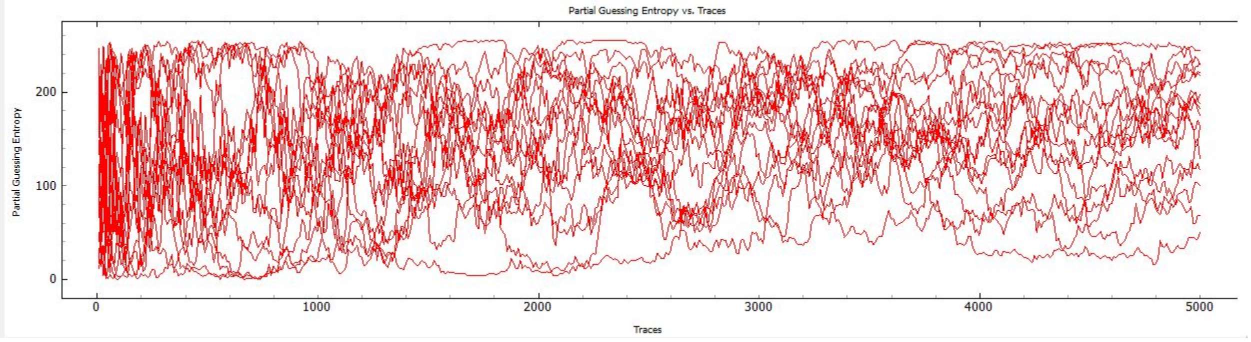
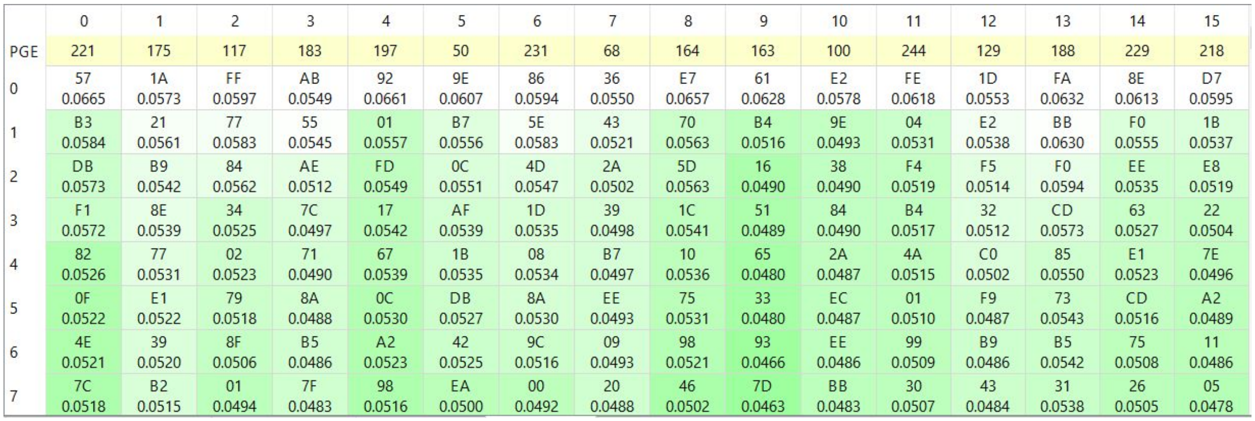
Ashton Gray & Grady Sullivan
ECE 371
Lab 3

Default CPA attack

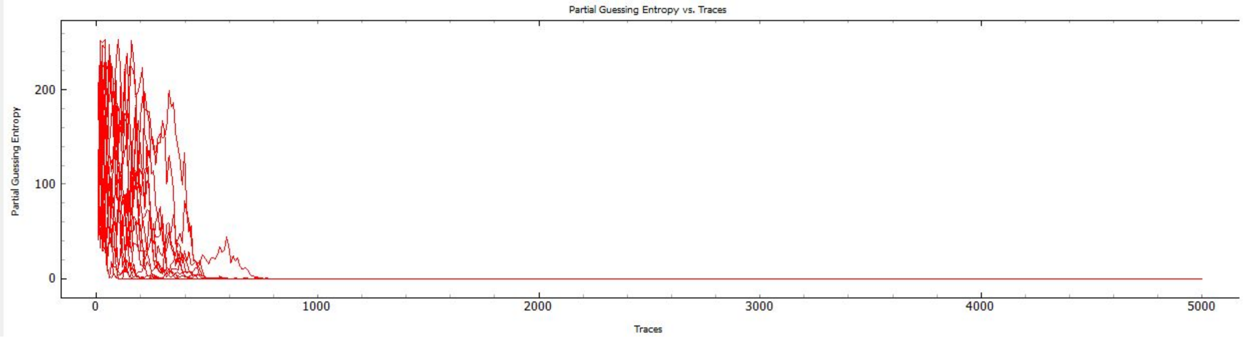


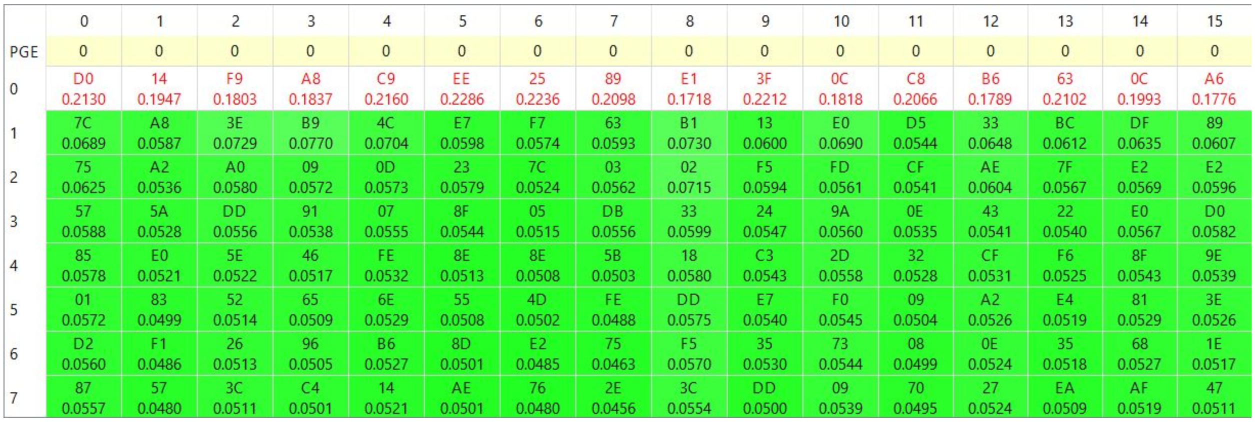


	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PGE	221	175	117	183	197	50	231	68	164	163	100	244	129	188	229	218
0	57 0.0665	1A 0.0573	FF 0.0597	AB 0.0549	92 0.0661	9E 0.0607	86 0.0594	36 0.0550	E7 0.0657	61 0.0628	E2 0.0578	FE 0.0618	1D 0.0553	FA 0.0632	8E 0.0613	D7 0.0595
1	B3 0.0584	21 0.0561	77 0.0583	55 0.0545	01 0.0557	B7 0.0556	5E 0.0583	43 0.0521	70 0.0563	B4 0.0516	9E 0.0493	04 0.0531	E2 0.0538	BB 0.0630	F0 0.0555	1B 0.0537
2	DB 0.0573	B9 0.0542	84 0.0562	AE 0.0512	FD 0.0549	0C 0.0551	4D 0.0547	2A 0.0502	5D 0.0563	16 0.0490	38 0.0490	F4 0.0519	F5 0.0514	F0 0.0594	EE 0.0535	E8 0.0519
3	F1 0.0572	8E 0.0539	34 0.0525	7C 0.0497	17 0.0542	AF 0.0539	1D 0.0535	39 0.0498	1C 0.0541	51 0.0489	84 0.0490	B4 0.0517	32 0.0512	CD 0.0573	63 0.0527	22 0.0504
4	82 0.0526	77 0.0531	02 0.0523	71 0.0490	67 0.0539	1B 0.0535	08 0.0534	B7 0.0497	10 0.0536	65 0.0480	2A 0.0487	4A 0.0515	C0 0.0502	85 0.0550	E1 0.0523	7E 0.0496
5	0F 0.0522	E1 0.0522	79 0.0518	8A 0.0488	0C 0.0530	DB 0.0527	8A 0.0530	EE 0.0493	75 0.0531	33 0.0480	EC 0.0487	01 0.0510	F9 0.0487	73 0.0543	CD 0.0516	A2 0.0489
6	4E 0.0521	39 0.0520	8F 0.0506	B5 0.0486	A2 0.0523	42 0.0525	9C 0.0516	09 0.0493	98 0.0521	93 0.0466	EE 0.0486	99 0.0509	B9 0.0486	B5 0.0542	75 0.0508	11 0.0486
7	7C 0.0518	B2 0.0515	01 0.0494	7F 0.0483	98 0.0516	EA 0.0500	00 0.0492	20 0.0488	46 0.0502	7D 0.0463	BB 0.0483	30 0.0507	43 0.0484	31 0.0538	26 0.0505	05 0.0478

This attack was not a success because the PGE values for each byte were high, therefore it took many guesses to find a higher probability of finding the correct key from power leakage. The more guesses means that it cannot be sure on which value is the correct key value for each key byte.

CPA with Last Round State key leakage

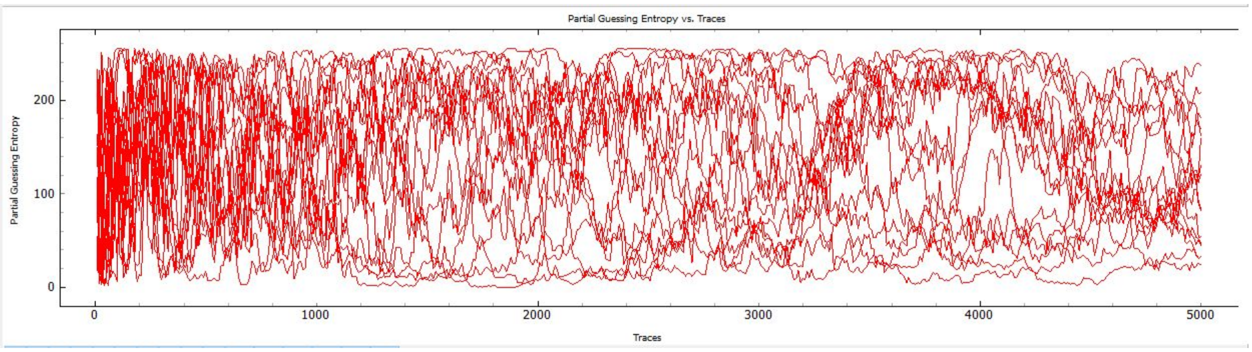


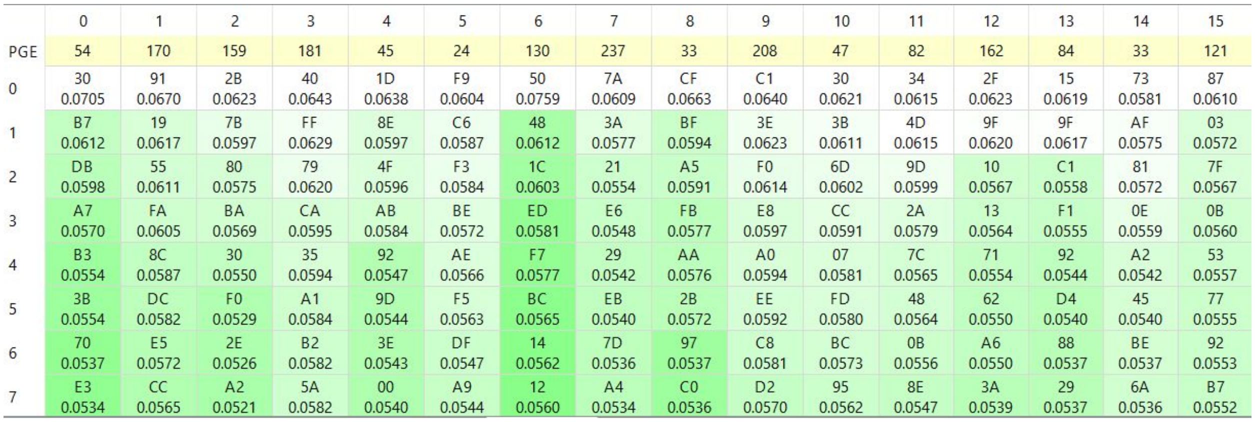


	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PGE	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	D0 0.2130	14 0.1947	F9 0.1803	A8 0.1837	C9 0.2160	EE 0.2286	25 0.2236	89 0.2098	E1 0.1718	3F 0.2212	0C 0.1818	C8 0.2066	B6 0.1789	63 0.2102	0C 0.1993	A6 0.1776
1	7C 0.0689	A8 0.0587	3E 0.0729	B9 0.0770	4C 0.0704	E7 0.0598	F7 0.0574	63 0.0593	B1 0.0730	13 0.0600	E0 0.0690	D5 0.0544	33 0.0648	BC 0.0612	DF 0.0635	89 0.0607
2	75 0.0625	A2 0.0536	A0 0.0580	09 0.0572	0D 0.0573	23 0.0579	7C 0.0524	03 0.0562	02 0.0715	F5 0.0594	FD 0.0561	CF 0.0541	AE 0.0604	7F 0.0567	E2 0.0569	E2 0.0596
3	57 0.0588	5A 0.0528	DD 0.0556	91 0.0538	07 0.0555	8F 0.0544	05 0.0515	DB 0.0556	33 0.0599	24 0.0547	9A 0.0560	0E 0.0535	43 0.0541	22 0.0540	E0 0.0567	D0 0.0582
4	85 0.0578	E0 0.0521	5E 0.0522	46 0.0517	FE 0.0532	8E 0.0513	8E 0.0508	5B 0.0503	18 0.0580	C3 0.0543	2D 0.0558	32 0.0528	CF 0.0531	F6 0.0525	8F 0.0543	9E 0.0539
5	01 0.0572	83 0.0499	52 0.0514	65 0.0509	6E 0.0529	55 0.0508	4D 0.0502	FE 0.0488	DD 0.0575	E7 0.0540	F0 0.0545	09 0.0504	A2 0.0526	E4 0.0519	81 0.0529	3E 0.0526
6	D2 0.0560	F1 0.0486	26 0.0513	96 0.0505	B6 0.0527	8D 0.0501	E2 0.0485	75 0.0463	F5 0.0570	35 0.0530	73 0.0544	08 0.0499	0E 0.0524	35 0.0518	68 0.0527	1E 0.0517
7	87 0.0557	57 0.0480	3C 0.0511	C4 0.0501	14 0.0521	AE 0.0501	76 0.0480	2E 0.0456	3C 0.0554	DD 0.0500	09 0.0539	70 0.0495	27 0.0524	EA 0.0509	AF 0.0519	47 0.0511

This attack was a success because the PGE values for each byte were low, therefore it took less guesses to find a higher probability of finding the correct key from power leakage.

CPA with Last Round State key leakage with noise





	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PGE	54	170	159	181	45	24	130	237	33	208	47	82	162	84	33	121
0	30 0.0705	91 0.0670	2B 0.0623	40 0.0643	1D 0.0638	F9 0.0604	50 0.0759	7A 0.0609	CF 0.0663	C1 0.0640	30 0.0621	34 0.0615	2F 0.0623	15 0.0619	73 0.0581	87 0.0610
1	B7 0.0612	19 0.0617	7B 0.0597	FF 0.0629	8E 0.0597	C6 0.0587	48 0.0612	3A 0.0577	BF 0.0594	3E 0.0623	3B 0.0611	4D 0.0615	9F 0.0620	9F 0.0617	AF 0.0575	03 0.0572
2	DB 0.0598	55 0.0611	80 0.0575	79 0.0620	4F 0.0596	F3 0.0584	1C 0.0603	21 0.0554	A5 0.0591	F0 0.0614	6D 0.0602	9D 0.0599	10 0.0567	C1 0.0558	81 0.0572	7F 0.0567
3	A7 0.0570	FA 0.0605	BA 0.0569	CA 0.0595	AB 0.0584	BE 0.0572	ED 0.0581	E6 0.0548	FB 0.0577	E8 0.0597	CC 0.0591	2A 0.0579	13 0.0564	F1 0.0555	0E 0.0559	0B 0.0560
4	B3 0.0554	8C 0.0587	30 0.0550	35 0.0594	92 0.0547	AE 0.0566	F7 0.0577	29 0.0542	AA 0.0576	A0 0.0594	07 0.0581	7C 0.0565	71 0.0554	92 0.0544	A2 0.0542	53 0.0557
5	3B 0.0554	DC 0.0582	F0 0.0529	A1 0.0584	9D 0.0544	F5 0.0563	BC 0.0565	EB 0.0540	2B 0.0572	EE 0.0592	FD 0.0580	48 0.0564	62 0.0550	D4 0.0540	45 0.0540	77 0.0555
6	70 0.0537	E5 0.0572	2E 0.0526	B2 0.0582	3E 0.0543	DF 0.0547	14 0.0562	7D 0.0536	97 0.0537	C8 0.0581	BC 0.0573	0B 0.0556	A6 0.0550	88 0.0537	BE 0.0537	92 0.0553
7	E3 0.0534	CC 0.0565	A2 0.0521	5A 0.0582	00 0.0540	A9 0.0544	12 0.0560	A4 0.0534	C0 0.0536	D2 0.0570	95 0.0562	8E 0.0547	3A 0.0539	29 0.0537	6A 0.0536	B7 0.0552

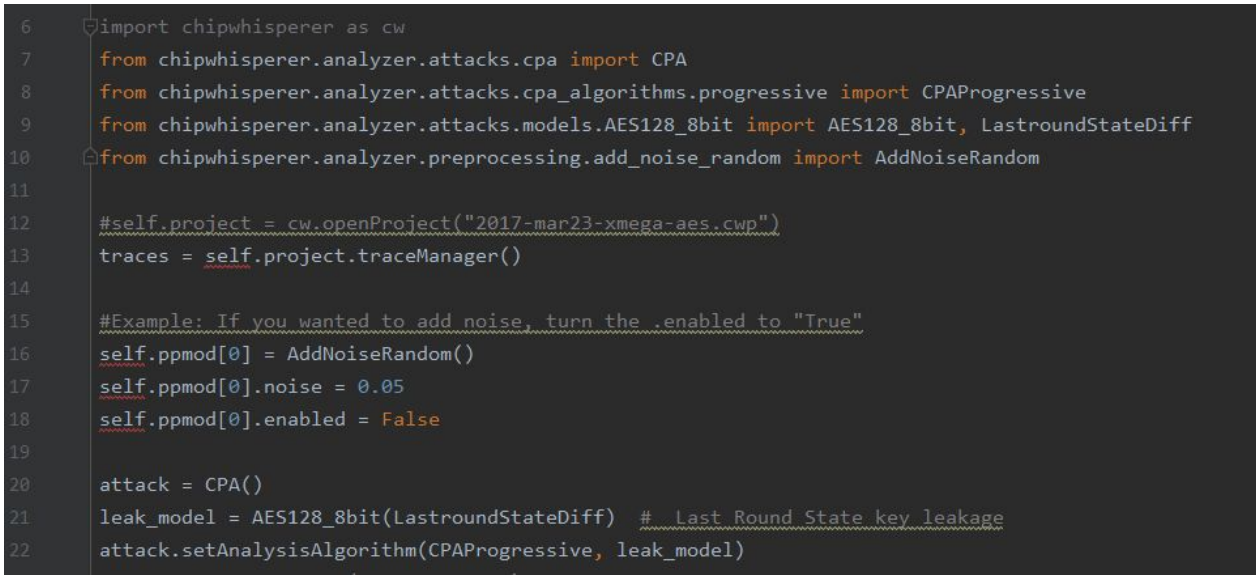
This attack was not a success because the PGE values for each byte were high, therefore it took many guesses to find a higher probability of finding the correct key from power leakage. The more guesses means that it cannot be sure on which value is the correct key value for each key byte.

Trace number for the first time where PGE <=5 and PGE = 0 for all bytes

	Default CPA attack			CPA with Last Round State key leakage			CPA with Last Round State key leakage with noise	
Byte Number	PGE <= 5	PGE = 0		PGE <= 5	PGE = 0		PGE <= 5	PGE = 0
0	x	x		230	300		2060	x
1	600	740		700	740		380	x
2	x	x		350	430		90	x
3	x	x		490	500		330	x
4	40	x		150	170		10	x
5	30	90		350	440		x	x
6	150	560		60	100		100	x
7	x	x		50	460		x	x
8	x	x		110	380		250	x
9	x	x		440	550		1120	x
10	90	x		320	560		x	x
11	x	x		100	100		x	x
12	620	670		160	170		170	700
13	470	x		150	250		60	x
14	40	x		400	440		1290	x
15	x	x		480	500		490	x

In this lab, we executed the correlated power analysis attacks under different circumstances. The default CPA attack's code was supplied for the lab, which we were able to run using Chipwhisperer. The results, as stated above, found that the attack was not successful due to the high PGE values for each bit.

For the CPA attack with the last round state key leakage, we edited the original CPA attack python file and imported the "LastroundStateDiff" module, instead of the "SBox_output. We then set this as the argument to the AES128_8bit function in order to change the leak model to a CPA attack with last state key leakage. These two changes can be found in lines 9 and 21 in the image below:



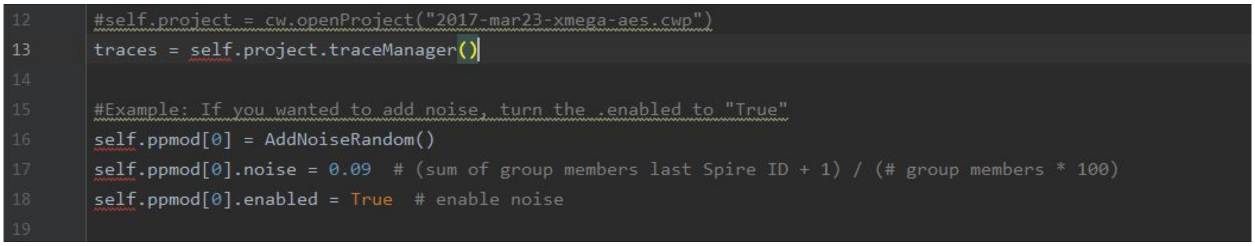
```python
import chipwhisperer as cw
from chipwhisperer.analyzer.attacks.cpa import CPA
from chipwhisperer.analyzer.attacks.cpa_algorithms.progressive import CPAProgressive
from chipwhisperer.analyzer.attacks.models.AES128_8bit import AES128_8bit, LastroundStateDiff
from chipwhisperer.analyzer.preprocessing.add_noise_random import AddNoiseRandom

#self.project = cw.openProject("2017-mar23-xmega-aes.cwp")
traces = self.project.traceManager()

#Example: If you wanted to add noise, turn the .enabled to "True"
self.ppmod[0] = AddNoiseRandom()
self.ppmod[0].noise = 0.05
self.ppmod[0].enabled = False

attack = CPA()
leak_model = AES128_8bit(LastroundStateDiff)  #  Last Round State key leakage
attack.setAnalysisAlgorithm(CPAProgressive, leak_model)
```

The PGE values for each bit was zero, indicating that it took fewer attempts to find a higher probability of finding the correct key from power leakage.

The final attack was the same as last round state key leakage CPA, but we included noise. The noise value was determined by using the following formula:

$$Noise = (Sum\ of\ last\ digit\ of\ SPIRE\ ID\ for\ all\ group\ members\ +\ 1) / (Number\ of\ Group\ Members\ *\ 100)$$

This gave our group a noise value of 0.09, which we were able to replace the default noise value of 0.05. We then enabled noise by setting the attribute of noise to true. The changes can be seen in the image below, in lines 17 and 18:



```python
12   #self.project = cw.openProject("2017-mar23-xmega-aes.cwp")
13   traces = self.project.traceManager()
14
15   #Example: If you wanted to add noise, turn the .enabled to "True"
16   self.ppmod[0] = AddNoiseRandom()
17   self.ppmod[0].noise = 0.09  # (sum of group members last Spire ID + 1) / (# group members * 100)
18   self.ppmod[0].enabled = True  # enable noise
19
```

This gave us higher PGE values than the previous test, which means the attack was not successful.