Attribute-based encryption and signatures

discussing ownership (vs possession) and its applications

Marcello Paris
marcello.paris@gmail.com

CifrisCloud
April 28, 2021 17:00

Agenda

Attribute-based encryption is a wide research area in cryptography started by A. Sahai and B. Waters in 2005. This talk aims to give a brief informal overview of some of the topics and techniques involved, highlighting some elementary consequences of an attribute-based notion of ownership.

- $\frac{1}{3}$ attributes-based cryptography primitives and standard applications
- $\frac{1}{3}$ an idea of the standard recipe:
  linear secret sharing schemes and pairing-based setups on curves
- $\frac{1}{3}$ analogies to script-based (transfer of) ownership and
  applications of attribute-based ownership
  to controlled anonymity for digital assets

DISCLAIMER the content of this article is solely the responsibility of the author.
The views expressed here are those of the author and strictly personal,
and do not necessarily reflect the views of his employer.

Users, attributes, authorities and policies

- authorities are entities capable of granting attributes to users [a], so users are granted labels such as ATTRIBUTE@AUTHORITY (email notation). Examples:

  STUDENT@SOMEUNIVERSITY, PROF@SOMEUNIVERSITY

  DOCTOR@SOMEHOSPITAL, NURSE@SOMEHOSPITAL

  ROLE@SOMECOMPANY, TEAM@SOMECOMPANY (possibly overlapping)

  LEVEL=2@SOMECOMPANY (encoding numeric properties)

  ROLE@MYSOCCERTEAM (finer granularity)

  ME@SOMEREGISTRY (individual)

- policies are monotone[b] boolean functions on attributes (viewed as boolean values: `true` if the user does have it, `false` otherwise). We need them to be able to securely select subsets of users (i.e. those that can satisfy the policy using their own bag of attributes).

  $\text{STUDENT@SOMEUNIVERSITY} \vee \text{PROF@SOMEUNIVERSITY}$

  $\text{TEAM0@SOMECOMPANY} \vee \text{TEAM1@SOMECOMPANY}$

  $\text{DEPT@SOMEHOSPITAL} \wedge \left(\text{DOCTOR@SOMEHOSPITAL} \vee \text{PHD@SOMEHOSPITAL}\right)$

[a] the so-called CP-ABE approach putting attributes on users and policies on messages

[b] those which (in reduced disjuctive normal form) does not contain negations

Attribute-based cryptography

attribute-based cryptograhpy is a kind of asymmetric cryptograhpy where

⟶ private keys (kept by users and forged by authorities) are associated with parameters (attributes and users)
$\textsc{student@someuniversity} \mapsto \mathrm{prvkey}_{\mathrm{user}}(\textsc{student@someuniversity})$

⟶ plaintext/cyphertext are associated with policies (association is up to the users, the policies select the users which are the intended recipients of the encryption)

⟶ authorities have their own key pairs

in such a way that

ABE plaintexts could be encrypted by anyone w.r.t. a policy (and the public keys of the relevant authorities) decided by the encryptor and any resulting ciphertext could be successfully decrypted only by those users selected by the policy (i.e., those users carrying an allowed collection of attributes satisfying the policy)

ABS plaintexts could be signed w.r.t a given policy only by those users carrying attributes so to satisfy the policy, so that the resulting signature may be verified by anyone, proving the signer had enough attributes that can sastify the policy, but releasing no information other this (privacy of signers)

Decentralized multi-authority setup

MA we are describing a natural multi-authority setup where each user possessing attributes coming possibly from different authorities

DMA an important features of multi-authority setup is whether it can be fully decentralized, i.e. all authorities sits on an equal floor w.r.t. the cryptographical setup: in particular, every entity can become and authority without any need of global coordination (except for a basic environment setup)

a multi-authority setup is crucial for the overall security of the system.
the private key of an authority gives clearly control over everything,
but, if a user has the possiblity to collect even the same attribute
from different and (cryptographically) independent authorities
(the policy reflecting the need for all these copies with a $\wedge$ operator),
then an attack over all the authorties become surely more problematic.
This is also why a setup without a central authority is so much fundamental.

so, even if the practical setup would require conceptually only 1 authority,
it could be important to go for decentralization and reflect this setup
in the policies

Collusion

COLLUSION in order policies to securely select subset of users, we should, at least, not allow any advantage in users to collude (say, sharing their attributes)

MULTI-PARTY please remark that this kind of cryptography (CP-ABE) is still user-centric as far as operations (encrypt/decrypt, sign/verify) are concerned.
Even if the user could be hidden behind its attributes, the no-collusion property is designed for a single-user to operate and policies are not selecting groups of users to operate in a multi-party setup.

NEGATION we could also drop the monotone assumptions including negative attributes, i.e. negations of attributes (say, ¬STUDENT@SOMEUNIVERSITY).
This is tractable and could be useful, it has neverthless consequences on the granting process (and we should require the policy to be satisfiable by some bag of attributes)

GRANT each practical application should define better the act of 'granting the user an attribute' from an authority (such as: the process itself or attributes management in general). In particular, negative attributes could be mandatory (user possibly losing priviledges, users may not ask for this)

Attribute-based group primitives

all users can

- encrypt a message according to a policy they decide, i.e. encrypt a message for the users selected by the policy
- verify a signed message

only users satisfying the policy could

- decrypt an encrypted message
- sign a message according to a given policy, i.e. produces a verifyable signature proving that the signer could satisfy the policy

please remark:

→ this is encryption **for a group**

→ user performing encrypt or verify may clearly not satisfy the policy
(may have no attributes at all)

→ user performing encrypt may ignore the users' recipient set (even if it is currently not empty)

→ this is signing **by a group**

→ user performing decrypt needs the policy with which the ciphertext was encrypted

→ user performing decrypt may not be able to identify the sender even if the sender signed the message

→ user performing sign may indeed produce different signatures for different policies using the same subset of attributes

Actual schemes implementing this setup

ABE decentralized multi-authority CP-ABE schemes are described in [RW15] (improving on the DMA setup of [LW11]) and [OT20].
Both approaches are based on pairing-based cryptography
(we'll not cover LWE-based approaches in this talk)

ABS [OT20] covers also attribute-based signatures
(defined using the same techniques employed for encryption primitives)
(ABS were firstly defined in [MPR11])

CORE at the core of both constructions, there is the notion of **span program**
(general for [OT20], monotone span programs or linear secret sharing schemes LSSS in the case of [RW15]). Both approaches (monotone case):

→ embded the scene in a bilinear group: attributes and users are hashed to the group,

→ move a plaintext (a point on the group) by a **secret** random (shifting) factor

→ represents the policy as a span program and share this secret shifting factor **among the attributes** mentioned in the policy

→ use the bilinear form on the group to balance and
play with this splitting in such a way that

→ providing attributes satisfying the policy is equivalent
to reconstruct the secret shifting factor (so, the plaintext)

Messaging, sharing & access control

before discussing any mathematical detail, let's see how all this could be so useful in many applications: authenticated group messaging, file sharing, code sharing, access control, trust negotiations (and surely many more)

MESSAGING a chat could be naturally encrypted, without users need to be aware of each identity. The possibility of ABS allows to include authentication (say, messages signed by senders proving that they are part of the chat or, anyway, allowed to send). If useful, the sender could be hidden.

FILE SHARING a basic yet powerful application, which could allow for more complex encryption and authentication management. Also, code repos (say, github.com) are natural targets for this application (code is shared among the developers, yet cannot be encrypted for any particular developer and often needs customized authentication and action policies)

ABAC switching from Role-Based to Attribute-Based Access Control means improving toward a more risk- and context-aware access control. It can be used also to develop easier and more robust authentication schemes.

IT ARCH authorization and so security could be brought at data-level, no more in an external layer typical of ring-fenced security. Each piece of data can bring with itself its own authorization layer

Span programs and access structures

- let $X := \{x_0, \ldots, x_{|K|-1}\}$ be a set of attributes. an access structure on X is a monotone collection of non-empty subsets of X (i.e., if $A \in \mathcal{A}$ and $A \subseteq B$, then $B \in \mathcal{A}$). Sets in $\mathcal{A}$ are call *authorized*
- a span program (M, ρ) over a field $\mathbb{F}$ with labels in X ([Bei11]) is a $n \times m$ matrix M over $\mathbb{F}$ (n is call the size $|M|$ of M) coming with a labeling ρ of its rows:
  to each row being attached $r \mapsto \rho(r)$ a *positive* x_k or a *negative* $\neg x_k$ attribute so that, for any configuration $\sigma \in 2^X$ of the attributes, a submatrtix M_σ in defined as those rows mapping to the given configuration.
  A span program is *monotone* if ρ involves only positive attributes
- let $\mathbf{1} := (1, 0, \ldots, 0) \in \mathbb{F}^m$ a fixed vector. we say that (M, ρ) accepts a configuration $\sigma \subseteq X$ if $\mathbf{1} \in \mathrm{span}(M_\sigma)$. we say that (M, ρ) accepts an access structure $\mathcal{A}$ on X if it accepts only sets in $\mathcal{A}$
- we defined policies as boolean functions $f(\sigma)$ on attributes
  (here $\sigma \subseteq X$ to be expanded as an input to f)
  A span program (M, ρ) computes a boolean function f
  if it accepts only inputs $\sigma : f(\sigma) = \mathrm{true}$.
  Monotone span programs correspond to monotone boolean functions

Linear secret sharing schemes

- given a monotone span program (M, ρ) over a field $\mathbb{F}$ with labels in X (accepting a certain access structure over X)

  we can construct a linear secret sharing scheme

  (indeed, it can be shown that a linear secret sharing scheme is an equivalent formulation of a monotone span program)

→ let $z \in \mathbb{F}$ be a secret to be shared among the $|K|$ attributes

→ let $n \times m$ be the shape of the matrix M and pick a random completion of z to a vector $\omega_z := (z, \omega_1, \ldots, \omega_{m-1}) \in \mathbb{F}^m$ (here, we are using the definition of the fixed vector $\mathbf{1} := (1, 0, \ldots, 0) \in \mathbb{F}^m$)

→ compute the vector $(s_0, \ldots, s_{n-1}) := M \cdot \omega_z$ of n shares, so that the share s_i belongs to party $\rho(i) = x_k$ for some $k \in K$ $(i = 0, \ldots, n-1)$ (there could me more share than attributes, each attribute possibly collecting more than 1 share of the secret)

→ the above procedure satisfy the *correctness* (any authorized set of parties can reconstruct the secret) and *perfect privacy* (unauthorized parties learns nothing about the secret from their shares) requirements to be a secret sharing scheme

Pairing-based setup

PAIRING $\mathbb{G}_1,\ \mathbb{G}_2,\ \mathbb{G}_T$ groups of prime order $p \in \mathbb{Z}$ (all isomorphic
but possibly very different in element representation and computations)
$e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ efficiently computable
$e(aP,\ bQ) = e(P,\ Q)^{ab}$ (bilinear) $(a, b \in \mathbb{Z})$ and
$G_T\ := \ e(G_1, G_2) \neq 1_T \in \mathbb{G}_T$ (non-degenerate)

HASH-TO-GROUP $H_{id}\ :\ \mathcal{U} \ni u \mapsto U \in \mathbb{G}_2$ ($\mathcal{U}$ are users)
$H_{attr}\ :\ \mathcal{A} \ni a \mapsto A \in \mathbb{G}_2$ ($\mathcal{A}$ are attributes)

AUTH KEY $(y, \alpha) \in \mathbb{Z}_p \times \mathbb{Z}_p$ random private key,
$(y, \alpha) \mapsto \left(yG_1,\ G_T^{\alpha}\right) \in \mathbb{G}_1 \times \mathbb{G}_T$ public key
(remark that $G_T^{\alpha} = e(G_1, \alpha G_2)$)

ATTR KEY within a given authority (y, α), for each $(u, a) \in \mathcal{U} \times \mathcal{A}$ user-attribute pair,
a corresponging attribute key can be produced, for each random $t_a \in \mathbb{Z}_p$
by $\left(t_a G_1,\ \alpha G_2 + yU + t_a A\right) \in \mathbb{G}_1 \times \mathbb{G}_2$

→ U is in the definition of the attribute key

→ key could be re-randomized by $\left((t_a + t_a')G_1,\ \alpha G_2 + yU + (t_a + t_a')A\right)$

POLICY given a policy f, we can construct a monotone span program (M, ρ)
implementing it

Encryption

ENCRYPT Let $p \in \mathbb{G}_T$ be a plaintext, we encrypt it according to an monotone $n \times m$ span program (M, ρ) as follows: take random secrets $v, w \in \mathbb{Z}_p^m$ with $(\mathbf{1} \cdot w) = 0$, the ciphertext is defined as:

a random perturbation $p\, G_T^{(\mathbf{1} \cdot v)} \in \mathbb{G}_T$ of the message +
the bag of points

$$\left\{ \left(-t_a G_1,\ t_a y G_1 + (M_a \cdot w) G_1,\ t_a A,\ G_T^{(M_a \cdot v)} G_T^{\alpha\, t_a} \right) \right\}_{a \in \mathrm{rows}(M)} \in \left(\mathbb{G}_1 \times \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_T \right)^{|M|}$$

where $t_a \in \mathbb{Z}_p$ is random for each attribute and M_a is the row of M corresponding (under ρ) to the attribute $a \in \mathcal{A}$. Please remark:

→ encryptor splits (according to the policy matrix M) $(\mathbf{1} \cdot v)$ over $\mathbb{G}_T$ as $\left\{ G_T^{(M_a \cdot v + t_a \alpha)} \right\}_{a \in \mathrm{rows}(M)} \in \mathbb{G}_T^{|M|}$ and $(\mathbf{1} \cdot w) = 0$ over $\mathbb{G}_1$ as $\{(M_a \cdot w + t_a y) G_1\}_{a \in \mathrm{rows}(M)} \in \mathbb{G}_1^{|M|}$

→ $(\mathbf{1} \cdot w)$ is split to prevent collusion attacks

→ encrypting a message $p \in \mathbb{G}_T$ produce a ciphertext of $1 + 4|M|$ points, and we may need to bring the policy (M, ρ) alongside with the ciphertext

→ the actual ciphertext depends on the access policy matrix (M, ρ) and could be different even for equivalent policies

→ encryption is probabilistic and the ciphertext could be re-randomized on $t_a \mapsto t_a + t'_a$

Decryption

REMARK the decryptor, given as ciphertext the shifted point $p\, G_T^{(\mathbf{1}\cdot v)} \in \mathbb{G}_T$ and the bag of points

$$\Big\{\Big(-\,t_a G_1,\ t_a y G_1 + (M_a\cdot w) G_1,\ t_a A,\ G_T^{(M_a\cdot v)}\, G_T^{\alpha\, t_a}\Big)\Big\}_{a\in\mathrm{rows}(M)} \in \big(\mathbb{G}_1\times\mathbb{G}_1\times\mathbb{G}_2\times\mathbb{G}_T\big)^{|M|}$$

can remark that, for each private attribute keys she/he have, $\Big(t_a G_1,\ \alpha G_2 + yU + t_a A\Big) \in \mathbb{G}_1 \times \mathbb{G}_2$, she/he can compute

$$G_T^{(M_a\cdot v)}\, G_T^{\alpha\, t_a} \cdot e\Big(-\,t_a G_1,\ \alpha G_2 + yU + t_a A\Big)\cdot$$
$$\cdot\, e\Big(t_a y G_1 + (M_a\cdot w) G_1,\ U\Big)\cdot e\Big(t_a G_1,\ t_a A\Big) \;=\; G_T^{(M_a\cdot v)} \cdot e\Big(G_1, U\Big)^{(M_a\cdot w)}$$

DECRYPT so, the decryptor having the attributes to satisfy the policy means that, if she/he knows the span program (M,ρ), she/he can find a solution to

$$\sum_{a\in\mathcal{A}} c_a\, M_a \;=\; \mathbf{1} \qquad (c_a \in \mathbb{Z}_p)$$

and recontruct the perturbation factor as

$$\prod_{a\in\mathcal{A}} \bigg(G_T^{(M_a\cdot v)} \cdot e\Big(G_1, U\Big)^{(M_a\cdot w)}\bigg)^{c_a} \;=\; G_T^{(\mathbf{1}\cdot v)}$$

Implementation issues

when facing actual usable implementation, a quatity of structure comes into play (some of them I put in my lib on github.com)

- attributes and policy management (input, monitoring, parsing, health checks, the actual subset of users selected by policies, admin priviledges, . . . , also because no optimization of the attributes/policies sets is actually envisaged by CP-ABE itself)
- numerical setup: selection of curves and pairings and underlying cryptographic library, hash-to-groups primitives
- quality of randomness, in-memory management and code security setup
- tests or (kind of) formal verifications
- identifiers for users, attributes and authorities
  should be consistent across each setup
- decentralized doesn't mean that all authorities do carry the same trust

(TRANSFER OF) OWNERSHIP VS POSSESSION

In order to introduce the application I wish to discuss,
I need some kind of bridge between these quasi-legal notions and cryptographic primitives.

- transfer of ownership is a **public** act, to be performed in front of the reference community, while transfer of possession is something private, somehow **exclusive**

→ ownership is something to be transfered by **signatures**,
something that everyone in the community could (should) verify

→ possession is something to be transfered by **encryption**.
To encrypt something is to transfer the possession of that object,
to put it into a drawer accessible only by someone else.
To decrypt something is to open our private drawer
(private because others are excluded)

EX onwership of houses is transferred by a public act,
possession of cash just from hand to hand

EX transforming ownership into possession (say, an ATM) is taking
the objects out from the community, so that it becomes a private exclusive thing

AN IMPORTANT EXAMPLE: BITCOIN SCRIPTING

Bitcoin is a programmable mathematical commodity. A Bitcoin transaction typically contains instructions written in terms of a stack-based scripting language

LOCK it contains a locking script (*scriptPubKey*) which need to eval to `true` in order for the transaction to be successful (this locking script is analogous to a line requesting a signature on a form)

UNLOCK leaving out for 1sec all the about mining, blockchain and protocol design, focusing only on transaction verification on single trust-free entity, the design is about unlocking the outputs of a former transaction (with something like a signature in standard asymmentric cryptography that could be verified by anyone), i.e., providing an unlocking input (*scriptSig*) to the locking script so that it will evaluate to `true`

P2PKH the very basic P2PKH (pay-to-pub-key-hash) script (maybe still a widely used transaction toghether with the more recent P2SH_P2WPKH), just essentially demands a verification of the signature (opcode OP_CHECKSIG) for a transaction input

Attribute-based (group) ownership

CRYPTO this cryptography-defined transfer of ownership lies at the core of Bitcoin, which is all about ownership (and not about possession) [a].
Bitcoin is a triumph of asymmetric cryptography and specifically of signatures[b], not of encryption. Bitcoin transactions are not encrypted (whom to encrypt them for ? To encrypt is to exclude someone):
here we are going for a public act
(where public means for everyone, not just in the community)

ABE we could therefore think

- of ABE policies as if they were 'locking' scripts, and
- of ABE attributes to behave like 'unlocking' scripts

in this sense, we could speak about attribute-based ownership (transfer-of), a kind of collective ownership that doesn't allow to further dig into the aggregations impied by the active policies (so the collective nature implies some level of anonimity)

[a] please take it as a rough statement, because of pay-to-script-hash and off-chain transaction

[b] Bitcoin's protocol does not require the use of signatures, it requires that a transaction output specify the way it could be unlocked, and checking of signatures are by far the most common way of locking a transaction output

Anonymity (within a group) and privacy

- Bitcoin address are rarely attributable to a single person or entity, mainly because there is no registry[a] of entity addresses, nevertheless we cannot really claim that Bitcoin is anonymous by construction, rather, it is quasi the opposite (in a sense): if anyone can link an address to an entity, the tracing is complete.
- while with attribute-based signatures we are facing something which could allow built-in anonimity by construction
- anonimity and not privacy, i.e. it is not that the signer in ABS is kept private at some level (to be disclosed only according to some rules), it is just completely unfeasible to know which of the users did sign among those who could (from the ABS structure alone, clearly)
- surely enough, you can design some privacy with ABS, but, as with many cryptography tools, either you go for identifying the signer or you don't, there is nothing in the middle

[a] this is due to wallet managements, users' habits but also to pay-to-script-hash or simply that a given entity could have many addresses. Moreover, please remark alongside that in standard asymmetric cryptography, we cannot possibly choose our public key

Controlled anonymity for digital assets

digital assets are the object of transfer of ownership betwteen users or groups

- privacy is not anonimity, but some level of anonimity for digital assets is something we may definitely need. Think of whatever digital asset: books, reports, photos, medical research, pharmacy drugs, pregnacy tests, . . .
- many people would list some privacy among basic human rights, harder to go for anonimity. however, there could be situations where freedom[a] is also allowed by a controlled level of anonimity (or, the other way around, not allowing any level of anonimity can put our democracy at risk if privacy is violated by a dictator or a Big Brother)

[a] this freedom is well different from the kind offered by Bitcoin, where non-censorship is maybe the most founding element. Bitcoin is offering (its own version of) freedom as non-censorship, not privacy, nor anonimity. Freedom that you could not possibly been blocked/stopped by anyone, even if everybody knows about you (and, possibly, disagree with you). So, in Bitcoin you are free to the extent that you are unstoppable (and this could be clearly so much debatable). Absense of registry, no 'head of Bitcoin', transparency, extreme blockchain decentralization, proof of work and random time beats, difficulty adjustments, all this are fundamentally instrumental to the non-censorship target (clearly enough, the owner of a registry or the beating of time or the composition of a block will be in censor-like position)

ECB public consultation on digital Euro

Take the current state in the design of a digital Euro.

PRIVACY The recent Eurosystem report on the public consultation on a digital euro (ECB, April 2021) highlights:

' *Privacy emerges as the key feature that a digital euro should offer, according to respondents to the public consultation. This is confirmed both indirectly – by the presence of comments on the importance of privacy in the responses to most questions – and directly – by the choice of two in five citizen respondents to rank privacy first among the nine features proposed in this question"*

(page 11, "I want my payments to remain a private matter" was rank 1 among "Preference for some digital euro features")

ANONIMITY This does not mean that people asked for anonimity.

"Faced with the issue of money laundering and the financing of terrorism, ... Two in five suggest that digital euro transactions should be visible to either intermediaries or the central bank, ... Almost one in ten citizen respondents support selective privacy, where lower-risk small payments under a threshold would remain fully private. About the same share suggest that, following the initial identification of a given user, all transactions should then be private, often referring to offline use and similarities with cash. Only less than one in ten ask for anonymity."

Attribute-based possession (& its limitations)

- standard asymmetric cryptography is for individual/atomic entities (but requires no authority). attribute-based cryptograhpy could be for individual entities, but expresses its full power when used for groups selected by (possibly complex) custom policies
- possession is about a user having a set of attributes capable of decrypting come ciphertext (in possession of the user).

  transfer of possession is about re-encrypting some ciphertext from one group to another (from one policy to another)
- possession (as ownership) could be shared but are always user-centric:

  the possibility for a greater level of customization of this shared possession is what could make attribute-bases possession even more interesting: there could be a single-user possession, a family policy (say, among the parents or, maybe, more inclusive), a family& kids policy or very different company policies with complex policies ruling over shared possession (or ownership, clearly)

Cash-like digital tokens: tentative setup

An example of controlled anonimity for a digital token implementing cash-like features (just 1 suppliers and users, without any intermediary):

MINT 100mln of tokens are minted (of digital cash) from 1 supplier, subdivided into 100 lots of 1mln each. The supplier defines as many attributes and marks each lot by the corresponding policy (here, the policy is very simple)

RATIONALE lots are supposed to be the digital cash available to several users, common shares of ownership with specific rules of possession

OWNERSHIP & POSSESSION so, each user receives her/his tokens from 1 lot:

OWN→ the attribute to unlock that lot +

POSSESS→ her/his set of tokens encrypted according to a possession policy (say, individual)

So, the user has ownership over a certain lot (via her/his attributes), but has also possession on certain amount of tokens (given by her/his decryption potential)

Tracing with controlled anonimity

CONTROL shares of ownership (traceable) should be big enough to hide the users but small enough to control the generated anonimity in ownership (in this case, say, 1k users for 1k average of tokens per user)

TRANSFER transfer is performed by unlocking the tokens of the given lot with attributes, re-encrypting the tokens for another possession policy and re-locking the tokens for the recipients' lot (so, nothing prevents a user from spending the money of another user in the same lot, expect for the fact that she/he has no access to them, that she/he is excluded [a]) (this transfer has some limitations about the interaction of ownership and possession policies)

TRACE transfer of ownership are traceable among the lots, so traceability is there but not so fined-graned as to identify a specific user. transfer of possession are not easy traceable because if the ciphertext wasn't given its policy alongside, the decryptor couldn't possibly know how to decrypt that ciphertext (money could also be lost, in a sense)

CENSORSHIP if necessary, a lot could be invalidated and a new ownership policy issues for that tokens (giving the corresponding new attributes to the users not to be stopped)

[a] but if the tokens were in the clear, another user could steal them and use them, if they were in a lot the robber can unlock: robbery is on the same floor of anonimity, there is no authority tracing, but the same authority cannot defend the robbered user

Conclusions

- attribute-based cryptography have gathered in its 16y of development all it needs to be effective in practical applications and services (also from public institutions)
- in general, looking at its embedding in everyday services, and given its simplicity, its development and its potential this kind of cryptography seems (to me) somehow understimated by the service providers
- also, not so many battle-tested implementations are available (maybe also hardware implementations could be interesting)
- cryptographic onwership (and possession) as defined by attribute-based primitives could be of some interest for many digital assets
- looking forward to:

  → on the math side: LWE-based schemes
  → on the code side: library for attribute-based signatures
  → on the application side: embedding attribute-based ownership into different blockchain designs

References

Amos Beimel, *Secret-sharing schemes: A survey*, Coding and Cryptology (Berlin, Heidelberg) (Yeow Meng Chee, Zhenbo Guo, San Ling, Fengjing Shao, Yuansheng Tang, Huaxiong Wang, and Chaoping Xing, eds.), Springer Berlin Heidelberg, 2011, pp. 11–46.

Allison Lewko and Brent Waters, *Decentralizing attribute-based encryption*, Advances in Cryptology – EUROCRYPT 2011 (Berlin, Heidelberg) (Kenneth G. Paterson, ed.), Springer Berlin Heidelberg, 2011, pp. 568–588.

Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek, *Attribute-based signatures*, Topics in Cryptology – CT-RSA 2011 (Berlin, Heidelberg) (Aggelos Kiayias, ed.), Springer Berlin Heidelberg, 2011, pp. 376–392.

Tatsuaki Okamoto and Katsuyuki Takashima, *Decentralized attribute-based encryption and signatures*, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **E103.A** (2020), no. 1, 41–73.

Yannis Rouselakis and Brent Waters, *Efficient statically-secure large-universe multi-authority attribute-based encryption*, IACR Cryptol. ePrint Arch. **2015** (2015), 16.

Thank you !

LIB https://github.com/marcellop71/mosaic is my lib on github.com implementing the DMA-ABE scheme in [RW15]

CONTACT marcello.paris@gmail.com or LinkedIn

DISCLAIMER The content of this article is solely the responsibility of the author. The views expressed here are those of the author and strictly personal, and do not necessarily reflect the views of his employer.