The Hidden Root Problem

F. Vercauteren*

Department of Electrical Engineering, University of Leuven
Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium
frederik.vercauteren@esat.kuleuven.be

Abstract. In this paper we study a novel computational problem called the *Hidden Root Problem*, which appears naturally when considering fault attacks on pairing based cryptosystems. Furthermore, a variant of this problem is one of the main obstacles for efficient pairing inversion. We present an algorithm to solve this problem over extension fields and investigate for which parameters the algorithm becomes practical. [1]

Keywords: finite fields, subgroups, hidden root problem, pairing inversion

1 Introduction

All known public key cryptosystems are based on a presumed hard mathematical problem, such as problems related to discrete logarithms [6], factoring [13] or finding short vectors in a lattice [11]. The security of the cryptographic protocol should ideally be implied directly by the hardness of the mathematical problem and the precise relation should be captured in a proof of security.

Since the inception of pairing based cryptography, a plethora of new supposedly hard problems has been introduced, but the main hard problem undoubtedly is pairing inversion [7], which has far-reaching implications [15, 16]. In this paper we formally define a new computational problem called the Hidden Root Problem (HRP), which arises naturally as a cryptanalytic problem when studying pairing inversion [7] and side-channel or fault attacks on pairing implementations [10, 17, 18]. The problem resembles the well-known Hidden Number Problem [2, 3, 14], but is of a more algebraic nature, since the information returned about the hidden root consist of a projection into a subgroup of the multiplicative group of a finite field.

The remainder of this paper is organized as follows: Section 2 formally defines several variants of the HRP and points out the similarities with the Hidden Number Problem. Section 3 provides the motivation for studying this problem

* Postdoctoral Fellow of the Research Foundation - Flanders (FWO)

[1] The work described in this paper has been supported in part by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT. The information in this document reflects only the author's views, is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability

and in Section 4 we describe an algorithm to solve the HRP over extension fields. In Section 5 we study the practicality of this algorithm for the linear version of the HRP by considering projection onto algebraic tori. Finally, Section 6 concludes the paper.

2 The Hidden Root Problem

In this section we define various versions of the Hidden Root Problem (HRP) over finite fields. We assume that the oracles appearing in the definitions are perfect, i.e. they always return the correct result. The following problem was first formulated in [7].

Definition 1 (Linear Hidden Root Problem). *Let $\mathbb{F}_q$ be a finite field with $q = p^n$ elements, where p is prime and let e be a positive integer with $e|(q-1)$. Let $\mathcal{O}_x(\cdot,\cdot)$ denote an oracle that on input $(a,b) \in \mathbb{F}_q^2$ returns*

$$\xi_{a,b} = (ax+b)^e$$

for a fixed secret $x \in \mathbb{F}_q$. The Linear Hidden Root Problem (LHRP) is to recover x in expected polynomial time in $\log q$ by querying the oracle repeatedly with chosen pairs (a_i, b_i).

The restriction $e|(q-1)$ can be explained as follows: let e' be a positive integer with $\gcd(e', q-1) = 1$, then e'-powering defines a permutation on $\mathbb{F}_q$ with inverse $(e'^{-1} \bmod (q-1))$-powering. The LHRP formulated for e' therefore is trivial to solve using only one query by computing

$$x = (\xi_{a,b}^{e'^{-1} \bmod (q-1)} - b)/a\,.$$

Similarly, if $d = \gcd(e', q-1)$, then (e'/d)-powering is a permutation which can easily be inverted, so the problem reduces to the LHRP with $e = d$ and thus $e|(q-1)$. In the remainder of this paper we will use the notation $h = (q-1)/e$, i.e. the cofactor of e.

Note that in the definition of the LHRP, the querying party can choose the pairs (a_i, b_i) himself. It would be possible to define a randomized version, where we are simply given a list of random pairs (a_i, b_i) with the corresponding responses $\mathcal{O}_x(a_i, b_i) = \xi_{a_i,b_i}$. Clearly, this randomized version cannot be easier than the LHRP as defined above, but the algorithms described in Section 4 work equally well in the randomized case. In fact, most cryptanalytic applications correspond to the randomized version, since the adversary does not control a, b directly. For instance, a, b could be related to the coordinates of a point R on an elliptic curve with $R = [m]P$, where the adversary is given P and can only choose m.

Obvious generalizations would be to allow degree d polynomials in x instead of a linear polynomial, or in fact any function in x of specified form, such as fractions of polynomials. Further, one can also replace x by any unknown quantity such as a vector of unknowns. This brings us to the following more general definition.

Definition 2 (Hidden Root Problem). *Let $\mathbb{F}_q$ be a finite field with $q = p^n$ elements, where p is prime and let e be a positive integer with $e|(q-1)$. Let $f_x(\cdot) : \mathcal{D}(\mathbb{F}_q) \to \mathbb{F}_q$ be a specified map, i.e. the precise description of which is given, depending on x from a domain $\mathcal{D}$ to $\mathbb{F}_q$. Let $\mathcal{O}_x(\cdot)$ denote an oracle that on input $\alpha \in \mathcal{D}$ returns*

$$\xi_\alpha = f_x(\alpha)^e$$

for a fixed secret $x \in \mathbb{F}_q$. The Hidden Root Problem is to recover x in expected polynomial time in $\log q$ by querying the oracle repeatedly for chosen $\alpha_i \in \mathcal{D}(\mathbb{F}_q)$.

An even further generalization would be to replace the multiplicative group $\mathbb{F}_q^*$ by any group G and the e-th powering by a projection into a proper subgroup of G.

The name "Hidden Root Problem" was chosen to point out the resemblance with the Hidden Number Problem [2, 3] and its generalizations as described in [14]. Recall that the $\mathbb{F}_p$-Hidden Number Problem (HNP) is defined as: for a secret $x \in \mathbb{F}_p$, we are given the k pairs

$$(t_i, MSB_{l,p}(xt_i)) \qquad i = 1, \ldots, k$$

for k elements $t_1, \ldots, t_k \in \mathbb{F}_p^*$, chosen independently and uniformly at random, and for some $l > 0$, where $MSB_{l,p}$ denotes (roughly speaking) the l most significant bits. The problem then is to recover x.

Note that in the LHRP, for each query we obtain roughly $\log_2 h$ bits of information about x, so in this sense the HRP is similar to the HNP. Heuristically, we therefore expect a unique solution to the LHRP after roughly $\log_h q$ queries. The main difference with the HNP are the following two facts: firstly, the HNP is randomized, i.e. the querying party cannot choose the t_i for $i = 1, \ldots, k$ and secondly, hiding the information about x in the HRP corresponds to an algebraic operation, namely e-th powering.

Finally, a last version of the HRP is the **subfield HRP**, i.e. the HRP with the restriction that the secret x lies in a strict subfield of $\mathbb{F}_{p^d} \subsetneq \mathbb{F}_q$ for $d|n$.

3 Applications in Pairing Based Cryptography

The main motivation for our study of the Hidden Root Problem is undoubtedly cryptanalysis of pairings, more specifically, side-channel and fault attacks on pairings [10, 17, 18] and pairing inversion [7].

Recall that the general setting of pairings is the following: let E be an elliptic curve over a finite field $\mathbb{F}_q$ and let r be a large prime with $\gcd(r, q) = 1$ and $r \mid \#E(\mathbb{F}_q)$. By definition, the embedding degree k is the smallest positive integer with $r \mid (q^k - 1)$. All variants of the Tate pairing can then be described as functions of the form

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mu_r \subset \mathbb{F}_{q^k} : (P, Q) \mapsto e(P, Q) = f_{S,P}(Q)^L\,, \tag{1}$$

where $\mathbb{G}_1, \mathbb{G}_2$ are given cyclic subgroups of $E(\mathbb{F}_{q^k})[r]$, $f_{S,P}$ is a Miller function, i.e. has divisor $S(P) - ([S]P) - (S-1)(\mathcal{O})$ and $L|(q^k - 1)$. Often, $L = (q^k - 1)/r$

with $r \mid \Phi_k(q)$, where Φ_k is the k-th cyclotomic polynomial. However, to speed up the final exponentiation, L is sometimes taken to have very low Hamming weight in base q, by moving some of the complexity into the scalar S (see for instance [1]). In most protocols, one of the input values to the pairing is public, e.g. the point P and so is the pairing value. The security of the protocol then relies on the inability of the adversary to recover the other input point Q.

3.1 Fault Attacks on Pairings

The function $f_{S,P}$ is computed using Miller's algorithm [9] and thus consists of a product of roughly $\log_2 S$ powers of evaluations of lines appearing in the scalar multiplication of P by S. Algorithm 1 below gives the pseudo-code of Miller's algorithm: $l_{T,P}$ denotes the line through the points T and P and v_T denotes the vertical line through T.

Algorithm 1 Miller's algorithm for elliptic curves

Inputs: $S \in \mathbb{N}$, $P, Q \in E[r]$
Outputs: $f_{S,P}(Q)$
  Write $s = \sum_{j=0}^{t} s_j 2^j$, with $s_j \in \{0, 1\}$ and $s_t = 1$.
  $T \leftarrow P$, $f \leftarrow 1$.
  for $j = t - 1$ down to 0 **do**
    $f \leftarrow c^2 \cdot l_{T,T}(Q)/v_{[2]T}(Q)$.
    $T \leftarrow [2]T$
    if $s_j = 1$ **then**
      $f \leftarrow f \cdot l_{T,P}(Q)/v_{T \oplus P}(Q)$.
      $T \leftarrow T \oplus P$
    end if
  end for
  Return f.

The parameter S typically is public knowledge, since it's either equal or related to the group order r. In a much simplified setting (see [10, 17, 18] for more realistic attacks) we could mount a fault attack on S resulting in a bit-flip of the least significant bit of S. As such the adversary will have access to two pairing values: the correct value $f_{S,P}(Q)^L$ and a faulted one $f_{S \oplus 1,P}(Q)^L$. Note that all steps in Miller's algorithm, except the last, will be exactly the same in the computation of both values. By dividing both values the adversary will know the value of (assuming S is odd)

$$\xi_P = \left(\frac{l_{[S-1]P,P}(Q)}{v_{[S]P}(Q)} \right)^L . \tag{2}$$

Furthermore, in most cases we can ignore the last vertical line $v_{[S]P}(Q)$, either due to denominator elimination or simply because $[S]P = \mathcal{O}$. By repeating the attack (for different known P), several equations of the form (2) can be gathered

and the adversary is left to solve an instance of the HRP (and in many cases the LHRP) with exponent L over $\mathbb{F}_{q^k}$.

3.2 Pairing Inversion

In [7], several approaches to invert the pairing function (1) itself were described. One of these approaches consists of solving two problems: firstly, inverting the final exponentiation and secondly, Miller inversion, i.e. given the evaluation $f_{S,P}(Q)$ and the point P, recover Q. Furthermore, it was shown that for some instances of the ate pairing [8], Miller inversion can be achieved in polynomial time and the security relies entirely on the final exponentiation. In these cases, the function $f_{S,P}$ is of low degree and to invert the final exponentiation, the adversary has to solve an instance of the HRP. Note that due to bilinearity of the pairing, the adversary can easily generate many equations from one given equation $z = f_{S,P}(Q)^L$ by the following simple rule

$$z^i = f_{S,[i]P}(Q)^L \,.$$

Finally, we note that in case of the ate pairing, the point Q is defined over the field $\mathbb{F}_q$, and pairing inversion corresponds to the subfield HRP.

4 An Algorithm for the HRP over Extension Fields

In this section we devise an algorithm to solve the HRP over extension fields $\mathbb{F}_{q^k}$ by combining Weil restriction and linearity of q-th powering. For simplicity we will focus on the LHRP, but the same technique applies to any algebraic function f_x. However, the feasibility of the algorithm will very much depend on the degree in x of f_x (see the paragraph "Complexity for HRP").

Version 0. Let $e|(q^k - 1)$ and write

$$e = \sum_{i=0}^{k-1} c_i q^i - \sum_{i=0}^{k-1} d_i q^i \,, \tag{3}$$

where $c_i, d_i \in \mathbb{N}_{\geq 0}$. Clearly there are many tuples c_i, d_i that give a valid expression for e, but we are only interested in those c_i, d_i that minimize the quantity

$$D_e := \max \left\{ \sum_{i=0}^{k-1} c_i, \ \sum_{i=0}^{k-1} d_i \right\} . \tag{4}$$

Consider $\mathbb{F}_{q^k}$ as a degree k extension over $\mathbb{F}_q$, i.e. $\mathbb{F}_{q^k} = \mathbb{F}_q[\theta]/(f(\theta))$ where $f \in \mathbb{F}_q[x]$ is an irreducible polynomial of degree k. In the LHRP we are given an equation $\xi_{a,b} = (ax+b)^e$ for some pair (a, b) and unknown x. By Weil restriction

we will consider this as k equations over $\mathbb{F}_q$ in k unknowns. More precisely, consider the map

$$\psi : \mathbb{F}_{q^k} \to (\mathbb{F}_q)^k : \alpha = \sum_{i=0}^{k-1} \alpha_i \theta^i \mapsto [\alpha_0, \ldots, \alpha_{k-1}]$$

and note that q-th powering is a linear operation, i.e. there exists an easily computable $k \times k$ matrix F such that $\psi(\alpha^q) = F \cdot \psi(\alpha)$. By substituting the expression (3) for e, we obtain

$$\xi_{a,b} = \frac{(ax+b)^{\sum_{i=0}^{k-1} c_i q^i}}{(ax+b)^{\sum_{i=0}^{k-1} d_i q^i}} = \frac{\prod_{i=0}^{k-1} (ax+b)^{c_i q^i}}{\prod_{i=0}^{k-1} (ax+b)^{d_i q^i}} .$$

Exploiting linearity of q^i-th powering, the above equation can be rewritten as

$$\prod_{i=0}^{k-1} \left(a^{q^i} x^{q^i} + b^{q^i} \right)^{c_i} = \xi_{a,b} \prod_{i=0}^{k-1} \left(a^{q^i} x^{q^i} + b^{q^i} \right)^{d_i} . \tag{5}$$

Finally, apply ψ to both sides of this equation to obtain k non-linear equations over $\mathbb{F}_q$ in the unknowns $x_0, \ldots, x_{k-1}$. Furthermore, since $\psi(x^{q^i}) = F^i \psi(x)$, each of the factors in the product is linear in the unknowns x_i, so the degree of the non-linear system of equations is precisely D_e defined in (4). Solving a system of non-linear equations over finite fields is in general a very hard problem. A notable exception however is when the system is highly overdetermined, which can be easily obtained by repeatedly querying the oracle. In the latter case, Groebner basis techniques [4] are rather efficient or one could resort to relinearization [5]. For both algorithms, the complexity is given by the time to solve a linear system of equations of dimension equal to the total number of monomials of degree less than or equal to D_e in k variables, which is given by

$$M_e := \binom{D_e + k}{D_e} .$$

The complexity of both algorithms then is $O(M_e^\omega)$ with $\omega \leq 3$ the matrix multiplication exponent. Since k is given, minimizing D_e is crucial, since only for reasonably sized M_e and thus very small D_e, will it be possible to even write down the system of non-linear equations. The experiments in Section 5 show that for the algorithm to succeed we need D_e to be very small (depending on k), e.g. at most 4 for $k = 30$. This implies that for fixed k and growing q, version 0 of the algorithm will only be efficient for a constant number of exponents. This situation will be much improved in versions 1 and 2.

Complexity for HRP. Version 0 (and also 1 and 2) also works for any algebraic function f_x. Assume that $f_x = h(x)/g(x)$ with $h, g \in \mathbb{F}_{q^k}[x]$, then the equivalent of equation (5) simply is:

$$\prod_{i=0}^{k-1} \left(h(x)^{q^i} \right)^{c_i} \cdot \prod_{i=0}^{k-1} \left(g(x)^{q^i} \right)^{d_i} = \xi_{a,b} \prod_{i=0}^{k-1} \left(h(x)^{q^i} \right)^{d_i} \cdot \prod_{i=0}^{k-1} \left(g(x)^{q^i} \right)^{c_i} .$$

By applying ψ to both sides of this equation, we again obtain k non-linear equations over $\mathbb{F}_q$ in the unknowns $x_0, \ldots, x_{k-1}$. The main difference however is the degree of these non-linear equations, namely

$$D_{f,e} := \max \left\{ \deg h \sum_{i=0}^{k-1} c_i + \deg g \sum_{i=0}^{k-1} d_i, \deg h \sum_{i=0}^{k-1} d_i + \deg g \sum_{i=0}^{k-1} c_i \right\}.$$

If $\deg h = \deg g = d$, we conclude that $dD_e \leq D_{f,e} \leq 2dD_e$, so the algorithm will only be efficient for f_x of very low degree.

Version 1. The main idea of version 1 is to drastically increase the applicability of version 0 of the algorithm by considering multiples of e. Indeed, each equation $\xi_{a,b} = (ax+b)^e$ gives rise to many other equations $\xi_{a,b}^m = (ax+b)^{me}$ for $m \in \mathbb{Z}$ and as long as $me \neq 0 \bmod (q^k - 1)$, these equations will be non-trivial. Note that by raising to the power m we are in fact ignoring information contained in the original equation. This is not a problem since we can query the oracle to obtain sufficient equations. The main advantage however is that a random looking exponent can be transformed in one with much more algebraic structure. The goal therefore is to find a multiple me of e with D_{me} as small as possible.

A first method to find good multiples of e is to exploit the algebraic factorization $x^k - 1 = \prod_{d|k} \Phi_d(x)$ with $\Phi_d \in \mathbb{Z}[x]$ the d-th cyclotomic polynomial. Since $e|(q^k - 1)$, we can determine an index set I and a polynomial $\Pi(x) := \prod_{d|k, d \in I} \Phi_d(x)$ such that $e|\Pi(q)$ and $\Pi(x)$ has low D (i.e. sum of positive coefficients minus the sum of the negative coefficients). If $\Pi(x) \neq (x^k - 1)$, we have found a good multiple $\Pi(q)$ of e.

Version 2. Although version 1 gives reasonable results for a wide variety of e, it is limited to finding multiples me of e that also divide $(q^k - 1)$. Version 2 relies on LLL [12] to automatically find the best multiple possible. Consider the lattice $L \subset \mathbb{Z}^k$ spanned by the vectors

$$L := \begin{pmatrix} e & 0 & 0 & \cdots & 0 \\ -q & 1 & 0 & \cdots & 0 \\ -q^2 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \ddots & \\ -q^{k-1} & 0 & \ldots & 0 & 1 \end{pmatrix}.$$

Note the inner product of all vectors in the lattice with the vector $[q^0, \ldots, q^{k-1}]$ is a multiple of e. By reducing the lattice L, we find a short vector $[s_0, \cdots, s_{k-1}]$ with $e|\sum_{i=0}^{k-1} s_i q^i$. Note that short vectors automatically have small D.

Subfield HRP. In case of the subfield HRP, we have the extra information that $x \in \mathbb{F}_q$, so all unknowns $x_i = 0$ for $i > 0$. The system of non-linear equations thus simplifies to a system of k univariate polynomials in $x = x_0$ of degree D_e (or

$D_{f,e}$ in case of an algebraic function f_x). To find the solution x, it suffices to take the GCD of several of these polynomials, each of which takes $O(D^2)$ operations in $\mathbb{F}_q$. The subfield HRP therefore is fundamentally easier than the full HRP, since its complexity is polynomial in the parameter D and the dependence on k only appears in the first phase, i.e. writing down the system of univariate polynomials.

5 LHRP and Projection onto Algebraic Tori

In this section we analyze the effectiveness of the various versions of the algorithm given in Section 4 in the special case of LHRP, where the e-th powering equals projection onto the algebraic torus $T_k(\mathbb{F}_q)$. The main reason to consider this special case is that $T_k(\mathbb{F}_q)$ is the biggest subgroup which is really contained in $\mathbb{F}_{q^k}$ itself and not in a strict subfield. Furthermore, all known pairings map into (a strict subgroup of) $T_k(\mathbb{F}_q)$, so the torus $T_k(\mathbb{F}_q)$ can be considered as the base case.

Recall that by definition we have

$$T_k(\mathbb{F}_q) = \{\alpha \in \mathbb{F}_{q^k} \mid N_{\mathbb{F}_{q^k}/\mathbb{F}_{q^d}}(\alpha) = 1 \text{ for all } d|k, d < k\},$$

and $|T_k(\mathbb{F}_q)| = \Phi_k(q)$ with Φ_k the k-th cyclotomic polynomial. Therefore, we choose the e in the LHRP equal to

$$e = (q^k - 1)/\Phi_k(q).$$

This implies that we can take $\Pi(x)$ in version 1 of the algorithm to be equal to $(x^k - 1)/\Phi_k(x)$. To investigate the degree D of the resulting non-linear system, we prove the following lemma.

Lemma 1. *Assume k has prime factorization $k = \prod_{i=1}^{t} p_i^{s_i}$ with $p_i \neq p_j$ for $i \neq j$ and $s_i > 0$. Define $\hat{k} = \prod_{i=1}^{t} p_i$, then*

$$\Phi_k(x) = \Phi_{\hat{k}}(x^{k/\hat{k}}).$$

Proof. This follows immediately from the explicit expression

$$\Phi_k(x) = \prod_{d|k} (x^d - 1)^{\mu(k/d)},$$

with μ the Möbius function: for $n \in \mathbb{N}_{>0}$, $\mu(n) = 0$ if n is not squarefree and $(-1)^k$ if n is the product of k distinct primes. Note $\mu(k/d) = 0$ except for d that are multiples of $k/\hat{k}$, i.e. d is of the form $d = v \cdot (k/\hat{k})$ for $v|\hat{k}$.

Corollary 1. *Using the notation of Lemma 1 we have $\Pi_k(x) = \Pi_{\hat{k}}(x^{k/\hat{k}})$, where $\Pi_k(x) = (x^k - 1)/\Phi_k(x)$.*

The above corollary implies that the degree D of the non-linear system only depends on $\hat{k}$ and not on k itself. A further analysis gives the following refinement: if $\hat{k} = p$, a prime, then $D = 1$ and thus the system of equations becomes linear. For $\hat{k} = pq$, we have $D = \min\{p, q\}$ which follows from the equality

$$\Pi_{pq}(x) = \frac{(x^p - 1)(x^q - 1)}{(x - 1)} .$$

Indeed, assume that $p < q$, then $(x^p - 1)/(x - 1)$ is the all-one polynomial of degree $(p-1)$ and since $p < q$, multiplication by $(x^q - 1)$ does not cancel out any non-zero terms. For three or more primes, the pattern becomes more intricate, but for $k = 2pq$ with $p < q$, we have $D = 2p$. These cases cover all k up to 100 which is sufficient for practical applications.

To compare version 1 and version 2 of the algorithm described in Section 4, we ran several tests using MAGMA for some interesting cases of k, namely $k = 3, 6, 12, 15, 30$. For each k, we determined the best multiple of $e = (q^k - 1)/\Phi_k(q)$ using version 1 and 2. We then derived the system of non-linear equations as described in Section 4 for the LHRP using a predetermined number of queries to the oracle. Finally, the system of non-linear equations was solved using the Groebner basis command in MAGMA. In these tests, the prime field $\mathbb{F}_p$ was kept constant for some fixed random 32-bit prime p, since the size of p only has a minor influence on the feasibility of the tests. Table 1 gives a summary of the test results. Especially the last two entries are interesting since version 2 outperforms version 1 considerably. For example, for $k = 30$, version 1 leads to a system of non-linear equations of degree $D = 6$ corresponding to

$$\Pi(x) = (x^{30}-1)/\Phi_{30}(x) = x^{22}-x^{21}+x^{20}+x^{17}-x^{16}+x^{15}-x^7+x^6-x^5-x^2+x-1$$

so it would be nearly impossible to even write down the system of equations, since each non-linear equation contains 1947792 terms. However, version 2 finds a much better multiple of e corresponding to

$$(x^5 + x^4 - x^2 - x - 1)\Pi(x) = x^{27} - x^{20} - x^{17} - x^{15} - x^{12} + x^5 + x^2 + 1 ,$$

leading to $D = 4$ and thus non-linear equations of only 46376 terms, which is just manageable, but already used up around 3 Gb of memory.

In each case, we made roughly $2 \log_h q^k$ queries to the oracle. The column "Min # queries" contains the minimum number of queries to the oracle that gives a unique solution to the system of non-linear equations. This number approximates the expected $\log_h q^k$ from the information theoretical viewpoint.

For most values of $k < 100$, the algorithm performs fairly well since D will often be very small. However, for k having many prime factors, the number of terms in the non-linear equations simply becomes too large, e.g. for $k = 70$, version 2 gives $D = 4$ and thus $M = 1150626$.

In stark contrast, the subfield LHRP with projection onto $T_k(\mathbb{F}_q)$ is easy for all practical k (for instance all $k < 1000$) and the algorithm runs very fast in a matter of seconds, e.g. for $k = 665$ we have $D = 85$ and solving the subfield

Table 1. Comparison of version 1 and 2 for the LHRP and projection on $T_k(\mathbb{F}_q)$ for various extension degrees k

k	D (v.1)	D (v.2)	$M = \binom{D+k}{D}$	Min # queries	Time (s)
3	1	1	4	2	0.01
6	2	2	28	4	0.01
12	2	2	91	4	0.04
15	3	2	816 / 136	5	0.15
30	6	4	1947792 / 46376	16	2420

LHRP by using GCD's only takes 240 seconds, most of which is spent in computing the system of univariate equations. For the full LHRP, the number of terms in the non-linear equations would be $M \simeq 10^{114}$, which is clearly infeasible.

The results in this section show that the LHRP will be solvable for moderately sized k if e is a divisor of $(q^k-1)/\Phi_k(q)$, since then it can be reduced to projection onto $T_k(\mathbb{F}_q)$. However, when e is of the form $(q^k-1)/s$ with s a proper divisor of $\Phi_k(q)$ with large cofactor (e.g. 2^{80}), i.e. e-th powering corresponds to projection into a strict subgroup of $T_k(\mathbb{F}_q)$ with large cofactor, then the HRP (and also the subfield HRP) remains hard. In implementing pairing based cryptosystems in a side-channel resistant manner, it is therefore paramount to ensure that the final exponentiation does not succumb to the algorithms described in Section 4. In particular, this implies that final exponentiations of low Hamming weight in base q should be avoided. The implications for the more general problem of pairing inversion are currently unclear: it is possible to define pairings with "easy" final exponentiation from an HRP point of view, but the obvious candidates all lead to Miller functions $f_{S,P}$ of restrictively high degree.

6 Conclusion

In this paper we studied a new computational problem called the Hidden Root Problem, motivated by immediate applications to cryptanalysis of pairing based cryptosystems. We have given the first algorithm to solve this problem over extension fields and concluded that for exponents e which are divisors of $(q^k-1)/\Phi_k(q)$ the problem can be solved efficiently for moderately sized k. However, for exponents e of the form $(q^k-1)/s$ with s a proper divisor of $\Phi_k(q)$ with large cofactor, the problem remains hard. In implementing pairing based cryptosystems, it is therefore advisable to avoid using Miller functions $f_{S,P}$ where S is a multiple of $\Phi_k(q)$.

Acknowledgements

The author would like to thank the anonymous referees of Pairing 2008 for suggesting several improvements to the exposition of the paper.

References

1. P.S.L.M. Barreto, S.D. Galbraith, C. Ó hÉigeartaigh, and M. Scott. Efficient pairing computation on supersingular abelian varieties. In *Designs, Codes and Cryptography*, vol 42(3), pages 239–271, 2007.
2. D. Boneh and R. Venkatesan. Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes. In *Lect.Notes in Comp. Sci.*, Springer-Verlag, Berlin, 1109, pages 129–142, 1996.
3. D. Boneh and R. Venkatesan. Rounding in lattices and its cryptographic applications. In *Proc. 8th Annual ACM-SIAM Symp. on Discr. Algorithms*, ACM, NY, pages 675–681, 1997.
4. B. Buchberger. A theoretical basis for the reduction of polynomials to canonical forms. *SIGSAM Bull.*, 10(3): 19–29, 1976.
5. N. Courtois, A. Klimov, J. Patarin and A. Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. *Advances in Cryptology – EUROCRYPT 2000*, Springer-Verlag LNCS 1807, 392–407, 2000.
6. T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inform. Theory*, 31(4):469–472, 1985.
7. S.D. Galbraith, F. Hess, and F. Vercauteren. Aspects of pairing inversion. Preprint, 2008. Available from `http://eprint.iacr.org/2007/256`.
8. F. Hess, N. Smart, and F. Vercauteren. The Eta-pairing revisited. *IEEE Transactions on Information Theory*, 52(10):4595–4602, 2006.
9. V.S. Miller. The Weil pairing, and its efficient calculation. *J. Cryptology*, 17(4):235–261, 2004.
10. D. Page and F. Vercauteren. A Fault Attack on Pairing Based Cryptography. In *IEEE Transactions on Computers*, volume 55(9), pages 1075–1080, July 2006.
11. J. Hoffstein, J. Pipher and J.H. Silverman. NTRU: a ring-based public key cryptosystem. *Algorithmic number theory – ANTS III*, Springer-Verlag LNCS 1423, 267–288, 1998.
12. A.K. Lenstra, H.W. Lenstra and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.
13. R.L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Comm. ACM*, 21(2):120–126, 1978.
14. I.E. Shparlinski. Playing "Hide-and-Seek" in finite fields: Hidden number problem and its applications. In *Proc. 7th Spanish Meeting on Cryptology and Information Security*, Vol.1, Univ. of Oviedo, 49–72, 2002.
15. E. Verheul. Evidence that XTR Is More Secure than Supersingular Elliptic Curve Cryptosystems. In B. Pfitzmann, editor, *EUROCRYPT*, volume of 2045 *Lecture Notes in Computer Science*, pages 195–210. Springer, 2001.
16. E. Verheul, Evidence that XTR is more secure than supersingular elliptic curve cryptosystems, *J. Crypt.*, **17**, No. 4 (2004) 277–296.
17. C. Whelan and M. Scott. The Importance of the Final Exponentiation in Pairings when Considering Fault Attacks. In Tsuyoshi Takagi, Tatsuaki Okamoto, Eiji Okamoto, and Takeshi Okamoto, editors, *First International Conference on Pairing-Based Cryptography - Pairing 2007*, volume 4575 of *Lecture Notes in Computer Science*, pages 225–246. Springer, 2007.
18. C. Whelan and M. Scott. Side Channel Analysis of Practical Pairings: Which Path is more Secure? In Phong Q. Nguyen, editor, *International Conference on Cryptology in Vietnam - VIETCRYPT 2006*, volume 4341 of *Lecture Notes in Computer Science*, pages 99–114. Springer Verlag, September 2006.