ARTICLE IN PRESS

Journal of Computational and Applied Mathematics ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

An overview of trace based public key cryptography over finite fields

Ersan Akyıldız [a,b], Muhammad Ashraf [a,*]

[a] Institute of Applied Mathematics, Middle East Technical University, Ankara, Turkey
[b] Department of Mathematics, Middle East Technical University, Ankara, Turkey

ARTICLE INFO

Article history:
Received 11 March 2013
Received in revised form 14 October 2013

Keywords:
Public key cryptosystems
Discrete logarithm problem
Finite field extensions
LFSR based exponentiation
Characteristic sequence
Digital signature scheme

ABSTRACT

The Discrete Log Problem (DLP), that is computing x, given $y = \alpha^x$ and $\langle\alpha\rangle = G \subset \mathbb{F}_q^*$, based Public Key Cryptosystem (PKC) have been studied since the late 1970's. Such development of PKC was possible because of the trapdoor function $f : \mathbb{Z}_\ell \to G = \langle\alpha\rangle \subset \mathbb{F}_q^*$, $f(m) = \alpha^m$ is a group homomorphism. Due to this fact we have; Diffie Hellman (DH) type key exchange, ElGamal type message encryption, and Nyberg–Rueppel type digital signature protocols. The cryptosystems based on the trapdoor $f(m) = \alpha^m$ are well understood and complete. However, there is another trapdoor function $f : \mathbb{Z}_\ell \to G, f(m) \to \mathrm{Tr}(\alpha^m)$, where $G = \langle\alpha\rangle \subset \mathbb{F}_{q^k}^*$, $k \geq 2$, which needs more attention from researchers from a cryptographic protocols point of view. In the above mentioned case, although f is computable, it is not clear how to produce protocols such as Diffie Hellman type key exchange, ElGamal type message encryption, and Nyberg–Rueppel type digital signature algorithm, in general. It would be better, of course if we can find a more efficient algorithm than repeated squaring and trace to compute $f(m) = \mathrm{Tr}(\alpha^m)$ together with these protocols. In the literature we see some works for a more efficient algorithm to compute $f(m) = \mathrm{Tr}(\alpha^m)$ and not wondering about the protocols. We also see some works dealing with an efficient algorithm to compute $\mathrm{Tr}(\alpha^m)$ as well as discussing the cryptographic protocols. In this review paper, we are going to discuss the state of art on the subject.

© 2013 Elsevier B.V. All rights reserved.

1. Introduction

The public key cryptography plays an important role to ensure confidentiality, integrity, authentication, and non-repudiation of the transmitted data. With the increase of computing speed the conventional public key cryptography over finite fields required revival to ensure such properties. The trace based public key cryptography is one of the quests which ensures higher security (over extension field), efficient transmission (over subfield) and faster computation (through linear recursive relation). Due to these properties the trace based Public Key Cryptosystems (PKC) have become popular in the last two decades. Therefore, the trace based PKC over finite fields are used for several modern cryptographic applications such as key agreement, encryption and digital signatures with or without message recovery.

The trace based PKC are generally based on the concept of Discrete Log Problem (DLP), that is computing x, given $y = \alpha^x$ and $\langle\alpha\rangle = G \subset \mathbb{F}_q^*$. Such development of PKC was possible because of the trapdoor function $f : \mathbb{Z}_\ell \to G = \langle\alpha\rangle \subset \mathbb{F}_q^*, f(m) = \alpha^m$ is a group homomorphism. The cryptosystems based on the trapdoor $f(m) = \alpha^m$ are well understood and complete. However, the trapdoor based on trace function $f : \mathbb{Z}_\ell \to G, f(m) \to \mathrm{Tr}(\alpha^m)$, where $G = \langle\alpha\rangle \subset \mathbb{F}_{q^k}^*$, $k \geq 2$, needs more attention from researchers from a cryptographic protocols point of view.

* Corresponding author. Tel.: +90 5433249442.
E-mail addresses: ersan@metu.edu.tr (E. Akyıldız), ashraf6061@gmail.com (M. Ashraf).

0377-0427/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.cam.2013.10.022

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

The development of the later trapdoor is based on the concept of the sum of the roots of an irreducible polynomial that is a minimal polynomial of degree k over subfield or prime field. The computation of the sum of the roots that is trace function (i.e; $\mathrm{Tr}(\alpha^m)$) is accomplished through a faster algorithm based on a linear recursive relation when compared with conventional square and multiply, and trace algorithms. Consequently, trace based PKC algorithms provide higher security of extension field and faster computation through linear recursive relation over subfield. Moreover, the transmitted data is also over subfield requiring less transmission bandwidth and hence efficient data transmission. Keeping these advantages in mind, we see some works in the literature which are presented by Smith, Lennon and Skinner (LUC-PKC) in 1994 [1,2], L. Horn and G. Gong, (GH-PKC) in 1998 [3,4], A.K. Lenstra and E.R. Verheul (XTR-PKC) in 2000 [5–8] and K. Giuliani, G. Gong (GG-PKC) in 2003 [3,4] and Koray Karabina (KK-PKC) in 2009 [9].

In the literature there are also so called torus based cryptosystems introduced by K. Rubin and A. Silverberg in 2003 [10]. These systems depend on the parametric representation of the group $\mathbb{T}_k(\mathbb{F}_q) \cong G_{q,k} = \langle\alpha\rangle \subset \mathbb{F}_{q^k}^*$ and $\mathbb{T}_k(\mathbb{F}_q)$ rather than the efficient algorithm to compute $\mathrm{Tr}(\alpha^m)$, where $\mathbb{T}_k(\mathbb{F}_q)$ is a torus consisting of the elements in $\mathbb{F}_{q^k}$ whose norm in unity down to every intermediate subfield. Therefore, torus based cryptography is an area which applies a different technique, namely rational parameterization of group elements, to develop cryptographic protocols. Therefore, it is not included in this note.

The aim of the note is to review the so called trace based cryptography studied in the literature and bring to the attention of the researchers the challenges faced in these systems. We will not only review these systems but also introduce cryptographic protocols which are not discussed in the literature.

This note is organized as follows: in Section 2, we will review the necessary mathematical background and protocols for DLP based PKC. In Section 3, we review the trace based systems including mathematical structure for an efficient algorithm to compute $f(m) = \mathrm{Tr}(\alpha^m)$ and cryptographic protocols such as Diffie Hellman key exchange, ElGamal type encryption scheme and Nyberg–Rueppel type Digital Signature Algorithm. Although most of the facts discussed here are well known, we have also added some new results such as ElGamal type encryption and Nyberg–Rueppel type signature schemes for some cases which have not been discussed in the literature. Finally, we conclude the note in Section 4.

2. Discrete log based PKC over finite fields

The general mathematical structure of the PKCs based on finite field and cryptographic protocols such as key exchange, encryption scheme, and Nyberg–Rueppel digital signature algorithm are basic building blocks for the PKC. Also the PKC is considered practical if we have cryptographic protocols and efficient algorithms for computations involved in these protocols. Keeping this in focus, we discuss DLP based PKC over subgroup G of a finite field.

2.1. DLP based PKC over G

Let,

$$G = \langle\alpha\rangle \subset \mathbb{F}_q^*, \quad q = p^r, \text{ i.e; prime power}$$
$$G = \langle\alpha\rangle \cong \mathbb{Z}_\ell, \quad ord(\alpha) = \ell$$
$$f : \mathbb{Z}_\ell \to G; \quad \text{with } f(m) = \alpha^m.$$

The f is a computable trapdoor, and well known repeated squaring algorithm computes f in polynomial time.

For the DLP on G such that given $y \in G$, find $f^{-1}(y)$, one has in general *Pollard's Rho*$(O(e^{\sqrt{\log \ell}}))$ and *Index Calculus* $(O(e^{\sqrt{2\log \ell \log\log \ell}}))$ algorithms which are sub exponential time. By choosing the order G a large prime ℓ one can avoid these algorithms so that f becomes a one-way function on $\mathbb{F}_q^*$. With this setup now we give the well known following cryptographic protocols:

2.2. Diffie Hellman key exchange

1. *System public parameters:* $G = \langle\alpha\rangle \subset \mathbb{F}_q^*$; $ord(\alpha) = \ell$.
2. **A**'s and **B**'s public keys are $P_A = \alpha^a$, and $P_B = \alpha^b$ respectively, and private keys are $1 < a < \ell$ and $1 < b < \ell$ respectively.
3. Their common key, $K = P_{AB} = P_{BA} = \alpha^{ab}$:

2.3. ElGamal encryption scheme

1. *System public parameters:* $G = \langle\alpha\rangle \subset \mathbb{F}_q^*$; $ord(\alpha) = \ell$.
2. **A** selects a random $1 < u < \ell$. **A**'s public key: $h = \alpha^u$, and private key $= u$.
3. *Assumption*: message is $M \in G$.
4. *Encryption*: **B** encrypts message M as follows and sends to **A**:
   (i) Chooses a random $1 < r < \ell$, then computes $c_1 = \alpha^r$, $s = h^r$ and $c_2 = Ms$.
   (ii) Sends ciphertext $C = \{c_1, c_2\} \in G^2$ to **A**.
5. *Decryption*: **A** decrypts $C \in G^2$ based upon his private key u as follows;

$$c_1^{-u} c_2 = \alpha^{-ur} M h^r = M\alpha^{ur}\alpha^{-ur} = M.$$

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

2.4. Nyberg–Rueppel digital signature algorithm

1. *System public parameters:* $G = \langle\alpha\rangle \subset \mathbb{F}_q^*$, $ord(\alpha) = \ell$, and bijection map $f : G \longrightarrow \mathbb{Z}_\ell$.
2. **A**'s public key $P_A = \alpha^u$, and private key: $u \in \mathbb{Z}$.
3. Assumption: messages M are in G.
4. *Signature*: **A** signs message $M \in G$ as follows:
   (i) Chooses a random $1 < v < \ell$.
   (ii) Computes $r = M\alpha^{-v}$ and $s = v^{-1}(1 - f(r)u)(\text{mod } \ell)$. **A**'s signature is $(r, s) \in (G \times \mathbb{Z}_\ell)$ and sends (M, r, s) to **B**.
5. *Verification:* **B** verifies by computing $(P_A)^{-f(r)}r^s = \alpha^{sv-1-sv}M^s = M^s\alpha^{-1}$.

On the DLP based PKC, the main point is the trapdoor function:

$$G = \langle\alpha\rangle \subset \mathbb{F}_q^*, \quad ord(\alpha) = \ell,$$
$$f : \mathbb{Z}_\ell \longrightarrow \mathbb{F}_q^*, \quad f(m) = \alpha^m.$$

Note 1. The trapdoor function f is a group homomorphism; $f(m + k) = f(m)f(k)$, and one should be careful to choose $\langle\alpha\rangle = G \subset \mathbb{F}_q^*$ to avoid algorithms to attack DLP on G and choose, if possible, the smallest size G such that DLP on G is computationally equivalent to DLP on $\mathbb{F}_q^*$.

3. Trace based public key cryptosystems over finite fields

Let α be an element over $\mathbb{F}_{q^k}$ with the characteristic polynomial;

$$g(x) = x^k - a_1x^{k-1} + \cdots + (-1)^k a_k,$$

over $\mathbb{F}_q$ and let, Tr, Norm : $\mathbb{F}_{q^k} \to \mathbb{F}_q$, given by;

$$\text{Tr}(\beta) = \sum_{i=0}^{k-1} \beta^{q^i}, \qquad \text{Norm}(\beta) = \prod_{i=0}^{k-1} \beta^{q^i}.$$

Note that,

$$g(x) = \prod_{i=0}^{k-1}(x - \alpha^{q^i}),$$

and $a_i = \sigma_i(\alpha, \alpha^q, \ldots, \alpha^{q^{k-1}})$, σ_i is the ith elementary symmetric function in k variables. For any integer m, let $g_m(x)$ be the characteristic polynomial of α^m,

$$\begin{aligned} g_m(x) &= \prod_{i=0}^{k-1}(x - (\alpha^m)^{q^i}), \\ &= x^k - f_1(m)x^{k-1} + \cdots + (-1)^i f_i(m)x^{k-i} + \cdots + (-1)^k f_k(m). \end{aligned}$$

Note that; $g_1 = g$.

This gives us for $i = 1, 2, \ldots, k$,

$$f_i : \mathbb{Z}_n \longrightarrow \mathbb{F}_q^*, \qquad f_i(m) = \sigma_i(\alpha^m, (\alpha^m)^q, \ldots, (\alpha^m)^{q^{k-1}}).$$

3.1. Relation of $f_i(m)$ with LFSR

Let, α be an element of finite field $\mathbb{F}_{q^k}$, with characteristic polynomial over $\mathbb{F}_q$:

$$g(x) = x^k + b_1x^{k-1} + \cdots + b_k.$$

For a given initial state $\{s_0, s_1, \ldots, s_{k-1}\}$ one produces a unique periodic sequence, $\{s_m = -b_1s_{m-1} - b_2s_{m-2} - \cdots - b_ks_{m-k},\ m \geq k\}$, generated by $g(x)$. The general term s_n can be expressed uniquely in the form:

$$s_n = c_1\alpha_1^n + c_2\alpha_2^n + \cdots + c_k\alpha_k^n, \quad \text{for some } c_i \in \mathbb{F}_q,$$

and

$$g(x) = \prod_{i=1}^{k}(x - \alpha_i), \quad \alpha_i \in \mathbb{F}_{q^k}.$$

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

Remark 1. It is well known that finding an efficient algorithm to compute $f_i(m) = \sigma_i(\alpha^m, (\alpha^m)^q, \ldots, (\alpha^m)^{q^{k-1}})$ is computationally equivalent to finding an algorithm to compute s_n, with given specific initial conditions $s_0, \ldots, s_{k-1}$, where for $i = 1, 2, \ldots, k$; $\alpha_i = (\alpha^m)^{q^{i-1}}$.

Remark 2. We note that by means of Newton's formula, having an efficient algorithm to compute $f_1(m) = \mathrm{Tr}(\alpha^m)$ is computationally equivalent to finding such an algorithm for $f_i(m)$.

From now on our work will be restricted to the function $f_1(m) = \mathrm{Tr}(\alpha^m)$ and for simplicity we denote f_1 as f. For $m = 1, 2, \ldots$ and $\alpha \in \mathbb{F}_{q^k}$, let,

$$g_m(x) = \prod_{i=0}^{k-1}(x - (\alpha^m)^{q^i}),$$

$$\widetilde{g}_m(x) = \prod_{i=0}^{k-1}(x - (\alpha^{-m})^{q^i}).$$

Here $\widetilde{g}_m(x) = x^k g_m(\frac{1}{x})$, is the reciprocal of $g_m(x)$.

Note that; for $f : \mathbb{Z}_\ell \to \mathbb{F}_q^*, f(m) = \mathrm{Tr}(\alpha^m)$, computing $f^{-1}(\beta)$ is computationally equivalent to solving DLP on $G = \langle\alpha\rangle \subset \mathbb{F}_{q^k}^*$. Since $f(m) = \mathrm{Tr}(\alpha^m)$ is not a group homomorphism, it is not known, in general, how to obtain cryptographic protocols like the ones in DLP case based on $\alpha \to \alpha^m$. Certainly, the PKC based on $f(m) = \mathrm{Tr}(\alpha^m)$ will give a more compressed system versus the system based on $\alpha \to \alpha^m$ in $\mathbb{F}_{q^k}^*$. With this development we define the trace based PKC as follows:

Definition 2. By a trace based cryptosystem we mean a Public Key Cryptosystem based on the one-way function,

$$f : \mathbb{Z}_\ell \to G = \langle\alpha\rangle \subset \mathbb{F}_{q^k}^*,$$
$$f(m) = \mathrm{Tr}(\alpha^m).$$

We mean a cryptosystem having an adaptability to cryptographic protocols such as, DH type key exchange, ElGamal type message encryption scheme, and Nyberg–Rueppel type digital signature algorithm, based on f.

Main challenges for trace based cryptography. Let α be an element of $\mathbb{F}_{q^k}$ having characteristic polynomial $g(x) = x^k - a_1x^{k-1} + \cdots + (-1)^k a_k$ over $\mathbb{F}_q$, *find conditions on* α *such that*:

(i) There is a more efficient polynomial time algorithm than the algorithm that first computes α^m using the square-and-multiply algorithm, then computes the trace of α^m, for a given $m \in \mathbb{Z}$.
(ii) Introduce cryptographic protocols similar to protocols discussed above for DLP-based system.
(iii) Compare the security and implementation of these protocols with the existing ones.

Remark 3. The problem (i) arose also in pairing based cryptosystems [11] and there have been some works done there for some values of k and special conditions on α such as $ord(\alpha) = \ell|\Phi_k(q)$, where $\Phi_k(q)$ is the kth cyclotomic polynomial evaluated at q.

By keeping in mind the above challenges, we now discuss the related work done in the literature and in this respect the following is the list of Public Key Cryptosystems.

(i) LUC-PKC: $k = 2,\ q = prime\ power$, p-arbitrary; [1,2].
(ii) GH-PKC: $k = 3,\ q = prime\ power$, p-arbitrary; [12,13].
(iii) GH-PKC Special Case: $k = 3,\ q = p^2,\ p$-arbitrary; [5–7].
(iv) GG PKC: $k = 5,\ q = prime\ power$, p-arbitrary; [3].
(v) GG-PKC Special Case: $k = 5,\ q = p^2,\ p$-arbitrary; [4].

3.2. Motivations to use trace based public key cryptography

The main reasons for developing cryptographic protocols on Trace Based construction are as follows:

(i) *Security:* In GH-PKC, the trace based construction provides security of extension field $\mathbb{F}_{q^k}$, that is DLP lies in $\mathbb{F}_{q^k}$ with faster exponentiation and compressed data for transmission.
(ii) *Transmission size:* Due to the compression of operands the overall data transmission size is also reduced to implement PKC protocols as given in Table 1.
(iii) The cost of exponentiation is better than the generic square and multiply algorithm and elliptic curve ($E(\mathbb{F}_q)$) based exponentiation. The details are given in Table 2.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

Table 1
Trace based compression factors for cryptographic protocols.

Extension field	Compression factor	PKC system
$\alpha^n \in \mathbb{F}_{q^2}, q = p^r, r \in \mathbb{Z}$	1/2	LUC-PKC
$\alpha^n \in \mathbb{F}_{q^3}, q = p^2,$	1/3	GH-PKC and XTR-PKC
$\alpha^n \in \mathbb{F}_{q^5}, q = p^2,$	2/5	GG-PKC
$\alpha^n \in \mathbb{F}_{q^6}, q = 3^{2r+1}, r \in \mathbb{Z}$	1/6	KK-PKC
$\alpha^n \in \mathbb{F}_{q^4}, q = 2^{2r+1}, r \in \mathbb{Z}$	1/4	KK-PKC

Table 2
Exponentiation comparison.

Process	Multiplications over $\mathbb{F}_q$	Squaring over $\mathbb{F}_q$
$\mathbb{F}_{q^2}$		
$\alpha^n \in \mathbb{F}_{q^2}, q = p^r, r \in \mathbb{Z}$	$weight(n) - 1$	$\lfloor \log n \rfloor$
$s_n, \alpha^n \in \mathbb{F}_{q^2}$	$0.75 \log n$	$0.17 \log n$
$nP, P \in E(\mathbb{F}_{q^2})$	$7 \log n$	$3.6 \log n$
F_{q^3}		
$\alpha^n \in \mathbb{F}_{q^3}, q = p^r, r \in \mathbb{Z}$	$(7r^{\log 3} + 10.8r - 10.8) \log n$	$3.8r^{\log 3} \log n$
$s_n, \alpha^n \in \mathbb{F}_{q^3}$	$5.2 \log n$	–
$nP, P \in E(\mathbb{F}_q)$	$4 \lceil \log n \rceil + 12(weight(n))$	$4 \lceil \log n \rceil + 4(weight(n))$
$\mathbb{F}_{q^5}$		
$\alpha^n \in \mathbb{F}_{q^5}, q = p^2,$	$45(weight(n) - 1)$	$30 \lfloor \log n \rfloor$
$s_n, \alpha^n \in \mathbb{F}_{q^5}, q = p^2$	$108.5 \log n$	$13 \log n$

3.3. Algorithm to compute mixed term $s_{u+v} = \mathrm{Tr}(\alpha^u \alpha^v)$, given $\mathrm{Tr}(\alpha), u, S_v = (s_{v-k+1} \cdots s_v), s_j = \mathrm{Tr}(\alpha^j)$ for $j \in \mathbb{Z}$, and v-unknown

Before discussing PKC systems we give a generalized algorithm to compute mixed terms, that is $s_{u+v} = \mathrm{Tr}(\alpha^u \alpha^v)$ given $u, \mathrm{Tr}(\alpha), S_v = (s_{v-k+1}, \ldots, s_v)$, v-unknown. This algorithm will be required in a Nyberg–Rueppel type digital signature system. Note that the algorithm to compute $\mathrm{Tr}(\alpha^u)$ given $\mathrm{Tr}(\alpha)$ and u will be discussed in respective PKC and here we assume such an algorithm already exists. In fact, one can always take repeated squaring and trace for such computations.

Let $G = \langle \alpha \rangle \subset \mathbb{F}_{q^k}^*$, p-arbitrary, $q = p^r$, and $g(x)$ be the characteristic polynomial of α:

$$g(x) = \prod_{i=0}^{k-1} (x - \alpha^{q^i}) = x^k - a_1 x^{k-1} + a_2 x^{k-2} - \cdots + (-1)^{k-1} a_{k-1} x + (-1)^k.$$

Let A be the companion matrix associated with $g(x)$:

$$A = \begin{pmatrix} 0 & 0 & \cdots & 0 & (-1)^{k-1} \\ 1 & 0 & \cdots & 0 & (-1)^{k-2} a_{k-1} \\ 0 & 1 & \cdots & 0 & (-1)^{k-3} a_{k-2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & a_1 \end{pmatrix}. \tag{1}$$

Using the algorithm for $\mathrm{Tr}(\alpha^u)$, we form the state matrix M_u associated with u such that:

$$M_u = \begin{pmatrix} s_{u-k+1} & s_{u-k+2} & \cdots & s_{u-1} & s_u \\ s_{u-k+2} & s_{u-k+3} & \cdots & s_u & s_{u+1} \\ s_{u-k+3} & s_{u-k+4} & \cdots & s_{u+1} & s_{u+2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_u & s_{u+1} & \cdots & s_{u+k-2} & s_{u+k-1} \end{pmatrix}. \tag{2}$$

Let the vector $S_m = (s_{m-k+1}, \ldots, s_m)$ be given for any integer m then we have,

$$S_{m+1} = S_m A,$$

and therefore for any integer r,

$$S_{m+r} = S_m A^r.$$

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

This gives, $M_u = M_0 A^u$ therefore, $A^u = M_0^{-1} M_u$ and vector,

$$S_{u+v} = S_v(M_0^{-1} M_u) = (s_{u+v-k+1}, \ldots, s_{u+v}). \tag{3}$$

Algorithm 1 to compute $S_{u+v} = (s_{u+v-k+1}, \ldots, s_{u+v})$ for v-unknown

Require: $k \geq 2, S_v, u, \mathrm{Tr}(\alpha)$.
Ensure: S_{u+v}.
1: Construct A as in Eq. (1),
2: Construct M_u as in Eq. (2),
3: Compute, M_0^{-1},
4: Compute, $A^u \leftarrow M_0^{-1} M_u$,
5: Compute, $C \leftarrow S_v(A^u)$.
6: **return** (C).

Remark 4. Following is the step by step computational complexities:

(i) The computational cost to construct M_u is the cost of algorithm to compute $S_u + O(k^3)$ field multiplications.
(ii) M_0 can be constructed from initial states with $O(k^3)$ field multiplications.
(iii) The complexity of computing M_0^{-1} and $M_0^{-1} M_u$ is also bounded by $O(k^3)$. Therefore, $s_{u+v} = \mathrm{Tr}(\alpha^{u+v})$ can be computed in about $O(k^3)$ field multiplications.
(iv) The precomputation phase that is computing S_u depends upon the extension field and corresponding sub field of the PKC. Therefore, it is not included in the generic version of the algorithm.

3.4. LUC-PKC: case $k = 2$, p-arbitrary, $q = p^r$, for any positive integer r

Let $G = \langle \alpha \rangle \cong \mathbb{Z}_\ell \subset \mathbb{F}_{q^2}^*$, $ord(\alpha) = \ell | (q+1)$, then, the polynomial

$$g(x) = x^2 - ax + 1, \quad a = \mathrm{Tr}(\alpha), \qquad \mathrm{Norm}(\alpha) = 1, \quad \text{and}$$
$$g_m(x) = (x - \alpha^m)(x - (\alpha^m)^q) = x^2 - V_m(a)x + 1, \quad m \in \mathbb{Z}.$$
$$\text{Note that,} \quad \tilde{g}(x) = g(x) \quad \text{and} \quad \tilde{g}_m(x) = g_m(x).$$

Notation: For any integer r, let $V_r : \mathbb{F}_q \to \mathbb{F}_q$ be a function given by $V_r(b) = \mathrm{Tr}(\beta^r)$, where β is a root of $x^2 - bx + 1 = 0$. It is clear that V_r is well defined, namely it is independent than the choice of the root β of $x^2 - bx + 1 = 0$. Note that for any integer r

$$h_r(x) = \prod_{i=0}^{1} (x - \beta^{rq^i}) = x^2 - V_r(b)x + 1,$$

in particular, $V_0(b) = 2$, $V_1(b) = b$. Furthermore, for any integer m, we have

$$h_{mr}(x) = \prod_{i=0}^{1} (x - (\beta^{mr})^{q^i}) = x^2 - V_m(V_r(b))x + 1$$

over $\mathbb{F}_q$. Which implies that, for any integers r and m we have $V_m(V_r(b)) = V_{mr}(b)$.

The LFSR sequence $\{s_m\}$ with initial conditions $\{2, a\}$ associated to characteristic polynomial $g(x) = x^2 - ax + 1$, gives us the general term:

$$s_m = V_m(a) = \mathrm{Tr}(\alpha^m).$$

Properties: Following properties for all integers m and n are proved in [1] and we list them;

(i) $V_m(a) = V_{-m}(a) = s_m$;
(ii) $V_m(V_n(a)) = V_{mn}(a) = s_{mn}$;
(iii) $s_{2m} = s_m^2 - 2$;
(iv) $s_{m+n} = s_m s_n - s_{m-n}$.

These properties give us the following efficient algorithm to compute $s_m = \mathrm{Tr}(\alpha^m)$.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

Algorithm 2 to compute $s_m = \mathrm{Tr}(\alpha^m)$

Require: $\alpha \in \mathbb{F}_{q^2}$, $ord(\alpha) = \ell|(q+1)$, $\mathrm{Tr}(\alpha) = a \in \mathbb{F}_q$ and $m = \sum_{j=0}^{t-1} \epsilon_j 2^j \in \mathbb{Z}$ with $\epsilon_j = \{0, 1\}$, $\epsilon_{t-1} = 1$.
Ensure: (s_m, s_{m+1})
1: $(s_y, s_{y+1}) \leftarrow (2, a)$
2: **for** $j \leftarrow t-1$ **to** 0 **do**
3: **if** $\epsilon_j = 1$ **then**
4: $s_y \leftarrow s_y s_{y+1} - s_1$, $s_{y+1} \leftarrow s_{y+1}^2 - 2$.
5: **else**
6: $s_y \leftarrow s_y^2 - 2$, $s_{y+1} \leftarrow s_y s_{y+1} - s_1$.
7: **end if**
8: **end for**
9: **return** (s_y, s_{y+1})

Remark 5. For any integer m this algorithm actually computes (s_{m-1}, s_m) from the initial conditions $\{2, a\}$.

This algorithm is more efficient than computing α^m and taking $\mathrm{Tr}(\alpha^m)$. In fact this algorithm does compute $\mathrm{Tr}(\alpha^m)$ by using $\log m$ multiplications and $\log m$ squaring on $\mathbb{F}_q$.

3.4.1. LUC-DH type key exchange

1. *System public parameters:* Let $G = \langle\alpha\rangle \subset \mathbb{F}_{q^2}^*$, $ord(\alpha) = \ell|(q+1)$, and $g(x) = x^2 - ax + 1$, be the characteristic polynomial of α.
2. *Private keys:* Random m, and r satisfying $1 < m, r < \ell$ are the private keys of **A** and **B** respectively.
3. *Public keys:* Both A and B computes their public keys as follows:
   (i) **A** computes her public key $P_A = V_m(a) = s_m$, by running Algorithm 2 with inputs $a = \mathrm{Tr}(\alpha)$ and her private key m.
   (ii) **B** computes his public key $P_B = V_r(a) = s_r$, by running Algorithm 2 with inputs $a = \mathrm{Tr}(\alpha)$ and his private key r.
4. *Common key:* Both A and B agrees on the common key $K = P_{AB} = P_{BA} = V_{mr}(a) = s_{mr}$ as follows:
   (i) **A** acquires **B**'s public key and constructs $g_r(x) = x^2 - P_B x + 1 = X^2 - V_r(a)x + 1$. Then she computes $g_{mr}(x) = x^2 - V_m(V_r(a)) + 1$, by running Algorithm 2 with inputs $P_B = V_r(a)$ and her private key m such that,
   $$K = P_{AB} = V_m(P_B) = V_m(V_r(a)) = V_{mr}(a) = s_{mr}.$$
   (ii) **B** does the similar things to compute,
   $$K = P_{BA} = V_r(P_A) = V_r(V_m(a)) = V_{rm}(a) = s_{rm}.$$

3.4.2. LUC-ElGamal encryption scheme

1. *System public parameters:* Let $G = \langle\alpha\rangle \subset \mathbb{F}_{q^2}^*$, $ord(\alpha) = \ell|(q+1)$, and $g(x) = x^2 - ax + 1$, be the characteristic polynomial of α.
2. **B** runs Algorithm 2 with inputs a and his private key $1 < t < \ell$ to compute his public key $P_B = V_t(a) = s_t$, and publishes P_B.
3. Assumption: messages M are in G.
4. **A** sends message $M \in G$ as follows:
   (i) Chooses a random $1 < r < \ell$ and runs Algorithm 2 with inputs a and r to compute mask $V_r(a) = s_r$.
   (ii) Acquires **B**'s public key $P_B = s_t$ and computes symmetric key $K = V_r(P_B) = V_r(V_t(a)) = V_{rt}(a) = s_{rt}$, by running Algorithm 2 with inputs $P_B = V_t(a) = s_t$ and session key r.
   (iii) **A** encrypts M by computing $c = KM = s_{rt}M$ and sends ciphertext $C = (s_r, c)$.
5. *Decryption:* **B** decrypts C as follows:
   (i) Based upon mask s_r and his private key t **B** computes symmetric key $K = V_t(s_r) = V_t(V_r(a)) = V_{tr}(a) = s_{tr}$.
   (ii) Decrypts c to obtain M with symmetric key K (computed in 5(i)), such that $M = K^{-1}c = s_{tr}^{-1}c$.

3.4.3. LUC-Nyberg–Rueppel type digital signature algorithm

A wants to send signed message $M \in \mathbb{F}_q$ to **B** and **B** verifies it. To do this, they proceed as follows:

1. *System public parameters:* Let $G = \langle\alpha\rangle \subset \mathbb{F}_{q^2}^*$, $ord(\alpha) = \ell|(q+1)$, and $g(x) = x^2 - ax + 1$, be the characteristic polynomial of α, and $H : G \to \mathbb{Z}_\ell$ valued hash function.
2. *Public keys:*
   (i) **A** randomly selects $0 < e < \ell$ and computes public key $P_{\mathbf{A}} = V_e(a) = s_e$ by running Algorithm 2 with inputs a and her secret key e.
   (ii) **B** randomly selects $0 < r < \ell$ and computes his public key $P_{\mathbf{B}} = V_r(a) = s_r$ by running Algorithm 2 with inputs a and his secret key r.
3. *Private keys:* The private keys of both **A** and **B** are e and r, respectively.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

Table 3
LUC-NR-DSA analysis.

Process	Cost
Signature generation	1 exponentiation $(0.75 \log q)\mathbf{M} + (0.75 \log q)\mathbf{S}$
Signature verification	1 exponentiation $(1.5 \log q)\mathbf{M} + (1.5 \log q)\mathbf{S}$
Comm. overhead	1 element over $\mathbb{F}_q$
Public key size	1 element over $\mathbb{F}_q$

Where **M** stands for multiplication and **S** stands for squaring.

4. *Signature*: **A** signs the message M containing agreed upon redundancy as follows:
   (i) Randomly selects ephemeral private key $0 < t < \ell$ and computes ephemeral public key $V_t(a) = s_t$, by running Algorithm 2 with inputs a, t and symmetric key $V_t(s_r) = V_{tr}(a, b) = s_{tr}$ by running the same algorithm with inputs s_r and session key t.
   (ii) Computes ciphertext C by encrypting the message M using generic symmetric encryption scheme and computes the hash value of C, i.e., $h = H(C) \bmod \ell$.
   (iii) Computes $n = t - he \bmod \ell$.
   (iv) **A** sends the signature (n, C) to **B**.
5. *Verification*: **B** verifies **A**'s signature and recovers message as follows:
   (i) Computes $h = H(C) \in \mathbb{Z}_\ell$,
   (ii) By running Algorithm 2 with inputs $P_A = s_e$ and h, computes $V_h(s_e) = V_{he}(a) = s_{he}$. Note that, as stated in Remark 5, Algorithm 2 computes both s_{he} as well as s_{he-1}.
   (iii) Computes s_{n+he} by running Algorithm 1 with inputs $k = 2$, $S_{he} = (s_{he-1}, s_{he})$, n.
   (iv) Computes symmetric key by using s_{n+he} and his private key r and decrypts C to M using generic symmetric encryption scheme. **B** accepts if M contains agreed upon redundancy.

The analysis of LUC-NR-DSA is given in Table 3.

3.5. GH-PKC: case $k = 3$, p-arbitrary, $q = p^u$, $u \geq 1$

Let, $G = \langle\alpha\rangle \cong \mathbb{Z}_\ell \subset \mathbb{F}^*_{q^3}$, $\alpha \in \mathbb{F}_{q^3}$, $ord(\alpha) = \ell | (q^2 + q + 1)$. Then, one can check that characteristic polynomial $g(x)$ of α and its reciprocal $\tilde{g}(x)$ have the following properties;

$$g(x) = \prod_{i=0}^{2}(x - \alpha^{q^i}) = x^3 - ax^2 + bx - 1, \quad \text{and}$$

$$\tilde{g}(x) = \prod_{i=0}^{2}(x - \alpha^{-q^i}) = x^3 - bx^2 + ax - 1,$$

namely $a = \mathrm{Tr}(\alpha)$, $b = \mathrm{Tr}(\alpha^{-1})$. In fact, for all integers m, let $g_m(x)$, $\tilde{g}_m(x)$ be the following polynomials;

$$g_m(x) = \prod_{i=0}^{2}(x - (\alpha^m)^{q^i}) = x^3 - s_m x^2 + s_{-m}x - 1, \quad \text{and}$$

$$\tilde{g}_m(x) = \prod_{i=0}^{2}(x - (\alpha^{-m})^{q^i}) = x^3 - s_{-m} x^2 + s_m x - 1,$$

where $s_m = \mathrm{Tr}(\alpha^m)$ and $s_{-m} = \mathrm{Tr}(\alpha^{-m})$.

Notation: For any integer r let $V_r : \mathbb{F}_q^2 \to \mathbb{F}_q$ be a function given by $V_r(c, d) = \mathrm{Tr}(\beta^r)$, where β is a root of $x^3 - cx^2 + dx - 1 = 0$. It is clear that V_r is well defined, namely it does not depend on the choice of a root β of $h(x) = 0$.

In particular, $V_0(c, d) = 3$, $V_1(c, d) = c$ and $V_2(c, d) = c^2 - 2d$. It is clear that:

$$h_r(x) = \prod_{i=0}^{2}(x - (\beta^r)^{q^i}) = x^3 - V_r(c, d)x^2 + V_{-r}(c, d)x - 1,$$

and for any $m \in \mathbb{Z}$ we also have, $V_m(V_r(c, d), V_{-r}(c, d)) = V_{mr}(c, d) = \mathrm{Tr}(\beta^{mr})$.

The LFSR sequence $\{s_i\}$ with initial conditions $\{3, a, a^2 - 2b\}$ associated to $g(x) = x^3 - ax^2 + bx - 1$, gives us the general term $s_n = V_n(a, b) = \mathrm{Tr}(\alpha^n)$, for any integer n.

Properties: For all integers m and n the following properties are proved in [12] and we list them here;

(i) $V_m(V_n(a, b), V_{-n}(a, b)) = V_{mn}(a, b) = s_{mn}$,
(ii) $s_{2m} = s_m^2 - 2s_{-m}$,
(iii) $s_{m+n} = s_m s_n - s_{m-n}s_{-n} + s_{m-2n}$.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

Based upon above properties the GH-PKC algorithm to Compute $s_m = \mathrm{Tr}(\alpha^m)$ given $m \in \mathbb{Z}$, $a, b \in \mathbb{F}_q$ and $\mathrm{Tr}(\alpha)$ is as follows:

Algorithm 3 GH Algorithm to Compute $s_m = \mathrm{Tr}(\alpha^m)$

Require: An integer $m = \sum_{i=0}^{t} \epsilon_i 2^{t-i} \epsilon_i \in \{-1, 0, 1\}$ with $T_0 = \epsilon_0 \neq 0$ and $T_j = \epsilon_j + 2T_{j-1}$, $1 \leq j \leq t$, $g(x) = x^3 - ax^2 + bx - 1 \in \mathbb{F}_q[x] : g(\alpha) = 0$.
Ensure: (s_{m-1}, s_m, s_{m+1}),
$(s_{T_j-1}, s_{T_j}, s_{T_j+1}) \leftarrow (3, a, a^2 - 2b)$,
for $j = 0$ **to** t **do**
  if $\epsilon_j = 0$ **then**
    $s_{T_j-1} = s_{T_{j-1}} s_{T_{j-1}-1} - b s_{-T_{j-1}} + s_{-(T_{j-1}+1)}$;
    $s_{T_j} = s_{T_{j-1}}^2 - 2 s_{-T_{j-1}}$;
    $s_{T_j+1} = s_{T_{j-1}} s_{T_{j-1}+1} - a s_{-T_{j-1}} + s_{-(T_{j-1}-1)}$.
  else if $\epsilon_j = 1$ **then**
    $s_{T_j-1} = s_{T_{j-1}}^2 - 2 s_{-T_{j-1}}$;
    $s_{T_j} = s_{T_{j-1}} s_{T_{j-1}+1} - a s_{-T_{j-1}} + s_{-(T_{j-1}-1)}$;
    $s_{T_j+1} = s_{T_{j-1}+1}^2 - 2 s_{-(T_{j-1}-1)}$.
  else
    $s_{T_j-1} = s_{T_{j-1}-1}^2 - 2 s_{-(T_{j-1}-1)}$;
    $s_{T_j} = s_{T_{j-1}} s_{T_{j-1}+1} - a s_{-T_{j-1}} + s_{-(T_{j-1}-1)}$;
    $s_{T_j+1} = s_{T_{j-1}+1}^2 - 2 s_{-(T_{j-1}+1)}$.
  end if
end for
return $(s_{T_j-1}, s_{T_j}, s_{T_j+1})$.

Remark 6. The GH Algorithm 4 requires $4\lfloor \log m \rfloor$ multiplications and $4\lfloor \log m \rfloor$ squaring on $\mathbb{F}_q$ for computing both $s_m = \mathrm{Tr}(\alpha^m)$ and $s_{-m} = \mathrm{Tr}(\alpha^{-m})$. Moreover, for any integer m, this algorithm actually computes (s_{m-2}, s_{m-1}, s_m) from the initial conditions $\{3, a, a^2 - 2b\}$.

Note that Algorithm 5 due to Stam and Lenstra can compute s_m given s_1 and any integer m in $5.2 \log m$ multiplications over F_p.

3.5.1. GH-DH type key exchange

1. *System public parameters.* Let, $G = \langle \alpha \rangle \subset \mathbb{F}_{q^3}^*$, $ord(\alpha) = \ell | (q^2 + q + 1)$, and $g(x) = x^3 - ax^2 + bx - 1$, be the characteristic polynomial of α.
2. **A** chooses a random $1 < e < \ell$ and computes her public key $P_A = (V_e(a, b), V_{-e}(a, b)) = (s_e, s_{-e})$, by running Algorithm 3 with inputs e and a, b.
3. Similarly, **B** chooses a random $1 < t <_\ell$ and computes his public key $P_B = (V_t(a, b), V_{-t}(a, b)) = (s_t, s_{-t})$, by running Algorithm 3 with inputs t and a, b.
4. Both **A** and **B** computes common key as follows:
   (i) **A** acquires **B**'s public key P_B and constructs $g_t(x) = x^3 - s_t x^2 + s_{-t} x - 1$. Then she computes K by running Algorithm 3 with inputs $P_B = (s_t, s_{-t})$ and her private key e such that,
   $$K = P_{AB} = (V_e(P_B), V_{-e}(P_B)) = (V_e(s_t, s_{-t}), V_{-e}(s_t, s_{-t})) = (s_{et}, s_{-et}).$$
   (ii) **B** does the similar things using his private key t to compute common key:
   $$K = P_{BA} = (V_t(P_A), V_{-t}(P_A)) = (V_t(s_e, s_{-e}), V_{-t}(s_e, s_{-e})) = (s_{te}, s_{-te}).$$

3.5.2. GH-PKC ElGamal type encryption scheme

The GH-ElGamal type encryption scheme was introduced by M. Ashraf and B.B. Kirlar in, 2013 [14] and it is as follows:

1. *System public parameters.* Let, $G = \langle \alpha \rangle \subset \mathbb{F}_{q^3}^*$, $ord(\alpha) = \ell | (q^2 + q + 1)$, and $g(x) = x^3 - ax^2 + bx - 1$, be the characteristic polynomial of α.
2. **A** chooses a random $1 < e < \ell$ and computes her public key $P_A = V_e(a, b), V_{-e}(a, b) = (s_e, s_{-e})$ by running Algorithm 3 with inputs a, b and her private key e. Similarly, **B** computes his public key $P_B = (V_t(a, b), V_{-t}(a, b)) = (s_t, s_{-t})$, by running Algorithm 3 with inputs a, b and his private key $1 < t < \ell$.
3. Private key: $1 < e < \ell$ and $1 < r < \ell$ are private keys of **A** and **B**, respectively.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

Table 4
Comparison of proposed encryption scheme with the similar ones.

	ElGamal	GH-RSA-type	XTR-ElGamal	Proposed
Encryption	$(2+2(9r^{\log 3}+13r-13)\log n)\mathbf{M}+4r^{\log 3}\log n\,\mathbf{S}$	$10\,\mathbf{M}'$	$10.4\log n\,\mathbf{M}$	$(10.4\log n)\,\mathbf{M}$
Decryption	$(2+2(9r^{\log 3}+13r-13)\log n)\mathbf{M}+4r^{\log 3}\log n\,\mathbf{S}$	$12\log N\,\mathbf{M}'$	$5.2\log n\,\mathbf{M}$	$(5.2\log n)\,\mathbf{M}$
Throughput	m_1	m_1, m_2	m_1	m_1
Comm. overhead in bits	$\|q\|$ with $q=p^r$	$2\|N\|$	$\|q\|$ with $q=p^2$	$2\|q\|$
Decryption overhead	$1I$	9 decryption keys	Symmetric decipher	$1I$

Where N is product of two distinct primes used in GH RSA encryption scheme [13] and I is field inversion.

4. *Encryption:* **A** encrypts a message $M \in \mathbb{F}_q^*$ as follows:
   (i) **A** randomly selects an ephemeral private key $t \in \mathbb{Z}$ satisfying $0 < t < \ell$ such that $\gcd(t, \ell) = 1$ and computes her ephemeral public key $(s_t, s_{-t}) \in \mathbb{F}_q^2$.
   (ii) **A** computes $(s_{tr}, s_{-tr}) = (s_t(s_r, s_{-r}), s_{-t}(s_r, s_{-r})) \in \mathbb{F}_q^2$ using the static public key $P_{\mathbf{B}} = (s_r, s_{-r})$ of **B**.
   (iii) **A** computes $\bar{K} = (s_{tr} + s_{-tr})$ and $c = M\bar{K}$ and, sends the ciphertext $C = (s_t, s_{-t}, c)$ to **B**.
5. *Decryption:* **B** decrypts C as follows:
   B runs Algorithm 3 with inputs, mask (s_t, s_{-t}) and his private key r to compute session key $(V_r(s_t, s_{-t}), V_{-r}(s_t, s_{-t})) = (s_{tr}, s_{-tr})$, and computes $\bar{K} = (s_{tr} + s_{-tr})$ and $M = c\bar{K}^{-1}$.

Remark 7 (*Security Analysis*)**.** The security of key exchange in the proposed encryption scheme depends on the difficulty of solving 3-LFSR-DLP, 3-LFSR-DHP and 3-LFSR-DDHP whereas semantic security depends upon splitting $z \in \mathbb{F}_q$ into two elements $(X, Y) \in \mathbb{F}_q^2$ such that $z = X + Y$, where $s_{tr} \mapsto X$ and $s_{-tr} \mapsto Y$. Moreover, $tr \in \mathbb{Z}_\ell$ and $\mathbb{Z}_\ell$ is chosen large enough so that the brute force attack becomes infeasible. The details are given in Remark 17.

Let $m_1 \in \mathbb{F}_q^*$, $m_2 \in \mathbb{F}_q^*$ be two messages. The comparison of related encryption schemes is given in Table 4.

3.5.3. GH-Nyberg–Rueppel type digital signature algorithm based on GH-ElGamal encryption scheme

Based upon GH-ElGamal encryption scheme by M. Ashraf and B.B. Kırlar given at Section 3.5.2 they also introduced GH-NR-DSA in [14]. Let **A** send signed message $M \in \mathbb{F}_q^*$ to **B** and **B** verifies it. They proceed as follows:

1. *System Public parameters.* Let $G = \langle\alpha\rangle \subset \mathbb{F}_{q^3}^*$, $ord(\alpha) = \ell | (q^2+q+1)$, $g(x) = x^3 - ax^2 + bx - 1$, be the characteristic polynomial of α, and $H : G \to \mathbb{Z}_\ell$ valued hash function.
2. *Public keys:*
   (i) **A** selects random $0 < e < \ell$ and computes public key $P_{\mathbf{A}} = (V_e(a, b), V_{-e}(a, b)) = (s_e, s_{-e})$ by running Algorithm 3 with inputs a, b and her private key e.
   (ii) Similarly, **B** selects random $0 < r < \ell$ and computes his public key $P_{\mathbf{B}} = (V_r(a, b), V_{-r}(a, b)) = (s_r, s_{-r})$ by running Algorithm 3 with inputs a, b and his private key r.
3. *Private keys:* The private keys of both **A** and **B** are e and r, respectively.
4. *Signature*: **A** signs the message M containing agreed upon redundancy as follows:
   (i) **A** randomly selects ephemeral private key $0 < t < \ell$ and computes ephemeral public key $(s_t, s_{-t}) = (V_t(a, b), V_{-t}(a, b))$, and symmetric key $(V_t(s_r, s_{-r}), V_{-t}(s_r, s_{-r})) = (s_{tr}, s_{-tr})$ by running Algorithm 3 with inputs a, b, t and s_r, s_{-r}.
   (ii) Computes the hash value of M, i.e., $h = H(M) \bmod \ell$ and computes $\bar{K} = s_{tr} + s_{-tr}$.
   (iii) Computes the ciphertext $C = M\bar{K}$ by encrypting the message M using GH-ElGamal encryption scheme [14].
   (iv) Computes $n = t - he \bmod \ell$.
   (v) **A** sends the signature (n, s_t, s_{-t}, C) to **B**.
5. *Verification*: **B** recovers message M and verifies **A**'s signature as follows:
   (i) Decrypts C to $M : M = C\bar{K}^{-1}$. If M contains agreed upon redundancy then computes $h = H(M) \bmod \ell$,
   (ii) B computes (s_{he}, s_{-he}) from h and **A**'s static public key (s_e, s_{-e}) by running Algorithm 3. Note that as stated in Remark 6 the Algorithm 3 computes $S_{he} = (s_{he-2}, s_{he-1}, s_{he})$.
   (iii) Computes (s_{n+he}) and $(s_{-(n+he)})$ by running Algorithm 1 with input $k = 3$, and vector S_{he}, n.
   (iv) Checks whether $(s_{n+he}, s_{-(n+he)})$ is equal to (s_t, s_{-t}). If yes, **B** accepts.

Let $m_1 \in \mathbb{F}_q^*$, $m_2 \in \mathbb{F}_q^*$ be two messages and P be the point on given elliptic curve. The comparison of related digital signatures is given in Table 5.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

Table 5
Comparison of GH-NR-DSA with the similar DSAs.

	GH-DSA	XTR-NR-DSA	EC-DSA	GH-NR-DSA
Double exponentiation	$(s_{c(h-dl)}, s_{-c(h-dl)})$	(s_{s+hk})	$(u_1+u_2d)P$	$(s_{n+eh}, s_{-(n+eh)})$
Cost	$16\log\ell\,\mathbf{M}+16\log\ell\,\mathbf{S}$	$6\log\ell\,\mathbf{M}$	$7\log\ell\,\mathbf{M}+3.7\log\ell\,\mathbf{S}$	$12\log\ell\,\mathbf{M}$
Throughput	m_1, m_2	m_1	m_1	m_1
Message recovery	No	Yes	No	Yes
Comm. overhead in bits	$2\|q\|+\|H\|$	$\|q\|+\|H\|$	$\|p\|+\|H\|$	$\|q\|+\|H\|$

3.6. GH-PKC: special Case $k = 3$, p-arbitrary, $q = p^2$

We keep the same notations above and now assume that $ord(\alpha) = \ell|(p^2 - p + 1)$. Note that $(q^2 + q + 1) = (p^2 - p + 1)(p^2 + p + 1)$. In the case $q = p^2$, the important point is that $\mathrm{Tr}(\alpha^{-m}) = \mathrm{Tr}(\alpha^m)^p$ for any integer m which was shown in [5]. Therefore for any integer m we get,

$$g_m(x) = \prod_{i=0}^{2}(x - (\alpha^m)^{q^i}) = x^3 - s_m x^2 + s_m^p x - 1,$$

where $s_m = \mathrm{Tr}(\alpha^m)$.

The LFSR sequence $\{s_m\}$ associated to $g(x) = x^3 - ax^2 + bx - 1$, with initial conditions $\{3, a, a^2 - 2b\}$ gives us the general term:

$$s_m = V_m(a, b) = \mathrm{Tr}(\alpha^m), \quad \text{and}$$
$$s_{-m} = V_{-m}(a, b) = V_m(a, b)^p = \mathrm{Tr}(\alpha^m)^p = s_m^p.$$

Properties: For all integers m and n the following properties are consequences in [5] and they are the special case of GH-PKC;

(i) $V_{-m}(a, b) = V_m(a, b)^p$.
(ii) $V_m(V_n(a, b), V_n(a, b)^p) = V_{mn}(a, b) = s_{mn}$.
(iii) $s_{2m} = s_m^2 - 2s_m^p$
(iv) $s_{m+n} = s_m s_n - s_{m-n}s_n^p + s_{m-2n}$.

Note that the advantage of special conditions on α and q is that $g(x)$ can be realized only by one element that is $\mathrm{Tr}(\alpha)$. However, the same Algorithm 3, given in GH-PKC, can be used to compute $\mathrm{Tr}(\alpha^m)$, given $\mathrm{Tr}(\alpha)$ and m, for the special case as well. Now we discuss the cryptographic protocols for the special case.

3.6.1. GH-DH type key exchange: special case

1. *System public parameters.* Let, $G = \langle\alpha\rangle \subset \mathbb{F}_{q^3}^*$, p-arbitrary, $q = p^2$, $ord(\alpha) = \ell|(p^2 - p + 1)$, and $g(x) = x^3 - ax^2 + bx - 1$, be the characteristic polynomial of α.
2. Both **A** and **B** choose random $1 < e < \ell$ and $1 < r < \ell$ respectively as their private keys.
3. **A** computes her public key $P_A = V_e(a, b) = s_e$, by applying Algorithm 3 with inputs e and a, b.
4. Similarly, **B** computes his public key $P_B = V_r(a, b) = s_r$, by applying Algorithm 3 with inputs r and a, b.
5. Both **A** and **B** compute their common key as follows:
   (i) **A** acquires **B**'s public key P_B and constructs $g_r(x) = x^3 - s_r x^2 + s_r^p x - 1$. Then she computes common key by running Algorithm 3 with inputs $P_B = (s_r, s_r^p)$ and her private key e such that,
   $$K = P_{AB} = V_e(s_r, s_r^p) = V_{er}(a, b) = s_{er}.$$
   (ii) **B** does the similar things using his private key r to compute common key:
   $$K = P_{BA} = V_r(s_e, s_e^p) = V_{re}(a, b) = s_{re}.$$

3.6.2. GH-ElGamal type encryption: special case

1. *System public parameters.* Let, $G = \langle\alpha\rangle \subset \mathbb{F}_{q^3}^*$, p-arbitrary, $q = p^2$, $ord(\alpha) = \ell|(p^2 - p + 1)$, and $g(x) = x^3 - ax^2 + bx - 1$, be the characteristic polynomial of α.
2. **B**'s static public key $P_B = V_r(a, b) = s_r$, static private key: random $1 < r < \ell$.
3. Assumption: message $M \in G$.
4. *Encryption:* **A** encrypts message $M \in \mathbb{F}_q^*$ and sends to **B** as follows:
   (i) Chooses a random private ephemeral key $1 < t < \ell$, computes ephemeral public key $V_t(a, b) = s_t$, and session key $K = V_t(s_r, s_r^p) = V_{tr}(a, b) = s_{tr}$ by running Algorithm 3 with inputs a, b, t and s_r, s_r^p.
   (ii) Encrypts message M by computing $c = MK = Ms_{tr}$.
   (iii) **A** sends ciphertext $C = (s_t, c)$ to **B**.
5. *Decryption:* **B** decrypts C by computing session key K with the public ephemeral key s_t and his private key r : $K = V_r(s_t, s_t^p) = V_{rt}(a, b) = s_{rt}$, and, decrypts c to obtain M : $M = cK^{-1} = cs_{rt}^{-1}$.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

3.6.3. GH-Nyberg–Rueppel type signature: special case

In the special case $q = p^2$, the GH-NR type signature given above in Section 3.5.3 namely, (n, s_r, s_{-r}, C) will be changed to (n, s_r, C) as we do not need to include s_{-r} in the signature because $s_{-r} = s_r^p$. So, we get a shorter signature by applying a special condition for $q = p^2$ above.

3.6.4. Lenstra and Verheul–Nyberg–Rueppel type signature algorithm

We would like to recall a work done by Lenstra and Verheul "The XTR Public Key system" in 2000. In their paper the public key cryptosystem for the parameters $k = 3, q = p^2$ is discussed. Note that the basic strength of XTR, that is the GH-PKC special case, is the underlying arithmetic operations. Moreover the main contribution of the XTR public key system to the literature as we see, is the Nyberg– Rueppel type signature scheme which was not discussed before in the literature. In the light of our note we now want to state their algorithm below.

Algorithm 4 to compute $\mathrm{Tr}(\alpha^n\alpha^{mh})$ for the given: $\mathrm{Tr}(\alpha) = s$, (s_{m-1}, s_m, s_{m+1}), $n, h \in \mathbb{Z}_\ell$, $m \in \mathbb{Z}$ is unknown.

Require: $n, h, S_m = (s_{m-1}, s_m, s_{m+1})$, $\mathrm{Tr}(\alpha) = s$.
Ensure: s_{n+mh}.
1: Compute $e = n/h \pmod{\ell}$,
2: Use Algorithm 3 to compute (s_{e-1}, s_e, s_{e+1}),
3: Based on s and (s_{e-1}, s_e, s_{e+1}) compute;

$$A = D^{-1}\begin{pmatrix} 2s^2 - 6s^p & 2s^{2p} + 3s - s^{p+2} & s^{p+1} - 9 \\ 2s^{2p} + 3s - s^{p+2} & (s^2 - 2s^p)^{p+1} - 9 & (2s^{2p} + 3s - s^{p+2})^p \\ s^{p+1} - 9 & (2s^{2p} + 3s - s^{p+2})^p & (2s^2 - 6s^p)^p \end{pmatrix}\begin{pmatrix} s_{e-1} \\ se \\ s_{e+1} \end{pmatrix}$$

where, $D = s^{2p+2} + 18s^{p+1} - 4(s^{3p} + s^3) - 27$,
4: Compute $(s_{m-1}, s_m, s_{m+1})A = s_{e+m}$,
5: $C \leftarrow V_h(s_{e+m}, s^p_{e+m}) = s_{h(e+m)} = s_{n+mh}$.
6: **return** (C).

Remark 8. The $\mathrm{Tr}(\alpha^n\alpha^{mh})$ can be computed in $8\log_2(n/h(\mathrm{mod}\ \ell)) + 8\log_2(h) + 34$ multiplications in $\mathbb{F}_p$.

3.7. Simultaneous double exponentiation for cubic extensions

The double exponentiation algorithm for cubic extensions and 4th extension were introduced by Stam and Lenstra [8] and Koray Karabina [9]. Let $G = \langle\alpha\rangle \subset \mathbb{F}_{q^3}$, $q = p^2$, $ord(\alpha) = \ell|(p^2-p+1)$ and $s_u = \mathrm{Tr}(\alpha^u)$ for $u \in \mathbb{Z}$. Let $0 < a, b, n, m < \ell$ be integers then computing s_{bn+am}, given $s_n, s_m,\ S_{n,m} = [s_{n-m}, s_{n-2m}]$ and positive integers a, b. Let $u = n, v = m, d = b$ and $e = a$, then we can write $ud+ve = bn+am$. The sum $d+e$ is decreased until $d = e$ in such a way that $ud+ve =$ constant and $d = e$. At the point $ud + vd = d(u + v) =$ constant, single exponentiation is applied to compute $s_{d(u+v)}$. The update rules and related algorithm are given in Table 6.

Algorithm 5 Double Exponentiation for NR-GH-DSA

Require: $(a > 0, b > 0) \in \ell$, s_l, s_k, s_{k-l}, and s_{k-2l}.
Ensure: s_{bk+al}
1: $f_2 \leftarrow 1, d \leftarrow b, e \leftarrow a, u \leftarrow k, v \leftarrow l$
2: **while** d and e are both even **do**
3: $d \leftarrow d/2, e \leftarrow e/2, f_2 \leftarrow 2f_2$
4: **end while**
5: **while** $d \neq e$ **do**
6: Execute the applicable rule in Table 6
7: **end while**
8: compute $s_{d(u+v)}$ using Algorithm [13, Algorithm 1] with inputs d and s_{u+v}.
9: **return** $(s_{f_2(d(u+v))})$

Remark 9. The Algorithm 5 can compute s_{bk+al} in $6\log(\max(a, b))$ multiplications in $\mathbb{F}_p$ given s_l, $0 < a, b < \ell$, s_k, s_{k-l}, and s_{k-2l}. Note that Algorithm 5 can be used for single exponentiation for given s_1 and $u \in \ell$. In this case, single exponentiation takes $5.2\log u$ multiplications in $\mathbb{F}_p$.

Now we describe Lenstra and Verheul's Nyberg–Rueppel Type Digital Signature Scheme.

1. *System public parameters.* Let, $G = \langle\alpha\rangle \subset \mathbb{F}^*_{q^3}$, p-arbitrary, $q = p^2$, $ord(\alpha) = \ell|(p^2 - p + 1)$, and $g(x) = x^3 - ax^2 + bx - 1$, be the characteristic polynomial of α, and $H : G \to \mathbb{Z}_\ell$ valued hash function.

Table 6
The rules for double exponentiation.

Rule	Condition	d	e	u	v	s_u	s_v	s_{u-v}	s_{u-2v}
If $d > e$									
R_1	$d \leq 4e$	e	$d-e$	$u+v$	u	s_{u+v}	s_u	s_v	$s_{(v-u)}$
R_2	$d \equiv 0 (\mathrm{mod}\ 2)$	$d/2$	e	$2u$	v	s_{2u}	s_v	s_{2u-v}	$s_{2(u-v)}$
R_3	e odd	$(d-e)/2$	e	$2u$	$u+v$	s_{2u}	s_{u+v}	s_{u-v}	s_{-2v}
R_4	$e \equiv 0 (\mathrm{mod}\ 2)$	d	$e/2$	$2v$	u	s_{2v}	s_u	s_{2v-u}	$s_{2(v-u)}$
Else									
Sub	$e > d$	e	d	v	u	s_v	s_u	s_{v-u}	s_{2v-u}

2. **A**'s public key $P_A = V_m(a,b) = (s_m) := (s_{m-1}, s_m, s_{m+1})$, private key: random $1 < m <\in \mathbb{Z}_\ell$ and **B**'s public key $P_B = V_e(a,b) = (s_e) := (s_{e-1}, s_e, s_{e+1})$, private key: random $1 < e <\in \mathbb{Z}_\ell$.
3. Assumption: message $M \in G$.
4. *Signature*: **A** signs the message M as follows:
   (i) Chooses a random $1 < r < \ell$ and computes s_r.
   (ii) Based on mask s_r determine session key $K = V_m(s_r, s_r^p) = V_{mr}(a,b) = s_{mr}$.
   (iii) Encrypts message $M \longrightarrow C$ using K and agreed upon symmetric encryption scheme.
   (iv) Computes $h = H(C) \bmod \ell$
   (v) Computes $n = (r - mh) (\mathrm{mod}\ \ell)$.
   (vi) **A**'s resulting signature on M is (n, C).
5. *Verification*: **B** verifies **A**'s signature as follows:
   (i) **B** checks whether $0 < n < \ell$; if not failure.
   (ii) Computes $h = H(C) \bmod \ell$,
   (iii) Running Algorithm 4 above with inputs h, n, (s_{m-1}, s_m, s_{m+1}), s, computes $s_{n+mh} = \mathrm{Tr}(\alpha^n \alpha^{mh})$.
   (iv) Computes symmetric key $\acute{K}$ based upon s_{n+mh} and decrypts C to M using this $\acute{K}$ and agreed upon symmetric scheme. If M contains required redundancy, **B** accepts.

Remark 10. The GH-NR digital signature algorithm we have given above in the GH-PKC case is due to M. Ashraf and B.B. Kirlar [14]. When we compare their algorithm with Lenstra and Verheul's algorithm we note the following:

(i) In one the digital signature is (n, C) and in the other the signature is (n, s_t, s_{-t}, C).
(ii) The static public key in one is s_{e-1}, s_e, s_{e+1} whereas in the other it is shorter i.e; (s_e, s_{-e}).
(iii) In one the ElGamal type encryption is used whereas in other agreed upon symmetric encryption is used.
(iv) With the introduction of simultaneous double exponentiation for the GH-PKC case [14], the resulting cost of computing mixed term s_{i+j} is similar for both XTR and GH digital signatures.

3.8. GG public key cryptosystem

We will introduce two cases given by K. Giuliani and G. Gong in 2003 [3,4] and fill the gap in this PKC by introducing an GG-ElGamal type encryption scheme and Nyberg–Rueppel type digital signatures.

Case 1: $k = 5$, p-arbitrary, $q = p^u$, $u \geq 1 \in \mathbb{Z}$.
Case 2: $k = 5$, p-arbitrary, $q = p^2$.

3.8.1. GG PKC: case $k = 5$, p-arbitrary, $q = p^u$

Let, $G = \langle \alpha \rangle \cong \mathbb{Z}_\ell \subset \mathbb{F}_{q^5}^*$, $ord(\alpha) = \ell | (q^4 + q^3 + q^2 + q + 1)$, note that $\Phi_5(q) = (q^4 + q^3 + q^2 + q + 1)$ is the $5th$ cyclotomic polynomial evaluated at q, then let $g(x)$ be the characteristic polynomial of α:

$$g(x) = \prod_{i=0}^{4} (x - \alpha^{q^i}) = x^5 - ax^4 + bx^3 - cx^2 + dx - 1.$$

One can check that,

$$a = \mathrm{Tr}(\alpha), \qquad d = \mathrm{Tr}(\alpha^{-1}), \qquad b = \sum_{0 \leq i < j \leq 4} \alpha^{q^i + q^j}, \qquad c = \sum_{0 \leq i < j \leq 4} \alpha^{-q^i - q^j}.$$

For all integers m, let $g_m(x)$ be the characteristic polynomial of α^m:

$$g_m(x) = \prod_{i=0}^{4} (x - (\alpha^m)^{q^i}) = x^5 - a_m x^4 + b_m x^3 - c_m x^2 + d_m x - 1.$$

One can check that,

$$a_m = \mathrm{Tr}(\alpha^m), \qquad d_m = a_{-m} = \mathrm{Tr}(\alpha^{-m}),$$
$$b_m = \sum_{0\le i<j\le 4} \alpha^{mq^i+mq^j}, \qquad c_m = b_{-m} = \sum_{0\le i<j\le 4} \alpha^{-mq^i-mq^j}.$$

So let, $V_m(a,b,c,d) = a_m = s_m$, and $\widehat{V}_m(a,b,c,d) = b_m = \hat{s}_m$. In particular, $s_0 = 5, s_1 = a, s_2 = a^2 - 2b, s_3 = a^3 - 3ab + 3c, s_4 = s_2^2 - 2b^2 - 4d + 4c$. Then we have,

$$g_m(x) = x^5 - s_m x^4 + \hat{s}_m x^3 - \hat{s}_{-m} x^2 + s_{-m} x - 1.$$

Now if we replace α above by $\beta = \alpha^r$ for any integer r, then we have the characteristic polynomial for β^m in the form,

$$h_m(x) = \prod_{i=0}^{4}(x - (\beta^m)^{q^i}) = \prod_{i=0}^{4}(x - (\alpha^{rm})^{q^i}) = g_{rm}(x).$$

Therefore,

$$V_m(s_r, \hat{s}_r, \hat{s}_{-r}, s_{-r}) = V_{mr}(a,b,c,d) = s_{mr},$$
$$\widehat{V}_m(s_r, \hat{s}_r, \hat{s}_{-r}, s_{-r}) = \widehat{V}_{mr}(a,b,c,d) = \hat{s}_{mr}.$$

Properties: For all integers m, n following are the properties:

(i) $s_{2n} = s_n^2 - 2\hat{s}_n$, and $\hat{s}_{2n} = \hat{s}_n^2 + 2s_{-n} - 2s_n\hat{s}_{-n}$.
(ii) $s_{3n} = s_n^3 - 3s_n\hat{s}_n + 3\hat{s}_{-n}$.
(iii) $\hat{s}_{3n} = \hat{s}_n^3 - 3s_n^3\hat{s}_n - 3s_n\hat{s}_n\hat{s}_{-n} + 3s_n^2 s_n + 3\hat{s}_{-2n} - 3s_n$.
(iv) $s_{n+m} = s_n s_m - s_{n-m}\hat{s}_m + s_{n-2m}\hat{s}_{-m} - s_{n-3m}s_{-m} + s_{n-4m}$.
(v) $\hat{s}_n\hat{s}_m - s_{-m}\hat{s}_{n-m} + 3\hat{s}_{n+m} = s_n s_m s_{n+m} s_{n-2m} s_{n-m} + s_{2n-3m} - s_{n+2m}s_n - s_{2n+m}s_m + s_{n+m}^2$.
(vi) $s_{n+2m} = s_{n+m}\hat{s}_m - s_n\hat{s}_m + s_{n-m}\hat{s}_{-m} - s_{n-2m}s_{-m} + s_{n-3m}$.

Using above properties the algorithm introduced by K. Giuliani and G. Gong in [3,4] to compute nth terms of sequences $\{s_u, \hat{s}_u\}$ is as follows:

Algorithm 6 to compute $s_n = \mathrm{Tr}(\alpha^n)$ and $\hat{s}_n = \sum_{0\le i<j\le 4} \alpha^{nq^i+nq^j}$ using triple-add-subtract

Require: $n \in \mathbb{Z}_\ell : n = \sum_{j=0}^{l} n_j 3^j \in \mathbb{Z}^+, n_j \in \{-1, 0, 1\}$ with $n_l \neq 0$, and initial conditions $(s_{-1}, 5, s_1, s_2, s_3)$, $(\hat{s}_{-1}, 5, \hat{s}_1, \hat{s}_2, \hat{s}_3)$.
Ensure: $(s_{n-2}, s_{n-1}, s_n, s_{n+1}, s_{n+2})$, $(\hat{s}_{n-2}, \hat{s}_{n-1}, \hat{s}_n, \hat{s}_{n+1}, \hat{s}_{n+2})$
1: $u \leftarrow 1, S \leftarrow (s_{-1}, 5, s_1, s_2, s_3), \hat{S} \leftarrow (\hat{s}_{-1}, 5, \hat{s}_1, \hat{s}_2, \hat{s}_3)$
2: **for** $i \leftarrow 0$ **to** l **do**
3: **if** $n_i = -1$ **then**
4: $S \leftarrow (s_{3u-3}, s_{3u-2}, s_{3u-1}, s_{3u}, s_{3u+1})$,
5: $\hat{S} \leftarrow (\hat{s}_{3u-3}, \hat{s}_{3u-2}, \hat{s}_{3u-1}, \hat{s}_{3u}, \hat{s}_{3u+1})$.
6: **else if** $n_i = 0$ **then**
7: $S \leftarrow (s_{3u-2}, s_{3u-1}, s_{3u}, s_{3u+1}, s_{3u+2})$,
8: $\hat{S} \leftarrow (\hat{s}_{3u-2}, \hat{s}_{3u-1}, \hat{s}_{3u}, \hat{s}_{3u+1}, \hat{s}_{3u+2})$.
9: **else**
10: $S \leftarrow (s_{3u-1}, s_{3u}, s_{3u+1}, s_{3u+2}, s_{3u+3})$,
11: $\hat{S} \leftarrow (\hat{s}_{3u-1}, \hat{s}_{3u}, \hat{s}_{3u+1}, \hat{s}_{3u+2}, \hat{s}_{3u+3})$.
12: **end if**
13: $u = 3u + n_i$
14: **end for**
15: **return** $((s_{n-2}, s_{n-1}, s_n, s_{n+1}, s_{n+2}), (\hat{s}_{n-2}, \hat{s}_{n-1}, \hat{s}_n, \hat{s}_{n+1}, \hat{s}_{n+2})) = (S_n, \hat{S}_n)$.

The average cost of computations is approximately $108.5 \log n$ multiplications, $13 \log n$ scalar multiplications and $280.1 \log n$ additions.

For the case $k = 5$, p-arbitrary, $q = p^u$, it is not difficult to give the cryptographic protocols to have a trace based cryptosystem. However, it is not interesting from the implementation point of view and therefore we leave this and discuss the next special case together with cryptographic protocols.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

3.8.2. GG PKC: special case, $k = 5$, p-arbitrary, $q = p^2$

K. Giuliani and G. Gong introduced special case for their cryptosystem in 2004 [4]. Let $G = \langle\alpha\rangle \subset \mathbb{F}_{q^5}^*$, p-arbitrary, $q = p^2$, $ord(\alpha) = \ell|(p^4 - p^3 + p^2 - p + 1)|(q^4 + q^3 + q^2 + q + 1)$. We keep the notations above and note that in this special case for any integer m we have,

$$\begin{aligned}\mathrm{Tr}(\alpha^{-m}) = \mathrm{Tr}(\alpha^m)^p &= s_m^p,\\ \sum_{0\le i<j\le 4} \alpha^{-mq^i - mq^j} = \mathrm{Tr}(\alpha^{-m(p+1)} + \alpha^{-m(p^2+1)}) &= \mathrm{Tr}(\alpha^{m(p+1)} + \alpha^{m(p^2+1)})^p\\ &= \hat{s}_m^p.\end{aligned}$$

Thus, for any integer m the characteristic polynomial $g_m(x)$ of α^m is

$$g_m(x) = \prod_{i=0}^{4}(x - \alpha^{mq^i}) = x^5 - s_m x^4 + \hat{s}_m x^3 - \hat{s}_m^p x^2 + s_m^p x - 1.$$

Properties: For all integers m and n the following properties are proved by K. Giuliani and G. Gong in [3,4] and we list them here. Let $\alpha \in \mathbb{F}_{q^5}$, having order $\ell|(p^4 - p^3 + p^2 - p + 1)$, and characteristic polynomial $g(x) = x^5 - ax^4 + bx^3 - cx^2 + dx - 1$, then,

(i) $s_m = \mathrm{Tr}(\alpha^m)$, $s_{-m} = s_m^p$ and,
$\hat{s}_m = \mathrm{Tr}(\alpha^{m(p+1)} + \alpha^{m(p^2+1)})$, $\hat{s}_{-m} = \hat{s}_m^p$.
(ii) $V_m(s_n, \hat{s}_n, \hat{s}_n^p, s_n^p) = s_{mn}$, and
$\widehat{V}_m(s_n, \hat{s}_n, \hat{s}_n^p, s_n^p) = \hat{s}_{mn}$.
(iii) $s_{2n} = s_n^2 - 2\hat{s}_n$, and
$\hat{s}_{2n} = \hat{s}_n^2 + 2s_{-n} - 2s_n\hat{s}_n^p$.
(iv) $s_{3n} = s_n^3 - 3s_n\hat{s}_n + 3\hat{s}_n^p$.
(v) $\hat{s}_{3n} = \hat{s}_n^3 - 3s_n^3\hat{s}_n - 3s_n\hat{s}_n\hat{s}_n^p + 3s_n^2 s_n + 3\hat{s}_{2n}^p - 3s_n$.
(vi) $s_{n+m} = s_n s_m - s_{n-m}\hat{s}_m + s_{n-2m}\hat{s}_m^p - s_{n-3m}s_m^p + s_{n-4m}$.
(vii) $\hat{s}_n\hat{s}_m - s_{-m}\hat{s}_{n-m} + 3\hat{s}_{n+m} = s_n s_m s_{n+m} s_{n-2m} s_{n-m} + s_{2n-3m} - s_{n+2m}s_n - s_{2n+m}s_m + s_{n+m}^2$.
(viii) $s_{n+2m} = s_{n+m}\hat{s}_m - s_n\hat{s}_m + s_{n-m}\hat{s}_m^p - s_{n-2m}s_m^p + s_{n-3m}$.

Remark 11. Algorithm 6 also computes the *nth* term just by replacing $s_{-1} = s_1^p$ and $\hat{s}_{-1} = \hat{s}_1^p$ and accordingly in subsequent computations. Actually the GG algorithm computes for a given m and $(s_1, \hat{s}_1)$; the terms $(s_{m-4}, s_{m-3}, s_{m-2}, s_{m-1}, s_m)$ and $(\hat{s}_{m-4}, \hat{s}_{m-3}, \hat{s}_{m-2}, \hat{s}_{m-1}, \hat{s}_m)$.

With this setup, now we discuss the cryptographic protocols and add the ElGamal type encryption scheme and Nyberg–Rueppel type digital signature scheme to GG-PKC.

3.8.3. GG-DH type key exchange: special case

1. *System public parameters:* Let, $G = \langle\alpha\rangle \subset \mathbb{F}_{q^5}^*$, p-arbitrary, $q = p^2$, $ord(\alpha) = \ell|(p^4 - p^3 + p^2 - p + 1)$, and $g(x) = x^5 - ax^4 + bx^3 - cx^2 + dx - 1$, be the characteristic polynomial of α.
2. **A** computes her public key $P_A = (s_e, \hat{s}_e)$, by running Algorithm 6 with inputs a, b, c, d and her private key, a random $1 < e < \ell$.
3. Similarly, **B** computes his public key $P_B = (s_r, \hat{s}_r)$, by running Algorithm 6 with inputs a, b, c, d and his private key, a random $1 < r < \ell$.
4. Both A and B compute common key $K = P_{AB} = P_{BA} = (s_{er}, \hat{s}_{re})$ as follows:
   (i) **A** acquires **B**'s public key P_B and constructs
   $$g_r(x) = x^5 - s_r x^4 + \hat{s}_r x^3 - \hat{s}_r^p x^2 + s_r^p x - 1.$$
   Then she computes common key by running Algorithm 6 with inputs $P_B = (s_r, \hat{s}_r, \hat{s}_r^p, s_r^p)$ and her private key e such that,
   $$K = P_{AB} = (V_e(s_r, \hat{s}_r, \hat{s}_r^p, s_r^p), \widehat{V}_e(s_r, \hat{s}_r, \hat{s}_r^p, s_r^p)) = (s_{er}, \hat{s}_{er}).$$
   (ii) **B** does the similar things using his private key r to compute the common key:
   $$K = P_{BA} = (V_r(s_e, \hat{s}_e, \hat{s}_e^p, s_e^p), \widehat{V}_r(s_e, \hat{s}_e, \hat{s}_e^p, s_e^p)) = (s_{re}, \hat{s}_{re}).$$

3.8.4. GG-ElGamal type encryption scheme: special case

For the sake of completeness of GG-PKC, the ElGamal type encryption scheme is being introduced for the first time. **A** wants to send a message $M \in \mathbb{F}_q^*$ to **B**, both proceed as follows:

1. *System public parameters:* Let, $G = \langle\alpha\rangle \subset \mathbb{F}_{q^5}^*$, p-arbitrary, $q = p^2$, $ord(\alpha) = \ell|(p^4 - p^3 + p^2 - p + 1)$, and $g(x) = x^5 - ax^4 + bx^3 - cx^2 + dx - 1$, be the characteristic polynomial of α.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

Table 7
GG-ElGamal type encryption scheme analysis.

Process	Cost
Encryption	$\approx 120 \log p$ **M**
Decryption	$\approx 120 \log p$ **M**
Comm. overhead	2 elements over $\mathbb{F}_q$
Public key size	2 elements over $\mathbb{F}_q$

2. *Public keys:* Following are the public keys of **A** and **B**:
   (i) **A** selects random $0 < e < \ell$, and computes her public key $P_A = (s_e, \hat{s}_e)$ by running Algorithm 6 with inputs a, b, c, d and e.
   (ii) Similarly, **B** selects random $0 < r < \ell$, and computes public key $P_B = (s_r, \hat{s}_r)$ by running Algorithm 6 with inputs a, b, c, d and r.
3. *Private keys:* e and r are the private keys of **A** and **B** respectively.
4. *Encryption:* **A** encrypts message M and sends to **B** as follows:
   (i) **A** randomly selects ephemeral private key $0 < t < \ell$ and computes ephemeral public key $(V_t(a, b, c, d), \hat{V}_t(a, b, c, d)) = (s_t, \hat{s}_t)$ by running Algorithm 6 with inputs a, b, c, d and t.
   (ii) **A** acquires **B**'s public key $P_B = (s_r, \hat{s}_r)$ and computes common key $(s_{tr}, \hat{s}_{tr}) = (V_t(s_r, \hat{s}_r, \hat{s}_r^p, s_r^p), \widehat{V}_t(s_r, \hat{s}_r, \hat{s}_r^p, s_r^p))$ by running Algorithm 6 with inputs $(s_r, \hat{s}_r, \hat{s}_r^p, s_r^p)$ and t. Also computes $\bar{K} = s_{tr} + \hat{s}_{tr}$.
   (iii) **A** computes, $c = M\bar{K}$, then she sends the ciphertext $C = ((s_t, \hat{s}_t), c)$ to **B**.
5. *Decryption:* **B** recovers the message M as follows:
   (i) Using ephemeral public key $(s_t, \hat{s}_t)$ and his private key r, computes common key $(s_{rt}, \hat{s}_{rt}) = V_r(s_t, \hat{s}_t, \hat{s}_t^p, s_t^p), \hat{V}_r(s_t, \hat{s}_t, \hat{s}_t^p, s_t^p)$and $\bar{K} = s_{tr} + \hat{s}_{tr}$ as above.
   (ii) Computes, $M = c\hat{K}^{-1}$.

Remark 12. The semantic security of GG-ElGamal type encryption is computationally equivalent to splitting $z \in \mathbb{F}_q$ into two numbers $(X, Y) \in \mathbb{F}_q^2$ such that $z = X + Y$ where $s_{tr} \mapsto X$ and $s_{-tr} \mapsto Y$. Moreover, $tr \in \mathbb{Z}_\ell$ and $\mathbb{Z}_\ell$ is chosen large enough so that the brute force attack becomes infeasible. The number of such (X, Y) are equal to $\frac{q-1}{2}$ as given in Remark 17.

The analysis of GG-ElGamal type encryption is given in Table 7.

3.8.5. GG-NR type digital signature algorithm based on GG-ElGamal encryption: special case

After introduction of GG-ElGamal type encryption scheme now we introduce Nyberg–Rueppel type digital signature scheme based on GG-ElGamal type encryption scheme. **A** wants to send signed message $M \in \mathbb{F}_q^*$ containing agreed upon redundancy to **B** and **B** verifies it. To do this, **A** and **B** proceed as follows:

1. *System public parameters:* Let, $G = \langle\alpha\rangle \subset \mathbb{F}_{q^5}^*$, p-arbitrary, $q = p^2$, $ord(\alpha) = \ell | (p^4 - p^3 + p^2 - p + 1)$, and $g(x) = x^5 - ax^4 + bx^3 - cx^2 + dx - 1$, be the characteristic polynomial of α and $H : G \to \mathbb{Z}_\ell$ valued hash function.
2. *Public keys:*
   (i) **A** chooses random $0 < e < \ell$, and computes public key $P_{\mathbf{A}} = (s_e, \hat{s}_e)$ by running Algorithm 6 with inputs a, b, c, d, and e.
   (ii) **B** chooses random $0 < r < \ell$, and computes public key $P_{\mathbf{B}} = (s_r, \hat{s}_r)$ by running Algorithm 6 with inputs a, b, c, d, and r.
3. *Private keys:* The private keys of **A** and **B** are e and r, respectively.
4. *Signature*: **A** signs the message M as follows:
   (i) **A** chooses random ephemeral private key $0 < t < \ell$, and computes ephemeral public key $(V_t(a, b, c, d), \widehat{V}_t(a, b, c, d)) = (s_t, \hat{s}_t)$, and common key $(V_t(s_r, \hat{s}_r, \hat{s}_r^p, s_r^p), \widehat{V}_t(s_r, \hat{s}_r, \hat{s}_r^p, s_r^p)) = (s_{tr}, \hat{s}_{tr})$. Also computes $\bar{K} = s_{tr} + \hat{s}_{tr}$.
   (ii) Computes hash value of $M : h = H(M) \bmod \ell$.
   (iii) **A** obtains ciphertext C by encrypting message $M : c = M\bar{K}$.
   (iv) **A** computes $n = t - he \bmod \ell$.
   (v) **A** sends the signature $(n, (s_t, \hat{s}_t), c)$ to **B**.
5. *Verification*: **B** recovers message M and verifies **A**'s signature as follows:
   (i) Checks $0 \le n < \ell$, if not failure.
   (ii) Computes common key $(V_r(s_t, \hat{s}_t, \hat{s}_t^p, s_t^p), \widehat{V}_r(s_t, \hat{s}_t, \hat{s}_t^p, s_t^p)) = (s_{rt}, \hat{s}_{rt})$, $\bar{K} = s_{tr} + \hat{s}_{tr}$ and decrypts c to $M : M = c\bar{K}^{-1}$. If M does not contain agreed upon redundancy then failure.
   (iii) Computes $h = H(M) \bmod \ell$, and $(s_{he}, \hat{s}_{he}) = (V_h(s_e, \hat{s}_e, \hat{s}_e^p, s_e^p), \hat{V}_h(s_e, \hat{s}_e, \hat{s}_e^p, s_e^p))$ from h and **A**'s public key $(s_e, \hat{s}_e)$. Note that Algorithm 6 computes vector $S_{he} = (s_{he-4}, \ldots, s_{he})$.
   (iv) Computes $(s_{n+he}, \hat{s}_{n+he})$ by running Algorithm 1 with inputs $k = 5, S_{he}, n$.
   (v) **B** checks whether $(s_{n+he}, \hat{s}_{n+he})$ is equal to $(s_t, \hat{s}_t)$, if yes accepts.

The analysis of GG-NR-DSA is given in Table 8.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

Table 8
GG-NR-DSA analysis.

Process	Cost
Signature generation	$\approx 120 \log p$ **M**
Signature verification	$\approx 240 \log p$ **M**
Comm. overhead	3 elements over $\mathbb{F}_q$
Public key size	2 elements over $\mathbb{F}_q$

3.9. Koray Karabina PKC: case $k = 4$, $p = 2$, $q = p^{2u+1}$, $u \geq 1$

K. Karabina introduced an efficient algorithm to compute $\mathrm{Tr}(\alpha^m)$ in [9] for the characteristic 2-case when q is an odd power of 2. However, he has not discussed some of the cryptographic protocols attached to his algorithm $\alpha \to \mathrm{Tr}(\alpha^m)$. We now discuss his algorithm and related protocols for the sake of completeness.

Let $G = \langle \alpha \rangle \subset \mathbb{F}^*_{q^4}$, $t = \sqrt{2q}$, $p = 2$, $q = p^{2u+1}$, $u \geq 1$, $ord(\alpha) = \ell | (q - t + 1) | (q^2 + 1)$. Note that, $\Phi_4(q) = q^2 + 1 = (q + t + 1)(q - t + 1)$, is the 4th cyclotomic polynomial evaluated at q. Let $g(x)$ be the characteristic polynomial of α:

$$g(x) = \prod_{i=0}^{3} (x - \alpha^{q^i}) = x^4 - ax^3 + bx^2 - cx + 1.$$

One can check that $a = \mathrm{Tr}(\alpha)$, $b = \mathrm{Tr}(\alpha^{q+1}) = \mathrm{Tr}(\alpha)^t$, and $c = \mathrm{Tr}(\alpha)$. So $g(x)$ is completely determined by only one variable that is $\mathrm{Tr}(\alpha)$. For any integer m, similarly $g_m(x)$ becomes;

$$g_m(x) = \prod_{i=0}^{3} (x - \alpha^{mq^i}) = x^4 - \mathrm{Tr}(\alpha^m)x^3 + (\mathrm{Tr}(\alpha^m)^t))x^2 - \mathrm{Tr}(\alpha^m)x + 1.$$

Let $V_n : \mathbb{F}_q \to \mathbb{F}_q$ be the function defined by $V_n(a) = \mathrm{Tr}(\beta^n)$, where β is the root of $h(x) = x^4 - ax^3 + a^t x^2 - ax + 1 = 0$. It is clear that V_n is independent of the choice of the root β of $h(x) = 0$. Now replacing α by $\beta = \alpha^m$ above we get for any integer r the characteristic polynomial $h_r(x)$ of β^r;

$$\begin{aligned} h_r(x) &= \prod_{i=0}^{3} (x - \beta^{rq^i}), \\ &= x^4 - \mathrm{Tr}(\alpha^{mr})x^3 + (\mathrm{Tr}(\alpha^{mr}))^t x^2 - \mathrm{Tr}(\alpha^m r)x + 1 = g_{mr}(x). \end{aligned}$$

This in particular gives, $V_m(V_r(a)) = V_{mr}(a) = s_{mr} = \mathrm{Tr}(\alpha^{mr})$.

Properties: For all integers m and n we have the following properties:

(i) $s_n = \mathrm{Tr}(\alpha^n) = V_n(a)$.
(ii) $s_n = s_{-n}$.
(iii) $s_{2u} = s_u^2$.
(iv) $s_m s_n = s_{m+n} + s_{m-n} + s_{m+n(t-1)} + s_{n+m(t-1)}$.
(v) $s_{m+n} = s_m s_n - s_{m-n} s_{nt} + s_{m-2n} s_n - s_{m-3n}$.

K. Karabina's algorithm to compute $\mathrm{Tr}(\alpha^n)$, given $\mathrm{Tr}(\alpha)$, and n, is as follows:

Algorithm 7 Compute $s_n = \mathrm{Tr}(\alpha^n)$, given $s = \mathrm{Tr}(\alpha) \in \mathbb{F}_q$, n, $q = 2^{2r+1}$

Require: $n \in \mathbb{Z}_\ell : n = \sum_{i=0}^{l-1} n_i 2^i$, $n_i \in \{0, 1\}$ with $n_{l-1} \neq 0$, and initial conditions $(s_1, 0, s_1, s_1^2)$.
Ensure: $(s_{n-1}, s_n, s_{n+1}, s_{n+2})$.
1: $S_u \leftarrow [s_{u-2}, s_{u-1}, 0, s_{u+1}] = (s_1, 0, s_1, s_1^2)$.
2: $m_1 \leftarrow 1/s_1^{t+1}$ and $m_2 \leftarrow 1/c_1$.
3: **for** $r \leftarrow l - 2$ **to** 0 **do**
4: $\quad s_{2u-1} \leftarrow m_1((s_{u+1} + s + s_{u-1} + s_{u-2})^2 + (s_u + s_{u-1})^2(s^t + s^2))$.
5: $\quad s_{2u} \leftarrow s^2$
6: $\quad s_{2u+1} \leftarrow s_{2u-1} + m_2((s_{u+1} + s_{u-1})^2 + s_u^2 s^t)$
7: $\quad$ **if** $n_r = 1$ **then**
8: $\quad\quad s_{2u+2} \leftarrow s_{u+1}^2$, $(s_{3u-3}, s_{3u-2}, s_{3u-1}, s_{3u}, s_{3u+1})$,
9: $\quad\quad S_u \leftarrow [s_{2u-1}, s_{2u}, s_{2u+1}, s_{2u+2}]$.
10: $\quad$ **else**
11: $\quad\quad s_{2u-2} \leftarrow s_{u-1}^2$,
12: $\quad\quad S_u \leftarrow [s_{2u-2}, s_{2u-1}, s_{2u}, s_{2u+1}]$.
13: $\quad$ **end if**
14: **end for**
15: **return** (S_u).

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

Table 9
The computations and rules for double exponentiation.

Rule	Condition	d	e	u	v	s_v	s_{2u-v}	$s_{u-2v}, s_{u-v}, s_{u+v}$
If $d > e$								
R_1	$d \le 4e$	$d-e$	e	u	$u+v$	s_{u+v}	s_{u-v}	s_{u+2v}, s_v, s_{2u+v}
R_2	$d \equiv e \pmod 2$	$(d-e)/2$	e	$2u$	$u+v$	s_{u+v}	s_{3u-v}	s_v^2, s_{u-v}, s_{3u+v}
R_3	$d \equiv 0 \pmod 2$	$d/2$	e	$2u$	v	s_v	s_{4u-v}	$s_{u-v}^2, s_{2u-v}, s_{2u+v}$
R_4	$e \equiv 0 \pmod 2$	d	$e/2$	u	$2v$	s_v^2	s_{u-v}^2	$s_{u-4v}, s_{u-2v}, s_{u+2v}$
Else								
Sub	$e > d$	e	d	v	u	s_v	s_{u-2v}	$s_{2u-v}, s_{u-v}, s_{u+v}$

The above algorithm computes $\mathrm{Tr}(\alpha^n)$ given $\mathrm{Tr}(\alpha)$, n in $(1I + 1M)$ as precomputation and $(4M + 4S)(l-1)$ operations for main loop, where I, M, and S stand for inversion, multiplication and squaring respectively.

3.9.1. Simultaneous double exponentiation

The double exponentiation for this case is given below in Algorithm 8. The variation rules and corresponding effects on variables are given in Table 9. Let $0 < a, b, m, n < \ell$ be integers then,

Note that the Algorithm 8 is similar to the Algorithm 5. The main difference lies in the update rules given in Table 9 and background construction of the recursive relations for the KK-PKC 3.9.

Algorithm 8 Double Exponentiation for Factor 4, for NR-KK-DSA

Require: $(a > 0, b > 0) \in \mathbb{F}_q$, s_m, s_n, s_{n-m}, and s_{n-2m}.
Ensure: s_{an+bm}
1: $f_2 \leftarrow 1, d \leftarrow a, e \leftarrow b, u \leftarrow n, v \leftarrow m$
2: Use Algorithm 5, with update rules from Table 9 and compute $s_{d(u+v)}$ using Algorithm 7.
3: **return** $(s_{d(u+v)})^{f_2}$

Remark 13. The Algorithm 8 can compute s_{an+ml} in $\approx 6.37 \log(\max(a, b))$ multiplications in $\mathbb{F}_p$ given $s_m, 0 < a, b < \ell$, s_n, s_{n-m}, and s_{n-2m}.

Now we look at the protocols to validate this trace based PKC.

3.9.2. KK-DH type key exchange: case $k = 4, p = 2, q = p^{2u+1}, u \ge 1$

1. *System public parameters:* Let $G = \langle\alpha\rangle \subset \mathbb{F}_{q^4}^*, p = 2, q = p^{2u+1}, u \ge 1, t = \sqrt{2q}, ord(\alpha) = \ell | (q - t + 1)$, and $g(x) = x^4 - ax^3 + bx^2 - cx + 1$, be the characteristic polynomial of α.
2. **A** computes her public key $P_A = V_m(a) = s_m$ by running Algorithm 7 with inputs a and her secret key e, and **B** computes his public key $P_B = V_r(a) = s_r$ by running Algorithm 7 with inputs a and his secret key r.
3. *Private keys:* Random $1 < m < \ell$ and $1 < r < \ell$ are secret keys of **A** and **B** respectively.
4. *Common key:* Both **A** and **B** compute their common key $K = P_{AB} = P_{BA} = s_{mr}$ as follows:
   (i) **A** acquires **B**'s public key and runs Algorithm 7 with input s_r and her private key m to obtain common key: $K = P_{AB} = V_m(s_r) = V_{mr}(a) = s_{mr}$.
   (ii) **B** does the similar things by using his private key r to compute common key: $K = P_{BA} = V_r(s_m) = V_{rm}(a) = s_{rm}$.

3.9.3. KK-ElGamal type encryption scheme: case $k = 4, p = 2, q = p^{2u+1}, u \ge 1$

1. *System public parameters:* Let $G = \langle\alpha\rangle \subset \mathbb{F}_{q^4}^*, p = 2, q = p^{2u+1}, u \ge 1, t = \sqrt{2q}, ord(\alpha) = \ell | (q - t + 1)$, and $g(x) = x^4 - ax^3 + bx^2 - cx + 1$, be the characteristic polynomial of α.
2. **B**'s public key $P_B = V_m(a) = s_m$, private key: $1 < m < \ell$.
3. Assumption: message $M \in G$.
4. **A** sends message $M \in \mathbb{F}_q^*$ as follows:
   (i) Chooses a random $1 < r < \ell$ and computes symmetric key $K = V_r(P_B) + V_r(P_B)^t = V_r(s_m) + V_r(s_m)^t = s_{rm} + s_{rm}^t$, mask s_r, by running Algorithm 7 and then computes $c = KM$.
   (ii) **A** sends ciphertext $C = (s_r, c)$ to **B**.
5. *Decryption:* **B** decrypts C as follows:
   (i) Based on mask s_r and his private key m computes symmetric key $K = V_m(s_r) + V_m(s_r)^t = s_{mr} + s_{mr}^t$ by running Algorithm 7 with inputs s_r and his private key m.
   (ii) Then decrypts message by computing, $M = K^{-1}c$.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

Table 10
KK-ElGamal type encryption scheme analysis.

Process	Cost
Encryption	$\approx 6.4\log(u)$ **M**
Decryption	$\approx 6.4\log(u)$ **M**
Comm. overhead	1 element over $\mathbb{F}_q$
Public key size	1 element over $\mathbb{F}_q$

Table 11
Analysis of KK-NR-DSA based on generic symmetric encryption.

Process	Cost
Signature generation	$\approx 6.37\log p$ **M**
Signature verification	$\approx 6.37\log p$ **M**
Comm. overhead	2 elements over $\mathbb{F}_q$
Public key size	4 elements over $\mathbb{F}_q$

Remark 14. The semantic security of KK-ElGamal type encryption is similar to the one given in Remark 12.

The analysis of KK-ElGamal type encryption scheme is given in Table 10.

3.9.4. KK-Nyberg–Rueppel type digital signature algorithm based on generic symmetric encryption: case $k = 4, p = 2, q = p^{2u+1}, u \geq 1$

This scheme was introduced by Koray Karabina in [15].**A** wants to send signed message $M \in \mathbb{F}_q^*$ to **B** and **B** verifies it. To do this, **A** and **B** do the following:

1. *System public parameters:* Let $G = \langle\alpha\rangle \subset \mathbb{F}_{q^4}^*, p = 2, q = p^{2u+1}, u \geq 1, t = \sqrt{2q}, ord(\alpha) = \ell|(q - t + 1), g(x) = x^4 - ax^3 + bx^2 - cx + 1$, be the characteristic polynomial of α and $H : G \to \mathbb{Z}_\ell$-valued hash function.
2. *Public keys:*
   (i) **A** randomly selects $1 < m < \ell$ and computes public key $P_A = V_m(a) := S_m = \{s_{m-2}, s_{m-1}, s_m, s_{m+1}\}$ by running Algorithm 7 with inputs a and her private key m.
   (ii) **B** also randomly selects $1 < r < \ell$ and computes public key $P_B = V_r(a) := S_r = \{s_{r-2}, s_{r-1}, s_r, s_{r+1}\}$ by running Algorithm 7 with inputs a and his private key r.
3. *Private keys:* The private keys of **A** and **B** are m and r, respectively.
4. *Signature*: **A** signs the message M as follows:
   (i) **A** randomly selects $1 < d < \ell$, and computes mask $V_d(a) = s_d$, and extracts session key $K = Ext(s_d)$ from s_d.
   (ii) **A** obtains ciphertext C by encrypting the message M with K, using generic symmetric encryption and computes the hash value of C, that is $h = H(C) \bmod \ell$.
   (iii) **A** computes $n = d + mh \bmod \ell$.
   (iv) **A** sends the signature (n, C) to **B**.
5. *Verification*: **B** verifies **A**'s signature and recovers message as follows:
   (i) Computes $h = H(C) \bmod \ell$ and replaces h by $-h$.
   (ii) Computes s_{hm+n} from $S_m = \{s_{m-2}, s_{m-1}, s_m, s_{m+1}\}$ and a using double exponentiation Algorithm 8.
   (iii) Computes session key $\acute{K} = Ext(s_{n+hm})$ from s_{n+mh} and computes $\acute{C}$ using $\acute{K}$ and M.
   (iv) **B** accepts if and only if $\acute{C} = C$.

The analysis of KK-NR-DSA based on generic symmetric encryption scheme is given in Table 11.

3.9.5. KK-Nyberg–Rueppel type digital signature algorithm based on KK-ElGamal type encryption: case $k = 4, p = 2, q = p^{2u+1}, u \geq 1$

A wants to send signed encrypted message $M \in \mathbb{F}_q^*$ containing agreed upon redundancy to **B** and **B** verifies and recovers the message. To do this, **A** and **B** do the following:

1. *System public parameters:* Let $G = \langle\alpha\rangle \subset \mathbb{F}_{q^4}^*, p = 2, q = p^{2u+1}, u \geq 1, t = \sqrt{2q}, ord(\alpha) = \ell|(q - t + 1), g(x) = x^4 - ax^3 + bx^2 - cx + 1$, be the characteristic polynomial of α and $H : G \to \mathbb{Z}_\ell$-valued hash function.
2. *Public keys:*
   (i) **A** randomly selects $1 < m < \ell$ and computes public key $P_A = V_m(a) = s_m$ by running Algorithm 7 with inputs a and her private key m.
   (ii) **B** also randomly selects $1 < r < \ell$ and computes public key $P_B = V_r(a) = s_r$ by running Algorithm 7 with inputs a and his private key r.
3. *Private keys:* The private keys of **A** and **B** are m and r, respectively.

Table 12
Analysis of KK-NR-DSA based on KK-ElGamal encryption.

Process	Cost
Signature generation	$\approx 6.37 \log p$ **M**
Signature verification	$\approx 12.74 \log p$ **M**
Comm. overhead	2 elements over $\mathbb{F}_q$
Public key size	1 element over $\mathbb{F}_q$

4. *Signature*: **A** signs the message M as follows:
   (i) **A** randomly selects ephemeral private key $1 < d < \ell$ and computes ephemeral public key $V_d(a) = s_d$, and computes common key $K = V_d(s_r) + V_d(s_r)^t = s_{dr} + s_{dr}^t$.
   (ii) **A** computes $h = H(M) \bmod \ell$, obtains ciphertext $C = MK$.
   (iii) **A** computes $n = d + mh (\mathrm{mod}\ \ell)$.
   (iv) **A** sends the signature (n, s_d, C) to **B**.
5. *Verification*: **B** recovers message M and verifies **A**'s signature as follows:
   (i) Checks $0 \leq n < \ell$, if not failure.
   (ii) Computes common key $K = V_r(s_d) + V_r(s_d)^t = s_{rd} + s_{rd}^t$, and decrypts C to $M : M = CK^{-1}$. If M does not contain agreed upon redundancy then failure.
   (iii) Computes $h = H(M) \bmod \ell$ and computes $S_{hm} := \{s_{hm-2}, s_{hm-1}, s_h m, s_{hm+1}\}$ using Algorithm 7.
   (iv) Computes s_{hm+n} from $S_m = \{s_{m-2}, s_{m-1}, s_m, s_{m+1}\}$ and a using double exponentiation Algorithm 8.
   (v) Accepts if $s_{hm+n} = s_d$.

The analysis of KK-NR-DSA based on the KK-ElGamal encryption scheme is given in Table 12.

Remark 15. The KK-NR-DSA based on KK-ElGamal type encryption scheme is resistant to both forgery attacks discussed in Remark 18.

3.10. KK-PKC: case $k = 6, p = 3, q = p^u, u \geq 1, u = odd, t = \sqrt{3q}$

Let $\alpha \in \mathbb{F}_{q^6}$ with $ord(\alpha) = \ell | (q \pm t + 1)$. Note that $\ell | \Phi_6(q) = (q^2 - q + 1) = (q + t + 1)(q - t + 1)$. Here $\Phi_6(q)$ is the 6th cyclotomic polynomial evaluated at q. It is shown in [9] that the characteristic polynomial $g(x)$ of α is nothing but in the form,

$$\begin{aligned} g(x) &= \prod_{i=0}^{6}(x - \alpha^{q^i}), \\ &= x^6 - (\mathrm{Tr}(\alpha))x^5 + (\mathrm{Tr}(\alpha)^t + \mathrm{Tr}(\alpha))x^4 - (\mathrm{Tr}(\alpha^2) + 2\mathrm{Tr}(\alpha) + 2)x^3 - (\mathrm{Tr}(\alpha)^t + \mathrm{Tr}(\alpha))x^2 + (\mathrm{Tr}(\alpha))x - 1. \end{aligned}$$

Note that it is also shown in [9] that $\mathrm{Tr}(\alpha^2) = \mathrm{Tr}(\alpha)^2 + \mathrm{Tr}(\alpha) + \mathrm{Tr}(\alpha)^t$, and therefore $g(x)$ becomes;

$$g(x) = x^6 - (\mathrm{Tr}(\alpha))x^5 + (\mathrm{Tr}(\alpha)^t + \mathrm{Tr}(\alpha))x^4 - (\mathrm{Tr}(\alpha)^2 + \mathrm{Tr}(\alpha)^t + 2)x^3 - (\mathrm{Tr}(\alpha)^t + \mathrm{Tr}(\alpha))x^2 + (\mathrm{Tr}(\alpha))x - 1.$$

Hence $g(x)$ is completely determined by $\mathrm{Tr}(\alpha)$. For any integer m, let,

$$g_m(x) = \prod_{i=0}^{6}(x - (\alpha^m)^{q^i}),$$

be the characteristic polynomial of α^m, then it can be checked that

$$\begin{aligned} g_m(x) = {} & x^6 - (\mathrm{Tr}(\alpha^m))x^5 + (\mathrm{Tr}(\alpha^m)^t + \mathrm{Tr}(\alpha^m))x^4 - (\mathrm{Tr}(\alpha^m)^2 + \mathrm{Tr}(\alpha^m)^t + 2)x^3 - (\mathrm{Tr}(\alpha^m)^t + \mathrm{Tr}(\alpha^m))x^2 \\ & + (\mathrm{Tr}(\alpha^m))x - 1. \end{aligned}$$

Now let us introduce $V_r : \mathbb{F}_q \to \mathbb{F}_q$ as follows, for any b; $V_r(b) = \mathrm{Tr}(\beta)$, where β is a root of,

$$h(x) = x^6 - bx^5 + (b + b^t)x^4 - (b^2 + b^t + 2)x^3 + (b^t + b)x^2 - bx + 1 = 0.$$

It is clear that V_r is well defined and for any integer m,

$$V_m(V_r(b)) = V_{mr}(b) = s_{mr}.$$

Properties: It is shown in [9] that for all integers m, n we have the following properties;

(i) $s_{-n} = s_n$.
(ii) $s_{2n} = s_n^2 + s_n + s_n^t$.
(iii) $s_n s_m = s_{n+m} + s_{n-m} + s_{n+m(t-1)} + s_{n+m(t-2)} + s_{m+n(t-2)}$.
(iv) $s_{n+m} = s_n s_m - (s_{n-m} + s_{n-3m})(s_v^t + s_v) + s_{n-2m}(s_v^2 + s_v^t + 2) + s_{n-4m}s_m - s_{n-5m}$.

Based upon above properties K. Karabina in [9] also gave algorithm to compute $s_n = \mathrm{Tr}(\alpha^n)$, given n and $\mathrm{Tr}(\alpha)$ which is as follows:

Algorithm 9 Compute $s_n = \mathrm{Tr}(\alpha^n)$, given $s = \mathrm{Tr}(\alpha)$, n, $q = 3^u$, $u = odd$

Require: $n \in \mathbb{Z}_\ell : n = \sum_{j=0}^{l-1} n_j 2^j$, $n_j \in \{0, 1\}$ with $n_{l-1} \neq 0$, and initial conditions $(s_1, 0, s_1, s_2, s_1^3, s_4)$.
Ensure: $(s_{n-2}, s_{n-1}, s_n, s_{n+1}, s_{n+2}, s_{n+3})$.
$C_2 \leftarrow s_1^2$ and $C_t \leftarrow s_1^{t+1}$, $C_{1,t} \leftarrow s_1 + s_1^t$, $C_{1,t,2} \leftarrow C_2 + s_1^t + 2$,
$s_2 \leftarrow C_{1,t,2} + s_1 + 1$, $s_4 \leftarrow s_2^2 + s_2 + s_2^t$.
$S_u \leftarrow (s_{u-2}, s_{u-1}, s_u, s_{u+1}, s_{u+2}, s_{u+3}) = (s_1, 0, s_1, s_2, s_1^3, s_4)$.
$M \leftarrow (s_1^t(s_1^3 + C^t + 2(s_1^t + 1)) + 2(s_1^3 + s_1 + 1))^{-1}$
for $i \leftarrow l - 2$ **to** 0 **do**
  $s_{2u-4} \leftarrow s_{u-2}^2 + s_{u-2} + s_{u-2}^t$, $s_{2u-2} \leftarrow s_{u-1}^2 + s_{u-1} + s_{u-1}^t$, $s_{2u} \leftarrow s_u^2 + s_u + s_u^t$, $s_{2u+2} \leftarrow s_{u+1}^2 + s_{u+1} + s_{u+1}^t$, $s_{2u+4} \leftarrow s_{u+2}^2 + s_{u+2} + s_{u+2}^t$, $s_{2u+6} \leftarrow s_{u+3}^2 + s_{u+3} + s_{u+3}^t$,
  $Y_1 \leftarrow s_1(s_{2u+2} + s_{2u-2}) + s_{2u}C_{1,t,2}$,
  $Y_2 \leftarrow s_1(s_{2u+4} + s_{2u}) + s_{2u+2}C_{1,t,2}$,
  $Y_4 \leftarrow C_{1,t}(s_{2u} + s_{2u-2}) + s_{2u+2} + s_{2u-4}$,
  $Y_5 \leftarrow C_{1,t}(s_{2u+2} + s_{2u}) + s_{2u+4} + s_{2u-2}$,
  $Y_6 \leftarrow C_{1,t}(s_{2u+4} + s_{2u+2}) + s_{2u+6} + s_{2u}$.
  $s_{2u-1} \leftarrow M((s_1^3 + 2(C_2 + s_1) + C_t)Y_1 + C_2Y_2 + (2(C_2 + s_1^t) + s_1 + 1)Y_4 + (2(C_2 + C_t) + s_1)Y_5 + 2s_1Y_6)$
  $s_{2u+1} \leftarrow M(2C_2(Y_1 + Y_2) + s_1(Y_4 + Y_6) + (2s_1^t + C_t + s_1 + 1)Y_5)$
  $s_{2u+3} \leftarrow M(C_2Y_1 + (s_1^3 + 2(C_2 + s_1) + C_t)Y_2 + 2s_1Y_4 + (2(C_2 + C_t) + s_1)Y_5 + (2(C_2 + s_1^t) + s_1 + 1)Y_6)$
  if $n_i = 1$ **then**
    $S_u \leftarrow (s_{u-1}, s_u, s_{u+1}, s_{u+2}, s_{u+3}, s_{u+4})$
  else
    $S_u \leftarrow (s_{u-2}, s_{u-1}, s_u, s_{u+1}, s_{u+2}, s_{u+3})$
  end if
end for
return (S_u).

The above algorithm computes $\mathrm{Tr}(\alpha^n)$ given, $\mathrm{Tr}(\alpha)$ and n in $(1I + 2M + 2S)$ as precomputation and $(53A + 6F + 23M + 6S)(l - 1)$ for the main loop, where A, M, S, F, I stands for addition, multiplication, squaring, exponentiation by the power of the finite field characteristic and inversion respectively. In literature we do not see cryptographic protocols for this case as well. We add the cryptographic protocols to this system and comment that it is a valid PKC.

3.10.1. KK-DH type key exchange: case $k = 6$, $p = 3$, $q = p^u$, $u = odd$

1. *System public parameters:* Let $G = \langle\alpha\rangle \subset \mathbb{F}_{q^6}^*$, $p = 3, q = p^u$, $u = odd, t = \sqrt{3q}$, $ord(\alpha) = \ell | (q - t + 1)$, and $g(x) = x^6 - ax^5 + bx^4 - cx^3 + dx^2 - ex + 1$, be the characteristic polynomial of α.
2. *Public keys:* **A**'s public key $P_A = V_m(a) = s_m$, and **B**'s public key $P_B = V_r(a) = s_r$.
3. *Private keys:* The private keys of **A** and **B** are random $1 < m < \ell$ and $1 < r < \ell$ respectively.
4. *Common key:* Both A and B computes common key $K = P_{AB} = P_{BA} = s_{mr}$.
   (i) **A** acquires **B**'s public key and runs Algorithm 9 with input s_r and her private key m to obtain common key: $K = P_{AB} = V_m(s_r) = s_{mr}$.
   (ii) **B** does the similar things to compute common key using his private key $r : K = P_{BA} = V_r(s_m) = s_{mr}$.

It is not difficult to give the KK-ElGamal Type encryption scheme and KK-Nyberg–Rueppel Type DSA for this case also. It is step by step similar to the one discussed in Sections 3.9.3 and 3.9.5 respectively, so we leave it to the reader for verification.

Remark 16. It is clear that the security of trace based Public Key Cryptosystems so far discussed depends on the trapdoor function $\alpha \to \mathrm{Tr}(\alpha^m)$ which is equivalent to DLP on the group $G = \langle\alpha\rangle \subset \mathbb{F}_{q^k}$. Therefore for the maximum security one should choose α such that $\mathbb{F}_{q^k} = \mathbb{F}_q(\alpha)$, namely the characteristic polynomial of α is equal to its minimal polynomial, and DLP on G is computationally equivalent to DLP on $\mathbb{F}_{q^k}^*$.

3.11. Security analysis

The security of the trace based encryption scheme and Nyberg–Rueppel type digital signatures are discussed below. Although the security of the trace based Diffie Hellman key exchange depends upon the solving DLP over respective extension fields but semantic security of the encryption scheme differs from this DLP which is given below. However, the trace based digital signatures exhibits natural resistance to forgery attacks.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

Semantic security. Suppose that $m_1 \in \mathbb{F}_q^*$ and $m_2 \in \mathbb{F}_q^*$ are two known messages from the attacker $\mathcal{E}$ and $\mathcal{A}$ encrypts message m_1 such that $c = m_1K = m_1(s_{tr} + s_{-tr})$ and sends the ciphertext c to attacker $\mathcal{E}$ who wants to determine whether c is an encryption of m_1 or, m_2. To do this attacker E divides c with m_1 such that $\acute{K} = c/m_1$. Now if attacker $\mathcal{E}$ can find out either s_{tr} or s_{-tr} from $\acute{K}$ then he can form cubic equation $\bar{g}(x) = x^3 - s_{tr}x^2 + s_{-tr}x - 1$ over $\mathbb{F}_q$. Then he determines the irreducibility of $\bar{g}(x)$, if it is irreducible the attacker $\mathcal{E}$ concludes that c is encryption of m_1. But to determine s_{tr} or s_{-tr} from $\acute{K}$ is computationally equivalent to splitting $z \in \mathbb{F}_q$ into two numbers $(X, Y) \in \mathbb{F}_q^2$ such that $z = X + Y$ where $s_{tr} \mapsto X$ and $s_{-tr} \mapsto Y$. The number of such (X, Y) are equal to $\frac{q-1}{2}$. Moreover, the product $tr \in \mathbb{Z}_\ell$ and $\mathbb{Z}_\ell$ is chosen large enough so that the brute force attack becomes infeasible.

Remark 17 (*Semantic Security*)**.** Given $G = \langle\alpha\rangle \subset \mathbb{F}_{q^3}$, (s_1, s_{-1}) and $z \in \mathbb{F}_q$. Finding (s_e, s_{-e}) such that $z = s_e + s_{-e}$ is computationally equivalent to computing $z = X + Y$, $(X, Y) \in \mathbb{F}_q^2$. Since there is no polynomial time algorithm to split z as a sum of two elements in $\mathbb{F}_q$, where the total number of unknown such as (X, Y) are equal to $\frac{q-1}{2}$ and $s_e \mapsto X$ and $s_{-e} \mapsto Y$. Moreover, $e \in \mathbb{Z}_\ell$ and $\mathbb{Z}_\ell$ is chosen large enough so that the brute force attack becomes infeasible.

Remark 18 (*Security of Trace Based NR Type DSA:*)**.** The well known attacks on the Nyberg–Rueppel signature scheme are forgery attacks such as the congruence equation attack and homomorphism attack [16]. In the proposed scheme such attacks are restricted due to the following reasons.

(i) The congruence equation attack is avoided by using hash value of encrypted message.
(ii) The homomorphism attack is restricted in this case due to the fact that trace function is not multiplicative and this attack exploits the fact that $\sigma(g^{u-1})/\sigma(g^u) = \sigma(1/g)$, where g is generator of subgroup and $\sigma : \mathbb{Z}_p \to \mathbb{Z}_p$. Therefore, this attack is restricted for trace based signature schemes. For example let $u, v \in \mathbb{Z}$ and any $\beta \in \mathbb{F}_{q^3} : ord(\beta) = Q|(q^2+q+1)$ then;

$$\mathrm{Tr}(\beta^2) = s_2 = s_1^2 - 2s_{-1}; \quad \text{and} \quad \mathrm{Tr}(\beta^{u+v}) = s_{u+v} = s_u s_v - s_{u-v}s_{-v} + s_{u-2v},$$
$$\Rightarrow \frac{s_{u+v}}{s_v} \neq s_u,$$
$$\therefore \frac{\mathrm{Tr}(\beta^{u+v})}{\mathrm{Tr}(\beta^v)} \neq \mathrm{Tr}(\beta^u).$$

4. Conclusion

For a given $\alpha \in \mathbb{F}_{q^k}$, one can compute for any integer m the $\mathrm{Tr}(\alpha^m)$ by repeated squaring and then trace mapping in a polynomial time. We have searched the literature and found out that for some special conditions on α, q, and k that there are more efficient algorithms to compute $\mathrm{Tr}(\alpha^m)$ from the above classical one. We also found out that in some of these cases there are protocols based on this algorithm and working more efficiently and securely. In this note we discussed all these cases and introduced cryptographic protocols to the ones that were not discussed. With this we formalized the problem in a general setting and brought it to the attention of the researchers from this point of view.

Acknowledgments

We would like to express our sincere gratitude to the anonymous referees for their critical analysis and valuable comments.

References

[1] P.J. Smith, M.J.J. Lennon, LUC: a new public key system, in: Proceedings of the IFIP TC11 Ninth International Conference on Information Security IFIP/Sec'1993, 1993, pp. 103–117.
[2] P. Smith, C. Skinner, A public-key cryptosystem and a digital signature system based on the Lucas function analogue to discrete logarithms, in: Advances in Cryptology—Asiacrypt 1994, in: Lect. Notes in Comp. Sci., vol. 917, Springer, Berlin, 1995, pp. 357–364.
[3] K. Giuliani, G. Gong, Analogues to the Gong-Harn and XTR cryptosystems, Combinatorics and Optimization Research Report CORR 2003-34, University of Waterloo Canada.
[4] K. Giuliani, G. Gong, Efficient key agreement and signature schemes using compact representations in GF(p^{10}), in: Proceedings of IEEE International Symposium on Information Theory—ISIT 2004, Chicago, IL, 2004.
[5] A.K. Lenstra, E.R. Verheul, The XTR public key cryptosystem, in: Advances in Cryptology, Crypto 2000, in: Lect. Notes in Comp. Sci., vol. 1880, Springer, Berlin, 2000, pp. 1–19.
[6] A.K. Lenstra, E.R. Verheul, An overview of the XTR public key system, in: Public key Cryptography and Computational Number Theory (Warsaw, 2000), de Gruyter, Berlin, 2001, pp. 151–180.
[7] W. Bosma, J. Hutton, E.R. Verheul, Looking beyond XTR, in: Advances in Cryptology—Asiacrypt 2002, in: Lect. Notes in Comp. Sci., vol. 2501, Springer, Berlin, 2002, pp. 46–63.
[8] M. Stam, A. Lenstra, Speeding up XTR, in: Advances in Cryptology—Asiacrypt 2001, in: Lect. Notes in Comp. Sci., vol. 2248, 2001, pp. 125–143.
[9] K. Karabina, Factor-4 and 6 compression of cyclotomic subgroups of $\mathbb{F}_{2^{4m}}$ and $\mathbb{F}_{3^{6m}}$, J. Math. Cryptol. 4 (2010) 1–42.
[10] K. Rubin, A. Silverberg, Torus-based cryptography, in: Advances in Cryptology—Crypto 2003, in: Lect. Notes in Comp. Sci., vol. 2729, 2003, pp. 349–365.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022

[11] L.C. Washington, Elliptic Curves: Number Theory and Cryptography, Chapman-Hall, New York, 2003.
[12] G. Gong, L. Harn, Public-key cryptosystems based on cubic finite field extensions, IEEE Trans. Inform. Theory 45 (1999) 2601–2605.
[13] G. Gong, L. Harn, H. Wu, The GH Public-key cryptosystem, in: SAC 2001, in: Lect. Notes in Comp. Sci., vol. 2259, 2001, pp. 284–300.
[14] M. Ashraf, B.B. Kırlar, Efficient message transmission for GH-public key cryptosystem, Institute of Applied Mathematics, Number 2013-08, Preprint, 2013.
[15] K. Karabina, Double-exponentiation in factor-4 group and its applications, eprint.iacr.org/2009/475.pdf?.
[16] A. Miyaji, Weakness in message recovery signature schemes based on discrete logarithm problems 1, in: IEICE Japan Tech. Rep., ISEC95-7, 1995.

Please cite this article in press as: E. Akyıldız, M. Ashraf, An overview of trace based public key cryptography over finite fields, Journal of Computational and Applied Mathematics (2013), http://dx.doi.org/10.1016/j.cam.2013.10.022