evolutionQ

A resource estimation framework for quantum attacks against cryptographic functions

GRI quantum risk assessment report Feb. 2017 - Aug. 2017

Vlad Gheorghiu and Michele Mosca

evolutionQ Inc., Waterloo ON, Canada

31 August 2017

Abstract. We analyze the security of asymmetric-key cryptographic primitives against an attack from a full-scale fault-tolerant quantum computer. In particular, we consider RSA and several discrete logarithm schemes based on elliptic curves, and provide benchmark estimates of the resources needed to break these schemes.

1 Introduction

Public-key cryptography, or asymmetric cryptography, is the branch of cryptography in which secrecy or authentication is achieved via using two types of keys: a *private key*, which is only known to the party that wishes to receive encrypted messages or to authenticate her/him self, and a *public key*, which is publicly available. Whereas symmetric key cryptography (the branch of cryptography in which the parties share a common secret key) is 4000 years old, public-key cryptography, on the other hand, is relatively new. The first academic proposal for public-key encryption scheme dates back to 1976, when Whitfield Diffie, Martin Hellman and Ralph Merkle [1,2] proposed the first public-key cryptographic scheme[1].

The basic idea of public-key cryptography is relatively simple. Any participant has the ability of generating efficiently (computationally easy) a pair of private/public keys. The private key is related to the public key and vice-versa. However, the scheme is designed such that recovering the private key from the public key alone is requires solving a computationally mathematical problem believed to be intractable (technically, the running time of any algorithm that

[1] In 1997, after the declassification of a series of British government documents, it was revealed that James Ellis, Clifford Cocks and Graham Williamson from the United Kingdom's Government Communications Headquarters (GCHQ) discovered the basics of public-key cryptography 4 years earlier, in 1972, although it is unclear if the government realized the profound implication such schemes would have.

solves the problem is believed to be super-polynomial or exponential). When some participant, Alice, wants to send an encrypted message[2] to some other participant, Bob, Alice must use Bob's public key to encrypt the information. To decrypt, Bob uses his private key (which is related to his public key). For authentication (digital signature) Alice is "signing" a message with her private key, then the rest of the world can verify using her public key that indeed Alice (and no one else, since that would imply that an adversary has solved a computation problem believed to be intractable) could have potentially signed the message.

The two most important types of computationally hard problems used in public-key cryptography are based on i) factoring large numbers, and ii) solving the discrete logarithm problem in a large Abelian group[3]. The hardness of factoring constitutes the basics of the RSA public-key encryption scheme [3], whereas the hardness of solving the discrete log problem is the foundation of several cryptographic schemes such as the Diffie-Helman key exchange [1] and variants of it such as Elliptic Curve cryptography (ECC) [4].

Quantum computing represents an entirely new model of computation, which harnesses the fundamental laws of quantum mechanics to perform computations. A number of quantum algorithms promise significant asymptotic speedups compared with their classical counterparts [5,6,7]. While most fields of research will be unaffected by these algorithms until large quantum computers are built, cryptography is affected by the possibility of these algorithms being run at any time in the future. The hardness assumptions underlying the public key cryptosystems currently in use – those related to factoring and variants of the discrete logarithm problem – are violated by quantum adversaries. Quantum Fourier sampling techniques break these cryptosystems in polynomial time [5,8]. As a result these cryptosystems can no longer be considered secure, and ultimately they will have to be replaced. Some standards bodies have already initiated activities toward transitioning to new public key cryptographic primitives [9,10].

In this report we investigate the temporal and spatial resources necessary to attack the RSA scheme and various ECC schemes. Such resource estimates are a central part of estimating when a large scale quantum computer will break ICT systems relying on RSA and ECC, which relate to the urgency of preparing public key primitives designed to resist quantum attacks.

[2] Typically, in practice, the "message" is a random string that will be used in a symmetric key algorithm.

[3] Solving the discrete logarithm problem in a large Abelian group $\mathcal{G}$ reduces to the following. Given an element of a group, call it g, of the form $g = b^k$, where b is another known element of the same group and k is an unknown integer, the problem is to find k.

2 Brief description of RSA and ECC

2.1 The RSA scheme

We briefly describe below the RSA encryption scheme, namely depict the Key Generation algorithm, the Encryption algorithm and the Decryption algorithm. For more details the interested reader can consult [4].

We start with the key generation algorithm.

RSA Key Generation

1. Choose at random two large prime numbers p and q.
2. Compute $N = pq$.
3. Compute the Euler function $\Phi(N) := (p-1)(q-1)$.
4. Choose $e \in \{1, 2, \ldots, \Phi(N) - 1\}$ such that
$$\gcd(e, \Phi(n)) = 1.$$
5. Compute the private key
$$k_{pr} = e^{-1} \bmod \Phi(N).$$
6. The public key is the tuple
$$k_{pub} = (N, e).$$

Next we describe the Encryption algorithm.

RSA Encryption

Given the public key $k_{pub} = (N, e)$ and the plaintext x, the encrypted version is

$$\mathrm{Enc}(x) := x^e \bmod N.$$

Here we assume that $x \in \mathbb{Z}_N := \{0, 1, \ldots N-1\}$, the ring of integers modulo N.

Finally we present the Decryption algorithm.

RSA Decryption

Given the private key k_{priv} and a ciphertext $y = Enc(x)$ encrypted with the corresponding public key k_{pub}, the decryption of y is obtained by

$$\mathrm{Dec}(y) := y^{k_{priv}} \bmod N.$$

Commonly the RSA scheme is often abbreviated as RSA-n, where $n = \log_2(N)$ is the size in bits of N, e.g. RSA-1024, RSA-2048 etc.

2.2 The ECC scheme

As mentioned before, the ECC public-key system is based on the hardness of solving the discrete logarithm problem over an Abelian group $\mathcal{G}$, the latter generated geometrically from an elliptic curve as we describe in the following.

An elliptic curve **E** over the real numbers is the collection of points

$$\mathbf{E} = \left\{(x, y) | y = x^3 + ax + b \text{ with } a, b \in \mathbb{R}, 4a^3 + 27b^2 \neq 0\right\}. \tag{1}$$

The condition $4a^3 + 27b^2 \neq 0$ guarantees that the curve is non-singular, i.e. that the equation $x^3 + ax + b$ has no repeated roots.

Point addition

It can be shown that any line passing through two arbitrary points $P, Q \in \mathbf{E}$ intersects **E** once and only once more in a point $R' \in \mathbf{E}$. The result of the addition operation $P + Q$ is then defined as the point R obtained from the reflection of R' about the x axis, see Fig. 1.

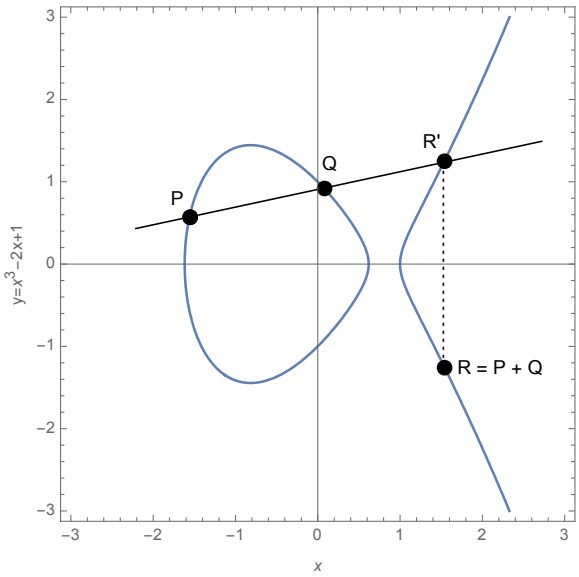

Fig. 1. Adding two different points P and Q on the elliptic curve $y = x^3 - 2x + 1$.

Point doubling

If P and Q coincide, then the addition operation $P + P = 2P$ (often called *point doubling*) is defined by drawing a line tangent to P which intersects **E** at R', followed by mirroring R' about the x axis to obtain $R = P + P = 2P$, see Fig. 2.

Associated group

It now follows that the collection of points on an elliptic curve **E** together with a special point $\mathcal{O}$ chosen as the neutral element, called the *point at infinity*, form an Abelian group $\mathcal{G}$. Hence every elliptic curve **E** uniquely determine an associated Abelian group $\mathcal{G}$, often called the *elliptic-curve group of* **E**.

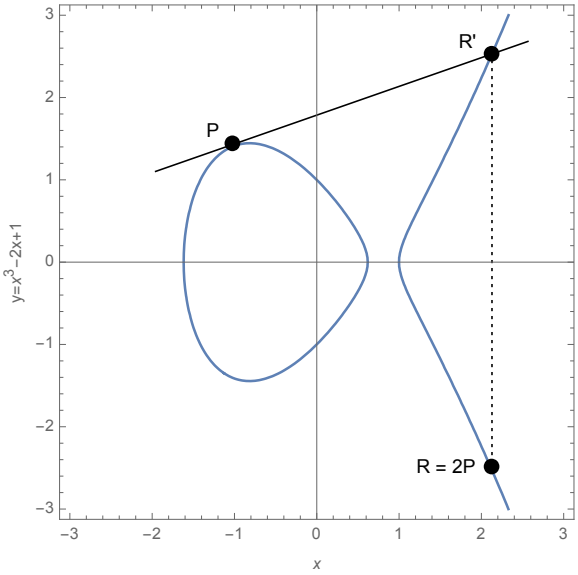

Fig. 2. Point doubling $2P$ on the elliptic curve $y = x^3 - 2x + 1$.

Elliptic curve point addition and multiplication over $\mathbb{Z}_p$
For cryptographic applications, the elliptic curve $\mathbf{E}$ is defined over some prime[4] finite field $\mathbb{Z}_p$ (instead of real numbers), where p is a prime, and the associated elliptic-curve group is

$$\mathcal{G} = \{(x, y)|y = x^3 + ax + b \bmod p, \text{ with } a, b \in \mathbb{Z}_p, 4a^3 + 27b^2 \neq 0 \bmod p\} \cup \{\mathcal{O}\}. \tag{2}$$

Given two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ on $\mathbf{E}$, then the result of the addition $P + Q$ is the point $R = P + Q = (x_3, y_3)$ with coordinates

$$\begin{aligned} x_3 &:= m^2 - x_1 - x_2 \bmod p \\ y_3 &:= m(x_1 - x_3) - y_1 \bmod p, \end{aligned} \tag{3}$$

where $m := (y_2 - y_1)(x_2 - x_1)^{-1} \bmod p$ for $P \neq Q$ (point addition) or $m := (3x_1^2 + a)(2y_1)^{-1} \bmod p$ for $P = Q$ (point doubling).
Hasse's bound and the security of ECC over prime finite fields
The size $|\mathcal{G}|$ of the associated group $\mathcal{G}$ determines the security of the ECC scheme over the prime field $\mathbb{Z}_p$, quantitatively given by Hasse's bound

$$p + 1 - 2\sqrt{p} \leqslant |\mathcal{G}| \leqslant p + 1 + 2\sqrt{p}. \tag{4}$$

The security of ECC schemes is based on the hardness of finding discrete logarithms in the elliptic-curve group $\mathcal{G}$, namely given a point $R = rP = P +$

[4] Elliptic curves can be constructed over arbitrary finite fields, in particular Galois fields $GF(2^m)$, but non-prime fields introduce additional complications, which, for the sake of simplicity, we avoid by restricting our analysis to prime fields.

$P + \ldots + P$ (r times), with $r \leqslant |\mathcal{G}|$, finding r is assumed to be a computationally hard problem.

ECC Key exchange

The key exchange algorithm over elliptic curves is a typical Diffie-Hellman scheme where the Abelian group is the associated elliptic-curve $\mathcal{G}$. Schematically, Alice and Bob established a shared secret key in the following way.

1. Both Alice and Bob publicly agree in advance on an elliptic curve $\mathbf{E}$ and a point $P \in \mathbf{E}$.
2. Alice starts by randomly choosing her private key $k^A_{priv} = a \in \{2, 3, \ldots, |\mathcal{G}|\}$.
3. Alice computes aP and announces the result (publicly) to Bob.
4. Bob randomly chooses his private key $k^B_{priv} = b \in \{2, 3, \ldots, |\mathcal{G}|\}$.
5. Bob computes bP and announces the result (publicly) to Alice.
6. Alice computes $a(bP) = abP$, and Bob computes $b(aP) = abP$ (the last equality holds because the group $\mathcal{G}$ is Abelian), i.e. Alice and Bob established a shared secret key abP.

Elliptic curves are also frequently used for digital signatures, e.g. in the standardized ECDSA algorithm (Elliptic Curve Digital Signature Algorithm). The main advantage of ECC over RSA is the key length: an ECC scheme over a prime field of size 2^{160} offers roughly the same security as an RSA-1024 scheme, whereas an ECC scheme over a prime field of size 2^{224} offers roughly the same security as an RSA-2048 scheme.

For more details about ECC the interested reader can consult e.g. [4].

3 Attacking RSA and ECC with a quantum computer

As shown by Peter Shor in 1994[5], both RSA and ECC schemes are vulnerable against a quantum adversary, e.g. both can be broken in polynomial time on a quantum computer. Shor's factoring algorithm is implemented at the logical layer by the quantum circuit schematically depicted in Fig. 3 A slight variation of the factoring circuit can be used to break the discrete logarithm over Abelian groups, as depicted in Fig. 4.

4 Methodology and results

We use the same analysis framework described in our Sep. 2016 - Feb. 2017 GRI report [11] and depicted schematically in Fig. 5. We assume a surface-code based fault-tolerant quantum adversary. Our cost metric is based on several assumptions, described in detail in Sec. 2.2 of [11] and summarized for completeness below.

Assumption 1 *The resources required for any large quantum computation are well approximated by the resources required for that computation on a surface code based quantum computer.*

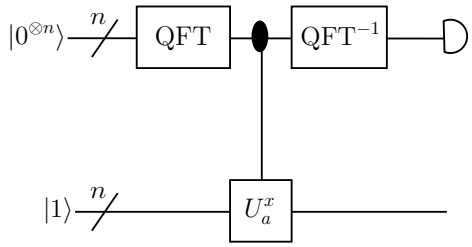

Fig. 3. Generic Shor's factoring algorithm, logical layer. The integer a is co-prime to N, i.e. $\gcd(a, N)$=1, where $N = 2^n$ is the closest power of two (from above) to the number to be factored. The modular exponentiation part of the circuit is depicted schematically by the controlled-U_a^x gate. The action of the latter on a computational basis state $|s\rangle$ is defined $U_a^x |s\rangle = U_{a^x} |s\rangle = |sa^x \bmod N\rangle$. The QFT/QFT^{-1} box represents the quantum Fourier transform and its inverse, respectively, on n qubits. The result is obtained by measuring the top n qubits and post-processing the measurement data.

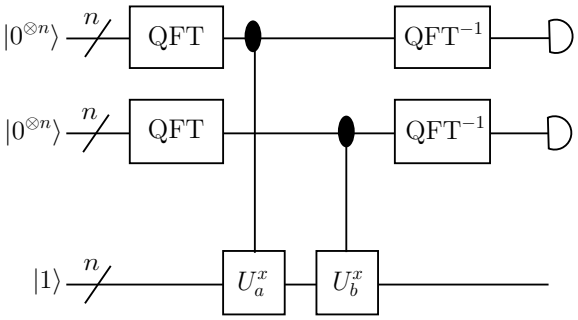

Fig. 4. Generic adaptation of Shor's algorithm for breaking the discrete logarithm problem over Abelian groups, logical layer. Given the publicly known fixed point $P \in \mathbf{E}$ used in the ECC scheme, and given Q guaranteed to be of the form $Q = kP$ for some integer k, the circuit computes the discrete logarithm of Q in $\mathbf{E}$, i.e. finds k. The modular exponentiation part of the circuit is depicted schematically by the controlled-U_a^x (U_b^x) gate(s). The action of the latter on a computational basis state $|s\rangle$ is defined $U_a^x |s\rangle = U_{a^x} |s\rangle = |sa^x \bmod N\rangle$. Here $N = 2^n$ is the closest power of two (from above) to the size of the cyclic group generated by the point P. The integers a and b represent the binary encodings of the points P and Q, respectively. The QFT/QFT^{-1} box represents the quantum Fourier transform and its inverse, respectively, on n qubits. The result is obtained by measuring the top $2n$ qubits and post-processing the measurement data.

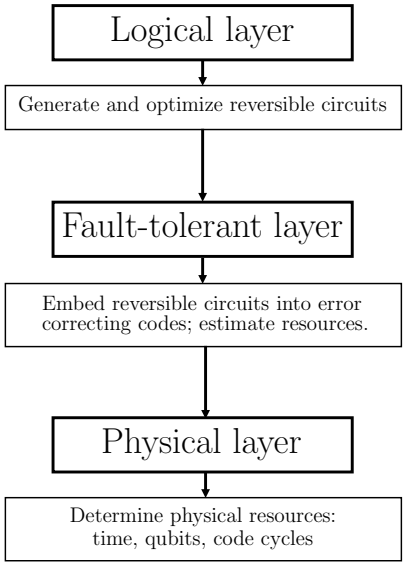

Fig. 5. Analyzing an attack against a asymmetric cryptographic function with a fault-tolerant quantum adversary.

Assumption 2 *The classical error correction routine for the surface code on an $L \times L$ grid of logical qubits requires an $L \times L$ mesh of classical processors (i.e. $C_a = n$).*

Assumption 3 *Each ASIC performs a constant number of operations per surface code cycle.*

Assumption 4 *The temporal cost of one surface code cycle is our fundamental unit of time.*

Combining Assumptions 1, 2, and 4 we arrive at the following metric for comparing the costs of classical and quantum computations.

Cost Metric 1 *The cost of a quantum computation involving ℓ logical qubits for a duration of σ surface code cycles is equal to $\ell \cdot \sigma$.*

In contrast to our previous analysis on hash function in [11] where we analyzed brute-force super-polynomial time attacks using Grover searching, Shor's quantum algorithm for breaking RSA, and its adaptation for breaking ECC, has polynomial running time on a quantum computer, and Clifford gates make up a negligible fraction of the algorithmic cost. In this case, we can safely ignore the cost of Clifford operations, as they are negligible in contrast to the cost of implementing the non-Clifford T gate fault-tolerantly (see [11] for more details). Therefore, in all of our estimations below we will only consider the cost associated to the T gates in the circuit.

In all of our analysis summarized below, we optimize for space and wall-time, i.e. we estimate the minimum time (wall-time) needed to break the listed cryptographic primitives for a given small sized quantum computer (in terms of physical qubits). We consider a physical error rate per gate $p_g = 10^{-4}$, considered optimistic with today's technology, and optimize over state-of-the-art surface code configurations to derive the optimized physical cost.

4.1 RSA results

We estimate the resources needed to attack the following public-key schemes based on factoring large numbers: RSA-768, RSA-1024, RSA-2048 and RSA-4096. We did a literature search and used the best (as of today) logical circuits for factoring with a quantum computer, described in [12]. We summarize all our findings in the tables below. For completeness, for each cryptographic primitive, we also mention its corresponding classical security parameter (bits of security).

Classical security parameter (bits)	64
RSA-768 cost estimates	
T-count	1.27×10^{11}
T-depth	4.23×10^{10}
Logical qubits	2290
Physical qubits	1.92×10^{6}
Total wall-time	1.51 hours

Table 1. Benchmark cost of attacking RSA-768 with a fully-scalable fault-tolerant quantum adversary.

Classical security parameter (bits)	80
RSA-1024 cost estimates	
T-count	3.00×10^{11}
T-depth	1.00×10^{11}
Logical qubits	2290
Physical qubits	2.56×10^{6}
Total wall-time	3.58 hours

Table 2. Benchmark cost of attacking RSA-1024 with a fully-scalable fault-tolerant quantum adversary.

4.2 ECC results

We estimate the resources needed to attack the following discrete-log based NIST standardized ECC schemes: P-160, P-192, P-256 and P-521. Based on an exten-

Classical security parameter (bits)	112
RSA-2048 cost estimates	
T-count	2.48×10^{12}
T-depth	8.02×10^{11}
Logical qubits	4338
Physical qubits	6.2×10^{6}
Total wall-time	28.63 hours

Table 3. Benchmark cost of attacking RSA-2048 with a fully-scalable fault-tolerant quantum adversary.

Classical security parameter (bits)	128
RSA-4096 cost estimates	
T-count	1.92×10^{13}
T-depth	6.21×10^{12}
Logical qubits	8434
Physical qubits	1.47×10^{7}
Total wall-time	229 hours

Table 4. Benchmark cost of attacking RSA-4096 with a fully-scalable fault-tolerant quantum adversary.

sive literature search, and our own analysis, we chose to benchmark the optimized logical circuits from [13]. We summarize all our findings in the tables below.

Classical security parameter (bits)	80
NIST ECC P-160 cost estimates	
T-count	2.97×10^{11}
T-depth	6.93×10^{10}
Logical qubits	1946
Physical qubits	1.83×10^{6}
Total wall-time	2.48 hours

Table 5. Benchmark cost of attacking the standardized NIST P-160 elliptic curve with a fully-scalable fault-tolerant quantum adversary.

5 Conclusions and future directions

We estimated the vulnerabilities of typical public-key encryption schemes used heavily in today's secure communication landscape, namely RSA and ECC, against full scale fault-tolerant quantum computers. Our benchmark estimates are based on several assumptions, the most important one being the assumption that fault-tolerance will be achieved with surface codes. This is a reasonable

Classical security parameter (bits)	96
NIST ECC P-192 cost estimates	
T-count	3.71×10^{11}
T-depth	1.24×10^{11}
Logical qubits	1994
Physical qubits	2.42×10^{6}
Total wall-time	4.42 hours

Table 6. Benchmark cost of attacking the standardized NIST P-192 elliptic curve with a fully-scalable fault-tolerant quantum adversary.

Classical security parameter (bits)	128
NIST ECC P-256 cost estimates	
T-count	8.82×10^{11}
T-depth	2.94×10^{11}
Logical qubits	2330
Physical qubits	3.21×10^{6}
Total wall-time	10.5 hours

Table 7. Benchmark cost of attacking the standardized NIST P-256 elliptic curve with a fully-scalable fault-tolerant quantum adversary.

Classical security parameter (bits)	260
NIST ECC P-521 cost estimates	
T-count	7.98×10^{12}
T-depth	2.66×10^{12}
Logical qubits	4959
Physical qubits	7.81×10^{6}
Total wall-time	95 hours

Table 8. Benchmark cost of attacking the standardized NIST P-521 elliptic curve with a fully-scalable fault-tolerant quantum adversary.

assumption today, as the surface code (or variants of it such as color codes or 3D topological codes) still remain the most promising candidate for scalable quantum computing.

We also observe that for the security strength parameter values used in practice the current benchmark estimates indicate only a modest difference in security against quantum attacks for RSA and ECC.

Our next steps will include improving over the benchmark estimates (i.e. improving the algorithms, circuits, and fault-tolerance overheads) and investigate additional cryptographic algorithms in use, both symmetric (private-key) and asymmetric (public-key). We will also investigate how various space/time trade-offs affect the wall-time.

References

1. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Transactions on Information Theory 22(6), 644–654 (Nov 1976)
2. Merkle, R.C.: Secure communications over insecure channels. Commun. ACM 21(4), 294–299 (Apr 1978), http://doi.acm.org/10.1145/359460.359473
3. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM 21(2), 120–126 (Feb 1978), http://doi.acm.org/10.1145/359340.359342
4. Katz, J., Lindell, Y.: Introduction to Modern Cryptography (Chapman & Hall/Crc Cryptography and Network Security Series). Chapman & Hall/CRC (2007)
5. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Journal on Computing 26(5), 1484–1509 (1997), http://link.aip.org/link/?SMJ/26/1484/1
6. Grover, L.K.: Quantum mechanics helps in searching for a needle in a haystack. Phys. Rev. Lett. 79, 325–328 (Jul 1997), http://link.aps.org/doi/10.1103/PhysRevLett.79.325
7. Jordan, S.: Quantum Algorithm Zoo (February 2016), http://math.nist.gov/quantum/zoo/
8. Boneh, D., Lipton, R.J.: Quantum Cryptanalysis of Hidden Linear Functions. In: Coppersmith, D. (ed.) Advances in Cryptology - CRYPTO'95, pp. 424–437. No. 963 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (Aug 1995)
9. Agency, U.S.N.S.: NSA Suite B Cryptography - NSA/CSS. NSA website (August 2015), https://www.nsa.gov/ia/programs/suiteb_cryptography/
10. Chen, L., Jordan, S., Liu, Y.K., Moody, D., Peralta, R., Perlner, R., Smith-Tone, D.: Report on post-quantum cryptography. National Institute of Standards and Technology Internal Report 8105 (February 2016)
11. Gheorghiu, V., Mosca, M.: A resource estimation framework for quantum attacks against cryptographic functions (2017), Global Risk Institute quantum risk assessment report, Sep. 2016 - Feb. 2017
12. Fowler, A.G., Mariantoni, M., Martinis, J.M., Cleland, A.N.: Surface codes: Towards practical large-scale quantum computation. Phys. Rev. A 86, 032324 (Sep 2012), http://link.aps.org/doi/10.1103/PhysRevA.86.032324
13. Roetteler, M., Naehrig, M., Svore, K.M., Lauter, K.: Quantum resource estimates for computing elliptic curves discrete logarithms (2017), arXiv:1706.06752 [quant-ph]