On Fast Calculation of Addition Chains for Isogeny-Based Cryptography

Brian Koziel (Texas Instruments) [corresponding author, kozielbrian@gmail.com], Reza Azarderakhsh (Florida Atlantic University), David Jao (U Waterloo), and Mehran Mozaffari-Kermani (Rochester Institute of Technology)

Inscrypt 2016
Beijing, China

Outline

Introduction

- Isogeny-based Cryptography utilizes isogenies between elliptic curves.
  - Can be visualized as moving from elliptic curve to elliptic curve
- Supersingular Isogeny Diffie-Hellman (SIDH), originally proposed by David Jao and Luca De Feo, is a public-key cryptography key exchange that relies on the difficulty to compute isogenies between supersingular elliptic curves.
  - This has been shown to be difficult, even for quantum computers.
  - SIDH features smooth isogeny primes of the form $p = \ell_A^{e_A} \ell_B^{e_B} f \pm 1$
- This work analyzes fast addition chains for smooth isogeny primes
- Currently, addition chains are utilized for inversions in SIDH as well as square roots in SIDH compression.

Notations used in this Work

Table: Notations used in this paper

Notation	Definition
$\mathbb{Z}$	The set of integers
$\mathbb{F}_{p^n}$	A finite field of size p^n
m	Power of 2 to represent families of special primes
k	Iterating over k bits at a time (as in k-ary method)
c	Optimal power of 2 for use in Hybrid Windowing Method
I, M, S, A	Inversion, Multiplication, Squaring, and Addition in $\mathbb{F}_p$
$\tilde{I}, \tilde{M}, \tilde{S}, \tilde{A}$	Inversion, Multiplication, Squaring, and Addition in $\mathbb{F}_{p^2}$

Addition Chain Introduction

Definition

An addition chain is a sequence of integers $(a_0, a_1, \ldots, a_r)$ with $a_0 = 1$ and $a_r = n$, such that $a_i = a_j + a_k$ for some possibly equal $j, k < i$.

Definition

An addition chain is optimal if its length is the smallest among all possible addition chains.

Definition

A Brauer chain is an addition chain that always uses the previous value for the next one. In other words, it is a sequence of integers $(a_0, a_1, \ldots, a_r)$ with $a_0 = 1$ and $a_r = n$, such that $a_i = a_j + a_{i-1}$.

- The k-ary method is a Brauer chain that iterates k bits at a time with a square-and-multiply approach, i.e. square k times and then multiply by a specific window.

Windowing Method

- Optimize k-ary method by breaking exponential into specific windows up to a maximum of k bits.
  - Instead of generating all possible windows (as is done in k-ary method), generate only the needed ones
- After generating the necessary windows, an addition sequence must be completed so that these windows can be reached.
- Several different methods to generate[1]:
  - Lucas chains
  - Division
  - Approximation
  - ...
- We can compare the efficiency of addition chains by evaluating:
  - Number of squarings and multiplications, (Assume $S = 0.8M$)
  - Number of temporary registers to store

[1] Bos, J., Coster, M.: *Addition Chain Heuristics*. CRYPTO' 89 Proceedings: 400-407.

Finite Field Inversion and Square Root

- General arithmetic in SIDH is over $\mathbb{F}_{p^2}$
- Inversion over $\mathbb{F}_{p^2}$ can be done with an inversion in $\mathbb{F}_p$.
- Square root over $\mathbb{F}_{p^2}$ is more complicated, depending on p mod 4[1]
  - 1 mod 4 requires an $\mathbb{F}_{p^2}$ exponentation by $\frac{p-1}{4}$, $\frac{p-1}{2}$, and p AND an expensive 1 mod 4 square root operation in $\mathbb{F}_p$
  - 3 mod 4 requires an $\mathbb{F}_{p^2}$ exponentiation by $\frac{p-3}{4}$, $\frac{p-1}{2}$, and p

[1] Adj, G., Rodriguez-Henriquez, F.: *Square Root Computation Over Even Extension Fields*. Cryptology ePrint Archive, Report 2012/685.

Overview of SIDH

- Proposed by David Jao and Luca De Feo[1]
- Public Parameters
  - Smooth Isogeny Prime - $p = \ell_A^{e_A} \ell_B^{e_B} f \pm 1$, where ℓ_A and ℓ_B are small primes, e_A and e_B are positive integers, and f is a small cofactor to make the number prime
  - Starting Supersingular Elliptic Curve, $E_0/\mathbb{F}_{p^2}$
  - Torsion bases $\{P_A, Q_A\}$ and $\{P_B, Q_B\}$ over $E_0[\ell_A^{e_A}]$ and $E_0[\ell_B^{e_B}]$, respectively
- Classical and quantum security is approximately $\sqrt[4]{p}$ and $\sqrt[6]{p}$, respectively.

[1] Jao, D., De Feo, L.: *Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies*. PQCrypto 2011: 19-34.

Overview of SIDH

- Each round is broken into computing a double point multiplication, $R = mP + nQ$, where m and n are secret scalars, and using R as a secret kernel for an isogeny, $\phi : E \to E/\langle R \rangle$.
  - $\phi_A : E \to E/\langle m_A P_A + n_A Q_A \rangle = E_A$ for Alice and
    $\phi_B : E \to E/\langle m_B P_B + n_B Q_B \rangle = E_B$ for Bob
- After the first round, Alice sends $\{E_A, \phi_A(P_B), \phi_A(Q_B)\}$ and Bob sends $\{E_B, \phi_B(P_A), \phi_B(Q_A)\}$
- After the second round, Alice and Bob have isomorphic curves, so the j-invariant can be used as a shared secret key.
  - $\phi'_A : E_B \to E_B/\langle m_A \phi_B(P_A) + n_A \phi_B(Q_A) \rangle = E_{AB}$ for Alice and
    $\phi'_B : E_A \to E_A/\langle m_B \phi_A(P_B) + n_B \phi_A(Q_B) \rangle = E_{BA}$ for Bob

SIDH Protocol

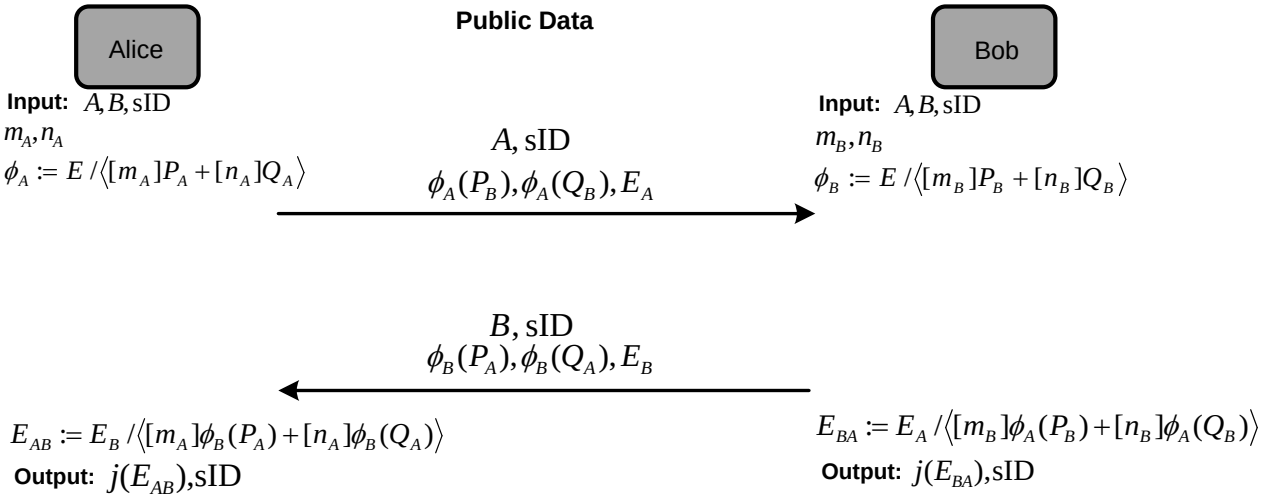


Figure: SIDH Protocol

SIDH Key Compression

- Proposed in [1] and improved in [2]
- Idea: compress tuple $\{E_A, \phi_A(P_B), \phi_A(Q_B)\}$ to $\{j(E_A), m_1, n_1, m_2, n_2\}$ (in Alice's perspective)
- Compression:
  1. Deterministically create a torsion basis from $j(E_A)$ over $E_A[\ell_A^{e_A}]$, S_A and T_A
  2. Compute generalized discrete log (easy for smooth isogeny primes), $\phi_A(P_B) = m_1 S_A + n_1 T_A$ and $\phi_A(Q_B) = m_2 S_A + n_2 T_A$
  3. Send $\{j(E_A), m_1, n_1, m_2, n_2\}$
- Decompression:
  1. Deterministically create a torsion basis from $j(E_A)$ over $E_A[\ell_A^{e_A}]$, S_A and T_A
  2. Perform a double point multiplication, $\phi_A(P_B) = m_1 S_A + n_1 T_A$ and $\phi_A(Q_B) = m_2 S_A + n_2 T_A$
- Reduces size of exchanged public keys to about $\frac{7}{2}\log_2 p$ bits

[1] Azarderakhsh, R., Jao, D., Kalach, K., Koziel, B., Leonardi, C.: *Key Compression for Isogeny-Based Cryptosystems*.AsiaPKC '16: 1-10.
[2] Costello, C., Jao, D., Longa, P., Naehrig, M, Renes, J., Urbanik, D..: *Efficient Compression of SIDH Public Keys*. Cryptology ePrint Archive, Report 2016/963

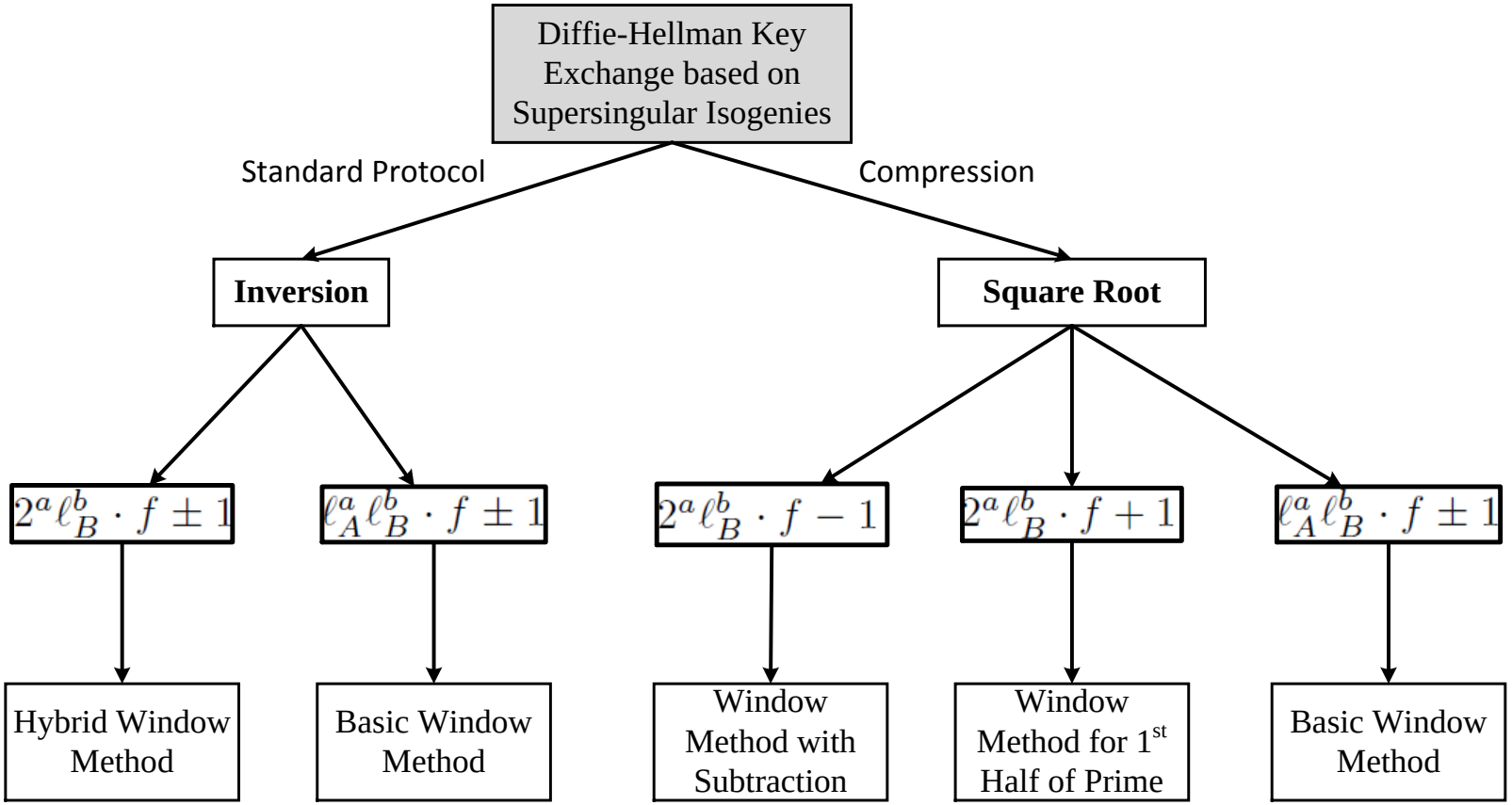
SIDH Applications for Addition Chains

- In original protocol by Jao and De Feo[1], each large-degree isogeny computation required $O(e_A)$ or $O(e_B)$ inversions per round, for SIDH prime $p = \ell_A^{e_A} \ell_B^{e_B} f \pm 1$
- Costello, Longa, and Naehrig[2] presented "projective" isogeny formulas which only require 1 inversion per round
- SIDH key compression requires a deterministic number of square root operations to create a torsion basis
  - Algorithm cycles through multiple possible points (x, y), where $y = \sqrt{x^3 + ax + b}$

[1] Jao, D., De Feo, L.: *Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies*. PQCrypto 2011: 19-34.
[2] Costello, C., Longa, P., Naehrig, M.: *Efficient Algorithms for Supersingular Isogeny Diffie-Hellman*. CRYPTO 2016: 572-601

Families of Smooth Isogeny Primes

- Three general shapes to account for: $2^{e_A}\ell_B^{e_B}f-1$, $2^{e_A}\ell_B^{e_B}f+1$, and general smooth isogeny primes



$2^{e_A}\ell_B^{e_B}f - 1$ Family

- Sample prime: $2^{63}3^{41}11 - 1 =$ 0xadfe79f4ba68e76207ffffffffffffff
- Based on structure of prime, $p \equiv 3 \bmod 4$. We need fast addition chains for $p-2$, $\frac{p-3}{4}$, and $\frac{p-1}{2}$
- Hybrid Windowing Method: Perform efficient windowing method for most significant half of prime and add an extra window for $2^c - 1$ that can be efficiently used for least significant half of prime.
- Strategy to choose pivots to complete the addition sequence:
  - Number of newly connected elements with the inclusion of the pivot
  - Cost to generate the pivot value based on existing values (doubles are scored higher)
  - Among high scoring pivots, the uniqueness of the connected elements are valued

$2^{e_A}\ell_B^{e_B}f-1$ Family - Hybrid Method

Input: Smooth Isogeny prime of form $2^{e_A}\ell_B^{e_B}f-1$
Output: Fast Addition Chains for $p-2, \frac{p-3}{4}$, p, and $\frac{p-1}{2}$
1. Split first half of prime into various max-sized windows
   1a. The best choice of window size varies based on the prime
2. Include an additional 2^c-1, such that c makes a large window of all '1'sthat minimizes the number of multiplication windows for the second half of theprime as well as minimizing the number of multiplications to generate
3. Determine good addition sequences to generate the windows
   3a. Add additional stored values if necessary
4. Slightly alter choice of c and multiplications to finish the addition chain between $p-2, \frac{p-3}{4}$, p, and $\frac{p-1}{2}$

$2^{e_A}\ell_B^{e_B}f-1$ Family - Hybrid Method Example

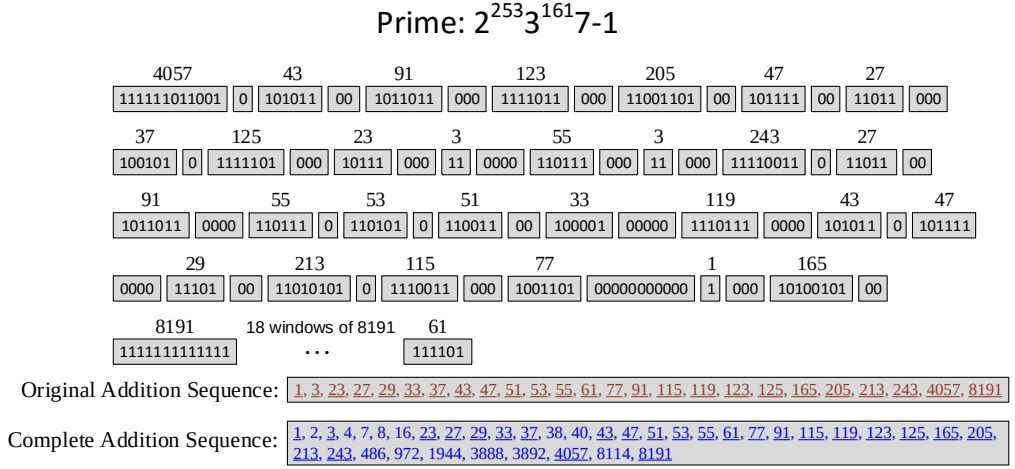


Figure: Hybrid windowing method for $2^{e_A}\ell_B^{e_B}f-1$

Table: Breakdown of Costs for Addition Chains for $2^{253}3^{161}7-3$

Operation	Cost
Window Generation	$28M+9S$
Applying Windows $1st$ Half	$28M+245S$
Applying Windows $2nd$ Half	$20M+254S$
Total	$76M+508S$

$2^{e_A}\ell_B^{e_B}f - 1$ Family - Windowing Method with a Subtraction

- For compression square roots, public information is only reconstructed $\rightarrow$ use fast inversion algorithm like extended Euclidean algorithm to have access to a subtraction in the addition chain
- Instead of computing an exponentiation by p, we can get an addition chain for $p+1$ and multiply by the inverse
  - $2^{63}3^{41}11$ =0xadfe79f4ba68e76208000000000000000
  - Exponentiation has many, many zeroes that do not require a window multiplication

$2^{e_A}\ell_B^{e_B}f+1$ Family

- Sample prime: $2^{60}3^{43}1-1 =$ 0x11cb7b04aa565d7b7b000000000000001
- Based on structure of prime, $p \equiv 1 \bmod 4$. We need fast addition chains for $p-2$, $\frac{p-1}{4}$, and $\frac{p-1}{2}$
- Hybrid Windowing Method for $p-2$
- Least significant half of prime for $\frac{p-1}{4}$ and $\frac{p-1}{2}$ is zero, so we only need to do a windowing method for the most significant half

General Smooth Isogeny Family

- Sample prime: $5^{34}7^{26}2-1 =$ 0x1ea0ca17d2408bd1e2bcaefbbdd5f347750ad11
- No simple structure of the prime to take advantage of
- p mod 4 depends on choice of isogeny graphs as well as f.
  - 3 mod 4 is preferred so that square root operation is simpler
- Generally focus on efficient windowing methods for any generic prime.

Reducing Temporary Registers

- Smart choices of addition sequence pivots and reuse of windows can reduce number of temporary registers

Input: Addition chain sequence based on the windowing method
Output: Efficient paths to perform the exponentiation with a reduced number ofregisters
1. Based on the addition sequence, generate a short path from 1 to the value of the first window
2. Remove values that are stored in registers based on the following criteria:
  2a. If a register has been used and is no longer required to make a path toother windows
  2b. If a separate register contains the value of a register multiplied by 2
3. As the windows are being applied, they can be performed by multiplying their factors directly instead of multiplying to a separate register.

Comparison of Addition Chain Methods

Table: Comparison of Addition Chains for Square Root Exponentiation by $p = 2^{253}3^{161}7 - 1$

Method	Win. Size	$\#\tilde{I}$	$\#\tilde{M}$	$\#\tilde{S}$	Time (μs)	# Reg
Binary	1	0	380	511	2.978	2
K-ary	2	0	224	511	2.435	4
K-ary	4	0	141	511	2.076	16
Standard Window	8	0	87	508	1.877	20
Hybrid Window	8	0	76	508	1.804	21
Window with a Subtraction	8	1	56	508	1.751	21

Results by Type of Smooth Isogeny Prime

Table: Comparison of Addition Chains for Smooth Isogeny Primes p_{512}

Expon.	#I	#M	#S	Win. Add. Seq. Len.	#Reg.	Max Win. Size	c
$p = 2^{253}3^{161}7 - 1$							
$p-2$	0	75	508	31	21	8	13
$\frac{p-1}{2}$	1	56	505	31	21	8	-
$p = 2^{254}3^{158}71 + 1$							
$p-2$	0	79	514	32	19	7	14
$\frac{p-1}{2}$	0	58	507	32	19	7	-
$p = 5^{108}7^{89}732 + 1$							
$p-2$	0	99	505	55	28	10	-
$\frac{p-1}{2}$	0	99	504	55	28	10	-

Results by Size of Smooth Isogeny Prime

Table: Comparison of Addition Chains for Smooth Isogeny Primes of Different Sizes

Exponen.	$\#I$	$\#M$	$\#S$	Win. Add. Seq. Len.	#Reg.	Max Win. Size	c
$p_{512} = 2^{253}3^{161}7 - 1$							
$p-2$	0	76	508	31	21	8	13
$\frac{p-1}{2}$	1	56	506	31	21	8	-
$p_{768} = 2^{379}3^{239}497 - 1$							
$p-2$	0	108	770	49	26	9	16
$\frac{p-1}{2}$	1	84	762	49	27	9	-
$p_{1024} = 2^{509}3^{320}107 - 1$							
$p-2$	0	134	1029	52	28	8	18
$\frac{p-1}{2}$	1	102	1020	52	29	8	-

Analysis of Results

- Cheapest inversion and square root exponentials for family $p = 2^{e_A}\ell_B^{e_B} \cdot f + 1$, however does not include square root computation for 1 mod 4 primes.
- Total number of registers scaled with max window size and addition sequence to generate the windows.
  - Smaller window sizes require fewer pivots to complete the addition sequence.
- For our choice of finishing the addition sequences, windows of size 7-10 achieved the best results.
- c value directly scaled with size of the prime.

Conclusions

- Review of SIDH protocol and addition chains
- Introduced hybrid windowing method for inversions in families $2^{e_A}\ell_B^{e_B} \cdot f + 1$ and $2^{e_A}\ell_B^{e_B} \cdot f - 1$
- Introduced window method with a subtraction for square root exponentiations in $2^{e_A}\ell_B^{e_B} \cdot f - 1$
- Provided methodology to reduce the total number or temporary registers throughout the exponentiation.
- Provided results for several primes using these methods
- For SIDH, fast addition chains can be determined and optimized outside of an implementation. These fast addition chains improve upon the speed of an implementation, almost for free if implemented correctly.

Thank You!