Power Analysis Attacks against QUAD

Weijian Li, Fuxiang Lu, and Huimin Zhao

***Abstract*—QUAD is a stream cipher whose provable security relies on the hardness of solving systems of multivariate quadratic equations (the MQ problem). In addition to resistance to quantum attacks and low cost, MQ-based cryptographic algorithms are believed to have strong natural resistance to side-channel attacks, because of their long key length and the absence of leaking operations. However, our research has found that serial implementations of QUAD leak secret information when computing monomials and restoring the results to the register, which leaves these implementations vulnerable to side-channel attack. In this article, we define single-bit and multi-bit side-channel leakage models appropriate for serial implementations of QUAD, and employ them to successfully perform correlation power analysis attacks. A comparison with reported cryptanalysis results for $QUAD(2, 160, 160)$ indicates that our method is the most efficient. Finally, defensive countermeasures against our attacks are proposed.**

***Index Terms*—Post-quantum cryptosystem, MQ problem, QUAD, Side-Channel Attacks.**

I. Introduction

The MQ problem, which consists of finding a solution to a multivariate quadratic system of m quadratic equations in n variables over a finite field $GF(q)$, is known to be NP-complete [1], even over a finite field $GF(2)$. In contrast to number theoretic problems such as factorization and the discrete logarithm problem, no efficient quantum algorithm is known to solve the MQ problem within polynomial time. Generic attacks on the MQ problem using the Gröbner basis commonly involve exponential complexity in time and space [2]. Therefore, under the threat of attacks by future quantum computers, cryptosystems based on the MQ problem are regarded as a possible alternative to number theoretic-based cryptosystems.

Since the first cryptosystem proposed by Matsumoto and Imai [3] in 1988, significant efforts have been made to construct cryptographic primitives based on the MQ problem. In asymmetric cryptography, which is also known as multivariate public-key cryptography (MPKC) [4], numerous public-key schemes have been proposed, such as SFLASH, UOV, HFE [5], and Rainbow [6]. In addition, a public-key identification scheme based on multivariate quadratic polynomials was recently proposed by Sakumoto et al. [7] in 2011. From the perspective of symmetric cryptography, Berbain et al. [8] proposed a stream cipher denoted as QUAD in 2006, whose provable security was based on the hardness of solving the MQ problem.

Moreover, cryptographic primitives based on the MQ problem are in general much more computationally efficient than number theoretic-based schemes. This efficiency supports the use of many cryptographic schemes with ubiquitous computing devices. The Internet of Things (IoT) is a novel paradigm that is rapidly gaining increasing interest in the information technology field. The IoT is essentially a network of pervasive devices that are able to share information and cooperate with neighboring devices to attain common goals through unique addressing schemes. Increasingly, everyday items are converted to pervasive devices by embedding computing power, resulting in a variety of devices such as radio-frequency identification (RFID) tags, sensors, application specific integrated circuits (ASICs), and smart cards. However, this embedded computing power introduces rigid cost constraints in terms of area, memory, computing power, and battery supply, which necessitates the use of algorithms with the highest levels of efficiency. Although the mass deployment of pervasive devices promises many benefits, security and privacy remain crucial issues, particularly for applications that are highly security and privacy sensitive (e.g., military and financial applications). Therefore, lightweight cryptography algorithms and protocols have been specifically developed to serve as security components in such applications.

However, the physical implementations of cryptosystems are vulnerable to side-channel attacks [9], and must be protected from such attacks prior to their implementation [10]. Although side-channel attacks have been developed over the past 20 years, to our best knowledge, few such attacks have been applied against cryptosystems based on the MQ problem. Steinwandt et al. [11] utilized XOR operations in a theoretical cryptanalysis to reveal the secret parameters Δ, s, and t of the SFLASH signature scheme. Okeya et al. [12] proposed an attack against addition operation modulo 2^{32} to reveal Δ of SFLASH implemented on an integrated circuit (IC) chip. Hashimoto et al. [13] proposed a fault attack on MPKC systems to change the coefficients of the central map.

Arditti et al. [14] demonstrated compact serial implementations of QUAD that were suited to lightweight devices with highly limited computation capabilities such as RFID tags. However, our research indicates that such implementations of QUAD leak secret information when computing monomials and restoring the results to the register, and an adversary could compromise multivariate cryptographic algorithms by taking advantage of this side-channel leakage. To demonstrate the extent of this vulnerability, the present work defines single-bit and multi-bit side-channel leakage models for serial implementations of QUAD, and employ

Manuscript received January 29, 2018; revised June 28, 2018. This work was supported by National Natural Science Foundation of China (no. 61872096), Guangdong Provincial Project of Science and Technology (no.2016A010101030), National Natural Science Foundation of China (no. 61672008), Guangdong Provincial Application-oriented Technical Research and Development Special fund project (no. 2016B010127006), and Scientific and Technological Projects of Guangdong Province (no. 2017A050501039).

Weijian Li is with the School of Computer Science, Guangdong Polytechnic Normal University, Guangzhou, 510665 China e-mail: (weijianlee@126.com).

Fuxiang Lu is with the School of Computer Science, Guangdong Polytechnic Normal University, Guangzhou, 510665 China e-mail: (294503349@qq.com).

Huimin Zhao is with the School of Computer Science, Guangdong Polytechnic Normal University, Guangzhou, 510665 China e-mail: (zhaohuimin@gpnu.edu.cn).

these models to successfully perform single-bit and multi-bit power analysis attacks against a field-programmable gate array (FPGA) serial implementation of QUAD.

The remainder of this paper is organized as follows. In section 2, we review the mathematical definition and serial FPGA implementation of the QUAD stream cipher. In Section 3, the differential power analysis security of the serial FPGA implementation is examined via the conduct of power analysis attacks, experimental results and complexity of our attacks are given. A brief defensive countermeasure against our attacks is proposed in Section 4. Section 5 concludes the paper.

II. Preliminaries

A. Mathematical definition of QUAD

Each multivariate quadratic equation is a polynomial of degree of at most 2 with n variables over a field $GF(q)[x_1, \cdots, x_n]$, which can be defined as

$$Q(x) = \sum_{1 \le i \le j \le n} \alpha_{ij} x_i x_j + \sum_{1 \le i \le n} \beta_i x_i + \gamma \qquad (1)$$

Here, coefficients α_{ij}, β_i, and γ are all over $GF(q)$. Note that the monomial forms $x_i x_i$ and x_i are equal in the particular case of $q = 2$. A multivariate quadratic system S consists of a set of m quadratic polynomials $(Q_1, \cdots, Q_m)$ in n variables over $GF(q)$. The MQ problem is defined as, given $S = (Q_1, \cdots, Q_m)$, find a value $x \in GF(q)^n$, if any, such that $Q_l(x) = 0$ for all $1 \le l \le m$ [8].

A particular QUAD stream cipher in n variables over $GF(q)$ can be specified as $QUAD(q, n, r)$, which produces r outputs per round [15], and includes an output function P: $GF(q)^n \to GF(q)^r$ consisting of r quadratic polynomials $P_1, P_2, \cdots, P_r$ in n variables, and an update function Q: $GF(q)^n \to GF(q)^n$ consisting of n quadratic polynomials $Q_1, Q_2, \cdots, Q_n$ in n variables. The parameters q, n, and r , and coefficients α_{ij}, β_i, and γ for P and Q are public. Denote the n-bit internal state by $X = (x_1, \cdots, x_n)$. The QUAD cipher expands a secret initial state $X_0 \in GF(q)^n$ into a sequence of secret states $X_0, X_1, X_2, \cdots \in GF(q)^n$ and a sequence of output vectors $Y_0, Y_1, Y_2, \cdots \in GF(q)^r$ as follows.

$$\begin{array}{ccccc} X_0 \to & X_1 = Q(X_0) \to & X_2 = Q(X_1) \to & \cdots \\ \downarrow & \downarrow & \downarrow & \\ Y_0 = P(X_0) & Y_1 = P(X_1) & Y_2 = P(X_2) & \cdots \end{array} \qquad (2)$$

Typically, q is a power of 2, allowing each output vector $y_i \in GF(q)^r$ to encrypt the next r bits of plaintext in a straightforward manner.

B. FPGA implementation of QUAD

The smallest compact implementation of QUAD introduced by Arditti et al. [14] is not only the smallest provably secure stream cipher, but is also a very good competitor among conventional stream ciphers. To achieve implementations with as small a size as possible, these researchers first focused on the Boolean setting $GF(q) = GF(2)$, over which each Q can be rewritten as

$$Q(x) = \sum_{1 \le i \le j \le n} \alpha_{ij} x_i x_j + \gamma \qquad (3)$$

because the monomial forms $x_i x_i$ and x_i are equal over $GF(2)$. Moreover, because α_{ij} and γ for P and Q are public and randomly generated, the need for large memory capacity is transformed into very small generation circuitry. During encryption, computations of each Q are performed sequentially. Each new monomial is computed at every clock tick and its contribution is accumulated to a temporary register for the output polynomial being computed.

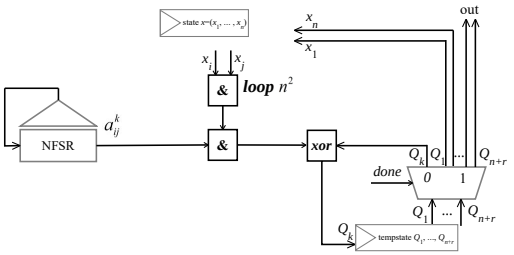


Fig. 1. Serial FPGA implementation of $QUAD(2, n, r)$ employing a nonlinear feedback shift register (NFSR) [14].

As shown in Fig. 1, the FPGA serial implementation of $QUAD(2, n, r)$ includes two main components. The first component is a nonlinear feedback shift register (NFSR), which generates the coefficients of each polynomial cycle by cycle. The second component simultaneously computes the value of the corresponding monomial. Their combination (a bit product) is accumulated to the temporary register $Q_1, Q_2, \cdots, Q_{n+r}$. The process flow is described as follows.

1. The implementation computes polynomial $Q_k(X), 1 \le k \le n + r$ sequentially.

2. At every clock tick, the NFSR generates coefficient α_{ij}, a new monomial $\alpha_{ij} x_i x_j$ of polynomial $Q_k(X)$ is computed, and its contribution is accumulated to the temporary register Q_k for the output polynomial $Q_k(X)$ being computed.

3. After $n(n+1)/2+1$ clock cycles, polynomial $Q_k(X)$ is computed, and the above process is repeated for $Q_{k+1}(X)$.

4. Once all $n + r$ polynomials are computed, r values are output as the keystream, and the other n values are used to update the internal state.

C. Power analysis attacks

A power analysis attack extracts the secret keys of a cryptographic algorithm based on the analysis of a large number of power traces obtained from cryptographic hardware devices while encrypting different plaintexts employing the same key. As described in Fig. 2, a general attack strategy is comprised of five steps [9].

Step 1: Choose an intermediate result of the algorithm and a power leakage model (usually the Hamming weight leakage model for software implementation, and the Hamming distance leakage model for hardware implementation). This intermediate result is denoted as selection function $D(C, k)$, where C is a known non-constant data value (usually part of the plaintext or cipher) and k is a small part of the key.

Step 2: Measure the power consumption $t_i (1 \le i \le N)$ of the cryptographic hardware device while it encrypts N different plaintexts $p_1, \cdots p_N$ with the same key. Denote C corresponding to the i-th plaintext or cipher as C_i.

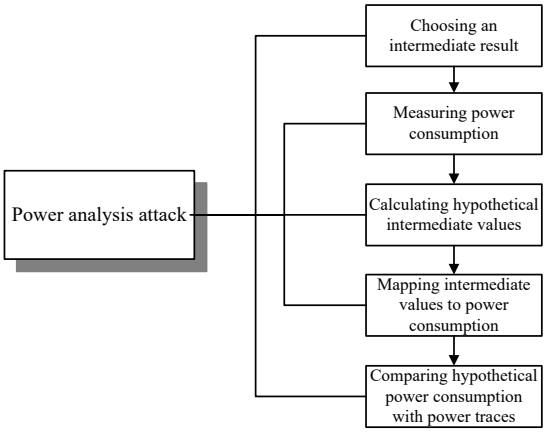


Fig. 2. Schematic of a general power analysis attack.

Step 3: Calculate hypothetical intermediate values $d_{i,s} = D(C_i, k_s), 1 \leq i \leq N$ for every possible choice k_s of the key.

Step 4: Map hypothetical intermediate values $d_{i,s}$ to the hypothetical power consumption $h_{i,s}$ using the appropriate power leakage model selected in Step 1.

Step 5: Compare the hypothetical power consumption with the actual power traces using statistical methods such as DPA (Eq. (4)) [16], CPA (Eq. (5)) [17] or MIA(Eq. (6)) [18] to reveal the secret key.

$$\begin{cases} G_{0,s} = \{t_i, i = 1, 2, \cdots, N | h_{i,s} < h/2\} \\ G_{1,s} = \{t_i, i = 1, 2, \cdots, N | h_{i,s} \geq h/2\} \\ \Delta_s = \frac{\sum_{G_{1,s}} t_i}{|G_{1,s}|} - \frac{\sum_{G_{0,s}} t_i}{|G_{0,s}|} \end{cases} \tag{4}$$

$$\begin{aligned} \Delta_s = \sigma_T H_s &= \frac{\mathrm{cov}(T, H_s)}{\sigma_T \sigma_{H_s}} = \frac{E(TH_s) - E(T) \cdot E(H_s)}{\sigma_T \sigma_{H_s}} \\ &= \frac{\sum_{i=0}^{n-1} (t_i - E(T))(h_{i,s} - E(H_s))}{\sqrt{\sum_{i=0}^{n-1} (t_i - E(T))^2} \sqrt{\sum_{i=0}^{n-1} (h_{i,s} - E(H_s))^2}} \end{aligned} \tag{5}$$

$$\begin{aligned} \Delta_s = \sum \Pr[T_{k_d} = t_{k_{d,p}} | H_{k_s} = h_{k_{s,p}}] \cdot \Pr[H_{k_s} = h_{k_{s,p}}] \\ \cdot \log(\frac{\Pr[T_{k_d} = t_{k_{d,p}} | H_{k_s} = h_{k_{s,p}}]}{\Pr[T_{k_d} = t_{k_{d,p}}]}) \end{aligned} \tag{6}$$

Here, h is the maximum value of the hypothetical power consumption, $T = \{t_1, \cdots, t_N\}$, and $H_s = \{h_{i,s}, \cdots, h_{N,s}\}$.

In theory, if the key hypothesis k_s is correct, $\Delta_s \neq 0$ at the instant when the intermediate value is handled, which means that the DPA trace will exhibit a peak. Otherwise, Δ_s tends to be 0, and no obvious peak appears.

III. Power analysis attacks against QUAD

A. Side-channel leakage model of QUAD

It is well known that the power consumption of registers in hardware implementations can be described very well by the HD model [9]. Registers Q_k in Fig. 1 are triggered by a clock signal, and change their values only once at each clock cycle. As such, an attacker can estimate the power consumption of a register $Q_k (1 \leq k \leq n + r)$ by calculating the Hamming distance of the values that are stored in consecutive clock cycles.

As shown in Fig. 1, at every clock tick, the serial implementation computes a new monomial $\alpha_{ij} x_i x_j$ of polynomial $Q_k(X)$, and accumulates its contribution to the temporary register Q_k. As a consequence, the value of Q_k changes from Q_k to $Q_k \oplus \alpha_{ij} x_i x_j$. The Hamming distance of Q_k can be expressed as $HD(Q_k, Q_k \oplus \alpha_{ij} x_i x_j) = HW(\alpha_{ij} x_i x_j)$, where $HW(\cdot)$ represents the Hamming weight of the monomial. Our single-bit power analysis attack focus on the computation of $\alpha_{ii} x_i x_i$ for simplicity and efficiency. Meanwhile, because monomials $x_i x_i$ and x_i are equal over $GF(2)$, the single-bit side-channel leakage model of QUAD can be defined as

$$h(Q(x)) = HD(Q_k, Q_k \oplus a_{ii} x_i x_i) = HW(\alpha_{ii} x_i) \tag{7}$$

Transitions $0 \rightarrow 0$ and $1 \rightarrow 1$ in Q_k lead to no excess power consumption, whereas transitions $0 \rightarrow 1$ and $1 \rightarrow 0$ involve excess power consumption. Therefore, a single-bit power analysis attack is utilized to reveal the internal state $X = (x_1, \cdots, x_n)$. Correlation traces will exhibit a positive peak if $x_i = 1$, and a non-positive peak if $x_i = 0$.

Algorithm 1: Single-bit correlation power analysis attack on QUAD

```
Input:
  coff:    array for coefficients α_ii
  traces: power consumption traces
Output:
  X: secret internal state X
/* total number of monomials for each polynomial */
coffLen = n(n+1)/2;
for keybit = 1 : n  /* attack bit by bit */
    /* array index for a_ii x_i x_i */
    coffIndex=coffLen-(n-keybit+1)*(n-keybit+2)/2+1;
    /* hypothetical power consumption */
    hd = coff [coffIndex,:];
    for i = 1 : N_P
        /* Eq.(4), Eq.(5),Eq.(6) and etc. */
        cor(i) = corrcoef(hd, traces[i,:]);
    end
    /* is peak generated?*/
    if max(cor) ≥ Threshold then
        X[keybit] = 1
    else
        X[keybit] = 0
    end
end
return X;
```

B. Single-bit power analysis attack against QUAD

For QUAD, the secret key is its internal state $X = (x_1, \cdots, x_n)$. An adversary begins with every possible key guess x_i and coefficient α_{ii}. The hypothetical power consumption of $HW(\alpha_{ii} x_i)$ is then computed according to the leakage model given by Eq. (7). The correlation coefficient between the hypothetical power consumption and actual power traces is subsequently computed. This is defined in pseudo-code as follows in Algorithm 1.

C. Multi-bit power analysis attack against QUAD

The proposed single-bit power analysis attack described in the previous subsection requires a threshold to determine whether or not the correlation trace exhibits a peak, and an appropriate threshold value is difficult to obtain in practice. Therefore, we define a multi-bit side-channel leakage model, and propose a corresponding multi-bit power analysis attack against QUAD, which is much more practical and efficient than single-bit power analysis attack.

Algorithm 2 : Precomputation for power traces of QUAD

Input:
i_s: first index of internal state X to attack
N_{gb}: length of bits to attack each time
traces: power consumption traces
Output:
newTraces: precomputed power traces
/* total number of monomials for each polynomial */
coffLen = n(n+1)/2+1;
/* start index of traces for the computations of $a_{i_s i_s}$ */
startIndex = coffLen-(n-i_s+1)*(n-i_s+2)/2 + 1;
newTraces[1:N_P, 1:N] = 0;
/* precomputing N_{gb}-bits by N_{gb}-bits */
for Δ_i = 1 : N_{gb}
  /* index of traces for the computations of a_{ii} */
  endIndx=coffLen-(n-i_s-Δ_i+2)*(n-i_s-Δ_i+3)/2+1;
  for Δ_j = 0 : (N_{gb}-Δ_i)
    /* difference value of index in traces for a_{ij} and $a_{i_s i_s}$ */
    pointWidth = (endIndx - startIndex + Δ_j) * pointsPerCycle;
    newTraces[:, 1:(N_P-pointWidth)] = newTraces[:, 1:(N_P-pointWidth)] + traces[:, (pointWidth+1):N_P];
  end
end

Algorithm 3: Multi-bit correlation power analysis attack on QUAD

Input:
coff: array for coefficients α_{ij}
traces: power consumption traces
N_{gb}: length of bits to attack each time
Output:
X: secret internal state X
for keyNum = 1 : n/N_{gb}
  $i_s = (keyNum - 1) * N_{gb} + 1$;
  /* Algorithm 2 */
  newTraces = Precomputation(i_s, N_{gb}, traces);
  for key guess $x_s = 0 : (2^{N_{gb}} - 1)$
    /* Eq. (9) */
    hd = calMultibitLeakage(i_s, N_{gb}, coff, x_s);
    for $i = 1 : N_P$
      /* Eq. (5) */
      cor[x_s,i] = corrcoef(hd, newTraces[i,:]);
    end
  end
  /* key guess corresponding to the peak of CPA trace is the correct key */
  X[i_s : (i_s+N_{gb}-1)] = indexofmax(cor);
end
return X;

For a subkey of length N_{gb}, multi-bit power analysis attack takes into consideration the following monomials:

$$Q\prime(x) = \sum_{i_s \le i \le j \le (i_s + N_{gb} - 1)} \alpha_{ij} x_i x_j \tag{8}$$

which are sequentially computed in $N_{gb}(N_{gb}+1)/2$ cycles, where i_s is the starting index of internal state X. The cumulative power consumption of operations in Eq. (8) can be described by the following multi-bit side-channel leakage model:

$$h(Q\prime(x)) = \sum_{i_s \le i \le j \le (i_s + N_{gb} - 1)} HW(\alpha_{ij} x_i x_j) \tag{9}$$

The measured power consumptions of these operations corresponding to $h(Q\prime(x))$ must also be accumulated. The precomputation for the measured power trace is described in Algorithm 2, and, subsequently, a multi-bit power analysis attack against QUAD is proposed according to Algorithm 3.

D. Experimental Results

A general evaluation platform for power analysis attacks is shown in Fig. 3, which includes a SASEBO-GII side-channel attack standard evaluation board, a PC including SASEBO software, an Agilent DSO9104 oscilloscope connected to the PC via a local area network (LAN), and a stable power supply. SASEBO-GII is a public standard platform for hardware security evaluation, which features a Xilinx Virtex-5 LX50 device as the target cryptographic FPGA for implementation evaluation, and a Xilinx Spartan3A device as the control FPGA. The cryptographic FPGA performs encryption operations, while the control FPGA controls the oscilloscope, and data flow and communication with the host PC, including the transmission of plaintexts to the cryptographic FPGA, and the return of ciphertexts. The SASEBO-GII and the host PC are connected by a USB cable, via which the modified SASEBO checker running on the PC transmits plaintext and keys to the evaluation board, and receives ciphertexts. The oscilloscope is activated by a trigger to begin measuring the power consumption waveforms of the cryptographic FPGA when executing encryption. The PC polls and copies the waveforms via the LAN, and conducts the power analysis attacks.

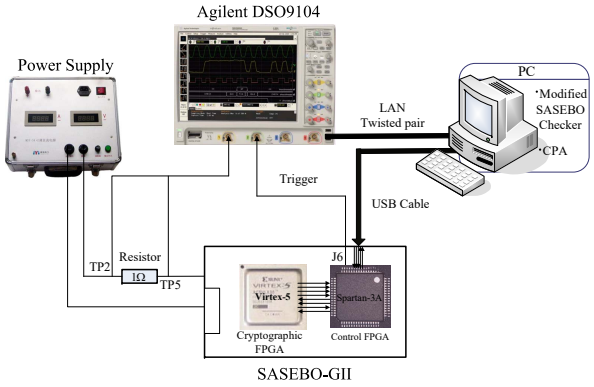


Fig. 3. Experimental setup including a SASEBO-GII side-channel attack standard evaluation board, a PC including SASEBO software, an Agilent DSO9104 oscilloscope connected to the PC via a local area network (LAN), and a stable power supply.

According to the serial implementation of $QUAD(2, n, r)$ illustrated in Fig. 1, an adversary will acquire $n + r$ power traces while capturing the power consumptions of the implementation in the experimental platform during encryption with the same initial internal state X.

The smallest secure version of QUAD that has been recommended [8], [15] has $n = 160$ variables and produces $r = 160$ outputs per round. We will therefore present our experimental results against QUAD(2,160,160).

Figures 4(a) and 4(b) illustrate the single-bit power analysis attack on $QUAD(2, 160, 160)$, where the adversary possesses 320 power traces, and $x_i = 0$ and $x_i = 1$, respectively. As can be observed from the correlation traces, when $x_i = 0$, no positive peak appears in the correlation trace, while, in contrast, when $x_i = 1$, a positive peak is observed in the correlation trace. Figure 4(c) illustrates the results of the single-bit power analysis attack on $QUAD(2, 160, 160)$, where the dashed line corresponds to the correct sample, and

the gray lines correspond to all other samples. Fewer than 15 measurements were required for a successful attack.

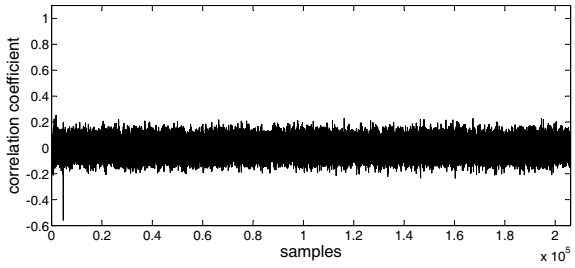
The results of the multi-bit power analysis attack against $QUAD(2, 160, 160)$ are presented in Fig. 4(d) for $N_{gb} = 4$, where the dashed line corresponds to the correct key hypothesis and blue traces represent wrong key hypotheses. The dashed line becomes distinguishable from the blue lines after about 40 measurements. Fig. 4(e) illustrates the success rate of our attack.

E. Complexity of the attack

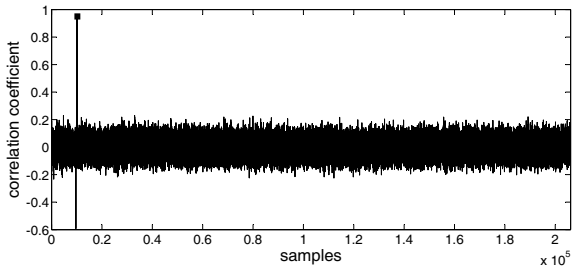
We refer to the length of the subkey attacked by the adversary each time as N_{gb} above, and L refers to the total times that the adversary obtains the entire key, which is equal to length of the key divided by N_{gb}. The number of power traces is given as N_p. For each subkey attack, $2^{N_{gb}}$ key hypotheses are taken into account, and Eq. (5) is computed $2^{N_{gb}}$ times, which yields a complexity of $2^{N_{gb}} \times N_p^2$. Therefore, the complexity of CPA is $L \times 2^{N_{gb}} \times N_p^2$. Based on this analysis, the single-bit CPA attacks on $QUAD(2, 160, 160)$ yield the values $N_{gb} = 1, L = 160, N_p = 320$, and the complexity of is $160 \times 2^1 \times 320^2 \approx 2^{25}$. For the multi-bit CPA attacks on QUAD(2, 160, 160), $N_{gb} = 4, L = 40, N_p = 320$, and the complexity is $40 \times 2^4 \times 320^2 \approx 2^{26}$.

Several cryptanalysis studies have been reported for assessing the security of QUAD, but, to the best of our knowledge, the present results represent the first physical attacks to have been reported. Yang et al. [15] discussed both the theoretical and practical aspects of algebraic attacks of QUAD. Their research pointed out that $QUAD(2, 160, 160)$ was unbroken, but provided no security proof, which, as the authors reported, would have required an estimated 2^{140} cycles. In 2013, Bardet et al. [19] presented an algorithm that reduced the complexity of finding all the common zeros of m quadratic polynomials in n unknowns over $GF(2)$ (i.e., the Boolean multivariate quadratic polynomial problem [Boolean MQ problem]). They showed that, under precise algebraic assumptions for the input system, the deterministic variant of the algorithm had a complexity bounded by $O(2^{0.841n})$ when $m = n$. Applying this algorithm, they analyzed the security of $QUAD(2, n, r)$, which was related to the difficulty of finding at least one solution of the Boolean MQ problem. In the case of $QUAD(2, 160, 160)$, the complexity of solving the Boolean MQ problem was $2^{0.841n} = 2^{0.841 \times 160} = 2^{134.56}$. In 2010, Wong et al. [20] presented a novel approach for preprocessing systems of polynomial equations via graph partitioning. The variable-sharing graph of a system of polynomial equations was defined. If such a graph is disconnected, then the corresponding system of equations can be split into smaller systems that can be solved individually. Based on this technique, the present authors split the multivariate quadratic polynomial of $QUAD(2, 160, 160)$ into 2^{20} smaller systems, each of which consisted of 56 equations in 56 unknowns and 84 equations in 84 unknowns. Applying the algorithm proposed by Bardet et al. [19] to solve these smaller systems, the complexity was estimated as $2^{20} \times 2^{0.841 \times 84} \approx 2^{90.64}$.

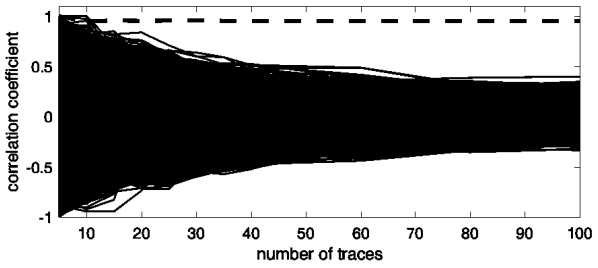
A comparison of the present complexity results with those of reported cryptanalysis results for $QUAD(2, 160, 160)$ are listed in Table I. The complexities of the reported



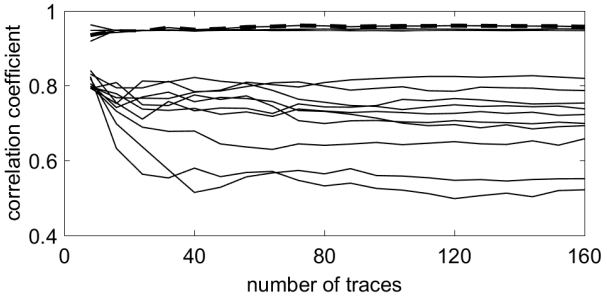
(a) Correlations when key $x_i = 0$



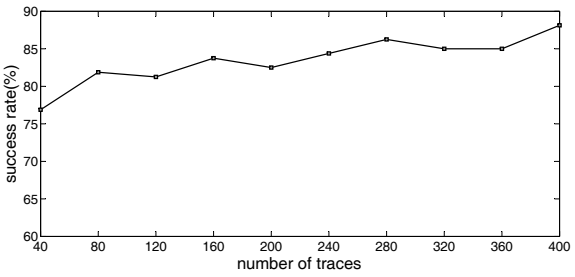
(b) Correlations when key $x_i = 1$



(c) Correlations of each sample with respect to the number of traces



(d) Multi-bit CPA attack against $QUAD(2, 160, 160)$ for a subkey length 4



(e) Success rate of multi-bit CPA attack against $QUAD(2, 160, 160)$

Fig. 4. Correlation power analysis attack on $QUAD(2, 160, 160)$.

TABLE I
A COMPARISON OF THE PRESENT COMPLEXITY RESULTS WITH THOSE OF REPORTED CRYPTANALYSIS RESULTS FOR $QUAD(2, 160, 160)$.

Source	Complexity	Attacks
Yang [15]	2^{140}	XL
Bardet [19]	$2^{134.56}$	SAT
Wong [20]	$2^{90.64}$	Graph Partitioning
Our Attack	2^{25}	Single-bit CPA
Our Attack	2^{26}	Multi-bit CPA

cryptanalysis results are all greater than 2^{80}, which is the generally accepted limit; thus, these cryptanalysis methods cannot put into practice. Meanwhile, the complexities of the proposed methods are much less than 2^{80}, which implies greater efficiency and practicality.

IV. SUGGESTED COUNTERMEASURES

For defensive countermeasures of power analysis attacks, it is naturally concerned to avoid or at lease reduce the dependency between the power consumption of devices and the intermediate values of cryptographic algorithms. Masking and hiding technologies are usual adopted methods, the former defends the power analysis attacks by randomizing the intermediate value during the operation process, while the latter by breaking the link between the power consumption of devices and the processed data values.

To mask the QUAD, a random n-bit mask $M = \{m_1, m_2, \cdots, m_n\}$ is generated inside the device, and X-ORed with the secret key $X = \{x_1, x_2, \cdots, x_n\}$ as masked key $X^{mask} = \{x_1^{mask}, x_2^{mask}, \cdots, x_n^{mask}\}$, which is stored into state register at the beginning of algorithm. A ternary masked multiplier is designed to compute monomial $x_i^{mask} \cdot x_j^{mask} \cdot \alpha_{ij}$, with the result of $(\alpha_{ij} x_i x_j) \oplus m'$, which is accumulated to the temporary register Q_k. The Hamming distance of Q_k is equal to $HW((\alpha_{ij} x_i x_j) \oplus m')$, which is randomized by the mask m' to defends the power analysis attacks.

Since the computation of monomials of each polynomial could be performed in arbitrary order, an alternative is shuffling these operations. The basic idea of this approach is to randomly changing the sequence of these operations, which doesn't change the result of polynomials.

It is the best strategy to counteract power analysis attacks by combining such masking and hiding technologies above.

V. CONCLUSION

Cryptosystems based on the MQ problem, such as QUAD, are regarded as possible alternatives to number theoretic-based cryptosystems under the threat of attacks by future quantum computers. However, unprotected implementations of cryptosystems are vulnerable to side-channel attacks, and must be protected prior to implementation. Although side-channel attacks have been developed over the past 15 years, few such successful attacks have been reported against cryptosystems based on the MQ problem. In this article, we first defined single-bit and multi-bit side-channel leakage models of QUAD based on our observation that MQ-based cryptographic algorithms leak the Hamming weights of monomials computed and restored to the register. We conducted single-bit and multi-bit power analysis attacks against an FPGA implementation of QUAD. A comparison with reported cryptanalysis results for $QUAD(2, 160, 160)$ indicated that our method is the most efficient method of all those considered. Finally, defensive countermeasures against our attacks are proposed.

REFERENCES

[1] Garey, M.R. and D.S. Johnson, A guide to the theory of np-completeness. New York : WH Freemann, 1979.

[2] M. Bardet, J. C. Faugere and B. Salvy, Complexity of gröbner basis computation for semi-regular overdetermined sequences over f_2 with solutions in f_2. Doctoral dissertation, INRIA, 2003.

[3] T. Matsumoto and H. Imai, "Public quadratic polynomial-tuples for efficient signature-verification and message-encryption," Advances in Cryptology-EUROCRYPT 1988, 25-27 May, 1988, Davos, Switzerland, pp419-453.

[4] J. Ding, J. E. Gower and D. S. Schmidt, Multivariate public key cryptosystems. Berlin, US: Springer ,2006.

[5] J. Patarin, "Hidden fields equations (HFE) and isomorphisms of polynomials (IP): Two new families of asymmetric algorithms," Advances in Cryptology-EUROCRYPT 1996, 12-16 May, 1996, Zaragoza, Spain, pp33-48.

[6] J. Ding and D. Schmidt, "Rainbow, a new multivariable polynomial signature scheme," Int. Conf. on Applied Cryptography and Network Security 2005, 7-10 June, 2005, New York, USA, pp164-175.

[7] K. Sakumoto, T. Shirai and H. Hiwatari, "Public-key identification schemes based on multivariate quadratic polynomials," Advances in Cryptology-CRYPTO 2011, 14-18 August, 2011, Santa Barbara, CA, USA, pp706-723.

[8] C. Berbain, H. Gilbert and J. Patarin, "Quad: A practical stream cipher with provable security," Advances in Cryptology-EUROCRYPT 2006, 28 May - 1 June, 2006, St. Petersburg, Russia, pp109-128.

[9] S. Mangard, E. Oswald and T. Popp, Power analysis attacks: Revealing the secrets of smart cards. Berlin, US : Springer , 2007.

[10] Z. He, T.Ao and M. Wan, "ERIST: An efficient randomized instruction insertion technique to counter side-channel attacks," IAENG International Journal of Computer Science, vol. 43, no. 1, pp65-71, 2016.

[11] R. Steinwandt, W. Geiselmann and T. Beth, "A theoretical dpa-based cryptanalysis of the nessie candidates flash and sflash," Int. Conf. on Information Security 2001, 1-3 October, 2001, Malaga, Spain, pp280-293.

[12] K. Okeya, T. Takagi and C. Vuillaume, "On the importance of protecting in sflash against side channel attacks," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. 88, no. 1, pp123-131, 2005.

[13] Y. Hashimoto, T. Takagi and K. Sakurai, "General fault attacks on multivariate public key cryptosystems," IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences, vol. 96, no. 1, pp196-205, 2013.

[14] D. Arditti, C. Berbain, O. Billet and H. Gilbert, "Compact fpga implementations of quad," Proceedings of the 2nd ACM symposium on Information, computer and communications security, 20-22 March, 2007, Singapore, pp347-349.

[15] B.Y. Yang, C.H. Chen, D.J. Bernstein, and J. M. Chen, "Analysis of quad," Int. Wksp. Fast Software Encryption, 26-28 March, 2007, Luxembourg, pp290-308.

[16] T.S. Messerges, E.A. Dabbish and R.H. Sloan, "Investigations of power analysis attacks on smartcards," USENIX workshop on Smartcard Technology, 10-11 May, 1999, Chicago, USA, pp151-162.

[17] E. Brier, C. Clavier and F. Olivier, "Correlation power analysis with a leakage model," Int. Wksp. on Cryptographic Hardware and Embedded Systems 2004, 11-13 August, Boston, USA, pp16-29.

[18] B. Gierlichs, L. Batina and P. Tuyls, "Mutual information analysis," Int. Wksp. on Cryptographic Hardware and Embedded Systems 2008, 10-13 August, 2008, Washington, D.C., USA, pp426-442.

[19] M. Bardet, J. C. Faugère, B. Salvy and P. J. Spaenlehauer, "On the complexity of solving quadratic boolean systems," Journal of Complexity, vol. 29, no. 1, pp53-75, 2013.

[20] K. K. H. Wong and G. V. Bard, "Improved algebraic cryptanalysis of quad, bivium and trivium via graph partitioning on equation systems," Australasian Conference on Information Security and Privacy, 5-7 July, 2010, Sydney, Australia, pp19-36.

Weijian Li received his B.S. degree from Nankai University in 2003, M.S. degree from Harbin Institute of Technology in 2005, and Ph.d. degree from South China University of Technology in 2009. His research interest is side-channel attacks.