Accurately Benchmarking Efficiency of Pairing-Based Attribute-Based Encryption

Antonio de la Piedra[1] **Marloes Venema**[2] Greg Alpár[3,4]

Kudelski Security Research Team, Cheseaux-sur-Lausanne, Switzerland

University of Wuppertal, Wuppertal, Germany

Open University of the Netherlands, Heerlen, the Netherlands

Radboud University, Nijmegen, the Netherlands

SIAM-AG23

BERGISCHE UNIVERSITÄT WUPPERTAL

Motivation

- Attribute-based encryption (ABE) is a versatile primitive that has been considered extensively to securely manage access to data
- Various use cases, e.g., cloud-based settings

Motivation

- Attribute-based encryption (ABE) is a versatile primitive that has been considered extensively to securely manage access to data
- Various use cases, e.g., cloud-based settings
- While many (pairing-based) schemes exist, few have working implementations
- Existing implementations may not be fairly comparable

Motivation

- Attribute-based encryption (ABE) is a versatile primitive that has been considered extensively to securely manage access to data
- Various use cases, e.g., cloud-based settings
- While many (pairing-based) schemes exist, few have working implementations
- Existing implementations may not be fairly comparable
- **Our goal:** accurately benchmarking and comparing schemes, efficiency analysis, new speed records

High-level overview

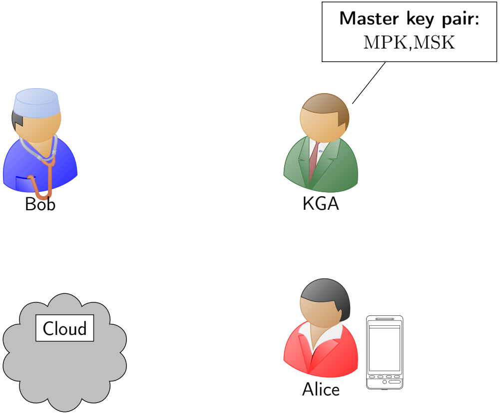
1. Introduction to ABE
2. Why is benchmarking ABE difficult?
3. ABE Squared
4. Towards automating ABE Squared
5. Conclusion

High-level overview

1. Introduction to ABE
2. Why is benchmarking ABE difficult?
3. ABE Squared
4. Towards automating ABE Squared
5. Conclusion

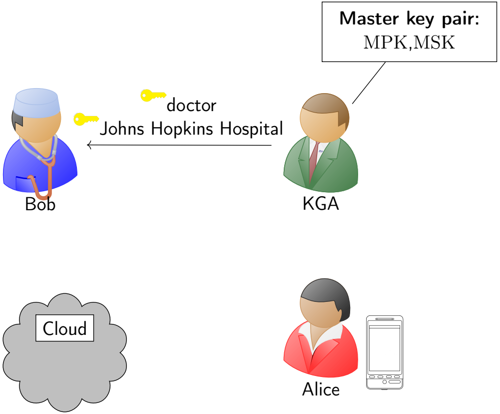
Ciphertext-policy attribute-based encryption (CP-ABE)

Setup:



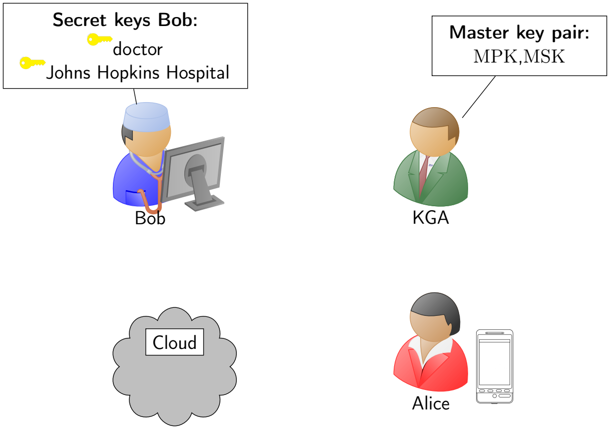
Ciphertext-policy attribute-based encryption (CP-ABE)

Key generation:



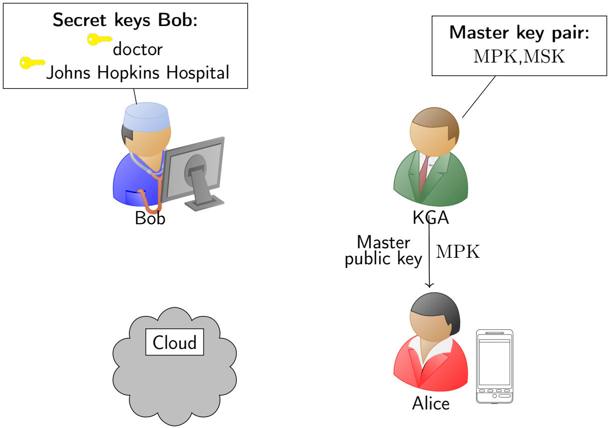
Ciphertext-policy attribute-based encryption (CP-ABE)

Key generation:



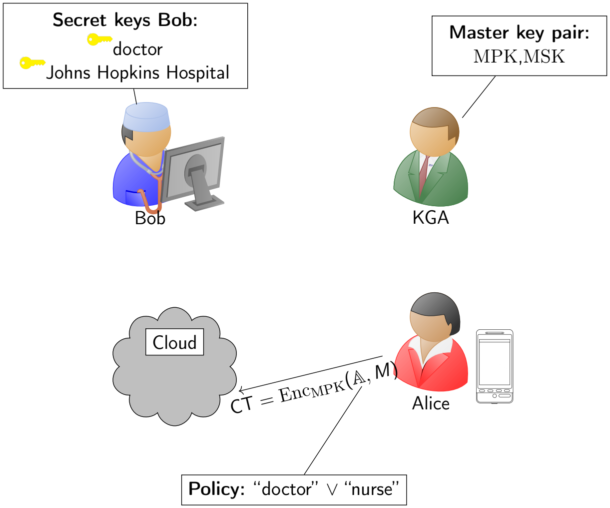
Ciphertext-policy attribute-based encryption (CP-ABE)

Encryption:

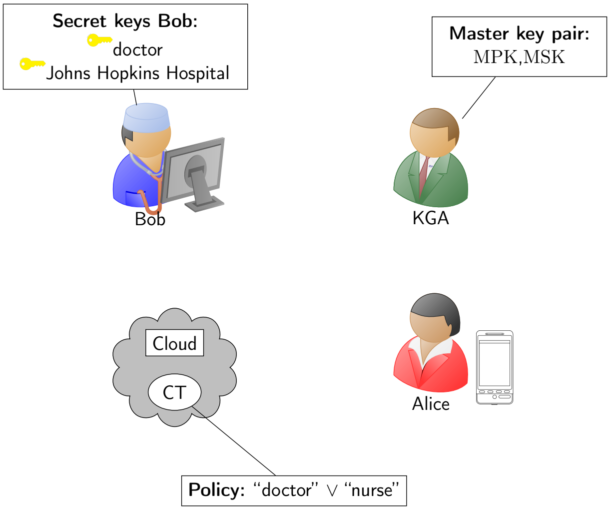


Ciphertext-policy attribute-based encryption (CP-ABE)

Encryption:

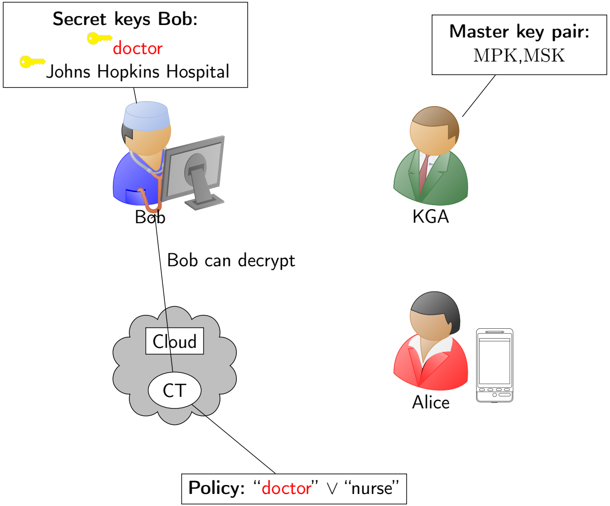


Ciphertext-policy attribute-based encryption (CP-ABE)



Ciphertext-policy attribute-based encryption (CP-ABE)

Decryption:



Enforcing access control with CP-ABE

- By its functionality, ABE implements access control
- Popular in settings in which data has to be stored on untrusted platforms

Enforcing access control with CP-ABE

- By its functionality, ABE implements access control
- Popular in settings in which data has to be stored on untrusted platforms
- The European Telecommunications Standards Institute (ETSI) considers several use cases for ABE, e.g., Cloud, IoT
- More recently, Cloudflare has presented an updated version of their Geo Key Manager: Portunus

Requirements for ABE

These use cases share many common requirements for ABE:

- **Expressive policies:** policies should support Boolean formulas consisting of AND and OR operators
- **Large universes:** attribute could be any arbitrary string, e.g., names, roles, MAC addresses
- **Unbounded:** no bounds on any parameters, such as the length of the policies or attribute sets

Some use cases also require **non-monotonicity**, i.e., the support of negations/NOT operators in the policies.

Requirements for ABE

These use cases share many common requirements for ABE:

- **Expressive policies:** policies should support Boolean formulas consisting of AND and OR operators
- **Large universes:** attribute could be any arbitrary string, e.g., names, roles, MAC addresses
- **Unbounded:** no bounds on any parameters, such as the length of the policies or attribute sets

Some use cases also require **non-monotonicity**, i.e., the support of negations/NOT operators in the policies.

Storage and computational efficiency requirements may vary per use case.

Requirements for storage and computational efficiency

Examples:

- Portunus and cloud settings: fast decryption
- Internet of Things: small ciphertexts, fast encryption

Pairing-based ABE

- We focus on pairing-based ABE
- Most established: many desirable practical properties, high security guarantees and efficient

Pairing-based ABE

- We focus on pairing-based ABE
- Most established: many desirable practical properties, high security guarantees and efficient
- Unfortunately, not post-quantum secure

Pairing-based ABE

- We focus on pairing-based ABE
- Most established: many desirable practical properties, high security guarantees and efficient
- Unfortunately, not post-quantum secure
- Post-quantum secure schemes exist
- However, still heavily under development, e.g., to achieve the same desirable properties

High-level overview

1 Introduction to ABE

2 Why is benchmarking ABE difficult?

3 ABE Squared

4 Towards automating ABE Squared

5 Conclusion

Benchmarking crypto

Usually:

- Make some choices (e.g., architecture, CPU, platform) required for a fair comparison
- Implement and optimize with a strategy in mind

Benchmarking crypto

Usually:

- Make some choices (e.g., architecture, CPU, platform) required for a fair comparison
- Implement and optimize with a strategy in mind

Typically, in ABE:

- Choose a framework for rapid prototyping, e.g., Charm [AGM+13]
- Implement, maybe optimize some parts

General problems in implementing pairing-based ABE

Many components need to be optimized, e.g.,

General problems in implementing pairing-based ABE

Many components need to be optimized, e.g.,

- access policies
- arithmetic and group operations
- many different pairing-friendly groups
- type conversion

General problems in implementing pairing-based ABE

Many components need to be optimized, e.g.,

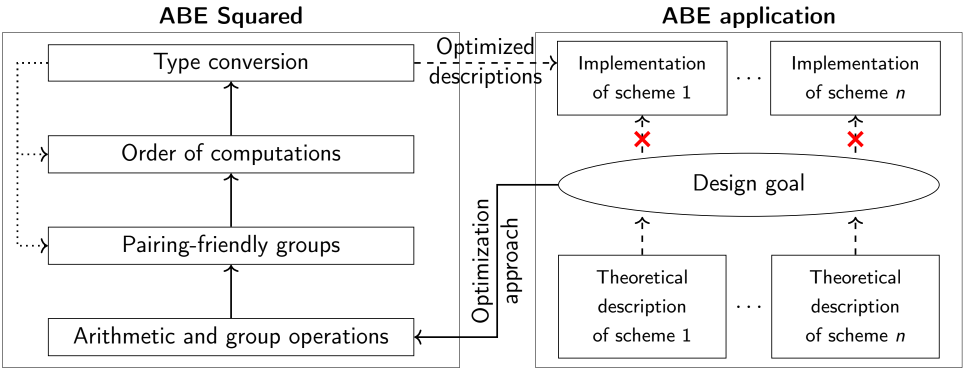
- access policies
- arithmetic and group operations
- many different pairing-friendly groups
- type conversion

Some of these really depend on what the designer tries to optimize, e.g., the decryption algorithm for Cloudflare's use case

High-level overview

1. Introduction to ABE
2. Why is benchmarking ABE difficult?
3. ABE Squared
4. Towards automating ABE Squared
5. Conclusion

Overview of ABE Squared



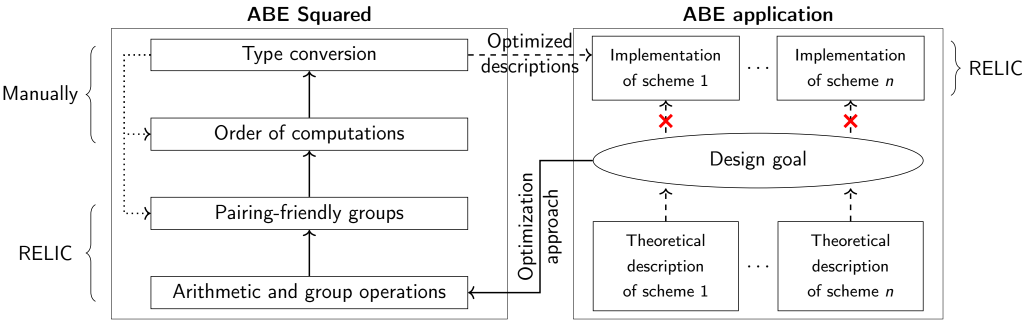
The arrows have the following meaning:

$a \longrightarrow b$ = "a influences b"

$a \cdots\rightarrow b$ = "a may require adjustment in b"

$a \dashrightarrow b$ = "a is input to b"/"b is output of a"

Overview of ABE Squared (continued)



The arrows have the following meaning:

$a \longrightarrow b$ = "a influences b"

$a \cdots\!\!\succ b$ = "a may require adjustment in b"

$a \dashrightarrow b$ = "a is input to b"/"b is output of a"

New heuristics

- Interaction between the upper two layers and how they interact with the lower two layers has not been investigated sufficiently for ABE yet

New heuristics

- Interaction between the upper two layers and how they interact with the lower two layers has not been investigated sufficiently for ABE yet
- Previous type-conversion methods typically only allow for optimized key or ciphertext sizes
- These cannot be used to optimize decryption

New heuristics

- Interaction between the upper two layers and how they interact with the lower two layers has not been investigated sufficiently for ABE yet
- Previous type-conversion methods typically only allow for optimized key or ciphertext sizes
- These cannot be used to optimize decryption
- We provide manual heuristics that take the interactions between the different layers into account
- Allows us to better optimize e.g., the decryption algorithm than previous methods allow

Optimized type conversion

Pairing: $e\colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$, where $\mathbb{G}, \mathbb{H}$ and $\mathbb{G}_T$ are groups of order p.

Optimized type conversion

Pairing: $e\colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$, where $\mathbb{G}, \mathbb{H}$ and $\mathbb{G}_T$ are groups of order p.

Theory: often assumes $\mathbb{G} = \mathbb{H}$
Practice: $\mathbb{G} \neq \mathbb{H}$

Optimized type conversion

Pairing: $e\colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$, where $\mathbb{G}, \mathbb{H}$ and $\mathbb{G}_T$ are groups of order p.

Theory: often assumes $\mathbb{G} = \mathbb{H}$
Practice: $\mathbb{G} \neq \mathbb{H}$ $\Rightarrow$ efficiency in $\mathbb{G} \neq$ efficiency in $\mathbb{H}$

Optimized type conversion

Pairing: $e\colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$, where $\mathbb{G}, \mathbb{H}$ and $\mathbb{G}_T$ are groups of order p.

Theory: often assumes $\mathbb{G} = \mathbb{H}$
Practice: $\mathbb{G} \neq \mathbb{H}$ $\Rightarrow$ efficiency in $\mathbb{G} \neq$ efficiency in $\mathbb{H}$

Type conversion: convert the scheme from setting with $\mathbb{G} = \mathbb{H}$ to $\mathbb{G} \neq \mathbb{H}$.

Optimized type conversion

Pairing: $e\colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$, where $\mathbb{G}, \mathbb{H}$ and $\mathbb{G}_T$ are groups of order p.

Theory: often assumes $\mathbb{G} = \mathbb{H}$
Practice: $\mathbb{G} \neq \mathbb{H}$ $\Rightarrow$ efficiency in $\mathbb{G} \neq$ efficiency in $\mathbb{H}$

Type conversion: convert the scheme from setting with $\mathbb{G} = \mathbb{H}$ to $\mathbb{G} \neq \mathbb{H}$.

Many ways to instantiate the scheme in $\mathbb{G}$ and $\mathbb{H}$!

Optimized type conversion

Pairing: $e \colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$, where $\mathbb{G}, \mathbb{H}$ and $\mathbb{G}_T$ are groups of order p.

Theory: often assumes $\mathbb{G} = \mathbb{H}$
Practice: $\mathbb{G} \neq \mathbb{H}$ $\Rightarrow$ efficiency in $\mathbb{G} \neq$ efficiency in $\mathbb{H}$

Type conversion: convert the scheme from setting with $\mathbb{G} = \mathbb{H}$ to $\mathbb{G} \neq \mathbb{H}$.

Many ways to instantiate the scheme in $\mathbb{G}$ and $\mathbb{H}$!

Each instantiation performs differently.

Optimized type conversion

Pairing: $e\colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$, where $\mathbb{G}, \mathbb{H}$ and $\mathbb{G}_T$ are groups of order p.

Theory: often assumes $\mathbb{G} = \mathbb{H}$
Practice: $\mathbb{G} \neq \mathbb{H}$ $\Rightarrow$ efficiency in $\mathbb{G} \neq$ efficiency in $\mathbb{H}$

Type conversion: convert the scheme from setting with $\mathbb{G} = \mathbb{H}$ to $\mathbb{G} \neq \mathbb{H}$.

Many ways to instantiate the scheme in $\mathbb{G}$ and $\mathbb{H}$!

Each instantiation performs differently.

Most efficient one depends on the lower three layers, i.e., arithmetic and group operations, chosen group and the order of the group operations.

Optimized type conversion

Pairing: $e\colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$, where $\mathbb{G}, \mathbb{H}$ and $\mathbb{G}_T$ are groups of order p.

Theory: often assumes $\mathbb{G} = \mathbb{H}$
Practice: $\mathbb{G} \neq \mathbb{H}$ $\Rightarrow$ efficiency in $\mathbb{G} \neq$ efficiency in $\mathbb{H}$

Type conversion: convert the scheme from setting with $\mathbb{G} = \mathbb{H}$ to $\mathbb{G} \neq \mathbb{H}$.

Many ways to instantiate the scheme in $\mathbb{G}$ and $\mathbb{H}$!

Each instantiation performs differently.

Most efficient one depends on the lower three layers, i.e., arithmetic and group operations, chosen group and the order of the group operations.

Our heuristics: find most efficient instantiation in $\mathbb{G}$ and $\mathbb{H}$ given a specific design goal. (Also depends on the chosen group!)

Benchmarks for differently optimized schemes

Implementation of Wat11-I in RELIC, on the BLS12-381 curve, based on their optimization approaches[1] (OA). The costs are expressed in 10^3 clock cycles[2].

OA	Key generation				Encryption				Decryption			
	# of attributes				# of attributes				# of attributes			
	1	10	100	Increase	1	10	100	Increase	1	10	100	Increase
OE & OD	759	3029	25653	143.0%	**990**	**4540**	**39951**	-	**2005**	**7379**	**58515**	-
OK	**317**	**1249**	**10555**	-	1756	10814	101181	153.3%	2016	7611	63151	7.9%

[1] OE/OD/OK = optimized encryption/decryption/key generation.
[2] AMD Ryzen 7 PRO 4750 processor, one single core at 4.1 GHz.

Benchmarks for differently optimized schemes

Implementation of Wat11-I in RELIC, on the BLS12-381 curve, based on their optimization approaches[1] (OA). The costs are expressed in 10^3 clock cycles[2].

OA	Key generation				Encryption				Decryption			
	# of attributes				# of attributes				# of attributes			
	1	10	100	Increase	1	10	100	Increase	1	10	100	Increase
OE & OD	759	3029	25653	143.0%	**990**	**4540**	**39951**	-	**2005**	**7379**	**58515**	-
OK	**317**	**1249**	**10555**	-	1756	10814	101181	153.3%	2016	7611	63151	7.9%

- Design goal influences the type conversion
- e.g., it yields a difference of a factor of ≈ 2.5 in computational costs for BLS12-381

[1] OE/OD/OK = optimized encryption/decryption/key generation.

[2] AMD Ryzen 7 PRO 4750 processor, one single core at 4.1 GHz.

Performance analysis

To demonstrate the framework, we have implemented and benchmarked three schemes with the same practical properties (achieved in different ways) in RELIC:

- Wat11-IV [Wat11]: implemented in libraries such as Charm and OpenABE
- RW13 [RW13]: implemented in Charm, outperformed by Wat11-IV
- AC17-LU [AC17]: not implemented

Many follow-up works build on these schemes and are structurally similar. Note that all these schemes satisfy the three important properties that we mentioned earlier (i.e., expressive, large-universe and unbounded).

Benchmarks for 100 attributes

OA	Scheme	Curve	Key generation		Encryption		Decryption	
			Costs	Increase %	Costs	Increase %	Costs	Increase %
OE	Wat11-IV	BLS12-381	42275	0.2%	77641	48.8%	58290	543.4%
	RW13	BLS12-381	51401	21.8%	54491	4.4%	112072	1137.1%
	AC17-LU	BLS12-381	**42196**	-	**52176**	-	9060	-
OK	Wat11-IV	BLS12-381	42135	94.6%	77898	48.9%	58441	543.9%
	RW13	BLS12-381	**21657**	-	128221	145.0%	118998	1211.2%
	AC17-LU	BLS12-381	41913	93.5%	**52326**	-	9076	-
OD	Wat11-IV	BLS12-381	**42275**	-	77641	42.5%	58290	1336.5%
	RW13	BLS12-381	51401	21.6%	**54491**	-	112072	2661.9%
	AC17-LU	BN382	45093	6.7%	59276	8.8%	**4058**	-

Benchmarks for 100 attributes

OA	Scheme	Curve	Key generation		Encryption		Decryption	
			Costs	Increase %	Costs	Increase %	Costs	Increase %
OE	Wat11-IV	BLS12-381	42275	0.2%	77641	48.8%	58290	543.4%
	RW13	BLS12-381	51401	21.8%	54491	4.4%	112072	1137.1%
	AC17-LU	BLS12-381	**42196**	-	**52176**	-	9060	-
OK	Wat11-IV	BLS12-381	42135	94.6%	77898	48.9%	58441	543.9%
	RW13	BLS12-381	**21657**	-	128221	145.0%	118998	1211.2%
	AC17-LU	BLS12-381	41913	93.5%	**52326**	-	9076	-
OD	Wat11-IV	BLS12-381	**42275**	-	77641	42.5%	58290	1336.5%
	RW13	BLS12-381	51401	21.6%	**54491**	-	112072	2661.9%
	AC17-LU	BN382	45093	6.7%	59276	8.8%	**4058**	-

- For an optimized encryption or decryption, use AC17-LU
- For an optimized key generation, use RW13

Benchmarks for 100 attributes

OA	Scheme	Curve	Key generation		Encryption		Decryption	
			Costs	Increase %	Costs	Increase %	Costs	Increase %
OE	Wat11-IV	BLS12-381	42275	0.2%	77641	48.8%	58290	543.4%
	RW13	BLS12-381	51401	21.8%	54491	4.4%	112072	1137.1%
	AC17-LU	BLS12-381	**42196**	-	**52176**	-	9060	-
OK	Wat11-IV	BLS12-381	42135	94.6%	77898	48.9%	58441	543.9%
	RW13	BLS12-381	**21657**	-	128221	145.0%	118998	1211.2%
	AC17-LU	BLS12-381	41913	93.5%	**52326**	-	9076	-
OD	Wat11-IV	BLS12-381	**42275**	-	77641	42.5%	58290	1336.5%
	RW13	BLS12-381	51401	21.6%	**54491**	-	112072	2661.9%
	AC17-LU	BN382	45093	6.7%	59276	8.8%	**4058**	-

- For an optimized encryption or decryption, use AC17-LU
- For an optimized key generation, use RW13
- **Surprising result:** RW13 outperforms Wat11-IV in the key generation and encryption algorithms

High-level overview

1. Introduction to ABE
2. Why is benchmarking ABE difficult?
3. ABE Squared
4. Towards automating ABE Squared
5. Conclusion

Future work: automation and more curves

- For future work, it would be valuable to automate ABE Squared
- Furthermore, it would be valuable to analyze the efficiency of schemes for more curves
- Existing libraries for curve arithmetic often support very few curves
- There exist many curves at the 128-bit security level (see `https://members.loria.fr/AGuillevic/pairing-friendly-curves/`):

Curve	k	D	u	ref	p (bits)	r (bits)	$p^{k/d}$ ($\mathbb{G}_2$, bits)	p^k (bits)
Curve for fastest pairing								
Barreto–Lynn–Scott BLS12, cyclotomic $r(x)$	12	3	$-(2^{73}+2^{72}+2^{50}+2^{24})$	eprint 2017/334	440	295	p^2, 880	5280
Barreto–Lynn–Scott BLS12, cyclotomic $r(x)$	12	3	$-(2^{12}-2^{48}-2^{71}+2^{74})$	eprint 2017/334	442	296	p^2, 884	5296
Barreto–Lynn–Scott BLS12, cyclotomic $r(x)$	12	3	$-(2^{74}+2^{73}+2^{63}+2^{57}+2^{50}+2^{17}+1)$	eprint 2019/885	446	299	p^2, 892	5352
Fotiadis–Martindale FM17, Aurifeuillean $r(x)$	12	3	$-2^{72}-2^{71}-2^{36}$	eprint 2019/555	447	296	p^2, 894	5356
Kachisa–Schaefer–Scott KSS16	16	1	$-2^{34}+2^{27}-2^{23}+2^{20}-2^{11}+1$	eprint 2017/334	330	257	p^4, 1320	5280
Kachisa–Schaefer–Scott KSS16	16	1	$2^{34}-2^{30}+2^{26}+2^{23}+2^{14}-2^5+1$	eprint 2019/1371	330	256	p^4, 1320	5268
Kachisa–Schaefer–Scott KSS16	16	1	$2^{35}-2^{32}-2^{18}+2^8+1$	eprint 2017/334	339	263	p^4, 1356	5411
Curve with small embedding degree k								
Cocks–Pinch modified	6	3	$2^{128}-2^{124}-2^{69}$, $h_t=-1$, $h_y=2^{80}-2^{70}-2^{66}-$ `0x3fe0` = `0xffbbffffffffffffc020`	eprint 2019/431	672	256	p, 672	4028
Cocks–Pinch modified	8	1	$2^{64}-2^{54}+2^{37}+2^{32}-4$, $h_t=1$, $h_y=$ `0xdc04`	eprint 2019/431	544	256	p^2, 1088	4349
Curve with smallest $\mathbb{G}_1$								

Finding the best curve

- Implementing curve arithmetic for over 20 curves is (too) ambitious
- Perhaps a better approach: theoretically approximate the efficiency for each curve
- Extrapolate the efficiency of schemes using these approximations
- Implement those that we believe are the best choices for ABE (applications)

Finding the best curve

- Implementing curve arithmetic for over 20 curves is (too) ambitious
- Perhaps a better approach: theoretically approximate the efficiency for each curve
- Extrapolate the efficiency of schemes using these approximations
- Implement those that we believe are the best choices for ABE (applications)
- Question: do we need to?
- Perhaps, BLS12-381 is already good enough

Finding the best curve

- Implementing curve arithmetic for over 20 curves is (too) ambitious
- Perhaps a better approach: theoretically approximate the efficiency for each curve
- Extrapolate the efficiency of schemes using these approximations
- Implement those that we believe are the best choices for ABE (applications)
- Question: do we need to?
- Perhaps, BLS12-381 is already good enough
- BLS12-381 believed to currently provide 126 bits of security
- ABE schemes typically lose some extra bits of security

Finding the best curve

- Implementing curve arithmetic for over 20 curves is (too) ambitious
- Perhaps a better approach: theoretically approximate the efficiency for each curve
- Extrapolate the efficiency of schemes using these approximations
- Implement those that we believe are the best choices for ABE (applications)
- Question: do we need to?
- Perhaps, BLS12-381 is already good enough
- BLS12-381 believed to currently provide 126 bits of security
- ABE schemes typically lose some extra bits of security
- Could be better to use a curve with $>$ 128 bits of security
- Additionally, different curves have different efficiency trade-offs
- Natural to think that, for each scheme and design goal, there may be a different optimal curve

Performance estimations for some curves

We estimate the costs (very roughly!) for RW13 using the field-arithmetic benchmarks in [GMT20], in milliseconds:

OA	Curve	Key generation	Encryption	Decryption
OE/OD	BLS12-446	400	332	141
	CP8-544	**173**	431	122
	KSS16-330	874	**218**	**102**
OK/OD	BLS12-446	133	994	141
	CP8-544	173	**431**	122
	KSS16-330	**87**	2169	**102**

- KSS16-330 may yield better efficiencies for the one-algorithm optimization strategies (OK/OE/OD)
- More "balanced" curves such as CP8-544 may be more suitable for more "balanced" efficiencies among the algorithms

High-level overview

1. Introduction to ABE
2. Why is benchmarking ABE difficult?
3. ABE Squared
4. Towards automating ABE Squared
5. Conclusion

Conclusion

- ABE Squared: a framework for accurately benchmarking efficiency of attribute-based encryption

Conclusion

- ABE Squared: a framework for accurately benchmarking efficiency of attribute-based encryption
- Aims to optimize ABE schemes for some chosen design goal by considering four optimization layers:
  - arithmetic and group operations
  - pairing-friendly groups
  - order of the computations
  - type conversion

Conclusion

- ABE Squared: a framework for accurately benchmarking efficiency of attribute-based encryption
- Aims to optimize ABE schemes for some chosen design goal by considering four optimization layers:
  - arithmetic and group operations
  - pairing-friendly groups
  - order of the computations
  - type conversion
- By optimizing multiple schemes with respect to the same goal, they can be compared more fairly

Conclusion

- ABE Squared: a framework for accurately benchmarking efficiency of attribute-based encryption
- Aims to optimize ABE schemes for some chosen design goal by considering four optimization layers:
  - arithmetic and group operations
  - pairing-friendly groups
  - order of the computations
  - type conversion
- By optimizing multiple schemes with respect to the same goal, they can be compared more fairly
- Existing open-source libraries providing ABE implementations, e.g., Charm, OpenABE, can greatly benefit from our heuristics
- **Design goals matter:** for different goals, different schemes may perform the best

Questions?

Thank you for your attention!

- Our paper:
  - TCHES: `tches.iacr.org/index.php/TCHES/article/view/9486`
  - eprint: `https://eprint.iacr.org/2022/038`
- Our code: `https://github.com/abecryptools/abe_squared`

References I

[AC17] S. Agrawal and M. Chase.
Simplifying design and analysis of complex predicate encryption schemes.
In J.-S. Coron and J. B. Nielsen, editors, *EUROCRYPT*, volume 10210 of *LNCS*, pages 627–656. Springer, 2017.

[AGM+13] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin.
Charm: a framework for rapidly prototyping cryptosystems.
J. Cryptogr. Eng., 3(2):111–128, 2013.

[GMT20] A. Guillevic, S. Masson, and E. Thomé.
Cocks-Pinch curves of embedding degrees five to eight and optimal ate pairing computation.
Des. Codes Cryptogr., 88(6):1047–1081, 2020.

[RW13] Y. Rouselakis and B. Waters.
Practical constructions and new proof methods for large universe attribute-based encryption.
In A.-R. Sadeghi, V. D. Gligor, and M. Yung, editors, *CCS*, pages 463–474. ACM, 2013.

[Wat11] B. Waters.
Ciphertext-policy attribute-based encryption - an expressive, efficient, and provably secure realization.
In D. Catalano, N. Fazio, R. Gennaro, and A. Nicolosi, editors, *PKC*, volume 6571 of *LNCS*, pages 53–70. Springer, 2011.