Distinguishing Multiplications from Squaring Operations

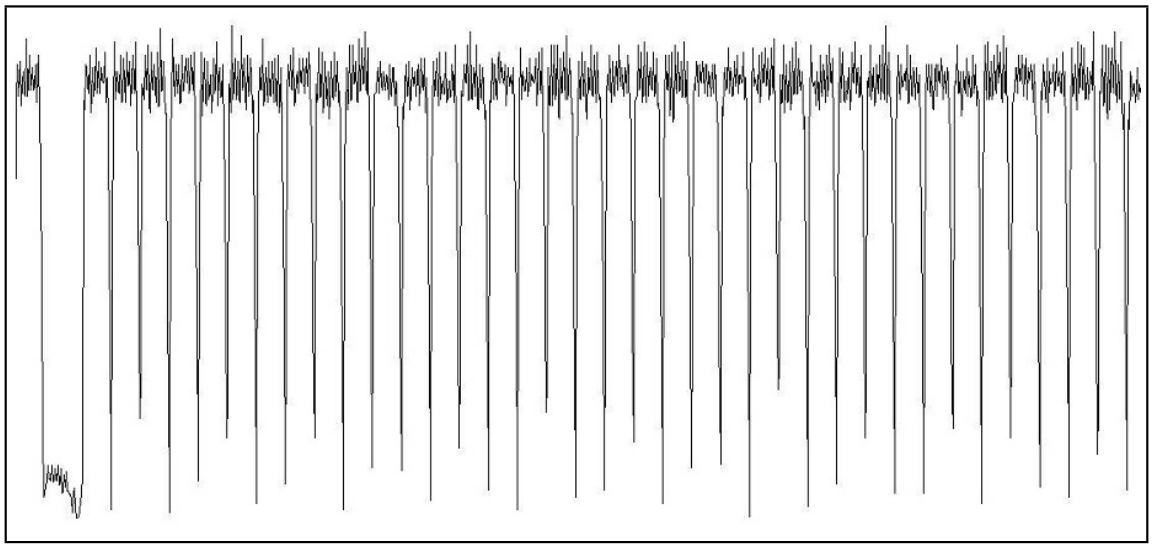
Frederic Amiel Benoit Feix Michael Tunstall Claire Whelan
William P. Marnane

Cork — May 20, 2008

Outline

Side Channel Atomicity

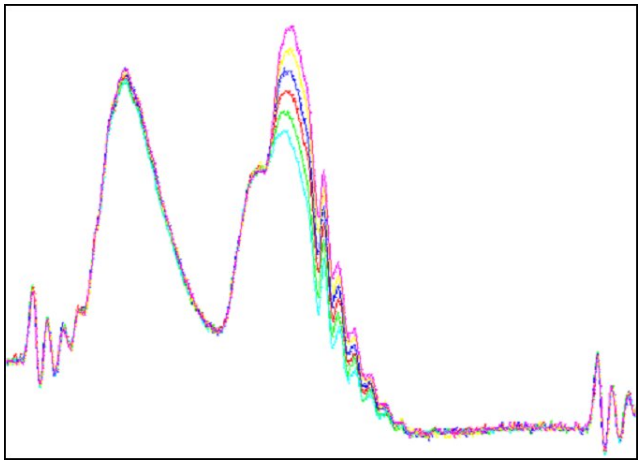
- A countermeasure against being able to distinguish operations is to make the code that is required to execute them identical (referred to as Side Channel Atomicity (Chevallier-Mames et al., 2004)).



- The squaring operation $x^2 \bmod n$ is replaced with $x \cdot x \bmod n$ to render it indistinguishable from a multiplication $x \cdot y \bmod n$ using side channel analysis.
- We present an attack based on the statistically expected Hamming weight of the result of these operations ...

The Hamming Weight

- Looking closely at superposed power consumption traces, small differences can be observed.



- Where the difference is typically either:
  - Proportional to the Hamming weight of the data being manipulated (Hamming weight model).
  - Proportional to the Hamming weight of the data being manipulated XORed with some unknown constant previous state (Hamming distance model).
- In this work we only consider the the Hamming weight model.
  - This is the model most commonly used for attacking microprocessor implementations of cryptographic algorithms.
  - It also applies to some hardware implementations (Amiel et al., 2007).

Differential Power Analysis

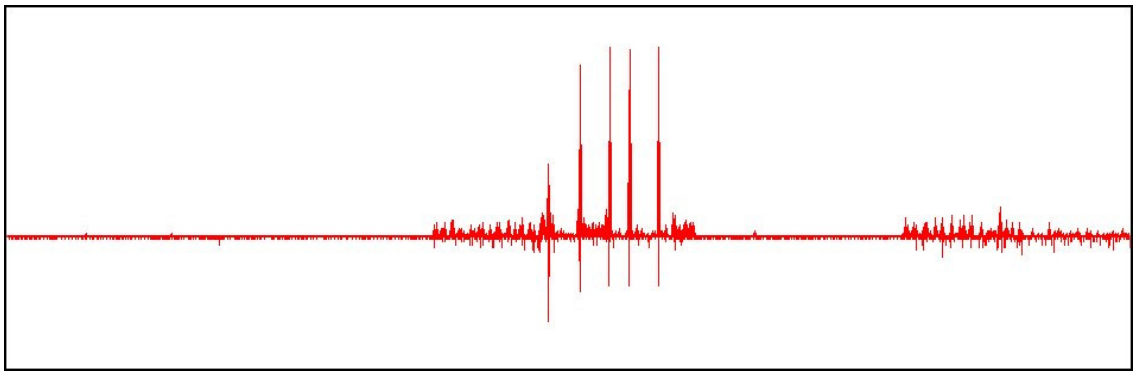
- N power consumption traces are acquired while a device is computing a cryptographic algorithm, with known variable messages.
- A bit b is chosen in some intermediate value, and the value of this bit is predicted for each of the N acquisitions (w_i for $1 \leq i \leq N$).
- The power traces are then divided up into two sets (S_0 and S_1) depending on whether b is equal to zero or one.
- A differential trace Δ_n is calculated by computing an average power consumption trace for each set, and subtracting the resulting traces from each other, i.e.

$$\Delta_n = \frac{\sum_{w_i \in S_0} w_i}{|S_0|} - \frac{\sum_{w_i \in S_1} w_i}{|S_1|}$$

where all the operations are conducted in a pointwise manner.

Differential Power Analysis

- If b is correctly predicted for each acquisition a difference in the two average will occur where bit b is manipulated.
- For example, if we predict one bit of the output the first s-box of DES and generate a corresponding differential trace:

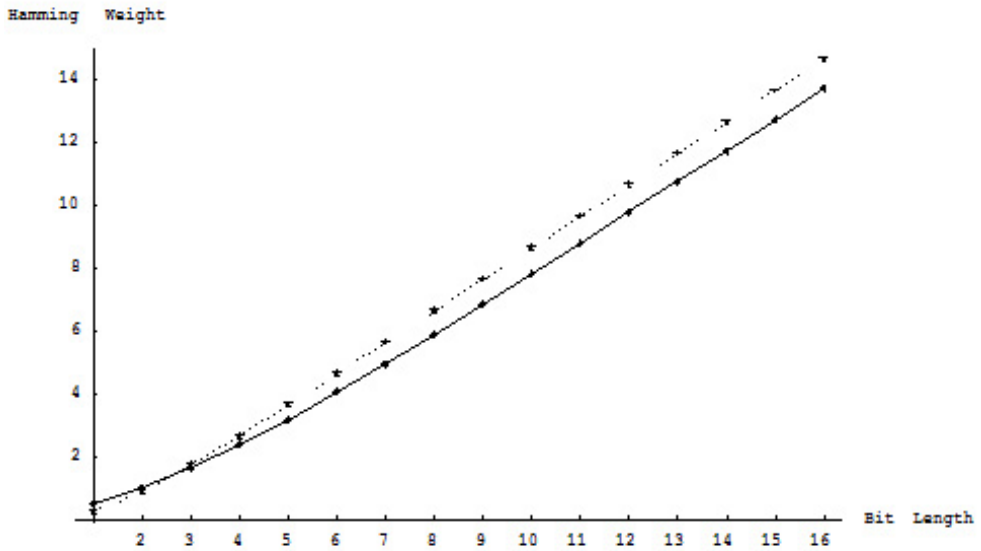


- A difference is visible where the output of the first s-box is generated, and then in four subsequent positions where the nibble conatiaing b is manipulated in the P-permutation.
- This can be used to confirm hypotheses on six bits of the first subkey used, as if these six bits are not known b cannot be predicted.

Outline

The Statistically Expected Difference

- Differential Power Analysis relies on correctly predicting a bit b and using this to confirm hypotheses.
- A similar treatment can be conducted if we consider the statistically expected difference in Hamming weight between the result of two operations.
- For example, if we compute the expected Hamming weight of multiplication and squaring operations for n-bit words ($1 \leq n \leq 16$), assuming random uniformly distributed inputs.



The Statistically Expected Difference

- Why does this occur?
- The probability of the least significant bit being equal to zero is.

	0	1
0	0	-
1	-	1

$\Pr(\text{bit} = 1) = \frac{1}{2}$

	0	1
0	0	0
1	0	1

$\Pr(\text{bit} = 1) = \frac{1}{4}$

- The probability of the second least significant bit being equal to zero is.

	00	01	10	11
00	0	-	-	-
01	-	0	-	-
10	-	-	0	-
11	-	-	-	0

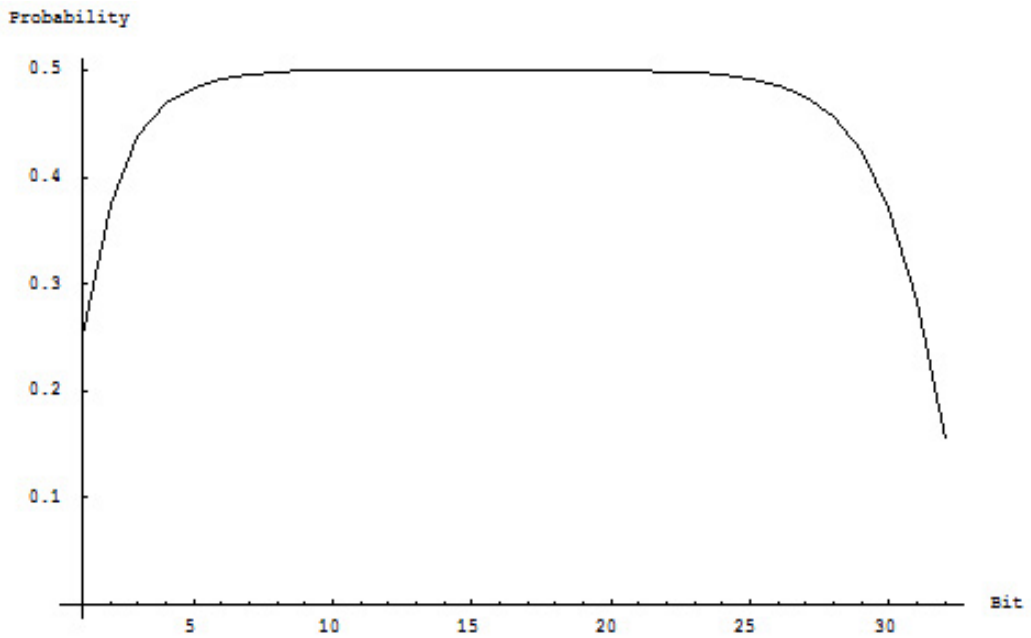
$\Pr(\text{bit} = 1) = 0$

	00	01	10	11
00	0	0	0	0
01	0	0	1	1
10	0	1	0	1
11	0	1	1	0

$\Pr(\text{bit} = 1) = \frac{3}{8}$

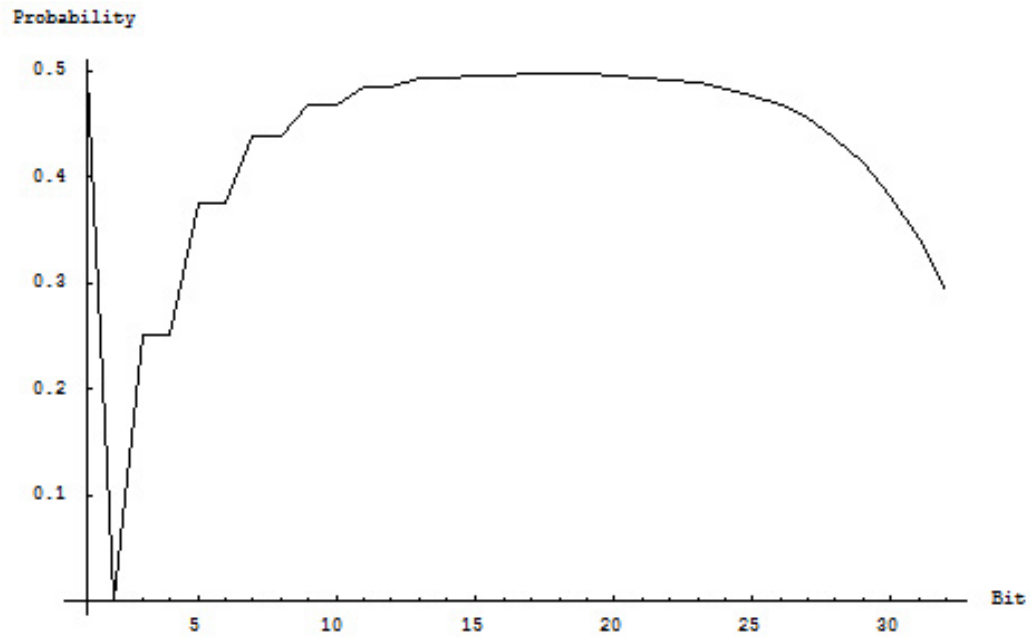
The Probability of Individual Bits Being Set to One

- The probability each bit in a 32-bit word produced by a multiplication of two random uniformly distributed 16-bit words.



The Probability of Individual Bits Being Set to One

- The probability each bit in a 32-bit word produced by a squaring of two random uniformly distributed 16-bit words.



Demonstrating the Difference

- The school book multiplication algorithm was implemented on an ARM7 chip (32-bit architecture).

Algorithm 1: Long Integer Multiplication

Input: $X = (x_{z-1}, \ldots, x_1, x_0)_b$, $Y = (y_{z-1}, \ldots, y_1, y_0)_b$
Output: $W = (w_{2z-1}, \ldots, w_1, w_0)_b = X \cdot Y$

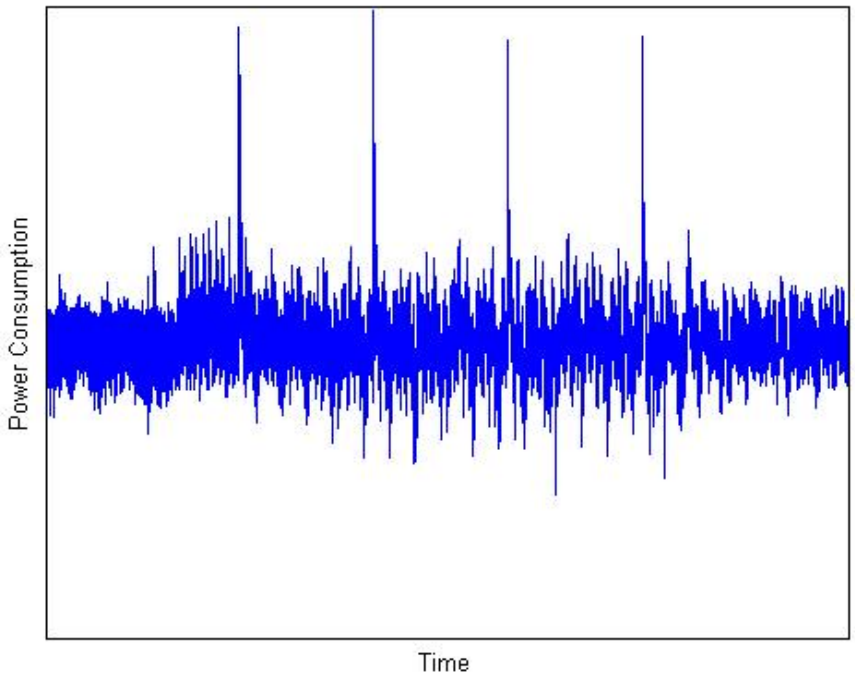
$W \leftarrow 0$
for $i = 0$ **to** $z - 1$ **do**
  $c \leftarrow 0$
  for $j = 0$ **to** $z - 1$ **do**
    $(uv)_b \leftarrow (w_{i+j} + x_j \cdot y_i) + c$
    $w_{i+j} \leftarrow v$; $c \leftarrow u$
  end
  $w_{2z-1} \leftarrow v$
end

return W

- A series of traces were acquired when this implementation was used to compute a multiplication of a squaring operation with random 128-bit inputs.

Demonstrating the Difference

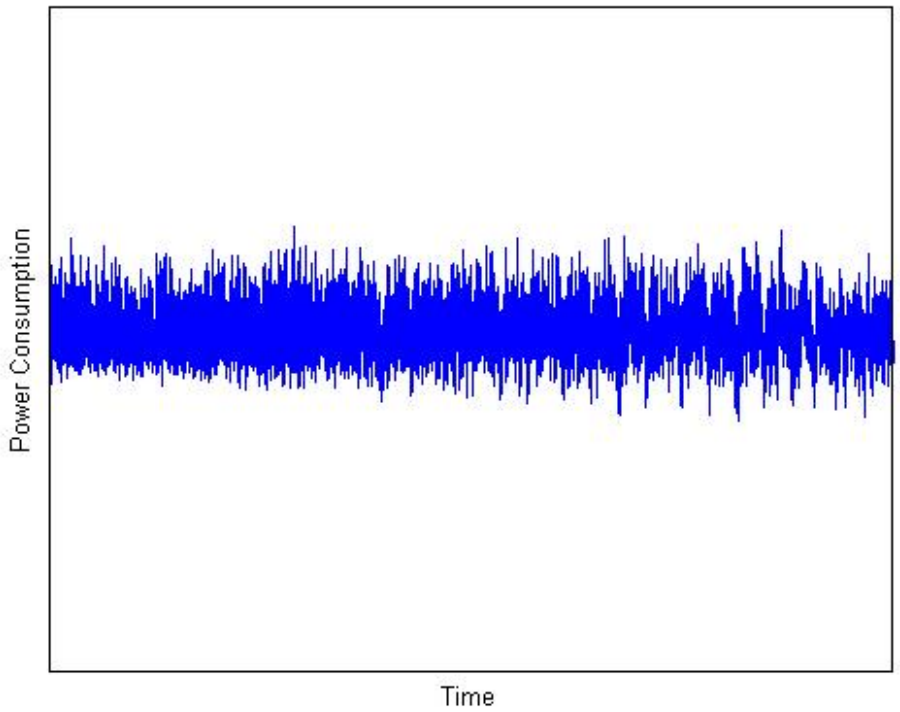
- The difference trace computed by comparing an average traces acquired during the computation of a multiplication and a squaring operation.



- The peaks in the difference correspond to the difference in Hamming weight produced when $x_i \cdot y_i$ is computed when $x = y$.

Demonstrating the Difference

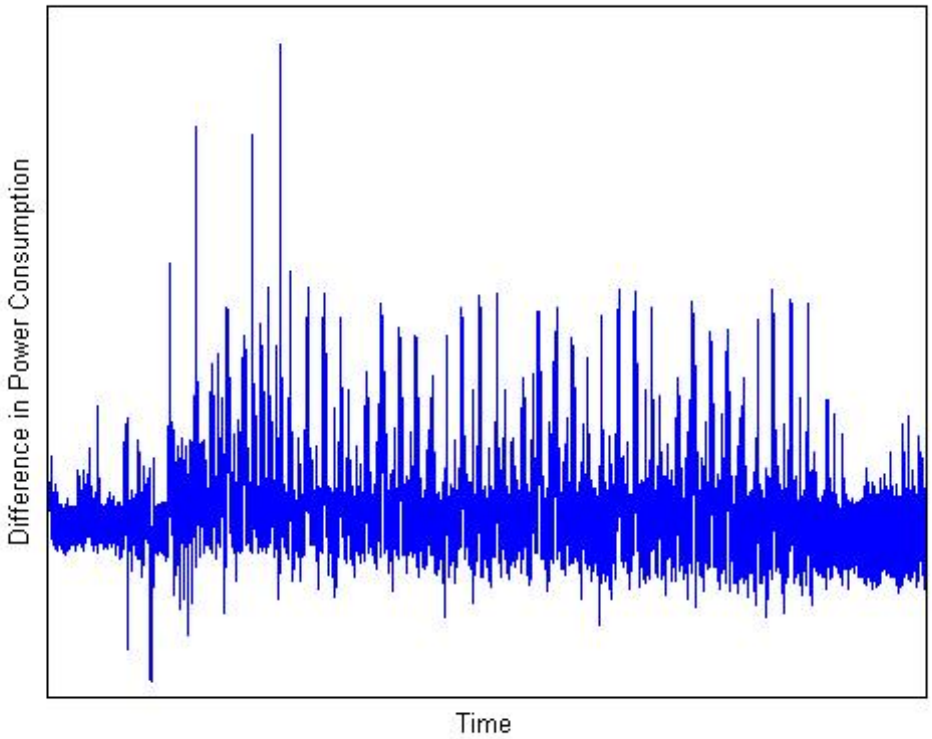
- The difference trace computed by comparing an two average traces acquired during the computation of a squaring operation.



- No peaks in the difference are observed.

Demonstrating the Difference

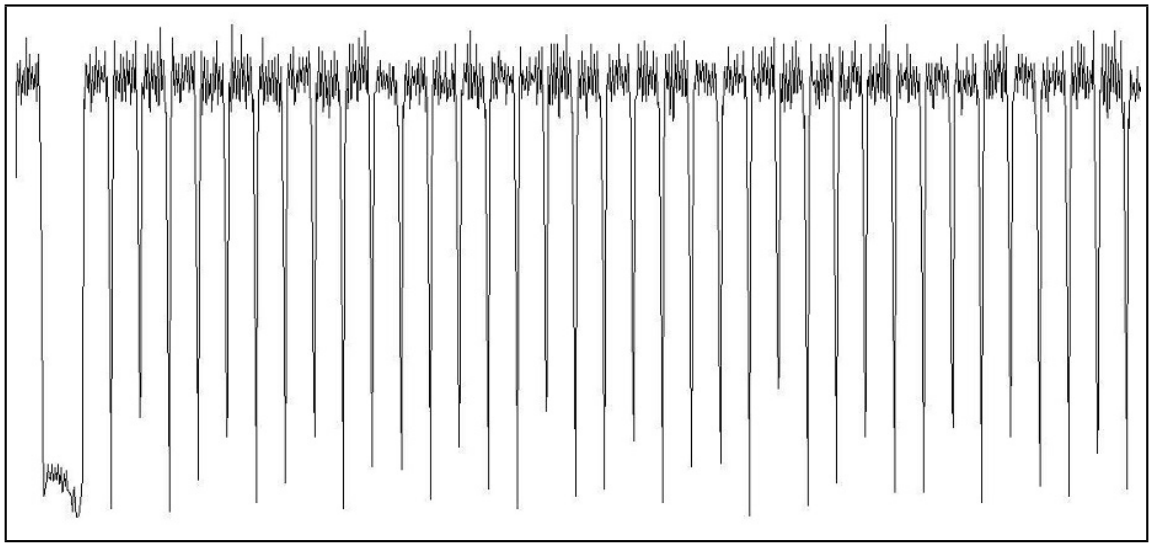
- Similar peaks were visible when the same analysis was conducted on an implementation of Montgomery multiplication.



Outline

Attacking an Exponentiation

- In side channel atomic implementations of a modular exponentiation, computed using the square and multiply algorithm.



- The difference in Hamming weight of adjacent blocks can be compared as described previously to attack algorithms, such as the square and multiply algorithm.
- This results an an attack similar to the Big Mac attack (Walter, 2001).

Application to Elliptic Curve Cryptography

- Side Channel Atomicity had been extended to Elliptic Curve Cryptography, referred to as Unified Addition Formulae (Brier and Joye, 2002), making addition and doubling operations side channel equivalent.
- By manipulating formulae required to compute the slope λ the formula for addition and doubling operations can be unified. For example, the slope calculated during the addition of the points $P = (x_1, y_1)$, $Q = (x_2, y_2)$ is

$$\lambda = \frac{x_1^2 + x_1 x_2 + x_2^2 + a_2 x_1 + a_2 x_2 + a_4 - a_1 y_1}{y_1 + y_2 + a_1 x_2 + a_3} .$$

- If $P = Q$ then $x_1 x_2$ will be a squaring operation, otherwise it will be a multiplication.
- An observable difference will, therefore, occur in the power consumption.

Outline

Blinding

- The operand blinding for modular exponentiation

Algorithm 2: Randomised Exponentiation Algorithm

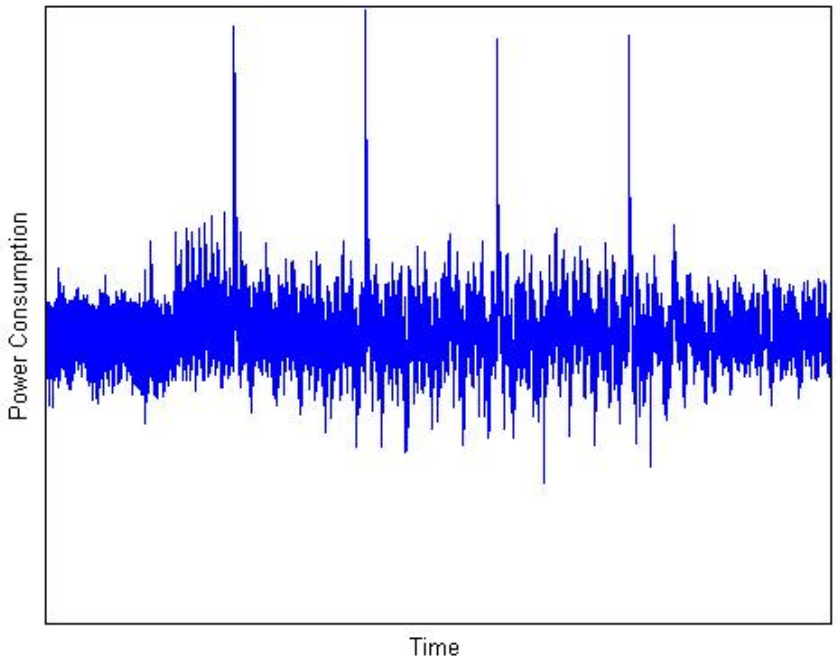
Input: M, d, N, small random values r_1, r_2, r_3
Output: $C = M^d \bmod N$

$M' \leftarrow M + r_1 \cdot N$
$d' \leftarrow d + r_2 \cdot \lambda(N)$
$N' \leftarrow r_3 \cdot N$
$C' \leftarrow M'^{d'} \bmod N'$
$C \leftarrow C' \bmod N$

return C

- The expected difference in the Hamming weight will occur if the message and modulus are blinded, as an attacker does not need to know the message.
- However, it is not possible to produce an average trace if the exponent is blinded.

Blinding



- As observed in (Walter, 2001) it may be possible to combine the points in one trace that show the difference in expected Hamming weight to try and distinguish a multiplication and a squaring operation to overcome exponent blinding.
- This will depend on the key length and the word size of the processor.

Resistant Algorithms

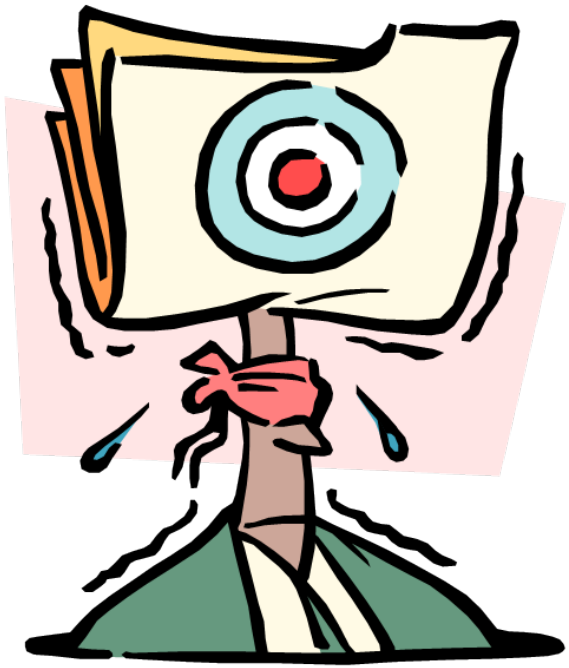
- These attacks will only work on algorithms that do not have a regular structure.
- The attack will not apply to:
  - square and multiply always algorithm.
  - the Montgomery Ladder.
  - the BRIP algorithm.
  - fixed window exponentiation.

Outline

Conclusion

- This work shows that the statistically expected difference in operations computed by a microprocessor can be used to distinguish between a multiplication and a squaring operation.
  - Applies in the presence of message and modulus blinding.
  - Also applies when classical padding schemes are used, as no knowledge of the plaintext is required.
  - Exponent blinding hinders the attack — theoretical attack.
- This is an improvement over previously published results, as the described attack requires no knowledge of the plaintext being manipulated or of the architecture of the multiplier.
- We are currently looking at inexpensive countermeasures, e.g. computing $a \cdot -a \bmod n$ for a squaring operation to change the distribution.

Comments/Questions?



http://www.geocities.com/mike.tunstall/