Attacks on Elliptic Curve Cryptography

Joshua Dong

December 11, 2016

Abstract

Assumptions on the hardness of problems such as the discrete logarithm problem and integer factorization are challenged by advances in modern cryptography and computing. Advances in quantum computing cast significant doubt on the security of currently used algorithms such as Diffie-Hellman, ElGamal, and RSA which rely on these assumptions of hardness. Growing discussion on elliptic curve cryptography is promising, but this paper looks to challenge and qualify the merits of elliptic curve cryptography. With many pitfalls in theory and implementation, elliptic curve protocols are reviewed with a focus on mitigating known attacks.

1 Background

1.1 Hyperelliptic Curves

A *hyperelliptic curve* is a plane curve of the form

$$y^2 = f(x), \tag{1}$$

where the n degree polynomial $f(x)$ has n distinct roots. Then the genus of such a curve is g where $n = 2g+1$ or $n = 2g + 2$. Elliptic curves are hyperelliptic curves of genus 1.

One may wonder why elliptic curves of genus 1, and not conics or hyperelliptic curves of genus greater than 1 are used in cryptography. The answer lies in a balance between computational simplicity, group operation complexity, and smoothness properties of the curve used. Conic cryptography, while fast, adds no security benefit in terms of hardness in proportion to key size (and in the case of singular curves with a cusp have polynomial time solutions) [15]. The weaknesses of hyperelliptic curves of higher genus is discussed later.

1.2 Edwards Curves

There are many forms of elliptic curves and many ways to represent them. Optimizing the group operations in finite fields over elliptic curves has been a major area of research. This section provides a brief review of the topics and notations relevant to this paper.

The general Weierstrass form of an elliptic curve E is:

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6. \tag{2}$$

$E(K)$ is the set of points $(x, y) \in K^2$ for some field K. Adding a "point at infinity" to serve as the identity element, the set forms an abelian group under point addition. When K does not have characteristic 2 or 3, E may be represented in the more common form

$$y^2 = x^3 + ax + b \tag{3}$$

where a and b are subject to constraints preventing cusps and self-intersections. Every elliptic curve over a finite field may be represented as an *Edwards curve* [4] of the form

$$x^2 + y^2 = c^2(1 + x^2y^2) \tag{4}$$

These Edwards curves not only generalize elliptic curves, but also benefit from a unified addition law that is beneficial for computation:

$$(x_1, y_1) + (x_2, y_2) = (\frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2}, \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2}). \tag{5}$$

The unified addition law is not only convenient, but is useful against simple power analysis attacks which exploit non-unified forms of the group operation as seen some other elliptic curves. The results of Bernstein, Birkner, Joye, Lange, and Peters deal with variants of Edwards curves over non-binary fields that improve efficiency of addition and doubling in the group.

1.3 Scalar Multiplication

A common hardness assumption in elliptic curve cryptography is the elliptic curve discrete logarithm problem. The idea is that for a point P on an elliptic curve E, a secret integer d is hard to determine only given P, E, and dP. There are many ways to multiply P by d in sub-linear time. Relevant multiplication algorithms are covered below.

The most straightforward logarithmic implementation of scalar multiplication is a variant of the "square-and-multiply" algorithm used in RSA.

Algorithm 1 Double-and-Add from the most significant bit

```
Require: $d, P$
Ensure: $Q = dP$
  $Q := P$
  for $i = n - 2 \to 0$ do
    if $d_i = 1$ then
      $Q := Q + P$
    end if
  end for
  return Q
```

This implementation is vulnerable to simple power analysis or timing attack due to the branch statement that causes uneven computations correlated to the bit pattern of the secret [7]. Thus, Coron proposed the double-and-add-always method which always executes the addition and prevents this attack.

Algorithm 2 Double-and-Add-Always from the most significant bit

```
Require: $d, P$
Ensure: $Q_0 = dP$
  $Q_0 := P$
  for $i = n - 2 \to 0$ do
    $Q_0 := 2Q_0$
    $Q_1 := Q_0 + P$
    $Q_0 := Q_{d_i}$
  end for
  return $Q_0$
```

This algorithm for scalar multiplication removes the computation difference between 0 and 1 bits.

Also used for scalar multiplication is the sliding-window method, which is the same as the double-and-add algorithm but chooses a fixed window value w. It similarly does not give away information in the manner of the Double-and-Add algorithm.

Algorithm 3 Sliding-Window Method

```
Require: $d, P$
Ensure: $Q_0 = dP$
  $Q = 0$
  for $i = m \to 0$ do
    if $d_i = 0$ then
      $Q := 2Q$
    else
      $t :=$ extract $min(j, w - 1)$ additional bits from $d$
      $i := i - j$
      if $j < w$ then
        $Q := tQ$
        return Q
      else
        $Q := 2^w Q$
        $Q := Q + tP$
      end if
    end if
  end for
```

Montgomery originally proposed the Montgomery method for scalar multiplication in constant time [16]. It is also resistant against simple power analysis [17].

Algorithm 4 Montgomery Ladder

```
Require: $d, P$
Ensure: $Q_0 = dP$
  $Q_0 := P$
  $Q_1 := 2P$
  for $i = n - 2 \to 0$ do
    $Q_{1-d_i} := Q_0 + Q_1$
    $Q_{d_i} := 2Q_{d_i}$
  end for
  return $Q_0$
```

These multiplication algorithms account for all major algorithms in use for scalar multiplication on elliptic curves. However, each of these algorithms have weaknesses to side-channel attacks, as explained later.

1.4 Obfuscations

Presented below are some of the commonly used obfuscations on scalar multiplication to prevent side-channel attacks on elliptic curve discrete logarithm protocols. The details are covered in Goubin's work, but a summary is provided here for context [10].

1.4.1 Random Projective Coordinates

Multiplication of a scalar d and a point P is performed in the projective coordinates. The base-point $P = (x, y)$ is represented as homogeneous or Jacobian projected coordinates on some random $\theta \in K$, where θ is invertible. Then dP can be computed in the projective space on θ, and then transformed back into affine coordinates using θ^{-1}.

1.4.2 Random Elliptic Curve Isomorphisms

For some elliptic curve $E : y^2 = x^3 + ax + b$ on a field K of characteristic not 2 or 3, scalar multiplication is done by choosing a random $\theta \in K$ and computing using homogeneous projective coordinates dP', where $P' = (\theta^2 x, \theta^3 y, 1)$ in $E' : Y^2 Z = X^3 + \theta^4 XZ^2 + \theta^6 bZ^3$.

1.4.3 Random Field Isomorphisms

An elliptic curve E is applied over a Galois field of order 2^m, $GF(2^m)$. Then the field can also be represented as the quotient group $GF(2)[x]/f(x)$, where $f(x)$ is an irreducible polynomial of degree m over $GF(2)$. Since many such polynomials exist, the idea is that many isomorphic fields can be created easily. Since all these finite fields are still isomorphic to $GF(2^m)P$, an inverse function (of the isomorphism) can be applied after multiplication in the image of the isomorphism in E.

1.5 Supersingular Isogenies

Quantum cryptography provides effective solutions to the discrete logarithm for elliptic curves. This is because the discrete logarithm can be efficiently computed over arbitrary fields using Shor's algorithm [19]. Fortunately, there are more hard problems in elliptic curve cryptography such as problems concerning supersingular isogenies, which are still hard for quantum computers to solve. We introduce the context of supersingular elliptic curves and isogenies here, but the application to security is discussed later.

Suppose we have two elliptic curves E_1 and E_2 over a finite field $\mathbb{F}_q$. A rational mapping between two algebraic varieties (such as elliptic curves) is the data of a partial function $f : E_1 \supset U \to X$ modulo the equivalence relation $f_1 : E_1 \supset U_1 \to X \equiv f_2 : E_1 \supset U_2 \to X$ if there exists $W \subset U_1 \cap U_2$ where f_1 and f_2 agree on W [23]. An *isogeny* $\varphi : E_1 \to E_2$ over $\mathbb{F}_q$ is a rational mapping that is also a group homomorphism from $E_1(\mathbb{F}_q)$ to $E_2(\mathbb{F}_q)$. Two elliptic curves E_1 and E_2 defined over $\mathbb{F}_q$ are *isogenous* over $\mathbb{F}_q$ if there exists an isogeny between them.

An *endomorphism* of an elliptic curve E defined over $\mathbb{F}_q$ is a self-homomorphism isogeny defined over $\mathbb{F}_{q^m}$ for some m. The set of endomorphisms of E together with the zero map is a ring under pointwise addition and function composition; this ring is called the *endomorphism ring* of E and denoted $End(E)$ [12]. When the ring End(E) is isomorphic to an order in a quaternion algebra we say that E is *supersingular*. That is, the number of basis vectors $(dim(Im(T)))$ of the endomorphism transformation is the rank of the endomorphism, which is always 4 for supersingular elliptic curves.

The curves over a field $\mathbb{F}_q$ with equivalence under isomorphism in $\overline{\mathbb{F}}_q$ form an *isogeny graph*. Less formally, it is a graph with nodes representing the equivalence classes of elliptic curves under isomorphism. Note that j-invariants are used to represent these equivalence classes, easy to compute from a given elliptic curve $y^2 = x^3 + px + q$:

$$j = 1728 \frac{4p^3}{4p^3 + 27q^2}. \tag{6}$$

2 Attacks

This section will look towards the exploitation of weaknesses in elliptic curve cryptography. There are both theoretical and implementation issues that arise when building a crypto-system, and elliptic curve cryptography is no exception. Attention to implementing a cryptographic protocol is important, as events such as the introduction of DUAL_EC_DRBG emphasize. While the topic of back-doors will not be discussed here, we will look at standard mathematical and side-channel attacks on currently existing implementations of elliptic curve cryptography.

2.1 Mathematical Attacks

Insecurity of a protocol can be shown with an algorithm that compromises the hardness assumptions of that protocol. Security is much harder to prove, as assumptions of hardness are but assumptions. In the following sections, some of the major attempts to find weaknesses in elliptic curve cryptography will be discussed. This paper does not cover the MOV attack in detail, an attack that reduces the elliptic curve discrete logarithm problems into the discrete logarithm over multiplicative integer groups by creating a bilinear pairing from $E(\mathbb{F}_q)$ into $\mathbb{F}_{q^k}$. The MOV attack can be mitigated by choosing curve parameters with a large embedding degree k.

2.1.1 Index Calculus

Index calculus algorithmically computes discrete logarithms of finite integer groups with prime order. However, the probabilistic algorithm may be applied to other groups. Attempts to use index calculus on elliptic curves have failed due to the differences between the smoothness of finite fields on elliptic curves and integers modulo some n.

Index calculus algorithms rule out the usage of hyperelliptic curves of genus greater than or equal to 3, because of the same smoothness characteristics that make multiplicative groups weak to index calculus [22] [9]. Hyperelliptic curves of increasing genus have increasingly hard-to-compute group operations, and hyperelliptic curves of genus 1 have the fast group operations [8]. For both these reasons, hyperelliptic curves of genus 1 are the focus of cryptographic protocols.

Since prime elements in elliptic curves do not have the same properties as primes in multiplicative groups, it is impossible to find an effective factor base for index calculus. The reason for this is that for index calculus in the multiplicative group, if r is the rank of the chosen factor base, then $p_1, p_2, ..., p_r$, the primes in the factor base, are small such that

$$\log p_i \approx \log i \leq \log r. \tag{7}$$

However, when choosing a rank r for elliptic curves, the prime elements of the factor base are on the order of $r \log r$ [21]. This means that there will be no rank r such that both computation is easy (sub-exponential) and the probability that a lifted candidate from the index calculus algorithm is p_r-smooth is high. Choosing a small r for sub-exponential computation never results in a decent collision probability, since the primes of elliptic curve groups are too large. At the same time, choosing a large r defeats the purpose of index calculus, as the system of equations (the computation bottleneck) to solve becomes proportionately large. Additionally, we have assumed that the during the lifting step of index calculus, the scalar multiplication was done into the global field $\mathbb{Q}$. Other lifting strategies have been proposed, none of which are effective [20]. Thus, index calculus is not particularly effective for solving elliptic curve discrete logarithms.

2.1.2 Quantum attacks

It is well known that integer factorization can be solved in sub-exponential time using quantum computing [1] [14]. Thus concerns have arisen over the use of RSA and similar algorithms which rely on the hardness of non-quantum integer factorization. However, other previously considered difficult problems such as the discrete logarithm problem also become solvable in sub-exponential time as a consequence of quantum computers [3].

While Shor's algorithm provides a means for sub-exponential integer factorization and discrete logarithms, it also provides a sub-exponential solution to the elliptic discrete logarithm problem. This is significant because fewer q-bits are needed in elliptic curve cryptography than for equivalent-hardness problems of integer factorization or the discrete logarithm over finite fields. In fact, the bottleneck in reducing the number of q-bits needed for solving elliptic discrete logarithms is in the use of the extended Euclidian algorithm during division, not modular exponentiation [18]. Thus, breaking 224-bit elliptic curve cryptography (equivalent to 2048-bit RSA) would require less than 1600 q-bits. To put this in perspective, the most powerful universal quantum computers of 2016 only had 5 q-bits of computing power [11]. While the future of quantum computing is far from certain, the theoretical risks are strong motivator to move to quantum-resistant hardness assumptions.

Supersingular isogeny Diffie-Hellman (SIDH) key exchange, however, relies on finding the isogeny mapping between two supersingular elliptic curves with the same number of points. Recall that an isogeny is a surjective mapping between curves that preserves the base points (generators) and structure (homomorphic). For elliptic curves with endomorphism rank not equal to 4, the endomorphism ring is commutative, and ideal classes form a finite abelian group. This property, along with the use of a special group action, turns the problem of finding an isogeny into the abelian group hidden shift problem, which can be solved in quantum sub-exponential time [5] [6]. As quaternion algebra in not commutative, no group action exists for supersingular elliptic curves and this problem reduction is not possible. The hardness of finding supersingular isogenies is the basis of SIDH, where public keys are supersingular curves and private keys are the isogenies. SIDH shared keys are produced by applying a private isogeny to a public curve, creating curves of equal j-invariant (isomorphic, thus equivalent under the isogeny graph) [12]. Feo, Jao, and Plut provide a detailed explanation.

2.2 Side-channel attacks

A side-channel attack is an attack which monitors a source of side-channel data leakage to infer sensitive data. Electro-magnetic radiation, current usage, and timing differences are all examples of side-channels through which data may be leaked. A motivating factor behind elliptic curve cryptography adoption is the smaller key sizes, as key compromise is easier with larger keys [24] [25]. While longer key sizes provide more mathematical security, the security of practical implementations generally suffer due to an increased attack surface with respect to side-channel data leaks.

For security protocols relying on the elliptic curve discrete logarithm, the most sensitive step of computation is the scalar multiplication of a point P by a factor n. Whatever the method used to multiply, if every operation of the processor were known, then the key would be easily extracted. However, operations or branches are only inferred by increases in power or time usage. This section investigates the some security concerns of elliptic curve cryptographic implementations currently in use, all of which rely on the elliptic curve discrete logarithm.

2.2.1 Differential Power Analysis

Differential power analysis is the interpretation of electrical usage over time using statistics. The attack exploits biases in power consumption of hardware during cryptographic computations. Incorporating signal processing methods, differential power analysis can extract secrets from measurements which contain random or systematic noise.

Random projective coordinates, random elliptic curve isomorphisms, and random field isomorphisms, are three common methods of obfuscating against power analysis attacks have be briefly covered above. This is because the introduced randomness denies an attacker knowledge of the internal state, given a random enough parameter. The natural approach to thwarting these measures uses statistics to attempt to "even out" the random noise. Differential power analysis can indeed thwart these obfuscation attempts, even when used in conjunction with even-power usage multiplications.

For example, when using Double-and-Add-Always to mitigate side-channel data leakage, two sets of inputs can be fed to the multiplier to find the value of a bit with index i (given prior knowledge of the bits before i). The two sets of inputs may be associated with some feature of the output in such a way that distinguishes the power consumption curves using correlation with the assigned set of the input. More formally, for an $n+1$ bit secret $d = (d_{n-1}, d_{n-2}, ..., d_0)$ and prior knowledge of bits $d_{n-1}, ..., d_{j+1}$, an attacker can guess that d_j is 1 and choose arbitrary points $P_0, P_1, ..., P_t$. Then the attacker computes $Q_r = \sum_{i=j}^{n-1} k_i 2^i P_r$ for every possible P_r. Then the attacker divides the set $\{P_0, P_1, ..., P_t\}$ into two sets $A = \{A_0, ...\}$ and $B = \{B_0, ...\}$ such that $P_i \in A$ if $\tau(Q_i) = true$, else $P_i \in B$, where τ is a boolean function dependent on its parameter but also splitting elements somewhat evenly between A and B ($|A| \neq 0, |B| \neq 0$). Encrypting the values $P_0, P_1, ..., P_t$ and observing the current, differences between the respective side-channel signals for the sets A and B will be near 0 if the guess is incorrect (no correlation with set assignment) and non-zero if the guess is correct (non-zero correlation with set assignment). Specifically, correlation between the signal $\phi(r)$ associated with computing kP_r can be measured with

$$Average_{P_r \in A}(\phi(r)) - Average_{P_r \in B}(\phi(r)). \tag{8}$$

After the value of d_j is determined, the same strategy can be applied recursively to determine the entire secret. This method will work for any side-channel, but fails when random transformations are used (such as random projective coordinates) [13].

To tackle implementations using both even-power scalar multiplication as well as randomization (via projection, group isomorphism, or curve isomorphism), a similar approach can be used. Since even under random isomorphism or random point projection, projective coordinate zeros are still zero, a special point $P' = (X, Y, Z)$ not equal to the identity with $X = 0$ or $Y = 0$ or $Z = 0$ can be used to determine the secret key. The details of attacking every combination of obfuscation method with every scalar multiplication algorithm are detailed by Goubin [10].

2.2.2 Flush+Reload

The Flush+Reload attack [26] is a specialized timing attack exploiting cache optimizations on Intel x86 processors. It operates by differentiating cached and non-cached memory using timing differences. A spy program periodically flushes the memory line from the processor cache, then after a short delay, loads data from memory. Since the memory line is flushed at the beginning, the spy program will know whether the cached instructions were used by measuring the access time of memory reads. The attack is mitigated by replacing branch instructions or implementing access controls to memory pages.

The Elliptic Curve Digital Signature Algorithm implemented in the OpenSSL libraries before version 1.0.0m were vulnerable to the Flush+Reload attack. Old versions used a naïve implementation of the Montgomery ladder for scalar multiplication:

```
for (; i >= 0; i--)
{
    word = scalar->d[i];
    while (mask)
    {
        if (word & mask)
        {
            if (!gf2m_Madd(group, &point->X, x1, z1, x2, z2, ctx))
                goto err;
            if (!gf2m_Mdouble(group, x2, z2, ctx))
                goto err;
        }
        else
        {
            if (!gf2m_Madd(group, &point->X, x2, z2, x1, z1, ctx))
                goto err;
            if (!gf2m_Mdouble(group, x1, z1, ctx))
                goto err;
        }
        mask >>= 1;
    }
    mask = BN_TBIT;
}
```

ec2_mult_old.c

In this code, the first `if` statement can be monitored by a Flush+Reload spy program to infer each bit of `word`, the secret key. Details of how this was executed are explained by Yarom and Benger [25].

The branching issue was mediated with a constant-time conditional swap. This implementation does not leak data because the memory lines do not hold differentiating instructions.

```
for (; i >= 0; i--) {
    word = bn_get_words(scalar)[i];
    while (mask) {
        BN_consttime_swap(word & mask, x1, x2, bn_get_top(group->field));
        BN_consttime_swap(word & mask, z1, z2, bn_get_top(group->field));
        if (!gf2m_Madd(group, point->X, x2, z2, x1, z1, ctx))
            goto err;
        if (!gf2m_Mdouble(group, x1, z1, ctx))
            goto err;
        BN_consttime_swap(word & mask, x1, x2, bn_get_top(group->field));
        BN_consttime_swap(word & mask, z1, z2, bn_get_top(group->field));
        mask >>= 1;
    }
    mask = BN_TBIT;
}
```

ec2_mult_new.c

Work has also produced similar success on the implementation of **secp256k1** in OpenSSL for sliding-window multiplication [2]. However, the attack described is more complicated as the partial knowledge of point addition and doubling sequences is combined with a lattice reduction approach to determine the 256 bit key.

3 Conclusion

Research on elliptic curve curve cryptography is still ongoing. The major selling point of elliptic curve cryptographic protocols is a much smaller key size. Computations are more expensive than equivalent-security algorithms using integer factorization or the multiplicative discrete logarithm problem. However, a smaller key size offers significant security benefits against side channel attacks. If quantum computing is to pose a real challenge to current cryptographic protocols, then those protocols built on the elliptic curve discrete logarithm will be the first to become ineffective. Other hard problems exist that build on elliptic curve theory, such as the problem of finding an isogeny mapping between supersingular elliptic curves – which turns out to be resistant to quantum attacks. Even in a world without powerful quantum computers, care must be taken when implementing cryptographic protocols, since side-channel attacks can take place invisibly, without contact. Elliptic curve theory is promising for cryptography, but like other basis of security assumptions, may only seem secure because fast algorithms have yet to be discovered.

References

[1] David Beckman et al. "Efficient Networks for Quantum Factoring". In: (1996). DOI: `10.1103/PhysRevA.54.1034`. eprint: `arXiv:quant-ph/9602016`.

[2] Naomi Benger et al. "fffdfffdfffdOoh Aah... Just a Little Bitfffdfffdfffd: A small amount of side channel can go a long way". In: *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer. 2014, pp. 75–92.

[3] Daniel J. Bernstein. *Post-quantum cryptography*. Introductory Chapter. 2009.

[4] Daniel J Bernstein et al. "Twisted edwards curves". In: *International Conference on Cryptology in Africa* (2008). Springer Berlin Heidelberg, pp. 389–405.

[5] Andrew M Childs and Wim van Dam. "Quantum algorithm for a generalized hidden shift problem". In: *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics. 2007, pp. 1225–1232.

[6] Andrew Childs, David Jao, and Vladimir Soukharev. "Constructing elliptic curve isogenies in quantum subexponential time". In: *Journal of Mathematical Cryptology* 8.1 (2014), pp. 1–29.

[7] Jean-Sébastien Coron. "Resistance against differential power analysis for elliptic curve cryptosystems". In: *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer. 1999, pp. 292–302.

[8] Craig Costello and Kristin Lauter. "Group Law Computations on Jacobians of Hyperelliptic Curves". In: *Selected Areas in Cryptography: 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers*. Ed. by Ali Miri and Serge Vaudenay. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 92–117. ISBN: 978-3-642-28496-0. DOI: `10.1007/978-3-642-28496-0_6`. URL: `http://dx.doi.org/10.1007/978-3-642-28496-0_6`.

[9] Pierrick Gaudry. "An Algorithm for Solving the Discrete Log Problem on Hyperelliptic Curves". In: *Advances in Cryptology — EUROCRYPT 2000: International Conference on the Theory and Application of Cryptographic Techniques Bruges, Belgium, May 14–18, 2000 Proceedings*. Ed. by Bart Preneel. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 19–34. ISBN: 978-3-540-45539-4. DOI: `10.1007/3-540-45539-6_2`. URL: `http://dx.doi.org/10.1007/3-540-45539-6_2`.

[10] Louis Goubin. "A refined power-analysis attack on elliptic curve cryptosystems". In: *International Workshop on Public Key Cryptography*. Springer. 2003, pp. 199–211.

[11] IBM. *The Dawn of Quantum Computing is Upon Us*. 2016. URL: `https://www.ibm.com/blogs/think/2016/05/the-quantum-age-of-computing-is-here/`.

[12] David Jao and Luca De Feo. "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies". In: *International Workshop on Post-Quantum Cryptography*. Springer. 2011, pp. 19–34.

[13] Marc Joye. “Elliptic curves and side-channel analysis”. In: *ST Journal of System Research* 4.1 (2003), pp. 17–21.

[14] C. Lavor, L. R. U. Manssur, and R. Portugal. *Shor’s Algorithm for Factoring Large Integers*. 2003. eprint: `arXiv:quant-ph/0303175`.

[15] Alfred J. Menezes and Scott A. Vanstone. “A note on cyclic groups, finite fields, and the discrete logarithm problem”. In: *Applicable Algebra in Engineering, Communication and Computing* 3.1 (1992), pp. 67–74. ISSN: 1432-0622. DOI: `10.1007/BF01189025`. URL: `http://dx.doi.org/10.1007/BF01189025`.

[16] Peter L Montgomery. “Speeding the Pollard and elliptic curve methods of factorization”. In: *Mathematics of computation* 48.177 (1987), pp. 243–264.

[17] Katsuyuki Okeya and Kouichi Sakurai. “Power analysis breaks elliptic curve cryptosystems even secure against the timing attack”. In: *International Conference on Cryptology in India*. Springer. 2000, pp. 178–190.

[18] John Proos and Christof Zalka. “Shor’s discrete logarithm quantum algorithm for elliptic curves”. In: (2003). eprint: `arXiv:quant-ph/0301141`.

[19] Peter W. Shor. “Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer”. In: (1995). eprint: `arXiv:quant-ph/9508027`.

[20] Joseph H. Silverman. “Lifting and Elliptic Curve Discrete Logarithms”. In: *Selected Areas in Cryptography: 15th International Workshop, SAC 2008, Sackville, New Brunswick, Canada, August 14-15, Revised Selected Papers*. Ed. by Roberto Maria Avanzi, Liam Keliher, and Francesco Sica. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 82–102. ISBN: 978-3-642-04159-4. DOI: `10.1007/978-3-642-04159-4_6`. URL: `http://dx.doi.org/10.1007/978-3-642-04159-4_6`.

[21] Joseph H. Silverman and Joe Suzuki. “Elliptic Curve Discrete Logarithms and the Index Calculus”. In: *Advances in Cryptology — ASIACRYPT’98: International Conference on the Theory and Application of Cryptology and Information Security Beijing, China, October 18–22, 1998 Proceedings*. Ed. by Kazuo Ohta and Dingyi Pei. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 110–125. ISBN: 978-3-540-49649-6. DOI: `10.1007/3-540-49649-1_10`. URL: `http://dx.doi.org/10.1007/3-540-49649-1_10`.

[22] Nicolas Thériault. “Index Calculus Attack for Hyperelliptic Curves of Small Genus”. In: *Advances in Cryptology - ASIACRYPT 2003: 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, November 30 – December 4, 2003. Proceedings*. Ed. by Chi-Sung Laih. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 75–92. ISBN: 978-3-540-40061-5. DOI: `10.1007/978-3-540-40061-5_5`. URL: `http://dx.doi.org/10.1007/978-3-540-40061-5_5`.

[23] R. Vakil. *Introduction to algebraic geometry, class 13*. Lecture notes for course 725 at Massachusetts Institute of Technology fall 1999. 1999. URL: `https://math.stanford.edu/~vakil/725/class13.pdf`.

[24] Colin D Walter. “Longer keys may facilitate side channel attacks”. In: *International Workshop on Selected Areas in Cryptography*. Springer. 2003, pp. 42–57.

[25] Yuval Yarom and Naomi Benger. “Recovering OpenSSL ECDSA Nonces Using the FLUSH+ RELOAD Cache Side-channel Attack.” In: *IACR Cryptology ePrint Archive* 2014 (2014), p. 140.

[26] Yuval Yarom and Katrina Falkner. “FLUSH+RELOAD: A High Resolution, Low Noise, L3 Cache Side-Channel Attack”. In: *23rd USENIX Security Symposium (USENIX Security 14)*. San Diego, CA: USENIX Association, Aug. 2014, pp. 719–732. ISBN: 978-1-931971-15-7. URL: `https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/yarom`.