Bit Security of the Hyperelliptic Curves Diffie-Hellman Problem

Fangguo Zhang[1,2]

[1]School of Information Science and Technology, Sun Yat-sen University
[2]Guangdong Provincial Key Laboratory of Information Security,
Guangzhou 510006, China
isszhfg@mail.sysu.edu.cn

Abstract. The Diffie-Hellman problem as a cryptographic primitive plays an important role in modern cryptology. The Bit Security or Hard-Core Bits of Diffie-Hellman problem in arbitrary finite cyclic group is a long-standing open problem in cryptography. Until now, only few groups have been studied. Hyperelliptic curve cryptography is an alternative to elliptic curve cryptography. Due to the recent cryptanalytic results that the best known algorithms to attack hyperelliptic curve cryptosystems of genus $g < 3$ are the generic methods and the recent implementation results that hyperelliptic curve cryptography in genus 2 has the potential to be competitive with its elliptic curve cryptography counterpart. In this paper, we generalize Boneh and Shparlinksi's method and result about elliptic curve to the case of Jacobians of hyperelliptic curves. We prove that the least significant bit of each coordinate of hyperelliptic curves Diffie-Hellman secret value in genus 2 is hard as the entire Diffie-Hellman value, and then we also show that any bit is hard as the entire Diffie-Hellman value. Finally, we extend our techniques and results to hyperelliptic curves of any genus.

Keywords: Hyperelliptic curves, Bit Security, Diffie-Hellman Problem.

1 Introduction

The discrete logarithm problem(DLP) and Diffie-Hellman problem(DHP) are basic cryptographic primitives, they play important role in modern cryptology. For example the Diffie-Hellman key exchange [16], the ElGamal encryption [18], the official U.S. Digital Signature Algorithm (DSA) [19], and the BLS short signature scheme [10], etc. Due to Pohlig and Hellman attack [39], it is restricted to groups of prime order p in this paper, where the DLP is the problem to compute $x \in \mathbb{Z}_p^*$ given (g, g^x), and the DHP or computational Diffie-Hellman problem(CDHP) is the problem to compute g^{ab} given (g, g^a, g^b), here $g \in G$ is a generator of group G. Maurer and Wolf [36, 37] have proved that, for every cyclic group G with prime order p, there exists polynomial time algorithm that

reduces the computation of DLP in G to the computation of CDHP in G if we are able to find an elliptic curve, called *auxiliary elliptic curve*, over $\mathbb{F}_p$ with smooth order.

From many cryptographic applications, we know it is very important that partial information of the secret key is not computable or predictable with any significant advantage over a random guess. This is related to Bit Security or Hard-Core Bits problem. Informally speaking, the Bit Security or Hard-Core Bits for DLP can be described as follows: given (g, g^x), if an adversary can compute certain bits(or more generally, certain predicates) of x? Blum and Micali[6] introduced the concept of hard-core bits for one-way functions and showed the existence of a hard-core predicate for the discrete logarithm function in any group G.

However, for the case of the DHP there is no such result has been proven. Informally speaking, the Bit Security or Hard-Core Bits for DHP or CDHP can be described as follows: given (g, g^a, g^b), if an adversary can compute certain bits(or more generally, certain predicates) of $K = g^{ab}$? In another word, if the hardness of CDH bits and the entire CDH is same? This is a long-standing open problem in cryptography. Until now, only few groups have been studied: Boneh and Venkatesan [9] formulated the hidden number problem (HNP) and showed that in the multiplicative group of finite field $\mathbb{F}_p$ computing approximately $(\log p)^{1/2}$ of the bits of the Diffie-Hellman secret is as hard as computing the entire secret. This result is improved in [25] and [7]. Boneh and Shparlinksi in [8] achieved a breakthrough for the elliptic curve Diffie-Hellman problem(i.e., the CDH problem defined over the group of points of an elliptic curve). By using certain twists of the given curve they showed that predicting the least significant bit of the elliptic curve Diffie-Hellman secret in a family of curves is as hard as computing the entire secret. Alternatively, if one looks for a polynomial time reduction of the DHP to the problem of predicting partial information on the same short Weierstrass model, some results have been established using Gröbner bases[29]. Fazio et al. modified Boneh and Shparlinski's idea and applied it to the case of finite fields $\mathbb{F}_{p^2}$, they proved the unpredictability of every single bit of one of the coordinates of the secret Diffie-Hellman value over finite fields $\mathbb{F}_{p^2}$. Wang, Zhan and Zhang [40] generalised this work to extension fields $\mathbb{F}_{p^m}$, where m is polynomial in $\log p$. Li, Näslund and Shparlinski [34] have studied the bit security of CDHP in LUC and XTR. Galbraith, Hopkins and Shparlinski [22] have studied the bit security of bilinear Diffie-Hellman problem in blinear pairing group. About the DHP and its bit security, the Chapter 21 in Galbraith's book[21] is a good reference.

Hyperelliptic curves are a natural generalisation of elliptic curves, and Jacobians of hyperelliptic curves was suggested by Koblitz [32] that they also been

considered for cryptographic applications. The main advantage of genus g over elliptic curve(genus 1) is that a much smaller base field(about g times fewer bits) with same security level. However, for large genus there are subexponential time attacks on the DLP[1]. For genus 2 curves, just as with their elliptic curve counterpart, the best known algorithms to solve the discrete logarithm in such groups are the generic attacks such as Pollard rho mehtod[24]. The practical potential of genus 2 curves in public-key cryptography has recently been highlighted by the fast performance numbers presented. Especially, Gaudry[23] showed that scalar multiplication on the Kummer surface associated with the Jacobian of a genus 2 curve can be more efficient than scalar multiplication on the Jacobian itself. After that, many papers[5, 11] showed that hyperelliptic curve cryptography in genus 2 has the potential to be competitive with its genus 1 elliptic curve cryptography counterpart.

Our Contributions. In this paper, we study the bit security of CDHP in Jacobian group of hyperelliptic curves. The contribution of the paper is as the following.

1. We firstly generalize Boneh and Shparlinksi's method to the case of Jacobians for genus 2 hyperelliptic curves. We prove that the least significant bit of any coordinate of hyperelliptic Diffie-Hellman value with genus 2 over finite fields is unpredictable.
2. We extend the least significant bit to very bit case, show that for genus 2 hyperelliptic curves, to compute any bit of any coordinate of Diffie-Hellman value is hard as for computing the entire Diffie-Hellman value.
3. We also generalize these results from genus 2 hyperelliptic curves to any genus hyperelliptic curves.

Organization. The rest of this paper is organized as follows. Section 2 introduces some mathematical preliminaries, including hyperelliptic curves and hyperelliptic curve Diffie-Hellman problem, twisting hyperelliptic curves and hidden number problem with chosen multiplier. Section 3 gives the main results and proofs about the the unpredictability of least significant bit of hyperelliptic Diffie-Hellman value with genus 2. Section 4 extends the least significant bit to very bit case. Section 5 generalizes the results of hyperelliptic Diffie-Hellman value with genus 2 to the case of any genus hyperelliptic curves. Section 6 gives the conclusions.

2 Mathematical Preliminaries

2.1 Hyperelliptic Curves and Hyperelliptic Curve Diffie-Hellman Problem

We first introduce the definition and operations of hyperelliptic curves over finite field, more details can be found in references[4, 21]. Let $\overline{\mathbb{K}}$ be the algebraic closure of the field $\mathbb{K}$. A hyperelliptic curve C of genus $g \geq 1$ over $\mathbb{K}$ is given by

$$C : y^2 + h(x)y = f(x) \tag{1}$$

where $f(x)$ is a monic polynomial of degree $2g+1$, $h(x)$ is a polynomial of degree at most g, and there are no solutions $(x, y) \in \overline{\mathbb{K}} \times \overline{\mathbb{K}}$ simultaneously satisfying the equation (1) and the partial derivative equations $2y + h(x) = 0$ and $h'(x)y - f'(x) = 0$. Let $P = (x, y)$ be a finite point on hyperelliptic curve C, the opposite of P is defined as $-P = (x, -y - h(x))$.

A divisor on C is a finite formal sum $D = \Sigma_P m_P P$, where m_P are integers that are 0 for almost all P. The degree of D is defined by $\deg D = \Sigma_P m_P$. The set of all the divisors defined over $\mathbb{K}$ forms an abelian group with the set of divisors of degree 0 as its subgroup, that is $Div_C^0 \subset Div_C$. The function field of C over $\mathbb{K}$, denoted $\mathbb{K}(C)$, is the field of fraction of the polynomail ring $\mathbb{K}[C] = \mathbb{K}[x, y]/(y^2+h(x)y-f(x))$. To every rational function $F \in \mathbb{K}(C)$, it can associate a divisor via the valuations at all points of the curve: $div(F) = \Sigma_{P \in C(\mathbb{K})} v_P(F)P$. These so called principal divisors are of degree zero and form a subgroup of Div_C^0. We denote the group of principal divisors as $Princ_C$. The Jacobian or the divisor class group of the curve C is given by $J_C = Div_C^0/Princ_C$.

From the work of Cantor[12] and Koblitz[32], the element $D = \Sigma m_i P_i - (\Sigma m_i)P_\infty$ (here $\Sigma m_i \leq g$, $P_i = (x_i, y_i)$, P_∞ is the point at infinity) of J_C has a Mumford representation, D can be only determined by two polynomails u and v in $\mathbb{K}[x]$, where $u(x) = \Pi(x - x_i)^{m_i}$, and u, v satisfy: 1) $\deg v < \deg u \leq g$; 2) $v(x_i) = y_i$, for all the i that made $m_i \neq 0$; 3) $v^2 + vh - f \equiv 0 \pmod{u}$. In general we write $D = (u(x), v(x))$, it can be represented by 2g-tuple $(u_{g-1}, ..., u_1, u_0, v_{g-1}, ..., v_1, v_0)$.

We will focus on the most cryptographically common case of genus 2 curves, where C is an imaginary hyperelliptic curve over a large prime field $\mathbb{F}_p$. A hyperelliptic curve C over $\mathbb{F}_p$ with genus 2 is defined by

$$C : \ y^2 = x^5 + f_3x^3 + f_2x^2 + f_1x + f_0 \tag{2}$$

In this case, any element $D = (u(x), v(x))$ of the Jacobian group $J_C(\mathbb{F}_p)$ will satisfy: $u(x)$ is monic, $\deg v < \deg u \leq 2$ and $u_i | v_i^2 - f$. When $\deg u = 0$, this is the zero element O; When $\deg u = 1$, this is the element of $(x - u_0, v_0)$; The

general case is $D = (u(x), v(x)) = (x^2 + u_1 x + u_0, v_1 x + v_0)$, we call the element with this form a general element. When we randomly choose an element D from $J_C(\mathbb{F}_p)$, D is a general element with the probability about $1 - \frac{1}{p}$. We also use (u_1, u_0, v_1, v_0) to represent a general element $D = (u(x), v(x))$.

Cantor's algorithm can perform addition and doubling operations in Jacobian group. In this paper, we will need the explicit formulas for the group operations. Harley[26] optimized Cantor's algorithm and obtained the first practical explicit formulas in genus 2, and then Lange [33] extended it and got significant improvements. The formulas were subsequently improved by Costello and Lauter[13] through a more direct geometric interpretation of the group law. Diao and Joye[17] presented efficient unified addition formule for hyperelliptic curve cryptography. Very recently, Hisil and Costello[28] combines several techniques to arrive at explicit formulas in Jacobian coordinates that are significantly faster than those in previous works. For the genus 2 curves over large prime field $\mathbb{F}_p$, let $D_1 = (u_{11}, u_{10}, v_{11}, v_{10})$ and $D_2 = (u_{21}, u_{20}, v_{21}, v_{20})$ be two general elements of the Jacobian group. Table 1 and 2 in Appendix A are the explicit affine formulas for general point addition and general point doubling which derived from the results in [13, 28].

Let $D \in J_C(\mathbb{F}_p)$ be an element of prime order q. The DLP in $J_C(\mathbb{F}_p)$ is: given another element $D' \in < D >$, to determine the integer m such that $D' = mD$. We define the hyperelliptic curve Diffie-Hellman function as

$$DH_{J,D}(aD, bD) = abD$$

where a, b are in $\mathbb{F}_q$. The hyperelliptic curve DHP is to compute $DH_{J,D}(D_1, D_2)$ given $(C, J_C(\mathbb{F}_p), D, D_1, D_2)$.

2.2 Twisting Hyperelliptic Curves

Let C be a curve with genus g defined over a field $\mathbb{K}$. A curve C' defined over $\mathbb{K}$ that is isomorphic to C over $\overline{\mathbb{K}}$, is called a twist of C.

For a hyperelliptic curve C of genus 2 over $\mathbb{F}_p$ given by the equation

$$C: \ y^2 = x^5 + f_3 x^3 + f_2 x^2 + f_1 x + f_0$$

For any $\lambda \in \mathbb{F}_p^*$, we define $\phi_\lambda(C)$ to be a twist of C:

$$\phi_\lambda(C): \ y^2 = x^5 + \lambda^4 f_3 x^3 + \lambda^6 f_2 x^2 + \lambda^8 f_1 x + \lambda^{10} f_0$$

For any point $P = (x, y) \in C$, $\phi_\lambda(P) = (\lambda^2 x, \lambda^5 y) \in \phi_\lambda(C)$. This curve isomorphism can endow an isomorphism between $J_C(\mathbb{F}_p)$ and $J_{\phi_\lambda(C)}(\mathbb{F}_p)$, we denote this group isomorphism as $\phi_\lambda^* : J_C(\mathbb{F}_p) \to J_{\phi_\lambda(C)}(\mathbb{F}_p)$.

The explicit formulas for ϕ_λ^* is:

$$\phi_\lambda^* : J_C(\mathbb{F}_p) \to J_{\phi_\lambda(C)}(\mathbb{F}_p)$$
$$O(= P_\infty - P_\infty) \to O'(= P'_\infty - P'_\infty)$$
$$(x_1, y_1) \to (\lambda^2 x_1, \lambda^5 y_1)$$
$$(u_1, u_0, v_1, v_0) \to (\lambda^2 u_1, \lambda^4 u_0, \lambda^3 v_1, \lambda^5 v_0)$$

Therefore, we have

$$DH_{\phi_\lambda^*(J),\phi_\lambda^*(D)}(\phi_\lambda^*(D_1), \phi_\lambda^*(D_2)) = \phi_\lambda^*(DH_{J,D}(D_1, D_2)).$$

In this paper, we are working with the family of curves $\{\phi_\lambda(C)\}_{\lambda\in\mathbb{F}_p^*}$ and their Jacobians $\{J_{\phi_\lambda(C)}(\mathbb{F}_p)\}_{\lambda\in\mathbb{F}_p^*}$ associated with a given curve C and its Jacobians $J_C(\mathbb{F}_p)$. Hence, if the hyperelliptic DHP is hard to compute in $J_C(\mathbb{F}_p)$, then it is also hard to compute for all $\{J_{\phi_\lambda(C)}(\mathbb{F}_p)\}_{\lambda\in\mathbb{F}_p^*}$.

2.3 HNP-CM Problem and HNP-CM^d Problems

The Hidden Number Problem with Chosen Multiplier(HNP-CM) is a variant of the Hidden Number Problem(HNP) which proposed by Boneh and Shparlinski[8].

We denote by LSB(z) the least significant bit of an integer $z > 0$.

Definition 1 (HNP-CM[8]). *Fix an $\varepsilon > 0$. Let p be a prime. For an $\alpha \in \mathbb{F}_p$ let $L : \mathbb{F}_p* \to \{0, 1\}$ be a function satisfying*

$$Pr_{t\in\mathbb{F}_p*}[L(t) = LSB(\alpha \cdot t \mod p)] \geq \frac{1}{2} + \epsilon$$

The HNP-CM problem is: given an oracle for $L(t)$, find α in polynomial time. For small ϵ there might be multiple α satisfying the above condition. In this case the list-HNP-CM problem is to find the list of all such $\alpha \in F_p^*$. Due to Alexi, Chor, Goldreich and Schnorr[1], there is an algorithm to solve the list-HNP-CM for any $\epsilon > 0$.

Theorem 1. *[8] Let p be a $n-$ bit prime and let $\epsilon > 0$. Then, given ϵ, the list HNP-CM problem can be solved in expected polynomial time in n and $1/\epsilon$.*

Informally speaking, suppose one has an oracle $\mathcal{A}$ such that $\mathcal{A}(t) = \text{LSB}(\alpha \cdot t \mod p)$, then one can compute α using $O(\log_2(p))$ oracle queries.

The HNP-CM^d problem is a variant of HNP-CM problem, it is defined as follows:

Definition 2 (HNP-CM^d[8]). *Fix an $\varepsilon > 0$. Let p be a prime. For an $\alpha \in \mathbb{F}_p$ let $L : \mathbb{F}_p* \to \{0,1\}$ be a function satisfying*

$$Pr_{t\in\mathbb{F}_p*}[L^d(t) = LSB(t^d \cdot \alpha \mod p)] \geq \frac{1}{2} + \epsilon$$

The HNP-CM^d problem is: given an oracle for $L^d(t)$, find α in polynomial time. For small ϵ there might be multiple α satisfying the condition. In this case the list-HNP-CM^d problem is to find all such $\alpha \in F_p^*$. We will use it for $d = 2$, $d = 3$, $d = 4$ and $d = 5$ in this paper. About the HNP-CM^d problem, Boneh and Shparlinski gave the following theorem:

Theorem 2. *[8] Fix an integer $d > 1$. Let p be a $n-$ bit prime and let $\epsilon > 0$. Then, given ϵ, the HNP-CM^d problem can be solved in expected polynomial time in $\log p$ and d/ϵ.*

3 Our Results for Least Significant Bit

For any general element $D = (u_1, u_0, v_1, v_0)$ of $J_C(\mathbb{F}_p)$, we use $u_1(D)$ to denote the u_1- coordinate of D, similarly for $u_0(D)$, $v_1(D)$ and $v_0(D)$. The main result for the least significant bit of hyperelliptic curve DHP is the following theorem.

Theorem 3. *Let p be a prime, and let C be a hyperelliptic curve with genus 2 over $\mathbb{F}_p$. Let $D \in J_C(\mathbb{F}_p)$ be an element of prime order. Given $(C, J_C(\mathbb{F}_p), D, aD, bD)$, if there is an efficient algorithm for predicting the least significant bit of any coordinate of abD, then there is an efficient algorithm for computing the DHP on $J_C(\mathbb{F}_p)$.*

Let $\mathcal{A}_{u_1}(C, J_C(\mathbb{F}_p), D, aD, bD)$ be an oracle that returns LSB$(u_1(abD))$ where $D \in J_C(\mathbb{F}_p)$. Similarly, let $\mathcal{A}_{u_0}(C, J_C(\mathbb{F}_p), D, aD, bD)$ be an oracle that returns LSB$(u_0(abD))$, $\mathcal{A}_{v_1}(C, J_C(F_p), D, aD, bD)$ returns LSB$(v_1(abD))$ and $\mathcal{A}_{v_0}(C, J_C(\mathbb{F}_p), D, aD, bD)$ returns LSB$(v_0(abD))$, respectively. To prove the above theorem, we need the following lemma.

Lemma 1. *Given $(C, J_C(\mathbb{F}_p), D, aD, bD)$, to compute any one coordinate of abD is hard as the entire abD.*

Proof. Let $aD = (u_{a,1}, u_{a,0}, v_{a,1}, v_{a,0})$, $bD = (u_{b,1}, u_{b,0}, v_{b,1}, v_{b,0})$ and $j(aD) = (u_{ja,1}, u_{ja,0}, v_{ja,1}, v_{ja,0})$. Assume that $abD = (u_{ab,1}, u_{ab,0}, v_{ab,1}, v_{ab,0})$.

Now, we prove that computing u_1-coordinate of abD is hard as the entire abD. Similar method can be used to prove other coordinate cases.

Assume that there is an oracle $\mathcal{B}$ that given $(C, J_C(\mathbb{F}_p), D, aD, bD)$ and returns $u_1(abD)$, that is

$$\mathcal{B}(C, J_C(\mathbb{F}_p), D, aD, bD) = u_1(abD) = u_{ab,1}.$$

We rewrite $abD = (u_{ab,1}, \mathbf{u_{ab,0}}, \mathbf{v_{ab,1}}, \mathbf{v_{ab,0}})$, and by the oracle $\mathcal{B}$, $u_{ab,1}$ is already known. Now we show how to find out $\mathbf{u_{ab,0}}, \mathbf{v_{ab,1}}$ and $\mathbf{v_{ab,0}}$, therefore the entire abD.

From the Mumford representation of the element in Jacobian group of hyperelliptic curve with genus 2, for $abD = (u_{ab,1}, u_{ab,0}, v_{ab,1}, v_{ab,0}) = (x^2 + u_{ab,1}x + u_{ab,0}, v_{ab,1}x + v_{ab,0})$, we have

$$(v_{ab,1}x + v_{ab,0})^2 - (x^5 + f_3x^3 + f_2x^2 + f_1x + f_0) \equiv 0 \mod (x^2 + u_{ab,1}x + u_{ab,0}).$$

Replacing x^2 with $-(u_{ab,1}x + u_{ab,0})$ on the left side,

$$(v_{ab,1}x+v_{ab,0})^2-(u_{ab,1}x+u_{ab,0})^2x+f_3(u_{ab,1}x+u_{ab,0})x+f_2(u_{ab,1}x+u_{ab,0})-f_1x-f_0 = 0.$$

Comparing the coefficients of x^i for $i = 1$ and 0, we get the following equations about $\mathbf{u_{ab,0}}, \mathbf{v_{ab,1}}, \mathbf{v_{ab,0}}$.

$$\mathbf{v_{ab,0}}^2 - \mathbf{v_{ab,1}^2}\mathbf{u_{ab,0}} + 2u_{ab,1}\mathbf{u_{ab,0}^2} + (f_2 - u_{ab,1}f_3 - u_{ab,1}^3)\mathbf{u_{ab,0}} - f_0 = 0 \quad (3)$$

$$2\mathbf{v_{ab,0}}\mathbf{v_{ab,1}} - \mathbf{u_{ab,0}^2} - u_{ab,1}\mathbf{v_{ab,1}^2} + (f_3 + 3u_{ab,1}^2)\mathbf{u_{ab,0}} + f_2u_{ab,1} - f_3u_{ab,1}^2 - u_{ab,1}^4 - f_1 = 0 \quad (4)$$

We now call $\mathcal{B}$ one more time as follows:

$$\mathcal{B}(C, J_C(\mathbb{F}_p), D, aD, bD + D) = u_1(abD + aD) = u_{ab+a,1}$$

From the explicit formula of general point addition in Jacobian group of hyperelliptic curve with genus 2, we know that $u_{ab+a,1}$ is also a function about $\mathbf{u_{ab,0}}, \mathbf{v_{ab,1}}, \mathbf{v_{ab,0}}$:

$$u_{ab+a,1} = u_{ab,1} - u_{a,1} + 2\frac{(\mathbf{v_{ab,0}} - v_{a,0})(u_{a,1}(u_{ab,1} - u_{a,1}) - (\mathbf{u_{ab,0}} - u_{a,0})) - u_{a,0}(u_{ab,1} - u_{a,1})(\mathbf{v_{ab,1}} - v_{a,1})}{(u_{ab,1} - u_{a,1})(\mathbf{v_{ab,0}} - v_{a,0}) - (\mathbf{u_{ab,0}} - u_{a,0})(\mathbf{v_{ab,1}} - v_{a,1})} - \frac{((\mathbf{u_{ab,0}} - u_{a,0})(u_{a,1}(u_{ab,1} - u_{a,1}) - (\mathbf{u_{ab,0}} - u_{a,0})) - u_{a,0}(u_{ab,1} - u_{a,1})^2)^2}{((u_{ab,1} - u_{a,1})(\mathbf{v_{ab,0}} - v_{a,0}) - (\mathbf{u_{ab,0}} - u_{a,0})(\mathbf{v_{ab,1}} - v_{a,1}))^2}$$

This is:

$$((u_{ab,1} - u_{a,1})(\mathbf{v_{ab,0}} - v_{a,0}) - (\mathbf{u_{ab,0}} - u_{a,0})(\mathbf{v_{ab,1}} - v_{a,1}))^2(u_{ab+a,1} - u_{ab,1} + u_{a,1}) - 2((\mathbf{v_{ab,0}} - v_{a,0})(u_{a,1}(u_{ab,1} - u_{a,1}) - (\mathbf{u_{ab,0}} - u_{a,0})) - u_{a,0}(u_{ab,1} - u_{a,1})(\mathbf{v_{ab,1}} - v_{a,1}))((u_{ab,1} - u_{a,1})(\mathbf{v_{ab,0}} - v_{a,0}) - (\mathbf{u_{ab,0}} - u_{a,0})(\mathbf{v_{ab,1}} - v_{a,1})) + ((\mathbf{u_{ab,0}} - u_{a,0})(u_{a,1}(u_{ab,1} - u_{a,1}) - (\mathbf{u_{ab,0}} - u_{a,0})) - u_{a,0}(u_{ab,1} - u_{a,1})^2)^2 = 0 \quad (5)$$

The equations (3), (4) and (5) form a 3-variates polynomial system. This multivariate polynomial system has total degree $3 \times 2 \times 4 = 24$. So, due to the Bézou Theorem, the number of the solution does not exceed 24. We solve this 3-variates polynomial system and obtain $u_{ab,0}, v_{ab,1}, v_{ab,0}$ of abD, so the entire abD.

A detailed Magma[35] implementation for such 3-variates polynomial system according to a genus 2 curve over $GF(2^{127}-1)$ which used in [11] is provided in Appendix B.

According to our experiments, most cases we can obtain one solution, therefore the entire abD is obtained. However, sometimes, this 3-variates polynomial system will output more than one solutions. In this case, we can find the correct solution for abD up to at most 24 times testings. In order to avoid these testing operations, we can construct an over-defined systems to obtain the unique solution. To do this, we can call $\mathcal{B}$ more times

$$\mathcal{B}(C, J_C(F_p), D, aD, bD + jD) = u_1(abD + jaD) = u_{ab+ja,1}, \text{for } j\text{=2 to } t,$$

and construct an over-defined systems of multivariate polynomial equations about $\mathbf{u_{ab,0}}, \mathbf{v_{ab,1}}, \mathbf{v_{ab,0}}$. The number of solutions of an over-defined system will be generally one. Using Courtois et al.'s [14] algorithm, we can efficiently solve this over-defined systems of multivariate polynomial equations and obtain $u_{ab,0}, v_{ab,1}, v_{ab,0}$ of abD. □

Before proving the Theorem 3, we can consider a special case of this theorem: $p = 2 \mod 3$. In such case, we have a very simple proof for Theorem 3.

Theorem 4. *Suppose $p = 2 \mod 3$. Then the hardness of the least significant bit of any coordinate of abD which given $(C, J_C(\mathbb{F}_p), D, aD, bD)$ is same as the entire abD.*

Proof. Firstly, we prove that "Predicting $\mathrm{LSB}(v_1(abD))$ is hard as $v_1(abD)$". Assume that there is an efficient algorithm $\mathcal{A}_{v_1}$ for predicting the the least significant bit of v_1-coordinate of abD, i.e., given $(C, J_C(\mathbb{F}_p), D, aD, bD)$,

$$\mathcal{A}_{v_1}(C, J_C(\mathbb{F}_p), D, aD, bD) = \mathrm{LSB}(v_1(abD)).$$

Now, we choose a random number $\lambda \in \mathbb{F}_p^*$ and call the oracle

$$\mathcal{A}_{v_1}(\phi_\lambda(C), \phi_\lambda^*(J_C(F_p)), \phi_\lambda^*(D), \phi_\lambda^*(aD), \phi_\lambda^*(bD))$$

to get $\mathrm{LSB}(v_1(\phi_\lambda^*(abD))) = \mathrm{LSB}(\lambda^3 v_1(abD))$. Since $\gcd(3, p-1) = 1$ it follows that cubing is a permutation of $\mathbb{F}_p^*$. This is an HNP-CM problem(here $t = \lambda^3$

and $\alpha = v_1(abD)$). So, in this case, we can get v_1 from LSB(v_1) using the solving algorithm of HNP-CM.

Then combining this with Lemma 1, we can get the entire abD. So Theorem 4 is proved. □

For the general case, similar to Boneh and Shparlinski 's[8] approach on elliptic curve case, we will use the method of Alexi, Chor, Goldreich and Schnorr[3] to deal with it. When λ^2, λ^3, $\lambda^4$and λ^5 are all not permutation of $\mathbb{F}_p^*$, we can not get v_i from LSB(v_i) or u_i from LSB(u_i) using HNP-CM directly. We may only use some $\delta-$fraction of the $\lambda \in \mathbb{F}_p^*$. Therefore, to prove the Theorem 1, we will also use the following lemma.

Lemma 2. *Let $\epsilon, \delta \in (0,1)$. Let p be a prime, and let C be a hyperelliptic curve with genus 2 over F_p. Let $D \in J_C(\mathbb{F}_p)$ be an element of prime order n. Suppose there is a $t-$time algorithm $\mathcal{A}_{u_1}$ such that*

$$|Pr_\lambda[\mathcal{A}_{u_1}(\phi_\lambda(C), J_{\phi_\lambda(C)}(\mathbb{F}_p), \phi_\lambda(D), \phi_\lambda(aD), \phi_\lambda(bD)) = LSB(\lambda^2 u_1(abD))] - \frac{1}{2}| > \epsilon$$

for at least a $\delta-$fraction of the $\lambda \in \mathbb{F}_p^*$.

Then there is an algorithm $\mathcal{B}$ for all $\lambda \in \mathbb{F}_p^*$ satisfying

$$Pr_\lambda[\mathcal{B}(\phi_\lambda(C), J_{\phi_\lambda(C)}(\mathbb{F}_p), \phi_\lambda(D), \phi_\lambda(aD), \phi_\lambda(bD)) = LSB(\lambda^2 u_1(abD))] > \frac{1}{2} + \frac{\epsilon\delta}{8}$$

is true with probability at least $\frac{\epsilon\delta}{8}$ over the choice of a, b in $[1, n-1]$.

Proof. This lemma is the hyperelliptic curve with genus 2 case of Lemma 1 and Lemma 2 in Boneh and Shparlinski 's[8] paper. Here we will give a sketch of the proof which mostly same as Boneh and Shparlinski 's proof. For more details to see Boneh and Shparlinski 's[8] original proof.

According to Boneh and Shparlinski 's proof, the algorithm $\mathcal{B}$ can be constructed as follows:
Input: $C, J_C(\mathbb{F}_p), D, D_1, D_2$.
Output: $\mathcal{A}_{u_1}(C, J_C(\mathbb{F}_p), D, D_1, D_2)$.

1. Pick $u = (4/\epsilon\delta)^3$ random pairs a, b from $[1, n-1]$ and run $\mathcal{A}_{u_1}$ on all tuples $< C, J_C(\mathbb{F}_p), D, aD, bD >$;
2. Let v be the number of runs in which $\mathcal{A}_{u_1}$ correctly outputs LSB($u_1(abD)$);
3. If $v > u/2$ then $\mathcal{B}$ outputs $\mathcal{A}_{u_1}(C, J_C(\mathbb{F}_p), D, D_1, D_2)$;
4. Otherwise outputs the complement of $\mathcal{A}_{u_1}(C, J_C(\mathbb{F}_p), D, D_1, D_2)$.

As same as Boneh and Shparlinski 's[8] discussion, for at least $\delta-$fraction of the $\lambda \in F_p^*$, we have

$$Pr_{a,b}[\mathcal{B}(\phi_\lambda(C), J_{\phi_\lambda(C)}(F_p), \phi_\lambda(D), \phi_\lambda(aD), \phi_\lambda(bD)) = \mathrm{LSB}(\lambda^2 u_1(abD))] > \frac{1}{2}+\frac{\epsilon}{2}$$

and for the remaining $\lambda \in F_p^*$, we have:

$$Pr_{a,b}[\mathcal{B}(\phi_\lambda(C), J_{\phi_\lambda(C)}(F_p), \phi_\lambda(D), \phi_\lambda(aD), \phi_\lambda(bD)) = \mathrm{LSB}(\lambda^2 u_1(abD))] > \frac{1}{2}-\frac{\epsilon\delta}{4}$$

Then using a standard counting argument, we have

$$Pr_{\lambda}[\mathcal{B}(\phi_\lambda(C), J_{\phi_\lambda(C)}(F_p), \phi_\lambda(D), \phi_\lambda(aD), \phi_\lambda(bD)) = \mathrm{LSB}(\lambda^2 u_1(abD))] > \frac{1}{2}+\frac{\epsilon\delta}{8}$$

is true with probability at least $\frac{\epsilon\delta}{8}$ over the choice of a, b in $[1, n-1]$ for all $\lambda \in \mathbb{F}_p^*$.

□

Now, we give the proof of Theorem 3.

The proof of Theorem 3: Let p be a prime, and let C be a hyperelliptic curve with genus 2 over $\mathbb{F}_p$. Let $D \in J_C(\mathbb{F}_p)$ be an element of prime order. Suppose there is an efficient algorithm $\mathcal{A}$ for predicting the LSB of any coordinate of abD given $(C, J_C(\mathbb{F}_p), D, aD, bD)$, formally, we assume that there is an expected $t-$time algorithm $\mathcal{A}$ such that

$$|Pr_{\lambda}[\mathcal{A}_{u_1}(\phi_\lambda(C), J_{\phi_\lambda(C)}(\mathbb{F}_p), \phi_\lambda(D), \phi_\lambda(aD), \phi_\lambda(bD)) = \mathrm{LSB}(\lambda^2 u_1(abD))]-\frac{1}{2}| > \epsilon$$

for at least a $\delta-$fraction of the $\lambda \in \mathbb{F}_p^*$.

To use the above Lemma 2, we first randomize the hyperelliptic curve DHP $(C, J_C(\mathbb{F}_p), D, aD, bD)$ by computing $D' = a_0 aD$ and $D'' = b_0 bD$ for random $a_0, b_0 \in [1, n-1]$. Then applying Lemma 2, there is an algorithm $\mathcal{B}$ for all $\lambda \in \mathbb{F}_p^*$ satisfying

$$Pr_{\lambda}[\mathcal{B}(\phi_\lambda(C), J_{\phi_\lambda(C)}(\mathbb{F}_p), \phi_\lambda(D), \phi_\lambda(D'), \phi_\lambda(D'')) = \mathrm{LSB}(\lambda^2 u_1(a_0 b_0 abD))] > \frac{1}{2}+\frac{\epsilon\delta}{8}$$

is true with probability at least $\frac{\epsilon\delta}{8}$ over the choice of a_0, b_0 in $[1, n-1]$.

Define

$$L^2(\lambda) = \mathcal{B}(\phi_\lambda(C), J_{\phi_\lambda(C)}(\mathbb{F}_p), \phi_\lambda(D), \phi_\lambda(D'), \phi_\lambda(D'').$$

From the knowledge of probability theory, when we repeat choosing a_0, b_0 in $[1, n-1]$ randomly $\lceil \frac{8}{\epsilon\delta} \rceil$ times, then there is at least one time we have

$$Pr_\lambda[L^2(\lambda) = \mathrm{LSB}(\lambda^2 u_1(a_0 b_0 abD))] > \frac{1}{2} + \frac{\epsilon\delta}{8}$$

with probability $1 - (1 - \frac{\epsilon\delta}{8})^{\lceil \frac{8}{\epsilon\delta} \rceil}$. This is an $HNP - CM^2$ problem where $u_1(a_0 b_0 abD)$ is the hidden number. Therefore, we can use the solving algorithm of Theorem 2 for all $\lceil \frac{8}{\epsilon\delta} \rceil$ cases to find a list of candidates $\{(a_i, b_i), u_1(a_i b_i abD)\}$ for i from 1 to $\lceil \frac{8}{\epsilon\delta} \rceil$.

For any candidates, applying Lemma 1, we can get a candidate value $a_i b_i abD$. There is at least one correct $a_i b_i abD$ with probability $1-(1-\frac{\epsilon\delta}{8})^{\lceil \frac{8}{\epsilon\delta} \rceil}$, and then using $((a_i b_i)^{-1} \mod n) a_i b_i abD$, we obtain the entire abD. □

4 Extention to Any Bit

For any $z = \sum_{i=0}^{n} z_i 2^i$, $bit_i(z)$ denotes the $i-$th bit of the binary representation of z, so $\mathrm{LSB}(z) = bit_0(z)$. In this section, we will show that if the hyperelliptic curve Diffie-hellman problem is hard, then not only the least significant bit, but also every bit (i.e., $bit_i(z)$) of the hyperelliptic curve Diffie-hellman value is unpredictable.

We have two approaches to achieve this goal.

One approach is from LSB-HNP-CM to bit_i-HNP-CM. As generalized by J.Hȧstad, M. Näslund [27] and E. Kiltz[31], HNP-CM can also be defined for every bit of z, and the related theorems also hold, i.e., Fix an $\varepsilon > 0$. Let p be a prime. For an $\alpha \in \mathbb{F}_p$ let $L : \mathbb{F}_p* \to \{0, 1\}$ be a function satisfying

$$\mathsf{Pr}_{t \in \mathbb{F}_p*}[L(t) = bit_i(\alpha \cdot t \mod p)] \geq \frac{1}{2} + \epsilon$$

The bit_i-NH problem is: given an oracle for $L(t)$, find α in polynomial time. As claimed in Theorem 5 of [31], for all odd primes p, the bit_i-HNP-CM is efficently solvable for all bits. Therefore, similar to the discussion for LSB case, it is not hard to extend the results of LSB to the case of any $i-$th bit.

Another approach is AGS-list decoding method. The list decoding approach for hard-core predicates is developed by Akavia, Goldwasser, and Safra[2] and extended by Morillo and Rafols[38]. A predicate will correspond to some error correcting code, predicting a predicate will correspond to access to a corrupted codeword, and the task of inverting one-way functions will correspond to the task of list decoding a corrupted codeword. The framework of [2] is: Firstly, Construct a codeword C_f, and such that the following properties hold for

C_f: **Accessibility**, **Concentration** and **Recoverability**, then using Lemma 1 and Theorem 6 of [2], it can be proved that the predicate is hard-core.

Following Akavia et al.'s framework, we can generalize the result of Fazio et al. in [20] for every bit of the elliptic curve DHP is hard-core to hyperelliptic curve DHP as follows:

Let p be a prime, and let C be a hyperelliptic curve with genus 2 over $\mathbb{F}_p$. Let $D \in J_C(\mathbb{F}_p)$ be an element of prime order. The $Q = abD$ is the Diffie-Hellman secret value of $(C, J_C(\mathbb{F}_p), D, aD, bD)$. For any $\lambda \in \mathbb{F}_p^*$, $\phi_\lambda(C)$ is the twist of C, $J_{\phi_\lambda(C)}(\mathbb{F}_p) = \phi_\lambda^*(J_C(\mathbb{F}_p))$. Let $bit_i : \mathbb{F}_p \to \{0,1\}$ denote the $i-th$ bit predicate(In [20], they use $\{\pm 1\}$, it is just the convention that a 0 bit is encoded as -1).

Consider the codeword:

$$C_Q : \mathbb{F}_p \to \{0,1\} \text{ defined as } C_Q(\lambda) = bit_i(\lambda u_1(Q)).$$

Similar to the proof in [20] for elliptic curve case, it can be proven that the codeword C_Q satisfies the properties of **Accessibility**, **Concentration** and **Recoverability**. Using Theorem 6 and the learning algorithm of [2] , it can be proved that this predicate is hard-core. For more detail, refer to [2] and [20] .

From above discussion, we give the following claim without proof:

Claim 1. *Let p be a prime, and let C be a hyperelliptic curve with genus 2 over $\mathbb{F}_p$. Let $D \in J_C(\mathbb{F}_p)$ be an element of prime order. If there is an efficient algorithm for predicting the any bit of any coordinate of abD given $(C, J_C(\mathbb{F}_p), D, aD, bD)$, then there is an efficient algorithm for computing the DHP on $J_C(\mathbb{F}_p)$.*

5 Generalization to General Hyperelliptic Curves

Let $C : y^2 + h(x)y = f(x)$ be a hyperelliptic curve of genus $g \geq 1$ over $\mathbb{F}_q$, $J(C;\mathbb{F}_q)$ be the Jacobian of C defined over $\mathbb{F}_q$. Let $D = (u_{g-1}, ..., u_1, u_0, v_{g-1}, ..., v_1, v_0)$ be an element of $J(C;\mathbb{F}_q)$ with order n. Costello and Lauter[13] gave an explicit formulas for addition and doubling for any genus hyperelliptic Jacobian group. So, we can define the hyperelliptic DHP on any genus hyperelliptic Jacobian group as same as genus 2 case: given $C, J_C(\mathbb{F}_p), D, aD, bD$, to compute abD.

For a hyperelliptic curve with genus g over $\mathbb{F}_p$(Similar discussion can be applied to non-prime fields),

$$C : \ y^2 = x^{2g+1} + f_{2g-1}x^{2g-1} + f_{2g-2}x^{2g-2} + \cdots + f_1 x + f_0$$

Let C' be another hyperelliptic curves with genus g over $\mathbb{F}_p$ with equation:

$$C' : \ y^2 = x^{2g+1} + f'_{2g-1}x^{2g-1} + f'_{2g-2}x^{2g-2} + \cdots + f'_1 x + f'_0$$

We say that C is isomorphic to C' if there exists $\lambda \in \mathbb{F}_p$ such that $f_i' = \lambda^{4g+2-2i} f_i \mod p$. The isomorphisms that preserve hyperelliptic curves given by above equations are all of the form $(x, y) \to (\lambda^2 x, \lambda^{2g+1} y)$ for some $\lambda \in \mathbb{F}_p^*$.

We define $\phi_\lambda : (x, y) \to (\lambda^2 x, \lambda^{2g+1} y)$, then $C' = \phi_\lambda(C)$ is a twist of C. This curves isomorphism can reduce an isomorphism between $J_C(\mathbb{F}_p)$ and $J_{C'}(\mathbb{F}_p)$, we denote this group isomorphism as $\phi_\lambda^* : J_C(\mathbb{F}_p) \to J_{\phi_\lambda(C)}(\mathbb{F}_p)$. Now, we can define the explicit formulas for ϕ_λ^* as follows: $\phi_\lambda^*(O) = O'$, $\phi_\lambda^*(x_1, y_1) = (\lambda^2 x_1, \lambda^{2g+1} y_1)$,

$$\begin{aligned}
\phi_\lambda^*(u(x), v(x)) &= \phi_\lambda^*(u_{g-1}, ..., u_1, u_0, v_{g-1}, ..., v_1, v_0) \\
&= \phi_\lambda^*(P_1 + P_2 + ... + P_g - gP_\infty) \\
&\quad (here\ P_i = (x_i, y_i),\ v(x_i) = y_i) \\
&= \phi_\lambda^*((x_1, y_1) + (x_2, y_2) + ... + (x_g, y_g) - gP_\infty) \\
&= (\lambda^2 x_1, \lambda^{2g+1} y_1) + (\lambda^2 x_2, \lambda^{2g+1} y_2) + ... + (\lambda^2 x_g, \lambda^{2g+1} y_g) - gP'_\infty) \\
&= (\prod_{i=1}^{g} (x - \lambda^2 x_i), v'_{g-1} x^{g-1} + ... + v'_1 x + v'_0) \\
&\quad (here\ v'(\lambda^2 x_i) = \lambda^{2g+1} y_i) \\
&= (\lambda^2 u_{g-1}, ..., \lambda^{2(g-1)} u_1, \lambda^{2g} u_0, \lambda^3 v_{g-1}, ..., \lambda^{2g-1} v_1, \lambda^{2g+1} v_0)
\end{aligned}$$

Therefore, we have $DH_{\phi_\lambda^*(J), \phi_\lambda^*(D)}(\phi_\lambda^*(D_1), \phi_\lambda^*(D_2)) = \phi_\lambda^*(DH_{J,D}(D_1, D_2))$. So if the hyperelliptic DHP is hard to compute in $J_C(\mathbb{F}_p)$, then it is also hard to compute for all $\{J_{\phi_\lambda(C)}(\mathbb{F}_p)\}_{\lambda \in \mathbb{F}_p^*}$. Similar to the case of $g = 2$, we can use HNP-CM^d to study the bit security of hyperelliptic curve DHP with any genus.

From above discussion, we give the following claim without proof:

Claim 2. *Let p be a prime, and let C be a hyperelliptic curve with genus g over $\mathbb{F}_p$. Let $D \in J_C(\mathbb{F}_p)$ be an element of prime order. Given $(C, J_C(\mathbb{F}_p), D, aD, bD)$, if there is an efficient algorithm for predicting any one bit of any coordinate of abD, then there is an efficient algorithm for computing the DHP on $J_C(\mathbb{F}_p)$.*

6 Conclusions and Further Works

Hyperelliptic curve cryptography is an alternative to elliptic curve cryptography. Due to the recent many research work on genus 2 hyperelliptic curve cryptography, especially for their cryptanalysis and fast implementation, that hyperelliptic curve cryptography in genus 2 has the potential to be competitive with its elliptic curve cryptography counterpart. In this paper, we studied the bit security of hyperelliptic Curves DHP, we show that the least significant bit of each coordinate of hyperelliptic Curves Diffie-Hellman secret value K in genus 2 is hard-core, and then we show that any bit is hard-core. Finally, we extend our techniques and results to any genus hyperelliptic curve.

There are some further works at this topic. Similar to elliptic curve case, we can also define a function whose domain is a subgroup of $J_C(\mathbb{F}_p)$, such as hyperelliptic pairing. When we consider the one-way function defined over the Jacobian of hyperelliptic curve, we call such function "hypereelliptic curve based one-way function", following the approach of Duc and Jetchev[17] for elliptic curve case, it seems that all the bits of hyperelliptic curve based one way functions are hard to compute too.

Jetchev and Venkatesan[30] studied the bits security of elliptic curve Diffie-Hellman secret keys using elliptic curves isogeny theory. The hyperelliptic Jacobians also have explicit isogenies, there are some research work on them. An natural question is if we can study the bits security of hyperelliptic curve Diffie-Hellman secret keys using hyperelliptic curves isogeny theory. It seems this can also be done.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 61379154 and U1135001). Part of this work was done during the author was visiting the UbiSeC lab at University at Buffalo, State University of New York.

References

1. L. Adleman, J. De Marrais, M.-D Huang, *A Subexponential Algorithm for Discrete Logarithms over the Rational Subgroup of the Jacobians of Large Genus Hyperelliptic Curves over Finite Fields*, in ANTS-1, Algorithmic Number Theory , Springer-Verlag, LNCS 877, pp. 28-40, 1994.
2. A. Akavia, S. Goldwasser, and S. Safra. Proving hard-core predicates using list decoding. FOCS 2003, pp. 146-157, IEEE Computer Society, 2003.
3. W.Alexi, B.Chor, o.Goldreich and C.Schnorr. RSA and Rabin functions: Certain parts are as hard as the whole. SIAM J. Computing, 17(1988), pp.194-209, Nov. 1988.
4. R.Avanzi, H.Cohen, C.Doche, G.Frey, T.Lange, K.Nguyen and F.Vercauteren, Handbook of Elliptic and Hyperelliptic Cryptography, Chapman and Hall/CRC, 2006.
5. D. J. Bernstein, C. Chuengsatiansup, T. Lange, and P. Schwabe. Kummer strikes back: new DH speed records. ASIACRYPT 2014, LNCS 8873, pp. 317-337, 2014.
6. M. Blum and S. Micali. How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits. SIAM Journal on Computing, Vol. 13, No. 4:850-864, 1984
7. I. F. Blake, T. Garefalakis, and I. E. Shparlinski. On the bit security of the Diffie- Hellman key. In Appl. Algebra in Engin., Commun. and Computing, volume 16, pages 397-404, 2006.
8. D. Boneh and I. E. Shparlinski. On the unpredictability of bits of the elliptic curve diffie-hellman scheme. CRYPTO 2001, LNCS 2139, pp. 201-212, 2001.
9. D. Boneh, R. Venkatesan. Hardness of Computing the Most Significant Bits of Secret Keys in Diffie-Hellman and Related Schemes. CRYPTO 1996. LNCS 1109, pp. 129-142. Springer, Heidelberg (1996)

10. D. Boneh, B. Lynn, and H. Shacham, *Short signatures from the Weil pairing*, Asiacrypt 2001, LNCS 2248, pp.514-532, Springer-Verlag, 2001.
11. J. W. Bos, C. Costello, H. Hisil, and K. Lauter. Fast cryptography in genus 2. In T. Johansson and P. Q. Nguyen, editors, EUROCRYPT 2013, LNCS 7881, pp.194-210. Springer, 2013
12. D.G. Cantor, *Computing in the Jacobian of a hyperelliptic curve*, Mathematics of Computation, Volume 48, pp.95-101, 1987.
13. C. Costello and K. Lauter. Group law computations on Jacobians of hyperelliptic curves. In A. Miri and S. Vaudenay, editors, Selected Areas in Cryptography, LNCS 7118, pp. 92-117. Springer, 2011.
14. N.Courtois, A. Klimov, J. Patarin, and A. Shamir. Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. Advances in Cryptology-Eurocrypt 2000, LNCS 1807, pp. 392-407, Springer, 2000.
15. O. Diao and M. Joye. Unified addition formulæ for hyperelliptic curve cryptosystems. In 3rd Workshop on Mathematical Cryptology (WMC 2012) and 3rd International Conference on Symbolic Computation and Cryptography (SCC 2012), pages 45-50, 2012.
16. W. Diffie and M. Hellman, New Directions in cryptography, IEEE Transactions on Information Theory, volume 22, pp. 644-654, 1976.
17. A. Duc and D. Jetchev. Hardness of computing individual bits for one-way functions on elliptic curves. CRYPTO 2012, LNCS 7417, pp. 832-849, 2012.
18. T. ElGamal, A public-key cryptosystem and a signature scheme based on discrete logarithms, IEEE Transactions on Information Theory, volume 31, pp. 469-472, 1985.
19. FIPS 186-2, Digital signature standard, Federal Information Processing Standards Publication 186-2, February 2000.
20. N. Fazio, R. Gennaro, I. M. Perera, and W. E. Skeith III. Hard-core predicates for a Diffie-Hellman problem over finite fields. CRYPTO 2013, LNCS 8043, pp. 148-165, 2013.
21. S.D. Galbraith, Mathematics of Public Key Cryptography. Cambridge university Press, 2012.
22. S.D. Galbraith, H.J. Hopkins, and I.E. Shparlinski. Secure Bilinear Diffie-Hellman Bits. ACISP 2004. LNCS 3108, pp. 370-378. Springer, Heidelberg (2004)
23. P. Gaudry. Fast genus 2 arithmetic based on theta functions. Journal of Mathematical Cryptology JMC, 1(3):243-265, 2007.
24. P. Gaudry, E.Thomé, N. Thériault and C. Diem. A doule large prime variation for small genus hyperelliptic index calculus. Math. Comp. 76(257), pp.475-492, 2007.
25. M. I. Gonzalez Vasco, M. Näslund, and I. E. Shparlinski. New results on the hardness of Diffie-Hellman bits. In PKC 04, LNCS 2947, pages 159-172, 2004.
26. R. Harley. Fast arithmetic on genus 2 curves. See http://cristal.inria.fr/∼harley/hyper for C source code and further explanations.
27. J.Håstad, M. Näslund, The security of all RSA and discrete log bits. J ACM 51(2):187-230, 2004.
28. H. Hisil1 and C. Costello. Jacobian Coordinates on Genus 2 Curves, AsiaCrypt 2014, LNCS 8873, pp.338-357, 2014.
29. D. Jao, D. Jetchev and R Venkatesan. On the bits of elliptic curve Diffie-Hellman keys. Indocrypt 2007, LNCS 4859, pp. 33-47, 2007.
30. D. Jetchev and R Venkatesan. Bits security of the elliptic curve Diffie-Hellman secret keys. Crypt 2008, LNCS 5157, pp. 75-92, 2008.
31. E. Kiltz, A primitive for proving the security of every bit and about universal hash functions and hard core bits, Proc. of FCT'01, pp.388-391, 2001.
32. N. Koblitz, *Hyperelliptic cryptography*, J.of Crypto., No.1, pp. 139-150, 1989.

33. Lange, T.: Formulae for Arithmetic on Genus 2 Hyperelliptic Curves. Applicable Algebra in Engineering, Communication and Computing 15(5), pp.295-328 (2005)
34. W.-C. W. Li, M. Näslund and I. E. Shparlinski, The hidden number problem with the trace and bit security of XTR and LUC, Crypto2002, LNCS 2442, Springer-Verlag, Berlin, (2002), pp.433-448.
35. MAGMA Computational Algebra System, http://magma.maths.usyd.edu.au/magma/
36. U. M. Maurer, Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete logatithms, CRYPTO94, LNCS 839, pp. 271-281, Springer-Verlag, 1994.
37. U. M. Maurer and S. Wolf, The Relationship Between Breaking the Diffie-Hellman Protocol and Computing Discrete Logarithms, SIAM J. Comput. 28(5): 1689-1721, 1999.
38. P. Morillo, C. Ràfols, The Security of All Bits Using List Decoding. PKC 2009. LNCS 5443, pp. 15-33. Springer, 2009.
39. S. C. Pohlig and M. E. Hellman, An improved algorithm for computing logarithms over GF(p) and its cryptographic significance, IEEE-Transactions on Information Theory 24, pp. 106-110, 1978.
40. M. Wang, T. Zhan, and H. Zhang. Bits Security of the CDH Problems over Finite Fields, in Cryptology ePrint Archive, Report 2014/685.

Appendix A: Explicit formulas for addition and doubling in genus 2

Input:$D_1 = (u_{11}, u_{10}, v_{11}, v_{10}), D_2 = (u_{21}, u_{20}, v_{21}, v_{20}))$	
Output:$D_3 = D_1 + D_2 = (u_{31}, u_{30}, v_{31}, v_{30})$	
Step	*Expression*
1	$A = (v_{10} - v_{20})(u_{21}(u_{11} - u_{21}) - (u_{10} - u_{20})) - u_{20}(u_{11} - u_{21})(v_{11} - v_{21})$ $B = (u_{10} - u_{20})(u_{21}(u_{11} - u_{21}) - (u_{10} - u_{20})) - u_{20}(u_{11} - u_{21})^2$ $C = (u_{11} - u_{21})(v_{10} - v_{20}) - (u_{10} - u_{20})(v_{11} - v_{21})$
2	$u_{31} = (u_{11} - u_{21}) + 2\frac{A}{C} - \frac{B^2}{C^2}$ $u_{30} = (u_{11} - u_{21})\frac{A}{C} + \frac{A^2}{C^2} + (u_{11} + u_{21})\frac{B^2}{C^2} - (v_{11} + v_{21})\frac{B}{C}$ $v_{31} = (u_{10} - u_{30})\frac{C}{B} - u_{31}(u_{11} - u_{31})\frac{C}{B} + (u_{11} - u_{31})\frac{A}{B} - v_{11}$ $v_{30} = (u_{10} - u_{30})\frac{A}{B} - u_{30}(u_{11} - u_{31})\frac{C}{B} - v_{10}$
3	$Output:\ (u_{31}, u_{30}, v_{31}, v_{30})$

Table 1. Addition in genus 2

From above explicit formulas, the coordinates u_{3i} or v_{3i} for $i = 0, 1$ of D_3 can be regarded as a rational function of $u_{11}, u_{10}, v_{11}, v_{10}, u_{21}, u_{20}, v_{21}$ and v_{20}.

Appendix B: Magma program

```
p:=2^127-1; K := GF(p);
P<x> := PolynomialRing(GF(p));
f3:= 34744234758245218589390329770704207149;
f2:= 132713617209345335075125059444256188021;
```

Input:$D_1 = (u_{11}, u_{10}, v_{11}, v_{10})$	
Output:$D_3 = 2D_1 = (u_{31}, u_{30}, v_{31}, v_{30})$	
Step	*Expression*
1	$A = ((u_{11}^2 - 4u_{10} + f_3)u_{11} - f_2 + v_{11}^2)(u_{11}v_{11} - v_{10}) + (3u_{11}^2 - 2u_{10} + f_3)(u_{10}v_{11})$ $B = 2(u_{11}v_{11}) - v_{10})v_{10} - 2u_{10}v_{11}^2$ $C = ((u_{11}^2 - 4u_{10} + f_3)u_{11} - f_2 + v_{11}^2)v_{11} + (3u_{11}^2 - 2u_{10} + f_3)v_{10}$
2	$u_{31} = 2\frac{A}{C} - \frac{B^2}{C^2}$ $u_{30} = \frac{A^2}{C^2} + 2u_{11}\frac{B^2}{C^2} - 2v_{11}\frac{B}{C}$ $v_{31} = (u_{10} - u_{30})\frac{C}{B} - u_{31}(u_{11} - u_{31})\frac{C}{B} + (u_{11} - u_{31})\frac{A}{B} - v_{11}$ $v_{30} = (u_{10} - u_{30})\frac{A}{B} - u_{30}(u_{11} - u_{31})\frac{C}{B} - v_{10}$
3	$Output: \ (u_{31}, u_{30}, v_{31}, v_{30})$

Table 2. Doubling in genus 2

```
f1:= 90907655901711006083734360528442376758;
f0:= 66679866221737283378235608571799928 16;
C := HyperellipticCurve(x^5+f3*x^3+f2*x^2+f1*x+f0);
J := Jacobian(C); D:=Random(J);
n:=289480223093290488481692399956590251384511779730915513 74
   101475732892580332259;

a:=Random(1,n); b:=Random(1,n);
A:=a*D;
ua1:=Coefficient(A[1], 1); ua0:=Coefficient(A[1], 0);
va1:=Coefficient(A[2], 1); va0:=Coefficient(A[2], 0);
B:=b*D; C:=a*B;
M:=(b+1)*A;
uab1:=Coefficient(C[1], 1);
uaba1:=Coefficient(M[1], 1);

P3<x,y,z> := PolynomialRing(K, 3);
g1:=z^2-y^2*x+2*uab1*x^2+(f2-uab1*f3-uab1^3)*x-f0;
g2:=2*z*y-x^2-uab1*y^2+(f3 +3*uab1^2)*x+f2*uab1-f3*uab1^2-uab1^4-f1;
g3:=((uab1-ua1) *(z-va0) - (x- ua0)* (y-va1))^2*(uaba1- uab1+ua1)
-2*((z-va0)*(ua1*(uab1-ua1)- (x- ua0)) - ua0*(uab1-ua1)*(y-va1))
*((uab1-ua1)*(z-va0) - (x- ua0)* (y-va1))+((x- ua0)* (ua1*(uab1-ua1)
 - (x- ua0))- ua0*(uab1-ua1)^2 )^2;

I := ideal<P3 | g1, g2, g3>;
v := Variety(I, K);
v;
```