Efficient Constructions of Bilinear Accumulators

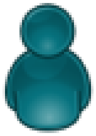
Ioanna Karantaidou, Foteini Baldimtsi

GEORGE MASON UNIVERSITY

Set Membership

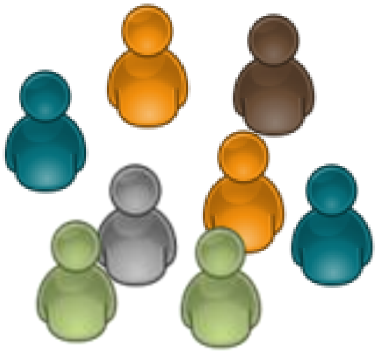
Bank, GMU, subscription-based service, etc

Alice



I am Alice

List of members
…
Alice
…
…



List of members as a Data structure

- **Size** of List: O(n)
- (at least one of) **Additions/Deletions,** lookups depends on n
- **Privacy** against list holder/membership verification in a privacy preserving way: Expensive!

Accumulator Setting

MANAGER

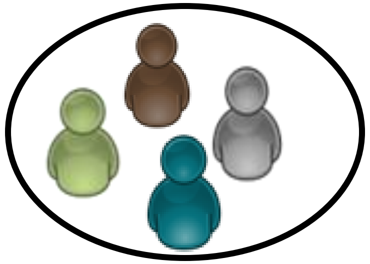
Initialize & Create Acc.v

Acc.v

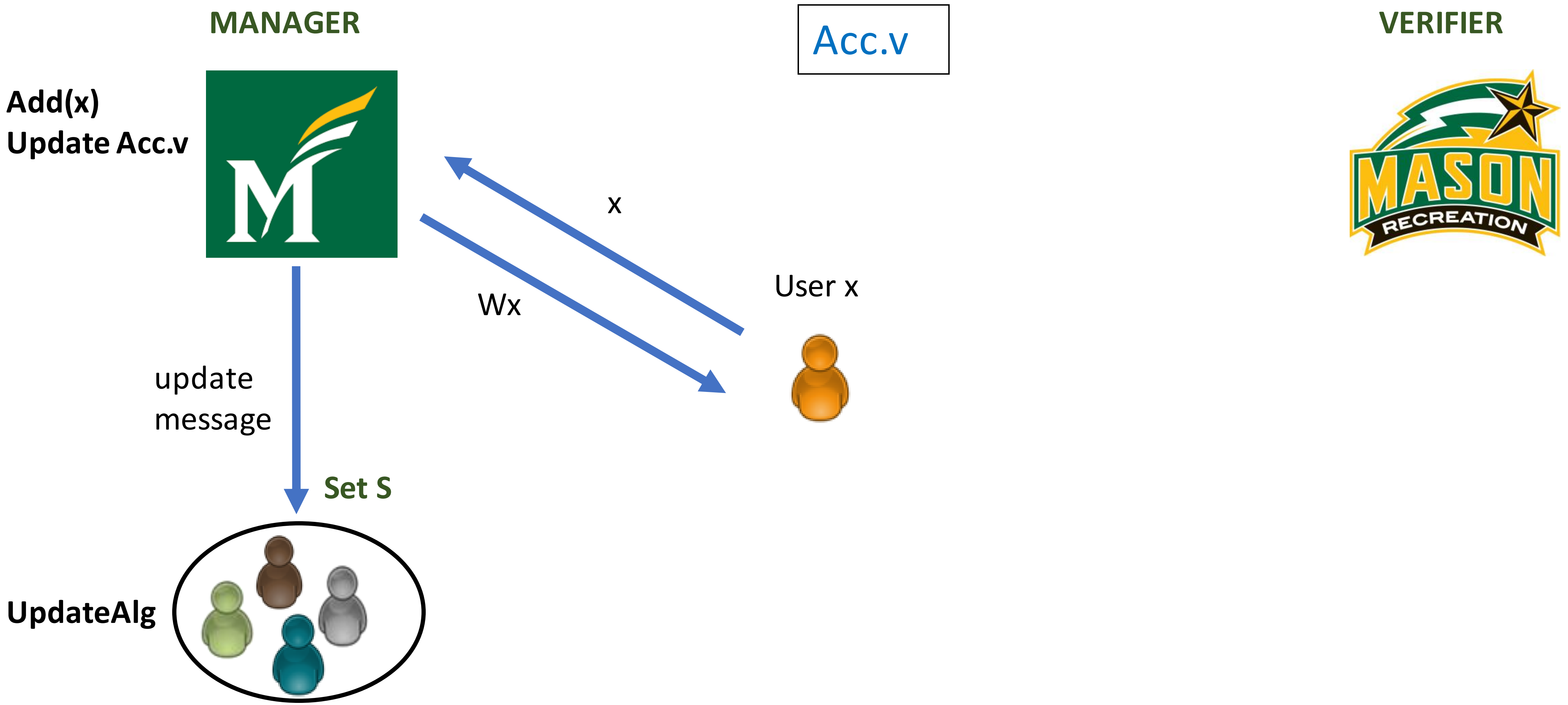
Accumulator Value: holds Set S

VERIFIER

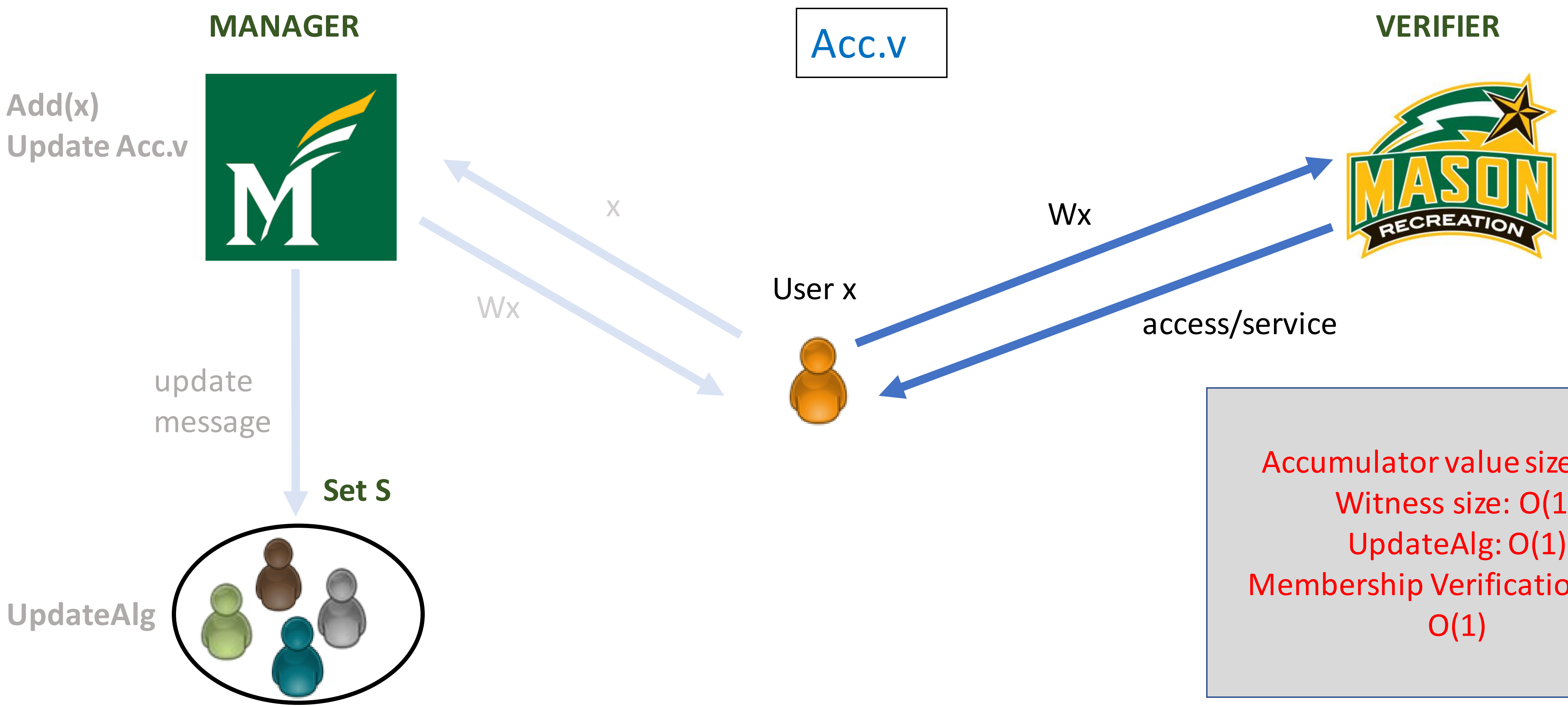
Set S



Positive Accumulator: adding User x



Positive Accumulator: proving membership



Accumulator value size: O(1)
Witness size: O(1)
UpdateAlg: O(1)
Membership Verification time: O(1)

Security Properties (membership)

…
Alice
…

Set/List	Accumulator acc
Verification=lookup	*Verification algorithm:VerMem(w_x)*

Alice is a member → verification ✓

$x \in acc$ → VerMem(w_x)=1

correctness

Charlie
Alice

Bob is **not** a member → verification ✗

$x \notin acc$ → VerMem(w_x)=0 (or =1 with negligible prob.)

soundness

2 Types of Accumulators

RSA based accumulators [CL02, LLX07, BdM93]

- Accumulate odd prime numbers
- Factorization of group hidden
- Strong RSA assumption

Bilinear Pairing based accumulators [N05, CKS09, ATSM09, ZKP17]

- Accumulate integers
- Known order groups
- Witness, accumulator value belong in pairing friendly groups
- q-SDH assumption

Choice depends on the application!

Common Issues with Known Accumulators

- Unnecessary accumulator updates that cause high communication costs
- Expensive non-membership operations
- Computational overhead due to extra properties

Can we do better if we take advantage of the presence of a trusted entity (manager)?

Discussion on the secret key model

- Most known constructions have a trusted setup
- Anonymous Credentials, subscription-based services, etc

Our Results

1. Positive Bilinear Accumulator with Optimal Communication Cost
2. Universal Bilinear Accumulator with Constant Non-Membership Witness Creation
3. ZK Accumulator with Constant Non-Membership Witness Creation and Update

FIRST CONSTRUCTION

Positive Bilinear Accumulator with
Optimal Communication Cost

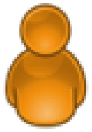
Positive Bilinear Accumulator

$$Acc.v = g^{(x_1+sk)\ldots(x_n+sk)}$$

(sk)

Add(x)

User x

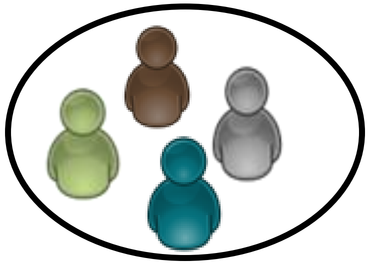


$$Acc.v = g^{(x_1+sk)\ldots(x_n+sk)(\boldsymbol{x}+sk)}$$

upmsg

$$w_x = g^{(x_1+sk)\ldots(x_n+sk)}$$

Set S



Positive Bilinear Accumulator Verification

$w_x = Acc.v^{(x+sk)^{-1}}$

$w_x{}^{(x+sk)} = Acc.v$

Public parameters: $g, g^{sk}, (g^{sk})^2, (g^{sk})^3, \ldots \rightarrow$

$w_x{}^{(x+sk)}$ ✕

$Acc.v = g^{(x_1+sk)\ldots(x_n+sk)(x+sk)}$

$w_x = g^{(x_1+sk)\ldots(x_n+sk)}$

Public parameters:

$g, g^{sk}, (g^{sk})^2, (g^{sk})^3, \ldots \rightarrow$

g^x, g^{sk}

e(w_x, g^x g^{sk})=e($Acc.v$, g) (VerMem)

Positive Bilinear Accumulator

$$Acc.v = g^{(x_1+sk)\ldots(x_n+sk)(\boldsymbol{x}+sk)}$$

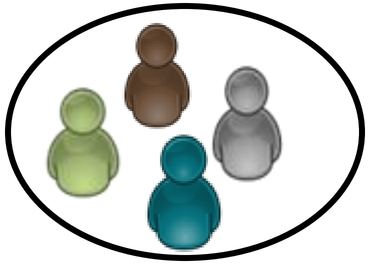
(sk)

Del(x)

User x

upmsg

Set S



$$Acc.v = g^{(x_1+sk)\ldots(x_n+sk)}$$

Positive Bilinear Accumulator

$$\boxed{Acc.v = g^{(x_1+sk)\dots(x_n+sk)}}$$

(sk)

Add(x)

User x

upmsg

$$\boxed{Acc.v = g^{(x_1+sk)\dots(x_n+sk)(x+sk)}}$$

$$w_x = g^{(x_1+sk)\dots(x_n+sk)}$$

Set S

Minimum communication bound (on update messages) for positive accumulators= |d| (number of deletions)

Camacho, Philippe, and Alejandro Hevia. "On the impossibility of batch update for cryptographic accumulators." *International Conference on Cryptology and Information Security in Latin America*. Springer, Berlin, Heidelberg, 2010.

Positive Bilinear Accumulator with Optimal Communication Cost-First try

(sk)

User x

$$Acc.v = g^{u}$$

Add(x)

$$w_x = g^{u\,(x+sk)^{-1}}$$

Del(x)

upmsg

$$Acc.v = g^{u\,(x+sk)^{-1}}$$

Positive Bilinear Accumulator with Optimal Communication Cost-First try

User x

(sk)

- Communication efficient
- Dynamic (add,del)
- Positive (membership)

- Correctness

$w_x^{(x+sk)} = Acc.v$

holds and **VerMem** same

- Soundness??

Acc

Positive Bilinear Accumulator with Optimal Communication Cost-First try

Proof overview:

- R (public parameters) runs an adversary A (public parameters)
- A submits lists of to-be-added, to-be-deleted elements L_A, L_D
- R simulates updates and witnesses
- A breaks acc soundness
- R breaks q-SDH assumption

q-SDH: Given (p, G, G_T, e, g), $\{g^{sk}\}^i, i = 0, \dots, q$ there is negligible probability of finding
$g^{\frac{1}{sk+x}}$ for $x \in \mathbb{Z}_p$

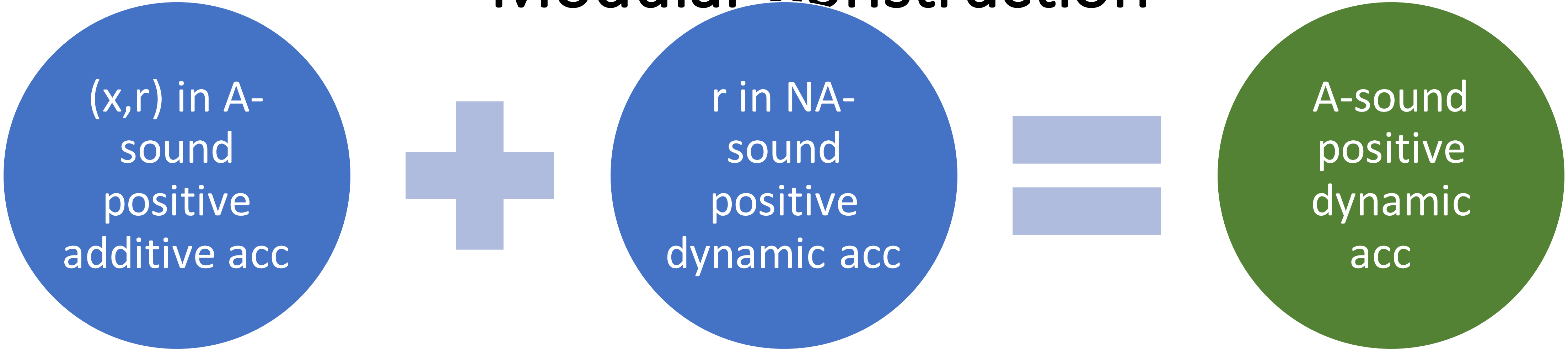
Positive Bilinear Accumulator with Optimal Communication Cost-First try

Proof overview:

- R (public parameters) runs an adversary A (public parameters)
- A submits lists of to-be-added, to-be-deleted elements L_A, L_D
- R simulates updates and witnesses
- A breaks acc soundness
- R breaks q-SDH assumption

Adaptive soundness not achieved

Positive Bilinear Accumulator with Optimal Communication Cost-Modular Construction



Baldimtsi, Foteini, et al. "Accumulators with applications to anonymity-preserving revocation." *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2017.

Positive Bilinear Accumulator with Optimal Communication Cost-Modular Construction

(x,r) in A-sound positive additive acc

+

r in NA-sound positive dynamic acc

=

A-sound positive dynamic acc

No updates for positive accumulator that supports additions only

- r=F(x), where F() is a pseudorandom function
- Updates for deletions

Communication cost= |d|
Optimal!

	Positive			
	Camenisch et al 09	Nguyen 05	this work (NA-sound)	this work (A-sound)
Add	1	1	1	1
Del	1	1	1	1
MemWitCreate	1	1	1	1
NonMemWitCreate	-	-	-	-
MemWitUpOnAdd	1	1	0	0
MemWitUpOnDel	1	1	1	1
NonMemWitUpOnAdd	-	-	-	-
NonMemWitUpOnDel	-	-	-	-
VerMem	1	1	1	1
VerNonMem	-	-	-	-
Manager storage	1	1	1	1
Parameters	2q	q	q	q
Com. cost	|a|+|d|	|a|+|d|	|d|	|d|
Efficient ZKPs	✓	✓	✓	✓
Adaptively-sound	✓	✓		✓

- Jan Camenisch, Markulf Kohlweiss, and Claudio Soriente. An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In PKC 2009
- Lan Nguyen. Accumulators from bilinear pairings and applications. In CT-RSA 2005.

SECOND CONSTRUCTION

Universal Bilinear Accumulator with
Constant Non-Membership Witness Creation

Additional Properties (non-membership:NM)

Charlie
Alice

Set/List

NM verification=lookup

Accumulator acc

NM verification algorithm:

VerNonMem$(\overline{w_x})$

Bob is **not** a member

→ NM verification

$x \notin acc$ → VerNonMem$(\overline{w_x})$=1

correctness

…
Alice
…

Alice is a member

→ NM verification

$x \in acc$ → VerNonMem$(\overline{w_x})$=0

(or =1 with negligible prob.)

soundness

Generic Universal Modular Construction motivation: Non membership for y

Bilinear **ATSM09**, S={x_i}, $x_i \in \mathbb{Z}_p$

$$\prod_{i=1}^{|S|}(x_i+sk) = c(sk)(y+sk)+d$$

Users (public parameters):

S={x_i}, polynomial division

Manager (sk):

$\prod_{i=1}^{|S|}(x_i+sk) \in \mathbb{Z}$, used as exponent

RSA **LLX07**, S={y_i}, y_i primes

a $(\prod_{i=1}^{|S|} y_i) + b\, y = 1$

Users (public parameters)/Manager (sk):

$\prod_{i=1}^{|S|} y_i \in \mathbb{Z}$, Euclidean algorithm

Generic Universal Modular Construction motivation: Non membership for y

Bilinear **ATSM09**, S={x_i}, $x_i \in \mathbb{Z}_p$

$$\prod_{i=1}^{|S|}(x_i+sk) = c(sk)(y$$

Users (public paran

S={x_i}, polynomial

Manager (sk):

$\prod_{i=1}^{|S|}(x_i+sk) \in \mathbb{Z}$, used as exponent

non-membership cost: |S|

RSA **LLX07**, S={y_i}, y_i primes

$\prod_{i=1}^{|S|} y_i) + b\, y = 1$

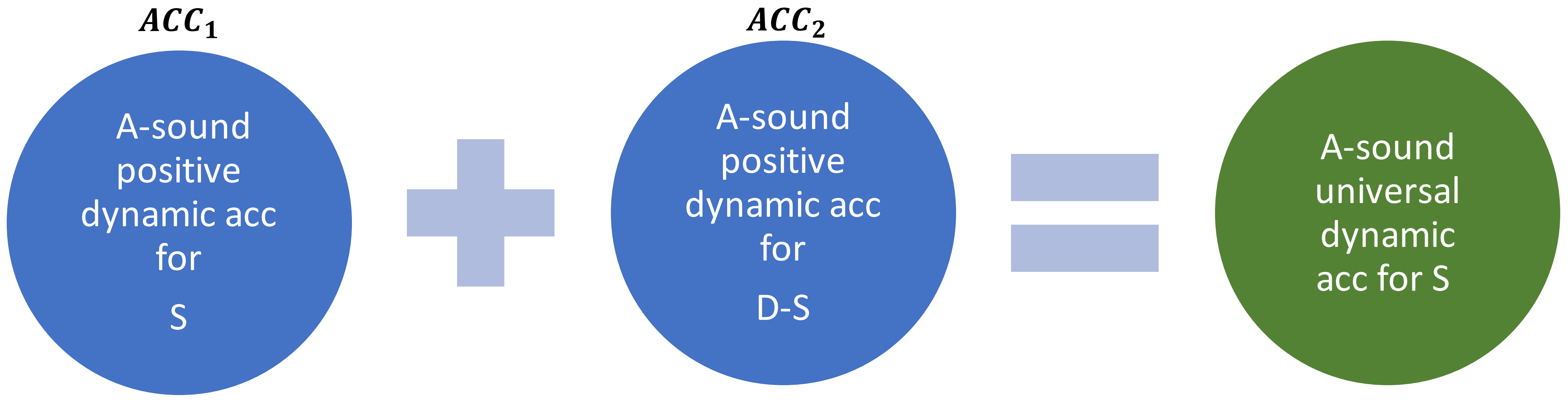
blic parameters)/Manager (sk):

$\prod_{i=1} y_i \in \mathbb{Z}$, Euclidean algorithm

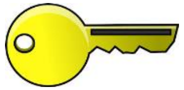
Generic Universal Modular Construction Overview

Can we replace non-membership with constant-runtime membership??
Yes, with a trusted manager

Can we make sure that ACC_1 and ACC_2 are disjoint?
The accumulator manager always signs the most up to date value of the accumulator



Generic Universal Modular Construction



(sk)

$\boldsymbol{ACC_1.\mathrm{Gen}(1^\lambda, \emptyset)}$
$\boldsymbol{ACC_2.\mathrm{Gen}(1^\lambda, \mathrm{D})}$

Generic Universal Modular Construction

(sk)

User x

$x \in S_2$

$\boldsymbol{ACC_1\,(S_1)}$

$\boldsymbol{ACC_2\,(S_2)}$

Add(x)

$\boldsymbol{ACC_1.\mathbf{add}(\mathbf{x})}$
$\boldsymbol{ACC_2.\mathbf{del}(\mathbf{x})}$

$\boldsymbol{ACC_1\,(S_1 \cup \{x\})}$

$\boldsymbol{ACC_2\,(S_2 - \{x\})}$

$w_x = ACC_1.w$

Generic Universal Modular Construction

(sk)

User x

$\boldsymbol{ACC_1\,(S_1)}$

$\boldsymbol{ACC_2\,(S_2)}$

$x \in S_1$

Del(x)

$\boldsymbol{ACC_1.\mathbf{del}(\mathbf{x})}$

$\boldsymbol{ACC_2.\mathbf{add}(\mathbf{x})}$

$\boldsymbol{ACC_1\,(S_1 - \{x\})}$

$\boldsymbol{ACC_2\,(S_2 \cup \{x\})}$

$\overline{w_x} = ACC_2.w$

$w\ (membership)$

$\overline{w}$ (non-membership)

Generic Universal Modular Construction

User x

VERIFIER

w_x

$ACC_1.\mathbf{VerMem}(\mathbf{w_x})$

User y

O(1)

$\overline{w_y}$

$ACC_2.\mathbf{VerMem}(\overline{w_y})$

Generic Universal Modular Construction

- **Gen**(1^λ)**:**
  1. $\mathsf{ACC}_1.Gen(1^\lambda, \emptyset)$
  2. $\mathsf{ACC}_2.Gen(1^\lambda, D)$ with the same random coins for parameters, where D is the accumulator's domain
- **Add**(x)**:**
  1. $\mathsf{ACC}_1.\mathrm{Add}(x)$
  2. $w_x = \mathsf{ACC}_1.\mathrm{MemWitCreate}(x)$
  3. $\mathsf{ACC}_2.\mathrm{Del}(x)$
- **Del**(x)**:**
  1. $\mathsf{ACC}_2.\mathrm{Add}(x)$
  2. $\overline{w_x}$=$\mathsf{ACC}_2.\mathrm{MemWitCreate}(x)$
  3. $\mathsf{ACC}_1.\mathrm{Del}(x)$
- **MemWitUpOnAdd/Del(y):**
  1. ACC_1.MemWitUpOnAdd/Del(y)
- **NonMemWitUpOnAdd/Del (y):**
  1. ACC_2.MemWitUpOnAdd/Del(y)
- **VerMem**(x)**:**
  1. $\mathsf{ACC}_1.\mathrm{VerMem}(x)$
- **VerNonMem**(x)**:**
  1. $\mathsf{ACC}_2.\mathrm{VerMem}(x)$

Fig. 3. Generic Construction for Universal Dynamic accumulator

<u>Note on Efficiency</u>

Concretes:

- Generation (run once) linear to Domain size
- Add/Del of double cost

Asymptotics:

All operations constant, independent of accumulated set S

Generic Universal Modular Construction-Soundness

Theorem: *A combination of accumulators* ACC_1, ACC_2 *is a universal dynamic adaptively-sound accumulator if* ACC_1, ACC_2 *are positive dynamic adaptively-sound accumulators of domain D and one is holding S⊂D and the other one* $\overline{S}$ *⊂D and public updates are not permitted.*

Generic Universal Modular Construction Soundness

Theorem: *A combination of accumulators ACC_1, ACC_2 is a universal dynamic adaptively-sound accumulator if ACC_1, ACC_2 are positive dynamic adaptively-sound accumulators of domain D and one is holding S⊂D and the other one $\overline{S}$ ⊂D and public updates are not permitted.*

INTUITION:
Information obtained by 2 accumulators with the same instantiation could be obtained by different states of 1 accumulator

Generic Universal Modular Construction Soundness

Theorem: *A combination of accumulators ACC_1, ACC_2 is a universal dynamic adaptively-sound accumulator if ACC_1, ACC_2 are positive dynamic adaptively-sound accumulators of domain D and one is holding $S \subset D$ and the other one $\overline{S} \subset D$ and public updates are not permitted*

PROOF:
R has access to Add/Del oracle.
A breaks ACC=(ACC_1, ACC_2)
soundness.
R breaks ACC_1 (positive) soundness

Efficiency Results

	Positive			Universal	
	Camenisch et al 09	Nguyen 05	this work (A-sound)	Au et al 09	This work- Instantiation with Nguyen 05
Add	1	1	1	1	1
Del	1	1	1	1	1
MemWitCreate	1	1	1	1	1
NonMemWitCreate	-	-	-	$\|S\|$	1
MemWitUpOnAdd	1	1	0	1	1
MemWitUpOnDel	1	1	1	1	1
NonMemWitUpOnAdd	-	-	-	1	1
NonMemWitUpOnDel	-	-	-	1	1
VerMem	1	1	1	1	1
VerNonMem	-	-	-	1	1
Manager storage	1	1	1	$\|S\|$	1
Parameters	2q	q	q	q	q?
Com. cost	$\|a\|+\|d\|$	$\|a\|+\|d\|$	$\|d\|$	$\|a\|+\|d\|$	$\|a\|+\|d\|$
Efficient ZKPs	✓	✓	✓	✓	✓
Adaptively-sound	✓	✓	✓	✓	✓

THIRD CONSTRUCTION

ZK Accumulator with
Constant Non-Membership Witness Creation and Update

Application: ZK Accumulator

S ??

Acc.v

MANAGER

x?

Member/non-member, witness

Adv

Esha Ghosh , Olga Ohrimenko , Dimitrios Papadopoulos , Roberto Tamassia and Nikos Triandopoulos "Zero-Knowledge Accumulators and Set Operations" *IACR Cryptology ePrint Archive* 2015 (2015): 404.

Application: ZK Accumulator

Acc.v

MANAGER

x?

Member/non-member, witness

Adv

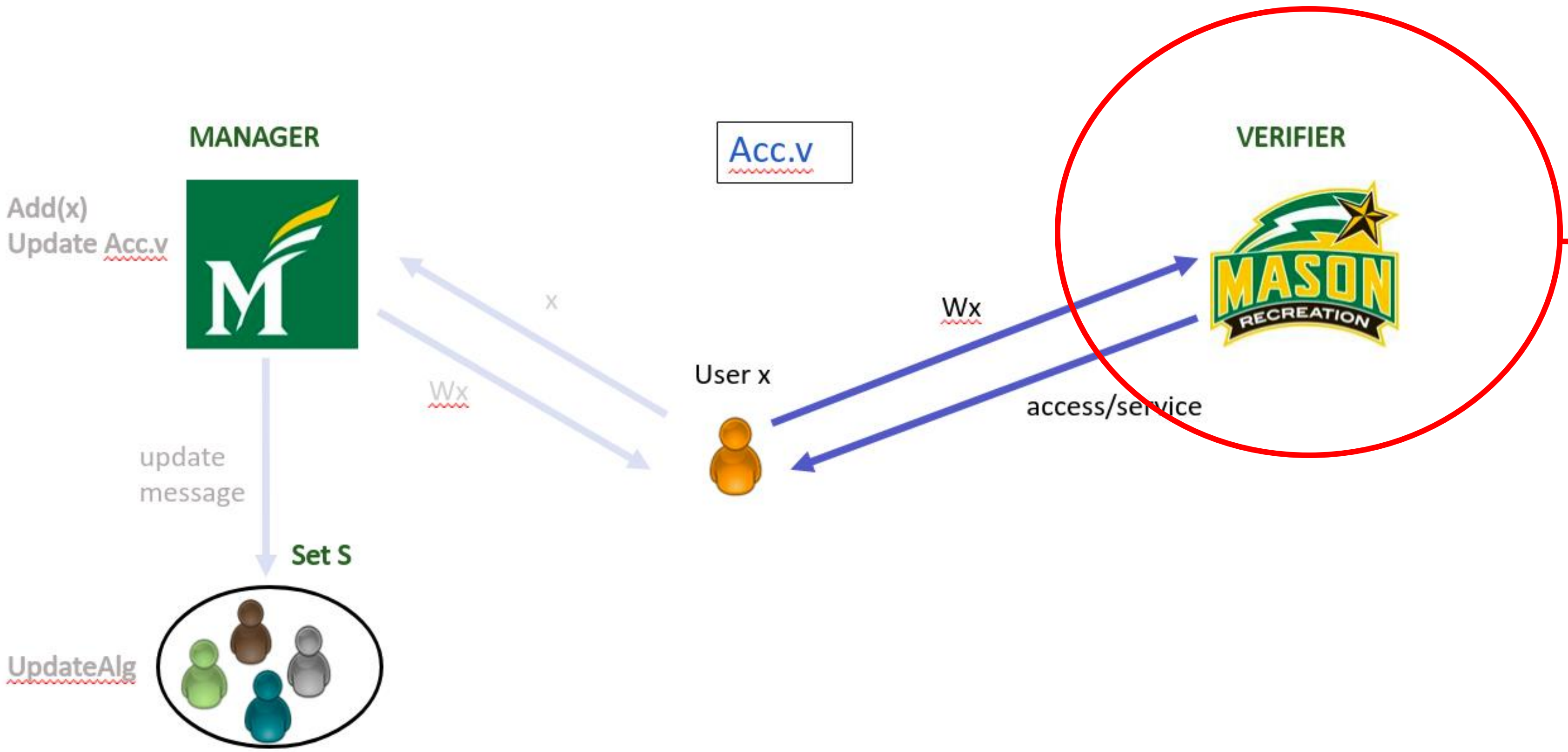
Goal:

- Adv can learn only the latest query answers

How:

- Randomness in the exponent

Application: ZK Accumulator-Why



1. Verifier is ZK adversary
2. ZK accumulator is managed by a trusted entity
3. Randomness harms efficiency

Application: ZK Accumulator- the need for randomness

1. First element's witness is g (generator is public information)
2. A guess about S can be verified with public information
3. A witness can be updated with public information (still valid?)

Public parameters: $g, g^{sk}, (g^{sk})^2, (g^{sk})^3, \dots$

$$S = \{x_1, x_2\}$$

$$acc.v = g^{(x_1+sk)(x_2+sk)} = g^{x_1x_2+(x_1+x_2)sk+sk^2} = g^{x_1x_2}(g^{sk})^{x_1+x_2}(g^{sk})^2$$

Application: ZK Accumulator- the need for randomness

MANAGER

$$\text{Acc.v=}g^{r(x_1+sk)(x_2+sk)}$$

$S = \{x_1, x_2\}$

Add(x)

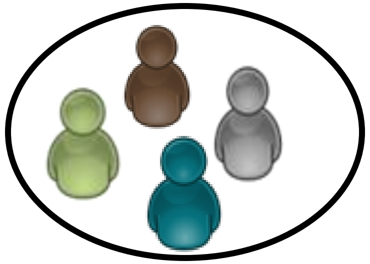
$$\text{Acc.v=}g^{r\,\boldsymbol{r'}(x_1+sk)(x_2+sk)(\boldsymbol{x}+sk)}$$

$S' = \{x_1, x_2, \boldsymbol{x}\}$

Verification works!

aux'={rr'}

Set S (aux=r)



Application: ZK Accumulator-Non-membership usually

Bilinear **ATSM09**, S={x_i}, $x_i \in \mathbb{Z}_p$

$$\prod_{i=1}^{|S|}(x_i+sk) = c(sk)(y+sk)+d$$

Users (public parameters):

S={x_i}, polynomial division

Manager (sk):

$\prod_{i=1}^{|S|}(x_i+sk) \in \mathbb{Z}$, used as exponent

$$g^u$$

$$u = \prod_{i=1}^{|S|}(x_i+sk) = c(sk)(y+sk)+d, \qquad d \neq 0$$

$$g^{ru}$$

Non-membership verification requires r

Application: ZK Accumulator-Non-membership by Ghosh et al

$$S \cap \{x\} \neq \emptyset$$

$$q_1[sk]\prod_{i=1}^{|S|}(x_i + sk) + q_2[sk]\,(y + sk) = 1$$

$$\overline{w_y} = (W_1, W_2) = \left(g^{(q_1[sk] + \gamma(y+sk))\,\boldsymbol{r^{-1}}}, g^{q_2[sk] - \gamma\prod_{i=1}^{|S|}(x_i + sk)}\right)$$

$$e(W_1, \boldsymbol{Acc.v})\,e\left(W_2, g^x g^{sk}\right) = e(g, g)$$

Remove accumulator randomness

Application: ZK Accumulator- Non-membership by Ghosh et al

$$S \cap \{x\} \neq \emptyset$$

$$q_1[sk] \prod_{i=1}^{|S|} (x_i + sk) + q_2[sk]\,(y + sk) = 1$$

$$\overline{w_y} = (W_1, W_2) = \left(g^{(q_1[sk] + \gamma(y+sk))\, r^{-1}},\, g^{q_2[sk] - \gamma \prod_{i=1}^{|S|}(x_i + sk)}\right)$$

$$e(W_1, Acc.v)\, e\left(W_2, g^x g^{sk}\right) = e(g, g)$$

Add query/witness specific randomness

Application: ZK Accumulator- Non-membership by Ghosh et al

(+) r not needed for verification

(-) no witness update algorithm

Update → NonMemWitCreate: O(|S|)

$\gamma \prod_{i=1}^{|S|}(x_i+sk)$)

Add query/witness specific randomness

$$e(W_1, Acc.v)e\left(W_2, g^x g^{sk}\right) = e(g,g)$$

Application: ZK Accumulator-Modular Construction

(+) r not needed for verific

(-) no witness update algo

Update → NonMemWitCreat

$$e(W_1, Acc.v)e\left(W_2, g^x g^{sk}\right) = e(g,$$

Solution:

instantiate our generic modular universal construction with ZK accumulators with membership operations

Result:

Non-membership witness creation, Non-membership witness update: O(1)

Summary

In the secret key model:

1. We can hit optimal communication cost (*Positive Bilinear Accumulator with Optimal Communication Cost*)
2. We can have constant non-membership (*Universal Bilinear Accumulator with Constant Non-Membership Witness Creation*)
3. We can have constant ZK (*ZK Accumulator with Constant Non-Membership Witness Creation and Update*)

Available on ePrint soon