



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2012-0028527
(43) 공개일자 2012년03월23일

(51) 국제특허분류(Int. Cl.)
H04L 9/14 (2006.01) H04W 84/18 (2009.01)
(21) 출원번호 10-2010-0090432
(22) 출원일자 2010년09월15일
심사청구일자 2010년09월15일

(71) 출원인
고려대학교 산학협력단
서울 성북구 안암동5가 1
(72) 발명자
홍석희
서울특별시 도봉구 마들로 859-19, 한신아파트
103동 107호 (도봉동)
서석충
서울특별시 중랑구 망우로72가길 22 (망우동)
한동국
인천 계양구 작전2동 772번지 타임빌라 가동 201호
(74) 대리인
특허법인충현

전체 청구항 수 : 총 8 항

(54) 발명의 명칭 GF(2ⁿ) 상의 nT 페어링 방법 및 이를 기록한 저장매체

(57) 요약

본 발명은 GF(2ⁿ)상의 nT 페어링 연산방법에 관한 것으로, GF(2ⁿ)상에서 랜덤 값(λ)을 선택하고, 상기 랜덤 값(λ)을 이용하여 제 1 내지 제4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)을 생성하여 초기점인 $P=(x_P, y_P)$ 와 $Q=(x_Q, y_Q)$ 의 좌표를 마스킹하는 제1 단계; 상기 랜덤 값(λ)과 상기 초기점인 $P=(x_P, y_P)$ 와 $Q=(x_Q, y_Q)$ 을 이용하여 초기 파라미터들(w, u, v, θ)을 세팅하는 제2 단계; 상기 초기 파라미터들(w, u, v, θ)을 이용하여 g_0, g_1, g_2 , 및 g_3 로 이루어진 랜덤 변수(G)를 구하는 제3 단계; 상기 제1 내지 제4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)과 상기 초기 파라미터들을 업데이트 하는 제 4 단계; 중간 페어링 연산 값(A)을 구하고, 상기 랜덤 변수(G)를 업데이트 하는 제5 단계; 상기 중간 페어링 값(A)과 상기 랜덤 변수(G)로 보정과정이 포함된 확장체 곱셈을 수행하는 제6 단계; 및 상기 중간 페어링 연산 값(A)을 연산하는 회수가 일정한 수 이하인 경우 상기 초기 점의 일부좌표(x_P, y_P)와 상기 초기 파라미터들(u, v, θ) 및 상기 제1 내지 제4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)을 업데이트 하여 상기 제5 및 제6 단계를 다시 수행하는 제7 단계를 포함함으로써, 기존의 RVA 기반의 DPA 대응방법의 취약성이 보완된다.

대표도 - 도4

Algorithm 3 nT Pairing Algorithm over GF(2ⁿ) with the Proposed DPA Countermeasure

```

1: Require  $P=(x_P, y_P), Q=(x_Q, y_Q)$ 
2: Ensure  $G \leftarrow \eta T(P, Q)$ 
3: Prepare random values  $\lambda$  in  $GF(2^n)$ .
4:  $w \leftarrow (x_P \cdot \lambda)$ . // (1M)
5:  $\lambda_0 \leftarrow \lambda, \lambda_1 \leftarrow \lambda_0^2, \lambda_2 \leftarrow \lambda_1^2, \lambda_3 \leftarrow \lambda_1$ . // (2S)
6:  $x_P \leftarrow x_P^2 + 1, y_P \leftarrow (y_P + \lambda_1 + \lambda_0)^2 + 1, w \leftarrow w^2, x_Q \leftarrow (x_Q + \lambda_1), y_Q \leftarrow (y_Q + \lambda_2 + \lambda_0)$ . // (3S)
7:  $u \leftarrow y_Q + b + 1, v \leftarrow (x_Q + 1), \theta \leftarrow x_P \cdot v$ . // (1M)
8:  $G \leftarrow (\theta + u + w + y_P) + (x_P + v + 1)s + t$ .
9:  $G \leftarrow \lambda_0 \cdot G + \lambda_1 + (\lambda_1 \cdot \lambda_0)s$ . // (3M)
10:  $G \leftarrow G^2$ . // (4S)
11:  $x_P \leftarrow x_P^2, y_P \leftarrow y_P^2, T \leftarrow \lambda_0 + \lambda_1, \lambda_3 \leftarrow \lambda_1, \lambda_0 \leftarrow \lambda_0^2, \lambda_1 \leftarrow \lambda_1^2, \lambda_2 \leftarrow \lambda_2^2, \lambda_3 \leftarrow \lambda_3 + \lambda_1, w \leftarrow w^2, T \leftarrow T + \lambda_2$ . // (12S)
12:  $u \leftarrow u + v + 1 + T, v \leftarrow v + 1 + \lambda_3, \theta \leftarrow x_P \cdot v$ . // (1M)
13: for from  $i \leftarrow 1$  to  $\frac{n-1}{2}$  do
14:    $A \leftarrow (\theta + u + w + y_P) + (x_P + v + 1)s + t$ .
15:    $G \leftarrow G^2$ . // (4S)
16:    $G \leftarrow A \cdot G$ . // Call the Algorithm 4 with (6M+2S).
17:   if  $i < \frac{n-1}{2}$  then
18:      $x_P \leftarrow x_P^2, y_P \leftarrow y_P^2, T \leftarrow T^2, \lambda_3 \leftarrow \lambda_1, \lambda_0 \leftarrow \lambda_0^2, \lambda_1 \leftarrow \lambda_1^2, \lambda_2 \leftarrow \lambda_2^2, \lambda_3 \leftarrow \lambda_3 + \lambda_1, w \leftarrow w^2$ . // (14S)
19:      $u \leftarrow u + v + 1 + T, v \leftarrow v + 1 + \lambda_3, \theta \leftarrow x_P \cdot v$ . // (1M)
20:   end if
21: end for
22:  $A \leftarrow (x_P^2 + v + 1)s + t$ . // (1S)
23:  $G \leftarrow G \cdot A$ . // (6M+2S)
24:  $G \leftarrow G^M, M = (2^{2n} - 1)(2^n + 2^{\frac{n-1}{2}} + 1)$ .
25: return  $G$ .
```

특허청구의 범위

청구항 1

$GF(2^n)$ 상에서 랜덤 값(λ)을 선택하고, 상기 랜덤 값(λ)을 이용하여 제 1 내지 제4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)을 생성하여 초기점인 $P=(x_P, y_P)$ 와 $Q=(x_Q, y_Q)$ 의 좌표를 마스킹하는 단계;

상기 랜덤 값(λ)과 상기 초기점인 $P=(x_P, y_P)$ 와 $Q=(x_Q, y_Q)$ 을 이용하여 초기 파라미터들(w, u, v, θ)을 세팅하는 단계;

상기 초기 파라미터들(w, u, v, θ)을 이용하여 g_0, g_1, g_2 , 및 g_3 로 이루어진 랜덤 변수(G)를 구하는 단계;

상기 제1 내지 제4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)과 상기 초기 파라미터들을 업데이트 하는 단계;

중간 페어링 연산 값(A)을 구하고, 상기 랜덤 변수(G)를 업데이트하는 단계;

상기 중간 페어링 값(A)과 상기 랜덤 변수(G)로 보정과정이 포함된 확장체 곱셈을 수행하는 단계; 및

상기 중간 페어링 연산 값(A)을 연산하는 회수가 일정한 수 이하인 경우 상기 초기 점의 일부좌표(x_P, y_P)와 상기 초기 파라미터들(u, v, θ) 및 상기 제1 내지 제4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)을 업데이트하여 상기 랜덤 변수(G)를 업데이트하는 단계 및 확장체 곱셈을 수행하는 단계를 다시 수행하는 단계를 포함하는 $GF(2^n)$ 상의 nT 페어링 연산방법.

청구항 2

제 1 항에 있어서,

상기 제 1 내지 제4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)을 생성하는 것은 상기 랜덤 값(λ)을 제공하거나 상기 랜덤 값(λ)을 그대로 이용하는 것을 특징으로 하는 $GF(2^n)$ 상의 nT 페어링 연산방법.

청구항 3

제 1 항에 있어서,

상기 초기점인 $P=(x_P, y_P)$ 와 $Q=(x_Q, y_Q)$ 의 좌표를 마스킹하는 것은 $x_P \leftarrow x_P^2 + 1$, $y_P \leftarrow (y_P + \lambda_1 + \lambda_0)^2 + 1$, $x_Q \leftarrow (x_Q + \lambda_1)$, $y_Q \leftarrow (y_Q + \lambda_2 + \lambda_0)$ 와 같은 연산을 수행하는 것을 특징으로 하는 $GF(2^n)$ 상의 nT 페어링 연산방법.

청구항 4

제 1 항에 있어서,

상기 초기 파라미터들(w, u, v, θ)을 세팅하는 것은 $w \leftarrow (x_P ? \lambda)$, $w \leftarrow w^2$, $u \leftarrow y_Q + b + 1$, $v \leftarrow (x_Q + 1)$, 및 $\theta \leftarrow x_P ? v$ 와 같은 연산으로 수행하는 것을 특징으로 하는 $GF(2^n)$ 상의 nT 페어링 연산방법.

청구항 5

제 1 항에 있어서,

상기 랜덤 변수(G)를 구하는 것은 x_P, y_P 와 초기 파라미터인 u, v , 및 θ 을 $(\theta + u + w + y_P) + (x_P + v + 1)s + t$ 에 대입하여 상기 랜덤 변수(G)를 얻고, 이를 다시 $\lambda_0 ? G + \lambda_1 + (\lambda_1 ? \lambda_0)s$ 에 대입하여 얻어진 상기 랜덤 변수(G)를 제공하여 $(g_0) + (g_1)s + (g_2)t + (g_3)st$ 의 형태로 변환하여 g_0, g_1, g_2 및 g_3 을 구하는 것을 특징으로 하는 $GF(2^n)$ 상의 nT 페어링 연산방법.

청구항 6

제 1 항에 있어서,

상기 랜덤 변수(G)를 업데이트하는 단계는 $x_P \leftarrow x_P^4$, $y_P \leftarrow y_P^4$, $T \leftarrow \lambda_0 + \lambda_1$, $\lambda_3 \leftarrow \lambda_1$, $\lambda_0 \leftarrow \lambda_0^4$, $\lambda_1 \leftarrow \lambda_1^4$, $\lambda_2 \leftarrow \lambda_2^4$, $\lambda_3 \leftarrow \lambda_3 + \lambda_1$, $w \leftarrow w^4$, $T \leftarrow T + \lambda_2$, $u \leftarrow u + v + 1 + T$, $v \leftarrow v + 1 + \lambda_3$, $\Theta \leftarrow x_P ? v$ 와 같은 연산을 수행하는 것을 특징으로 하는 GF(2^n)상의 nT 페어링 연산방법.

청구항 7

제 1 항에 있어서,

상기 확장체 곱셈을 수행하는 단계는 상기 중간 페어링 연산 값(A)과 상기 랜덤 변수(G) 및 상기 제1 및 제2 랜덤 값(λ_0 , λ_1)을 요구하고, 상기 중간 페어링 연산 값(A)의 상기 제1 및 제2 랜덤 값(λ_0 , λ_1)을 업데이트하고, 상기 제1 및 제2 랜덤 값(λ_0 , λ_1)이 업데이트된 상기 중간 페어링 연산 값(A)과 상기 중간 랜덤변수(G)를 확장체 곱셈 연산하는 것을 특징으로 하는 GF(2^n)상의 nT 페어링 연산방법.

청구항 8

제1항 내지 제7항 중 어느 하나의 방법을 컴퓨터에서 실행시키기 위한 프로그램을 기록한 컴퓨터로 읽을 수 있는 기록매체.

명세서

기술분야

[0001] 본 발명은 GF(2^n)상의 nT 페어링 방법에 관한 것으로서, 더욱 상세하게는 기존의 RVA 기반의 DPA 대응방법의 보안 취약성을 보완한 GF(2^n)상의 nT 페어링 연산방법 및 이를 기록한 저장매체에 관한 것이다.

배경기술

[0002] 타원 곡선상의 페어링은 타원 곡선상에서 정의되는 두 점을 유한체 상의 원소로 매핑하는 함수로서 곱선형성(bilinearity)과 비중첩성(non-degeneracy)을 특징으로 한다. 최근 페어링 연산의 곱선형성과 비중첩성을 이용하는 ID 기반의 암호/서명/인증/키 공유 기법 등 다양한 프로토콜들이 개발되고 있다.

[0003] 그런데 페어링 암호 연산은 기존의 타원곡선 암호에 비하여 상대적으로 큰 부하를 발생시키기 때문에 실질적인 프로토콜에의 적용을 위하여 효율적인 페어링 암호에 대한 연구가 진행되어왔다. 초기에는 Tate 페어링 연산을 최적화하는 것에 대한 연구가 진행되었다. 즉, [문헌 1]과 [문헌 2]는 밀러(Miller) 루프 연산에서의 분모(denominator) 제거 방법 및 GF(3^n)상에서의 효율적인 타원곡선 삼중(Tripling) 연산 및 페어링 연산을 제안하였다. [문헌 3]은 위수 3인 유한체 상에서 페어링 연산을 구성하는 타원곡선 연산과 약수(divisor) 계산을 효율적인 닫힌 공식(closed formula)으로 구성하였다. [문헌 1]은 초특이(supersingular) 커브 상에서 [문헌 3]에서 제시된 방법을 이진체에 적용하여 유사한 결과를 도출하였으며 뿐만 아니라 기존의 [문헌 3]에서 제시된 방법에서의 밀러 루프 횟수를 절반으로 줄인 nT 페어링 방법을 일반화하여 제시하였다. [문헌 4]는 초특이(supersingular) 커브에서 제안된 nT 페어링 방법을 일반적인 커브로 확장한 Ate 페어링을 제안하였다.

[0004] 부채널 분석(Side Channel Analysis)은 암호 알고리즘이 특정 장치에서 동작할 때 의도하지 않게 발생하는 내부 데이터 및 키와 연관된 부가적인 정보(시간 정보, 전력 소모량, 전자기파 등)를 이용하여 암호 시스템의 비밀 정보를 알아내는 기법이다.

[0005] 페어링 알고리즘에 대한 부채널 분석에 대한 연구는 [문헌 5], [문헌 6], [문헌 7], [문헌 8], [문헌 9]에 의하여 수행되어왔다. 타원곡선 암호와는 달리 페어링 알고리즘의 밀러(Miller) 루프에서 사용되는 스칼라는 비밀 정보가 아니기 때문에 페어링 암호에서의 부채널 분석은 비밀키 역할을 하는 타원곡선 점을 알아내는 것을 목표로 한다. 즉, 페어링 연산에서의 유한체 연산에 단순전력분석(SPA: Simple Power Analysis), 차분전력분석

(DPA: Differential Power Analysis)을 적용하여 연산되는 타원곡선 점의 값을 알아낸다.

- [0006] 현재까지 제안된 페어링 암호에서의 부채널 분석 대응방법들은 크게 세 가지로 분류할 수 있다.
- [0007] 첫 번째는 페어링 연산의 곱선형성(bilinearity)을 이용하는 방법으로서, [문헌 5]에서는 타원곡선 점을 랜덤 점으로 숨기기(blinding) 위하여 곱선형성(bilinearity)을 이용하였다.
- [0008] 두 번째는 페어링 연산의 중간 변수들에 랜덤한 값을 곱하는 방법으로서 가장 많이 연구된 방법이다. [문헌 6]은 Tate 페어링 계산 시에 SPA와 DPA를 방어하기 위한 방법으로 밀러 루프의 연산에 사용되는 변수들에 랜덤변수를 곱하는 방법을 제안하였으며, [문헌 8]과 [문헌 7]은 $GF(2^n)$ 상의 nT 페어링의 밀러 루프의 연산에 사용되는 모든 중간 변수들에 랜덤 값을 곱하는 방법을 제안하였다. [문헌 8]은 밀러 루프의 중간 변수들을 랜덤화하기 위하여 무작위사영좌표(randomized projective coordinate)(RPC)방법을 적용하였다. [문헌 9]는 [문헌 8]의 방법을 개량하여 좀 더 적은 연산량으로 동일한 방법의 대응방법을 제시하였다.
- [0009] 세 번째 방법은 [문헌 10]에서 제안한 방법으로서 $GF(3^n)$ 상의 nT 페어링 알고리즘의 부채널 대응방법으로서 랜덤 값을 더하는 방법(Random Value Addition: RVA)이다. RVA 기반의 대응방법은 $GF(2^n)$ 상의 nT 페어링 방법에 쉽게 적용될 수 있다. 하지만, 이 방법은 효율성 측면에서는 매우 뛰어나지만 밀러 루프의 연산 과정에서 랜덤 값이 상쇄되어 중간 결과 값이 노출되는 취약성이 있다.

선행기술문헌

비특허문헌

- [0010] (비특허문헌 0001) [문헌 1] P. S. L. M. Barreto, H. Y. Kim, B. Lynn, and M. Scott, "Efficient Algorithms for Pairing-based Cryptosystems", CRYPTO 2002, LNCS 2442, pp. 354-368, 2002.
- (비특허문헌 0002) [문헌 2] S. D. Galbraith, K. Harrison, and D. Soldera, "Implementing the Tate Pairing", ANTS V, LNCS 2369, pp. 324-337, 2002.
- (비특허문헌 0003) [문헌 3] I. Duursma and H. S. Lee, "Tate Pairing Implementation for Hyperelliptic Curves $y^2=xp-x+d$ ", Asiacrypt 2003, LNCS 2894, pp. 111-123, 2003.
- (비특허문헌 0004) [문헌 4] F. Hess, N. Smart, and F. Vercauteren, "The Eta Pairing Revisited", IEEE Trans. Inf. Theory. 52 no. 10 pp. 4595-4602, 2006.
- (비특허문헌 0005) [문헌 5] D. Page and F. Vercauteren, "Fault and Side-Channel Attacks on Pairing Based Cryptography", Cryptology ePrint Archive, Report 2004/283, 2005.
- (비특허문헌 0006) [문헌 6] M. Scott, "Computing the Tate Pairing", CT-RSA 2005, LNCS 3376, pp. 293-304, 2005.
- (비특허문헌 0007) [문헌 7] C. Whelan and M. Scott, "Side Channel Analysis of Practical Pairing Implementations: Which Path is More Secure?", VIETCRYPT 2006, LNCS 4341, pp. 99-114, 2006.
- (비특허문헌 0008) [문헌 8] T. H. Kim, T. Takagi, D.-G. Han, H. W. Kim and J. Lim, "Power Analysis Attacks and Countermeasures on Pairing over Binary Fields", ETRI Journal, Volume 30, Number 1, pp. 68-80, 2008.
- (비특허문헌 0009) [문헌 9] Dooho Choi, Dongguk Han, and Howon Kim, "Construction of Efficient and Secure Pairing Algorithm and Its Application", Journal of Communications and Networks, Vol. 10, No. 4, pp. 437-443, 2008.
- (비특허문헌 0010) [문헌 10] M. Shirase, T. Takagi, and E. Okamoto "An Efficient Countermeasure against Side Channel Attacks for Pairing Computation", ISPEC 2008, LNCS 4991, pp. 290-303, 2008.

발명의 내용

해결하려는 과제

- [0011] 따라서 본 발명이 해결하고자 하는 첫 번째 과제는 기존의 RVA 기반의 DPA 대응방법의 취약성을 보완한 $GF(2^n)$ 상의 nT 페어링 방법을 제공하는 것이다.
- [0012] 또한, 본 발명이 해결하고자 하는 두 번째 과제는 기존의 RVA 기반의 DPA 대응방법의 취약성을 보완한 $GF(2^n)$ 상의 nT 페어링 방법을 기록한 기록 매체를 제공하는 것이다.

과제의 해결 수단

- [0013] 본 발명은 상기 첫 번째 과제를 달성하기 위하여, $GF(2^n)$ 상에서 랜덤 값(λ)을 선택하고, 상기 랜덤 값(λ)을 이용하여 제 1 내지 제 4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)을 생성하여 초기점인 $P=(x_p, y_p)$ 와 $Q=(x_q, y_q)$ 의 좌표를 마스킹하는 제1 단계; 상기 랜덤 값(λ)과 상기 초기점인 $P=(x_p, y_p)$ 와 $Q=(x_q, y_q)$ 을 이용하여 초기 파라미터들(w, u, v, θ)을 세팅하는 제2 단계; 상기 초기 파라미터들(w, u, v, θ)을 이용하여 g_0, g_1, g_2 , 및 g_3 로 이루어진 랜덤 변수(G)를 구하는 제3 단계; 상기 제1 내지 제 4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)과 상기 초기 파라미터들을 업데이트 하는 제 4 단계; 중간 페어링 연산 값(A)을 구하고, 상기 랜덤 변수(G)를 업데이트 하는 제5 단계; 상기 중간 페어링 값(A)과 상기 랜덤 변수(G)로 보정과정이 포함된 확장체 곱셈을 수행하는 제6 단계; 및 상기 중간 페어링 연산 값(A)을 연산하는 회수가 일정한 수 이하인 경우 상기 초기 점의 일부좌표(x_p, y_p)와 상기 초기 파라미터들(u, v, θ) 및 상기 제1 내지 제 4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)을 업데이트 하여 상기 제5 및 제6 단계를 다시 수행하는 제7 단계를 포함하는 $GF(2^n)$ 상의 nT 페어링 연산방법을 제공한다.
- [0014] 본 발명의 일실시예에 의하면, 상기 제 1 내지 제 4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)을 생성하는 것은 상기 랜덤 값(λ)을 제공하거나 상기 랜덤 값(λ)을 그대로 이용하는 것을 특징으로 한다.
- [0015] 또한, 상기 초기점인 $P=(x_p, y_p)$ 와 $Q=(x_q, y_q)$ 의 좌표를 마스킹하는 것은 $x_p \leftarrow x_p^2 + 1$, $y_p \leftarrow (y_p + \lambda_1 + \lambda_0)^2 + 1$, $x_q \leftarrow (x_q + \lambda_1)$, $y_q \leftarrow (y_q + \lambda_2 + \lambda_0)$ 와 같은 연산을 수행하는 것을 특징으로 한다.
- [0016] 또한, 상기 초기 파라미터들(w, u, v, θ)을 세팅하는 것은 $w \leftarrow (x_p ? \lambda)$, $w \leftarrow w^2$, $u \leftarrow y_q + b + 1$, $v \leftarrow (x_q + 1)$, 및 $\theta \leftarrow x_p ? v$ 와 같은 연산으로 수행하는 것을 특징으로 한다.
- [0017] 또한, 상기 랜덤 변수(G)를 구하는 것은 x_p, y_p 와 초기 파라미터인 u, v , 및 θ 을 $(\theta + u + w + y_p) + (x_p + v + 1)s + t$ 에 대입하여 상기 랜덤 변수(G)를 얻고, 이를 다시 $\lambda_0 ? G + \lambda_1 + (\lambda_1 ? \lambda_0)s$ 에 대입하여 얻어진 상기 랜덤 변수(G)를 제공하여 $(g_0) + (g_1)s + (g_2)t + (g_3)st$ 의 형태로 변환하여 g_0, g_1, g_2 및 g_3 을 구하는 것을 특징으로 한다.
- [0018] 또한, 상기 제5 단계는 $x_p \leftarrow x_p^4$, $y_p \leftarrow y_p^4$, $T \leftarrow \lambda_0 + \lambda_1$, $\lambda_3 \leftarrow \lambda_1$, $\lambda_0 \leftarrow \lambda_0^4$, $\lambda_1 \leftarrow \lambda_1^4$, $\lambda_2 \leftarrow \lambda_2^4$, $\lambda_3 \leftarrow \lambda_3 + \lambda_1$, $w \leftarrow w^4$, $T \leftarrow T + \lambda_2$, $u \leftarrow u + v + 1 + T$, $v \leftarrow v + 1 + \lambda_3$, $\theta \leftarrow x_p ? v$ 와 같은 연산을 수행하는 것을 특징으로 한다.
- [0019] 또한, 상기 제6 단계는 상기 중간 페어링 연산 값(A)과 상기 랜덤 변수(G) 및 상기 제1 및 제2 랜덤 값(λ_0, λ_1)을 요구하고, 상기 중간 페어링 연산 값(A)의 상기 제1 및 제2 랜덤 값(λ_0, λ_1)을 업데이트하고, 상기 제1 및 제2 랜덤 값(λ_0, λ_1)이 업데이트된 상기 중간 페어링 연산 값(A)과 상기 중간 랜덤변수(G)를 확장체 곱셈 연산하는 것을 특징으로 한다.
- [0020] 본 발명은 상기 두 번째 과제를 달성하기 위하여, 상기 방법을 컴퓨터에서 실행시키기 위한 프로그램을 기록한 컴퓨터로 읽을 수 있는 기록매체를 제공한다.

발명의 효과

- [0021] 본 발명에 따르면, 기존의 RVA 기반의 DPA 대응방법의 취약성이 보완된다. 즉, 랜덤 값에 대한 보정이 확장체 곱셈 연산과정에서 이루어짐으로써 중간연산 값의 노출이 없게 되어 소비 전력 분석에 대응할 수 있게 된다. 또

한, 본 발명에 따르면, 연산 부하를 기존에 제안된 DPA 대응방법과 비교하여 크게 줄일 수 있다.

도면의 간단한 설명

- [0022] 도 1은 종래의 부 채널 분석에 대응방법이 없는 $GF(2^n)$ 상에서 정의되는 nT 페어링 연산 알고리즘을 나타낸 도면이다.
- 도 2는 종래의 문헌 10에서 제안한 RVA를 기반으로 한 대응방법을 $GF(2^n)$ 상에서 수정하여 나타낸 알고리즘이다.
- 도 3은 본 발명의 일실시예에 따른 $GF(2^n)$ 상의 nT 페어링 연산방법을 나타낸 흐름도이다.
- 도 4는 본 발명의 일실시예에 따른 $GF(2^n)$ 상의 nT 페어링 연산방법을 구체화한 알고리즘이다.
- 도 5는 본 발명의 일실시예에 따른 $GF(2^n)$ 상의 nT 페어링 연산방법의 보정을 포함한 확장체 곱셈 연산방법을 나타낸 흐름도이다.
- 도 6은 본 발명의 일실시예에 따른 $GF(2^n)$ 상의 nT 페어링 연산방법의 보정을 포함한 확장체 곱셈 연산방법의 구체적인 알고리즘을 나타낸 도면이다.

발명을 실시하기 위한 구체적인 내용

- [0023] 본 발명에 관한 구체적인 내용의 설명에 앞서 이해의 편의를 위해 본 발명이 해결하고자 하는 과제의 해결 방안의 개요 혹은 기술적 사상의 핵심을 우선 제시한다.
- [0024] 본 발명의 일실시예에 따른 $GF(2^n)$ 상의 nT 페어링 연산방법은 $GF(2^n)$ 상에서 랜덤 값(λ)을 선택하고, 상기 랜덤 값(λ)을 이용하여 제 1 내지 제4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)을 생성하여 초기점인 $P=(x_P, y_P)$ 와 $Q=(x_Q, y_Q)$ 의 좌표를 마스킹하는 제1 단계; 상기 랜덤 값(λ)과 상기 초기점인 $P=(x_P, y_P)$ 와 $Q=(x_Q, y_Q)$ 을 이용하여 초기 파라미터들(w, u, v, θ)을 세팅하는 제2 단계; 상기 초기 파라미터들(w, u, v, θ)을 이용하여 g_0, g_1, g_2 및 g_3 로 이루어진 랜덤 변수(G)를 구하는 제3 단계; 상기 제1 내지 제4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)과 상기 초기 파라미터들을 업데이트 하는 제 4 단계; 중간 페어링 연산 값(A)을 구하고, 상기 랜덤 변수(G)를 업데이트 하는 제5 단계; 상기 중간 페어링 값(A)과 상기 랜덤 변수(G)로 보정과정이 포함된 확장체 곱셈을 수행하는 제6 단계; 및 상기 중간 페어링 연산 값(A)을 연산하는 회수가 일정한 수 이하인 경우 상기 초기 점의 일부좌표(x_P, y_P)와 상기 초기 파라미터들(u, v, θ) 및 상기 제1 내지 제4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)을 업데이트 하여 상기 제5 및 제6 단계를 다시 수행하는 제7 단계를 포함한다.
- [0025] 이하, 바람직한 실시예를 들어 본 발명을 더욱 상세하게 설명한다. 그러나 이들 실시예는 본 발명을 보다 구체적으로 설명하기 위한 것으로, 본 발명의 범위가 이에 의하여 제한되지 않는다는 것은 당업계의 통상의 지식을 가진 자에게 자명할 것이다. 아울러 본 발명과 관련된 공지 기능 혹은 구성에 대한 구체적인 설명 그리고 그 이외의 제반 사항이 본 발명의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우, 그 상세한 설명을 생략한다.
- [0026] 이하에서는 본 발명의 일실시예에 따른 $GF(2^n)$ 상의 nT 페어링 연산 방법을 설명하기에 앞서 $GF(2^n)$ 상에서 정의되는 nT 페어링 및 이의 부 채널 분석의 취약성을 설명하기로 한다.
- [0027] [$GF(2^n)$ 상에서 정의되는 nT 페어링]
- [0028] E 를 $GF(2^n)$ 상에서 아래의 수학적 식 1과 같은 초특이(supersingular) 타원곡선(E_b)으로 정의한다.

수학적 식 1

$$E_b : Y^2 + Y = X^3 + X + b, b \in \{0, 1\}.$$

[0029]

[0030] l 은 $l \mid \#E(GF(2^n))$ 인 최대 소수(largest prime)이고, $l \mid 2^{kn}$ 를 만족하게 하는 타원곡선의 embedding degree인 최소 양의 정수 k 는 구체적으로는 4로 정의한다. 그러면 아래의 수학식 2와 같은 초특이 타원곡선 집합의 원소 개수(N)를 얻는다.

수학식 2

$$N = |E_b(GF(2^n))| = 2^n + 1 + (-1)^b 2^{(n+1)/2}, \text{ if } n \equiv 1, 7 \pmod{8} \\ = 2^n + 1 - (-1)^b 2^{(n+1)/2}, \text{ if } n \equiv 3, 5 \pmod{8}$$

[0032] 그 다음, T 를 $2^n - N$ 로 선택하면 [문헌 1]이 제안한 nT 페어링은 아래의 수학식 3과 같은 맵핑 함수가 된다.

수학식 3

$$\eta_T : E(GF(2^n))[l] \times E(GF(2^n))[l] \rightarrow GF(2^{4n})^*.$$

[0034] 즉, $nT(P, Q) = f_{T,P}(\Psi(Q))$ 에서 $T < 0$ 이면 P 를 $-P$ 로 대체하고 T 를 $-T$ 로 대체할 수 있다. Ψ 는 뒤틀림 맵(distortion map)이고, 이는 $E(GF(2^n))[1]$ 의 원소를 $E(GF(2^{4n}))[1]$ 의 원소로 변환한다. $\Psi(x, y)$ 는 $(x + s^2, y + sx + t)$ 로 정의된다. 이때, $s^2 + s + 1 = 0$, $t^2 + t + s = 0$ 이고, $s, t \in GF(2^{4n})$ 이다. 따라서 기저인 $1, s, t, st$ 들을 이용하여 $GF(2^n)$ 의 확장체로 $GF(2^{4n})$ 의 원소를 표현할 수 있다.

[0035] [문헌 1]에 따르면, 마지막 제곱(final powering)을 포함하는 $f_{T,P}^M$ 는 아래의 수학식 4와 같다.

수학식 4

$$f_{T,P}(\psi(Q))^M = (l(\psi(Q)) \prod_{i=0}^{\frac{n-1}{2}} g_{[2^i P]}(\psi(Q))^{2^{\frac{n-1}{2}-i}})^M$$

[0037] 여기서, $M = (2^{2n} - 1)(2^n \mp 2^{\frac{n+1}{2}} + 1)$ 이다.

[0038] 수학식 4에서 점을 이중화하는 공식은 아래의 수학식 5와 같다.

수학식 5

$$[2^i]P = (x_P^{(2i)} + i, y_P^{(2i)} + ix_P^{(2i)} + \epsilon_i)$$

[0040] 여기서, $i \equiv i \pmod{n}$ 인 경우 $x_P^{(2i)} = x_P^{2^{2i}}$ 와 같으며, ϵ_i 는 $i \equiv 0, 1 \pmod{4}$ 인 경우에는 0이며 그렇지 않은 경우에는 1이다.

[0041] 또한, $g_V(x, y)$ 는 $2(V) + (-2)V - 3(\infty)$ 와 같은 약수를 가지며 $E(GF(2^{4n}))[1]$ 상에 정의되는 유리함수이다. 여기서, V 는 $E(GF(2^n))[1]$ 에 포함되는 (x_V, y_V) 이고, (x, y) 는 $E(GF(2^{4n}))[1]$ 에 포함된다. 이러한 $g_V(x, y)$ 는 아래의 수학식 6과 같다.

수학식 6

$$g_V(x, y) = (x_V^2 + 1)(x_V + x) + y_V + y = x(x_V^2 + 1) + y_V^2 + y + b.$$

또한, l_V 는 ϵ 가 $\epsilon = (-1)^{b+\epsilon(n+1)/2}$. 일 때, $2^{(n+1)/2}V$ 와 ϵV 를 지나는 선의 방정식으로 아래의 수학식 7과 같다.

수학식 7

$$l_V(x, y) = y + y_V + b + \epsilon_{(n+1)/2} + (x_V + \frac{n-1}{2})(x + x_V).$$

Tate 페어링과 같이, nT 페어링은 s, t 가 정수이고 P, Q 가 $E(GF(2^n))$ 에 속하는 경우, $nT(sP, tQ) = nT(tP, sQ) = nT(P, Q)^{st}$ 와 같이 곱셈형성을 가진다.

도 1은 종래의 부 채널 분석에 대응방법이 없는 $GF(2^n)$ 상에서 정의되는 nT 페어링 연산 알고리즘을 나타낸 도면이다. 도 1에서 M 은 $GF(2^n)$ 상에서의 곱셈연산을 의미하고, S 는 $GF(2^n)$ 상에서의 제곱연산을 의미한다. 도 1에서 도시되는 바와 같이, $GF(2^n)$ 상에서 정의되는 nT 페어링 연산 알고리즘은 마지막 거듭제곱(final powering)을 제외하면 $(3.5n+9.5)M+(4n+3)S$ 의 연산이 필요하다.

[부 채널 분석에 대응방법이 없는 경우 $GF(2^n)$ 상에서 정의되는 nT 페어링의 취약성]

밀러루프의 시퀀스는 비밀정보가 아닌 공개정보이므로, 페어링 알고리즘의 덧셈연산과 곱셈연산과 같은 유한체 연산은 부 채널 분석의 대상이 된다. 즉, 부 채널 분석의 목적은 고정 개인 점인 x 또는 y 의 좌표를 알아내는 것이다. 공격자는 부 채널 분석을 수행하기 위해 실제 중간 결과값을 알아내야 한다. 예를 들면, $(k \circ p)$ 와 같은 연산에서 k 는 공격자에게 비공개되어 고정되고 p 는 공격자에게 공개되어 제어 가능한 경우, 공격자는 소비전력 피크를 관찰하여 k 를 알아낼 수 있다. 즉, 공격자는 $(k \oplus p)$ 연산의 결과 중 최소 특이 바이트(least significant bit: LSB)를 p 의 LSB가 공격자에게 1로 알려진 경우 1로 추측한 후, DPA 시뮬레이션 중에 특이한 피크가 있는 경우, k 의 LSB는 0이다.

위와 같은 부 채널 분석에 대한 대응방법이 없는 $GF(2^n)$ 상에서 정의되는 nT 페어링 연산 알고리즘에서의 부 채널 분석의 대상은 아래와 같이 정리될 수 있다.

1.(고정 및 비밀 값)+(가변 및 공개 값)

(1) 과정 7에서 $(x_P+v+1)s$ 연산

(2) 과정 7에서 (y_P+u) 연산이 $(y_P+\Theta)$ 연산 또는 $(\Theta+u)$ 연산보다 먼저 되는 경우 $(y_P+\Theta+u)$ 연산

(3) 과정 16에서 x_P^2+v+1 연산

2.(고정 및 비밀 값)*(가변 및 공개 값)

(1) 과정 5에서 $\Theta \leftarrow x_P \cdot v$ 연산

(2) 과정 13에서 $\Theta \leftarrow x_P \cdot v$ 연산

위와 같은 nT 페어링이 부 채널 분석되는 것을 방지하기 위해 아래와 같은 부 채널 분석에 대한 대응방법이 제

안되었다.

[0058] $GF(2^n)$ 에 대한 nT 페어링의 부채널 분석에 대한 대응방법]

[0059] 1. 곱선형성을 이용한 대응방법

[0060] [문헌 5]에서는 $e(P,Q)=e(sP, sQ)^{1/st}$ 와 같은 개인 데이터를 랜덤화하기 위해 곱선형성(bilinearity)을 이용하였다. 여기서, s, t 는 랜덤 값이다. 나아가서, [문헌 5]에서는 $e(P,Q)=e(P,Q+R)?e(P,R)^{-1}$ 와 같이 계산하여 공개 입력 점을 은닉하였다. 여기서, R 은 랜덤 점이다.

[0061] 2. RPC에 기반으로 한 대응방법

[0062] [문헌 8]은 타원 곡선 암호를 부 채널 분석하는 것을 방지하기 위해 이용되던 무작위 사영 좌표(randomized projective coordinate)를 이용하였다. [문헌 8]은 입력 점인 $Q=(x_Q, y_Q)$ 을 $(X_Q, Y_Q, Z_Q)=(\lambda x_Q, \lambda y_Q, \lambda)$ 와 같이 변환하였다. 여기서, λ 는 $GF(2^n)$ 상의 랜덤 값이다. 그 다음, [문헌 8]은

[0063] $\frac{1}{Z_Q} \{ (x_P(Z_Q\sqrt{x_P}+X_Q)+Z_Q(\sqrt{y_P}+\sqrt{x_P})+Y_Q)+(Z_Qx_P+X_Q)s+(Z_Q)t \}$ 와 같은 공식을 제안하였다. 여기서 입력점 (Q) 의 좌표가 랜덤화되었을 뿐만 아니라, 개인 좌표점(P)은 랜덤화된 중간값으로 마스킹 된다. 그러나 곱셈연산의 수가 1에서 4로 증가하였다.

[0064] [문헌 9]는 $E_b: Y^2+Y=X^3+X+b$ 와 같은 바이어슈트라스 방정식(weierstrass equation)을 이용하여 [문헌 8]이 제안한 공식을 단순화하였다. [문헌 9]는 RPC 친화적인 nT 페어링 연산 알고리즘을 제안하였다. [문헌 9]는 [문헌 8]

이 제안한 연산에서 $(g \leftarrow wx_Q + y_Q + (\frac{(n+1)}{2}w + y_P + b + \epsilon_{(n-1)/2}) + (w + x_Q)s + t)$ 연산을 $(g \leftarrow w\bar{X}_Q + \bar{Y}_Q + \bar{Z}_Q(\frac{(n+1)}{2}w + y_P + b + \epsilon_{(n-1)/2}) + (\bar{Z}_Qw + X_Q)s + \bar{Z}_Qt)$ 연산으로 대체하였다. 이러한 [문헌 9]가 제안한 연산은 [문헌 8]이 제안한 방법에 비하여 연산이 줄었다.

[0065] 3. RVA에 기반으로 한 대응방법

[0066] RPC에 기반으로 한 대응방법은 상대적으로 많은 연산을 필요로 하는 단점이 있다. 이에 따라, [문헌 10]은 RVA에 기반으로 한 대응방법을 제안하였다. 이는 초기값을 마스킹하기 위해 부 채널 분석에 취약한 중간 변수에 랜덤 값을 더하는 것이다. $p=2$ 또는 $p=3$ 인 경우 $GF(P^n)$ 상에서의 덧셈연산은 곱셈연산보다 적은 연산량을 요구한다. 그러나 Shirase 외등이 제안하는 방법은 연산 순간에는 부 채널 분석에 안전하지만, 연산을 완료하는 경우 마스킹이 상쇄되어 부 채널 분석에 취약한 단점이 있다. 이는 마스킹 효과가 상쇄된 값과 연관된 소비전력을 추적하여 부 채널 분석을 수행할 수 있기 때문이다. 아래에서 이에 대한 더욱 자세히 설명하기로 한다.

[0067] [[문헌 10]에서 제안한 RVA에 기반으로 한 대응방법에 대한 보안성 분석]

[0068] 이하에서는 [문헌 10]에서 제안한 RVA에 기반으로 한 대응방법의 세부단계에서의 보안 취약성을 분석하기로 한다. 도 2는 종래의 [문헌 10]에서 제안한 RVA를 기반으로 한 $GF(3^n)$ 상에서의 nT 페어링을 $GF(2^n)$ 상으로 수정하여 나타낸 알고리즘이다.

[0069] 1. 과정 11에서 (x_P+v+1) 연산

[0070] (x_P+v+1) 의 실제 과정은 과정 11에서 $(x_P^2+x_Q+1)=((x_P^2+\lambda^2+1)+(x_Q+\lambda^2+1)+1)$ 이다. 이러한 과정은 아래와 같은 어셈블리 코드들로 실행할 수 있다. 여기서 X, Y, Z 와 같은 어드레스 레지스터들 각각은 $x_P'=(x_P^2+\lambda^2+1)$, $x_Q'=(x_Q+\lambda^2)$, $(x_P' \oplus x_Q')$ 연산의 결과값의 어드레스를 취한다.

[0071] - 어셈블리 코드

```

1. Loop:
2. tst R2
3. breq Exit
4. dec R2
5. R4, X+ // loading ( $x_P^2 + \lambda^2 + 1$ )
6. R6, Y+ // loading ( $x_Q + \lambda^2$ )
7. eor R4, R6 //  $R4 \leftarrow (x_P^2 + x_Q + 1)$ 
8. st Z+, R4 // storing ( $x_P^2 + x_Q + 1$ )
9. jmp Loop
10. Exit:

```

[0073] X, Y, Z는 각각 메모리에서 x_P , x_Q , 그리고 임시적으로 $(x_P + x_Q)$ 의 결과값이 저장되는 주소를 저장하는 주소 레지스터이다.

[0074] 위 코드들에서 x_P 와 x_Q 는 $GF(2^n)$ 상의 원소이므로 각각 메모리 공간에서 $t = \lceil n/WORDSIZE \rceil$ 바이트를 점유한다. 따라서 위의 코드들은 과정 5 및 6과 같이 메모리로부터 x_P 와 x_Q 의 하위 워드에 저장된 값을 레지스터 R4, R6으로 로딩하여 과정 7과 같이 레지스터 R4와 R6에 대해 XOR 연산을 수행한다. 그 다음 과정 8과 같이 과정 7의 결과를 메모리에 저장한다. 이와 같은 과정을 순차적으로 주소의 변위를 변화시키면서 t번 수행한다.

[0075] 과정 5와 6을 수행하는 동안에는 x_P^2 와 x_Q 가 랜덤값 λ^2 에 의해 마스킹되어 메모리로부터 레지스터로 이동되므로 공격자의 공격에 안전하다. 그러나 과정 7에서 λ^2 가 제거된다. 따라서 과정 8은 마스킹이 없는 상태에서 $(x_P^2 + x_Q + 1)$ 연산 결과값이 버스를 경유하여 메모리에 저장된다. 이때, 공격자는 x_P^2 와 연관된 소비전력을 추적하여 부 채널 분석을 수행할 수 있게 된다. 실제로 큰 전력이 데이터가 버스를 경유하여 이동할 때 소비되므로, 공격자들은 수집한 소비전력 추적 이력을 이용하여 부 채널 분석을 수행한다.

[0076] 2. 과정 11에서 $(\Theta + u + T + y_P)$ 연산

[0077] $(\Theta + u + T + y_P)$ 연산에 대한 공격은 처리순서에 따른다. 즉, $(u + y_P)$ 연산이 먼저 이루어지는 경우 부 채널 분석이 가능하다. 다시 말하면, $(u + y_P)$ 의 실제 연산은 $(y_Q + y_P^2 + b) = ((y_Q^2 + \lambda^2 + b + 1) + (y_P^2 + \lambda^2 + 1))$ 이다. 따라서 공격자는 연산결과를 메모리에 저장하는 동안 발산되는 전력 소비를 분석하여 부 채널 분석을 할 수 있다.

[0078] 3. 과정 13에서 $G \leftarrow A?G$ 연산

[0079] $GF(2^{4n})$ 상에서의 곱셈인 $A?G$ 연산을 수행하기 위해서는 A의 값이 레지스터에 로딩되어야 한다. 그러나 $A = ((\Theta + u + T + y_P) + (x_P + v + 1)s + t)$ 연산은 랜덤 값에 의해 마스킹 되지 않은 상태로 저장된다. 따라서 A를 메모리로부터 레지스터로 로딩 하는 동안 A의 값 특히, $(x_P + v + 1)s$ 연산 값에 연관된 소비 전력이 발생한다. 따라서 공격자는 부 채널 분석을 할 수 있게 된다.

[0080] 위에서 언급된 바와 같은 $GF(2^n)$ 상에서의 [문헌 10]에서 제안한 RVA를 기반으로 한 대응방법의 보안 결함은 확장 곱셈연산인 과정 8 이전에 정확한 페어링 연산을 위해 마스킹된 랜덤 값이 제거되기 때문이다.

[0081] [문헌 10]에서 제안한 RVA를 기반으로 한 대응방법은 아래와 같은 시점에서 부 채널 분석에 대응하기 위해 적절한 랜덤 마스킹 값들로 보호되어야 하고, 정확한 페어링 연산을 보장하여야 한다.

[0082] 1. 시점 1: 로딩 데이터가 메모리로부터 레지스터로 이동하는 시점

[0083] 2. 시점 2: 중앙연산장치(CPU)에서 연산이 이루어지는 시점

[0084] 3. 시점 3: 연산 결과가 메모리에 저장되는 시점

- [0085] 이하에서는 본 발명의 일실시예에 따른 $GF(2^n)$ 상의 nT 페어링 연산방법을 도면과 함께 설명하기로 한다.
- [0086] 도 3은 본 발명의 일실시예에 따른 $GF(2^n)$ 상의 nT 페어링 연산방법을 나타낸 흐름도이고, 도 4는 본 발명의 일 실시예에 따른 $GF(2^n)$ 상의 nT 페어링 연산방법을 구체화한 알고리즘이다.
- [0087] 도 3 및 도 4를 참조하면, 우선, 도 4에 도시되는 과정 3 내지 5와 같이, $GF(2^n)$ 상에서 랜덤 값(λ)을 선택하고, 이를 이용하여 제 1내지 제4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)을 생성한다(410). 여기서, 제1 내지 제4 랜덤 값($\lambda_0, \lambda_1, \lambda_2, \lambda_3$)을 생성하는 방법은 랜덤 값(λ)을 제공하거나 랜덤 값(λ)을 그대로 이용하는 방법이 일례이다. 이에 대한 구체적인 방법은 도 4에 도시되는 과정 3과 5와 같다.
- [0088] 그 다음, 초기점인 $P=(x_P, y_P)$ 와 $Q=(x_Q, y_Q)$ 의 좌표들을 $x_P \leftarrow x_P^2 + 1, y_P \leftarrow (y_P + \lambda_1 + \lambda_0)^2 + 1, x_Q \leftarrow (x_Q + \lambda_1), y_Q \leftarrow (y_Q + \lambda_2 + \lambda_0)$ 와 같이 마스킹 한다(420). 이에 대한 구체적인 방법은 도 4에 도시되는 과정 6과 같다.
- [0089] 이어서, 랜덤 값(λ)과 초기점인 $P=(x_P, y_P)$ 와 $Q=(x_Q, y_Q)$ 을 이용하여 초기 파라미터들을 세팅 한다(430). 다시 말하면, 도 4에 도시되는 과정 4와 6 및 7과 같이 초기 파라미터인 w, u, v , 및 θ 을 $w \leftarrow (x_P ? \lambda), w \leftarrow w^2, u \leftarrow y_Q + b + 1, v \leftarrow (x_Q + 1)$, 및 $\theta \leftarrow x_P ? v$ 와 같이 연산하여 세팅한다.
- [0090] 한편, 위와 같은 과정에서 x_P, y_P, x_Q, y_Q 모두들이 마스킹된 상태를 유지함으로써 부 채널 분석을 할 수 없게 한다.
- [0091] 이하에서는 위와 같은 과정에서 부 채널 분석을 할 수 없음을 검증하기 위해, x_P, y_P, x_Q, y_Q 가 $x_P \leftarrow x_P, y_P \leftarrow (y_P + \lambda + \lambda^2), x_Q \leftarrow (x_Q + \lambda^2), y_Q \leftarrow (y_Q + \lambda + \lambda^4)$ 와 같이 마스킹되는 경우, x_P, y_P, x_Q, y_Q 에 대한 $A' \leftarrow (y_P + \theta + u + w) + (x_P + v + 1)s + t$ 연산이 위와 같은 시점 1, 2 및 3에서의 부 채널 분석에 대응할 수 있어야 한다.
- [0092] ($A' \leftarrow (y_P + \theta + u + w) + (x_P + v + 1)s + t$) 연산에서 $x_P \leftarrow x_P, y_P \leftarrow (y_P + \lambda + \lambda^2), x_Q \leftarrow (x_Q + \lambda^2), y_Q \leftarrow (y_Q + \lambda + \lambda^4), w \leftarrow (x_P ? \lambda)$ 이다.
- [0093] 시점 1, 2 및 3에서의 부 채널 분석에 대응할 수 있는 것을 검증하기 위해서는 $A \leftarrow (y_P + \theta + u + w) + (x_P + v + 1)s + t$ 연산을 종료한 후에도 마스킹 값이 유지되는 것을 보여야 한다. 즉, A 가 $(a_0) + (a_1)s + t$ 와 같이 표현되는 경우, $A = (y_P + \theta + u + w) + (x_P + v + 1)s + t$ 가 $A = (a_0 + \lambda) + (a_1 + \lambda^2)s + t$ 의 형태를 가지고 있음을 보여야 한다.
- [0094] 한편, $A' = (y_P + \theta + u + w) + (x_P + v + 1)s + t$ 가 루프 카운터 i 에 따라 아래와 같이 4개의 형태로 표현된다. 여기서, $i \equiv 0, 1, 2, 3 \pmod{4}$ 이다.
1. Case $i \equiv 0 \pmod{4}$.
 $(x_P^{(2i+1)} x_Q + x_P^{(2i+1)} + x_Q + y_P^{(2i+1)} + y_Q + w^{(2i+1)} + b + 1) + (x_P^{(2i+1)} + x_Q + 1)s + t.$
 2. Case $i \equiv 1 \pmod{4}$.
 $(x_P^{(2i+1)} x_Q + y_P^{(2i+1)} + y_Q + w^{(2i+1)} + b) + (x_P^{(2i+1)} + x_Q)s + t.$
 3. Case $i \equiv 2 \pmod{4}$.
 $(x_P^{(2i+1)} x_Q + x_P^{(2i+1)} + x_Q + y_P^{(2i+1)} + y_Q + w^{(2i+1)} + b) + (x_P^{(2i+1)} + x_Q + 1)s + t.$
 4. Case $i \equiv 3 \pmod{4}$.
 $(x_P^{(2i+1)} x_Q + y_P^{(2i+1)} + y_Q + w^{(2i+1)} + b + 1) + (x_P^{(2i+1)} + x_Q)s + t.$
- [0095]
- [0096] 여기서, $y_P^{(2i+1)} \leftarrow (y_P^{(2i+1)} + \lambda^{(2i+1)} + \lambda^{(2i+2)}), x_Q \leftarrow (x_Q + \lambda^{(2i+1)}), x_P^{(2i+1)} \leftarrow x_P^{(2i+1)}, y_Q \leftarrow (y_Q + \lambda^{(2i+1)} + \lambda^{(2i+2)}), w^{(2i+1)} \leftarrow (x_P ?$

$\lambda)^{(2i+1)}$ 이다.

[0097] i 가 0인 경우 첫 번째 형태의 식은 $(x_P^2 \cdot x_Q + x_P^2 + y_P^2 + x_Q + y_Q + \lambda + b + 1) + (x_P^2 + x_Q + \lambda^2 + 1)s + t$ 이고, i 의 원래 식은 $(x_P^2 \cdot x_Q + x_P^2 + y_P^2 + x_Q + y_Q + b + 1) + (x_P^{(2k+1)} + x_Q + 1)s + t$ 이다. i 가 $k(k \equiv 0 \pmod{4})$ 인 경우 첫 번째 형태의 식은 $(x_P^{(2k+1)} \cdot x_Q + x_P^{(2k+1)} + y_P^{(2k+1)} + x_Q + y_Q + \lambda^{(2k)} + b + 1) + (x_P^{(2k+1)} + x_Q + \lambda^{(2k+1)} + 1)s + t$ 이고, i 의 원래 식은 $(x_P^{(2k+1)} \cdot x_Q + x_P^{(2k+1)} + y_P^{(2k+1)} + x_Q + y_Q + b + 1) + (x_P^{(2k+1)} + x_Q + 1)s + t$ 이다. 마지막으로 i 가 $k+4$ 인 경우 첫 번째 형태의 식은 $(x_P^{(2k+9)} \cdot x_Q + x_P^{(2k+9)} + y_P^{(2k+9)} + x_Q + y_Q + \lambda^{(2k+8)} + b + 1) + (x_P^{(2k+9)} + x_Q + \lambda^{(2k+9)} + 1)s + t$ 이고, i 의 원래 식은 $(x_P^{(2k+9)} \cdot x_Q + x_P^{(2k+9)} + y_P^{(2k+9)} + x_Q + y_Q + b + 1) + (x_P^{(2k+9)} + x_Q + 1)s + t$ 이다.

[0098] 따라서 $i \equiv (\text{mod } 4)$ 인 경우 첫 번째 형태의 식은 마스킹 값을 포함하는 정규형태이다. 나머지 형태의 식도 마스킹 값을 포함함을 쉽게 알 수 있을 것이다.

[0099] 즉, 위의 첫 번째 형태의 식은
$$\underbrace{(x_P^{(2i+1)} x_Q + x_P^{(2i+1)})}_D + \underbrace{x_Q + y_P^{(2i+1)}}_E + \underbrace{y_Q + w^{(2i+1)}}_C + \underbrace{b + 1}_F + (x_P^{(2i+1)} + x_Q + 1)s + t.$$
 와 같이 표현될 수 있다.

[0100] 한편, 위에서 E는 마스킹 값을 포함하지 않고 있고, D와 F는 마스킹 값이 곱해진 형태이므로, D, E 및 F는 마스킹 값 포함 여부에 영향을 주지 않는다. 따라서 A, B 및 C에 대한 덧셈 연산의 순서가 바뀌어도 마스킹 값이 제거되지 않는다. 이는 아래와 같은 연산 순서에 따른 연산식과 같이 A, B 및 C에 대한 연산순서가 바뀌어도 마스킹 값이 제거되지 않는 것에 의해 검증되었다.

[0101] [연산 순서에 따른 연산식]

$$\begin{aligned} (A + B) + C &= (x_Q + \lambda^{(2i+1)} + y_P^{(2i+1)} + \lambda^{(2i+1)} + \lambda^{(2i+2)}) + y_Q + \lambda^{(2i)} + \lambda^{(2i+2)}, \\ (B + C) + A &= (y_P^{(2i+1)} + \lambda^{(2i+1)} + \lambda^{(2i+2)} + y_Q + \lambda^{(2i)} + \lambda^{(2i+2)}) + x_Q + \lambda^{(2i+1)}, \\ (A + C) + B &= (x_Q + \lambda^{(2i+1)} + y_Q + \lambda^{(2i)} + \lambda^{(2i+2)}) + y_P^{(2i+1)} + \lambda^{(2i+1)} + \lambda^{(2i+2)}. \end{aligned}$$

[0103] 그 다음, 도 4에서 도시되는 과정 7에 의해 연산된 초기 파라미터 u , v 및 θ 를 이용하여 g_0 , g_1 , g_2 및 g_3 로 이루어진 랜덤 변수(G)를 구한다(440).

[0104] 즉, 도 4에 도시된 과정 8 내지 10과 같이, x_P , y_P 와 초기 파라미터인 u , v 및 θ 를 $(\theta + u + w + y_P) + (x_P + v + 1)s + t$ 에 대입하여 랜덤 변수(G)를 얻고, 이를 다시 $\lambda_0 \cdot G + \lambda_1 + (\lambda_1 \cdot \lambda_0)s$ 에 대입한다. i 의 결과 값을 제공하여 $(g_0) + (g_1)s + (g_2)t + (g_3)st$ 의 형태로 변환하여 g_0 , g_1 , g_2 및 g_3 를 구한다.

[0105] 그 다음, 도 4에 도시된 과정 11과 12와 같이 랜덤 값 및 초기 파라미터들을 업데이트 한다(450).

[0106] 즉, $x_P \leftarrow x_P^4$, $y_P \leftarrow y_P^4$, $T \leftarrow \lambda_0 + \lambda_1$, $\lambda_3 \leftarrow \lambda_1$, $\lambda_0 \leftarrow \lambda_0^4$, $\lambda_1 \leftarrow \lambda_1^4$, $\lambda_2 \leftarrow \lambda_2^4$, $\lambda_3 \leftarrow \lambda_3 + \lambda_1$, $w \leftarrow w^4$, $T \leftarrow T + \lambda_2$, $u \leftarrow u + v + 1 + T$, $v \leftarrow v + 1 + \lambda_3$, $\theta \leftarrow x_P \cdot v$ 와 같은 연산을 수행한다.

[0107] 이어서, 도 4에 도시된 과정 14 및 15와 같이 중간 페어링 연산 값(A)을 구하고 랜덤 변수(G)를 갱신한다(460). 즉, 도 4에 도시된 과정 11과 12를 통해 업데이트된 랜덤 값과 초기 파라미터들을 $(\theta + u + w + y_P) + (x_P + v + 1)s + t$ 에 대입하여 중간 페어링 연산 값(A)을 구한다. 그리고 랜덤 변수(G)를 제공하여 랜덤 값(G)을 갱신한다.

[0108] 그 다음, 중간 페어링 값(A)과 랜덤 변수(G)로 확장체 곱셈 연산을 수행한다(480). 이에 대해서는 후에 더욱 상세히 설명하기로 한다.

[0109] 그 다음, 페어링 연산 회수가 일정한 회수 이상 수행 했는가를 판단한다(490). 즉, 도 4에 도시되는 과정 17과 같이 i 가 $(n-1)/2$ 보다 작은지를 판단한다.

[0110] 이때, 페어링 연산 회수가 일정한 회수 이상이 아니면, 초기 입력 값들을 업데이트 한다(520). 즉, 도 4에 도시

되는 과정 18과 같이 초기 입력좌표와 초기 파라미터 및 제 1내지 제 4 랜덤 값들을 업데이트한다. 그 다음, 앞서 설명된 480 단계 및 그 이후의 단계를 수행 한다.

[0111] 반면, 페어링 연산 회수가 일정한 회수 이상이면, 중간 페어링 값(A)에 $(x_p^2 + v + 1) + s$ 을 더하고, 이의 결과와 이제까지의 랜덤 변수(G)를 이용하여 도 4에 도시된 과정 23과 같이 마지막 확장체 곱셈연산을 수행한다(500).

[0112] 그 다음, 도 4에 도시된 과정 24와 같이 마지막 확장체 곱셈연산의 결과 값(G)을 $G \leftarrow G^M, M = (2^{2n} - 1)(2^n + 2^{\frac{n+1}{2}} + 1)$ 와 같이 곱 제공하면서(510) 종료한다.

[0113] 이하에서는 본 발명의 일실시예에 따른 $GF(2^n)$ 상의 nT 페어링 연산방법의 보정을 포함한 확장체 곱셈 연산방법에 대해 설명하기로 한다.

[0114] 본 발명의 일실시예에 따른 확장체 곱셈 연산의 개요를 설명하기로 한다.

[0115] 만일 원래 형태가 $(a_0) + (a_1)s + t$ 이라고 가정하면, $(\hat{A} \leftarrow (y_p + \theta + u + w) + (x_p + v + 1)s + t)$ 을 계산한 결과로서, $\hat{A}$ 는 $(a_0 + \lambda_0) + (a_1 + \lambda_1)s + t$ 의 형태를 갖는다.

[0116] 종래의 RVA 방법에서, 마스킹 값들은 정확한 페어링 값을 계산하기 위해 A?G를 계산하기 전에 제거되며, 이로 인해 보안 문제가 발생한다.

[0117] 따라서, 본 발명의 일실시예에 따른 확장체 곱셈 연산은 $GF(2^n)$ 상에서의 [문헌 10]에서 제안한 RVA를 기반으로 한 대응방법과 다르게 확장체 곱셈 연산 A?G 이전에 보정을 위해 마스킹을 제거하는 보정을 하지 않고, 이를 확장체 곱셈연산 중에 수행한다. 이를 위해 $\hat{A} \cdot G$ 연산과 A?G 연산이 동일하도록, 카라슈바(karatsuba) 방법을 따른 아래의 수학적식 8과 같이 확장체 곱셈 연산을 수행한다.

수학적식 8

[0118]
$$H \leftarrow A \cdot G$$

[0119] 여기서, H는 $h_0 + h_1s + h_2t + h_3st$ 이고, A는 $a_0 + a_1s + t$ 이고, G는 $g_0 + g_1s + g_2t + g_3st$ 이다. 이때, h_0, h_1, h_2 및 h_3 는

$$h_0 \leftarrow a_0g_0 + a_1g_1 + g_3,$$

$$h_1 \leftarrow (a_0 + a_1)(g_0 + g_1) + a_0g_0 + g_2 + g_3,$$

$$h_2 \leftarrow (g_0 + g_2)a_0 + (g_1 + g_3)a_1 + a_0g_0 + a_1g_1 + g_0 + g_2,$$

[0120]
$$h_3 \leftarrow (g_0 + g_1 + g_2 + g_3)(a_0 + a_1) + (g_0 + g_2)a_0 + (g_0 + g_1)(a_0 + a_1) + a_0g_0 + g_1 + g_3.$$

[0121] 와 같은 연산에 의해 얻어진다.

[0122] 이러한 연산 방법은 $GF(2^n)$ 에서 6M이 필요하고, 종래의 연산 방법에서는 8M이 필요하다.

[0123] 본 발명의 일 실시예에 따르면, G에 포함된 g_0, g_1, g_2 및 g_3 이 랜덤 변수라고 하면, g_0, g_1, g_2 및 g_3 을 일종의 랜덤 마스킹 값으로 사용한다.

[0124] 한편, 본 발명의 일실시예에 따른 $GF(2^n)$ 상의 nT 페어링 연산방법에서 A가 $(a_0 + \lambda_0) + (a_1 + \lambda_1)s + t$ 와 같은 형태를 갖고 있으므로, λ_0, λ_1 을 g_0, g_1, g_2 및 g_3 를 이용하여 적절하게 변경한다. 즉, $(a_0 + \lambda_0)$ 와 $(a_1 + \lambda_1)$ 을 $(a_0 + g_1), (a_1 + g_0), (a_0 + a_1 + g_0), (a_0 + g_2 + g_3), (a_1 + g_2),$ 및 $(a_0 + a_1 + g_2)$ 중 하나로 변경한다. 이에 따라, h_0, h_1, h_2 및 h_3 에 대한 연산은

$$\begin{aligned} h_0 &\leftarrow (a_0 + g_1)g_0 + (a_1 + g_0)g_1 + g_3, \\ h_1 &\leftarrow (a_0 + a_1 + g_0)(g_0 + g_1) + (a_0 + g_1)g_0 + g_2 + g_3 + g_0^2, \\ h_2 &\leftarrow (a_0 + g_2 + g_3)g_2 + (a_1 + g_2)g_3 + g_0 + g_2 + g_2^2, \\ h_3 &\leftarrow (a_0 + a_1 + g_2)(g_2 + g_3) + (a_0 + g_2 + g_3)g_2 + g_1 + g_3. \end{aligned}$$

[0125]

[0126]

와 같게 된다.

[0127]

이제는 본 발명의 일실시예에 따른 $GF(2^n)$ 상의 nT 페어링 연산방법의 보정을 포함한 확장체 곱셈 연산방법의 구체적인 방법에 대해 도면과 함께 설명하기로 한다.

[0128]

도 5를 참조하면, 우선, 도 6에 도시되는 과정 1과 같이, $(f_0 + \lambda_0) + (f_1 + \lambda_1)s + t$ 와 같은 중간 페어링 결과 값(F)과 $g_0 + g_1s + g_2t + g_3st$ 와 같은 랜덤 변수(G) 및 랜덤 값(λ_0, λ_1)을 요구한다(610).

[0129]

그 다음, 도 6에 도시되는 과정 3 내지 11과 같이 중간 페어링 결과 값(F)의 랜덤 값(λ_0, λ_1)을 업데이트한다(620).

[0130]

마지막으로, 랜덤 값(λ_0, λ_1)이 업데이트된 중간 페어링 결과 값(F)과 중간 랜덤변수(G)를 확장체 곱셈 연산한다(630).

[0131]

본 발명의 일실시예에 따른 본 발명의 일실시예에 따른 $GF(2^n)$ 상의 nT 페어링 연산방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 본 발명을 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다. 상기된 하드웨어 장치는 본 발명의 동작을 수행하기 위해 하나 이상의 소프트웨어 모듈로서 작동하도록 구성될 수 있으며, 그 역도 마찬가지이다.

[0132]

이상과 같이 본 발명에서는 구체적인 구성 요소 등과 같은 특정 사항들과 한정된 실시 예 및 도면에 의해 설명되었으나 이는 본 발명의 보다 전반적인 이해를 돕기 위해서 제공된 것일 뿐, 본 발명은 상기의 실시 예에 한정되는 것은 아니며, 본 발명이 속하는 분야에서 통상적인 지식을 가진 자라면 이러한 기재로부터 다양한 수정 및 변형이 가능하다. 따라서, 본 발명의 사상은 설명된 실시 예에 국한되어 정해져서는 아니되며, 후술하는 특허청구범위뿐만 아니라 이 특허청구범위와 균등하거나 등가적 변형이 있는 모든 것들은 본 발명 사상의 범주에 속한다고 할 것이다.

산업상 이용가능성

[0133]

nT 페어링 알고리즘이 센서모트, 스마트폰 등의 저사양의 통신기기에서 사용될 경우에 본 발명에서 제안한 DPA 방법을 적용할 경우, 대응방법으로 인한 부하가 크지 않기 때문에 효율적으로 동작가능하다. 뿐만 아니라 본 발명에서 제안하는 DPA 대응방법의 부하는 기존의 대응방법과 달리 nT 페어링 알고리즘의 연산량을 크게 증가시키지 않기 때문에 저전력/저사양의 장비에서 사용 가능하다.

도면

도면1

Algorithm 1 ηT Pairing Algorithm over $GF(2^n)$ without a DPA Countermeasure

```

1: Require  $P=(x_P, y_P), Q=(x_Q, y_Q)$ 
2: Ensure  $C=\eta T(P, Q)$ 
3:  $C \leftarrow 1$ .
4:  $x_P \leftarrow x_P^2+1, y_P \leftarrow y_P^2+1$ . (2S)
5:  $u \leftarrow y_Q + b + 1, v \leftarrow x_Q + 1, \theta \leftarrow x_P \cdot v$ . (1M)
6: for from  $i \leftarrow 0$  to  $\frac{n-1}{2}$  do
7:    $A \leftarrow (y_P + \theta + u) + (x_P + v + 1)s + t$ .
8:    $C \leftarrow C^2$ . (4S)
9:    $C \leftarrow C \cdot A$ . (6M)
10:  if  $i < \frac{n-1}{2}$  then
11:     $x_P \leftarrow x_P^4, y_P \leftarrow y_P^4$ . (4S)
12:     $u \leftarrow u + v + 1, v \leftarrow v + 1$ .
13:     $\theta \leftarrow x_P \cdot v$ . (1M)
14:  end if
15: end for
16:  $A \leftarrow A + (x_P^2 + v + 1)s$ . (1S)
17:  $C \leftarrow C \cdot A$ . (6M)
18:  $C \leftarrow C^M$ , where  $M=(2^{2n}-1)(2^n+2^{\frac{n+1}{2}}+1)$ .
19: return  $C$ .
```

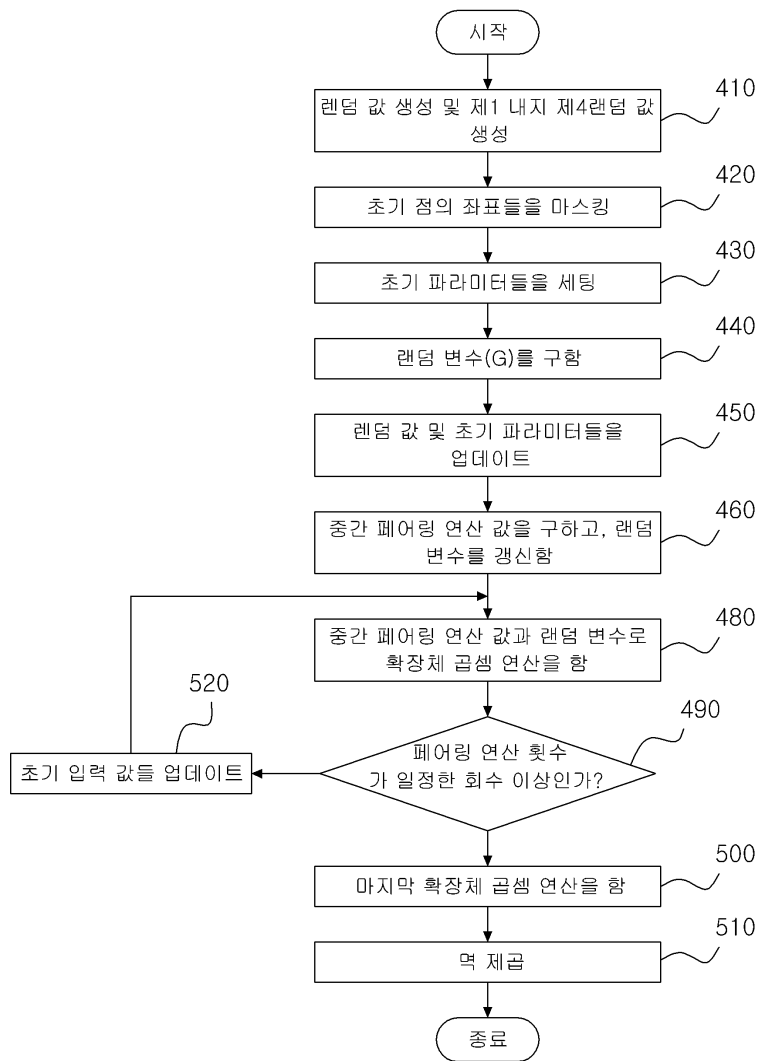
도면2

Algorithm 2 Modified Shirase's ηT Pairing Algorithm for $GF(2^n)$

```

1: Require  $P=(x_P, y_P), Q=(x_Q, y_Q)$ 
2: Ensure  $G=\eta T(P, Q)$ 
3: Prepare random values  $\lambda, \lambda'$  in  $GF(2^n)$ .
4:  $G \leftarrow \lambda'$ .
5:  $\lambda_0 \leftarrow \lambda, \lambda_1 \leftarrow \lambda^2, a \leftarrow 1$ . (1S)
6:  $w \leftarrow (x_P \cdot \lambda)$ . (1M)
7:  $x_P \leftarrow x_P + \lambda_0, y_P \leftarrow y_P + \lambda_0, x_Q \leftarrow x_Q + \lambda_1, y_Q \leftarrow y_Q + \lambda_1$ .
8:  $x_P \leftarrow (x_P^2 + 1), y_P \leftarrow (y_P^2 + 1), w \leftarrow w^2, \lambda_0 \leftarrow \lambda_1, \lambda_1 \leftarrow \lambda_1^2, T \leftarrow (w + \lambda_1 + x_Q \cdot \lambda_0)$ . (1M+4S)
9:  $u \leftarrow y_Q + b + 1, v \leftarrow (x_Q + 1), \theta \leftarrow x_P \cdot v$ . (1M)
10: for from  $i \leftarrow 0$  to  $\frac{n-1}{2}$  do
11:    $A \leftarrow (\theta + u + T + y_P) + (x_P + v + 1)s + t$ .
12:    $G \leftarrow G^2$ . (4S)
13:    $G \leftarrow A \cdot G$ . (6M)
14:   if  $i < \frac{n-1}{2}$  then
15:      $x_P \leftarrow x_P^4, y_P \leftarrow y_P^4, w \leftarrow w^4, v \leftarrow v + \lambda_0 + 1, \lambda_0 \leftarrow \lambda_0^4, \lambda_1 \leftarrow \lambda_1^4, T \leftarrow (\lambda_0 \cdot a + \lambda_1 + w + x_Q \cdot \lambda_0), a \leftarrow a + 1$ . (1M+10S)
16:      $u \leftarrow u + v + \lambda_0 + 1, v \leftarrow v + \lambda_0, \theta \leftarrow x_P \cdot v$ . (1M)
17:   end if
18: end for
19:  $A \leftarrow A + (x_P^2 + v + 1)s$ . (1S)
20:  $G \leftarrow G \cdot A$ . (6M)
21:  $G \leftarrow G^M, M = (2^{2n} - 1)(2^n + 2^{\frac{n+1}{2}} + 1)$ .
22: return  $G$ .
```

도면3



도면4

Algorithm 3 ηT Pairing Algorithm over $GF(2^n)$ with the Proposed DPA Countermeasure

```

1: Require  $P=(x_P, y_P), Q=(x_Q, y_Q)$ 
2: Ensure  $G=\eta T(P, Q)$ 
3: Prepare random values  $\lambda$  in  $GF(2^n)$ .
4:  $w \leftarrow (x_P \cdot \lambda)$ . // (1M)
5:  $\lambda_0 \leftarrow \lambda, \lambda_1 \leftarrow \lambda_0^2, \lambda_2 \leftarrow \lambda_1^2, \lambda_3 \leftarrow \lambda_1$ . // (2S)
6:  $x_P \leftarrow x_P^2 + 1, y_P \leftarrow (y_P + \lambda_1 + \lambda_0)^2 + 1, w \leftarrow w^2, x_Q \leftarrow (x_Q + \lambda_1), y_Q \leftarrow$   

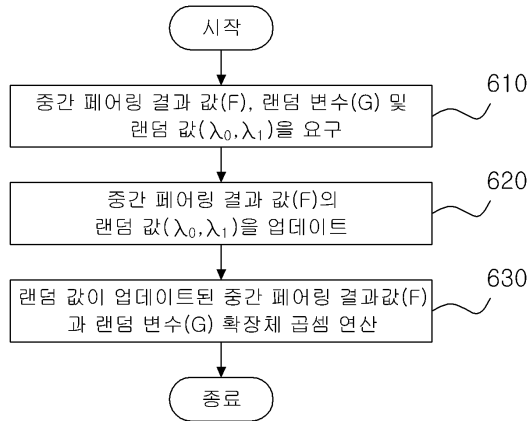
 $(y_Q + \lambda_2 + \lambda_0)$ . // (3S)
7:  $u \leftarrow y_Q + b + 1, v \leftarrow (x_Q + 1), \theta \leftarrow x_P \cdot v$ . // (1M)
8:  $G \leftarrow (\theta + u + w + y_P) + (x_P + v + 1)s + t$ .
9:  $G \leftarrow \lambda_0 \cdot G + \lambda_1 + (\lambda_1 \cdot \lambda_0)s$ . // (3M)
10:  $G \leftarrow G^2$ . // (4S)
11:  $x_P \leftarrow x_P^4, y_P \leftarrow y_P^4, T \leftarrow \lambda_0 + \lambda_1, \lambda_3 \leftarrow \lambda_1, \lambda_0 \leftarrow \lambda_0^4, \lambda_1 \leftarrow \lambda_1^4, \lambda_2 \leftarrow \lambda_2^4,$   

 $\lambda_3 \leftarrow \lambda_3 + \lambda_1, w \leftarrow w^4, T \leftarrow T + \lambda_2$ . // (12S)
12:  $u \leftarrow u + v + 1 + T, v \leftarrow v + 1 + \lambda_3, \theta \leftarrow x_P \cdot v$ . // (1M)
13: for from  $i \leftarrow 1$  to  $\frac{n-1}{2}$  do
14:    $A \leftarrow (\theta + u + w + y_P) + (x_P + v + 1)s + t$ .
15:    $G \leftarrow G^2$ . // (4S)
16:    $G \leftarrow A \cdot G$ . // Call the Algorithm 4 with (6M+2S).
17:   if  $i < \frac{n-1}{2}$  then
18:      $x_P \leftarrow x_P^4, y_P \leftarrow y_P^4, T \leftarrow T^4, \lambda_3 \leftarrow \lambda_1, \lambda_0 \leftarrow \lambda_0^4, \lambda_1 \leftarrow \lambda_1^4, \lambda_2 \leftarrow \lambda_2^4,$   

 $\lambda_3 \leftarrow \lambda_3 + \lambda_1, w \leftarrow w^4$ . // (14S)
19:      $u \leftarrow u + v + 1 + T, v \leftarrow v + 1 + \lambda_3, \theta \leftarrow x_P \cdot v$ . // (1M)
20:   end if
21: end for
22:  $A \leftarrow A + (x_P^2 + v + 1)s$ . // (1S)
23:  $G \leftarrow G \cdot A$ . // (6M+2S)
24:  $G \leftarrow G^M, M = (2^{2n} - 1)(2^n + 2^{\frac{n+1}{2}} + 1)$ .
25: return  $G$ .

```

도면5



도면6

Algorithm 4 Multiplication Algorithm over $GF(2^{4n})$ with Correction Mechanism

```

1: Require  $F = (f_0 + \lambda_0) + (f_1 + \lambda_1)s + t, G = g_0 + g_1s + g_2t + g_3st, \lambda_0, \lambda_1$ .
2: Ensure  $H = h_0 + h_1s + h_2t + h_3st = F \cdot G$ .
3:  $T_0 \leftarrow (\lambda_0 + g_1), T_1 \leftarrow (\lambda_1 + g_0), A \leftarrow (f_0 + \lambda_0), B \leftarrow g_0, C \leftarrow (f_1 + \lambda_1),$   

 $D \leftarrow g_1$ .
4:  $A \leftarrow A + T_0, C \leftarrow C + T_1$ .
5:  $T_2 \leftarrow A \cdot B, h_0 \leftarrow C \cdot D, h_0 \leftarrow h_0 + T_2 + g_3$ . // (2M)
6:  $C \leftarrow C + f_0, D \leftarrow D + g_0$ .
7:  $h_1 \leftarrow C \cdot D, h_1 \leftarrow h_1 + T_2 + g_2 + g_3, T_2 \leftarrow g_0^2, h_1 \leftarrow h_1 + T_2$ . // (1M+1S)
8:  $T_0 \leftarrow g_0 + g_2, T_1 \leftarrow B + g_1, A \leftarrow A + T_1, B \leftarrow g_2 + g_3, C \leftarrow C + T_0, D \leftarrow$   

 $g_2$ .
9:  $T_2 \leftarrow A \cdot D, h_3 \leftarrow B \cdot C, h_3 \leftarrow h_3 + T_2 + g_1 + g_3$ . // (2M)
10:  $B \leftarrow g_3, C \leftarrow C + f_0$ .
11:  $h_2 \leftarrow B \cdot C, h_2 \leftarrow h_2 + T_2 + T_0, T_2 \leftarrow g_2^2, h_2 \leftarrow h_2 + T_2$ . // (1M+1S)
12: return  $H$ .

```