Cryptographic accumulators

Présentation groupe de travail Grâce

Anaïs Barthoulot [1,2], Olivier Blazy [3], Sébastien Canard [1]

[1]Orange,
[2]Université de Limoges,
[3]École Polytechnique

November 8, 2022

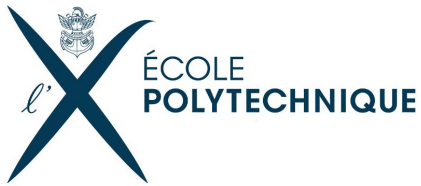
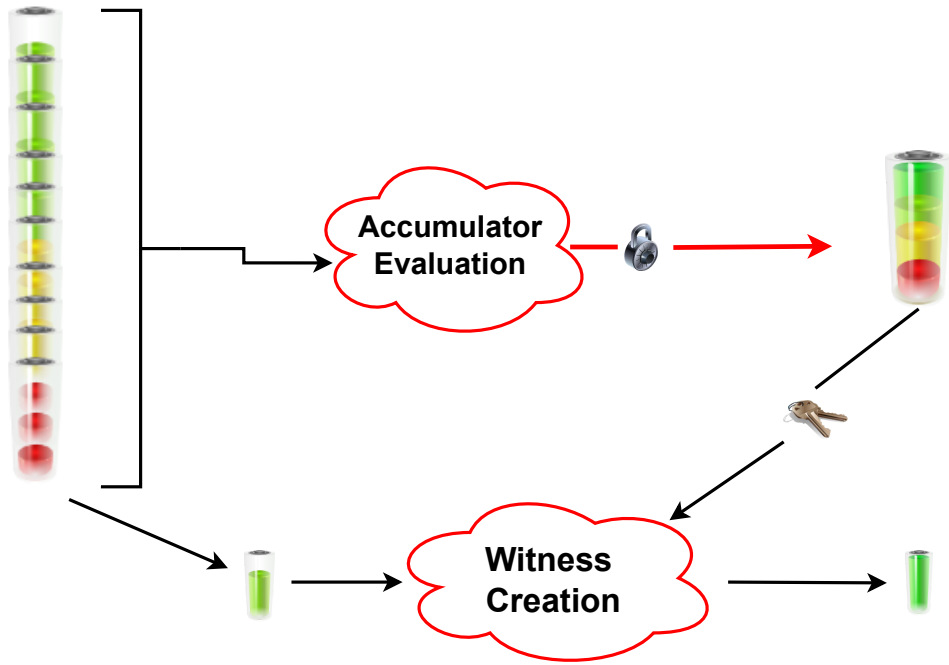


Table of contents

Cryptographic Accumulators

(Simplified) Asymmetric Accumulator [Bd94, DHS15]

- $\mathsf{Gen}(1^\lambda) \to \mathsf{pk}_{\mathsf{acc}}, \mathsf{sk}_{\mathsf{acc}}$
- $\mathsf{Eval}((\mathsf{sk}_{\mathsf{acc}}), \mathsf{pk}_{\mathsf{acc}}, \mathcal{X}) \to \mathsf{acc}$
- $\mathsf{WitCreate}((\mathsf{sk}_{\mathsf{acc}}), \mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}, x) \to \mathsf{wit}_x$ if $x \in \mathcal{X}$, $\perp$ otherwise
- $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}, \mathsf{wit}_x) \to \mathit{TRUE}$ if $x \in \mathcal{X}$, FALSE otherwise



Problems of Accumulators

- Many different definitions
- Several properties: dynamic, universal, ...
- Uniform model from [DHS15] incomplete

$\Longrightarrow$ Propose a uniform model with all properties

$\Longrightarrow$ Propose an instantiation satisfying many of them

(We do not consider symmetric accumulators in this work)

Table of contents

Witness Generation And Sizes

4 ways to generate witnesses:

- Using only public key $\mathsf{pk}_{\mathsf{acc}}$ ([LRY16])
- Using only secret key $\mathsf{sk}_{\mathsf{acc}}$ ([MR15])
- Using public key $\mathsf{pk}_{\mathsf{acc}}$ or secret key $\mathsf{sk}_{\mathsf{acc}}$ to be more efficient ([ATSM09, DHS15])
- Using evaluation key $\mathsf{ek}_{\mathsf{acc}}$, derived from $\mathsf{sk}_{\mathsf{acc}}$ and kept secret by user (AC:GOPTT16)

An important feature for accumulators (as stated in [CKS09]):

- Constant accumulator size (i.e., independent of the number of accumulated elements)
- Constant witness size

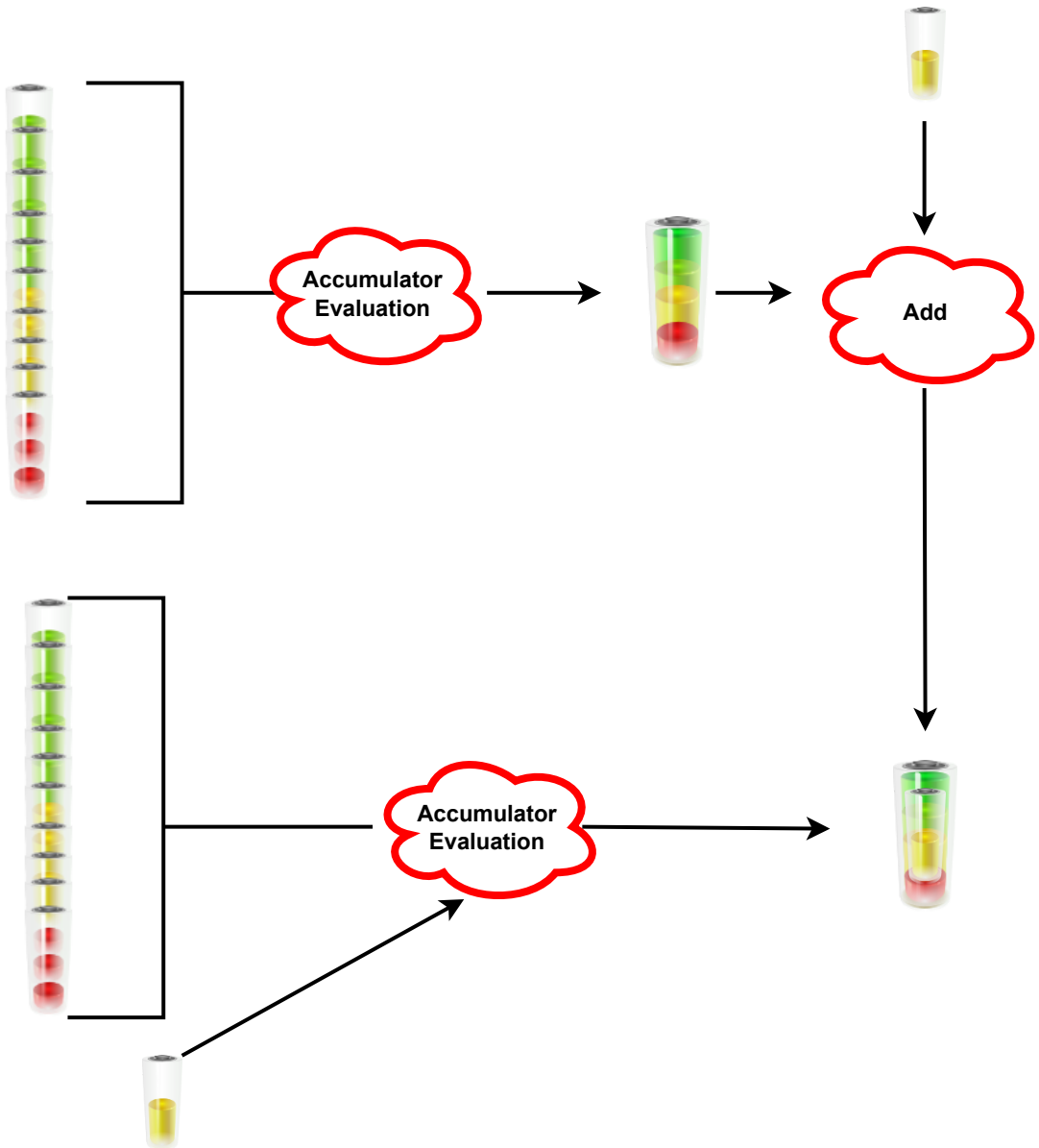
Dynamic Accumulator [CL02]

- **Additive**: add values in the accumulator with algorithm Add
- **Soustractive**: remove values from the accumulator with algorithm Delete
- **Dynamic**: do both with algorithm Update

In all cases, algorithm WitnessUpdate must be efficient (i.e,. witnesses are updated, not computed from the start)

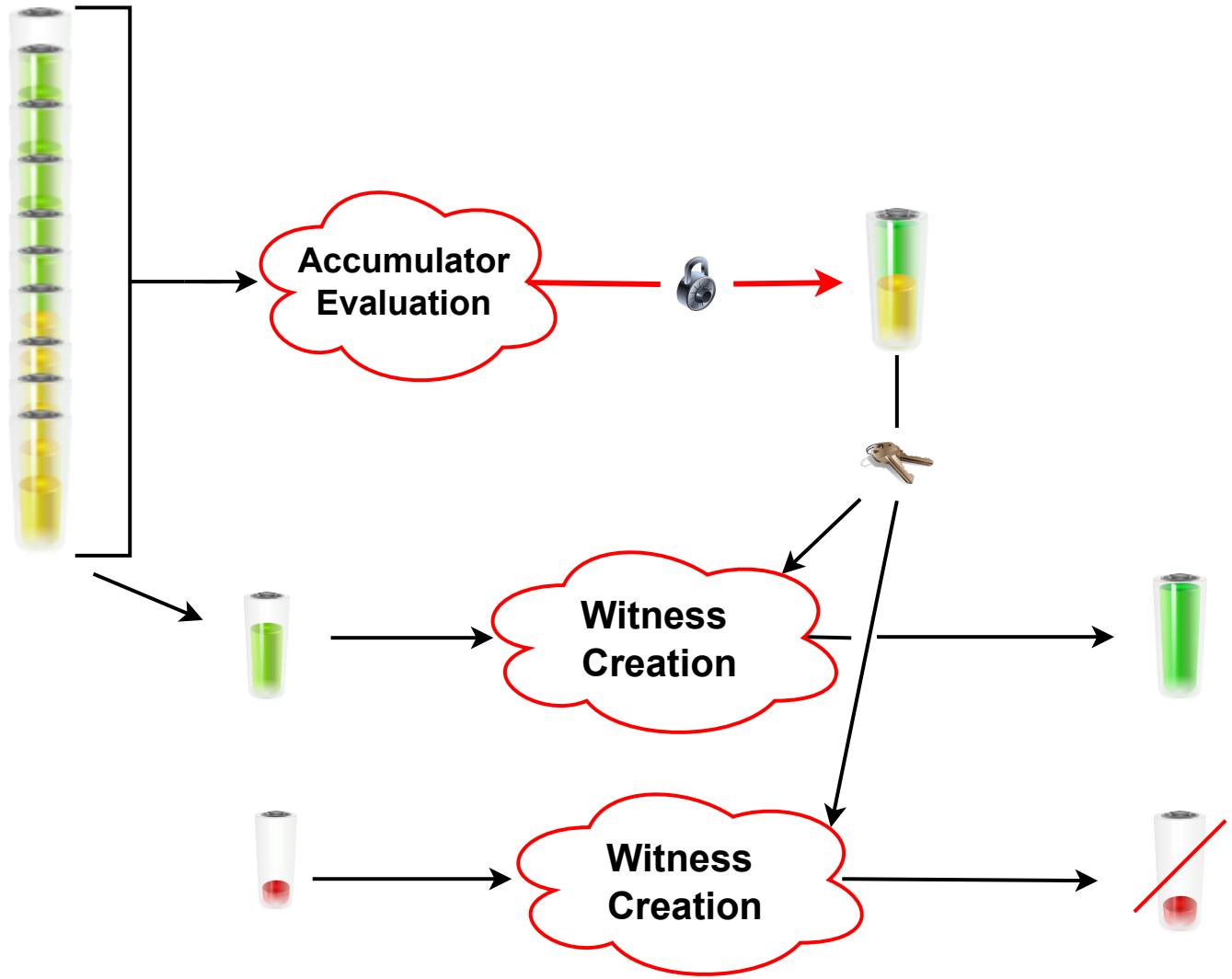
If witnesses and accumulators are updated only with the use of $\mathsf{pk}_{\mathsf{acc}}$, the accumulator scheme is said to be **publicly updatable** [DHS15]

Additive Accumulator



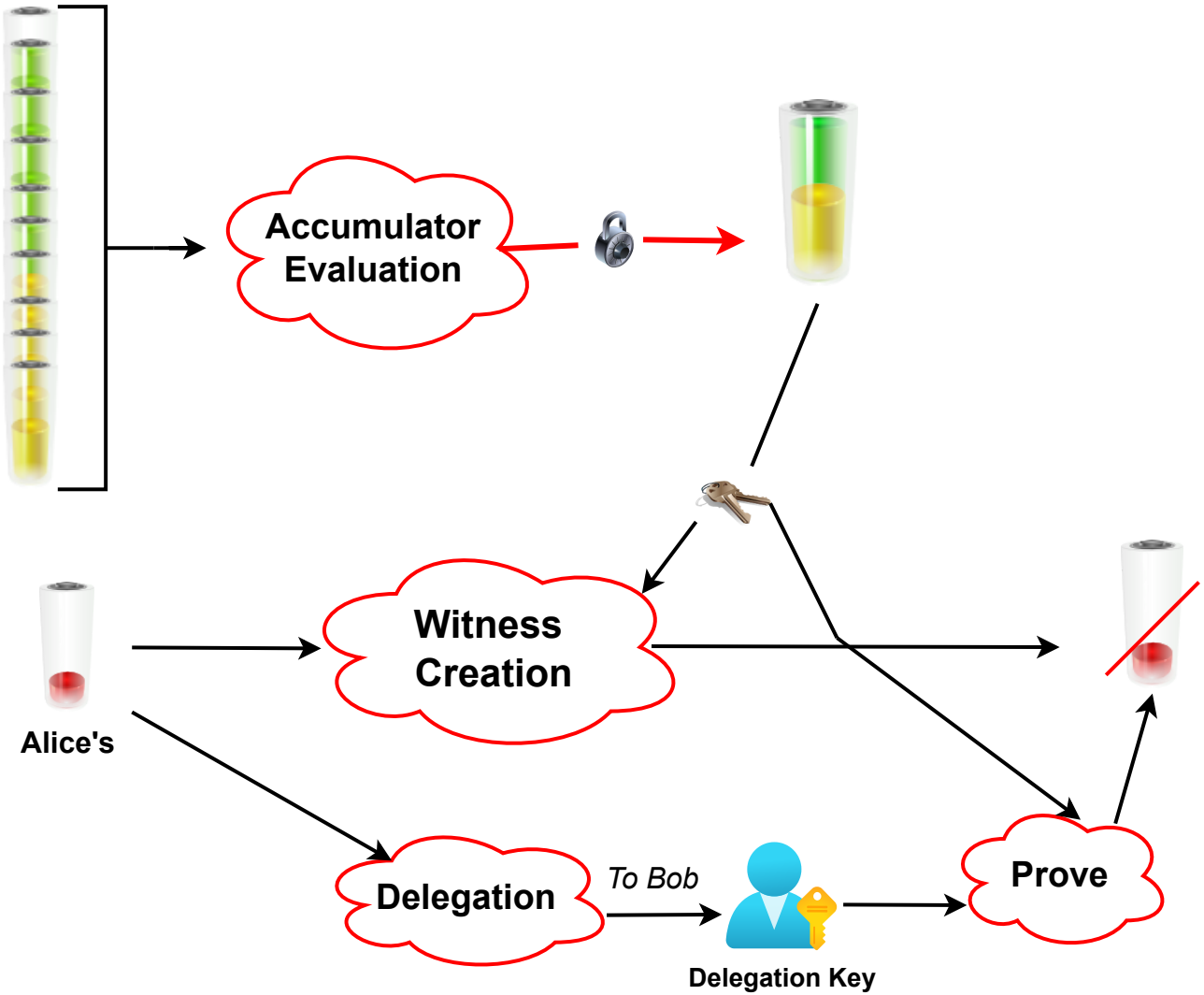
Universal Accumulator [LLX07]

Provides proofs of membership and proofs of non-membership



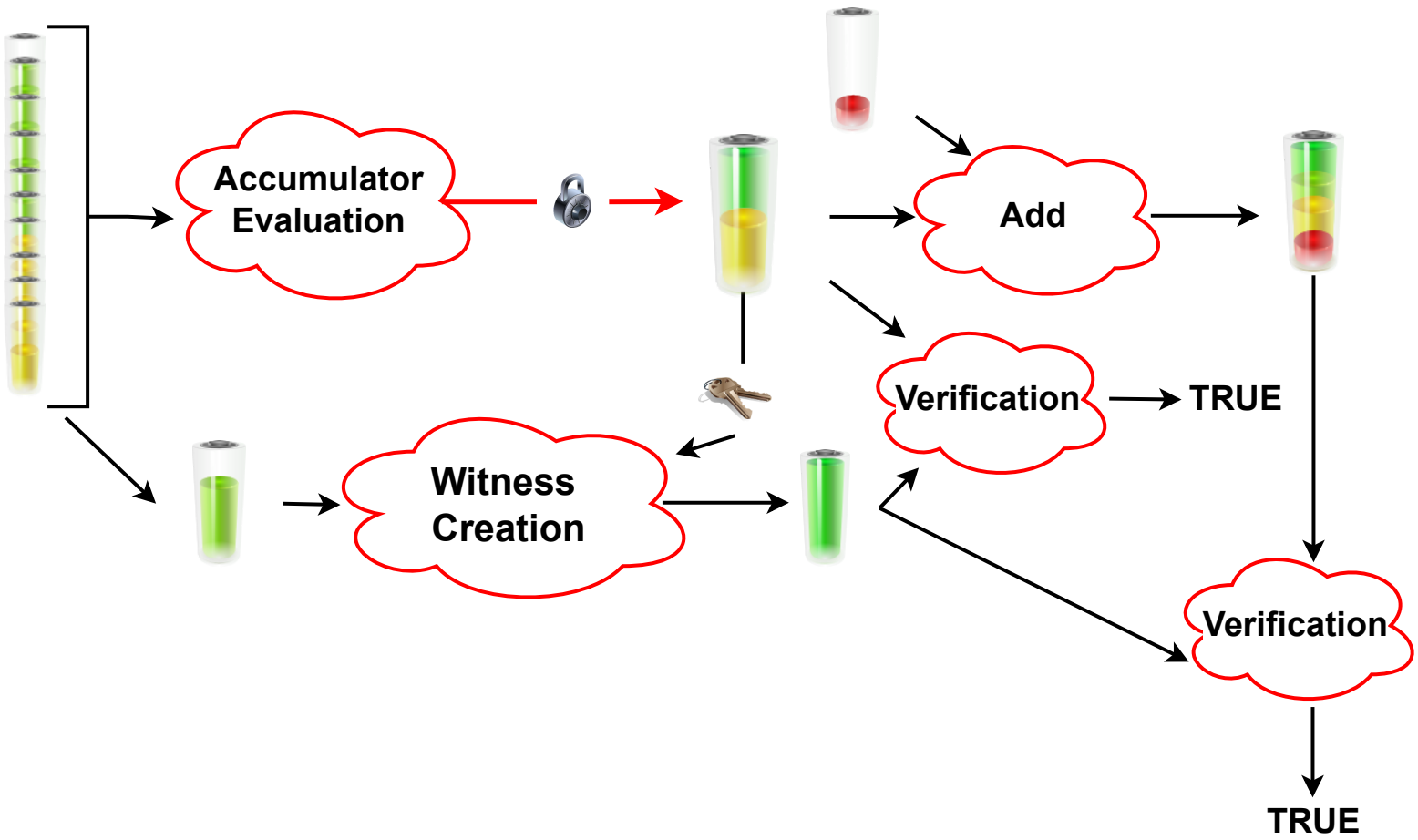
Delegatable Non-Membership Proofs [AN10]

Allows one to prove non-membership of an other's element



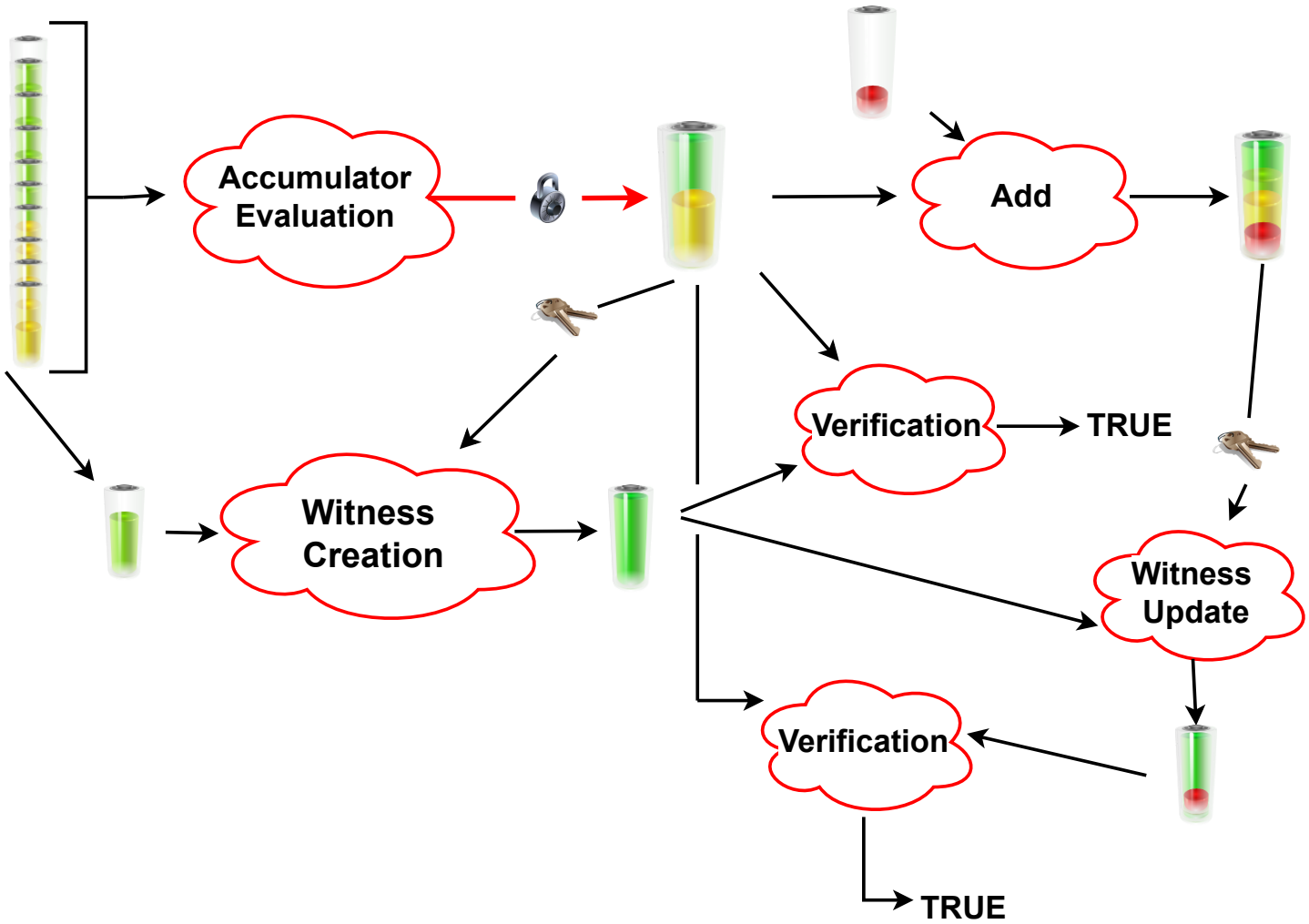
Asynchronous [RY16]: Old Witness Compatibility

Verification of membership is possible with old witness and updated accumulator. *Low Update Frequency*: witness must be updated a sub-linear number of times (in the number of element additions)



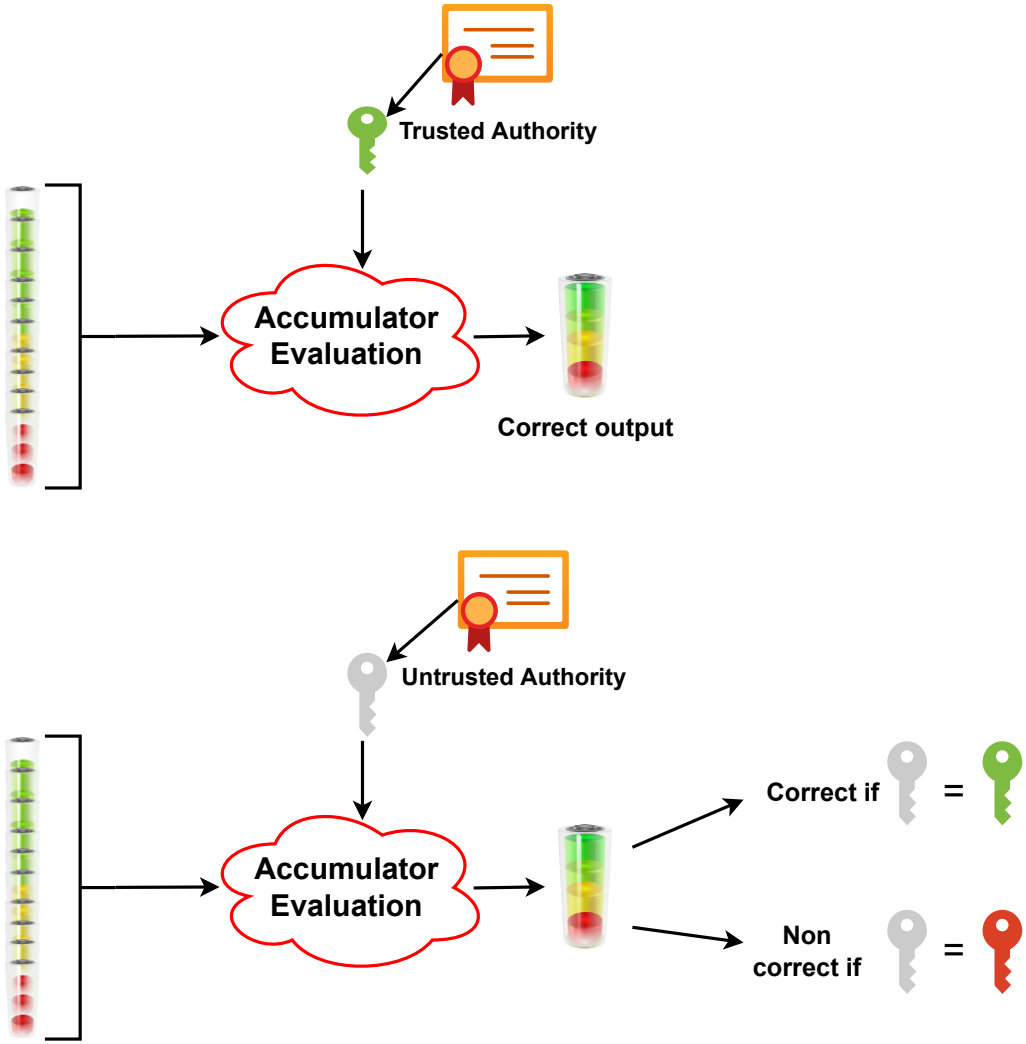
Asynchronous [RY16]: Old Accumulator Compatibility

Verification of membership is possible with updated witness and old accumulator



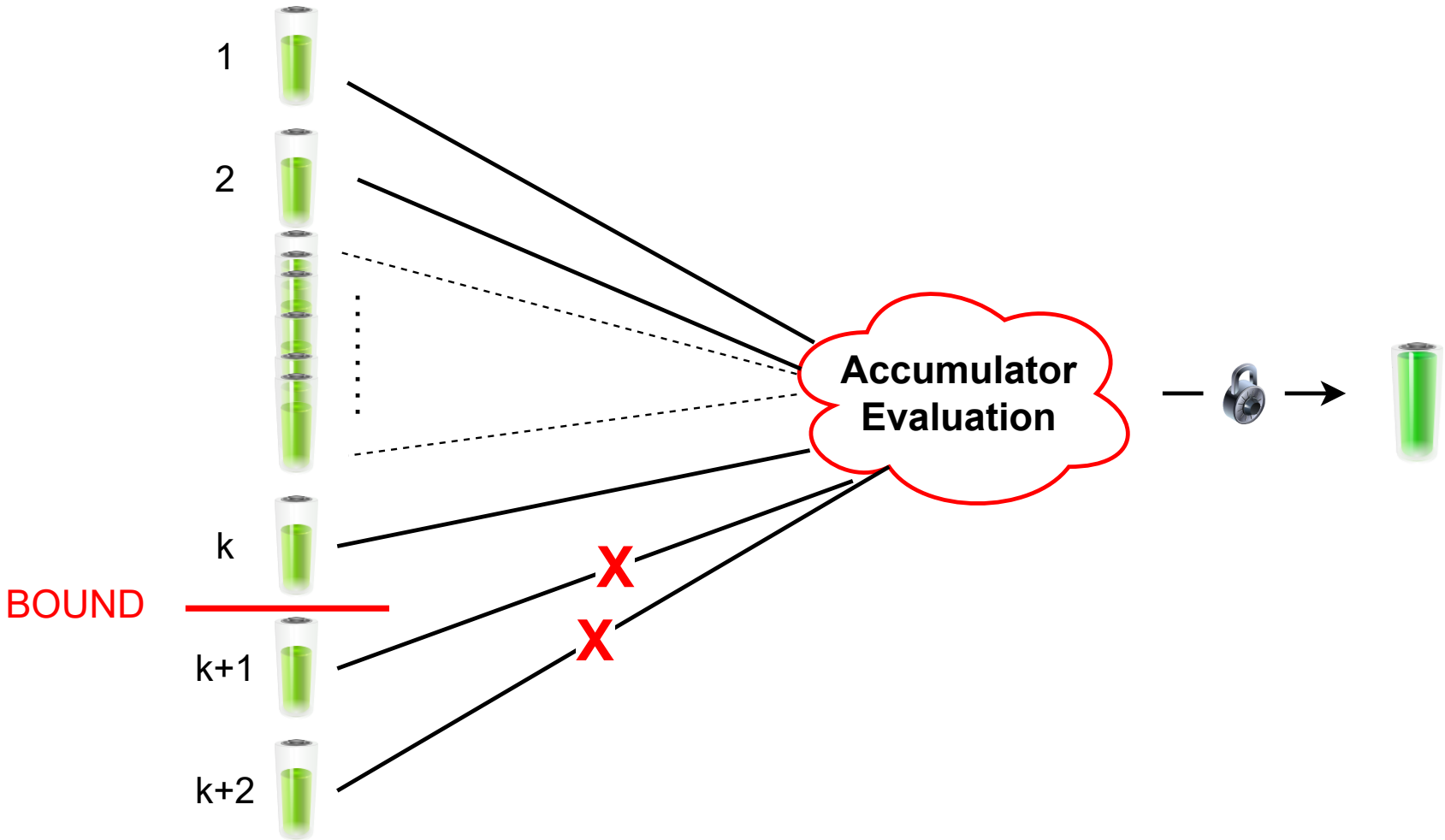
Trusted vs. Non-Trusted Setup [Lip12, DHS15]

When accumulator has sk_{acc}, accumulator authority might not be trusted



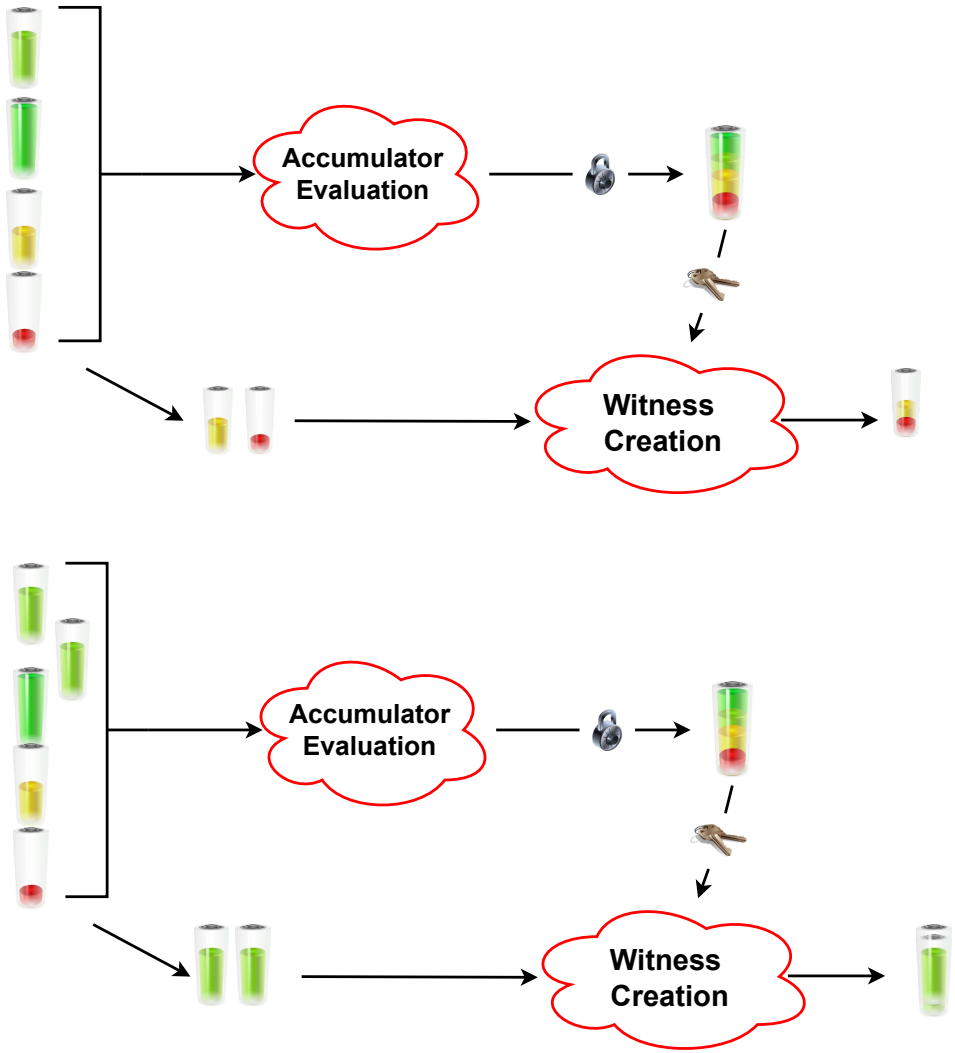
Bounded Accumulator [AWSM07]

Only a given number of elements can be accumulated



Subset Query [DKNS04] and Multiset Setting [LFZ14]

Witnesses for several elements instead of one & elements can be accumulated several times in the accumulator

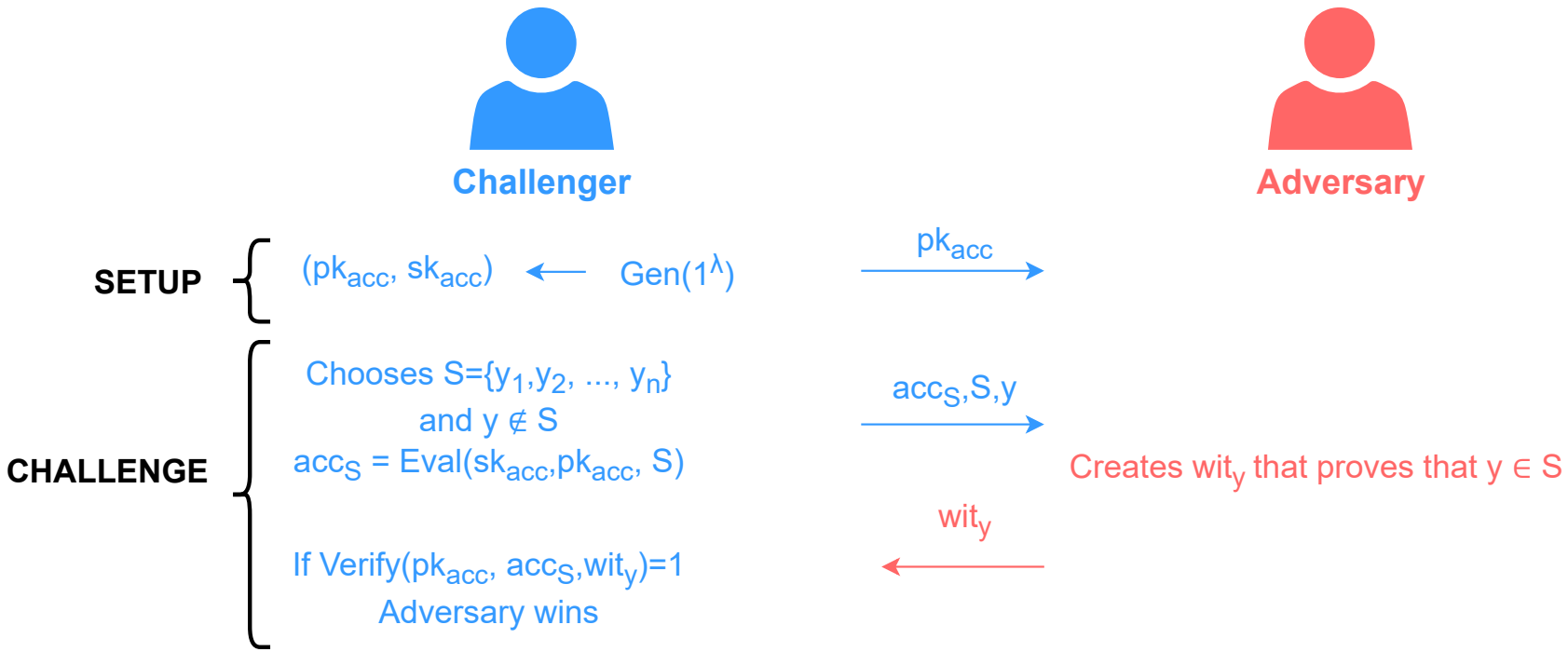


Functionalities and Properties of [DHS15]

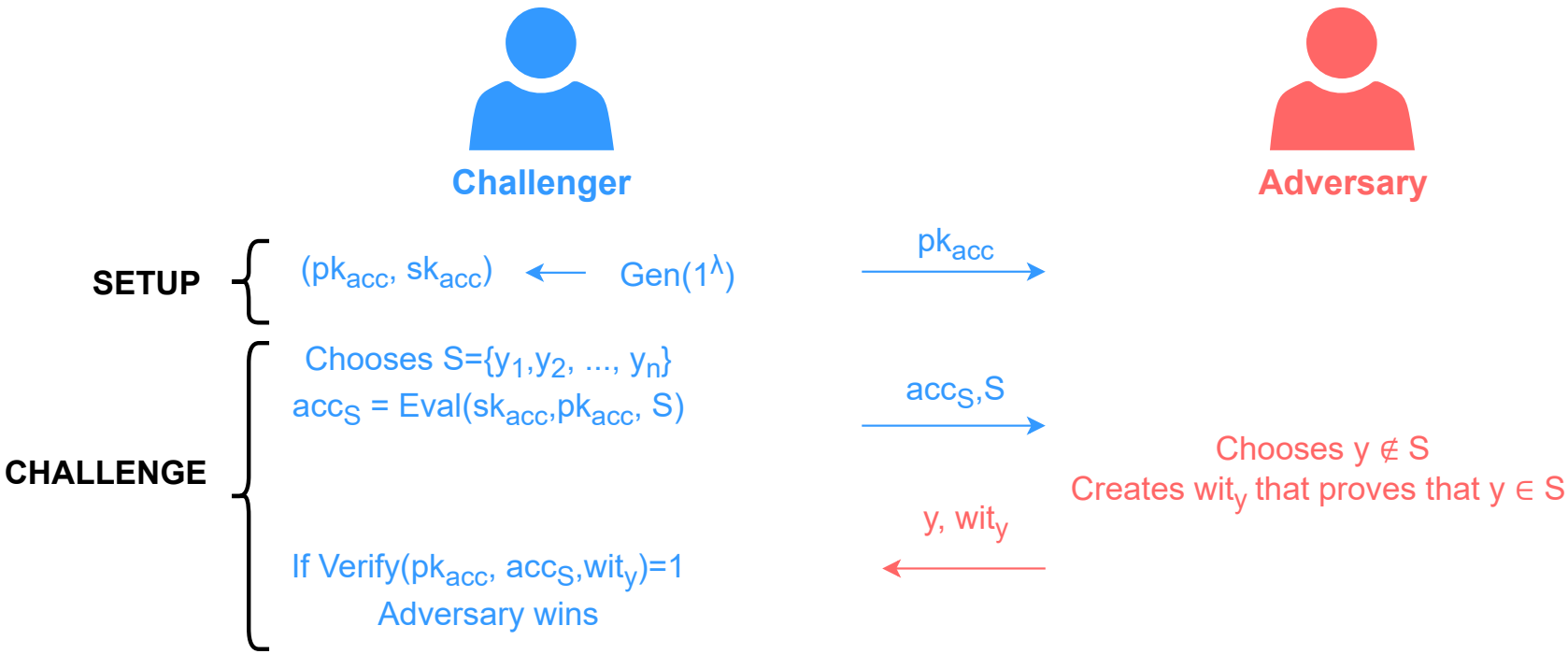
	[DHS15]
Dynamic	$\surd$
Publicly updatable	$\surd$
Universal	$\surd$
Delegatable non-membership proofs	$\times$
Asynchronous	$\times$
Setup	Trusted
Bounded	$\surd$
Subset Query	$\times$
Multiset Setting	$\times$

Table of contents

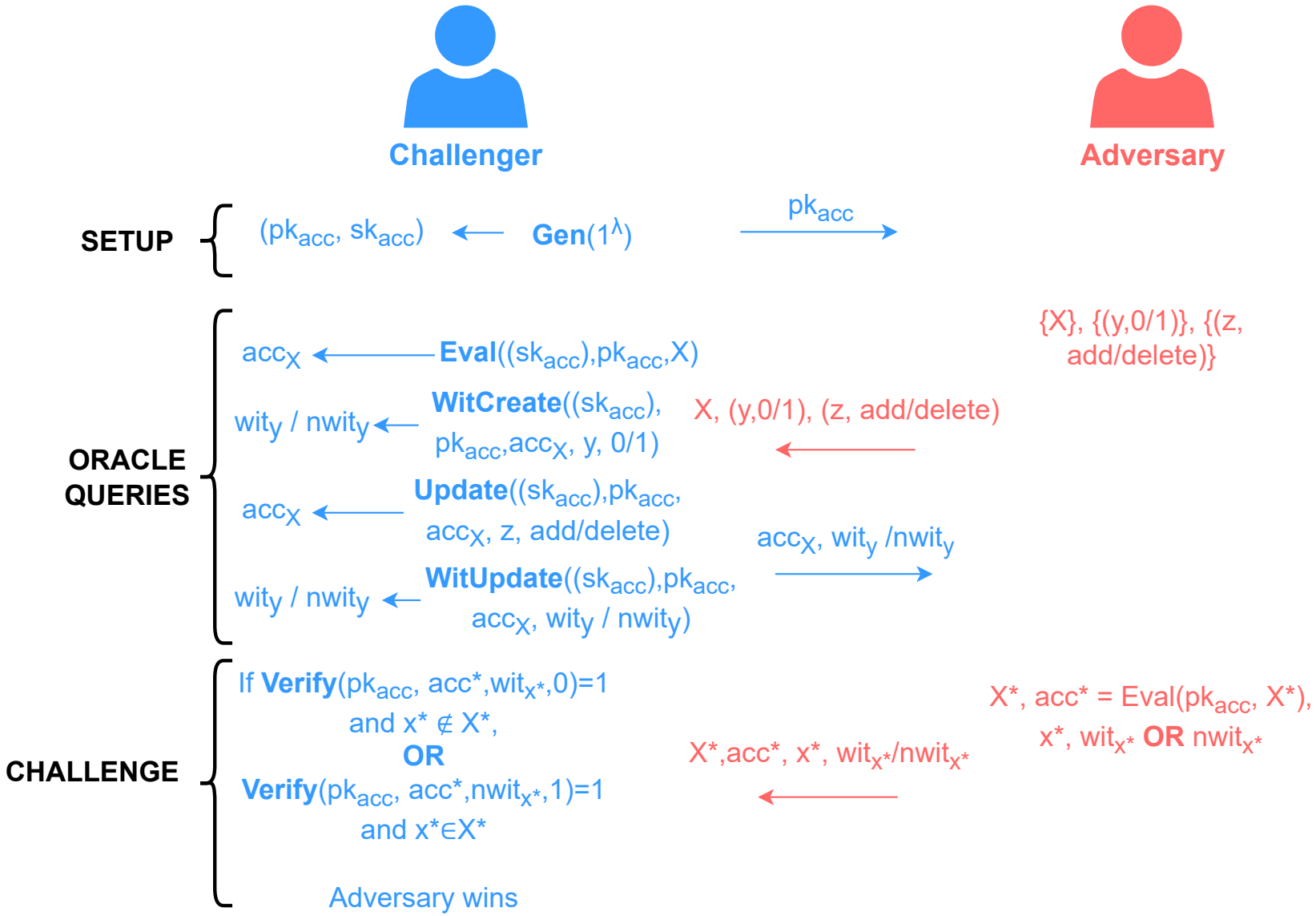
One-Wayness [Bd94]



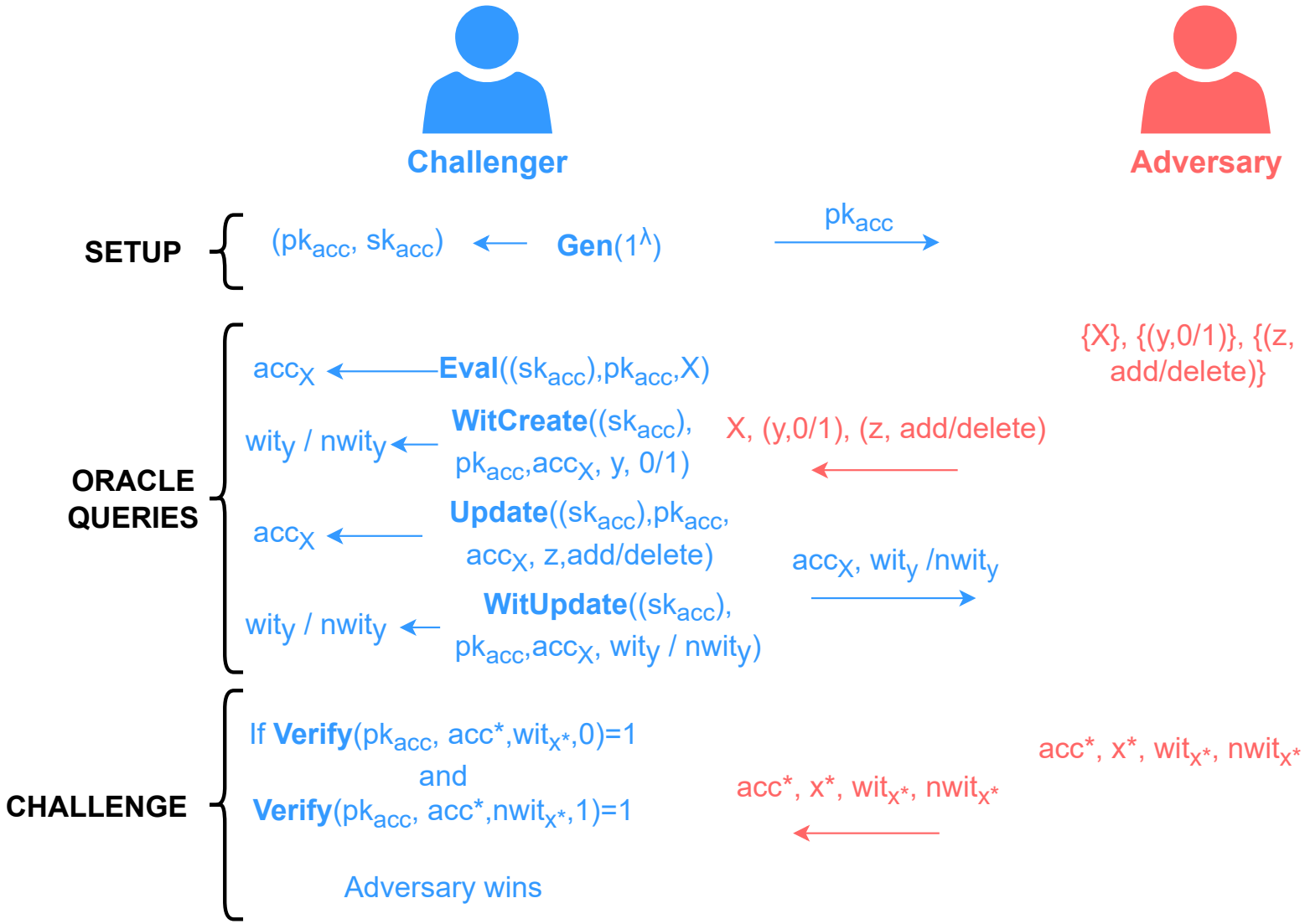
Strong One-Wayness [BP97]



Collision Resistance [BP97]

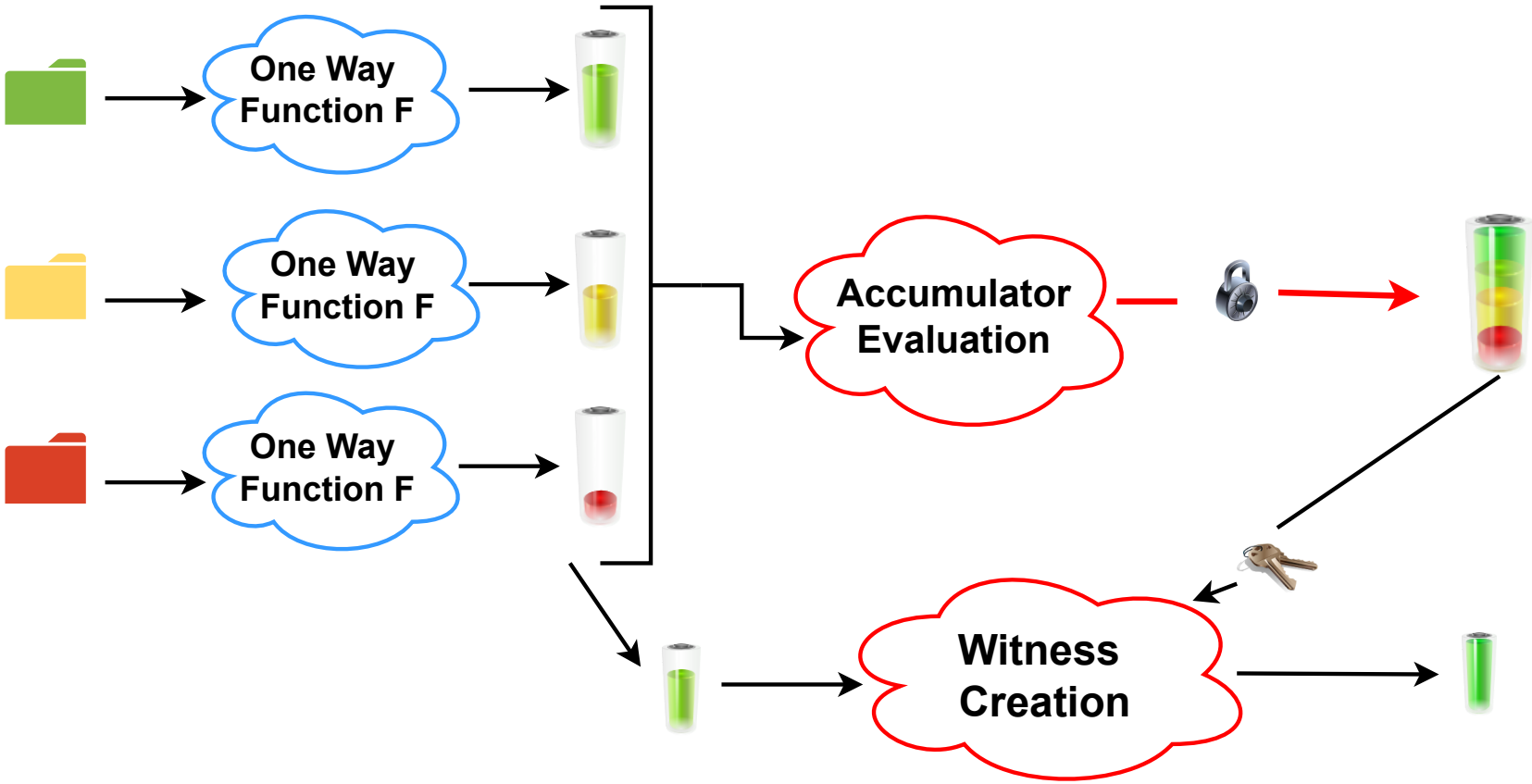


Undeniability [Lip12]

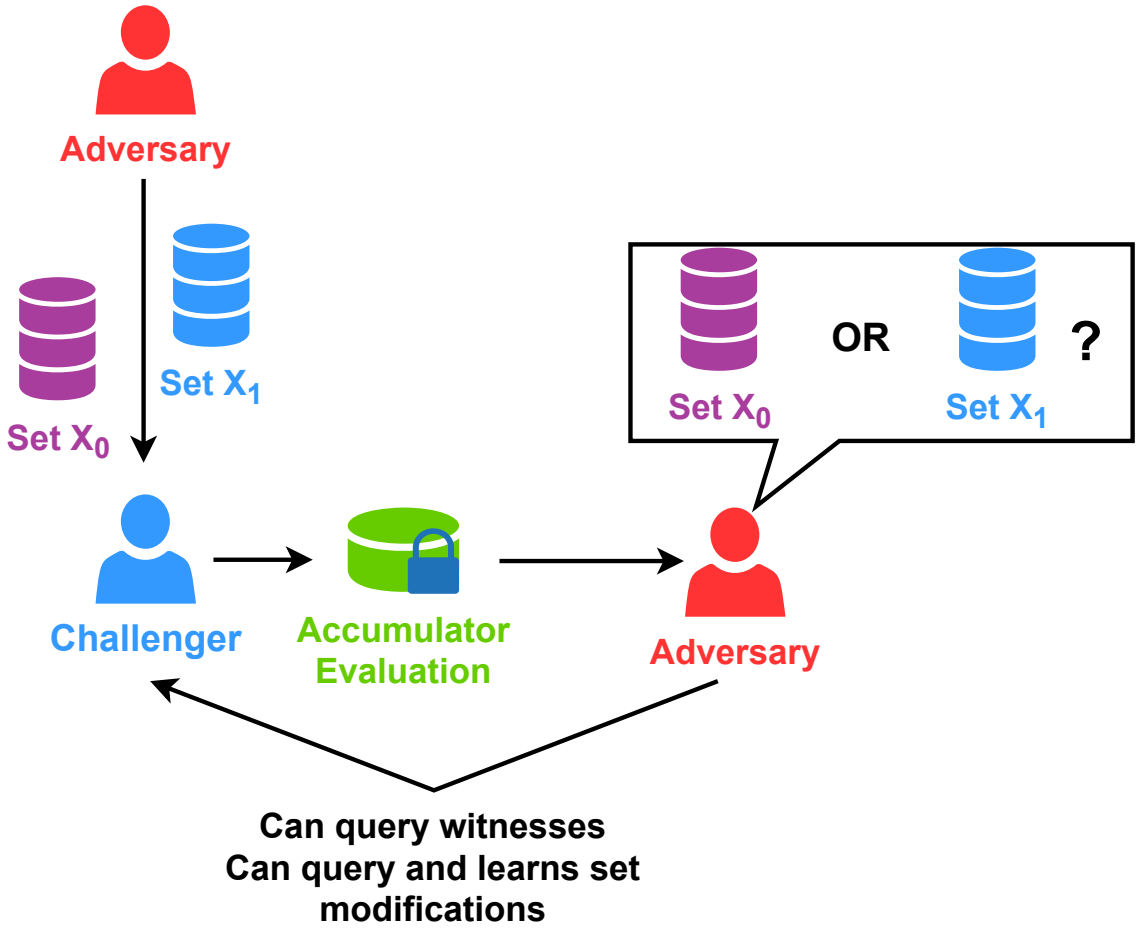


One-Way Domain [DKNS04]

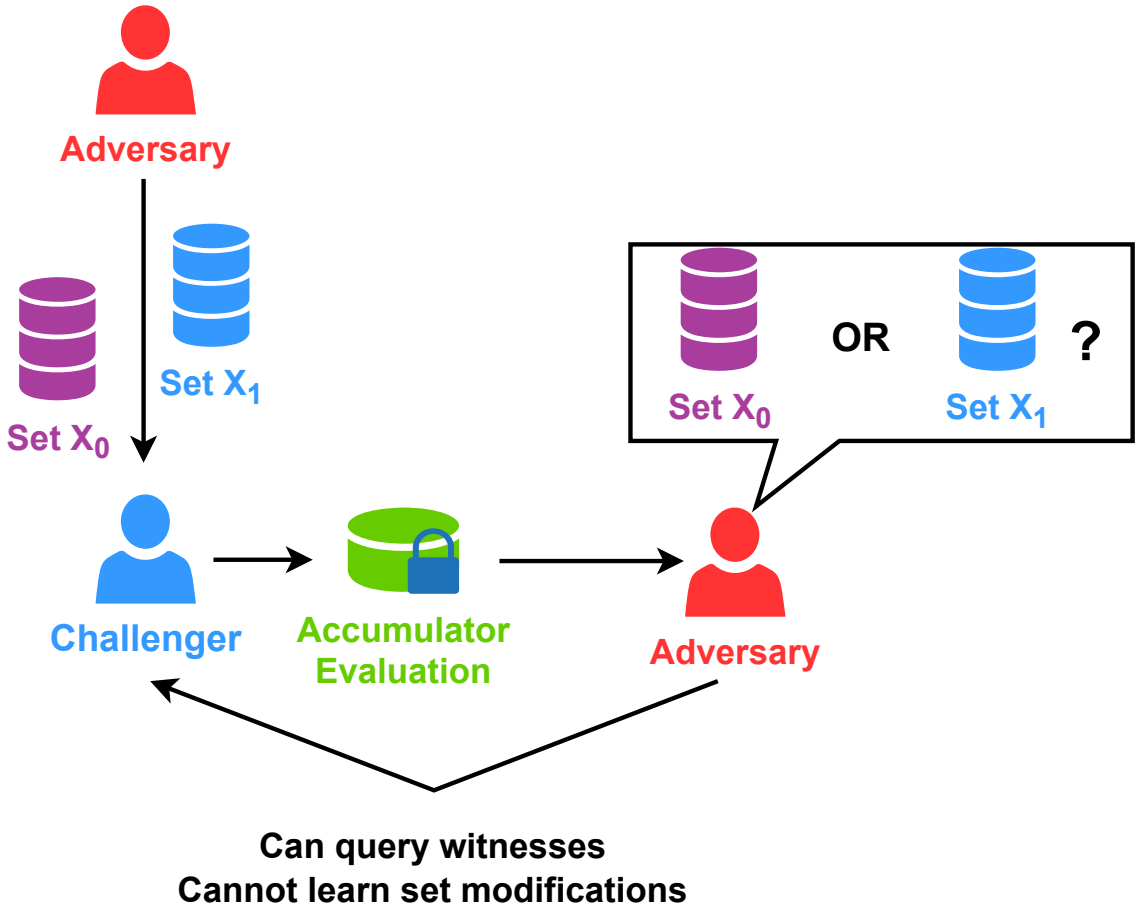
Collision resistant accumulator with set of values that can be accumulated equal to the span of a way one function



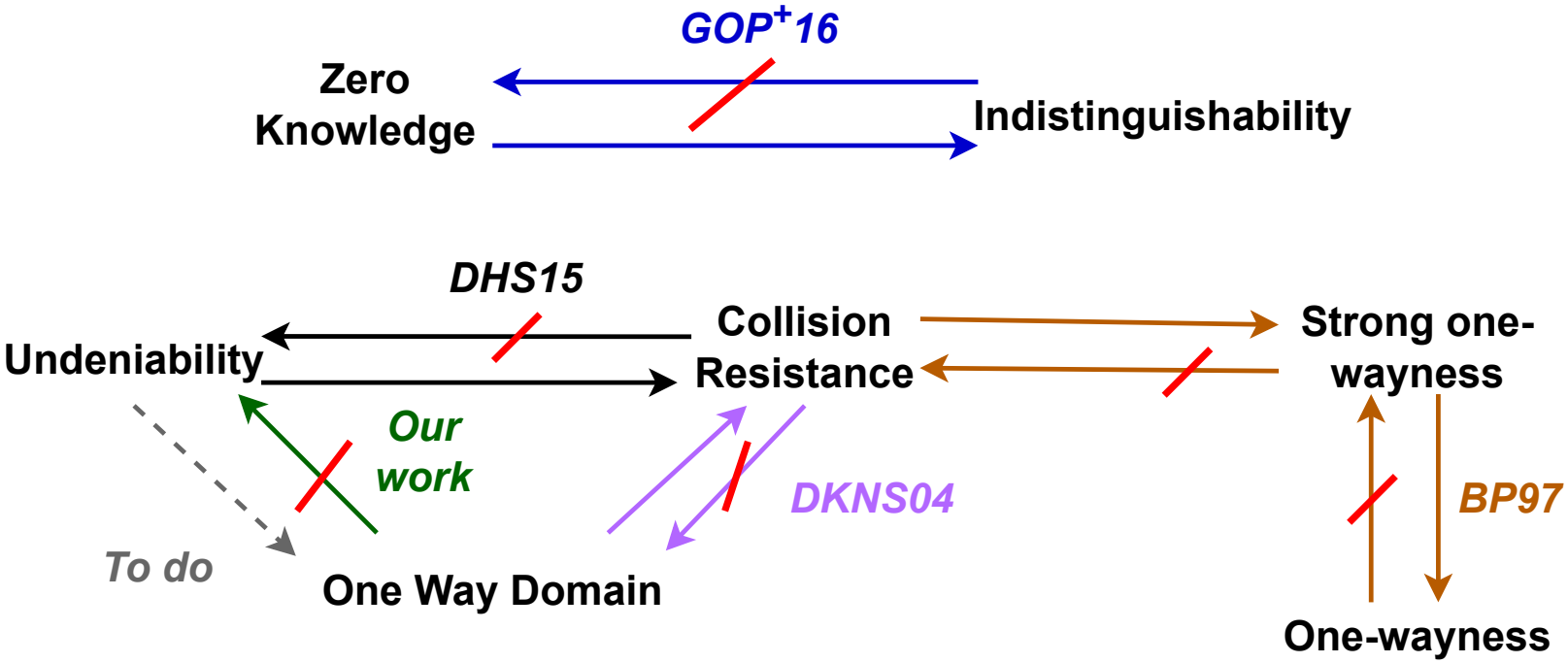
Indistinguishability [DHS15]



Zero-Knowledge [GOP+16]



Relations Between Security Properties



Security Properties of [DHS15]

	[DHS15]
One-wayness	$\Leftarrow$
Strong one-wayness	$\Leftarrow$
Collision resistance	$\surd$
Undeniability	$\surd$
One-way domain	$\times$
Indistinguishability	$\surd$
Zero-knowledge	$\times$

Table of contents

Starting point: [DHS15]'s model

[DHS15]'s model:

- Dynamic
- Publicly updatable
- Universal
- Trusted Setup
- Bounded

- Collision resistant
- Undeniable
- Indistinguishable

Our model

We add to [DHS15]'s model:

- Subset query
- Multiset setting and bounded multiplicity: i.e., element cannot be present in $\mathcal{X}$ more than a given number of times
- Asynchronous
- Delegatable (non-)membership proofs
- Zero-knowledge

Adding Subset Query and Multiset Setting

Different kind of witnesses, that can be expressed with six bases witnesses:

- **Universal**:
  - Membership: $x \in \mathcal{X}$
  - Non-membership: $x \notin \mathcal{X}$
- **Subset query**:
  - Subset membership: $S \subseteq \mathcal{X}$
  - Subset non-membership: $S \not\subseteq \mathcal{X}$
- **Multiset** setting:
  - Multiset membership: $\left\{ \underbrace{x, \cdots, x}_{\text{k times}} \right\} \subseteq \mathcal{X}$
  - Multiset non-membership: $\left\{ \underbrace{x, \cdots, x}_{\text{k times}} \right\} \not\subseteq \mathcal{X}$

Modifying Witness Creation and Verification Algorithms

- Witness creation: additional inputs
  - Boolean Subset: indicates if element or subset
  - Boolean Type: indicates if membership or non-membership (already used in [DHS15])
  - Integer k:
    - ★ Received when witness for an element
    - ★ Indicates multiplicity of the element
    - ★ When k not given or equals to 1, witness not considered as multiset witness

- Verification:
  - Takes also Subset, Type and eventually k
  - If witness is not created for Subset, Type, (k): outputs $\perp$

Modifying Update and Witness Update Algorithms

- Update:
  - If a subset as input: each element is added/removed once
  - If an element as input with an integer k': element is added/removed k' times

- Witness update:
  - Modifies witness according to Subset, Type, k
  - If no concordance between Subset, Type, k and new parameters, it outputs $\perp$

  - *Example*: witness created for x k times in $\mathcal{X}$; x removed once from $\mathcal{X}$; witness $=\perp$

Witnesses

Table: Witness notation according to the associated parameters. $\mathscr{Y}$ either equals to y or $Y = \{y_1, \cdots, y_l\}$.

Name of witness	**Parameters**			**Notation**
	Subset	Type	k	
Unspecified	Unspecified	Unspecified	Unspecified	$\mathsf{uwit}_{\mathscr{Y}}$
Membership unspecified	Unspecified	0	Unspecified	$\mathsf{umwit}_{\mathscr{Y}}$
Non-membership unspecified	Unspecified	1	Unspecified	$\mathsf{unmwit}_{\mathscr{Y}}$
Witness	0	Unspecified	$\times$	$\mathsf{wit}_{\mathscr{Y}}$
Membership witness	0	0	$\times$	$\mathsf{mwit}_{\mathscr{Y}}$
Non-membership witness	0	1	$\times$	$\mathsf{nmwit}_{\mathscr{Y}}$
Subset witness	1	Unspecified	$\times$	$\mathsf{swit}_{\mathscr{Y}}$
Subset membership witness	1	0	$\times$	$\mathsf{smwit}_{\mathscr{Y}}$
Subset non-membership witness	1	1	$\times$	$\mathsf{snmwit}_{\mathscr{Y}}$
Multiset witness	0	Unspecified	Unspecified	$\mathsf{mwit}^{\star}_{\mathscr{Y}}$
Multiset witness	0	Unspecified	k	$\mathsf{mwit}^{k}_{\mathscr{Y}}$
Multiset membership witness	0	0	Unspecified	$\mathsf{mmwit}^{\star}_{\mathscr{Y}}$
Multiset non-membership witness	0	1	Unspecified	$\mathsf{mnmwit}^{\star}_{\mathscr{Y}}$
Multiset membership witness	0	0	k	$\mathsf{mmwit}^{k}_{\mathscr{Y}}$
Multiset non-membership witness	0	1	k	$\mathsf{mnmwit}^{k}_{\mathscr{Y}}$

Verification Functions

- **CompatibilityAcc**($\mathsf{pk}_{\mathsf{acc}}$, (Subset, Type, $\mathscr{Y}$, k)): consistency between $\mathsf{pk}_{\mathsf{acc}}$ and (Subset, Type, $\mathscr{Y}$, k) given as input of algorithm
- **ParametersConc**((($\mathscr{Y}$, Subset, Type), ($\mathscr{Y}'$, Subset$'$, Type$'$)): verifies the equality
- **AddCompatibility**($\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathscr{Y}', \mathsf{k}'$): verifies the possibility of adding an element $\mathscr{Y}'$, with multiplicity k$'$ if $|\mathscr{Y}'| = 1$ to a set $\mathcal{X}$
- **DeleteCompatibility**($\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathscr{Y}', \mathsf{k}'$): verifies the possibility of removing an element $\mathscr{Y}'$, with multiplicity k$'$ if $|\mathscr{Y}'| = 1$ from a set $\mathcal{X}$

Delegatable (Non-)Membership Proofs

- Property first introduced only for non-membership proofs
- We present it for both membership and non-membership proofs
- Witness are now (non-)membership proofs and Verify verifies the correctness of the proofs

Accumulator with delegatable proofs

4 new algorithms:

- Dele: creates delegation key for an element
- Vali: verifies that delegation key is correct
- Rede: creates new delegation key from another
- CompProof: creates a (non-)membership proof from delegation key for accumulator acc

Delegatable Proofs properties

Properties

- **Delegability**: proof computed from delegation key is indisinguishable from normal proof (i.e., a witness)
- **Unlinkability**: cannot distinguish element associated to delegation key
- **Redelegability**: cannot distinguish a delegation key from one computed from Rede
- **Verifiability**: one is able to validate that a delegating key is correctly built

Accumulator with Delegatable Proofs

- Only one instantiation [AN10, AN11]
- There is no generic way to build accumulator with delegatable proofs

  $\Longrightarrow$ We propose one based on [AN10, AN11]'s work

Proof System and Properties

Proof system

R: efficiently computable relation, for parameters Para, witness Wit and statement Sta. A **non-interactive proof system** for R consists of 3 algorithms:

- $\text{Para} \leftarrow \text{Setup}(1^\kappa)$
- $\text{Proof} \leftarrow \text{Prove}(\text{Para}, \text{Sta}, \text{Wit})$
- $\text{Verify}(\text{Para}, \text{Sta}, \text{Proof}) = 1$ if $(\text{Para}, \text{Sta}, \text{Wit}) \in R$ and 0 if $(\text{Para}, \text{Sta}, \text{Wit}) \notin R$

Completness

For all PPT adversary $\mathcal{A}$, $\Pr[\text{Para} \leftarrow \text{Setup}(1^\kappa); (\text{Sta}, \text{Wit}) \leftarrow \mathcal{A}(\text{Para}); \text{Proof} \leftarrow \text{Prove}(\text{Para}, \text{Sta}, \text{Wit}) : \text{Verify}(\text{Para}, \text{Sta}, \text{Proof}) = 1$ if $(\text{Para}, \text{Sta}, \text{Wit}) \in R]$ is overwhelming.

Proof System Properties

Soudness

For all PPT adversary $\mathcal{A}$, $\Pr[\text{Para} \leftarrow \text{Setup}(1^\kappa); (\text{Sta}, \text{Proof}) \leftarrow \mathcal{A}(\text{Para}) : \text{Verify}(\text{Para}, \text{Sta}, \text{Proof}) = 0 \text{ if } (\text{Para}, \text{Sta}, \text{Wit}) \notin R]$ is overwhelming.

Witness indistinguishability

Verifier cannot determine which witness was used in the proof.

Randomizable proof system

Proof system has another algorithm $\text{RandProof}(\text{Para}, \text{Sta}, \text{Proof}) \rightarrow \text{Proof}'$, which is valid and indistinguishable from a proof produced by Prove.

Homomorphic Proof System

Homomorphic proof system [AN10]

(Setup, Prove, Verify) proof system for R and Para

- Subset Π of all (Sta, Wit, Proof) such that
  - $(\text{Para}, \text{Sta}, \text{Wit}) \in R$
  - and $\text{Verify}(\text{Para}, \text{Sta}, \text{Proof}) = 1$
- Operation $+_\Pi : \Pi \times \Pi \to \Pi$

Π is a set of **homomorphic proofs** if $(\Pi, +_\Pi)$ satisfies:

- **Closure**: $(\text{Sta}_1, \text{Wit}_1, \text{Proof}_1) +_\Pi (\text{Sta}_2, \text{Wit}_2, \text{Proof}_2) \in \Pi$
- **Associativity**: $((\text{Sta}_1, \text{Wit}_1, \text{Proof}_1) +_\Pi (\text{Sta}_2, \text{Wit}_2, \text{Proof}_2)) +_\Pi (\text{Sta}_3, \text{Wit}_3, \text{Proof}_3) = (\text{Sta}_1, \text{Wit}_1, \text{Proof}_1) +_\Pi ((\text{Sta}_2, \text{Wit}_2, \text{Proof}_2) +_\Pi (\text{Sta}_3, \text{Wit}_3, \text{Proof}_3))$
- **Commutativity**: $(\text{Sta}_1, \text{Wit}_1, \text{Proof}_1) +_\Pi (\text{Sta}_2, \text{Wit}_2, \text{Proof}_2) = (\text{Sta}_2, \text{Wit}_2, \text{Proof}_2) +_\Pi (\text{Sta}_1, \text{Wit}_1, \text{Proof}_1)$

Delegatable Proofs from Homomorphic Proof System

- Accumulator scheme uses:
  - membership proof system for membership witness
  - non-membership proof system for non-membership witness

- Accumulator should be express as a linear combination of some public elements

- Dele: creates proofs for each public element
- Vali: runs proof system verification algorithm
- Rede: runs proof system randomization algorithm
- CompProof: computes proof for the accumulator by computing linear combination of proofs and statements

Properties

Lemma

Delegability is satisfied thanks to the homomorphics proofs and the randomizable property of the (non-)membership proof system.

Lemma

Unlinkability is satisfied thanks to the witness indistinguishability property of the (non-)membership proof system.

Lemma

Redelagability is satisfied thanks to the randomizable property of the (non-)membership proof system.

Lemma

Verifiability is satisfied thanks to the completeness and soudnness of the (non-)membership proof system.

Functionalities and Properties of Our Model

	Our Model
Dynamic	√
Publicly updatable	√
Universal	√
Delegatable (non-)membership proofs	√
Asynchronous	√
Setup	Trusted
Bounded	√
Subset Query	√
Multiset Setting	√
Collision resistance	√
Undeniability	√
One-way domain	√
Indistinguishability	√
Zero-knowledge	√

Table of contents

[GOP+16]'s scheme

- Based on [Ngu05]'s bilinear accumulator
- Closed to [ATSM09]'s accumulator ...
- ... but different non-membership witnesses ...
- ... thus avoid [BUV21]'s attack on [ATSM09]

- Their scheme is:
  - Universal
  - Dynamic
  - Collision Resistant

- Satisfies: Zero-knowledge

- And uses: symmetric pairing in prime order group

Our scheme

- We proved that [GOP+16]'s scheme also works for
  - Subset query
  - Multisets
- And satisfies one-way domain as proven by [SALY17] for [ATSM09]'s accumulator
- We moved it to asymmetric pairing in prime order group
- We:
  - added delegatable membership proofs at the cost of dynamic property
  - gave intuition for delegatable non-membership proofs
- But we could not add asynchronous

Subset Query and Multiset Setting

In [GOP+16]'s scheme, for $\mathcal{X} = \{x_1, \cdots, x_l\}$:

- Accumulator for $\mathcal{X}$ uses $Ch_{\mathcal{X}}[z] = \prod_{i=1}^{l}(z + x_i)$
- Witness for membership and non-membership uses $Ch_{\mathcal{X}\setminus\{x\}}[z]$
- Computations work for $Ch_{\mathcal{X}\setminus\{x,y\}}[z]$, thus subset queries possible
- If $\exists i, j$ s.t. $x = x_i = x_j$, $x_i, x_j \in \mathcal{X}$, $Ch_{\mathcal{X}}[z]$ still coherent
- Computations work for $Ch_{\mathcal{X}\setminus\{x,x\}}[z]$, thus multiset setting possible

Delegatable Membership Proofs

- In [GOP+16]'s, $s \leftarrow \mathbb{Z}_p = \mathsf{sk}_{\mathsf{acc}}$, $g, g^s, g^{s^2}, \cdots, g^{s^q}$ public
- Accumulator is linear combination of $g, g^s, g^{s^2}, \cdots, g^{s^q}$
- Need homomorphic proofs system: Groth Sahai Proofs are homomorphic [AN10]
- We can rewrite membership verification equation to obtain GS statement, witness and proof, for SXDH based GS
- Then we can build the proof for accumulator with homomorphic operation
- Finally replace membership witnesses by Groth Sahai proofs

Limitations

- The obtained scheme is no longer dynamic: must recompute GS proofs when element added/removed from the accumulator
- Same way does not work for non-membership proofs: cannot express verification equation as GS statement, witness and proof so that GS is homomorphic
- Might be possible to obtain delegatable non-membership proofs by:
  - using GS proofs proving that element does not belong to the accumulator
  - using [BCV15]'s work that deals with non-interactive zero-knowledge proofs of non-membership

Functionalities and Properties of Our Instantiations

	Instantiation 1	Instantiation 2
Dynamic	$\surd$	$\times$
Publicly updatable	$\surd$	$\times$
Universal	$\surd$	$\surd$
Delegatable (non-)membership proofs	$\times$	Membership
Asynchronous	$\times$	$\times$
Setup	Trusted	Trusted
Bounded	$\surd$	$\surd$
Subset Query	$\surd$	$\surd$
Multiset Setting	$\surd$	$\surd$
Collision resistance	$\surd$	$\surd$
Undeniability	$\surd$	$\surd$
One-way domain	$\surd$	$\surd$
Indistinguishability	$\surd$	$\surd$
Zero-knowledge	$\surd$	$\surd$

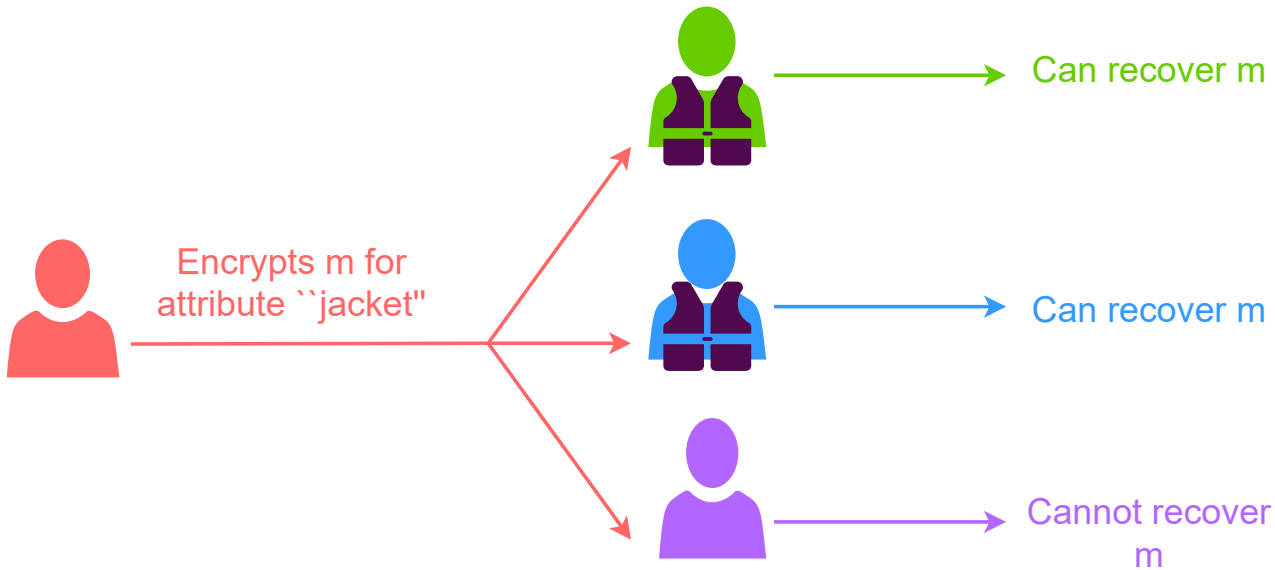
Table of contents

Attribute Based Encryption

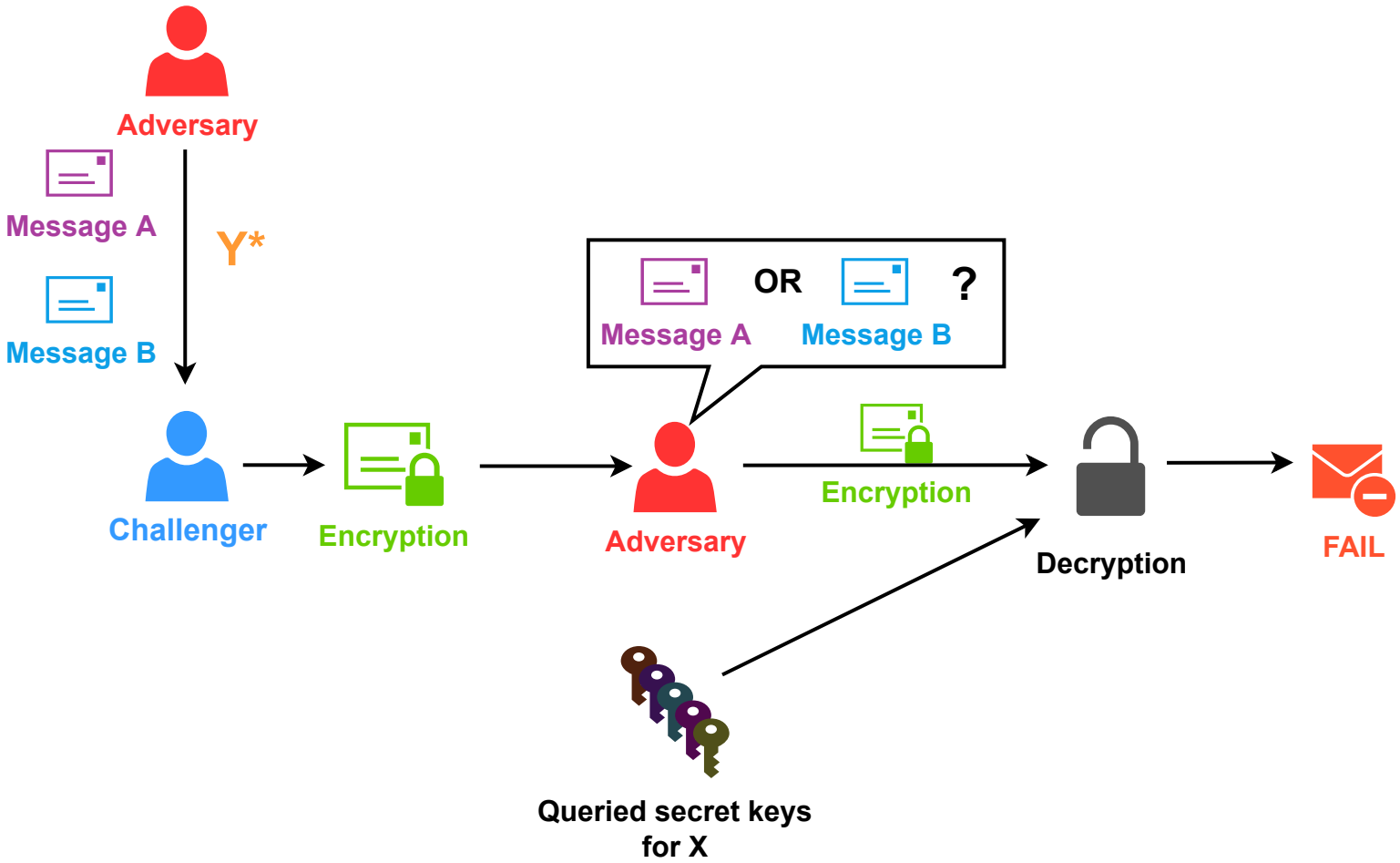
Attribute Based Encryption (ABE) [SW05, AY20]

An ABE scheme for relation $R = \{R_\lambda : A_\lambda \times B_\lambda \to \{0,1\}\}_\lambda$ is defined by:

- $\mathsf{Setup}(1^\lambda) \to (\mathsf{pk}, \mathsf{msk})$
- $\mathsf{Encrypt}(\mathsf{pk}, Y \in B_\lambda, \mathsf{m}) \to \mathsf{ct}$
- $\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, X \in A_\lambda) \to \mathsf{sk}_X$
- $\mathsf{Decrypt}(\mathsf{pk}, \mathsf{sk}_X, X, \mathsf{ct}, Y) \to \mathsf{m}$ if $R(X, Y) = 1$ or $\perp$



ABE Security



Context and What We Want to Do

- ABE suffers from linear (in the number of attributes) ciphertexts and/or secret keys
- We try to build ABE with constant size ciphertext and secret key
- To do so, we use cryptographic accumulators with
  - Private evaluation
  - Public witness generation
  - Constant size accumulated value and witness
  - Collision resistance
- And Smooth Projective Hash Functions satisfying pseudo randomness

Our Ideas

- Using accumulator to accumulate X in the secret key
- Using accumulator to accumulate Y in the ciphertext
- To decrypt, user computes witness for the correct subset of $D \subseteq X$ s.t. $R(D, Y) = 1$
- Verification is done through SPHF
- We will start with policies expressed as conjunction or disjunction only

Adding Properties

- More efficient access policy
- **Subset query**: more efficient schemes
- **Universal accumulator**: non-monotonic access structure
- **Dynamic accumulator**: dynamic ABE
- **Multisets**: threshold cryptography
- **Indistinguishable (or zero-knowledge) accumulator**: ABE with hidden policy

Conclusion

Our contributions

- Complete model of cryptographic accumulators that supersedes [DHS15]'s model
- Generic construction of cryptographic accumulator with delegatable proofs
- Pairing based accumulator satisfying most of the existing functionalities and properties

Future works

- Adapt our model for symmetric accumulators
- Building efficient ABE from accumulators and SPHF
- Adding properties to the underlying accumulator to improve the ABE
- Having (quantum resistant) ABE instantiations

Any questions?

Thanks for your attention!

Bibliography I


Tolga Acar and Lan Nguyen.
Revocation for delegatable anonymous credentials.
Technical Repost MSR-TR-2010-170, Microsoft Research, 2010.


Tolga Acar and Lan Nguyen.
Revocation for delegatable anonymous credentials.
pages 423–440, 2011.


Man Ho Au, Patrick P. Tsang, Willy Susilo, and Yi Mu.
Dynamic universal accumulators for DDH groups and their application to attribute-based anonymous credential systems.
pages 295–308, 2009.


Man Ho Au, Qianhong Wu, Willy Susilo, and Yi Mu.
Compact e-cash from bounded accumulator.
pages 178–195, 2007.


Shweta Agrawal and Shota Yamada.
Optimal broadcast encryption from pairings and LWE.
pages 13–43, 2020.


Olivier Blazy, Céline Chevalier, and Damien Vergnaud.
Non-interactive zero-knowledge proofs of non-membership.
pages 145–164, 2015.

Bibliography II


Josh Cohen Benaloh and Michael de Mare.
One-way accumulators: A decentralized alternative to digital sinatures (extended abstract).
pages 274–285, 1994.


Niko Bari and Birgit Pfitzmann.
Collision-free accumulators and fail-stop signature schemes without trees.
pages 480–494, 1997.


Alex Biryukov, Aleksei Udovenko, and Giuseppe Vitto.
Cryptanalysis of a dynamic universal accumulator over bilinear groups.
CT RSA, 2021.


Jan Camenisch, Markulf Kohlweiss, and Claudio Soriente.
An accumulator based on bilinear maps and efficient revocation for anonymous credentials.
pages 481–500, 2009.


Jan Camenisch and Anna Lysyanskaya.
Dynamic accumulators and application to efficient revocation of anonymous credentials.
pages 61–76, 2002.


David Derler, Christian Hanser, and Daniel Slamanig.
Revisiting cryptographic accumulators, additional properties and relations to other primitives.
pages 127–144, 2015.

Bibliography III


Yevgeniy Dodis, Aggelos Kiayias, Antonio Nicolosi, and Victor Shoup.
Anonymous identification in ad hoc groups.
pages 609–626, 2004.


Esha Ghosh, Olga Ohrimenko, Dimitrios Papadopoulos, Roberto Tamassia, and Nikos Triandopoulos.
Zero-knowledge accumulators and set algebra.
pages 67–100, 2016.


Helger Lipmaa, Prastudy Fauzi, and Bingsheng Zhang.
Efficient non-interactive zero knowledge arguments for set operations.
International Conference on Financial Cryptography and Data Security, 2014.


Helger Lipmaa.
Secure accumulators from euclidean rings without trusted setup.
pages 224–240, 2012.


Jiangtao Li, Ninghui Li, and Rui Xue.
Universal accumulators with efficient nonmembership proofs.
pages 253–269, 2007.


Benoît Libert, Somindu C. Ramanna, and Moti Yung.
Functional commitment schemes: From polynomial commitments to pairing-based accumulators from simple assumptions.
pages 30:1–30:14, 2016.

Bibliography IV


Jhanwar Mahabir and Safavi-Naini Reihaneh.
Compact accumulator using lattices.
International Conference on Security, Privacy, and Applied Cryptography Engineering, 2015.


Lan Nguyen.
Accumulators from bilinear pairings and applications.
CT-RSA, 2005.


Leonid Reyzin and Sophia Yakoubov.
Efficient asynchronous accumulators for distributed pki.
Internarional Conference on Security and Cryptography fo Networks, 2016.


Shi-Feng Sun, Man Ho Au, Joseph K. Liu, and Tsz Hon Yuen.
RingCT 2.0: A compact accumulator-based (linkable ring signature) protocol for blockchain cryptocurrency monero.
pages 456–474, 2017.


Amit Sahai and Brent R. Waters.
Fuzzy identity-based encryption.
pages 457–473, 2005.