Power Analysis Part VII: 2nd Order DPA

Cryptographic Hardware for Embedded Systems
ECE 3170

Fall 2025

Assoc. Prof. Vincent John Mooney III

Georgia Institute of Technology

Reading

- This lecture is based on the following source:
- "Using Second-Order Power Analysis to Attack DPA Resistant Software," by T.S. Messerges, CHES 2000, *Lecture Notes in Computer Science* (LCNS) vol. 1965, pp. 238–251.
- All figures in this lecture are from the above publication.

Second-order Differential Power Analysis

- Preprocess the data
- In 2nd order DPA, the data is combined in a particular way prior to looking for differences among groups of power traces
- Recall that in 1st order DPA, raw power trace values are used directly

Consider the Following Masking Type

- Boolean
  - Intermediate values *v* and *u* are concealed (at different points in time and at different parts of the cryptographic algorithm) by XOR with the identical mask *m*
  - $v_m = v \oplus m$
  - $u_m = u \oplus m$
- Why the same mask?
  - In order to avoid the necessity of the equivalent of a one-time pad, there is some amount of mask reuse in practical implementations

Possible Preprocessing Step

- $w = comb(u_m,v_m) = u_m \oplus v_m$
- Note that $u_m \oplus v_m = u \oplus m \oplus v_m = u \oplus m \oplus v \oplus m = u \oplus v \oplus m \oplus m$ $= u \oplus v \oplus \mathbf{0}$ (where **0** has all zeros and is the same size as u, v and m) $= u \oplus v$
- Therefore $w = comb(u_m,v_m) = u_m \oplus v_m = u \oplus v = comb(u,v)$!!
  - Note that we know that $u_m \oplus v_m = u \oplus v$ independent of the mask, i.e., without knowing the specific value of m !!!

Preprocessed Traces

- Typically, not certain exactly when u_m and v_m occur
- However, there is a reasonable interval over which can compare
- E.g., let the interval be $I = t_{r+1}, \dots, t_{r+\ell}$ which likely contains u_m and v_m
- Preprocessing compares points t_x, t_y with $x \neq y$
- $(pre(t_{r+1},t_{r+2}), pre(t_{r+1},t_{r+3}),\dots, pre(t_{r+1},t_{r+\ell}), pre(t_{r+2},t_{r+3}),\dots, pre(t_{r+\ell-1},t_{r+\ell}))$
- Size of $pre(t_{r+1},t_{r+2}), pre(t_{r+1},t_{r+3}),\dots, pre(t_{r+1},t_{r+\ell})$ is ℓ-1
- Size of $pre(t_{r+2},t_{r+3}), pre(t_{r+2},t_{r+4}),\dots, pre(t_{r+2},t_{r+\ell})$ is ℓ-2
- …
- Total length is $(\ell-1) + (\ell-2) + \dots + 2 + 1 = \ell(\ell-1)/2$

2^{nd} and Higher Order DPA Attacks

1. Choose an intermediate part of the algorithm to attack
   a. For example, function $f(d,k)$ where d is a data input and k is a small part of the secret key stored in the device under attack
   b. Typically d is either plaintext or ciphertext
2. Make a large number of power measurements
   a. Keep track of the known data values d_i as recording the measurements
   b. For each d_i there exists a power trace of size T: $\boldsymbol{t'_i} = (t_{i,1}, \ldots, t_{i,T})$
   c. Preprocess the power traces; for 2^{nd} order, a window of size ℓ adds $\sim\ell^2$ preprocessing calculations
3. Calculate hypothetical intermediate values
   a. For each k, the K possible choices are $\mathbf{k} = (k_1, \ldots, k_K)$
   b. The possible choices are used in conjunction with $f(d,k)$
4. Map hypothetical intermediate values to resulting predicted power values
   a. For 2^{nd} order, duplicate the $\sim\ell^2$ preprocessing calculations
5. Compare predicted power values with the actual trace values
   a. Statistical tests involve differences, whether difference of means, covariance, or other

Two Definitions Introduced by Messerges

- The prior slides were generic & intended to explain/cover a larger number of prior papers and approaches in industry
- The next set of slides are based on one specific paper

Two Definitions Introduced by Messerges

- Defn. 1: an **n^{th}-order DPA** attack makes use of n different samples in the power (rate of energy consumption) signal that corresponds to n different intermediate values calculated during the execution of an algorithm.
- Defn. 2: A DPA attack against an algorithm's secret key is **sound** when it is theoretically possible to use power (rate of energy consumption) information to learn the values of all of the bits of the secret key.

Power Model of Messerges

- Energy consumption (i.e., power) at time *j* represented by $P[j] = \epsilon * d[j] + L + n$
- where d[*j*] represents the Hamming Weight (HW) of the intermediate data result at time *j*
- ϵ represents the incremental amount of power for each extra '1' in the HW
- *L* represents a constant amount of energy consumption (i.e., power)
- *n* represents the noise which is assume to also have a mean of zero (and hence can typically be ignored when calculating averages)

```
W1(PTI)
{
    A: Result = PTI ⊕ SecretKey
    . . .
    other operations . . .
    . . .
    return CTO
}
```

Vulnerable to first-order DPA Attack

```
W2(PTI)
{
    B: RandomMask = rand()
        mPTI = PTI ⊕ RandomMask
    C: Result = mPTI ⊕ SecretKey
    . . .
    other operations . . .
    . . .
    return CTO
}
```

Vulnerable to second-order DPA Attack

Fig. 1. Routines that Are Vulnerable to DPA Attacks

DPA (1st Order) Attack on W1

- Assume that the hardware has energy consumption varying significantly based on the HW of the data being operated upon
- Use the power model $P[j] = \epsilon * d[j] + L$ where the noise contribution is not included because it is assumed to be averaged out

```
W1 (PTI)
{
→ A: Result = PTI ⊕ SecretKey
```

Proposition 1. When the W_1 algorithm is implemented in an N-bit processor, where there is a linear relationship between the instantaneous power consumption and the Hamming weight of the data being processed, the following DPA attack is sound:

1. *Repeat for i equal to* 0 *through* $N - 1$ {
2. *Repeat for* $b = 0$ *to* 1 {
3. *Calculate the average power signal* $A_b[j]$ *by repeating the following:*{
4. *Set the ith bit of the PTI input to b.*
5. *Set the remaining PTI bits to random values.*
6. *Collect the algorithm's power signal.* } }
7. *Create the DPA bias signal* $T[j] = A_0[j] - A_1[j]$.
8. $T[j]$ *will have a positive bias spike when the ith secret key bit is a one, and will have a negative DPA bias spike when ith secret key bit is a zero.* }

1st Order DPA

- PTI = Plain Text Input
- Let the i^{th} bit of PTI be p_i
- Let the i^{th} bit of SecretKey be k_i
- Recall that the data have N bits each
- Now consider the expected value of the HW “d” of Result
- $E[d \mid k_i \oplus p_i = 0] = \frac{N-1}{2}$
- $E[d \mid k_i \oplus p_i = 1] = \frac{N+1}{2}$

```
W1 (PTI)
{
→ A: Result = PTI ⊕ SecretKey
```

1^{st} Order DPA (cont'd)

- Recall that $A_0[j^*]$ corresponds to line A with $p_i = 0$
- Similarly, $A_1[j^*]$ corresponds to line A with $p_i = 1$

When $k_i = 0$, equations for $A_0[j^*]$ and $A_1[j^*]$ can be written in terms of the expected values of P

$$A_0[j^*] \approx \mathrm{E}[P \,|\, k_i = 0, p_i = 0] = \mathrm{E}[d\varepsilon + L + n \,|\, k_i = 0, p_i = 0] = \frac{N-1}{2}\varepsilon + L \qquad (2)$$

$$A_1[j^*] \approx \mathrm{E}[P \,|\, k_i = 0, p_i = 1] = \mathrm{E}[d\varepsilon + L + n \,|\, k_i = 0, p_i = 1] = \frac{N+1}{2}\varepsilon + L \qquad (3)$$

Taking the difference of Equations (2) and (3) yields

$$T[j^*] = A_0[j^*] - A_1[j^*] \approx -\varepsilon \qquad \text{when } k_i = 0 \qquad (4)$$

Similarly, when $k_i = 1$, equations for $A_0[j^*]$ and $A_1[j^*]$ can be written in terms of the expected values of power consumption P

$$A_0[j^*] \approx \mathrm{E}[P \,|\, k_i = 1, p_i = 0] = \mathrm{E}[d\varepsilon + L + n \,|\, k_i = 1, p_i = 0] = \frac{N+1}{2}\varepsilon + L \quad (5)$$

$$A_1[j^*] \approx \mathrm{E}[P \,|\, k_i = 1, p_i = 1] = \mathrm{E}[d\varepsilon + L + n \,|\, k_i = 1, p_i = 1] = \frac{N-1}{2}\varepsilon + L \quad (6)$$

Taking the difference of Equations (6) and (5) yields

$$T[j^*] = A_0[j^*] - A_1[j^*] \approx \varepsilon \qquad \text{when } k_i = 1 \quad (7)$$

So, it is clear from Equations (4) and (7) that there should be a positive bias spike when $k_i = 1$ and a negative bias spike when $k_i = 0$. Thus, Proposition 1 is a sound DPA attack.

But…

- We already knew that W_1 is susceptible to 1st order DPA attacks
- OK, so let us now consider W_2 which has been masked…

```
W2(PTI)
{
  B:  RandomMask = rand()
      mPTI = PTI ⊕ RandomMask
  C:  Result = mPTI ⊕ SecretKey
```

Proposition 2. When the W_2 algorithm is implemented in an N-bit processor, where there is a linear relationship between the instantaneous power consumption and the Hamming weight of the data being processed, the following second-order DPA attack is sound:

1. *Repeat for i equal to* 0 *through* $N-1$ {
2. *Repeat for* $b = 0$ *to* 1 {
3. *Calculate average statistic* $\bar{S}_b = |P_B - P_C|$ *by repeating the following:*{
4. *Set the ith bit of the PTI input to b.*
5. *Set the remaining PTI bits to random values.*
6. *Collect the algorithm's instantaneous power consumption as lines B and C. Call these values* P_B *and* P_C*, respectively.* } }
7. *Calculate the DPA bias statistic* $T = \bar{S}_0 - \bar{S}_1$.
8. *If* $T > 0$ *then the ith key bit is a one, otherwise it is a zero.*

2^{nd} Order DPA

- $P_B = \epsilon_B * d_B + L_B$
- $P_C = \epsilon_C * d_C + L_C$
- For some situations (e.g., a smartcard), we have $\epsilon_B = \epsilon_C$ and $L_B = L_C$
- Let $\epsilon = \epsilon_B = \epsilon_C$
- Then $|P_B - P_C| = \epsilon |d_B - d_C|$

2^{nd} Order DPA (cont'd 1)

```
W2(PTI)
{
→ B: RandomMask = rand()
     mPTI = PTI ⊕ RandomMask
→ C: Result = mPTI ⊕ SecretKey
```

- Let the i^{th} bit of RandomMask be r_i
- Energy consumption at B depends on r_i
- Energy consumption at C depends on $r_i \oplus k_i \oplus p_i$

$$
\begin{aligned}
\mathrm{E}\left[d_B \,|\, r_i = 1\right] &= \mathrm{E}\left[d_C \middle| r_i \oplus k_i \oplus p_i = 1\right] = (N+1)/2 \\
\mathrm{E}\left[d_B \,|\, r_i = 0\right] &= \mathrm{E}\left[d_C \middle| r_i \oplus k_i \oplus p_i = 0\right] = (N-1)/2
\end{aligned}
\tag{10}
$$

$$
\begin{aligned}
\bar{S}_0 &= \frac{1}{2}\mathrm{E}\left[\varepsilon\left|d_B - d_C\right| \,\middle|\, r_i = k_i = p_i = 0\right] + \frac{1}{2}\mathrm{E}\left[\varepsilon\left|d_B - d_C\right| \,\middle|\, r_i = 1,\ k_i = p_i = 0\right] \\
&= 0
\end{aligned}
$$

$$\bar{S}_1 = \frac{1}{2}\mathrm{E}\left[\varepsilon|d_B - d_C| \,\middle|\, p_i = 1,\ r_i = k_i = 0\right] + \frac{1}{2}\mathrm{E}\left[\varepsilon|d_B - d_C| \,\middle|\, r_i = p_i = 1,\ k_i = 0\right]$$

$$= \varepsilon$$

The combination of Equations (11) and (12) yields

$$T = \bar{S}_0 - \bar{S}_1 = -\varepsilon$$