothing but the ratio of the variance of  $E[L_t]$  given  $S$  to the variance of  $L_t$  minus  $E[L_t]$  given  $S$ . Because this stands for the signal or this measures the signal whereas and the if this part as if the numerator is more, then more are the hamming classes separable ok. And likewise like if this part of the denominator is more, then there is a overlap or there is a increased overlap due to which there is more ambiguity and that is exactly what noise does right. So, therefore, this is a measure of noise. So, the SNR is a very useful metric

for assessing the threat of a power attack on a given implementation and the question is that how can you would evaluate in experiments. And these are very important point to understand because, you know like when you do that experiments you have to estimate the SNR of the power trace.

So, suppose you have got 10000 traces and you target a byte say I denoted as  $S$ , depending on the hamming weight say you know like we split the trace into 9 bins or I say that suppose I am considering on a byte of data ok. So, as I said there are I am targeting a byte and the byte can take 0 to 8 that is 9 possible hamming classes, I basically distribute; I basically distribute these power traces into this 9 bins ok. At a time  $t$  therefore, the  $i$ th bin consists of you know like  $P_{i0}, P_{i1}, \dots, P_{in_i-1}$  assuming that  $n_i$  number of traces are going to the  $i$ th bin and then I take the average of all these values.

If I take the average of these values, I get the I get an estimation of  $E L_t$  given  $S_i$  because that is the  $i$ th possible class. So, we calculate the so, I can write  $E L_t$  given  $S$  equal to  $i$  also. So, we calculate the variance for these signal. So, note that for every hamming class right, I will get corresponding average. And therefore, right I will get if there are 9 classes right, I will get 9 points in the I will I will get 9 points and therefore, I can compute the variance of this distribution I will get a distribution I will get because each mean itself is essentially a point on the on my distribution.

And therefore, I calculate the variance of this distribution and that gives me my variance of  $E L_t$  given  $S$  which is the numerator of my SNR definition. And now I want the that the denominator I want the denominator. So, what I do is, I take each of these bins and then I subtract out the average of that bin ok. So, I average I subtract of the average of the bin and therefore, right I will have some raw noise I will have a measure of the noise in each of the bins ok. So, now, I basically combine I kind of observe these combined distribution because every bin will have some noise values and then I find out the variance of this overall distribution and the that gives me the denominator and I take the ratio to get the SNR.

(Refer Slide Time: 09:18)

Relating Correlation with SNR

- $L_t = P_{det} + N$ , where  $P_{det}$  is the deterministic component of power and  $N$  is the noise.
- Thus the correlation between the total leakage  $L_t$  and the hypothetical power value for the  $i^{th}$  key  $H_i$  as  $Corr(H_i, L_t)$ :
- $Corr(H_i, L_t) = Corr(H_i, P_{det} + N) = \frac{Cov(H_i, (P_{det} + N))}{\sqrt{Var(H_i)(Var(P_{det}) + Var(N))}}$
- Thus,  $Corr(H_i, L_t) = \frac{E(H_i \cdot (P_{det} + N)) - E(H_i)E(P_{det} + N)}{\sqrt{Var(H_i)Var(P_{det})} \sqrt{1 + \frac{Var(N)}{Var(P_{det})}}}$

The slide also features logos for Swamyam and other educational institutions, and a small video feed of the presenter in the bottom right corner.

So, now, you can actually do interesting manipulations once we have defined this SNR. For example, as we have seen that we are calculating the correlation for doing a DPA. Now, we can try to relate this correlation metric with the SNR ok. So, let us see how we can do it. So,  $L_t$  is nothing but  $P_{det}$  as I said this is a deterministic portion, I mean  $P_{det}$  deterministic which as which I was denoting it as a  $S$  for example, these deterministic portion because I can control it by the input as well as the key the component of the key plus  $N$  which  $N$  stands for noise ok.

So, therefore, when we are calculating the correlation so that we are doing is we are calculating the correlation between total leakage  $L_t$  and the hypothetical power value for the  $i^{th}$  key ok. The hypothetical power value for the  $i^{th}$  is denoted as  $H_i$  ok. So, we are calculating the correlation between  $H_i$  and  $L_t$ . So, what is the correlation between  $H_i$  and  $L_t$ ? The correlation between  $H_i$  and  $L_t$  is nothing but the correlation of  $H_i$  comma  $P_{det}$  plus  $N$ .

So, this  $L_t$  is nothing, but  $P_{det}$  plus noise and therefore, right I can write that as nothing but the covariance of  $H_i$  and  $P_{det}$  plus noise divided by the square root of the variance of  $H_i$  and the variance of  $P_{det}$  plus noise ok. So, note that when I am writing this, I am basically making an assumption because I am writing variance of  $P_{det}$  plus noise as variance of  $P_{det}$  plus variance of noise ok.

So, I am assuming that a noise is independent of the deterministic component there I am not assuming, then there is any there is any covariance among them. And then, I can write this as covariance, I mean the covariance of  $H_i$  comma  $L_t$  is therefore, I if I see the numerator; the numerator to nothing but the expectation of  $H_i$  and  $P$  deterministic plus noise that is this these two parts minus  $E$  of  $H_i$  into  $E$  of  $P$  deterministic plus noise ok.

And now I take, I basically take out, I take these variance of  $P$  deterministic as common. And therefore if I write square root of variance of  $H_i$  into variance of  $P$  deterministic, then this is divided by  $P$  deterministic or variance of  $P$  deterministic. So, it becomes 1 plus variance of noise divided by variance of  $P$  deterministic ok.

And this part essentially is nothing but 1 by SNR because this is the variance of  $P$  deterministic and this is the variance of noise. So, this is 1 by SNR. And what is this part? If you observe here, then there is an additional part which were adding because of noise, but again since noise is independent, you can write this pretty much is nothing but the correlation of the deterministic component of this  $P$  deterministic with  $L_t$  ok.

(Refer Slide Time: 12:28)

**Relating Correlation with SNR**

- If  $H_i$  and  $N$  are independent,  $Cov(H_i, N) = 0 \Rightarrow 0 = E(H_i \cdot N) - E(H_i)E(N) \Rightarrow E(H_i \cdot N) = E(H_i)E(N)$ .
- $\therefore E(H_i \cdot (P_{det} + N)) - E(H_i)E(P_{det} + N) = E(H_i \cdot P_{det}) + E(H_i \cdot N) - E(H_i)E(P_{det}) - E(H_i)E(N) = E(H_i \cdot P_{det}) - E(H_i)E(P_{det})$
- Thus,  $Corr(H_i, L_t) = \frac{E(H_i \cdot P_{det}) - E(H_i)E(P_{det})}{\sqrt{Var(H_i)Var(P_{det})} \sqrt{1 + \frac{Var(N)}{Var(P_{det})}}} = \frac{Corr(H_i, P_{det})}{\sqrt{1 + \frac{1}{SNR}}}$

The above result can be used to evaluate an architecture based on simulated power traces. The simulation computes the value of  $Corr(H_i, P_{det})$ . However, the real correlation can be computed using the SNR.

So, what I mean is that now and I know that if  $H_i$  and  $N$  are independent; that means, if the noise independent, then again the covariance between  $H_i$  and  $N$  is 0 and therefore, I can write that  $E$  of  $H_i$  comma  $N$  as nothing but  $E$  of  $H_i$  into  $E$  of  $N$  and therefore, the

numerator as we saw previously was  $E\{H_i \text{ comma } P \text{ deterministic plus noise minus } E\{H_i \text{ multiplied with } E\{P \text{ t plus noise.}$

This I can write as  $E\{H_i \text{ comma } P \text{ deterministic I mean } E\{H_i \text{ multiplied with } P \text{ deterministic plus } E\{H_i \text{ multiplied with } N \text{ minus } E\{H_i \text{ multiplied with } E\{P \text{ deterministic minus } E\{H_i \text{ multiplied with } E\{N \text{ because this } E\{P \text{ deterministic plus noise. I can write it as } E\{P \text{ deterministic plus } E\{N \text{ ok, I can break it up. And this if I assume right that this particular relationship that is } E\{H_i \text{ comma } N \text{ is equal to } E\{H_i \text{ into } E\{N, \text{ then there is a cancelling that I can do here. And therefore, I get } E\{H_i \text{ comma } P \text{ sorry } E\{H_i \text{ into } P \text{ deterministic minus } E\{H_i \text{ into } E\{P \text{ deterministic. And this if I divide by the square root of variance of } H_i \text{ and variance of } P \text{ deterministic, then I get that as the correlation between } H_i \text{ and } P \text{ deterministic.}$

And this is divided by square root of  $1 + 1 \text{ by SNR}$ . So, this is a very useful result because now it tells me that I can calculate this component that is the correlation of  $H_i$  and  $P$  deterministic both that is the hypothetical power and this is a deterministic component. And note that deterministic component you can simulate. You can simulate based upon your hamming weight or hamming leakage model and then in order to get the actual estimate or at least to understand better, you just have to add up this SNR component ok.

These also tells us that the correlation and the SNR are you know like if I increase the SNR for example, then  $1 \text{ by SNR}$  will reduce and therefore, the denominator here will reduce and therefore, what will have that have an effect on the correlation. You will say a correlation will also increase ok. So, you can basically understand the relationship between correlation and SNR from this simple equation.



(Refer Slide Time: 14:59)

The slide is titled "DPA on an Iterative Architecture with Parallel S-Boxes". It features a block diagram of an AES round, a graph showing the relationship between Guessing Entropy and Power Traces, and several text boxes explaining the process.

**Diagram:** A block diagram showing "PlainText" and "Last Round" entering an "AES Round" block, which then outputs "Ciphertext".

**Text Box 1:** "Power traces are collected and correlated with hypothetical power values to perform CPA. All the S-Boxes are in parallel in this architecture, providing algorithmic noise."

**Text Box 2:** "We acquired 70,000 power traces and divided into sets of 3000 traces each. We perform the CPA attacks on them, and plot the average Guessing Entropy for 3000 traces."

**Graph:** A line graph titled "Plotting Guessing Entropy for Parallel AES". The y-axis is "Guessing Entropy" (0 to 60) and the x-axis is "Power Traces" (0 to 3000). The curve starts at approximately 60 for 0 traces and drops sharply to 0 by around 1500 traces.

**Logos:** The bottom of the slide contains logos for "swayam" and "INDIA RISE, LITERACY RISE".

So, this is a useful result thereby. So, now, what we can do is we can try to compare between you know like architectures. For example, let us take the DPA of consider the DPA on an iterative architecture. So, this is the AES round which has been repeated and that is what we have considering. In particular, this is parallel architecture which means all the S boxes all the 16 S boxes are executing in parallel ok. So, if we are targeting one of the S box and the other 15 S boxes are contributing to generate algorithmic noise.

So, if you consider this particular architecture we basically, you know like did a dp attack on this we took around 70000 power traces divided it into 3000 traces and here it shows you know like how the guessing entropy falls with 3000 power traces, but it plots the average guessing entropy. So, this average is done on all these you know like 70000 divided by 3000 possible classes ok.

So, basically what it means is that you get the guessing entropy for every 3000 power traces and you plot it on the y axis; plot it on the y axis with the number of observations. As you can see, around 1500 power traces, the guessing entropy has reduced to 0 which means you know with certainty what is the key ok, the key has been found out. If you compare this remember that there is an algorithmic noise here and you took around 1500 observations to get the actual key.

(Refer Slide Time: 16:26)

**DPA on a Serialized Architecture of AES**

- Serialized architecture, wherein there is only one S-Box.
- A MUX is used to select out one byte from the 128-bit state and process by the S-Box.
- Less algorithmic noise.

Plot of average Guessing Entropy for 1000 traces, averaged over 40 sets. Compared to the parallel architecture, one can see the less number of observations required.

| Power Traces | Guessing Entropy |
|--------------|------------------|
| 0            | 100              |
| 100          | 75               |
| 200          | 55               |
| 300          | 45               |
| 400          | 35               |
| 500          | 25               |
| 600          | 15               |
| 700          | 10               |
| 800          | 5                |
| 900          | 2                |
| 1000         | 0                |

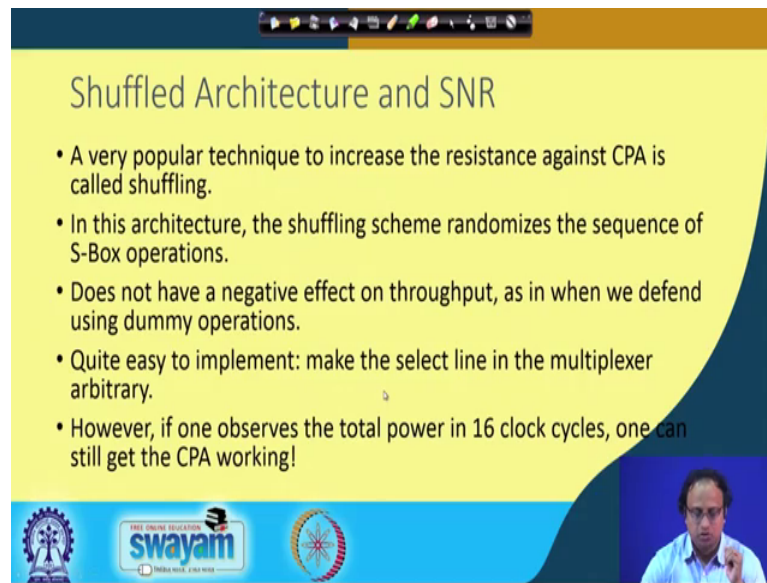
Logos: Swamyam (Free Online Education), IIT Madras, and a university logo.

If you now compare this with the serialize architecture, in a serialize architecture these are the diagram of the serialized architecture, you see that there is only one single S box which is essentially being you know doing the computation and there is a multiplexer at the beginning. So, the multiplexer takes out any byte from this 128 bit register does a cohering and then you know like puts it back into it is corresponding position ok. And of course, there are diffusion layers and other functionalities of an AES round.

So, here in this serialized architecture, there is only one S box. So that means, that there is no algorithmic noise, when you are targeting that S box, there is no parallel S box which is also under play because there is only one S box in the hardware. There is no there are no parallel S boxes in the architecture. So, therefore, right there is less algorithmic noise and exactly that is what you see here when you are doing the power attack.

For example, in this case we observe the guessing entropy for 1000 power traces, note that is lesser than the 300 power traces that we took and here around 500 to 600 right observations, the guessing entropy goes to 0 which means now the attack is faster it is you know like the success rate is more, the guessing entropy is reducing faster.

(Refer Slide Time: 17:47)



The slide is titled "Shuffled Architecture and SNR" and contains the following bullet points:

- A very popular technique to increase the resistance against CPA is called shuffling.
- In this architecture, the shuffling scheme randomizes the sequence of S-Box operations.
- Does not have a negative effect on throughput, as in when we defend using dummy operations.
- Quite easy to implement: make the select line in the multiplexer arbitrary.
- However, if one observes the total power in 16 clock cycles, one can still get the CPA working!

The slide also features a video inset of a man speaking in the bottom right corner, and logos for "swayam" and "INDIA RISE, EDUCATION RISE" at the bottom.

So, you can actually do an another interesting variation of this. So, is called as a shuffled architecture. So, in a shuffled architecture, what you do is, you basically try to randomise this select line of the multiplexer. So, basically; that means, that this 16 S boxes that we have in one round of AES, you do them in any arbitrary order because they are independent.

They can be done in any order ok. If you do this, so then you can observe that a it is a very if there is a interesting counter measure because it does not have any negative effect on the throughput. For example, if you think of nice counter measures like for example, adding a dummy operation that will add on to your throughput. In this case, there is no penalty negative penalty on the throughput.

So, it is quiet easy to implement because you just need to make the select line in the multiplexer arbitrary ok; however, if you observe right you can still make interesting things that you can still make the DP attack work. The reason why you can make it work is that if you observe the total power in the 16 clock cycles, then you still remains the same because although you are doing them in a any arbitrary order, but if you add up the power assumption for every S box right, the total power in 16 clock cycles does not change ok. And this is a trick which you can adopt to attack shuffled architectures ok.

(Refer Slide Time: 19:03)

**Comparing based on SNR Computations**

- **Serialized Architecture:**
  - We observe the power consumption say during the 0<sup>th</sup> S-Box computation.
  - We compute the Hamming Distance (or weight) corresponding to the 0<sup>th</sup> S-Box. Note for this we need the key.
    - Classes can be from 0 to 8.
  - We calculate the average of each class:  $p_i$ , for  $0 \leq i \leq 8$ .
  - The number of elements in each class is say  $f_i$ .
  - Thus, Signal is  $Var(E[L_r|S]) = \frac{\sum_{i=0}^8 f_i p_i^2}{\sum_{i=0}^8 f_i} - \left( \frac{\sum_{i=0}^8 f_i p_i}{\sum_{i=0}^8 f_i} \right)^2$
  - For the noise component from each power trace, deduct the average  $p_i$  if the trace belongs to the  $i^{th}$  class. Calculate the variance across all the classes.

Handwritten notes on the slide include:  $p_0, p_1, \dots, p_8$  in boxes labeled '0-bin', '1-bin', ..., '8-bin'. Below this,  $T_0, T_1, \dots, T_{15} \rightarrow P_0$  is written. At the bottom,  $0 \rightarrow i^{\text{th}} \rightarrow \int T_0 - p_0, T_1 - p_1, \dots, T_{15} - p_{15}$  is written.

So, therefore, now let us again you know like look back and compare this architectures using SNR as my tool. So, in the serialized architectures, so let us start with the serialized architecture. So, here we observe the power consumption say during the 0th S box computation. So, suppose let us consider the 0th S box computation, there is there are you know like you are doing like 0 to 15 right. So, there are 16 as times that you are computing on the S box.

So, you observe the power consumption say during the 0th S box computation, you note or you compute the hamming distance or hamming weight depending upon your power model. Corresponding to the 0th S box, note that for this we need the key ok. And the you know as we know that the classes can from 0 to 8. So, remember I told you right that when we are computing the SNR and if you are you know like you need you know the key ok.

So, this is typically done in a template setting where you basically have an access to the key you know the key. If you do not know, the key you cannot guess the SNR in this way. So, if you note the key, then you can find out depending upon. So, you are doing from the ciphertext, you can find out the target register that you are talking about what is the corresponding hamming weight or the hamming distance corresponding to that state ok. So, that would imply that if you take many power observations, you can basically split them into the into bins depending upon it is hamming class.

So, therefore, now what we do is that we calculate the average of each class and say  $\mu_i$  denoted as  $\mu_i$  and imagine that in a particular  $i$ th bin. So, the number of values that is over there is that denoted as  $f_i$  ok. So, therefore, the variance of  $E_L t$  given  $S$  can be find out by simply taking this  $E$  of  $x^2$  minus  $E x$  whole square. So, this is you know nothing but a  $\sum_{i=0}^8 f_i \mu_i^2$  because a hamming classes 8 a  $\mu_i$  if there are 9 hamming classes 0, 1, 2, 3 till 8. And then I multiply  $f_i$  into  $\mu_i^2$  divided by  $\sum_{i=0}^8 f_i$  minus I take the mean and I square. So, this is the nothing but the mean of squares minus square of the mean ok.

So, I take the mean here find out the mean which is given by the ratio of  $\sum_{i=0}^8 f_i \mu_i$  divided by  $\sum_{i=0}^8 f_i$  and I square it ok. So, if you take this, then you get the variance of  $E_L t$  given  $S$ . Now for the noise component what we do is from each power trace, you deduct the average  $\mu_i$  if the trace belongs to the  $i$ th class. So, if the trace belongs to the  $i$ th class, then you subtract out  $\mu_i$  ok.

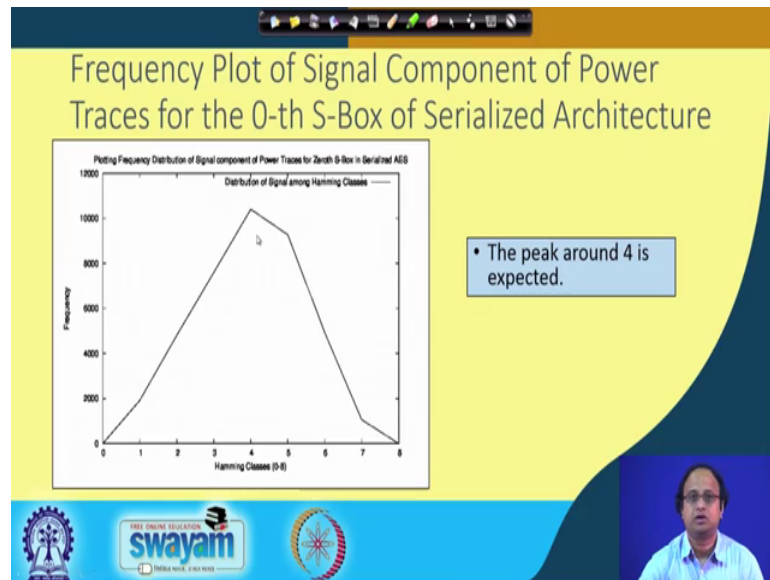
So, now, we have got you know like. So, for example, in the first class, so I call it as 0th class you had a  $f_0$  observations right. In the second class, we had  $f_1$  observations  $f_2, f_3$  and so on till  $f_8$  observations. So, in the first class, if there are  $f_0$  observations, then for each of these observations we have subtracted out the mean of that particular bin ok.

So, therefore, you we then get  $f_0$  values and all of them are noise values. Likewise in the second bin you have got  $f_1$  noise values ok. So, I hope I am clear about here. So, what I mean is if we considered that these are your bins ok. So, we are basically drawing bins depending upon you know like the hamming classes. Suppose I call this as 0 bin, this is your one bin ok, likewise suppose this is your 8th bin. So, I am considering that there are 9 bins and you take your power values and suppose here there are  $f_0$  power values which goes in over here there are  $f_1$  power values and there are suppose there are  $f_8$  power values. So, therefore, for all these power values here like suppose the power values are you know like, let me call this as say  $T_0, T_1$  and so on till say  $T_{f_0-1}$ .

So, what I do is that, I find out the average for these and then I subtract out the average from these values ok. So, suppose the average of these is  $\mu_0$ . So, therefore, my processing what I do is, I basically make it  $T_0 - \mu_0$ , then  $T_1 - \mu_0$  like this  $T_{f_0-1} - \mu_0$ . So, like this like this is for the 0th bin, you have similarly for all the possible 8 bins. And then you basically find out the average of these distribution, I

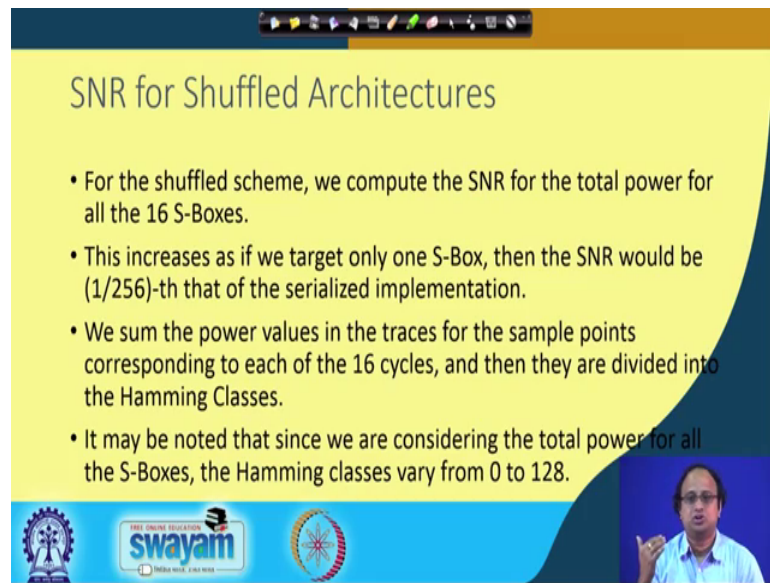
mean the variance of this distribution ok. And that essentially gives you the variance of the noise ok. And therefore, now you can take the ratio between them and calculate your SNR ok that is the simple idea.

(Refer Slide Time: 23:46)



So, now, with this background right, let us move ahead and see how and the frequency plot of the signal components. So, I am just observing the 0th S box of the serialized architecture. And I am seeing the frequency plot. So, again note that as expected around the 4 hamming class which occurs with maximum frequency I get a peak and therefore, I get the almost normal looking distribution.

(Refer Slide Time: 24:06)



The slide is titled "SNR for Shuffled Architectures" and contains the following bullet points:

- For the shuffled scheme, we compute the SNR for the total power for all the 16 S-Boxes.
- This increases as if we target only one S-Box, then the SNR would be (1/256)-th that of the serialized implementation.
- We sum the power values in the traces for the sample points corresponding to each of the 16 cycles, and then they are divided into the Hamming Classes.
- It may be noted that since we are considering the total power for all the S-Boxes, the Hamming classes vary from 0 to 128.

The slide also features a video inset of a speaker in the bottom right corner and logos for Swamyam and other organizations at the bottom.

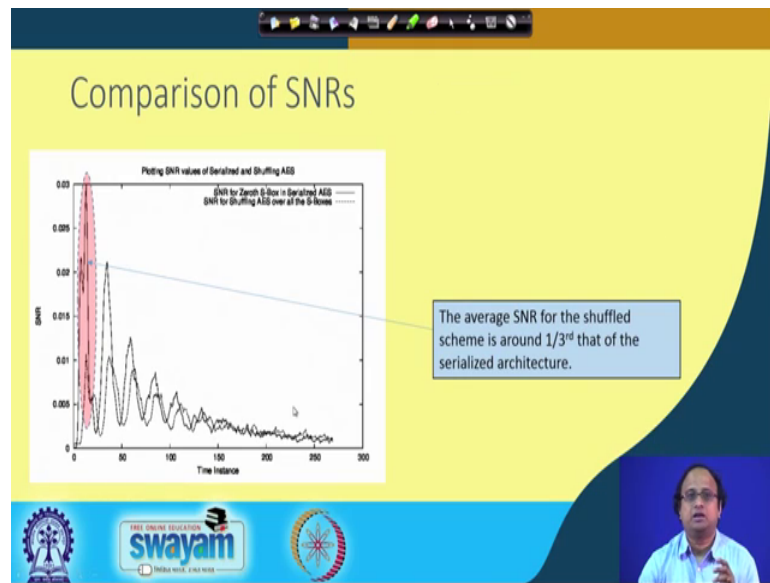
So, now you can actually do the similar thing for the SNR for calculation of shuffle architectures. Note that, in if you just you know like apply the SNR in a similar way as we doing for the serialized architecture, there you will basically get a degradation of 1 by 256 because you do not, your average right will be reduced by 1 by 16 and if you take the square of that because in the, you have the numerator a variance, so, it will be reduced by 1 by 256.

So, rather as we have discussed right attack strategy would be to basically sum up the power values for 16 block cycles. So, therefore, right for the shuffled scheme, we compute the SNR for the total power for all the 16 S boxes. This increases as if we target only one S box, then the SNR would be 1 by 256 that of the serialized implementation. So, what we do is we sum the power values in the traces for the sample points corresponding to each of the 16 cycles and then you basically find out this distributed in hamming class. Note that when you are adding them up, now the number of hamming classes is not between 0 to 8, but it is between 0 to 128 because you are adding up all the individual bytes ok.

So, if the first one is say a maximum value can be 8, the second one can be 8 and so on if you add 16 eights then it can lead up to 128 ok. So, you have more hamming classes from 0 to 128 ok.



(Refer Slide Time: 25:33)

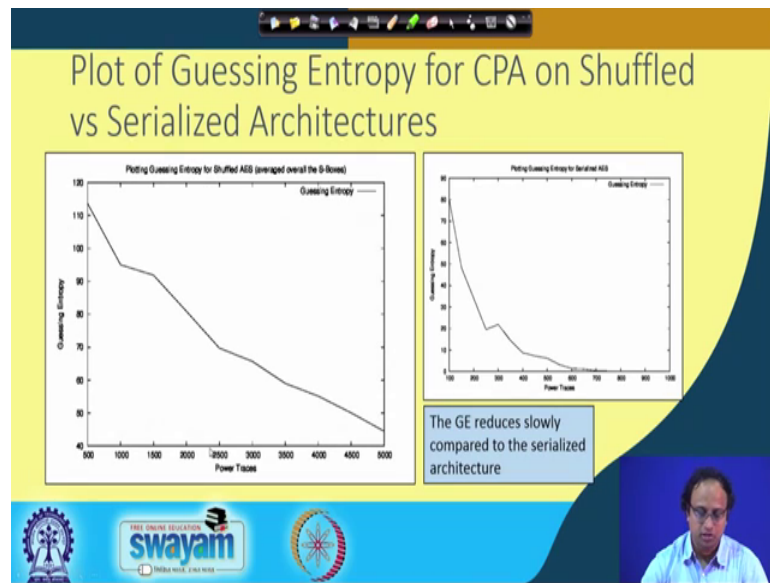


So, now if you do that and if you plot the SNR, so here is a comparison of the SNR. So, we are basically plotted the SNR with the time instance. Interestingly, you find out the average SNR for the shuffled scheme is still lesser than that of the serialized architecture, but now it is only one third lesser a not like 1 by 256 as would have been if I have done the attack targeting only 1 byte.

So, if you for example, compare this is if you see the certain line right and if you compare this peak with this peak, then you will find a rough ratio of 3 which means that you know like in the serialize architecture the SNR is more almost 3 times more than that of the shuffled architecture; which means that shuffled architecture essentially is able to reduce the signal to noise ratio and therefore is more protective against a differential power attack.



(Refer Slide Time: 26:24)



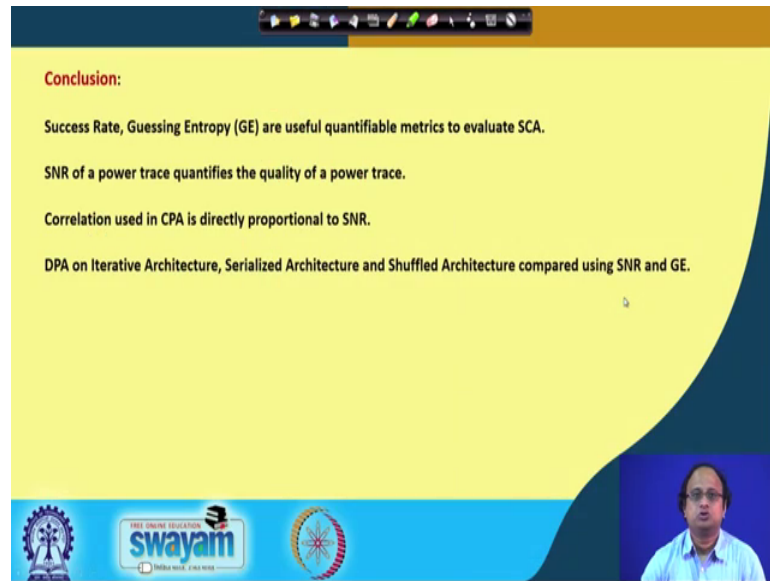
And that is you we can observe it from the you know like guessing entropy plots also. So, if you remember like, we showed that for the serialize AES we require around 600 to 700 power traces, here it is more than 5000 power traces ok. So, you can see that the guessing entropy reduces and much slower ok. And therefore, this shows that it is a H i would say you know like better protection compared to the serialized or is a better design compared to the serialize architecture with respect to power attacks ok.

(Refer Slide Time: 26:56)



So, again just to conclude this is the usual referring that I have been following.

(Refer Slide Time: 27:01)



**Conclusion:**

- Success Rate, Guessing Entropy (GE) are useful quantifiable metrics to evaluate SCA.
- SNR of a power trace quantifies the quality of a power trace.
- Correlation used in CPA is directly proportional to SNR.
- DPA on Iterative Architecture, Serialized Architecture and Shuffled Architecture compared using SNR and GE.

swayam  
INDIA WISE

So, to conclude what we have discussed is, we have discussed about success rate guessing entropy GE which are useful quantifiable matrix to evaluate side channel attacks. We have discussed that SNR of a power trace can be used to quantify the quality of a power trace and the correlation which is often a very useful metric for CPA or Correlation Power Attack is directly proportional to SNR and the DPA on iterative architecture serialized architectures and shuffle architectures can be compared using these metrics SNR and GE.

So, therefore, using SNR and G E, we can give comparative we can compare between the performances of differential power attack across these architectures ok. So, we will see how we can you know like augmented and the do more interesting variations of these metrics in some later class.

So thank you for your attention.