A Fault Attack on Pairing Based Cryptography

Dan Page (Bristol) and Fré Vercauteren (KU Leuven)

FDTC 2005

Introduction

- Pairing based cryptography is a (fairly) new area:
  - Has provided new instantiations of Identity Based Encryption.
  - Has provided a wealth of new "hard problems" and proof techniques.
  - Has opened a new area for those interested in implementation.
- Like all new ideas, we want to have a good understanding of the security properties:
  - More and more, such properties include resilience to side-channel and fault attack.
  - In reality, it is just fun to try and break things.
- Our goal here is to start looking at fault attacks on the pairing.

Pairing Based Cryptography (1)

- For our purposes, the pairing is just a map between groups:

  $$e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$$

  where we usually set $\mathbb{G}_1 = E(\mathbb{F}_q)$ and $\mathbb{G}_2 = \mathbb{F}_{q^k}$.
- The main interesting property of the map is termed bilinearity:

  $$e(a \cdot P, b \cdot Q) = e(P, Q)^{a \cdot b}$$

  which means we can play about with the exponents at will.
- To work in a useful way, the map also needs to be:
  - Non-degenerate, i.e. not all $e(P, Q) = 1$.
  - Computable, i.e. we can evaluate $e(P, Q)$ easily.
- In real applications we generally use the Tate or Weil pairing.

Pairing Based Cryptography (2)

- Such pairings were originally thought to only be useful in a destructive setting.
- Boneh-Franklin identity based encryption is perhaps the most interesting constructive use:
  - The trust authority or TA has a public key $P_{TA} = s \cdot P$ for a public value P and secret value s.
  - A users public key is calculated from the string ID using a hash function as $P_{ID} = H_1(ID)$.
  - A users secret key is calculated by the TA as $S_{ID} = s \cdot P_{ID}$.
- To encrypt M, select a random r and compute the tuple:

$$C = (r \cdot P, M \oplus H_2(e(P_{ID}, P_{TA})^r)).$$

- To decrypt $C = (U, V)$, we compute the result:

$$M = V \oplus H_2(e(S_{ID}, U)).$$

Pairing Based Cryptography (3)

- We are interested in the case where $q = 3^m$ and $k = 6$ since this is attractive from a parameterisation perspective.
- Along with the standard Miller-style BKLS algorithm, there are two closed-form algorithms in this case.
- Both compute $e(P, Q)$ with $P = (x_1, y_1)$ and $Q = (x_2, y_2)$.

The Duursma-Lee Algorithm

$f \leftarrow 1$
for $i = 1$ **upto** m **do**
    $x_1 \leftarrow x_1^3$
    $y_1 \leftarrow y_1^3$
    $\mu \leftarrow x_1 + x_2 + b$
    $\lambda \leftarrow -y_1 y_2 \sigma - \mu^2$
    $g \leftarrow \lambda - \mu\rho - \rho^2$
    $f \leftarrow f \cdot g$
    $x_2 \leftarrow x_2^{1/3}$
    $y_2 \leftarrow y_2^{1/3}$
return $f^{q^3 - 1}$

The Kwon-BGOS Algorithm

$f \leftarrow 1$
$x_2 \leftarrow x_2^3$
$y_2 \leftarrow y_2^3$
$d \leftarrow mb$
for $i = 1$ **upto** m **do**
    $x_1 \leftarrow x_1^9$
    $y_1 \leftarrow y_1^9$
    $\mu \leftarrow x_1 + x_2 + d$
    $\lambda \leftarrow y_1 y_2 \sigma - \mu^2$
    $g \leftarrow \lambda - \mu\rho - \rho^2$
    $f \leftarrow f^3 \cdot g$
    $y_2 \leftarrow -y_2$
    $d \leftarrow d - b$
return $f^{q^3 - 1}$

The Fault Attack (1)

- Goal: given the result $R = e(P, Q)$ and knowledge of Q, find P.
- To make things easier, assume we use the Duursma-Lee algorithm and can reverse the final powering by $q^3 - 1$.
- Let e_Δ denote the pairing where we replace the loop bound m with Δ so instead of producing the product:

$$\prod_{i=1}^{m} \left[(-y_1^{3^i} y_2^{1/3^{i-1}} \sigma - (x_1^{3^i} + x_2^{1/3^{i-1}} + b)^2) - (x_1^{3^i} + x_2^{1/3^{i-1}} + b)\rho - \rho^2 \right]$$

the instead produces:

$$\prod_{i=1}^{\Delta} \left[(-y_1^{3^i} y_2^{1/3^{i-1}} \sigma - (x_1^{3^i} + x_2^{1/3^{i-1}} + b)^2) - (x_1^{3^i} + x_2^{1/3^{i-1}} + b)\rho - \rho^2 \right]$$

- If we can force the device to compute $R_1 = e_{m \pm r+0}(P, Q)$ and $R_2 = e_{m \pm r+1}(P, Q)$ by provoking some random error r, then $T = R_1 / R_2$ gives just one factor of the product.

The Fault Attack (2)

- With just one factor, we can extract recover x_1 and y_1 given knowledge of x_2, y_2, r and b:
  - We make the target device to lots of pairings and provoke random errors in the value of m to get $m \pm r$.
  - Using a passive timing attack, we can tell how many loop iterations are done and hence what r was.
  - A usable pair of $m \pm r + 0$ and $m \pm r + 1$ will come along after not too many attempts due to a similar argument as the birthday paradox.
  - Finally, we use the collected results to recover the secret point.
- Boneh-Franklin survives this attack because it doesn't allow the attacker to get direct access to pairing results, other schemes are less secure ...

The Fault Attack (3)

- So far, we side-stepped the problem of reversing the final powering:
  - We assumed we compute $T = R_1/R_2$ but actually we get T^{q^3-1}.
- Lidl and Niederreiter describe a method to compute roots of $X^{q^3} - T = 0$ which they call a q-polynomial.
- We have $X^{q^3-1} - T = 0$ so we just multiply by X to get $X^{q^3} - TX = 0$.
- Then we just use their text-book method:
  - Write $X = x_0 + \sigma x_1$ and $T = t_0 + \sigma t_1$ with $x_0, x_1, t_0, t_1 \in \mathbb{F}_{q^3}$.
  - The above equation is equivalent to

$$M \cdot X = \begin{pmatrix} 1 - t_0 & t_1 \\ t_1 & 1 + t_0 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} = 0.$$

  - The kernel of M then provides all solutions to $X^{q^3-1} - R = 0$.

The Fault Attack (4)

- The problem now is that there are $q^3 - 1$ possible roots and we want to find one specific root !
- We are saved from failure because factors from the Duursma-Lee algorithm have a sparse form:

$$T = t_0 + t_1\rho - \rho^2 + t_2\sigma$$

where there are no $\rho\sigma$ or $\rho^2\sigma$ coefficients.
- From the root finding algorithm we get $T' = c \cdot T$ for some $c \in \mathbb{F}_{q^3}$.
- The goal is to compute $d = c^{-1} = T/T'$ and hence T which boils down to solving:

$$\begin{pmatrix} t'_1 & t'_0 + t'_1 \\ t'_2 & t'_1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \end{pmatrix} = \begin{pmatrix} t'_1 + t'_2 \\ t'_0 + t'_2 \end{pmatrix}.$$

The Fault Attack (5)

- All is not lost, we can use bilinearity to try and defend against the attack by denying the attacker knowledge of x_2 and y_2:
  - Pick random integers a and b so that $a \cdot b = 1 \pmod{\#\mathbb{G}_1}$.
  - Take our P and Q and compute $P' = a \cdot P$ and $Q' = b \cdot Q$.
  - Now calculate the pairing as:

$$e(P', Q') = e(a \cdot P, b \cdot Q) = e(P, Q)^{a \cdot b} = e(P, Q).$$

- The difference is, now the values going into the pairing are randomised: trying to apply the attack yields random stuff rather than the required value.
- Software defences like this are probably preferable to changing the hardware since this is costly and hard to get right.

Conclusion

- This is quite a nice but fairly trivial attack on pairing based cryptography.
  - In reality, unless the protocol is badly designed the attack is probably unrealistic.
- However, this is a new topic and there are plenty of interesting open problems to think about:
  - What happens if P or Q are not on any curve ?
  - What happens if P or Q are not on the expected curve ?
  - What happens if $\mathbb{F}_q$ is faulty ?
  - How can one attack the BKLS algorithm rather than the closed form versions ?
- The pairing is quite resilient to most things we can think of:
  - The final powering mops up dodgy outputs and forces the pairing to be degenerate.
  - Maybe the answer is to attack protocols rather than the pairing itself ...