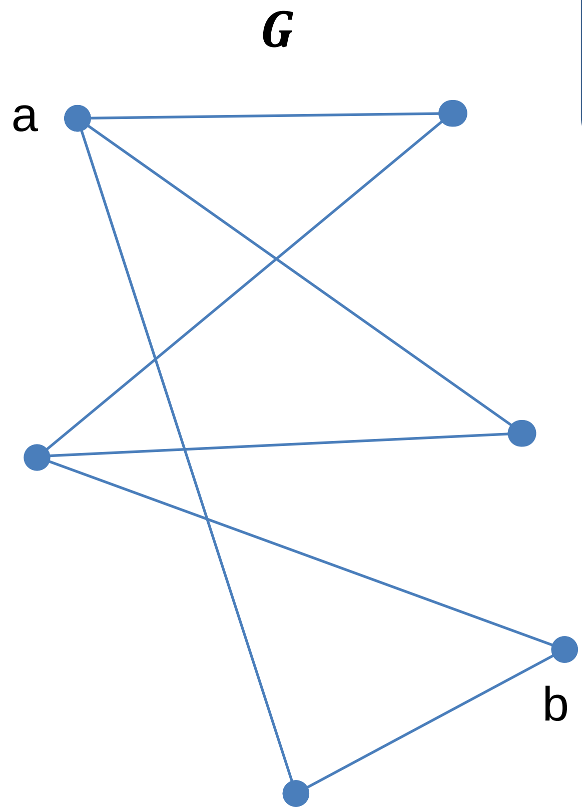
RSA CONFERENCE Where The World Talks Security
San Francisco, Feb 27-Mar 2 2012

Short Transitive Signatures for Directed Trees

Philippe Camacho and Alejandro Hevia
University of Chile

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
UNIVERSIDAD DE CHILE

How do we sign a graph?



Trivial solutions

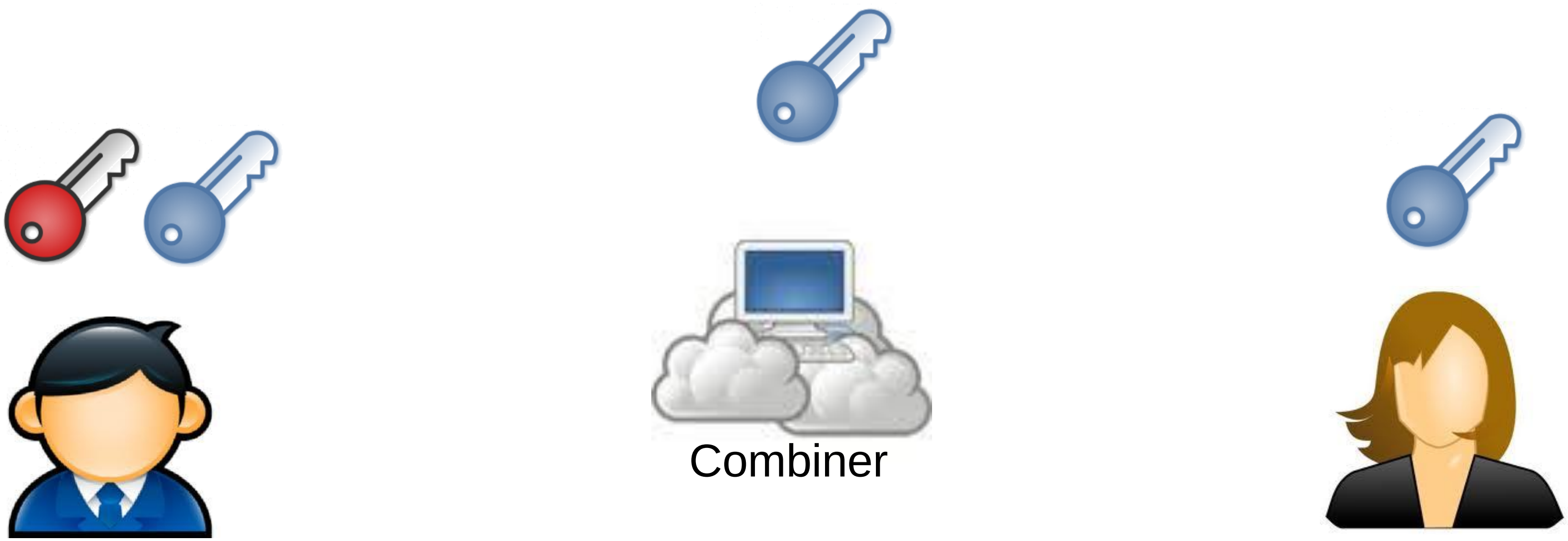
Let $\boldsymbol{n} = |\boldsymbol{G}|$, security parameter $\boldsymbol{\kappa}$

When adding a new node…

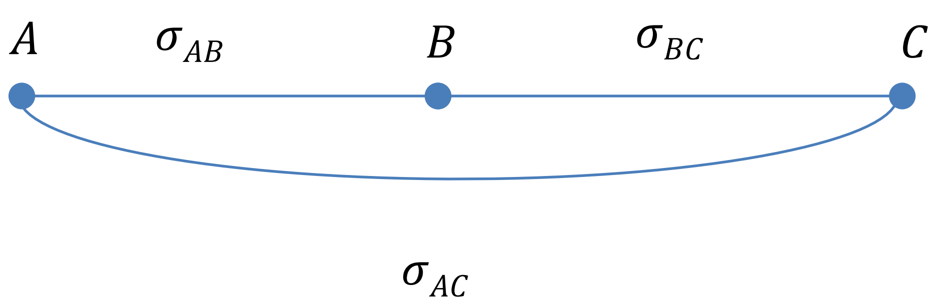
- Sign each edge
  - Time to sign: $\boldsymbol{O}(\mathbf{1})$
  - Size of signature: $\boldsymbol{O}(\boldsymbol{n\kappa})$ bits

- Sign each path
  - Time to sign (new paths): $\boldsymbol{O}(\boldsymbol{n})$
  - Size of signature: $\boldsymbol{O}(\boldsymbol{\kappa})$ bits

Transitive signature schemes [MR02,BN05,SMJ05]



$\sigma_{XY} \leftarrow TSign(X, Y, \sigma_{XY},)$ $\sigma_{AC} \leftarrow Combine(\sigma_{AB}, \sigma_{BC},)$ $\leftarrow TVerify(A, C, \sigma_{AC},)$



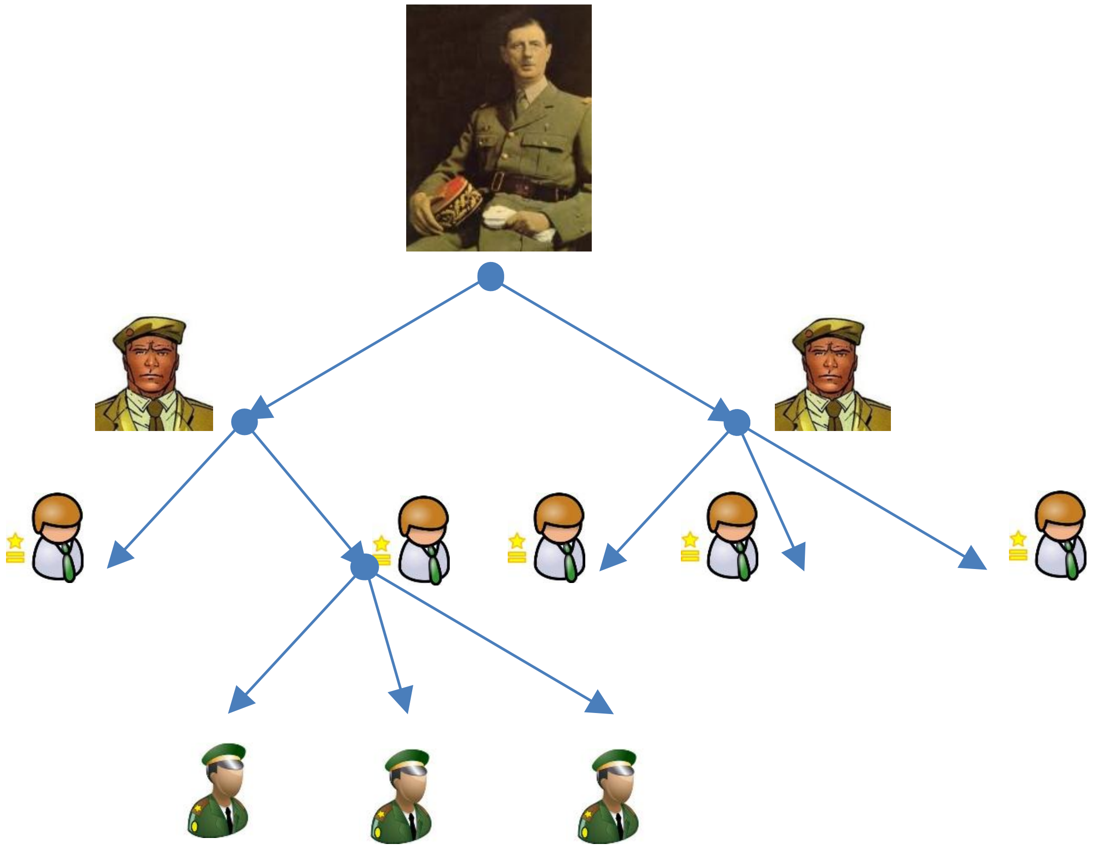
Landscape

- **[MR02,BN05,SMJ05]**
  for UNDIRECTED graphs

- Transitive Signatures for
  Directed Graphs (DTS) still OPEN

- **[Hoh03]**
  DTS $\Rightarrow$ Trapdoor Groups with
  Infeasible Inversion

Transitive Signatures for Directed Trees



Previous Work

- **[Yi07]**
  - Signature size: $\boldsymbol{O}(\boldsymbol{n} \log(\boldsymbol{n} \log \boldsymbol{n}))$ bits
    - Better than $\boldsymbol{O}(\boldsymbol{n\kappa})$ bits for the trivial solution
  - RSA related assumption

- **[Neven08]**
  - Signature size: $\boldsymbol{O}(\boldsymbol{n} \log \boldsymbol{n})$ bits
  - Standard Digital Signatures

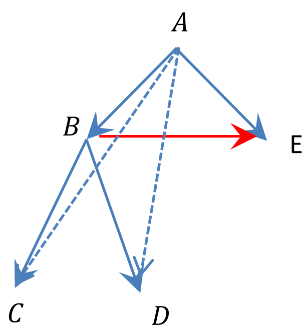
$\boldsymbol{O}(\boldsymbol{n} \log \boldsymbol{n})$ bits still impractical

Our Results

- For $\boldsymbol{\lambda} \geq \mathbf{1}$
  - Time to sign edge / verify path signature: $\boldsymbol{O}(\boldsymbol{\lambda})$
  - Time to compute a path signature: $\boldsymbol{O}(\boldsymbol{\lambda}(\boldsymbol{n}/\boldsymbol{\kappa})^{1/\lambda})$
  - Size of path signature: $\boldsymbol{O}(\boldsymbol{\lambda}\ \boldsymbol{\kappa})$ bits

Examples	$\lambda = 1$	$\lambda = 2$	$\lambda = \log(n)$
Time to sign edge / verify path signature	$\boldsymbol{O}(\mathbf{1})$	$\boldsymbol{O}(\mathbf{1})$	$\boldsymbol{O}(\log \boldsymbol{n})$
Time to compute a path signature	$\boldsymbol{O}(\boldsymbol{n}/\boldsymbol{\kappa})$	$\boldsymbol{O}(\sqrt{\boldsymbol{n}/\boldsymbol{\kappa}})$	$\boldsymbol{O}(\log \boldsymbol{n})$
Size of path signature	$\boldsymbol{O}(\boldsymbol{\kappa})$	$\boldsymbol{O}(\boldsymbol{\kappa})$	$\boldsymbol{O}(\boldsymbol{\kappa} \log \boldsymbol{n})$

Security [MR02]



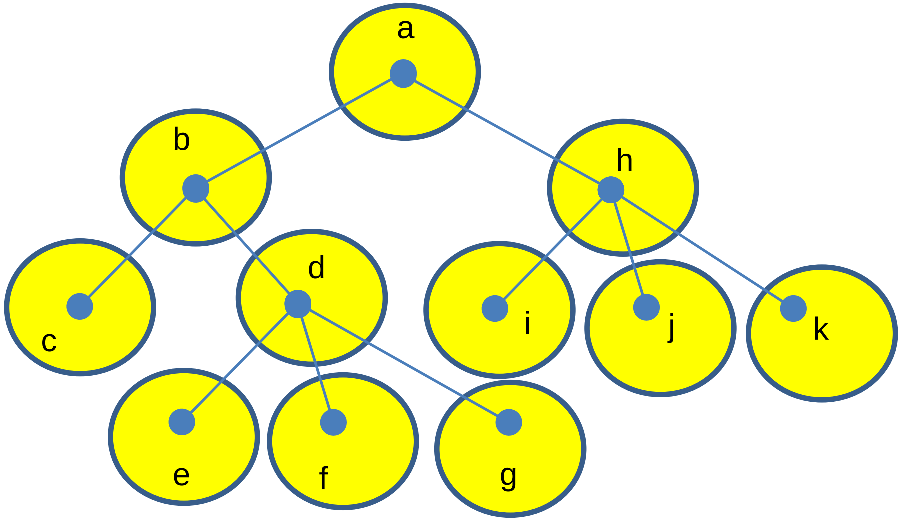
(σ^*, B, E):

$\leftarrow TVerify(B, E, \sigma^*, \;)$ and

There is **no path** from ***B*** to ***E***

BASIC CONSTRUCTION

Pre/Post Order Tree Traversal



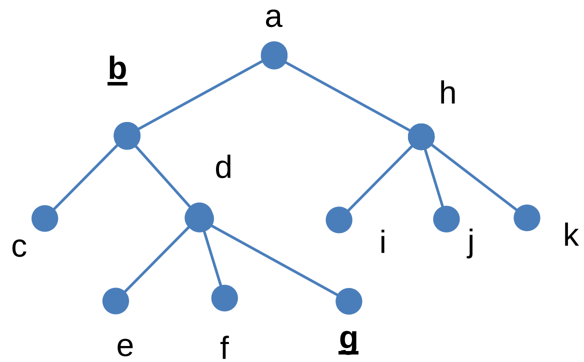
Pre order: a b c d e f g h i j k

Post order: c e f g d b i j k h a

Property of Pre/Post order Traversal

- **Proposition** [Dietz82]

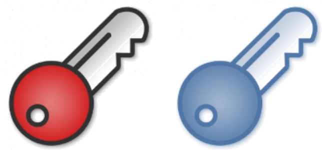
There is a path from $\boldsymbol{x}$ to $\boldsymbol{y}$ $\Longleftrightarrow$ $\boldsymbol{pos}(\boldsymbol{x}) < \boldsymbol{pos}(\boldsymbol{y})$ in $\boldsymbol{Pre}$ $\boldsymbol{pos}(\boldsymbol{y}) < \boldsymbol{pos}(\boldsymbol{x})$ in $\boldsymbol{Post}$



Pre order: a **b** c d e f **g** h i j k

Post order: c e f **g** d **b** i j k h a

Idea



- Compute $\boldsymbol{pos(g)}$ in $\boldsymbol{Pre}$ and $\boldsymbol{Post}$
- Sign $\boldsymbol{g||7||4}$ and resign **values** that have changed

Is there a path from a to e?

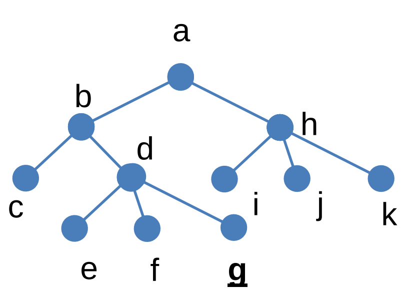
Signature of path $(\boldsymbol{a}, \boldsymbol{e})$:

- Signature of $\boldsymbol{a||1||11}$
- Signature of $\boldsymbol{e||5||2}$

- Check signatures
- Check

$$\mathbf{1} < \mathbf{5}$$
$$\mathbf{11} > \mathbf{2}$$

$\boldsymbol{G}$



Position	1	2	3	4	5	6	7	8	9	10	11
Pre	a	b	c	d	e	f	h	i	j	k	k
Post	c	e	f	d	b	i	j	k	h	a	a

How do we avoid recomputing a lot of signatures when an element is inserted?

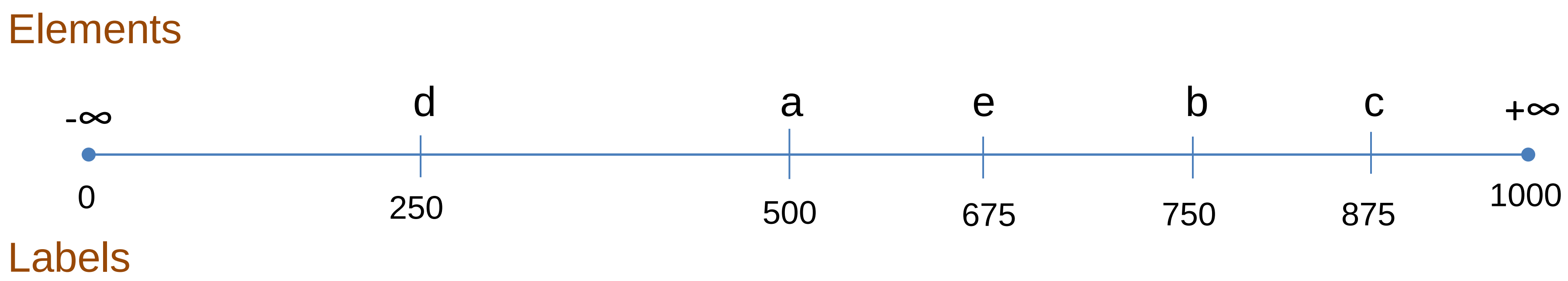
Order Data Structure

- Enables to
  - Insert elements **dynamically**
  - Compare them efficiently

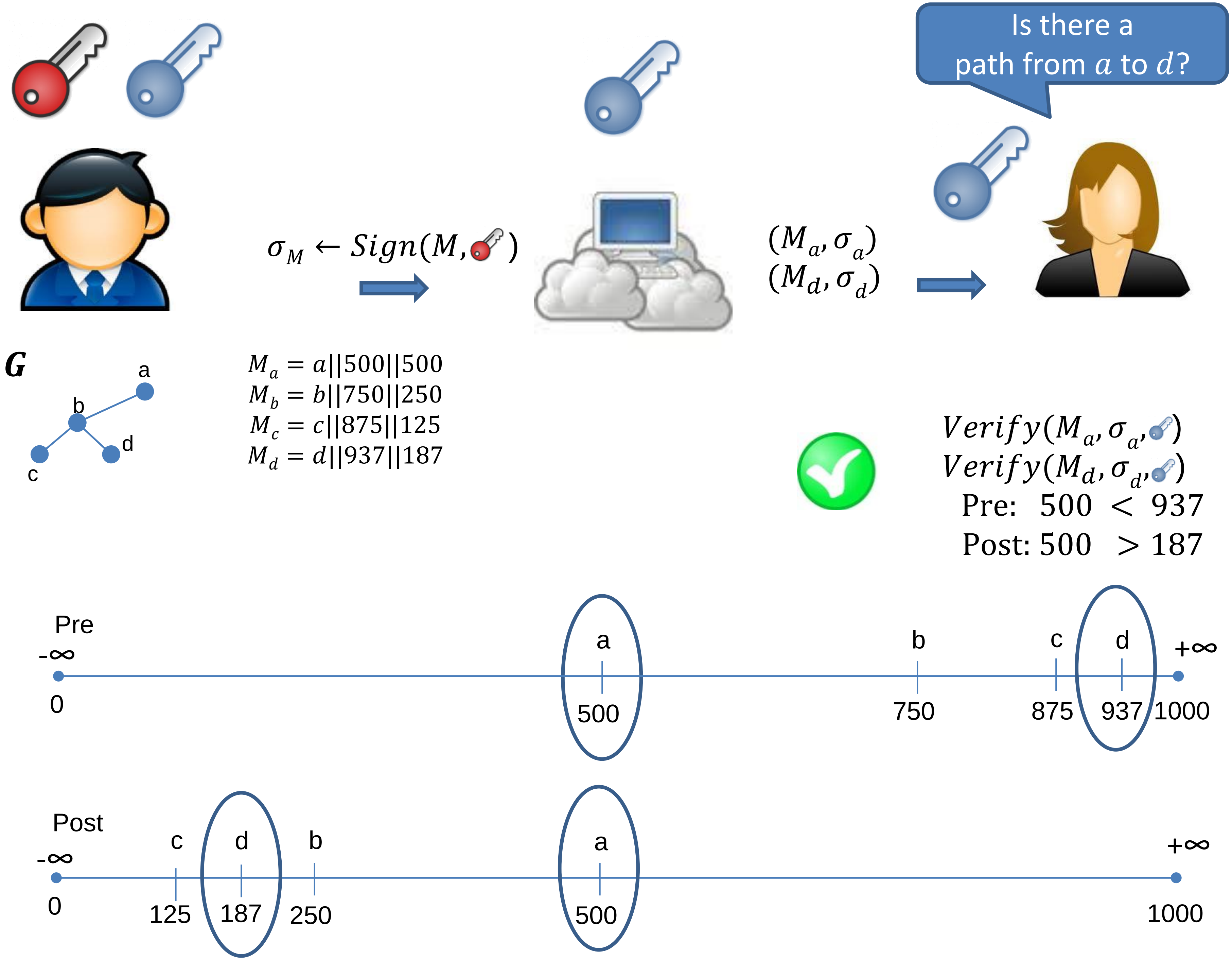
- **Definition [Dietz82, MR+02]**
  - $ODInsert(X, Y)$
  - $ODCompare(X, Y)$

Trivial Order Data Structure

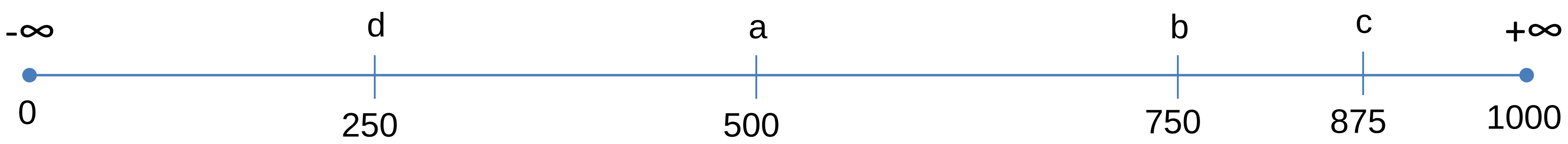
A Toy Example



For n insertions we need to handle n bits


Is there a path from a to d?
$\sigma_M \leftarrow Sign(M, \cdot)$
(M_a, σ_a)
(M_d, σ_d)
G
a
b
c
d
$M_a = a||500||500$
$M_b = b||750||250$
$M_c = c||875||125$
$M_d = d||937||187$
$Verify(M_a, \sigma_a, \cdot)$
$Verify(M_d, \sigma_d, \cdot)$
Pre: $500 < 937$
Post: $500 > 187$
Pre
-∞
+∞
0
500
750
875
937
1000
Post
125
187
250

Trivial Order Data Structure



- Signature of size $O(n)$
- Better than $O(n \log n)$ **[Neven08]**, but still room for improvement.

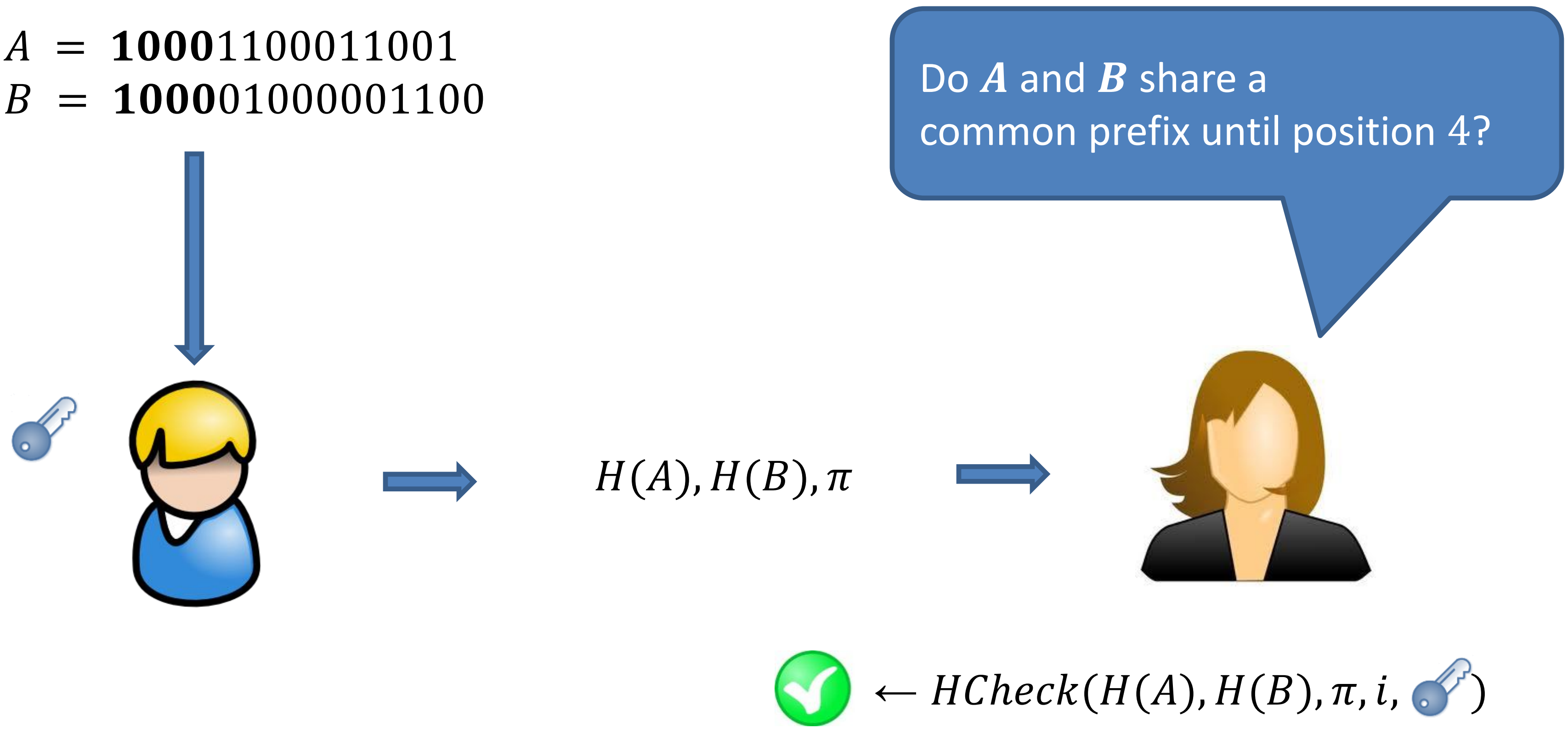
New CRHF! It allows to:

- compress the strings
- efficiently compare them from their hashes

HASHING WITH COMMON PREFIX PROOFS

The Idea

$A = \mathbf{1000}1100011001$
$B = \mathbf{1000}01000001100$



We want:

$\boldsymbol{H}$ collision resistant hash function + proofs

Security

$HGen(1^{\kappa}, n) \rightarrow PK$

(A, B, i, π)

$$Adv(A) = \Pr\left[\begin{array}{c} HCheck(H(A), H(B), \pi, i, PK) = True \\ \wedge \\ A[1..i] \neq B[1..i] \end{array}\right]$$

n-BDHI assumption [BB04]

$\boldsymbol{e}: \boldsymbol{G} \times \boldsymbol{G} \rightarrow \boldsymbol{G_T}$
$\boldsymbol{s} \leftarrow \boldsymbol{Z_p}$
$\boldsymbol{g}$ generator of $\boldsymbol{G}$
$(\boldsymbol{g^s}, \boldsymbol{g^{s^2}}, \dots, \boldsymbol{g^{s^n}})$

$$\boldsymbol{e(g, g)^{1/s}}$$

The hash function

- $\boldsymbol{HGen(1^\kappa, n)}$

$(\boldsymbol{p}, \boldsymbol{G}, \boldsymbol{G_T}, \boldsymbol{e}, \boldsymbol{g}) \leftarrow BMGen(1^\kappa)$

$\boldsymbol{s} \leftarrow \boldsymbol{Z_p}$
$\boldsymbol{T} := (\boldsymbol{g^s}, \boldsymbol{g^{s^2}}, \dots, \boldsymbol{g^{s^n}})$

return $\boldsymbol{PK} := (\boldsymbol{p}, \boldsymbol{G}, \boldsymbol{G_T}, \boldsymbol{e}, \boldsymbol{g}, \boldsymbol{T})$

- $\boldsymbol{HEval(M, PK)}$

$$\boldsymbol{H(M)} := \prod_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{g^{M[i]s^i}}$$

Toy example: $\boldsymbol{M = 1001} \Rightarrow \boldsymbol{H(M) = g^s . g^{s^4}}$

Generating & Verifying Proofs

- $\boldsymbol{A} = \boldsymbol{A}[\mathbf{1..n}] = \mathbf{1000111001}$
- $\boldsymbol{B} = \boldsymbol{B}[\mathbf{1..n}] = \mathbf{1000101100}$

- $\boldsymbol{\Delta} := \frac{H(A)}{H(B)} = \frac{g^{s}g^{s^5}g^{s^6}g^{s^7}g^{s^{10}}}{g^{s}g^{s^5}g^{s^7}g^{s^8}} = g^{s^6}\, g^{-s^8}\, g^{s^{10}}$

- $\boldsymbol{\Delta} = \prod_{j=1}^{n} g^{C[j]s^j}$ with $C = [0, 0, 0, 0, 0, 1, 0, -1, 0, 1]$

Generating & Verifying Proofs

- $\boldsymbol{\Delta} = \prod_{j=1}^{n} g^{C[j]s^j}$ with $\boldsymbol{C} = [0, 0, 0, 0, 0, 1, 0, -1, 0, 1]$

- "Remove" factor $\mathbf{s}^{\mathbf{i+1}}$ in the exponent <u>without knowing</u> $\boldsymbol{s}$

$$\boldsymbol{\pi} := \boldsymbol{\Delta}^{\frac{1}{s^{i+1}}} = \prod_{j=i+1}^{n} g^{C[j]s^{j-i-1}} = g\, g^{-s^2} g^{s^4}$$

- Check the proof : $\boldsymbol{e}(\boldsymbol{\pi}, g^{s^{i+1}}) = \boldsymbol{e}(\boldsymbol{\Delta}, \boldsymbol{g})$

Security

- **Proposition:**
  If the n-BDHI assumption holds then the previous construction is a secure HCPP family.

- Proof (idea)

$$\begin{aligned} \mathbf{A} &= \mathbf{100010} \\ \mathbf{B} &= \mathbf{101001} \\ \mathbf{i} &= \mathbf{3} \end{aligned}$$

$$\mathbf{H(A)} = \mathbf{g}^{s}\,\mathbf{g}^{s^5}$$

$$\mathbf{H(B)} = \mathbf{g}^{s}\,\mathbf{g}^{s^3}\,\mathbf{g}^{s^6}$$

$$\boldsymbol{\Delta} = \frac{\mathbf{H(A)}}{\mathbf{H(B)}} = \mathbf{g}^{-s^3}\,\mathbf{g}^{s^5}\,g^{-s^6}$$

$$\boldsymbol{\pi} = \boldsymbol{\Delta}^{\frac{1}{s^4}} = \mathbf{g}^{-1/s}\,\mathbf{g}^{s}\,\mathbf{g}^{s^2}$$

CRHF is incremental

$$A = \mathbf{1000}$$

$$B = \mathbf{1000}1$$

$$\boldsymbol{H}(\boldsymbol{B}) = \boldsymbol{H}(\boldsymbol{A})\ \boldsymbol{g}^{s^{\mathbf{5}}}$$

It's fast to compute $\boldsymbol{H}(\boldsymbol{B})$ from $\boldsymbol{H}(\boldsymbol{A})$
(we don't need the preimage $\boldsymbol{A}$)

Comparing strings

- $\boldsymbol{A < B \Leftrightarrow \; CommonPrefix(A, B, i) \;\; \wedge \;\; A[i+1] < B[i+1]}$

$$\left.\begin{aligned} \text{E.g:}\;\; A &= \mathbf{100}01 \\ B &= \mathbf{100}10 \end{aligned}\right\} \quad C = 100$$

- Check:

$e(H(A)/H(C), g) \;=\; e(\pi_1, g^{s^4})$ // C is a prefix of A

$e(H(B)/H(C), g) \;=\; e(\pi_2, g^{s^4})$ // C is a prefix of B

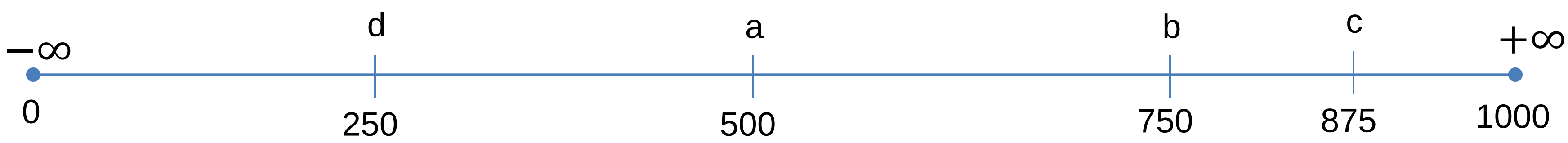
$e(H(C)H(0^3||0)/H(A), g) \;=\; e(\pi_3, g^{s^5})$ // $C||0$ is a prefix of A

$e(H(C)H(0^3||1)/H(B), g) \;=\; e(\pi_4, g^{s^5})$ // $C||1$ is a prefix of B

$0 < 1$

FULL CONSTRUCTION

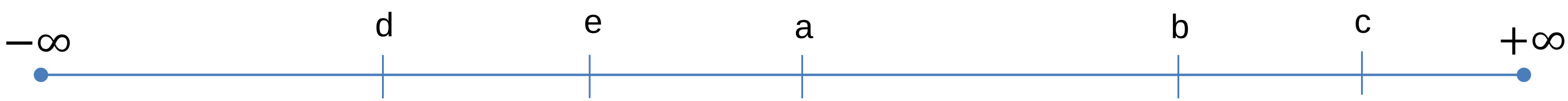
Trivial Order Data Structure



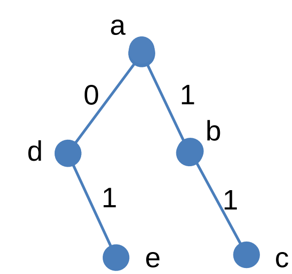
Signer has to compute new labels before hashing them
$\Rightarrow$ Time to sign an edge still $\boldsymbol{O(n)}$.

New Order Data Structure:
$\boldsymbol{ODInsert(X,Y)}$ s.t. new label $\boldsymbol{Z}$
shares every bit except one with $\boldsymbol{X}$ or $\boldsymbol{Y}$

New Order Data Structure

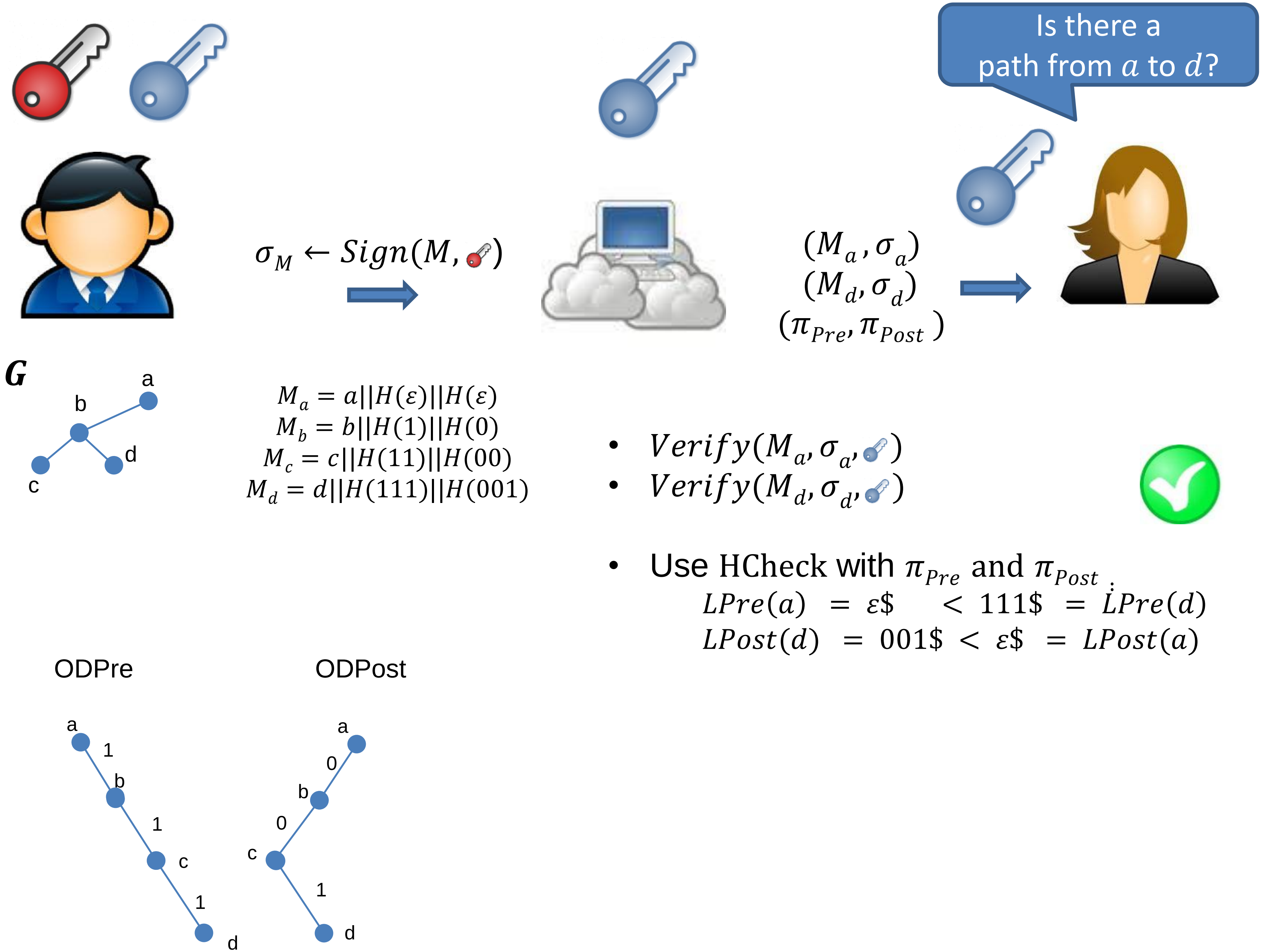


Use a binary tree to obtain an «incremental» order data structure



$$L(a) = \varepsilon$$
$$L(b) = 1$$
$$L(c) = 11$$
$$L(d) = 0$$
$$L(e) = 01$$

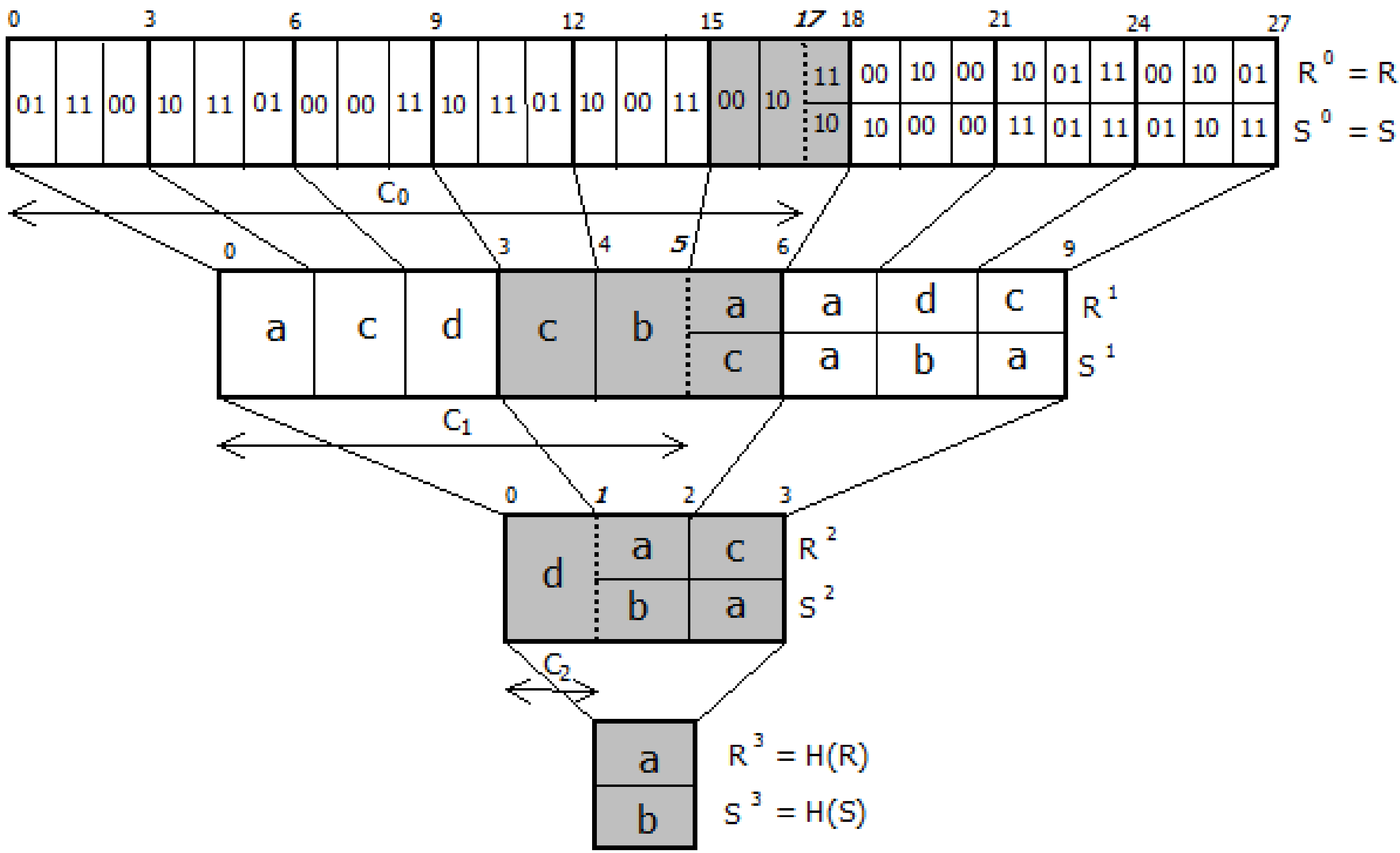
$$0 < \$ < 1$$
$$L(d) = 0\$ < L(a) = \varepsilon\$$$
$$L(d) = 0\$ < L(b) = 1\$$$
$$L(b) = 1\$ < L(c) = 11\$$$
$$L(e) = 01\$ < L(a) = \varepsilon\$$$



Is there a path from a to d?
$\sigma_M \leftarrow Sign(M, \cdot)$
(M_a, σ_a)
(M_d, σ_d)
(π_{Pre}, π_{Post})
G
a
b
c
d
$M_a = a||H(\varepsilon)||H(\varepsilon)$
$M_b = b||H(1)||H(0)$
$M_c = c||H(11)||H(00)$
$M_d = d||H(111)||H(001)$
• $Verify(M_a, \sigma_a, \cdot)$
• $Verify(M_d, \sigma_d, \cdot)$
• Use HCheck with π_{Pre} and π_{Post}
$LPre(a) = \varepsilon\$ < 111\$ = LPre(d)$
$LPost(d) = 001\$ < \varepsilon\$ = LPost(a)$
ODPre
ODPost
a
1
b
1
c
1
d
a
0
b
0
c
1
d

Trade off

$$n = 54, \quad \kappa = 2, \quad \Sigma = \{a, b, c, d\}$$
$$n/\kappa = 54/2 = 27$$
$$\lambda = 3 \Rightarrow (n/\kappa)^{1/\lambda} = 3$$



Conclusion and Open Problems

- Efficient transitive signature scheme for directed trees
- Possible to balance the time to compute and to verify the proof
- Based on a general new primitive HCPP
- New constructions / applications for HCPP
- Can we improve the trade off?
- **Stateless** transitive signatures for directed trees

Thank you!

Short Attribute-Based Signatures for Threshold Predicates

Javier Herranz[1], Fabien Laguillaumie[2], Benoît Libert[3], **Carla Ràfols**[4]

[1] Univ. Politècnica Catalunya
[2] Univ. Caen Basse-Normandie
[3] Univ. Catholique de Louvain
[4] Ruhr Univ. Bochum, previously Univ. Rovira i Virgili

CT-RSA, San Francisco, Feb. 28th

Some history: IBE $\to$ fuzzy IBE $\to$ ABE

Ciphertext	$\leftrightarrow$	**vector$_1$**	**Secret key**	$\leftrightarrow$	**vector$_2$**
C	$\leftrightarrow$	$\vec{v}_1$	sk	$\leftrightarrow$	$\vec{v}_2$

Decryption works $\iff$ $R(\vec{v}_1, \vec{v}_2) = 1$

Identity Based Encryption (Shamir, 1984)

fuzzy Identity Based Encryption (Sahai-Waters, 2005)

(Threshold) **Attribute Based Encryption** (Goyal et al., 2006)

(Threshold) Attribute Based Signatures

The key of user U is associated with some attributes
$\mathsf{sk}_U \longleftrightarrow S_U = \{\mathsf{at}_1, \ldots, \mathsf{at}_m\}$

Each signature σ associated with a **predicate** $\Gamma = (S, t)$ or **signing policy**, where

$S = \{\mathsf{at}_{i_1}, \ldots, \mathsf{at}_{i_s}\}$	the set of attributes of Γ
$t \in \mathbb{N}$, $1 \leq t \leq s$.	is the threshold of Γ.

If σ is correctly verified then the signer's secret key sk_U is such that $|S_U \cap S| \geq t$.

General Attribute Based Signatures

The key of user U is associated with some attributes
$\mathsf{sk}_U \longleftrightarrow S_U = \{\mathsf{at}_1, \ldots, \mathsf{at}_m\}$

Each signature σ is associated with a **predicate** Γ which maps any set of attributes to $\{0, 1\}$.

If σ is correctly verified then the signer's secret key sk_U satisfies that $\Gamma(S_U) = 1$.

APPLICATIONS: proving you are entitled to something but not saying **who** you are: anonymous credentials & access control..

ABS: Syntactic definition

- $\mathsf{ABS.Setup}(1^\lambda, \mathcal{U}^*) \to (\mathsf{params}, \mathsf{msk})$: Run by master entity on input $\mathcal{U}^*$, the universe of attributes. The string params includes a description of **admissible signing policies**.
- $\mathsf{ABS.Ext}(\mathsf{params}, U, \mathsf{msk}) \to sk_U$: user U proves to master entity possession of his attributes S_U and gets sk_U.
- $\mathsf{ABS.Sign}(\mathsf{params}, \mathsf{sk}_U, \Gamma, M) \to \sigma$: signer chooses an admissible signing policy Γ, such that $\Gamma(S_U) = 1$.
- $\mathsf{ABS.Vrfy}(\mathsf{params}, \sigma, \Gamma) \to b$: outputs 1 iff σ is valid.

Correctness If $\sigma \leftarrow \mathsf{ABS.Sign}(\mathsf{params}, \mathsf{sk}_U, \Gamma, M)$ and $\Gamma(S_U) = 1$, then the verification outputs 1 .

Design goals

Construct ABS schemes which

1. admit large families of admissible signing policies **(expressive** signing policies),
2. while providing strong security guarantees **(security)**
3. and good performance **(efficiency)**

In practice there is a tradeoff between these properties....

Unforgeability against selective predicate and adaptive message attack

Initialize: Adversary $\mathcal{A}$ outputs (S^*, t^*).

Setup: Challenger $\mathcal{C}$ runs Setup, gives params to $\mathcal{A}$.

Queries: Adaptatively, $\mathcal{A}$ can request:

- Secret key queries for any user U such that $|\mathsf{sk}_U \cap S^*| < t^*$.
- Signature queries for any $\langle M, U, (S, t)\rangle$.

Output: $\mathcal{A}$ outputs a tuple (σ^*, M^*)

$\mathcal{A}$ wins if the signature is a non-trivial forgery for (S^*, t^*).

We also consider **privacy** of attributes: the signature does not leak which subset of size t of S was used to sign.

Our results

We give two constructions of ABS:

- with **constant size** signatures
- having unforgeability against selective predicate and adaptive message attacks
- for threshold access policies but extendable to larger families of predicates

Our result shows that **specific families** of predicates allow for more compact signatures.

Overview construction I

J. Herranz, F. Laguillaumie, C. Ràfols. Constant-size ciphertexts in threshold attribute-based encryption. In PKC'10.

Starting from ABE scheme with constant-size ciphertexts, the idea is to prove the ability to decrypt,

For this we use a technique of Malkin *et al.* (2011):

- Use Groth-Sahai zero-knowledge proofs,
- BUT, to link the proof to the message create a message dependent common reference string.

More specifically...

ABS.Setup chooses random generators $g_1, g_2 \leftarrow \mathbb{G}$ and defines $\overrightarrow{g_1} = (g_1, 1, g)^\top, \overrightarrow{g_2} = (1, g_2, g)^\top \in \mathbb{G}^3$ and $\{\overrightarrow{g_{3,i}} = \overrightarrow{g_1}^{\xi_{i,1}} \cdot \overrightarrow{g_2}^{\xi_{i,2}}\}_{i=0}^k$, for random $\xi_{i,1}, \xi_{i,2}$.

params include $\overrightarrow{g_1},\ \overrightarrow{g_2},\ \{\overrightarrow{g_{3,i}}\}_{i=0}^k$, where we write $\overrightarrow{g_{3,i}}$ as $\left(g_{X,i},\ g_{Y,i}, g_{Z,i}\right)^\top$.

ABS.Sign computes $H(M, \Gamma) = m_1 \ldots m_k \in \{0,1\}^k$, where H is Water's hash function and then computes the GS CRS related to this value:

$$\overrightarrow{g}_{3,\mathbf{m}} = \left(g_{X,0} \cdot \prod_{i=1}^k g_{X,i}^{m_i},\ g_{Y,0} \cdot \prod_{i=1}^k g_{Y,i}^{m_i},\ g_{Z,0} \cdot \prod_{i=1}^k g_{Z,i}^{m_i}\right)^\top.$$

Signer knows T_1, T_2 which satisfy a pairing product equation: prove knowledge with CRS $\left(\overrightarrow{g}_1,\ \vec{g}_2,\ \vec{g}_{3,\mathbf{m}}\right)$.

Idea of security proof...

Under DLIN, $\overrightarrow{g_1}$, $\overrightarrow{g_2}$, $\{\overrightarrow{g_{3,i}}\}_{i=0}^{k}$, is indistinguishable from some other set of vectors where $\overrightarrow{g_1}$, $\overrightarrow{g_2}$, $\overrightarrow{g_{3,i}}$ are lin. ind. for all $i \in \{0, 1, \ldots, k\}$.

With this other set of vectors we have a CRS which corresponds to the simulation setting of the DLIN instantiation of Groth Sahai proofs.

In the simulation setting we can answer signing queries.
adaptive message, DLIN

Secret key queries simulated as in the underlying ABE scheme.
selective predicate, aMSE-CDH problem

Overview construction II

N. Attrapadung, B. Libert, E. de Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In PKC'11.

Similar ideas:

- selective predicate security also inherited from underlying ABE.
- adaptive message comes from using Water's hash function as $H(M, \Gamma)$,
- security proof avoids Groth Sahai ZK proofs (efficiency gain in size of ciphertexts)

(Daza et al. 2011) Scheme can be extended to a much larger class of predicates: like hierarchical or compartmented. Good tradeoff efficiency/expression of signing policies.

Summary: Main features of our construction

1st construction

- Signature: **15** group elements; Key of user *U*: $\mathbf{n} + |\mathbf{S_U}|$ group elements, *n* bound on maximal size of signing policy.
- Unforgeability: DLIN + aMSE-CDH problems.
- Privacy of attributes: DLIN.
- Limited extension to weighted threshold access structures.

2nd construction

- Signature: **3** group elements; Key size of user *U*: $(\mathbf{2n} + \mathbf{2}) \times (|\mathbf{S_U}| + \mathbf{n})$.
- Unforgeability: *n*-Diffie-Hellman Exponent problem.
- Privacy of attributes: Unconditional.
- Hierarchical access structures, compartmented access structures (Daza et al, 2011).