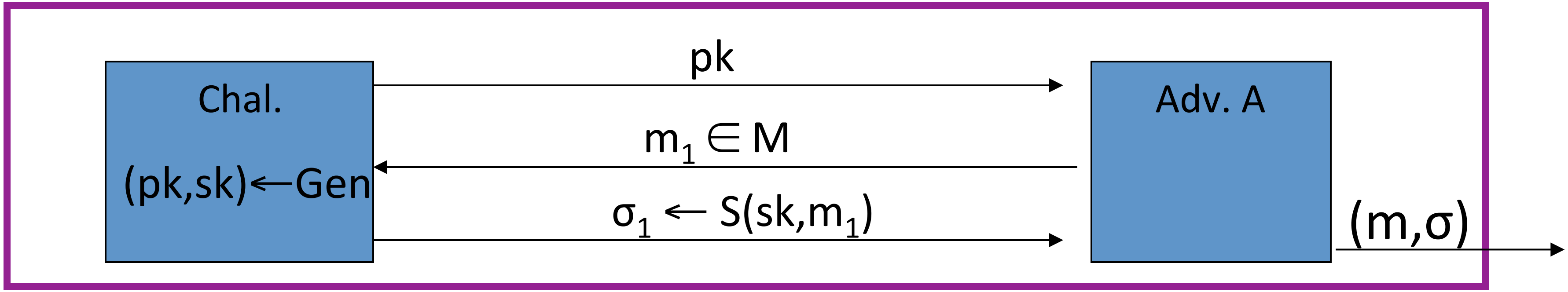
Sigs. with special properties

Fast one-time signatures and applications

One-time signatures: definition

Suppose signing key is used to sign a single message

Can we give a simple (fast) construction SS=(Gen,S,V) ?



A wins if $V(pk,m,\sigma)$ = `accept' and $m \neq m_1$

Security: for all "efficient" A, $\text{Adv}_{\text{1-SIG}}[A,SS]$ = Pr[A wins] ≤ negl

Application: fast online signatures

1. Next section: secure one-time sigs ⇒ secure many-time sigs

2. Fast online signatures: signing can be slow on a weak device

   Goal:

   - Do heavy signature computation **<u>before</u>** message is known
   - Quickly output signature once user supplies message

Fast online signing using one-time sigs

(Gen, S, V): secure many-time signature (slow)

$(\text{Gen}_{1T}, S_{1T}, V_{1T})$: secure one-time signature (fast)

slow

- $\text{Gen} \longrightarrow (pk, sk)$
- PreSign(sk): $(pk_{1T}, sk_{1T}) \longleftarrow \text{Gen}_{1T}$, $\sigma \longleftarrow S(sk, pk_{1T})$
- $S_{\text{online}}\big((\sigma, sk_{1T}, pk_{1T}) , m \big)$: $\sigma_{1T} \longleftarrow S_{1T}(sk_{1T}, m)$ fast

  output $\boldsymbol{\sigma^* \longleftarrow (pk_{1T}, \sigma, \sigma_{1T})}$
- $V_{\text{online}}\big(pk, m, \boldsymbol{\sigma^* = (pk_{1T}, \sigma, \sigma_{1T})} \big)$:

  accept if $V(pk, pk_{1T}, \sigma) = V_{1T}(pk_{1T}, m, \sigma_{1T}) =$ "accept"

Sigs. with special properties

Constructing fast one-time signatures

One-time signatures

Goal: one-time sigs from fast **one-way functions** (OWF)

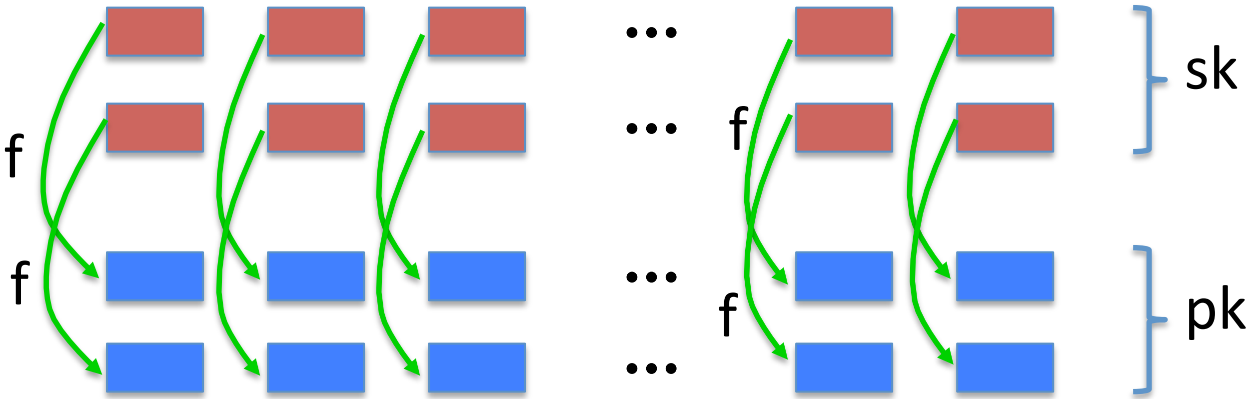
- $f: X \longrightarrow Y$ is a OWF if (1) $f(x)$ is efficiently computable,
  (2) hard to invert on random $f(x)$

- Examples: (1) $f(x) = \text{AES}(\underset{\substack{\uparrow \\ \text{key}}}{x}, 0^{128})$, (2) $f(x) = \text{SHA256}(x)$

Lamport one-time signatures (simple)

f: X ⟶ Y a one-way function. Msg space: $M = \{0,1\}^{256}$

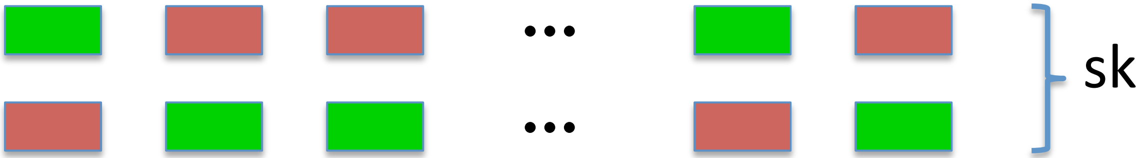
Gen: generate 2×256 random elements in X



Lamport one-time signatures (simple)

f: X ⟶ Y a one-way function. Msg space: $M = \{0,1\}^{256}$

Gen: generate 2×256 random elements in X



m = 0 1 1 … 0 1

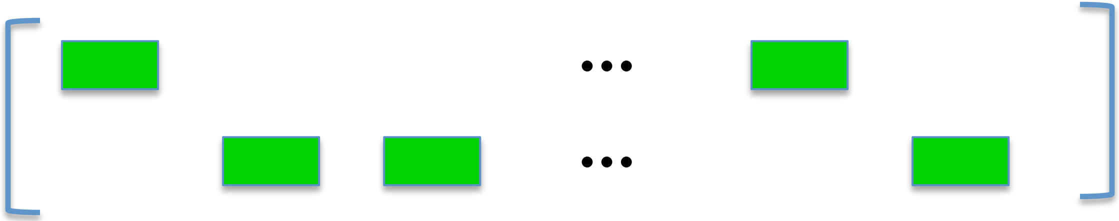
S(sk, m): σ = (pre-images corresponding to bits of m)

Lamport one-time signatures (simple)

$f: X \longrightarrow Y$ a one-way function. Msg space: $M = \{0,1\}^{256}$

Gen: generate 2×256 random elements in X

$\sigma =$



$\in X^{256}$ (2KB)

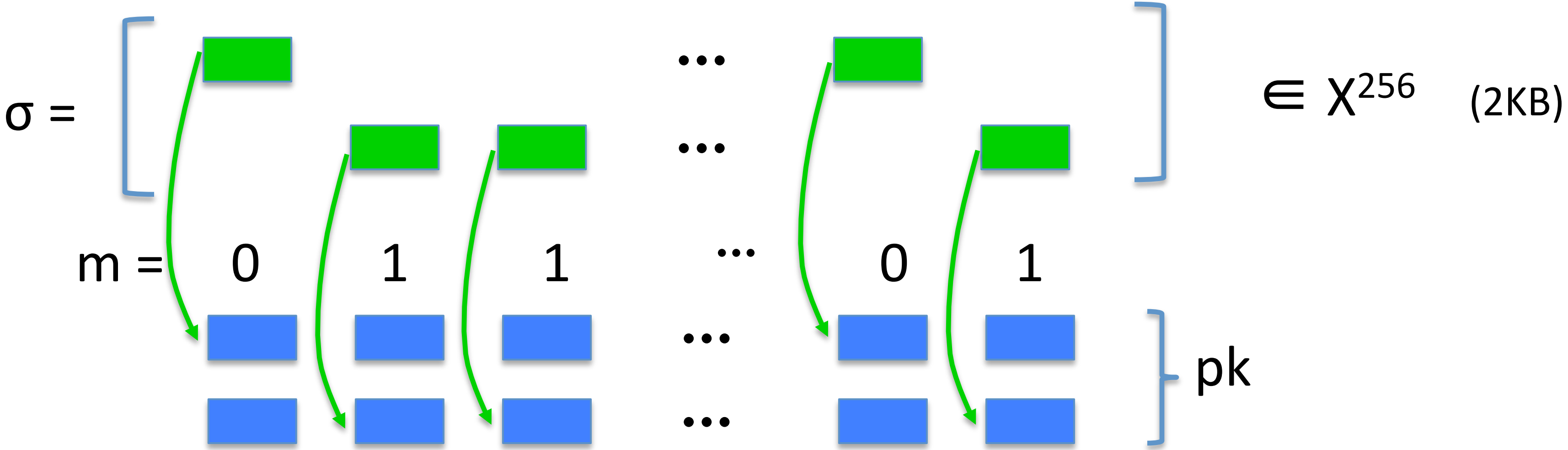
m = 0 1 1 … 0 1

S(sk, m): σ = (pre-images corresponding to bits of m)

Lamport one-time signatures (simple)

$f: X \longrightarrow Y$ a one-way function. Msg space: $M = \{0,1\}^{256}$

Gen: generate 2×256 random elements in X



$\sigma =$ $\in X^{256}$ (2KB)

$m =$ 0 1 1 … 0 1

pk

V(pk, m, σ): accept if all pre-images in σ match values in pk

Very fast signature system. Will prove one-time security in a bit.

Not two-time secure:

The attacker can ask for a signature on 0^{128} and on 1^{128}.
He gets all of **sk** which he can use to sign new messages.

Abstraction: cover free set systems

Sets: $S_1, \; S_2, \; \ldots, \; S_{2^{256}} \;\subseteq\; \{1, \ldots, n\}$

<u>Def</u>: $\boldsymbol{S} = \{S_1, \; S_2, \; \ldots, \; S_{2^{256}}\}$ is **cover-free** if $S_i \not\subseteq S_j$ for all $i \neq j$

Example: if all sets in ***S*** have the same size k then ***S*** is cover free

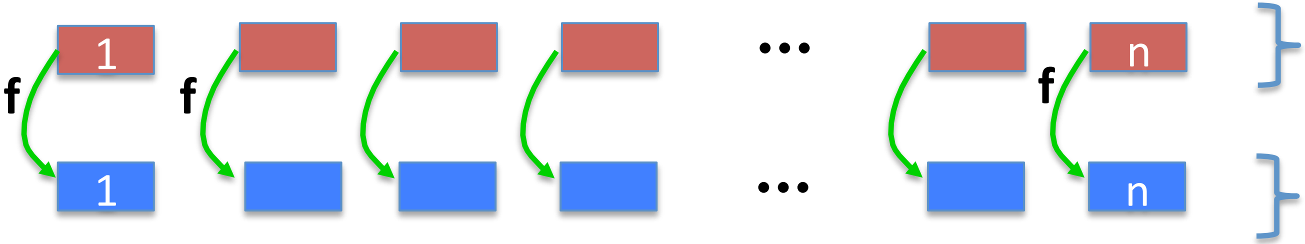
Abstract Lamport signatures

$f: X \longrightarrow Y$ a one-way function. Msg space: $M = \{0,1\}^{256}$

$\boldsymbol{S} = \{S_1, \; S_2, \; \ldots, \; S_{2^{256}}\}$ is **cover-free** over $\{1,..,n\}$

$H: \{0,1\}^{256} \longrightarrow \boldsymbol{S}$ a bijection (one-to-one)

Gen: generate n random elements in X



$sk \in X^n$

$pk \in Y^n$

Abstract Lamport signatures

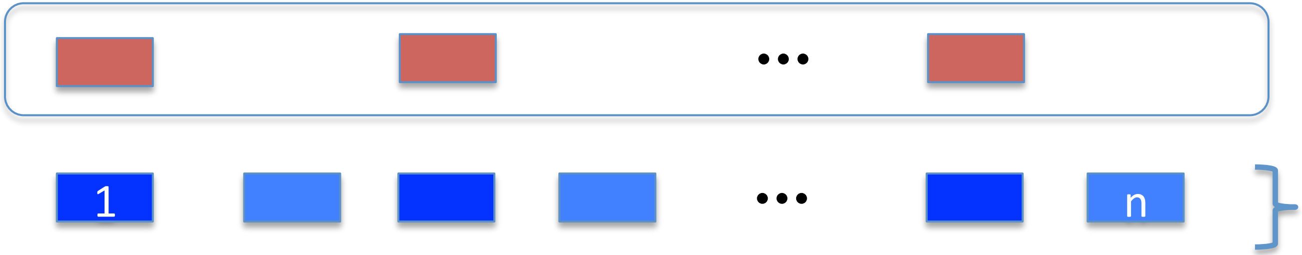
$f: X \longrightarrow Y$ a one-way function. Msg space: $M = \{0,1\}^{256}$

$\boldsymbol{S} = \{S_1, \ S_2, \ \ldots, \ S_{2^{256}}\}$ is **cover-free** over $\{1,..,n\}$

$H: \{0,1\}^{256} \longrightarrow \boldsymbol{S}$ a bijection (one-to-one)

Gen: generate n random elements in X

$\sigma =$



$pk \in Y^n$

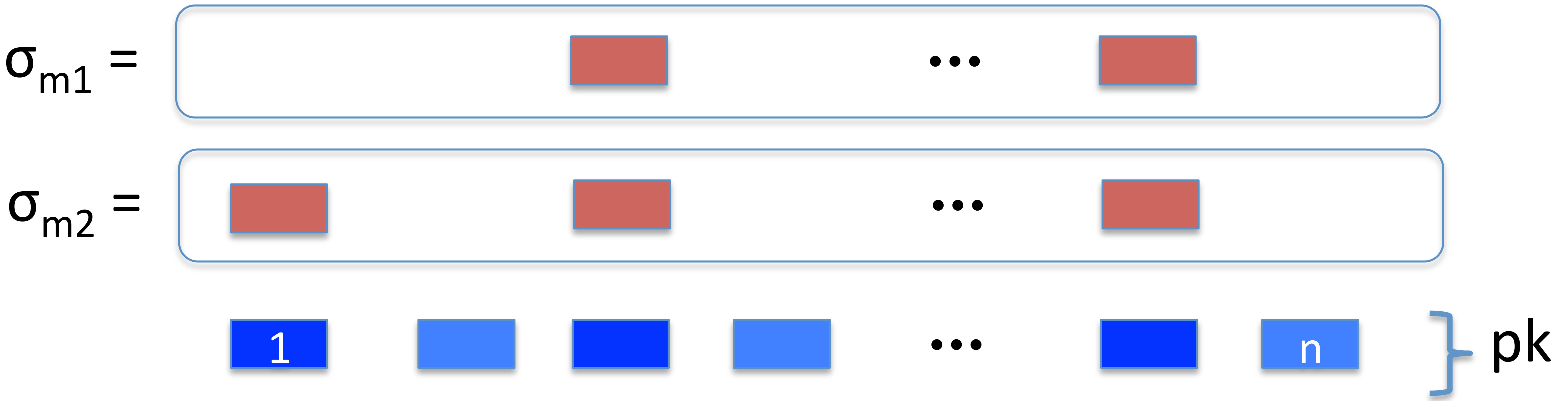
S(sk, m): $\sigma = \big($ pre-images corresponding to elements of H(m) $\big)$

Why cover free?

Suppose ***S*** were not cover free

⇒ exists m_1, m_2 such that $H(m_1) \subset H(m_2)$

⇒ signature on m_2 gives signature on m_1



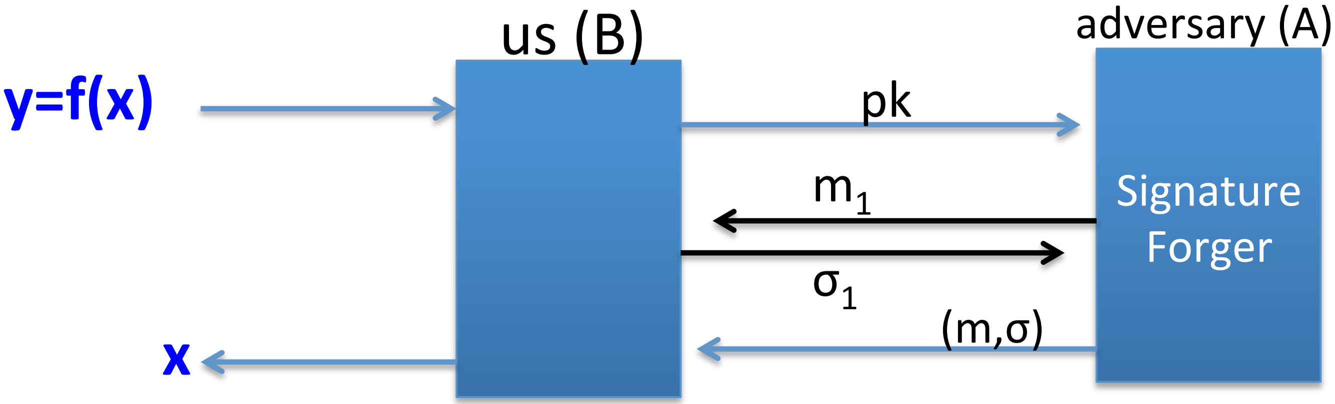
S(sk, m): $\sigma = \Big($ pre-images corresponding to elements of H(m) $\Big)$

Security statement

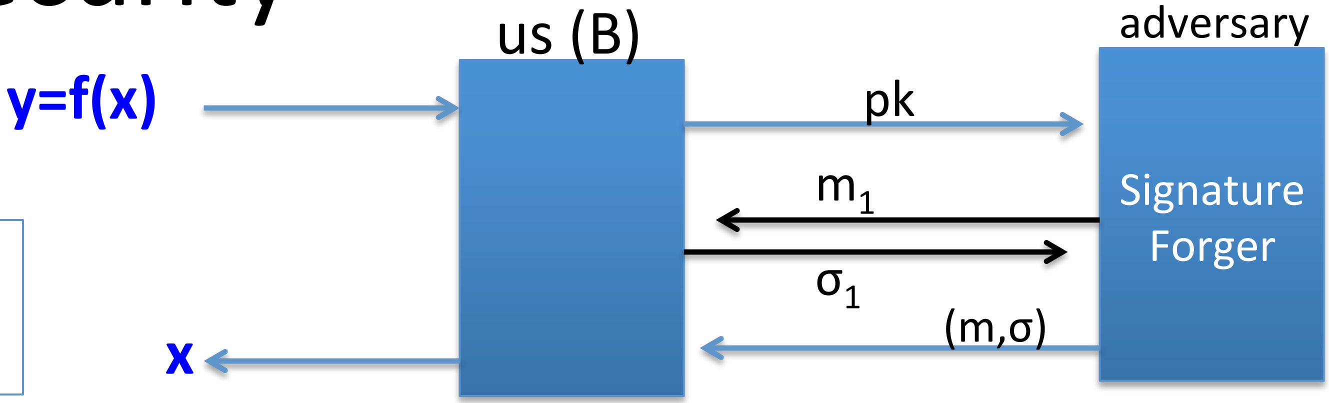
Thm: if f: X ⟶ Y is one-way and $\boldsymbol{S}$ is cover-free
then Lamport signatures (Lam) are one-time secure.

$$\forall A \ \exists B: \quad \mathrm{Adv}_{\text{1-SIG}}[A,\mathrm{Lam}] \le n \cdot \mathrm{Adv}_{\text{OWF}}[B,f]$$

Proving security:



Proving security



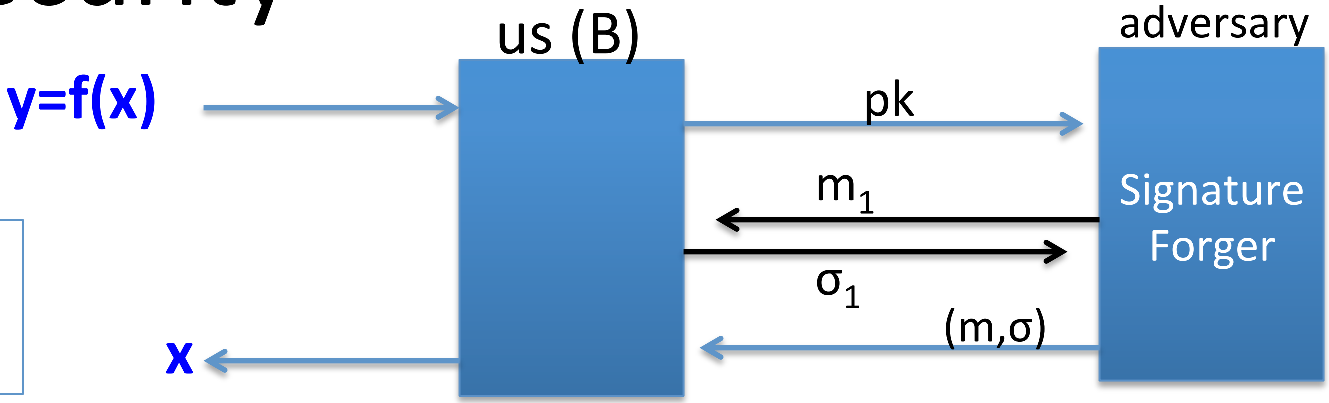
choose: $i \leftarrow \{1,\ldots,n\}$
$x_1,\ldots,x_n \leftarrow X$

pk= $\boxed{f(x_1)}$ … $\boxed{f(x_{i-1})}$ $\boxed{y}$ $\boxed{f(x_{i+1})}$ … $\boxed{f(x_n)}$

$$\begin{cases} i \notin H(m_1) \implies \text{we (alg. B) can generate } \sigma_1 \\ i \in H(m) \implies \sigma \text{ from adv. reveals pre-image } x \end{cases}$$

$\implies$ B wins if $i \in H(m)$ but $i \notin H(m_1)$

Proving security



choose: $i \leftarrow \{1,\ldots,n\}$

$x_1,\ldots,x_n \leftarrow X$

pk= $f(x_1)$ ••• $f(x_{i-1})$ y $f(x_{i+1})$ ••• $f(x_n)$

S cover free $\Longrightarrow \exists\, i^*$ s.t. $i^* \notin H(m_1)$ but $i^* \in H(m)$

$\boxed{\Pr[i=i^*] \geq \frac{1}{n}}$.

So: $\mathrm{Adv}_{\mathrm{OWF}}[B,f] = \Pr[i=i^*] \cdot \mathrm{Adv}_{1\text{-sig}}[A, \mathrm{Lamport}]$

$\geq \frac{1}{n} \cdot \mathrm{Adv}_{1\text{-sig}}[A, \mathrm{Lamport}]$ ■

Parameters (f: $X \longrightarrow Y$ where $X = Y$)

$\boldsymbol{S} = \{S_1, S_2, \ldots, S_{2^{256}}\}$ is **cover-free** over $\{1,..,n\}$

In particular: $\boldsymbol{S} = \big($ all subsets of $\{1,\ldots,n\}$ of size k $\big)$

$pk \in Y^n \quad \Rightarrow$ pk size = (n elements of Y)

sig. size = (k elements of X)

Msg-space $= \{0,1\}^{256} \quad \Rightarrow \quad |\boldsymbol{S}| = (n \text{ choose } k) \geq 2^{256}$

- To shrink signature size, choose small k

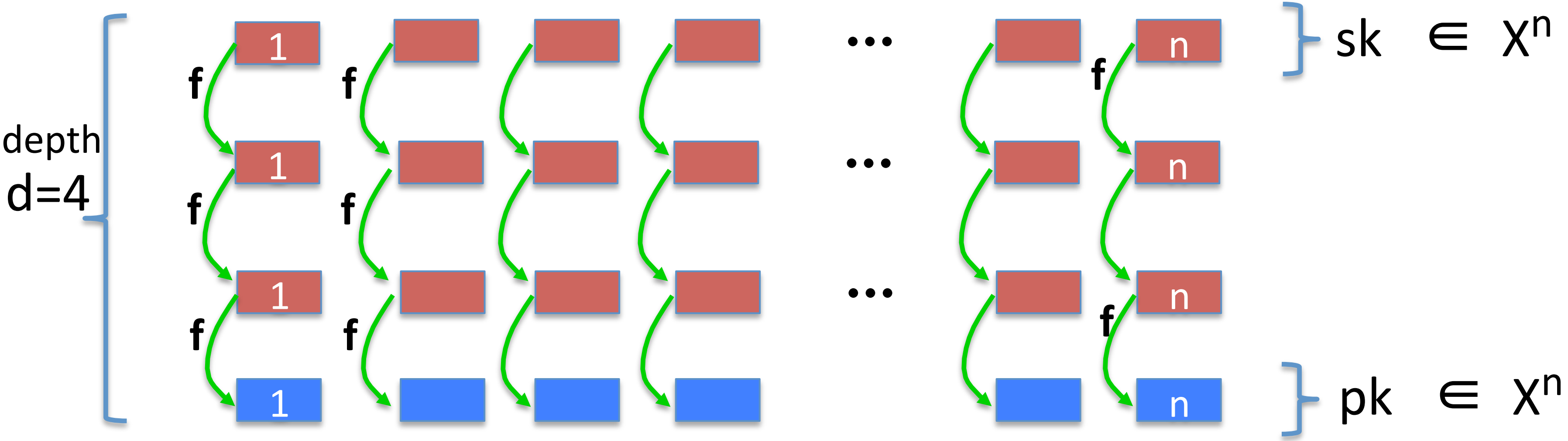
  example: $k=32 \quad \Rightarrow \quad n \geq 3290$

- For optimal (sig-size + pk-size) choose $n = 261, k = 123$

  (sig-size + pk-size) $\approx 1.5 \times 256$ elements of X (3KB)

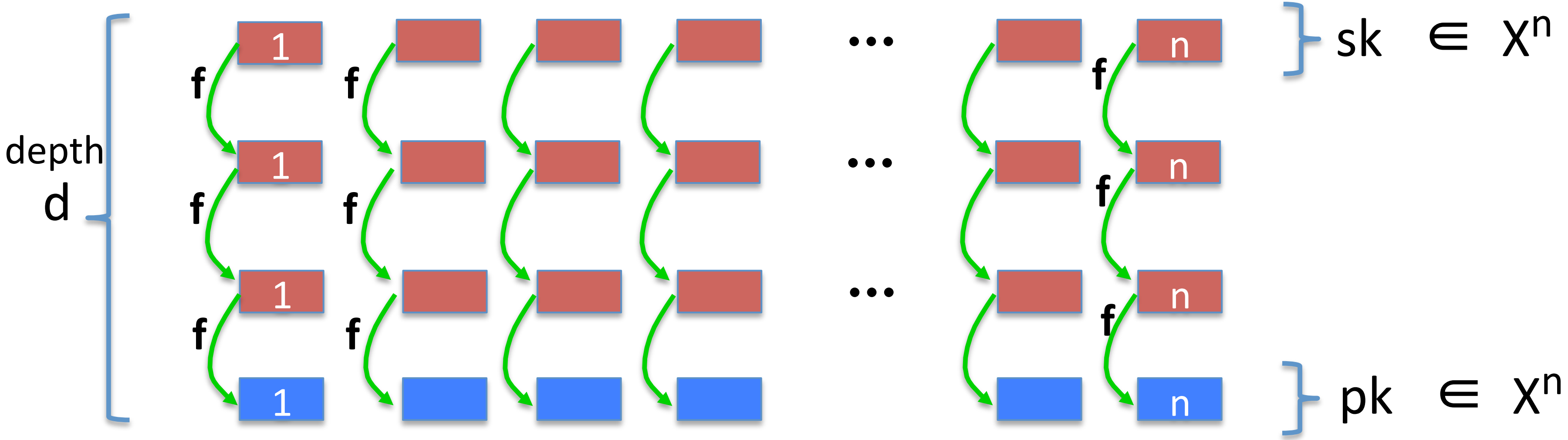
Further improvement: Winternitz

Gen: generate n random elements in X : (f: X ⟶ X)



Further improvement: Winternitz

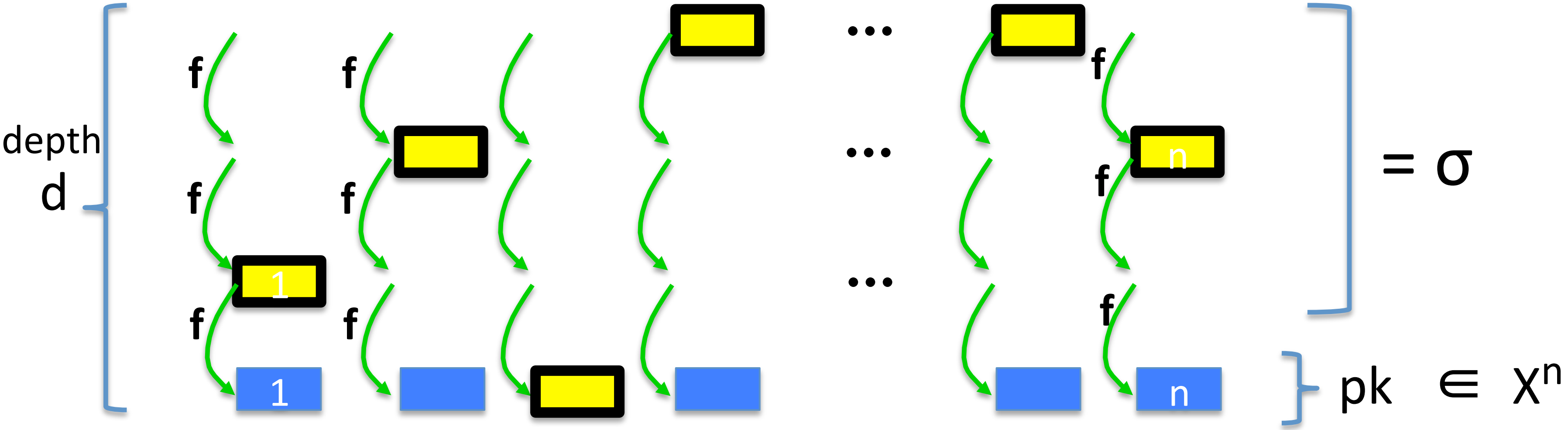
$H: \{0,1\}^{256} \longrightarrow \{0,1,\ldots,d\text{-}1\}^n$



S(sk, m): $\sigma = \Big($ pre-images indicated by H(m) $\Big)$

Further improvement: Winternitz

$H: \{0,1\}^{256} \longrightarrow \{0,1,\ldots,d\text{-}1\}^n$ ex: $H(0^{256}) = (2, 1, 3, 0, \ldots, 0, 1)$



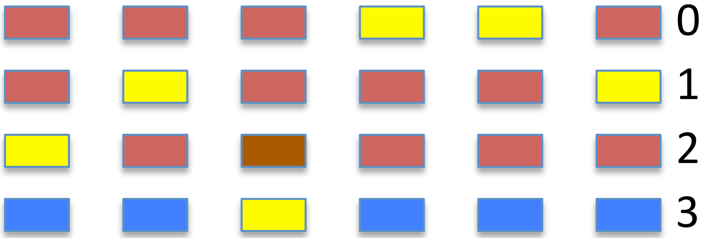
S(sk, m): σ = (pre-images indicated by H(m))

For what H is this a secure one-time signature?

Suppose $H(0^{256}) = (2, 1, 3, 0, 0, 1)$

$H(1^{256}) = (2, 2, 3, 1, 1, 2)$

Is the signature one-time secure?



○ No, from a sig. on 0^{256} one can construct a sig. on 1^{256}

○ No, from a sig. on 1^{256} one can construct a sig. on 0^{256}

○ Yes, the signature is one-time secure

○ It depends on how H behaves at other points

Optimized parameters

For one-time security need that:

for all $m_0 \neq m_1$ we have $H(m_0)$ does not “cover” $H(m_1)$

Parameters:

- Time(sign) = Time(verify) = $O(n \cdot d)$
- pk size = sig. size = (n elements in X)
- msg-space = $\{0,1\}^{256}$ $\Rightarrow$ $n > 256 / \log_2(d)$ (approx.)

(pk size)+(sig. size) $\approx 256 \times (2/\log_2(d))$ elems. of X

For Lamport: (pk size)+(sig. size) $\approx 256 \times (1.5)$ elems. of X

Sigs. with special properties

One-time signatures ⇒ many-time signatures

Review

One-time signatures need not be 2-time secure

example: Lamport signatures

Goal: convert any one-time signature into a many-time signature

Main tool: collision resistant hash functions

Construction

$(\text{Gen}_{1T}, S_{1T}, V_{1T})$: secure one-time signature (fast)

Four-time signature: (stateful version)

- **Gen**:

$$\text{Gen}_{1T} \longrightarrow \boxed{(\ pk_{0123},\ sk_{0123}\ )}$$

(pk_{01},sk_{01}) (pk_{23},sk_{23})

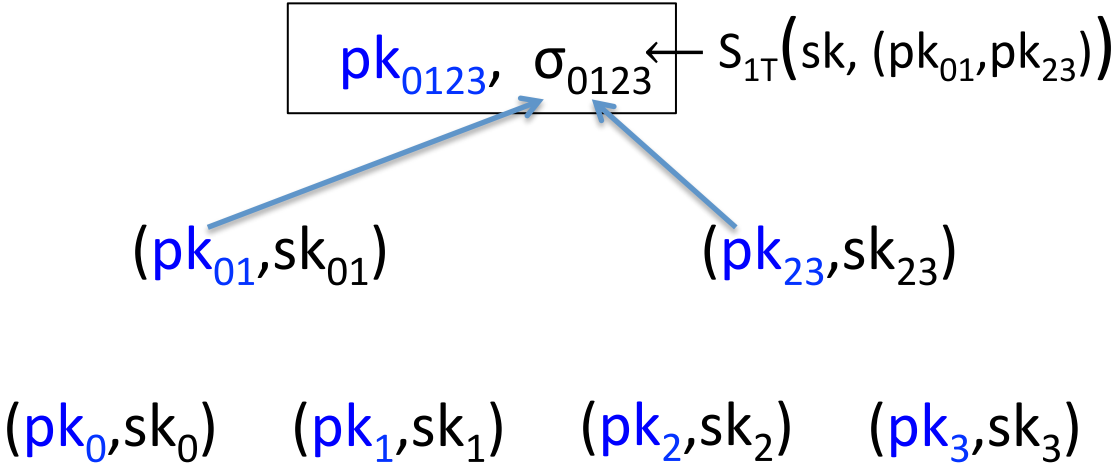
(pk_0,sk_0) (pk_1,sk_1) (pk_2,sk_2) (pk_3,sk_3)

Construction

$(Gen_{1T}, S_{1T}, V_{1T})$: secure one-time signature (fast)

Four-time signature: (stateful version)

- **Gen**:



$\boxed{pk_{0123},\ \sigma_{0123}} \leftarrow S_{1T}(sk, (pk_{01}, pk_{23}))$

(pk_{01}, sk_{01}) (pk_{23}, sk_{23})

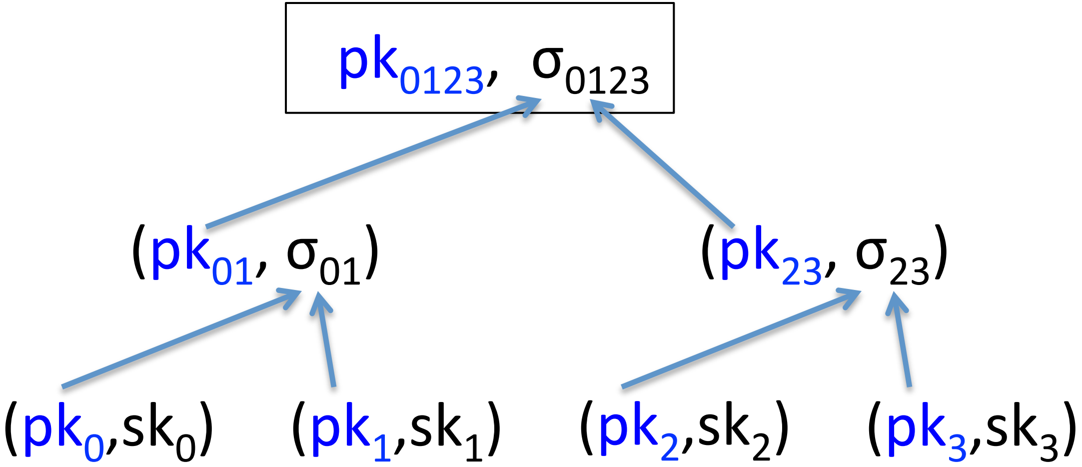
(pk_0, sk_0) (pk_1, sk_1) (pk_2, sk_2) (pk_3, sk_3)

Construction

$(\text{Gen}_{1T}, S_{1T}, V_{1T})$: secure one-time signature (fast)

Four-time signature: (stateful version)

- **Gen**:



pk_{0123}, σ_{0123}

(pk_{01}, σ_{01}) (pk_{23}, σ_{23})

(pk_0, sk_0) (pk_1, sk_1) (pk_2, sk_2) (pk_3, sk_3)

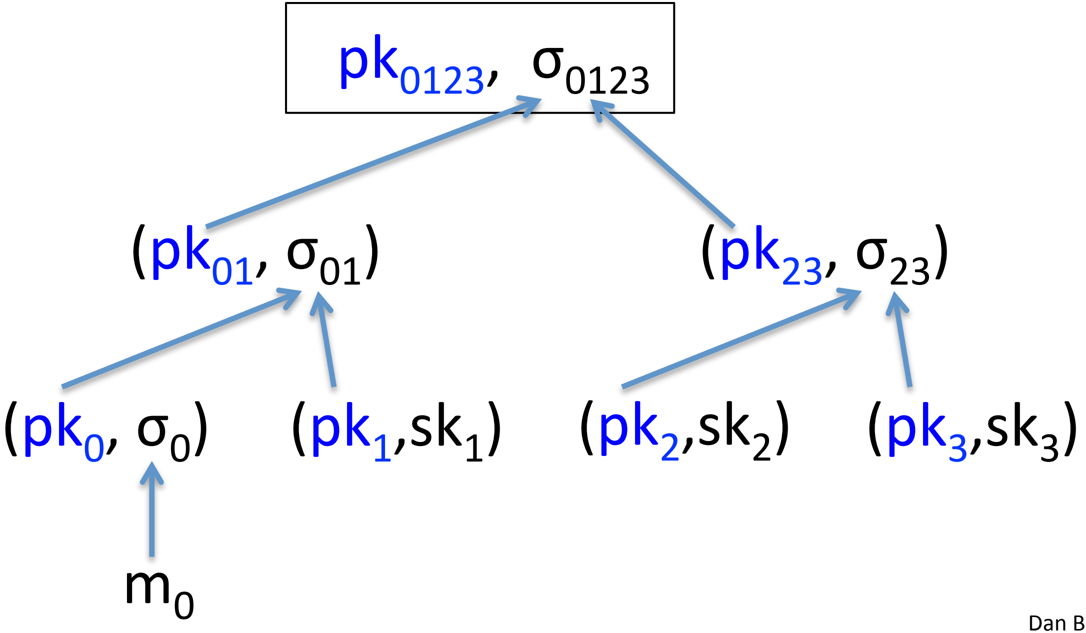
Construction

$(Gen_{1T}, S_{1T}, V_{1T})$: secure one-time signature (fast)

Four-time signature: (stateful version)

Sig. on msg m_0:

$(\sigma_{0123}, \sigma_{01}, \boldsymbol{\sigma_0},$
$pk_{01}, pk_{23}, pk_0, pk_1)$



pk_{0123}, σ_{0123}

(pk_{01}, σ_{01}) (pk_{23}, σ_{23})

(pk_0, σ_0) (pk_1, sk_1) (pk_2, sk_2) (pk_3, sk_3)

m_0

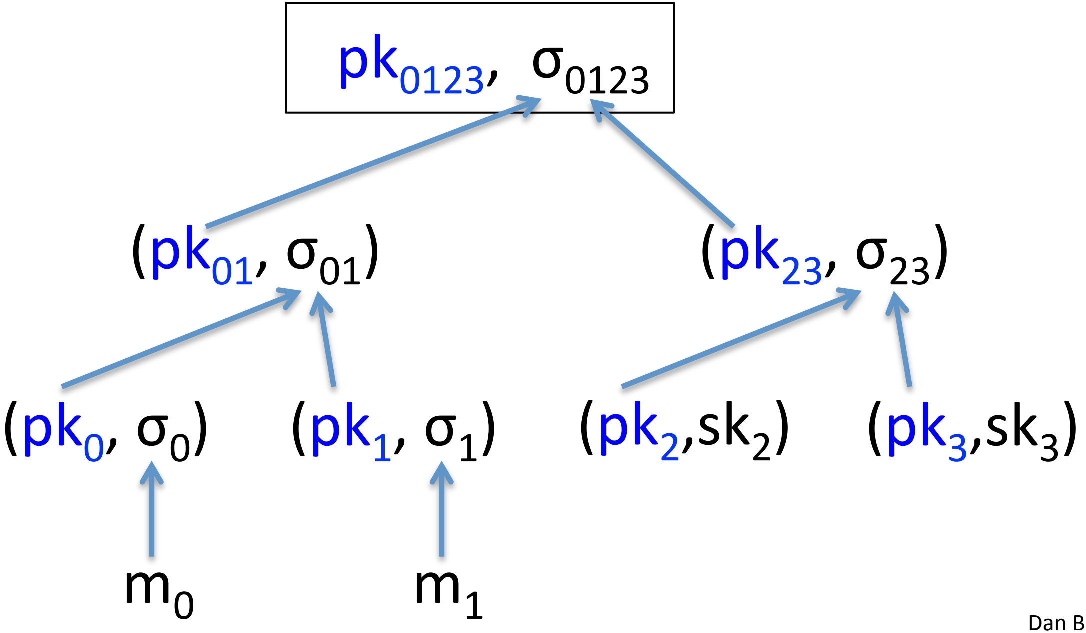
Construction

$(Gen_{1T}, S_{1T}, V_{1T})$: secure one-time signature (fast)

Four-time signature: (stateful version)

Sig. on msg m_1:

(σ_{0123}, σ_{01}, $\boldsymbol{\sigma_1}$,
pk_{01}, pk_{23}, pk_0, pk_1)



pk_{0123}, σ_{0123}

(pk_{01}, σ_{01}) (pk_{23}, σ_{23})

(pk_0, σ_0) (pk_1, σ_1) (pk_2, sk_2) (pk_3, sk_3)

m_0 m_1

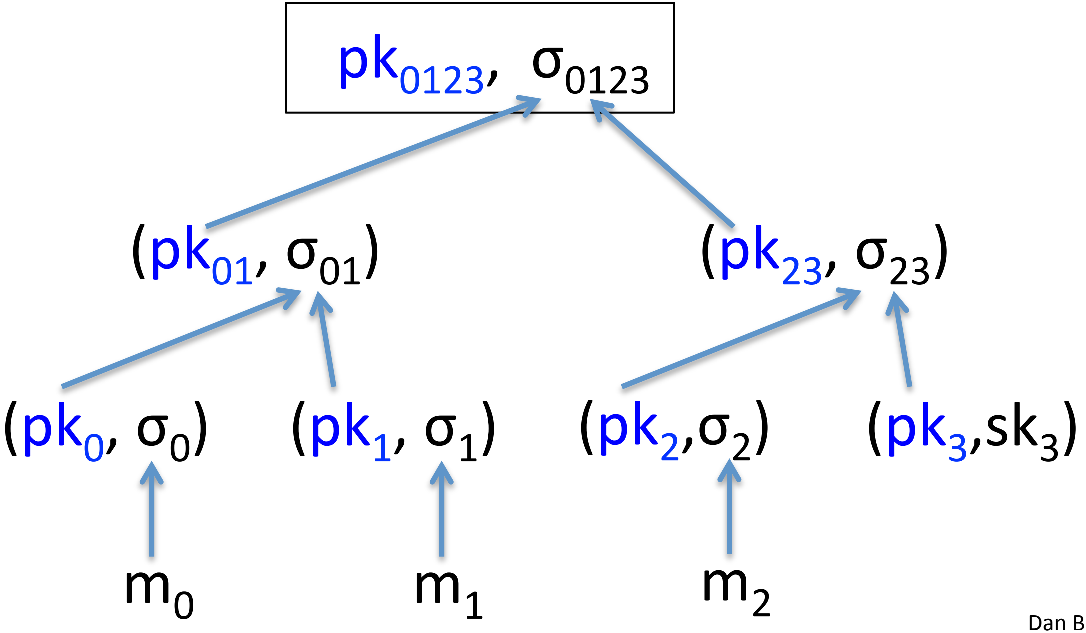
Construction

$(Gen_{1T}, S_{1T}, V_{1T})$: secure one-time signature (fast)

Four-time signature: (stateful version)

Sig. on msg m_2:

$(\ \sigma_{0123}, \sigma_{23}, \boldsymbol{\sigma_2},$
$pk_{01}, pk_{23}, \mathbf{pk_2}, \mathbf{pk_3}\ )$



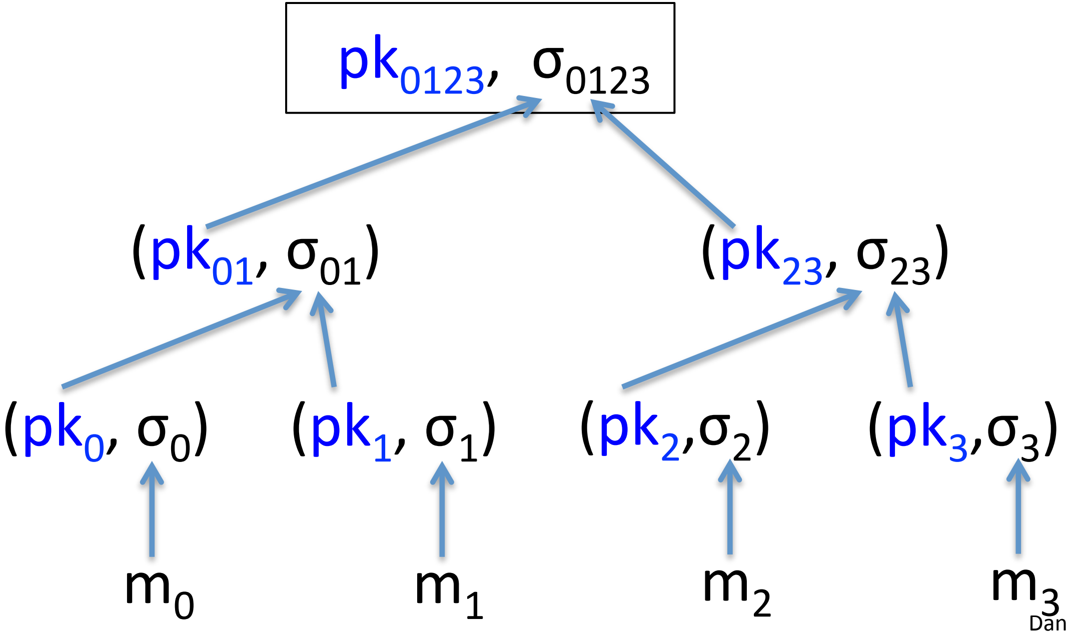
Construction

$(\text{Gen}_{1T}, S_{1T}, V_{1T})$: secure one-time signature (fast)

Four-time signature: (stateful version)

Sig. on msg m_3:

$(\ \sigma_{0123}, \sigma_{23}, \boldsymbol{\sigma_3},$
$\ pk_{01}, pk_{23}, \mathbf{pk_2}, \mathbf{pk_3}\ )$



More generally: 2^d-time signature

Tree of depth d:

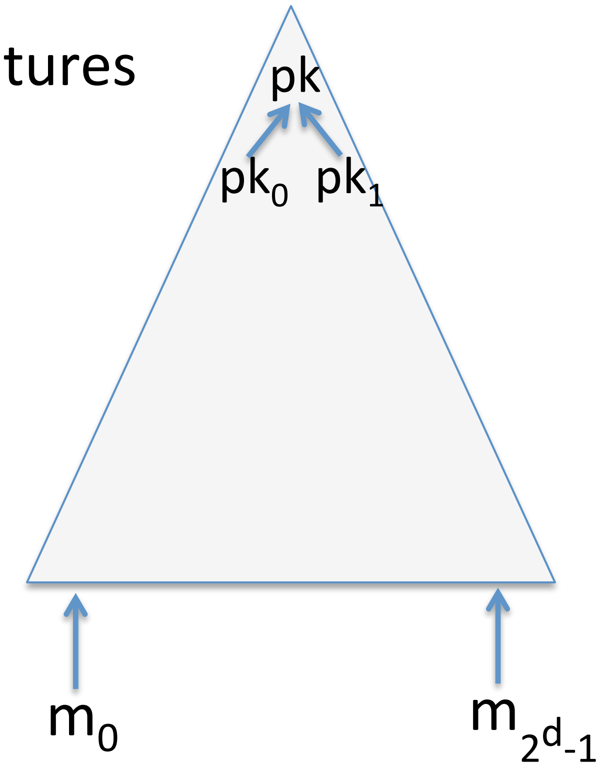
- Every signature contains d+1 one-time signatures along with associated pk's

Tree is generated on-the fly:

- Signer stores only d secret keys at a time

Stateful signature:

- Signer maintains a counter indicating which leaf to use for signature
- Every leaf must only be used once!



Optimized 2^d-time signatures

Combined with Lamport signatures:

- collision resistant hash funs ⇒ many-time signature

With further optimizations:

- For 2^{40} signatures: (stateful) signature size is ≈ 5KB

  … signing time is about the same as RSA signatures

- Recall: RSA sig size is 256 bytes (2048 bit RSA modulus)

THE END