Attribute-Based Signatures for Unbounded Languages from Standard Assumptions

Yusuke Sakai (AIST, Japan)
Shuichi Katsumata (AIST, Japan / U. Tokyo, Japan)
Nuttapong Attrapadung (AIST, Japan)
Goichiro Hanaoka (AIST, Japan)

Our Contribution

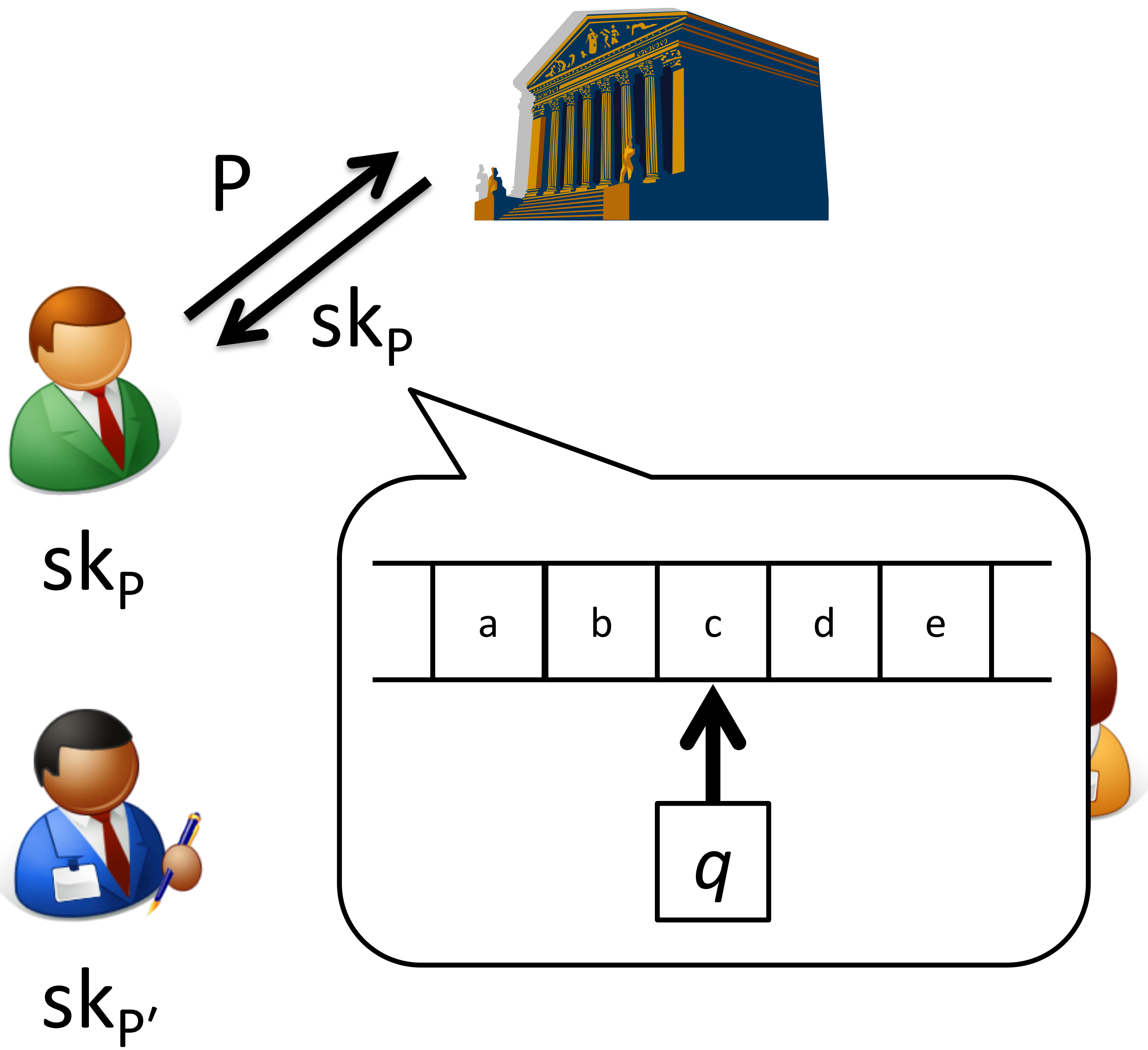
- Propose attribute-based signature scheme for ***Turing machines***
  - A key-policy variant
  - The policy is described by a Turing machine (TM)
  - The attribute is an input to a TM

The scheme allows policies that accept ***unbounded*** inputs!

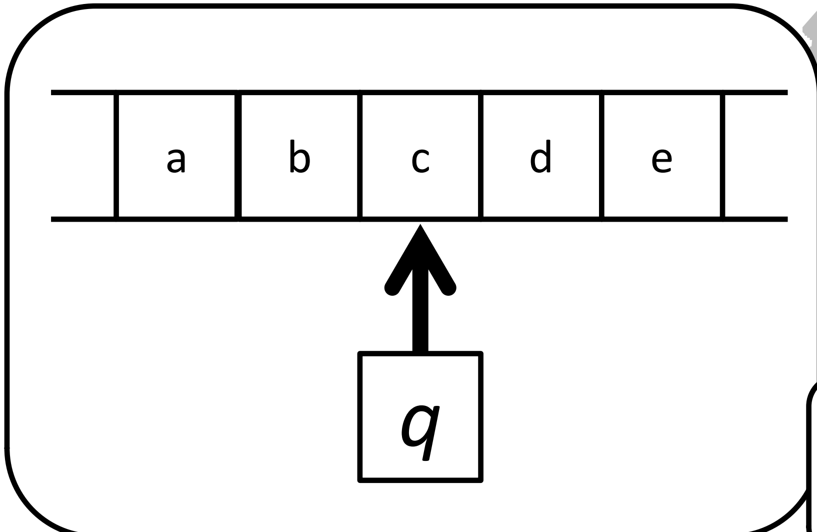
Agenda

- Attribute-Based Signatures
- Security Requirement
- Certificate Approach
- Idea 1: History of Computation
- Idea 2: Locality of Rewriting
- Overview of the Scheme
- Conclusion

Attribute-Based Signatures (ABS)



Attribute-Based Signatures

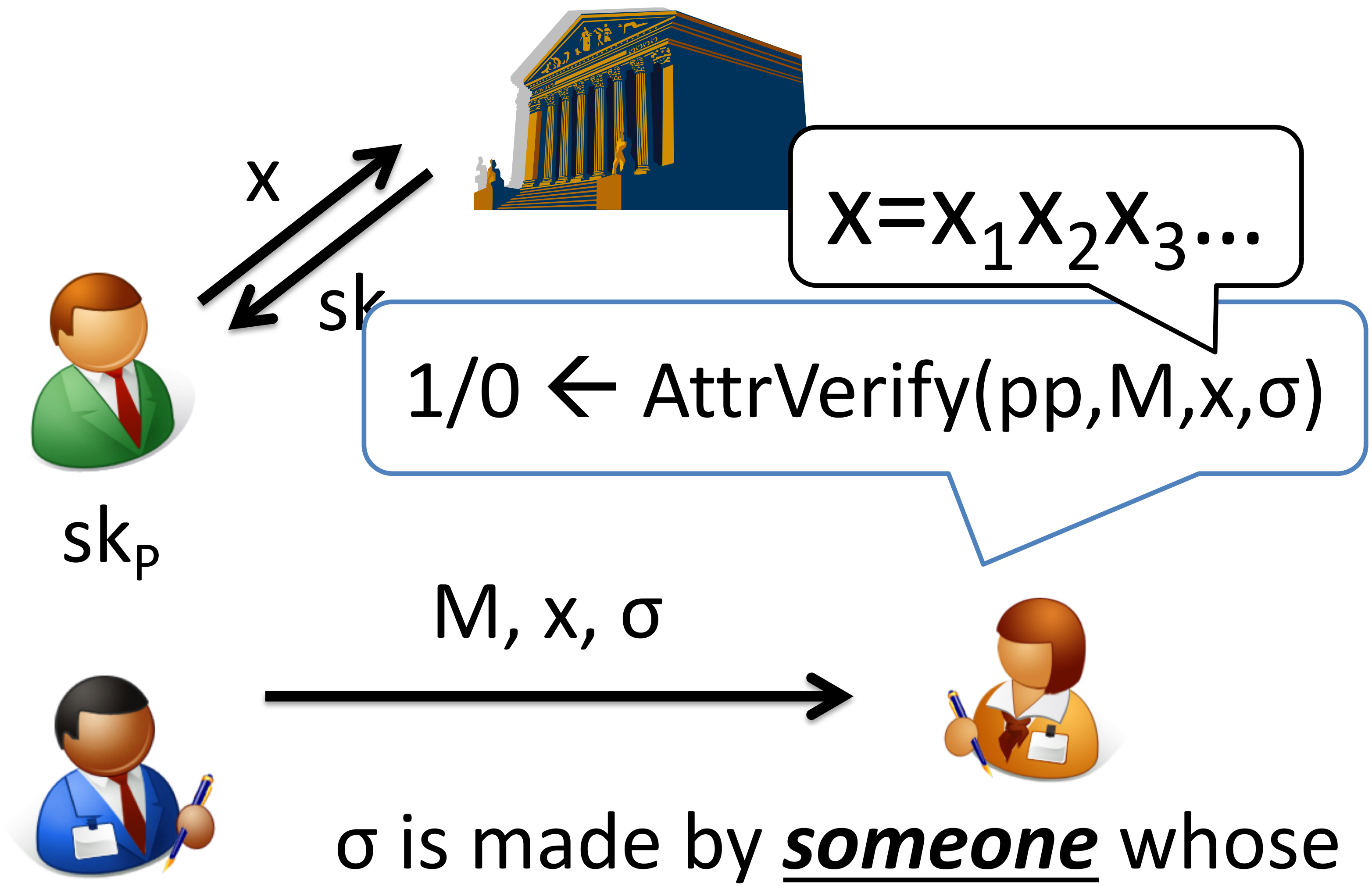


$x = x_1 x_2 x_3 \ldots$

$\sigma \leftarrow \mathrm{AttrSign}(pp, sk_P, M, x)$

$sk_{P'}$

Attribute-Based Signatures

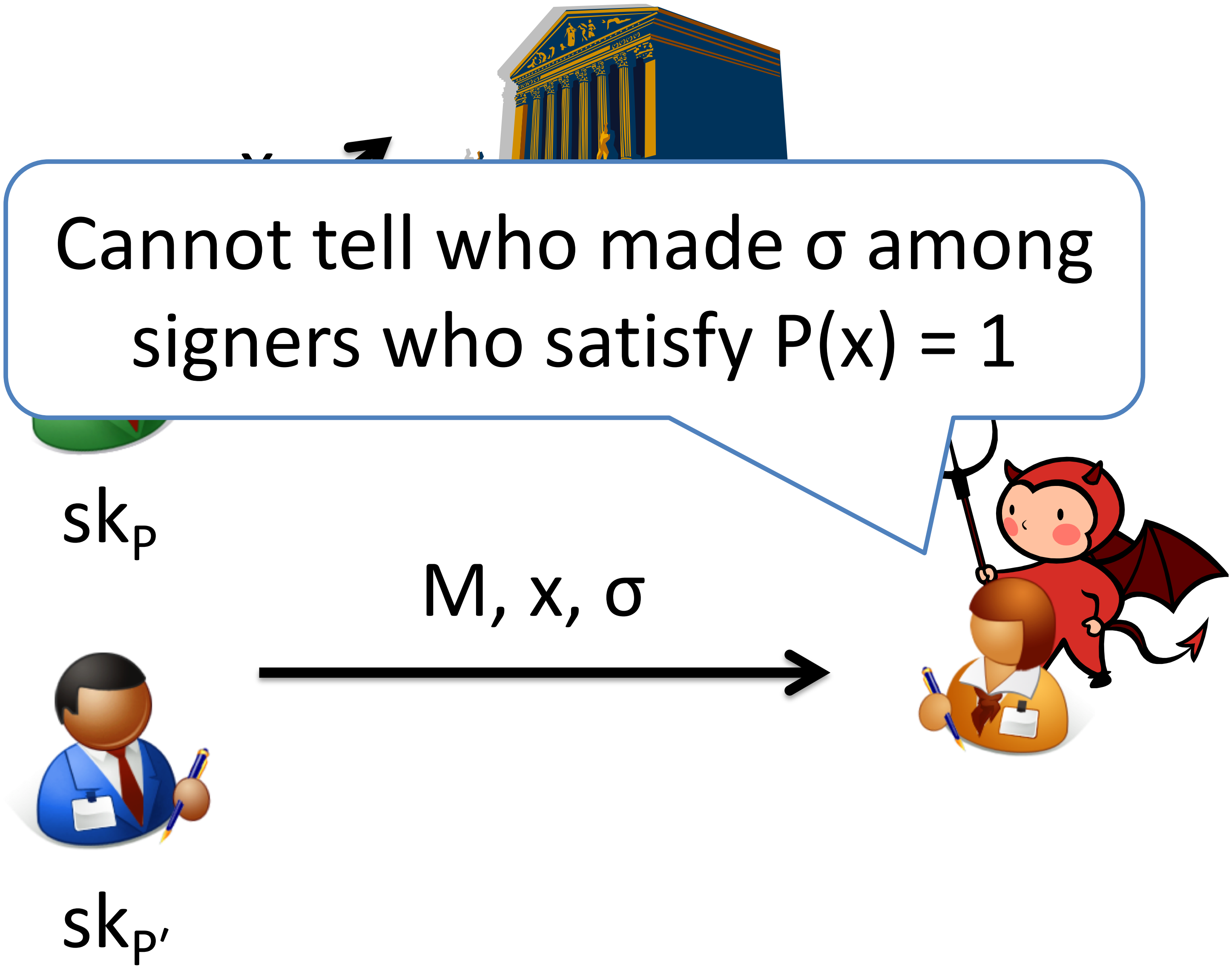


σ is made by ***<u>someone</u>*** whose policy P satisfy P(x) = 1

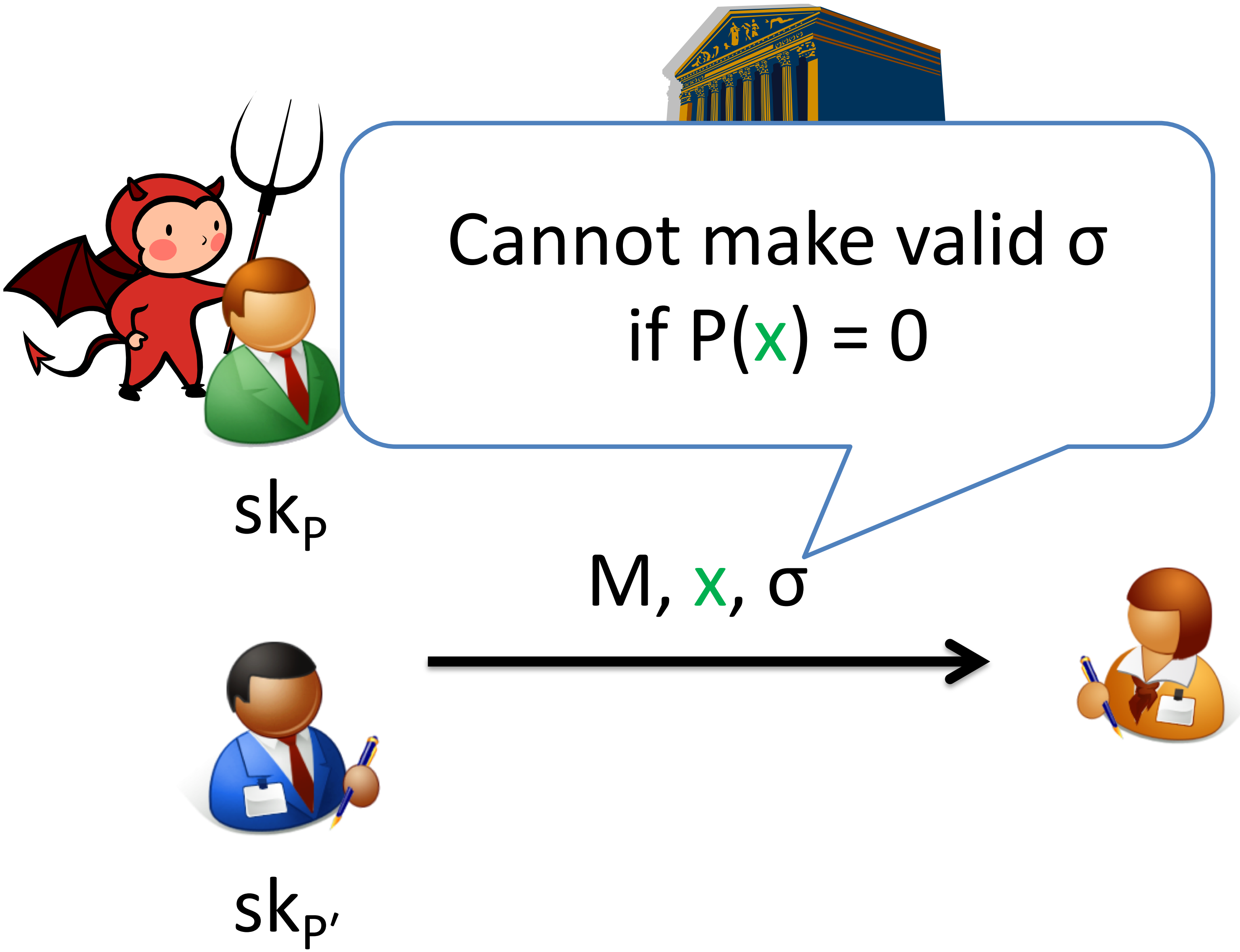
Agenda

- Attribute-Based Signatures
- Security Requirement
- Certificate Approach
- Idea 1: History of Computation
- Idea 2: Locality of Rewriting
- Overview of the Scheme
- Conclusion

Anonymity


Cannot tell who made σ among signers who satisfy P(x) = 1
sk_P
M, x, σ
$sk_{P'}$

Unforgeability



Agenda

- Attribute-Based Signatures
- Security Requirement
- Certificate Approach
- Idea 1: History of Computation
- Idea 2: Locality of Rewriting
- Overview of the Scheme
- Conclusion

Certificate Approach (1/2)

msk

x

Each signer receives ***<u>a signature on his policy</u>***

$sk_{P} = \theta_{P} = \mathrm{Sign}(msk, P)$

$sk_{P'} = \theta_{P'} = \mathrm{Sign}(msk, P')$

Certificate Approach (2/2)

msk

x

sk_P

Prove knowledge of (P, θ):

(1) $\mathrm{Verify}(P, \theta) = 1$

(2) $P(x) = 1$

$sk_P = \theta_P = \mathrm{Sign}(msk, P)$

M, x, σ

$sk_{P'} = \theta_{P'} = \mathrm{Sign}(msk, P')$

Difficulty

Prove knowledge of (P, θ_P):

(1) $\text{Verify}(P, \theta_x) = 1$

(2) $P(x) = 1$

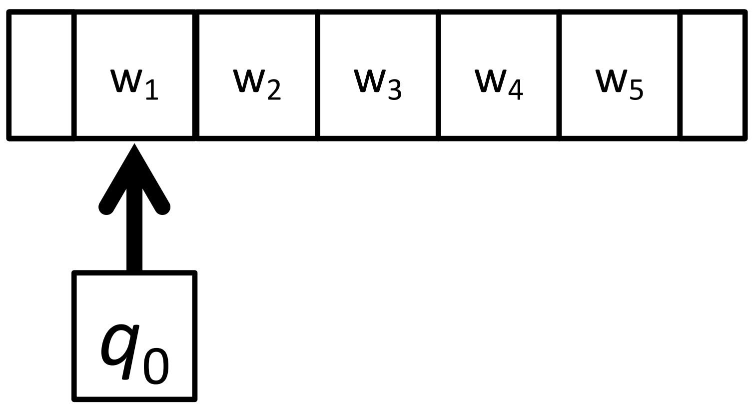
- How to prove the complex condition $P(x) = 1$
  - Remind that P is a Turing machine
- General zero-knowledge is inefficient, so we will ***decompose*** the statement

Agenda

- Attribute-Based Signatures
- Security Requirement
- Certificate Approach
- Idea 1: History of Computation
- Idea 2: Locality of Rewriting
- Overview of the Scheme
- Conclusion

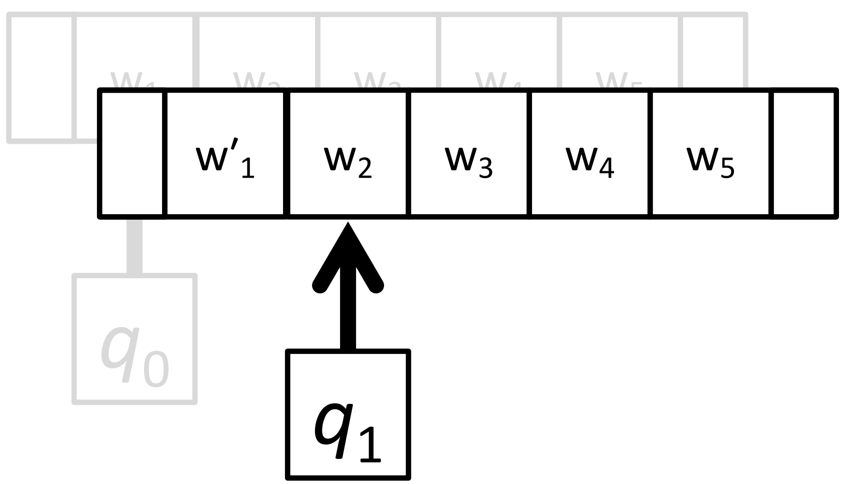
Idea: History of Computation

- While a TM's computation is complex, the computation proceeds sequentially
- The computation defines a sequence of "snapshots" of the machine



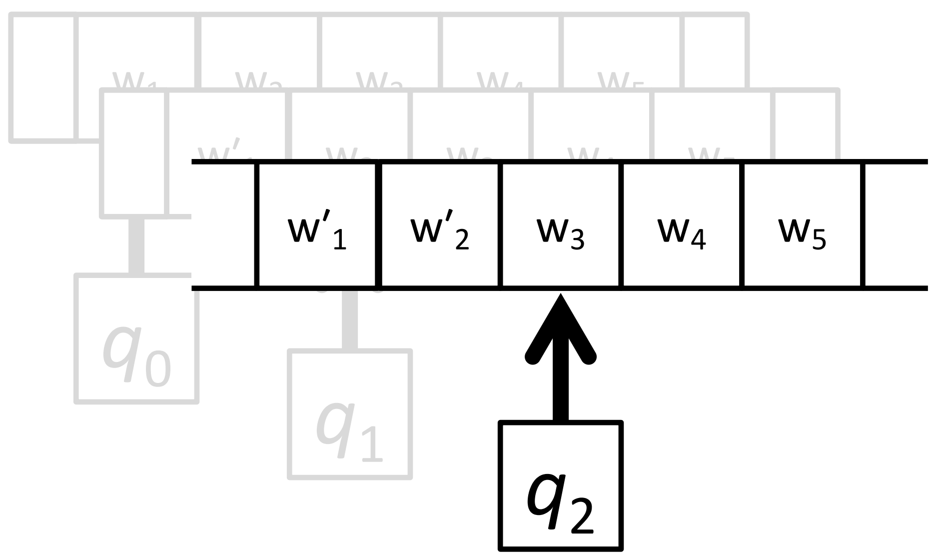
Idea: History of Computation

- While a TM's computation is complex, the computation proceeds sequentially
- The computation defines a sequence of "snapshots" of the machine



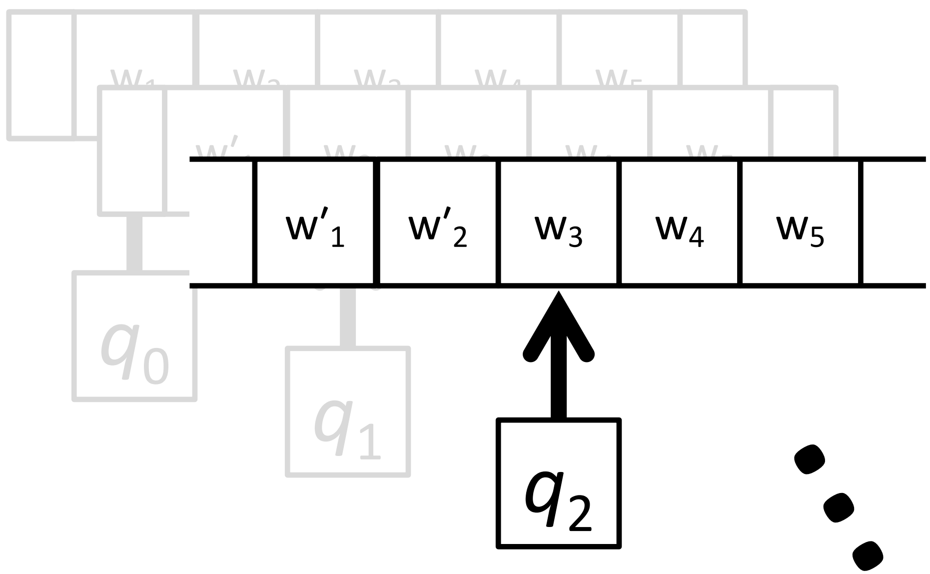
Idea: History of Computation

- While a TM's computation is complex, the computation proceeds sequentially
- The computation defines a sequence of "snapshots" of the machine



Idea: History of Computation

- While a TM’s computation is complex, the computation proceeds sequentially
- The computation defines a sequence of “snapshots” of the machine



Implement the Certificate Approach

- Using the sequence of the snapshot (s_1, …, s_T) we can rephrase the proof as follows:

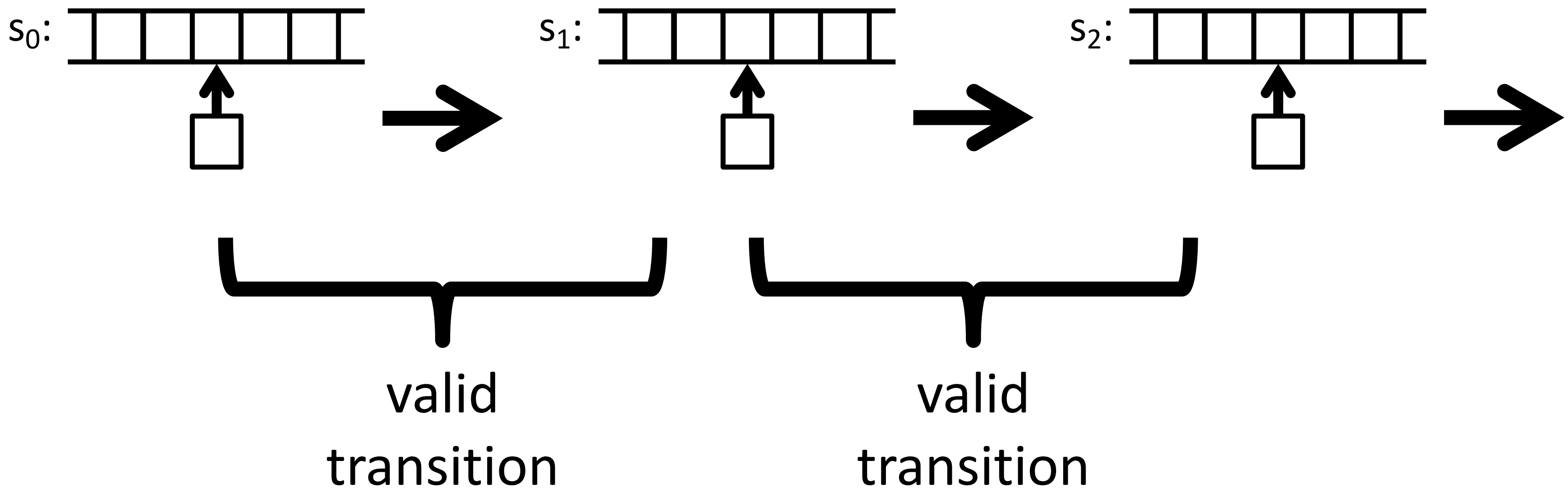
Prove knowledge of (s_1, …, s_T):
(1) $s_i \rightarrow s_{i+1}$ follows the transition function

- To enforce validity of transition, the KGC signs on all possible valid transition:

$$\theta[s,s'] \leftarrow \mathrm{Sign}(msk, (s,s'))$$

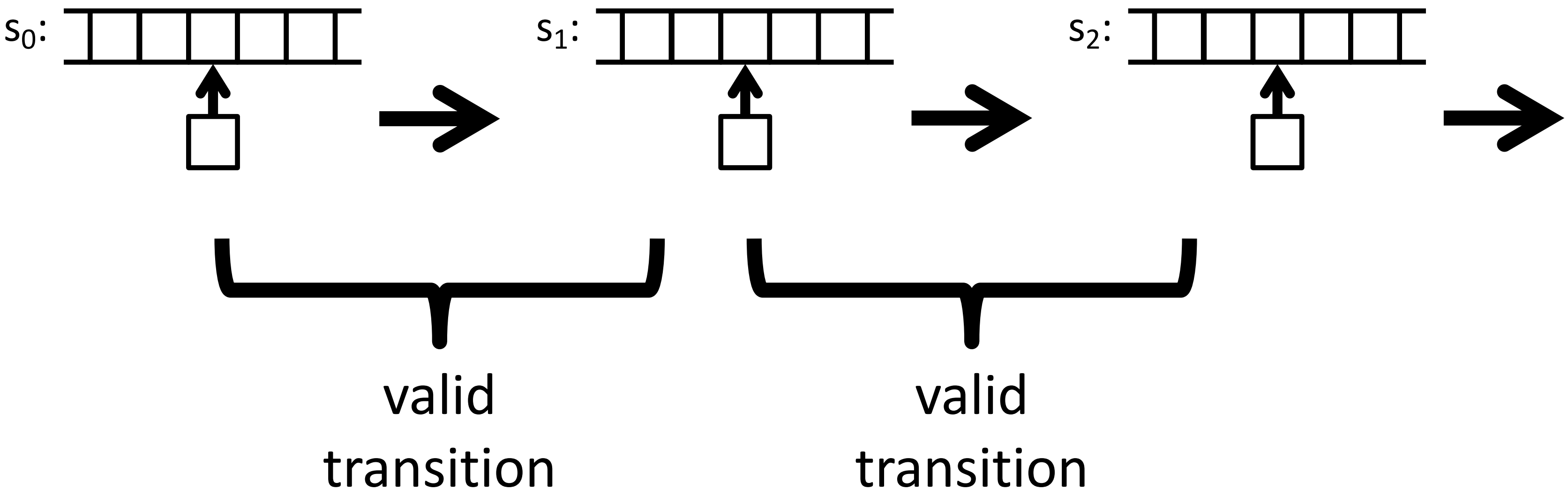
$$\forall s \rightarrow s': \text{valid transition}$$

Signing Every Possible Transition



Prove knowledge of (s_0, s_1, θ_1):
$\text{Verify}(vk, (s_0, s_1), \theta_1) = 1$

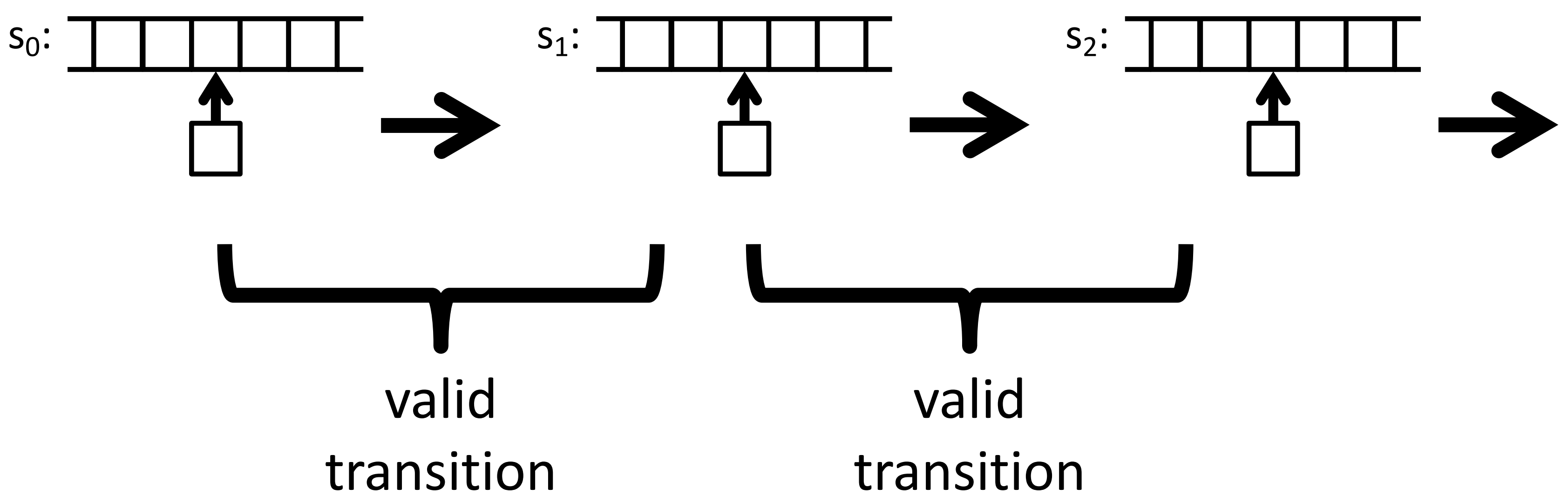
Signing Every Possible Transition



Prove knowle
Verify(vk,

Prove knowledge of (s_1, s_2, θ_2):
$\text{Verify}(vk, (s_1, s_2), \theta_2) = 1$

Signing Every Possible Transition



Prove knowledge of $(s_1, \ldots, s_T, \theta_1, \ldots, \theta_T)$:

(1) $\mathrm{Verify}((s_{i-1}, s_i), \theta_i) = 1$

Main Difficulty

Prove knowledge of $(s_1, \ldots, s_T, \theta_1, \ldots, \theta_T)$:

(1) $\mathrm{Verify}((s_{i-1}, s_i), \theta_i) = 1$

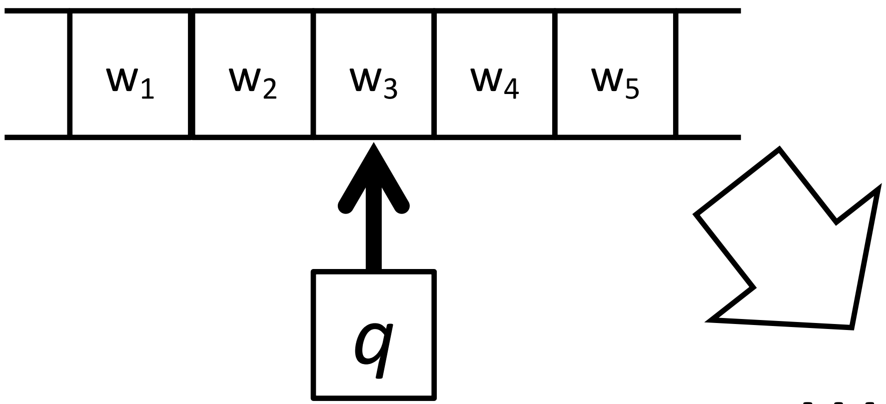
- Possible pairs of snapshots are infinitely many,
  - since snapshots have unbounded lengths
- We further decompose this condition

Agenda

- Attribute-Based Signatures
- Security Requirement
- Certificate Approach
- Idea 1: History of Computation
- Idea 2: Locality of Rewriting
- Overview of the Scheme
- Conclusion

Configuration

- A snapshot is encoded into a single string, ***configuration***



$$\ldots\ w_1\ w_2\ q\ w_3\ w_4\ w_5\ \ldots$$

- Consists of (1) the content of the tape interleaved with (2) the state symbol q
  - the position of q encodes the position of the head

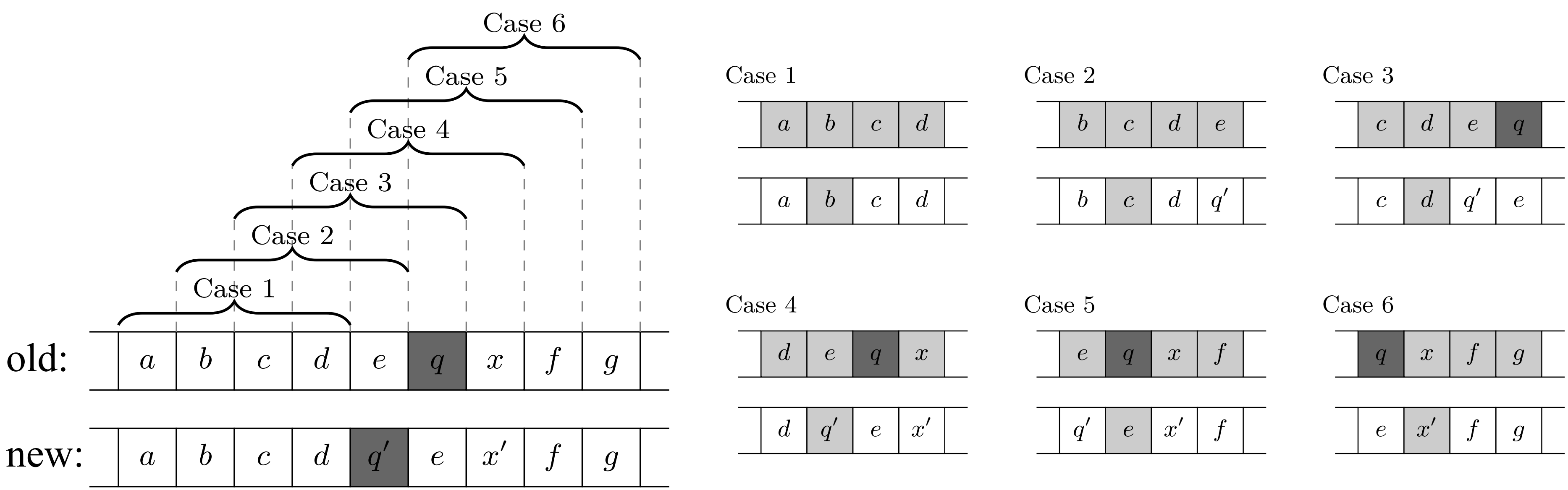
Locality of Rewriting

step t:	w_1	w_2	q	w_3	w_4	w_5
step t+1:	w_1	q'	w_2	w'_3	w_4	w_5

- Each symbol in a new configuration is determined by ***neighbors in the old configuration***
- Four neighbors are sufficient for any case

The General Cases

- Each cell will be determined by the four neighbors in the old configuration

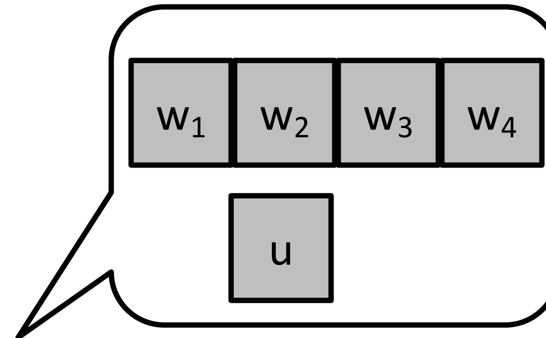


Enforcing Validity of Transition

- To enforce validity of transition
  KGC signs on every valid 5-tuple:

  $\theta[w_1, w_2, w_3, w_4, u]$

  $\leftarrow$ Sign(msk, (w_1, w_2, w_3, w_4, u))



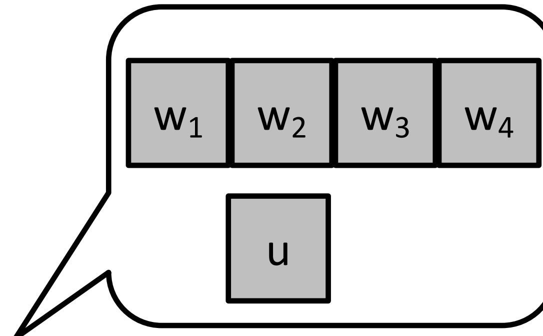
- The signer proves the knowledge of signature for ***<u>every</u>*** symbol in the new configuration

old:	w_1	w_2	q	w_3	w_4	w_5
new:	w_1	q'	w_2	w'_3	w_4	w_5

Enforcing Validity of Transition

- To enforce validity of transition
  KGC signs on every valid 5-tuple:

  $\theta[w_1, w_2, w_3, w_4, u]$



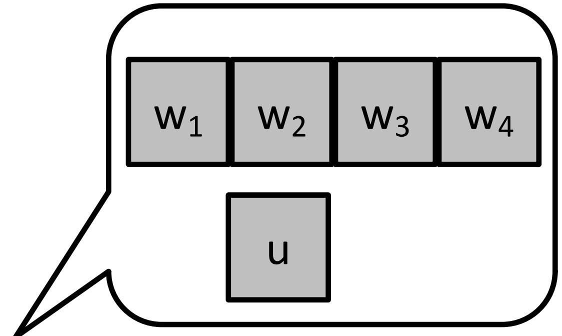
Prove knowledge of $(w_1, w_2, q, w_3, q', \theta_1)$:
$\mathrm{Verify}(vk, (w_1, w_2, q, w_3, q'), \theta_1) = 1$

old:	w_1	w_2	q	w_3	w_4	w_5
new:	w_1	q'	w_2	w'_3	w_4	w_5

Enforcing Validity of Transition

- To enforce validity of transition
  KGC signs on every valid 5-tuple:

  $\theta[w_1, w_2, w_3, w_4, u]$



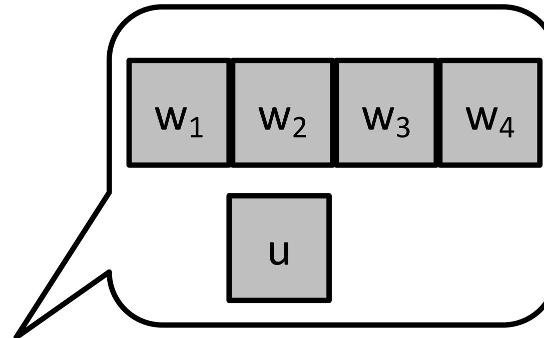
Prove knowledge of $(w_2, q, w_3, w_4, w_2, \theta_2)$:
Verify(vk, $(w_2, q, w_3, w_4, w_2), \theta_2) = 1$

old:	w_1	w_2	q	w_3	w_4	w_5
new:	w_1	q'	w_2	w'_3	w_4	w_5

Enforcing Validity of Transition

- To enforce validity of transition KGC signs on every valid 5-tuple:

  $\theta[w_1, w_2, w_3, w_4, u]$



Prove knowledge of $(q, w_3, w_4, w_5, w'_3, \theta_3)$:
$\text{Verify}(vk, (q, w_3, w_4, w_5, w'_3), \theta_3) = 1$

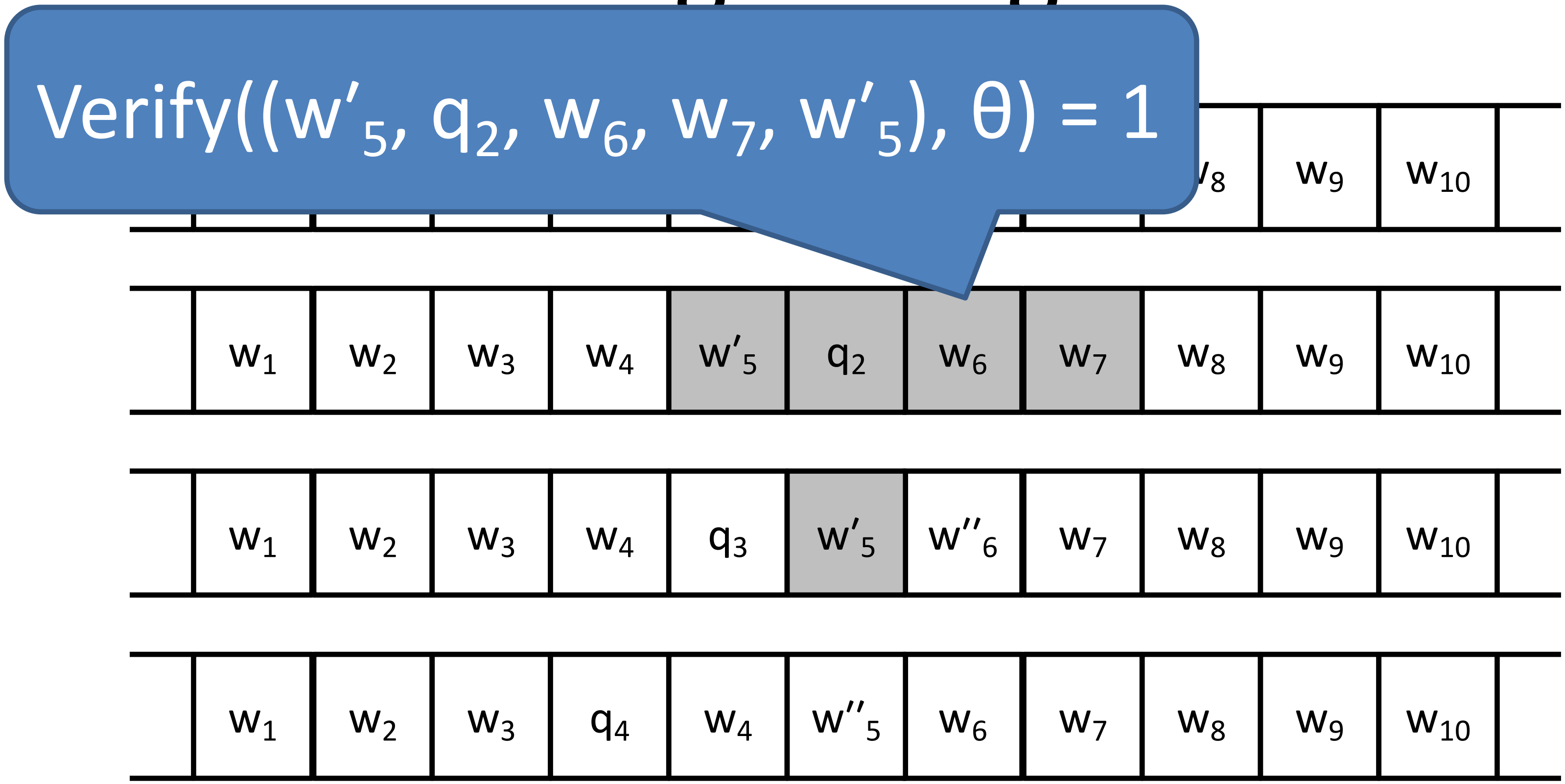
- The s… for ***every*** sym… new configuration

old:	w_1	w_2	q	w_3	w_4	w_5
new:	w_1	q'	w_2	w'_3	w_4	w_5

Agenda

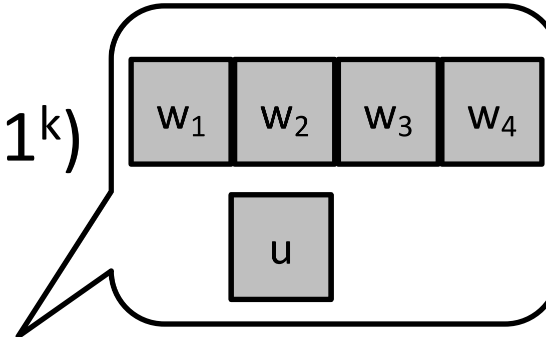
- Attribute-Based Signatures
- Security Requirement
- Certificate Approach
- Idea 1: History of Computation
- Idea 2: Locality of Rewriting
- Overview of the Scheme
- Conclusion

Putting All Together



- Proves the knowledge of signatures on the neighbors (quadratic in running time of TM)
- Every symbol is hidden as a witness

The Scheme



- Setup:
  - $crs \leftarrow CRSGen(1^k)$, $(vk, sk) \leftarrow SigKg(1^k)$
- KeyGen:
  - for every valid 5-tuple (w_1, w_2, w_3, w_4, u):
    - $\theta_{[w1,\ w2,\ w3,\ w4,\ u]} \leftarrow SigSign(sk, (w_1, w_2, w_3, w_4, u))$
- Sign: $\{w_{i,j}\}_{i,j}$: 2D arrangement of configurations
  - $\pi_{i,j} \leftarrow Prove(crs, (w_{i-1,j-2}, w_{i-1,j-1,}, w_{i-1,j}, w_{i+1,j}, w_{i,j}, \theta))$
- Verify: for all (i,j) verify $\pi_{i,j}$

Main Theorem

Theorem If the non-interactive proof system is witness-indistinguishable and extractable, the signature scheme is unforgeable, the proposed scheme is anonymous and unforgeable

Instantiate this with
GS proofs in SXDH setting and
structure-preserving signatures

Theorem If SXDH assumption holds, the proposed scheme satisfies anonymity and unforgeability

Efficiency

Signing key length	Signature length	Verification time
$O(\|\Gamma\|^4)$	$O(T^2)$	$O(T^2)$

$|\Gamma|$: The size of the tape alphabet
T: The running time of the TM

- The scheme is reasonably efficient!

Agenda

- Attribute-Based Signatures
- Security Requirement
- Certificate Approach
- Idea 1: History of Computation
- Idea 2: Locality of Rewriting
- Overview of the Scheme
- Conclusion

Summary

- Proposed attribute-based signature scheme for ***unbounded languages (Turing machines)***
  - Uniform model of computation as the policy
  - No bound on the sizes of both TMs and attributes
  - Can be instantiated from the SXDH assumption in bilinear groups