Efficient and Side-Channel Resistant Design of High-Security Ed448 on ARM Cortex-M4

Mila Anastasova[1], Mojtaba Bisheh-Niasar[1], Hwajeong Seo[2], Reza Azarderakhsh[1], *Member*, *IEEE*, and Mehran Mozaffari Kermani[3], *Senior Member*, *IEEE*

[1]Florida Atlantic University, Boca Raton, FL
[2]Hansung University, Seoul 136-792, Korea
[3]University of South Florida, Tampa

Abstract & Introduction

The most extensively used classical Elliptic Curve Cryptography (ECC) primitives, suitable for both high-end and low-end devices, need constant timing, power consumption, and memory needs enhancements. In this work, we present the first implementation of the Edwards Curve Digital Signature Algorithm (EdDSA) based on the Ed448 targeting the ARM Cortex-M4-based STM32F407VG microcontroller, which forms a large part of the Internet of Things (IoT) world. We optimize the high-level group operations by implementing the efficient scalar multiplication over the Ed448 isogenous map to reduce the computation complexity, providing a side-channel analysis (SCA) and fault attack protected design. Our optimized architecture performs a signature and verification in 39.88ms and 51.54ms, respectively, where SCA protection can be achieved at less than 6.4% cost of performance overhead.

Background

The Edwards-Curve Digital Signature Algorithm (EdDSA) is defined over Ed448 in [2], where the points satisfying the equation $Ed/\mathbb{F}_p : ax^2 + y^2 = 1 + dx^2y^2$ are laying on the twisted Edwards curve over a finite field, defined as $\mathbb{F}_p$ with $p = 2^{448} - 2^{224} - 1$ and $d = -39081$, where $P = (x, y)$ and $x, y \in \mathbb{F}_p$.

Public Parameters: $p = 2^{448} - 2^{224} - 1$, $E/\mathbb{F}_p = ax^2 + y^2 = 1 + dx^2y^2$, G

Alice		**Bob**
Key Generation **Input:** $seed$ **Output:** sk_A, pk_A		
1. $sk_A \in_R^{seed} \mathbb{Z}/\mathbb{F}_p$ 2. $(p, s) \leftarrow H(sk_A)$ 3. $pk_A \leftarrow [s] \cdot G$		
Signing **Input:** p, sk_A, M **Output:** $sign \equiv R\|\|S$	$sign$ $\longrightarrow$	**Verifying** **Input:** $pk_A, M, sign$ **Output:** $true/fasle$
1. $r \leftarrow (H(p\|\|M))(modL)$ 2. $R \leftarrow [r] \cdot G$ 3. $k \leftarrow (H(R\|\|pk_A\|\|M))(modL)$ 4. $S \leftarrow (r + k * s)(modL)$		1. $k \leftarrow H(R\|\|pk_A\|\|M)$
Return $R\|\|S$		**Return** $[S] \cdot G == R + [k] \cdot pk_A$

Fig. 1: Ed448 algorithm [2]. H denotes $SHAKE256$, L - the curve order, and G - the base point.

Graphical representation of the Ed448 DSA protocol, executed by both computational parties, is shown in Figure 1. The key generation procedure receives a seed value and outputs a key pair sk_A and pk_A. The signing function receives the value of the secret key sk_A along with a message value M and returns the signature, composed by R and S. Finally, the verification subroutine receives the public key pk_A, the message M, and the signature of the message, to verify the authenticity of the signing party.

Platform & Arithmetic Optimizations

Multi-precision Multiplication: We propose Refined-Operand Caching (R-OC) technique, illustrated in Figure 2, where the size of the rows is increased from 3 to 4 by increasing the register utilization. According to Figure 2, different strategies are implemented for the beginning (marked with light grey color), the middle, and the end (marked in blue color) of each row. Due to the use of `UMULL` and `UMALL` instructions (marked with black and white dots/rectangles, respectively), the zero initialization of the registers is omitted.

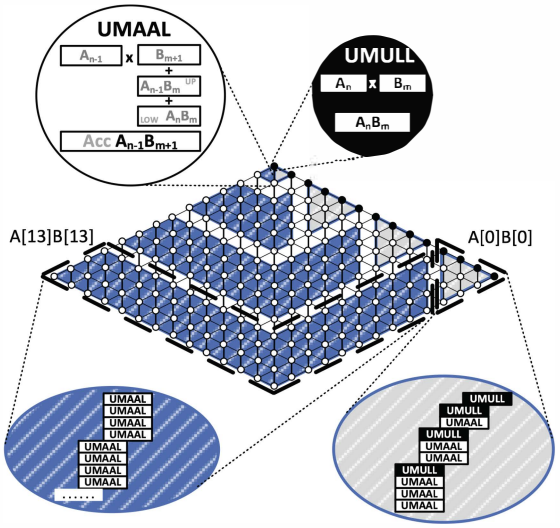

Fig. 2: Rhombus representation of the multiplication strategy.

Our design Table 1 is more than 100 CC faster for the $\mathbb{F}_p$ multiplication, $1.5\times$ more efficient for point addition and point doubling, and almost $2\times$ better for point multiplication compared to counterparts.

Ref.	Timing [cc]						
	Curve448/Ed448 operations						
	$\mathbb{F}_p$				Group		
	Add	Sub	Multiply	Invert	Add	Double	Multiply
Seo et al. [3]	164	161	821	363,485	6,566	6,567	6,218,135
This work [C]	337	350	2,962	1,369,7543	34,075(total)		15,200,3398
This work [ASM]	139	137	705	325,997	8,465(total)		3,703,755
Ref.	**Ed448 Specific operations**						
	Mod l	y-Recovery	Shake256	Decode	PointMultiply		
This work [C]	745,082	41,119	13,966	2,681,880	12,558,965		
This work [ASM]	-	10,581	-	643,611	3,503,308		

Tab. 1: Finite field operations for Curve448/Ed448 targeting ARMv7-M.

Performance & SCA protection

Table 2 reports the performance results of, to the best of our knowledge, the first implementation of Ed448 DSA on the Cortex-M4 target processor, using the STM32F407VG microcontroller. The target platform offers operation mode @24MHz and @168MHz, where the former shows precise latency results ensuring zero wait state, and the latter simulates a real-world scenario.

Work	Platform	Freq. [MHz]	KeyGen [KCCs]	KeyGen [ms]	Sign [KCCs]	Sign [ms]	Verify [KCCs]	Verify [ms]
Ed25519 [1]	Cortex-M4	84	389.5	4.64	543.7	6.47	1,331.4	15.85
Ed448 [3]	AVR	32	103,229	3,225.9	-	-	-	-
Ed448 [3]	MSP	25	73,478	2,939.1	-	-	-	-
This work [C]	Cortex-M4	24	11,326	471.91	13,828	576.16	22,062	919.25
		168	11,694	69.60	14,198	84.51	22,730	135.29
This work [ASM]	Cortex-M4	24	4,069	169.54	6,571	273.79	8,452	352.16
		168	4,195	**24.97**	6,699	**39.87**	8,659	**51.54**

Tab. 2: Ed25519/Ed448 DSA performance on IoT platforms.

Scalar Blinding: *hide* the value of the secret scalar by removing the point swapping data dependency during point multiplication. **Point Randomization**: obtain the randomized base point $G_{rand} = (\lambda \cdot x, \lambda)$ and $G_{rand} = (\lambda \cdot X, \lambda \cdot Y, \lambda)$ in affine and projective coordinates, respectively.

Scheme	Timing [ms]			Memory
	KeyGen	Sign	Verify	[B]
No SCA	24.97	39.88	51.54	3,612
Scalar Blinding	25.42	40.33	51.54	3,612
Point Randomization	27.19	42.10	51.54	3,612
Both Countermeasures	27.69	42.60	51.54	3,612

Tab. 3: Ed448 DSA side-channel protected design performance on STM32F407VG.

Table 3 reports the additional latency when deploying SCA countermeasures. We note that when applying point randomization and scalar blinding, the added timing is around 3ms for both - the key generation and the signing function.

Conclusions

In this work, we presented the first implementation of the Ed448 DSA protocol targeting the highly demanded low-end device ARM-based Cortex-M4. We evaluate the performance results based on pure C code implementation design and target-specific assembly language. Finally, we provide side-channel and fault attack protected design and report the performance. We keep evaluating our proposed countermeasures using the TVLA technique over Cortex-M4 as future work.

References

[1] Hayato Fujii **and** Diego F Aranha. **?**Curve25519 for the Cortex-M4 and beyond**?** **in**(2017): **pages** 109–127.

[2] Simon Josefsson **and** Ilari Liusvaara. *Edwards-Curve Digital Signature Algorithm (EdDSA)*. RFC 8032. **january** 2017. DOI: 10.17487/RFC8032. URL: https://rfc-editor.org/rfc/rfc8032.txt.

[3] Hwajeong Seo **and** Reza Azarderakhsh. **?**Curve448 on 32-Bit ARM Cortex-M4**?** **in***International Conference on Information Security and Cryptology*: Springer. 2020, **pages** 125–139.