TU Graz

Revisiting Cryptographic Accumulators, Additional Properties and Relations to other Primitives

David Derler, Christian Hanser, and Daniel Slamanig,
IAIK, Graz University of Technology

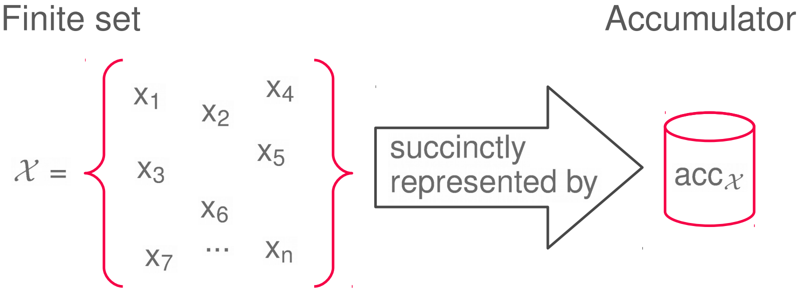
April 21, 2015

Outline

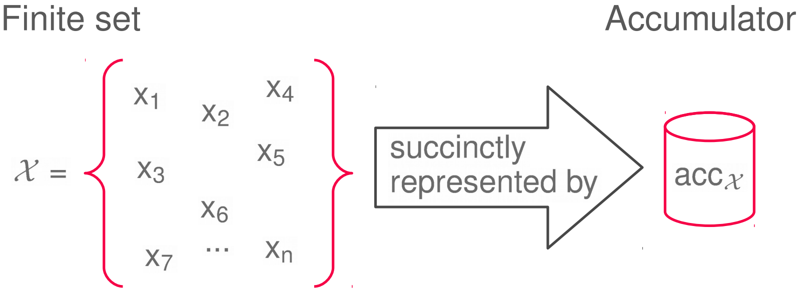
Outline

1. Introduction
2. A Unified Formal Model
3. Accumulators from Zero-Knowledge Sets
4. Black-Box Construction of Commitments

Static Accumulators



Static Accumulators



Witnesses wit_x certifying membership of x in $\mathrm{acc}_{\mathcal{X}}$

- Efficiently computable $\forall\, x \in \mathcal{X}$
- Intractable to compute $\forall\, x \notin \mathcal{X}$

A Simple Example - RSA Accumulator

- RSA modulus N
- Accumulator for $\mathcal{X} = \{x_1, \ldots, x_n\}$
  - $\mathsf{acc}_{\mathcal{X}} \leftarrow g^{x_1 \cdot \ \cdots \ \cdot x_{i-1} \cdot x_i \cdot x_{i+1} \cdot \ \cdots \ \cdot x_n} \mod N$

A Simple Example - RSA Accumulator

- RSA modulus N
- Accumulator for $\mathcal{X} = \{x_1, \ldots, x_n\}$
  - $\mathsf{acc}_{\mathcal{X}} \leftarrow g^{x_1 \cdot \ \cdots \ \cdot x_{i-1} \cdot x_i \cdot x_{i+1} \cdot \ \cdots \ \cdot x_n} \mod N$
- Witness for x_i:
  - $\mathsf{wit}_{x_i} \leftarrow g^{x_1 \cdot \ \cdots \ \cdot x_{i-1} \cdot x_{i+1} \cdot \ \cdots \ \cdot x_n} \mod N$

A Simple Example - RSA Accumulator

- RSA modulus N
- Accumulator for $\mathcal{X} = \{x_1, \ldots, x_n\}$
  - $\mathrm{acc}_{\mathcal{X}} \leftarrow g^{x_1 \cdot \ \cdots \ \cdot x_{i-1} \cdot x_i \cdot x_{i+1} \cdot \ \cdots \ \cdot x_n} \mod N$
- Witness for x_i:
  - $\mathrm{wit}_{x_i} \leftarrow g^{x_1 \cdot \ \cdots \ \cdot x_{i-1} \cdot x_{i+1} \cdot \ \cdots \ \cdot x_n} \mod N$
- Verify witness:
  - Check whether $(\mathrm{wit}_{x_i})^{x_i} \equiv \mathrm{acc}_{\mathcal{X}} \mod N$.

A Simple Example - RSA Accumulator

- RSA modulus N
- Accumulator for $\mathcal{X} = \{x_1, \ldots, x_n\}$
  - $\mathsf{acc}_{\mathcal{X}} \leftarrow g^{x_1 \cdot \ldots \cdot x_{i-1} \cdot x_i \cdot x_{i+1} \cdot \ldots \cdot x_n} \mod N$
- Witness for x_i:
  - $\mathsf{wit}_{x_i} \leftarrow g^{x_1 \cdot \ldots \cdot x_{i-1} \cdot x_{i+1} \cdot \ldots \cdot x_n} \mod N$
- Verify witness:
  - Check whether $(\mathsf{wit}_{x_i})^{x_i} \equiv \mathsf{acc}_{\mathcal{X}} \mod N$.
- Witness for $y \notin \mathcal{X}$
  - Would imply breaking strong RSA
  - ...unless factorization of N is known.

Dynamic and Universal Features

Dynamically add/delete elements

- ...to/from accumulator $\mathsf{acc}_{\mathcal{X}}$
- Update witnesses accordingly
- All updates independent of $|\mathcal{X}|$

Dynamic and Universal Features

Dynamically add/delete elements

- ...to/from accumulator $\mathrm{acc}_{\mathcal{X}}$
- Update witnesses accordingly
- All updates independent of $|\mathcal{X}|$

Universal features

- Demonstrate non-membership
- Non-membership witness $\underline{\mathrm{wit}}_x$
  - Efficiently computable $\forall\ x \notin \mathrm{acc}_{\mathcal{X}}$
  - Intractable to compute $\forall\ x \in \mathrm{acc}_{\mathcal{X}}$

Motivation

Accumulators widely used in various applications

- e.g., credential revocation, malleable signatures, ...
- Previous models tailored to specific constructions
  - Different features
  - Private/public updatability

Motivation

Accumulators widely used in various applications

- e.g., credential revocation, malleable signatures, ...
- Previous models tailored to specific constructions
  - Different features
  - Private/public updatability

Thus, accumulators not usable as black-boxes

- Limited exchangeability when used in other constructions
- Relations to other primitives hard to study

Contribution

- Unified formal model for
  - **Static/dynamic/universal** accumulators
  - Introduces **randomized** and **bounded** accumulators
  - Introduces **indistinguishability**
  - Includes **undeniability**

Contribution

- Unified formal model for
  - **Static/dynamic/universal** accumulators
  - Introduces **randomized** and **bounded** accumulators
  - Introduces **indistinguishability**
  - Includes **undeniability**
- First constructions fulfilling new notions
  - First **indistinguishable, dynamic** acc
  - First **undeniable, indistinguishable, universal** acc

Contribution

- Unified formal model for
  - **Static/dynamic/universal** accumulators
  - Introduces **randomized** and **bounded** accumulators
  - Introduces **indistinguishability**
  - Includes **undeniability**
- First constructions fulfilling new notions
  - First **indistinguishable, dynamic** acc
  - First **undeniable, indistinguishable, universal** acc
- **Black-box relations** to commitments and ZK-sets

Contribution

- Unified formal model for
  - **Static/dynamic/universal** accumulators
  - Introduces **randomized** and **bounded** accumulators
  - Introduces **indistinguishability**
  - Includes **undeniability**
- First constructions fulfilling new notions
  - First **indistinguishable, dynamic** acc
  - First **undeniable, indistinguishable, universal** acc
- **Black-box relations** to commitments and ZK-sets
- **Exhaustive classification** of existing schemes (see Paper)

Outline

1. Introduction
2. A Unified Formal Model
3. Accumulators from Zero-Knowledge Sets
4. Black-Box Construction of Commitments

Algorithms

Static Accumulators - Algorithms

- *Gen*
- *Eval*
- *WitCreate*
- *Verify*

Algorithms

Static Accumulators - Algorithms

- *Gen*
- *Eval*
- *WitCreate*
- *Verify*

We call accumulators

- t-bounded, if an upper bound for the set size exists
- randomized, if *Eval* is probabilistic
  - Eval_r to make used randomness explicit

Algorithms

Static Accumulators - Algorithms

- *Gen*
- *Eval*
- *WitCreate*
- *Verify*

We call accumulators

- *t*-bounded, if an upper bound for the set size exists
- randomized, if *Eval* is probabilistic
  - $Eval_r$ to make used randomness explicit

Dynamic Accumulators additionally provide

- *Add*
- *Delete*
- *WitUpdate*

Algorithms - Universal Accumulators

Static or dynamic accumulator, but in addition

- *WitCreate* and *Verify* take additional parameter *type*

Algorithms - Universal Accumulators

Static or dynamic accumulator, but in addition

- *WitCreate* and *Verify* take additional parameter *type*
  - Membership ($type = 0$) vs. non-membership mode ($type = 1$)

Algorithms - Universal Accumulators

Static or dynamic accumulator, but in addition

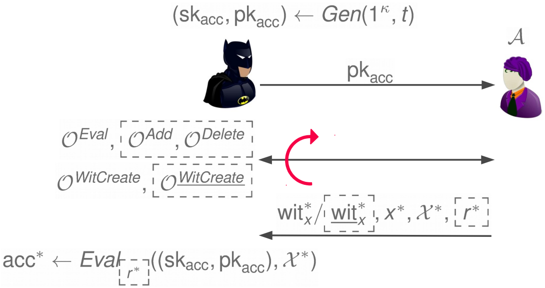
- *WitCreate* and *Verify* take additional parameter *type*
  - Membership ($type = 0$) vs. non-membership mode ($type = 1$)
- For dynamic accumulator schemes
  - The same additionally applies to *WitUpdate*

Security

- Correctness
- Collision freeness
- Undeniability
- Indistinguishability

Security - Collision Freeness

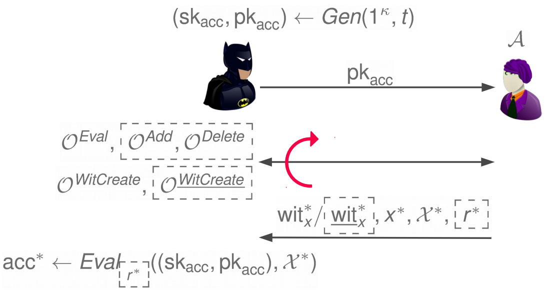
Experiment $\mathbf{Exp}^{cf}_{\kappa}(\cdot)$:



David Derler, IAIK, Graz University of Technology
April 21, 2015

Security - Collision Freeness

Experiment $\mathbf{Exp}^{cf}_{\kappa}(\cdot)$:

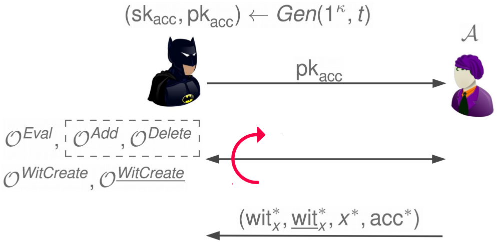


- $\mathcal{A}$ wins if
  - wit^*_x is membership witness for non-member, or
  - $\underline{\mathsf{wit}}^*_x$ is non-membership witness for member

Security - Undeniability

Defined for universal accumulators

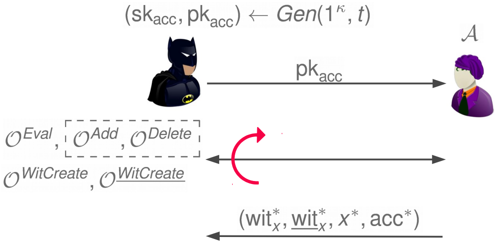
Experiment $\textbf{Exp}_{\kappa}^{ud}(\cdot)$:



Security - Undeniability

Defined for universal accumulators

Experiment $\mathbf{Exp}_{\kappa}^{ud}(\cdot)$:



- $\mathcal{A}$ wins if verification succeeds for both wit_x^* and $\underline{\mathsf{wit}}_x^*$

Undeniability $\overset{\not\Leftarrow}{\Rightarrow}$ Collision Freeness

We show that

- Efficient $\mathcal{A}^{cf}$ can be turned into efficient $\mathcal{A}^{ud}$

Undeniability $\overset{\not\Leftarrow}{\Rightarrow}$ Collision Freeness

We show that

- Efficient $\mathcal{A}^{cf}$ can be turned into efficient $\mathcal{A}^{ud}$

Other direction does not hold [BLL02]

Security - Indistinguishability I

So far, no meaningful formalization

- Existing formalization allows to prove indistinguishability
- for trivially distinguishable accumulators

Security - Indistinguishability I

So far, no meaningful formalization

- Existing formalization allows to prove indistinguishability
- for trivially distinguishable accumulators

We provide formalization

- not suffering from shortcomings above

Security - Indistinguishability II

Experiment $\mathbf{Exp}_{\kappa}^{ind}(\cdot)$:

$(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}) \leftarrow \mathit{Gen}(1^{\kappa}, t)$,
$b \xleftarrow{R} \{0, 1\}$

$\mathcal{A}$

$\xrightarrow{\mathsf{pk}_{\mathsf{acc}}}$

$\xleftarrow{\mathcal{X}_0, \mathcal{X}_1}$

$(\mathsf{acc}_{\mathcal{X}_b}, \mathsf{aux}) \leftarrow \mathit{Eval}((\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}), \mathcal{X}_b)$

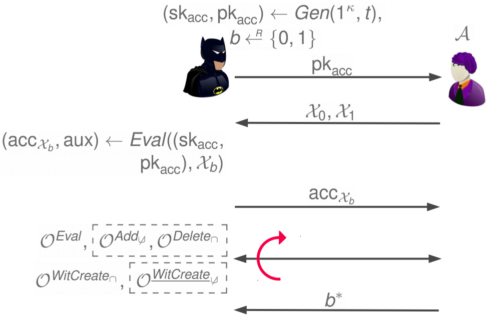
$\xrightarrow{\mathsf{acc}_{\mathcal{X}_b}}$

$\mathcal{O}^{\mathit{Eval}}, \boxed{\mathcal{O}^{\mathit{Add}_{\not\subseteq}}, \mathcal{O}^{\mathit{Delete}_{\cap}}}$
$\mathcal{O}^{\mathit{WitCreate}_{\cap}}, \boxed{\mathcal{O}^{\underline{\mathit{WitCreate}}_{\not\subseteq}}}$ $\longleftrightarrow$

$\xleftarrow{b^*}$

Security - Indistinguishability II

Experiment $\textbf{Exp}^{ind}_{\kappa}(\cdot)$:



- $\mathcal{A}$ wins if guess correct

Security - Indistinguishability III

Ad-hoc solution in literature

- Insert a (secret) random value z into acc.

Security - Indistinguishability III

Ad-hoc solution in literature

- Insert a (secret) random value z into acc.

However, weakens collision freeness

- Witness for z efficiently computable by definition

Security - Indistinguishability III

Ad-hoc solution in literature

- Insert a (secret) random value z into acc.

However, weakens collision freeness

- Witness for z efficiently computable by definition

Thus, we distinguish

- Indistinguishability
- Collision freeness weakening (cfw)-indistinguishability

Security - Indistinguishability III

Ad-hoc solution in literature

- Insert a (secret) random value z into acc.

However, weakens collision freeness

- Witness for z efficiently computable by definition

Thus, we distinguish

- Indistinguishability
- Collision freeness weakening (cfw)-indistinguishability

We modify [Ngu05] to provide indistinguishability

- First **indistinguishable** ***t*-bounded dynamic** accumulator

Outline

1. Introduction

2. A Unified Formal Model

3. Accumulators from Zero-Knowledge Sets

4. Black-Box Construction of Commitments

Zero-Knowledge Sets

Commit to a set $\mathcal{X}$

- Prove predicates of the form
  - $x \in \mathcal{X}$
  - $x \notin \mathcal{X}$
  - While not revealing anything else about $\mathcal{X}$

Zero-Knowledge Sets

Commit to a set $\mathcal{X}$

- Prove predicates of the form
  - $x \in \mathcal{X}$
  - $x \notin \mathcal{X}$
  - While not revealing anything else about $\mathcal{X}$

Observation

- Similar to undeniable indistinguishable accumulators

Zero-Knowledge Sets

Commit to a set $\mathcal{X}$

- Prove predicates of the form
  - $x \in \mathcal{X}$
  - $x \notin \mathcal{X}$
  - While not revealing anything else about $\mathcal{X}$

Observation

- Similar to undeniable indistinguishable accumulators
- Algorithms compatible
- Security notions similar

Accumulators from Zero-Knowledge Sets

Security notions

- Perfect completeness $\equiv$ correctness
- Soundness $\equiv$ undeniability

Accumulators from Zero-Knowledge Sets

Security notions

- Perfect completeness $\equiv$ correctness
- Soundness $\equiv$ undeniability
- Zero-knowledge
  - Simulation-based notion
  - $\exists$ simulator $\mathcal{S}$, negl. ϵ, s.t. $\forall$ PPT distinguishers:
    $\Pr[\text{distinguish sim/real}] \leq \epsilon(\kappa)$

Accumulators from Zero-Knowledge Sets

Security notions

- Perfect completeness $\equiv$ correctness
- Soundness $\equiv$ undeniability
- Zero-knowledge
  - Simulation-based notion
  - $\exists$ simulator $\mathcal{S}$, negl. ϵ, s.t. $\forall$ PPT distinguishers:
    Pr [distinguish sim/real] $\leq \epsilon(\kappa)$
  - We show that "zero-knowledge $\implies$ indistinguishability"
  - Other direction unclear, sim-based notion seems stronger

Accumulators from Zero-Knowledge Sets

Security notions

- Perfect completeness $\equiv$ correctness
- Soundness $\equiv$ undeniability
- Zero-knowledge
  - Simulation-based notion
  - $\exists$ simulator $\mathcal{S}$, negl. ϵ, s.t. $\forall$ PPT distinguishers: Pr [distinguish sim/real] $\leq \epsilon(\kappa)$
  - We show that "zero-knowledge $\implies$ indistinguishability"
  - Other direction unclear, sim-based notion seems stronger

First undeniable, unbounded, indistinguishable acc

- Nearly ZK sets $\rightarrow$ t-bounded

Outline

Commitments

- Compute commitment $\mathcal{C}$ to message m
- Later: provide opening $\mathcal{O}$ demonstrating that
  - $\mathcal{C}$ is commitment to m

Commitments

- Compute commitment $\mathcal{C}$ to message m
- Later: provide opening $\mathcal{O}$ demonstrating that
  - $\mathcal{C}$ is commitment to m
- Security (informal):
  - Correctness: straight forward

Commitments

- Compute commitment $\mathcal{C}$ to message m
- Later: provide opening $\mathcal{O}$ demonstrating that
  - $\mathcal{C}$ is commitment to m
- Security (informal):
  - Correctness: straight forward
  - Binding: Intractable to find $\mathcal{C}$, $\mathcal{O}$, $\mathcal{O}'$ such that $\mathcal{C}$ opens to two different messages $m \neq m'$

Commitments

- Compute commitment $\mathcal{C}$ to message m
- Later: provide opening $\mathcal{O}$ demonstrating that
  - $\mathcal{C}$ is commitment to m
- Security (informal):
  - Correctness: straight forward
  - Binding: Intractable to find $\mathcal{C}$, $\mathcal{O}$, $\mathcal{O}'$ such that $\mathcal{C}$ opens to two different messages $m \neq m'$
  - Hiding: For $\mathcal{C}$ to either m_0 or m_1. Intractable to decide whether $\mathcal{C}$ opens to m_0 or m_1

Commitments from Accumulators I

Use 1-bounded indistinguishable accumulators

- $\mathcal{C} \leftarrow \mathsf{acc}_{\{m\}}$
- $\mathcal{O} \leftarrow (m, r, \mathsf{wit}_m, aux)$ such that
  - $\mathsf{acc}_{\{m\}} = \mathit{Eval}_r((\emptyset, \mathsf{pk}_{\mathsf{acc}}), \{m\})$
  - $\mathit{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\{m\}}, \mathsf{wit}_m, m) = \mathtt{true}$

Commitments from Accumulators I

Use 1-bounded indistinguishable accumulators

- $\mathcal{C} \leftarrow \mathsf{acc}_{\{m\}}$
- $\mathcal{O} \leftarrow (m, r, \mathsf{wit}_m, aux)$ such that
  - $\mathsf{acc}_{\{m\}} = \mathit{Eval}_r((\emptyset, \mathsf{pk}_{\mathsf{acc}}), \{m\})$
  - $\mathit{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\{m\}}, \mathsf{wit}_m, m) = \mathtt{true}$
- Collision-freeness $\Rightarrow$ Binding

Commitments from Accumulators I

Use 1-bounded indistinguishable accumulators

- $\mathcal{C} \leftarrow \mathsf{acc}_{\{m\}}$
- $\mathcal{O} \leftarrow (m, r, \mathsf{wit}_m, aux)$ such that
  - $\mathsf{acc}_{\{m\}} = \mathit{Eval}_r((\emptyset, \mathsf{pk}_{\mathsf{acc}}), \{m\})$
  - $\mathit{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\{m\}}, \mathsf{wit}_m, m) = \mathtt{true}$
- Collision-freeness $\Rightarrow$ Binding
- Indistinguishability $\Rightarrow$ Hiding

Commitments from Accumulators I

Use 1-bounded indistinguishable accumulators

- $\mathcal{C} \leftarrow \mathsf{acc}_{\{m\}}$
- $\mathcal{O} \leftarrow (m, r, \mathsf{wit}_m, aux)$ such that
  - $\mathsf{acc}_{\{m\}} = \mathit{Eval}_r((\emptyset, \mathsf{pk}_{\mathsf{acc}}), \{m\})$
  - $\mathit{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\{m\}}, \mathsf{wit}_m, m) = \mathtt{true}$
- Collision-freeness $\Rightarrow$ Binding
- Indistinguishability $\Rightarrow$ Hiding

Observe: cfw-indistinguishability not useful

Commitments from Accumulators II

Straight forward extension to set-commitments

- Use t-bounded accumulators
- Opening w.r.t. entire set

Commitments from Accumulators II

Straight forward extension to set-commitments

- Use t-bounded accumulators
- Opening w.r.t. entire set

Trapdoor commitments

- Use sk_{acc} as trapdoor

Conclusion

Unified model for accumulators

- Covering all features existing to date

Conclusion

Unified model for accumulators

- Covering all features existing to date

Introduce indistinguishability notion

- Provide first indistinguishable dynamic scheme

Conclusion

Unified model for accumulators

- Covering all features existing to date

Introduce indistinguishability notion

- Provide first indistinguishable dynamic scheme

Show relations to other primitives

- Commitments
- Zero-knowledge sets
  - Yields first undeniable, unbounded, indistinguishable, universal accumulator
- Inspiration for new constructions

Thank you.

david.derler@iaik.tugraz.at

Extended version: `http://eprint.iacr.org/2015/087`

References I

[BLL02] Ahto Buldas, Peeter Laud, and Helger Lipmaa. Eliminating counterevidence with applications to accountable certificate management. *Journal of Computer Security*, 10(3):273–296, 2002.

[Ngu05] Lan Nguyen. Accumulators from bilinear pairings and applications. In *Topics in Cryptology - CT-RSA 2005, The Cryptographers' Track at the RSA Conference 2005, San Francisco, CA, USA, February 14-18, 2005, Proceedings*, pages 275–292, 2005.

Non-Interactive Zero-Knowledge Proofs of Non-Membership

O. Blazy, C. Chevalier, D. Vergnaud

XLim / Université Paris II / ENS

Proof of Knowledge

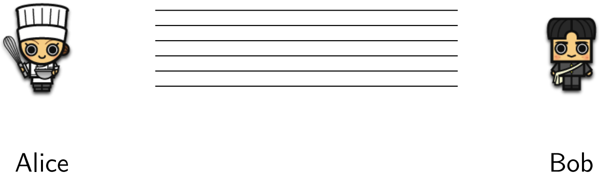
Alice

Bob

- Interactive method for one party to prove to another the knowledge of a secret $\mathcal{S}$.

Classical Instantiations : Schnorr proofs, Sigma Protocols . . .

Proving that a statement is not satisfied



- Interactive method for one party to prove to another the knowledge of a secret $\mathcal{S}$ that does not belong to a language $\mathcal{L}$.

Applications

- Credentials
- Enhanced Authenticated Key Exchange

Additional properties

- Non-Interactive
- Zero-Knowledge
- Implicit

Applications

- Credentials
- Enhanced Authenticated Key Exchange

Additional properties

- Non-Interactive
- Zero-Knowledge
- Implicit

Applications

- Credentials
- Enhanced Authenticated Key Exchange

Additional properties

- Non-Interactive
- Zero-Knowledge
- Implicit

Applications

- Credentials
- Enhanced Authenticated Key Exchange

Additional properties

- Non-Interactive
- Zero-Knowledge
- Implicit

Applications

- Credentials
- Enhanced Authenticated Key Exchange

Additional properties

- Non-Interactive
- Zero-Knowledge
- Implicit

A
B
C

Zero-Knowledge Proof Systems

- Introduced in 1985 by Goldwasser, Micali and Rackoff.

  $\rightsquigarrow$ Reveal nothing other than the validity of assertion being proven

- Used in many cryptographic protocols
  - Anonymous credentials
  - Anonymous signatures
  - Online voting
  - ...

Zero-Knowledge Proof Systems

- Introduced in 1985 by Goldwasser, Micali and Rackoff.

  $\rightsquigarrow$ Reveal nothing other than the validity of assertion being proven

- Used in many cryptographic protocols
  - Anonymous credentials
  - Anonymous signatures
  - Online voting
  - ...

Zero-Knowledge Proof Systems

- Introduced in 1985 by Goldwasser, Micali and Rackoff.

  $\leadsto$ Reveal nothing other than the validity of assertion being proven

- Used in many cryptographic protocols
  - **Anonymous credentials**
  - **Anonymous signatures**
  - **Online voting**
  - ...

Zero-Knowledge Interactive Proof

Alice   Bob

- interactive method for one party to prove to another that a statement $\mathcal{S}$ is true, without revealing anything other than the veracity of $\mathcal{S}$.

1. **Completeness:** if $\mathcal{S}$ is true, the honest verifier will be convinced of this fact
2. **Soundness:** if $\mathcal{S}$ is false, no cheating prover can convince the honest verifier that it is true
3. **Zero-knowledge:** if $\mathcal{S}$ is true, no cheating verifier learns anything other than this fact.

Zero-Knowledge Interactive Proof

Alice

Bob

- interactive method for one party to prove to another that a statement $\mathcal{S}$ is true, without revealing anything other than the veracity of $\mathcal{S}$.

1. **Completeness:** if $\mathcal{S}$ is true, the honest verifier will be convinced of this fact
2. **Soundness:** if $\mathcal{S}$ is false, no cheating prover can convince the honest verifier that it is true
3. **Zero-knowledge:** if $\mathcal{S}$ is true, no cheating verifier learns anything other than this fact.

Non-Interactive Zero-Knowledge Proof

Alice

Bob

- non-interactive method for one party to prove to another that a statement $\mathcal{S}$ is true, without revealing anything other than the veracity of $\mathcal{S}$.

1. **Completeness:** $\mathcal{S}$ is true $\leadsto$ verifier will be convinced of this fact
2. **Soundness:** $\mathcal{S}$ is false $\leadsto$ no cheating prover can convince the verifier that $\mathcal{S}$ is true
3. **Zero-knowledge:** $\mathcal{S}$ is true $\leadsto$ no cheating verifier learns anything other than this fact.

Certification of Public Keys: SPHF [ACP09]

A user can ask for the certification of pk, but if he knows the associated sk only:

With a Smooth Projective Hash Function

$\mathcal{L}$: pk and $C = \mathcal{C}(\mathsf{sk}; r)$ are associated to the same sk

- U sends his pk, and an encryption C of sk;
- A generates the certificate Cert for pk, and sends it,
  masked by $\mathsf{Hash} = \mathsf{Hash}(\mathsf{hk}; (\mathsf{pk}, C))$;
- U computes $\mathsf{Hash} = \mathsf{ProjHash}(\mathsf{hp}; (\mathsf{pk}, C), r))$, and gets Cert.

Certification of Public Keys: SPHF [ACP09]

A user can ask for the certification of pk, but if he knows the associated sk only:

With a Smooth Projective Hash Function

$\mathcal{L}$: pk and $C = \mathcal{C}(\mathsf{sk}; r)$ are associated to the same sk

- U sends his pk, and an encryption C of sk;
- A generates the certificate Cert for pk, and sends it, masked by Hash $=$ Hash(hk; (pk, C));
- U computes Hash $=$ ProjHash(hp; (pk, C), r)), and gets Cert.

Implicit proof of knowledge of sk

Smooth Projective Hash Functions [CS02]

Definition [CS02,GL03]

Let $\{H\}$ be a family of functions:

- X, domain of these functions
- L, subset (a language) of this domain

such that, for any point x in L, $H(x)$ can be computed by using

- either a *secret* hashing key hk: $H(x) = \mathsf{Hash}_L(\mathsf{hk}; x)$;
- or a *public* projected key hp: $H'(x) = \mathsf{ProjHash}_L(\mathsf{hp}; x, w)$

Public mapping $\mathsf{hk} \mapsto \mathsf{hp} = \mathsf{ProjKG}_L(\mathsf{hk}, x)$

SPHF Properties

For any $x \in X$, $H(x) = \mathsf{Hash}_L(\mathsf{hk}; x)$
For any $x \in L$, $H(x) = \mathsf{ProjHash}_L(\mathsf{hp}; x, w)$

w witness that $x \in L$, $\mathsf{hp} = \mathsf{ProjKG}_L(\mathsf{hk}, x)$

Smoothness

For any $x \notin L$, $H(x)$ and hp are independent

Pseudo-Randomness

For any $x \in L$, $H(x)$ is pseudo-random, without a witness w

The latter property requires L to be a hard-partitioned subset of X.

SPHF Properties

For any $x \in X$, $H(x) = \mathsf{Hash}_L(\mathsf{hk}; x)$
For any $x \in L$, $H(x) = \mathsf{ProjHash}_L(\mathsf{hp}; x, w)$
w witness that $x \in L, \mathsf{hp} = \mathsf{ProjKG}_L(\mathsf{hk}, x)$

Smoothness

For any $x \notin L$, $H(x)$ and hp are independent

Pseudo-Randomness

For any $x \in L$, $H(x)$ is pseudo-random, without a witness w

The latter property requires L to be a hard-partitioned subset of X.

SPHF Properties

For any $x \in X$, $H(x) = \mathsf{Hash}_L(\mathsf{hk}; x)$
For any $x \in L$, $H(x) = \mathsf{ProjHash}_L(\mathsf{hp}; x, w)$
w witness that $x \in L, \mathsf{hp} = \mathsf{ProjKG}_L(\mathsf{hk}, x)$

Smoothness

For any $x \notin L$, $H(x)$ and hp are independent

Pseudo-Randomness

For any $x \in L$, $H(x)$ is pseudo-random, without a witness w

The latter property requires L to be a hard-partitioned subset of X.

SPHF Properties

For any $x \in X$, $H(x) = \mathsf{Hash}_L(\mathsf{hk}; x)$
For any $x \in L$, $H(x) = \mathsf{ProjHash}_L(\mathsf{hp}; x, w)$
w witness that $x \in L, \mathsf{hp} = \mathsf{ProjKG}_L(\mathsf{hk}, x)$

Smoothness

For any $x \notin L$, $H(x)$ and hp are independent

Pseudo-Randomness

For any $x \in L$, $H(x)$ is pseudo-random, without a witness w

The latter property requires L to be a hard-partitioned subset of X.

Global Idea

- π: Proof that $W \in \mathcal{L}$
- π: Randomizable, Indistinguishability of Proof
- π': Proof that π was computed honestly

Global Idea

- π: Proof that $W \in \mathcal{L}$
- π: Randomizable, Indistinguishability of Proof
- π': Proof that π was computed honestly

Global Idea

- π: Proof that $W \in \mathcal{L}$
- π: Randomizable, Indistinguishability of Proof
- π': Proof that π was computed honestly

Global Idea

- π: Proof that $W \in \mathcal{L}$
- π: Randomizable, Indistinguishability of Proof
- π': Proof that π was computed honestly

To prove that $W \notin \mathcal{L}$

- Try to prove that $W \in \mathcal{L}$ which will output a π
- π will not be valid
- Compute π' stating that π was computed honestly

Global Idea

- π: Proof that $W \in \mathcal{L}$
- π: Randomizable, Indistinguishability of Proof
- π': Proof that π was computed honestly

To prove that $W \notin \mathcal{L}$

- Try to prove that $W \in \mathcal{L}$ which will output a π
- π will not be valid
- Compute π' stating that π was computed honestly

Global Idea

- π: Proof that $W \in \mathcal{L}$
- π: Randomizable, Indistinguishability of Proof
- π': Proof that π was computed honestly

To prove that $W \notin \mathcal{L}$

- Try to prove that $W \in \mathcal{L}$ which will output a π
- π will not be valid
- Compute π' stating that π was computed honestly

From a very high level

- If an adversary forges a proof, this means that both π and π' are valid
- Either π was not computed honestly, and under the Soundness of π' this should not happen
- Or π was computed honestly but lead to an invalid proof, and under the Completeness of π this should not happen

From a very high level

- If an adversary forges a proof, this means that both π and π' are valid
- Either π was not computed honestly, and under the Soundness of π' this should not happen
- Or π was computed honestly but lead to an invalid proof, and under the Completeness of π this should not happen

From a very high level

- If an adversary forges a proof, this means that both π and π' are valid
- Either π was not computed honestly, and under the Soundness of π' this should not happen
- Or π was computed honestly but lead to an invalid proof, and under the Completeness of π this should not happen

Possible Instantiations

Proof π	Proof π'	Interactive	Properties
Groth Sahai	Groth Sahai	No	Zero-Knowledge
SPHF	SPHF	Yes	Implicit
Groth Sahai	SPHF	Depends	ZK, Implicit

Anonymous Credentials

Allows user to authenticate while protecting their privacy.

- Recent work, build non-interactive credentials for NAND
- By combining with ours, it leads to *efficient* Non-Interactive Credentials
- No accumulators are needed

Anonymous Credentials

Allows user to authenticate while protecting their privacy.

- Recent work, build non-interactive credentials for NAND
- By combining with ours, it leads to *efficient* Non-Interactive Credentials
- No accumulators are needed

Anonymous Credentials

Allows user to authenticate while protecting their privacy.

- Recent work, build non-interactive credentials for NAND
- By combining with ours, it leads to *efficient* Non-Interactive Credentials
- No accumulators are needed

Language Authenticated Key Exchange

Alice

$\rightarrow \mathcal{C}(M_B)$

$\mathcal{C}(M_A), \mathsf{hp}_B \leftarrow$

$\rightarrow \mathsf{hp}_A$

Bob

$H_B \cdot H'_A$

$H'_B \cdot H_A$

Same value iff languages are as expected, and users know witnesses.

Summing up

- Proposed a generic framework to prove negative statement *
- Gives several instantiation of this framework, allowing some modularity
- Works outside pairing environment

Open Problems

- Be compatible with post-quantum cryptography
- Weaken the requirements, on the building blocks

Summing up

- Proposed a generic framework to prove negative statement *
- Gives several instantiation of this framework, allowing some modularity
- Works outside pairing environment

Open Problems

- Be compatible with post-quantum cryptography
- Weaken the requirements, on the building blocks