Short Signatures from the RSA Assumption

Susan Hohenberger
JOHNS HOPKINS UNIVERSITY

Brent Waters
THE UNIVERSITY OF TEXAS AT AUSTIN™

Signatures Today

Schemes mostly fall into one of two classes:

Tree-Based Signatures

-- [GMR85, G86, M89, DN89, BM90, NY94, R90, CD95, CD96, ...]

-- tradeoff in size of signature and public key

"Hash-and-Sign" Signatures

-- [RSA78, E84, S91, O92, BR93, PS96, GHR99, CS00, CL01, BLS04, BB04, CL04, W05, GJKW07, GPV08, ...]

-- short signatures and short public keys

-- what practitioners expect

Focus on "Hash-and-Sign"

Again, most things fall into three classes:

Random Oracle Model

-- RSA [RSA78]
-- Discrete logarithm [E84,S91]
-- Lattices [GPV08]

Strong Assumptions

-- Strong RSA [GHR99, CS00]
-- q-Strong Diffie-Hellman [BB04]
-- LRSW [CL04]

Stateful

-- RSA, Computational Diffie-Hellman [HW09a]

Exception?
Waters '05 sigs from CDH.
They are short, but PK needs O(k) elements for sec. parameter k.

Our Main Result [HW09b]

<u>Immediate</u>: a digital signature scheme:

-- under the RSA assumption

-- standard model

-- stateless

-- short signatures (1 element, 1 integer)

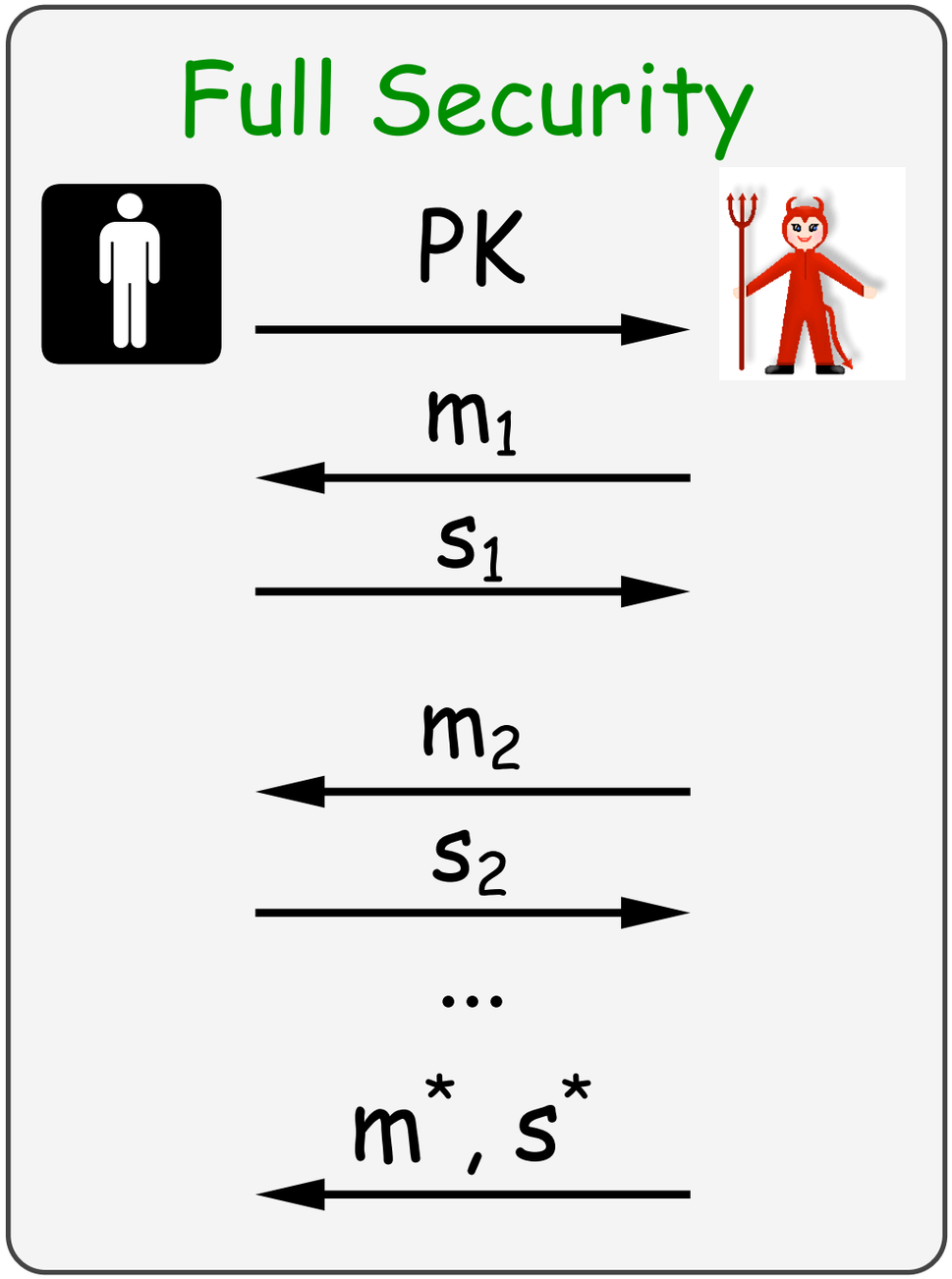
-- short public keys (modulus, 1 element, hash parameters)

<u>Longer-term</u>: a technique for:

-- designing short, standard model signatures

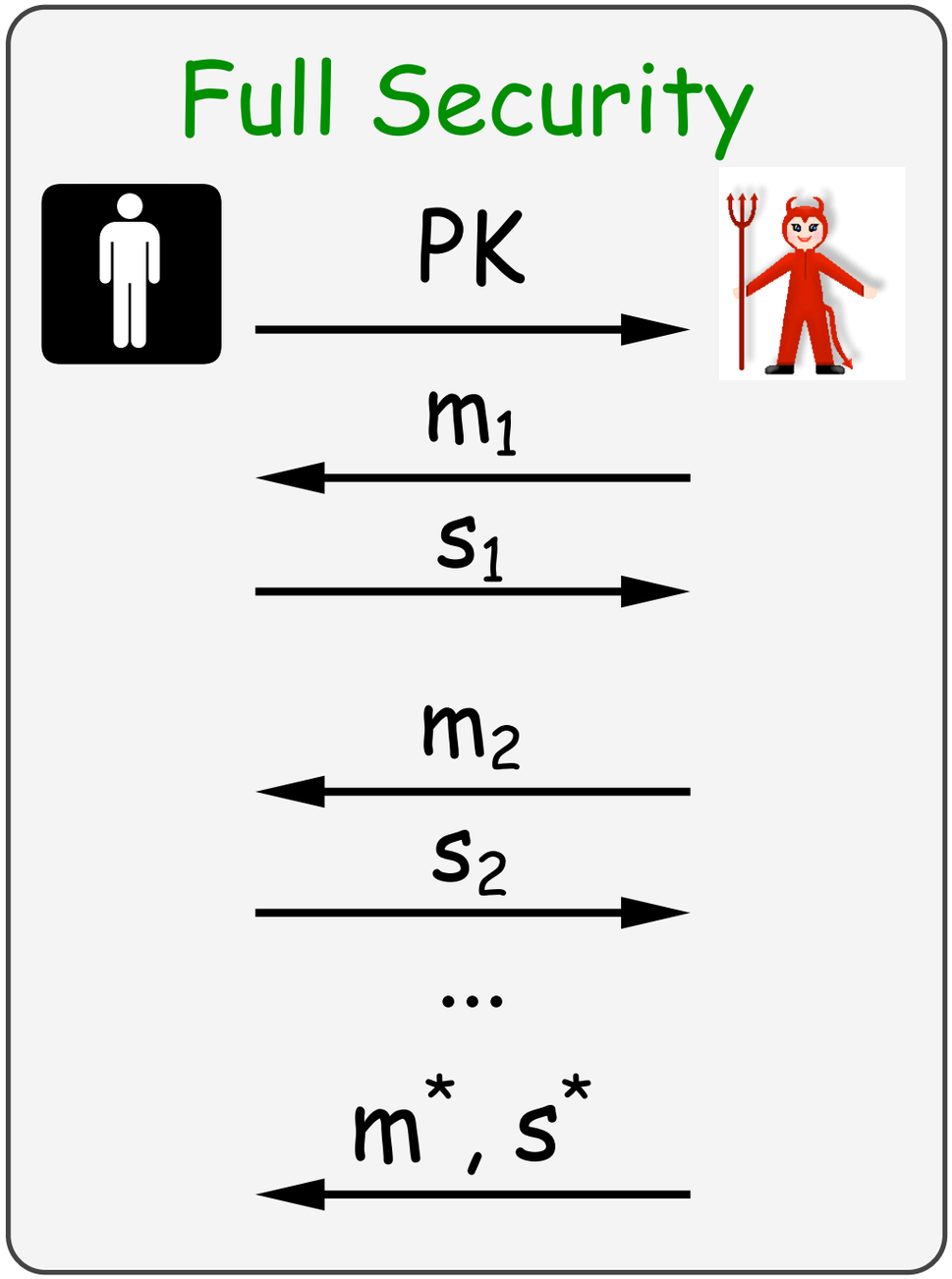
-- non-generic path from selective to full security

Goldwasser-Micali-Rivest Definition



Negligible probability that Verify(PK, m^*, s^*)=1 and m^* is new.

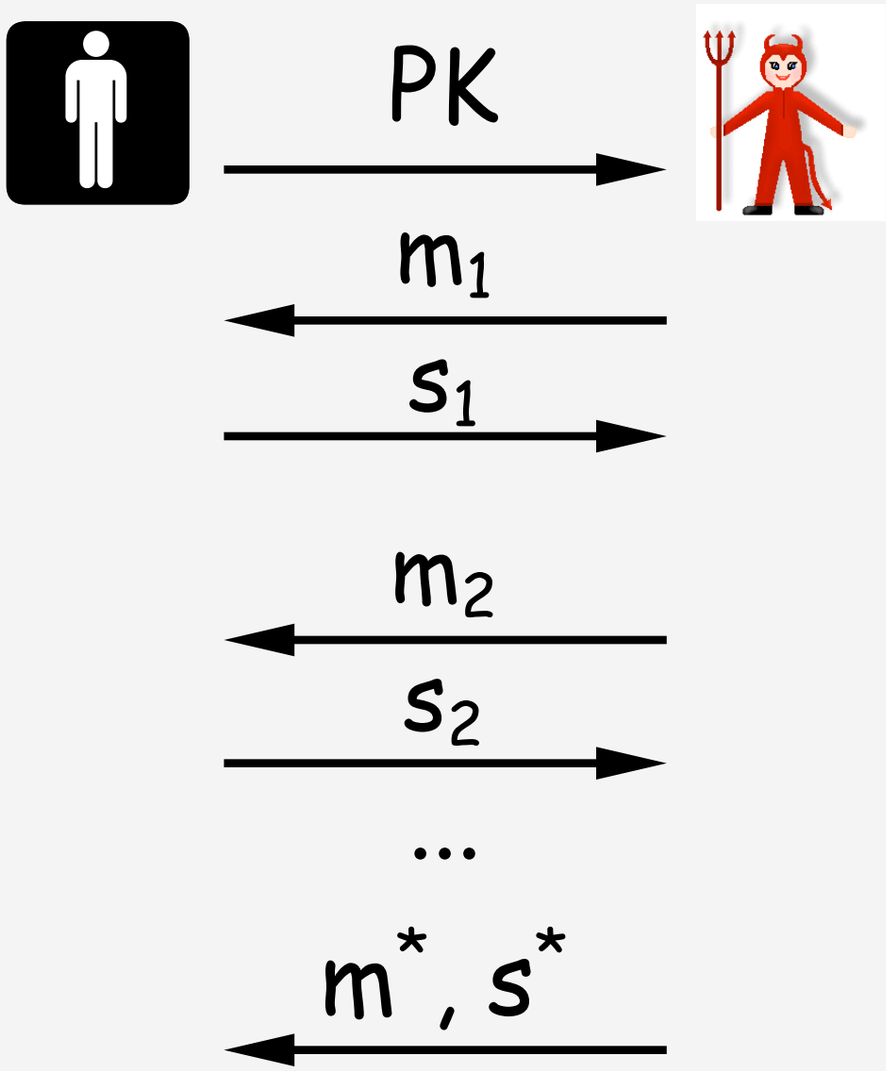
Goldwasser-Micali-Rivest Definition



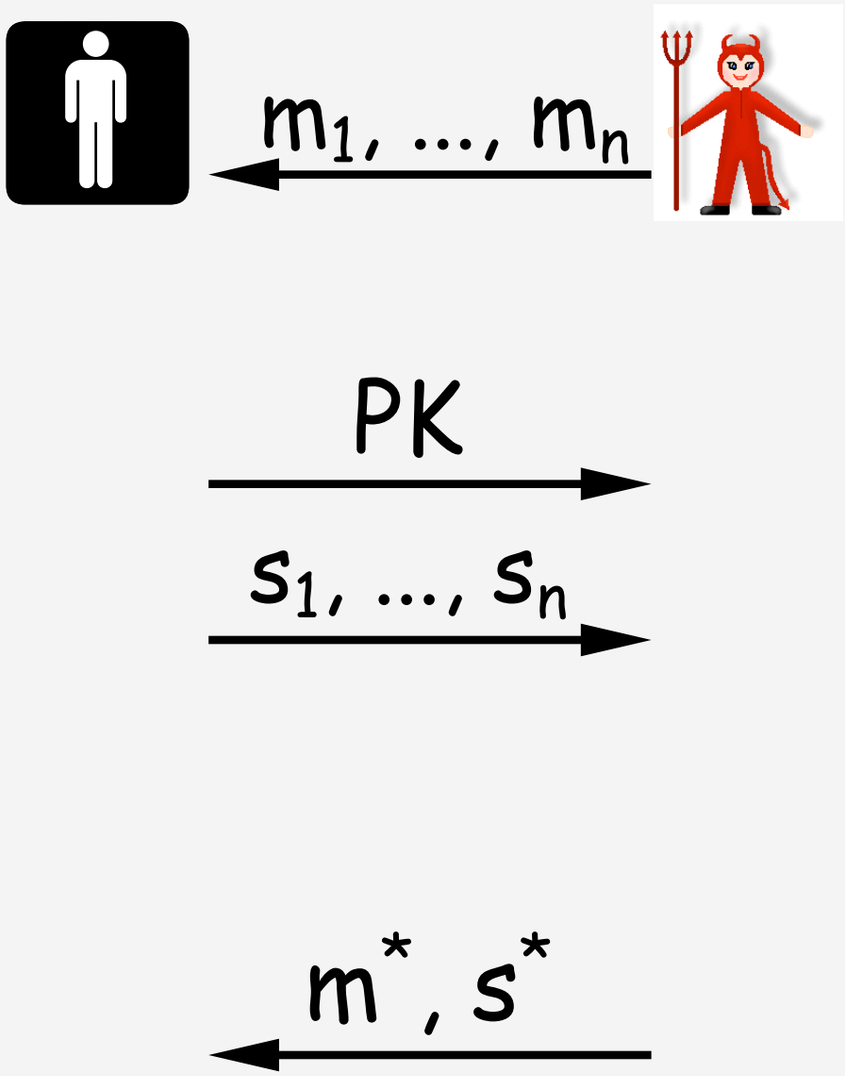
Proofs are tricky. How to answer all queries, except m*?

Definitions of Security

Full Security



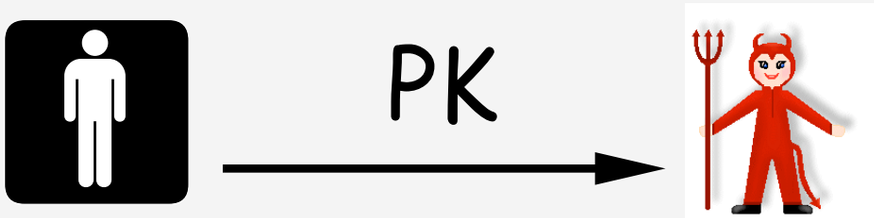
Weak Security



Negligible probability that Verify(PK, m^*, s^*)=1 and m^* is new.

Definitions of Security

Full Security



PK →

← m_1

s_1 →

← m_2

s_2 →

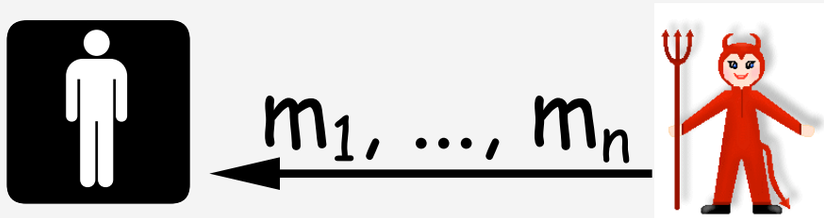
...

← m^*, s^*

<=

+

Weak Security



← $m_1, \ldots, m_n$

PK →

$s_1, \ldots, s_n$ →

← m^*, s^*

Chameleon Hash

exist under

-- factoring

-- RSA

-- discrete log

Theorem [ST01]: Full Signatures <= Chameleon Hash + Weak Signatures.

Gennaro-Halevi-Rabin Weak Sigs

Public Key: N, h, H: $\{0,1\}^* \to$ primes.

Sign: $s := h^{1/H(m)} \bmod N$.

Verify: Accept iff $h = s^{H(m)} \bmod N$.

Gennaro-Halevi-Rabin Weak Sigs

Public Key: N, h, H: {0,1}* -> primes.

Sign: $s := h^{1/H(m)} \bmod N$.

Verify: Accept iff $h = s^{H(m)} \bmod N$.

Strong RSA: Given (N,y), find any (x,e) s.t. $e > 1$ and $x^e = y \bmod N$.

Gennaro-Halevi-Rabin Weak Sigs

Public Key: N, h, H: {0,1}* -> primes.

Sign: $s := h^{1/H(m)} \bmod N$.

Verify: Accept iff $h = s^{H(m)} \bmod N$.

Strong RSA: Given (N,y), find any (x,e) s.t. e >1 and $x^e = y \bmod N$.

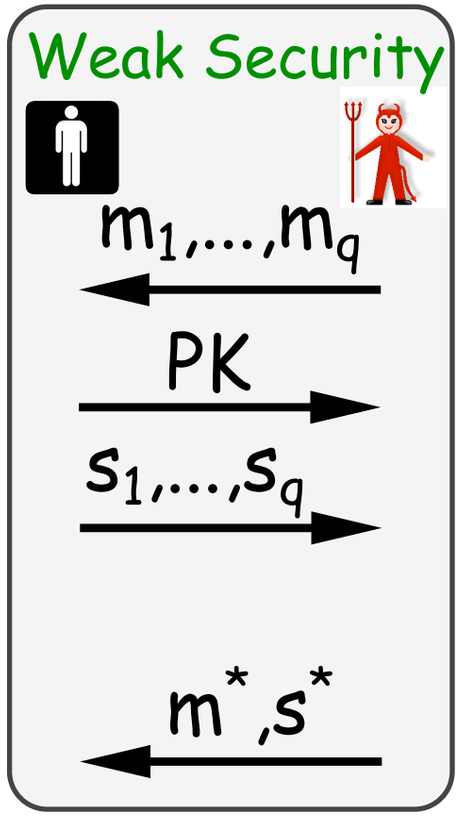
<u>Proof sketch</u>. Adversary gives $m_1, \ldots, m_q$.

Set $h := y^{e1\ e2 \ldots eq} \bmod N$, where $H(m_i) = e_i$.

To sign m_i, leave e_i out of product.

On forgery, $s^{*\ e^*} = h = y^{e1\ e2 \ldots eq}$, where $H(m^*) = e^*$.

Use Shamir's trick to get x s.t. $x^{e^*} = y \bmod N$.



Gennaro-Halevi-Rabin Weak Sigs

Public Key: N, h, H: {0,1}* -> primes.

Sign: $s := h^{1/H(m)} \bmod N$.

Verify: Accept iff $h = s^{H(m)} \bmod N$.

Strong RSA: Given (N,y), find **any** (x,**e**) s.t. e >1 and $y = x^e \bmod N$.

Pr… gives $m_1, \ldots, m_q$.

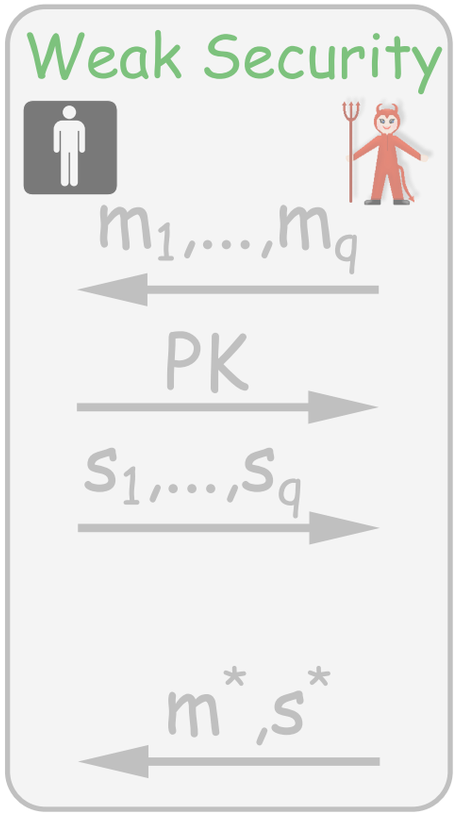
…ere $H(m_i) = e_i$.

…

On… $e_2 \ldots e_q$, where $H(m^*) = e^*$.

Use Shamir's trick to get x s.t. $x^{e^*} = y \bmod N$.

No idea where to embed single e, so push issue to the assumption.



Gennaro-Halevi-Rabin Weak Sigs

Public Key: N, h, H: {0,1}* -> primes.

Sign: $s := h^{1/H(m)} \bmod N$.

Verify: Accept iff $h = s^{H(m)} \bmod N$.

RSA: Given (N,y,e*), find the x s.t. e >1 and $y = x^{e^*} \bmod N$.

Pr gives $m_1, \ldots, m_q$.

ere $H(m_i) = e_i$.

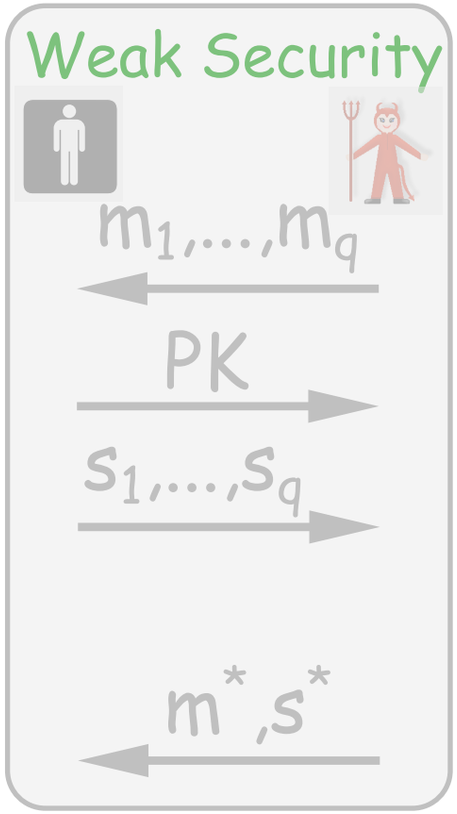
t.

On e2...eq, where $H(m^*) = e^*$.

Use Shan to get x s.t. $x^{e^*} = y \bmod N$.

If we knew m*,
we could program H
with single RSA e*.

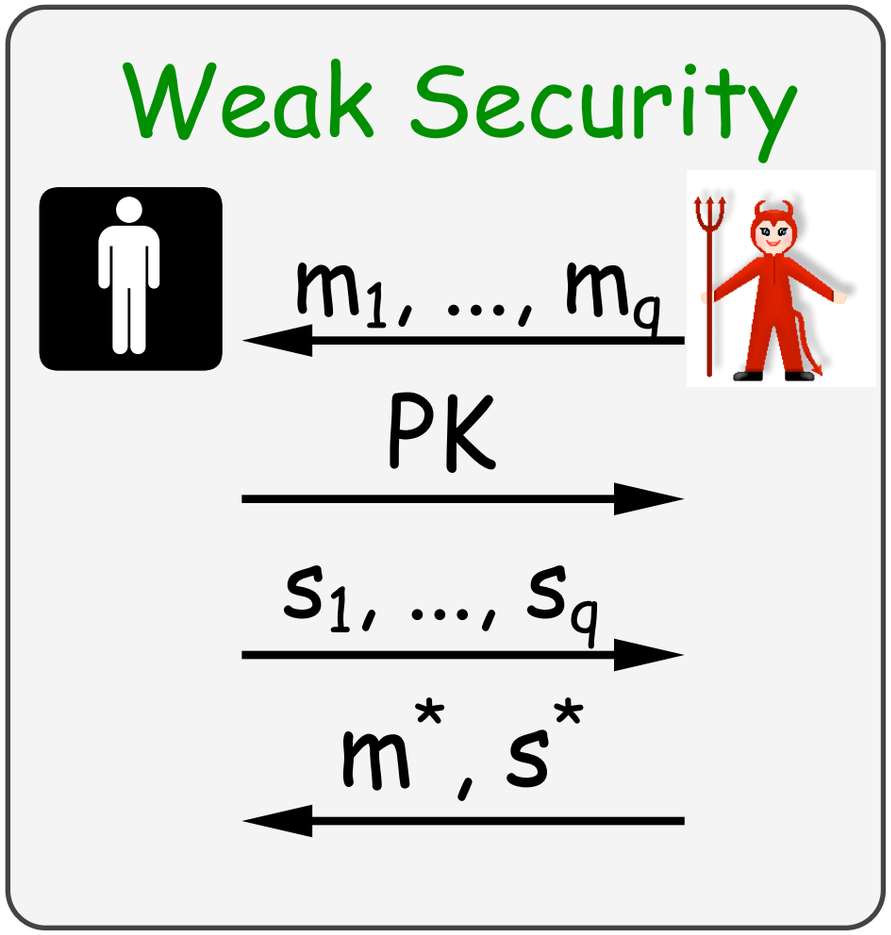
... what do we know
about m*??



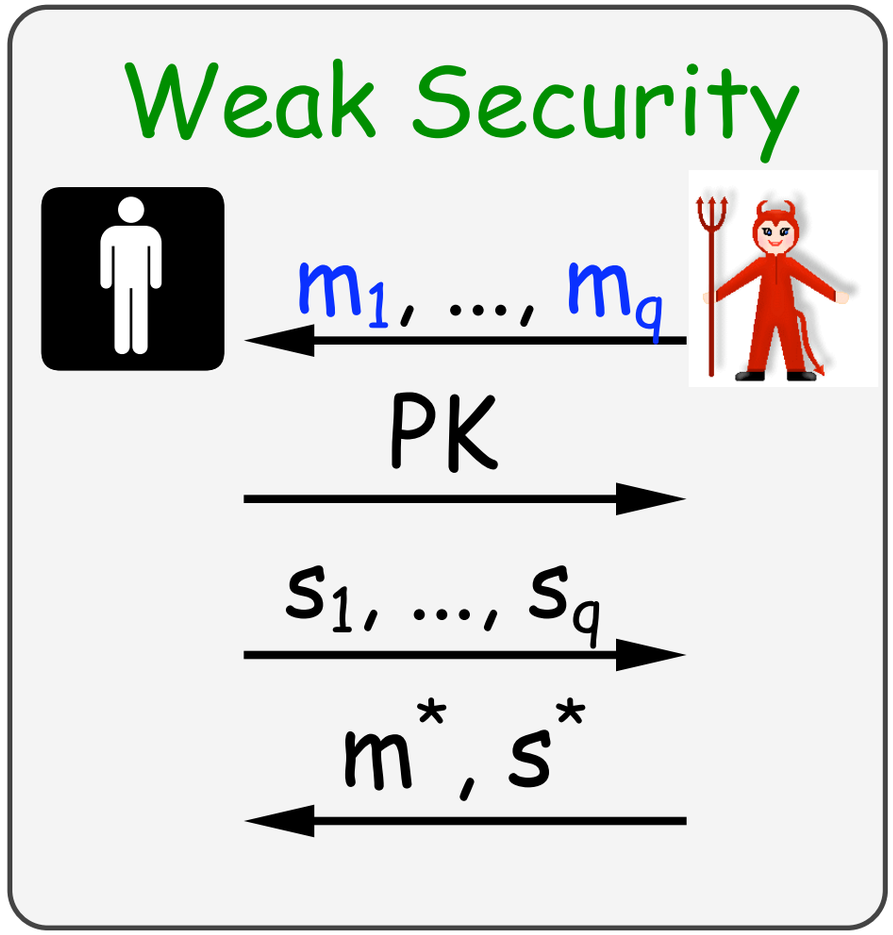
A New Technique for Designing

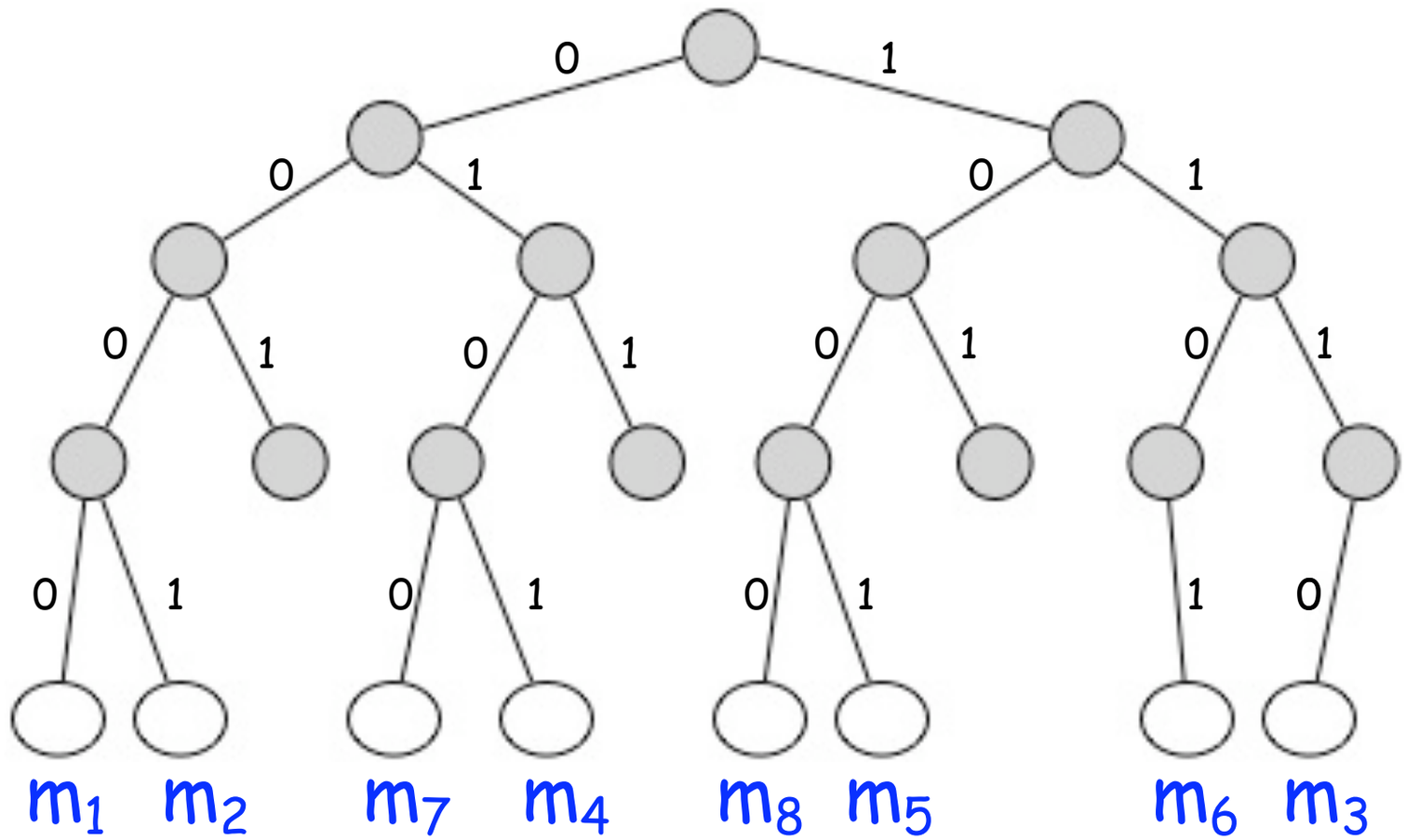
S
Si
Sig
Sign
Signa
Signat
Signatu
Signatur
Signature
Signatures

What about m*?

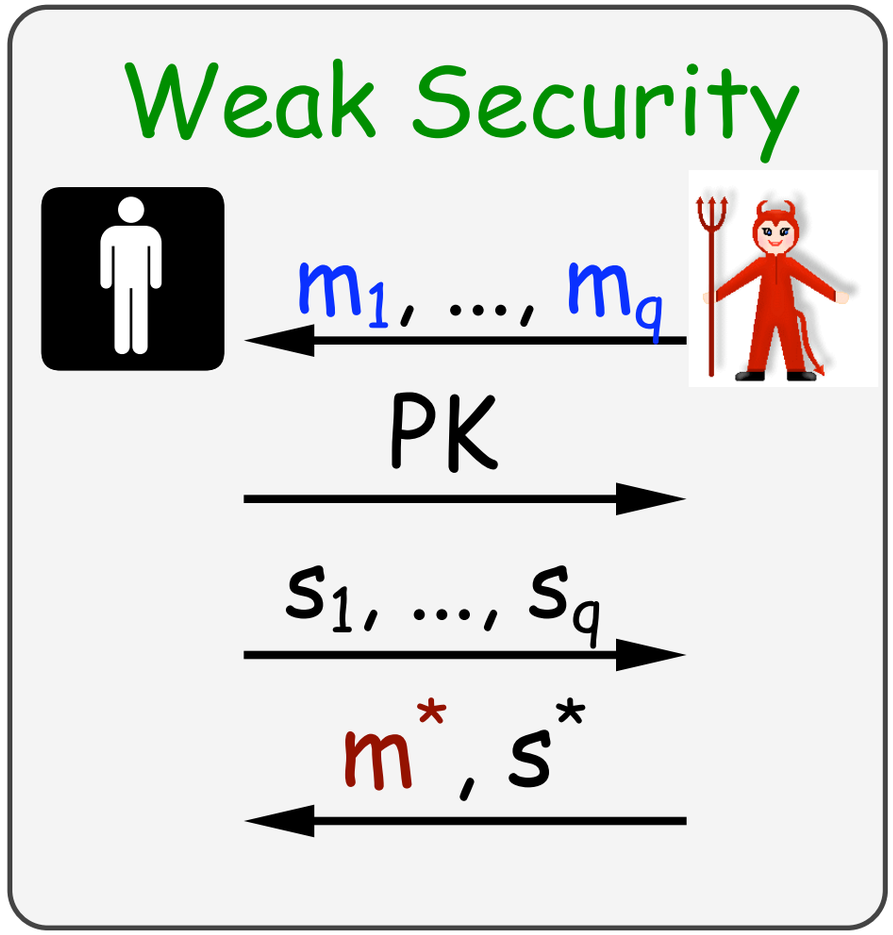


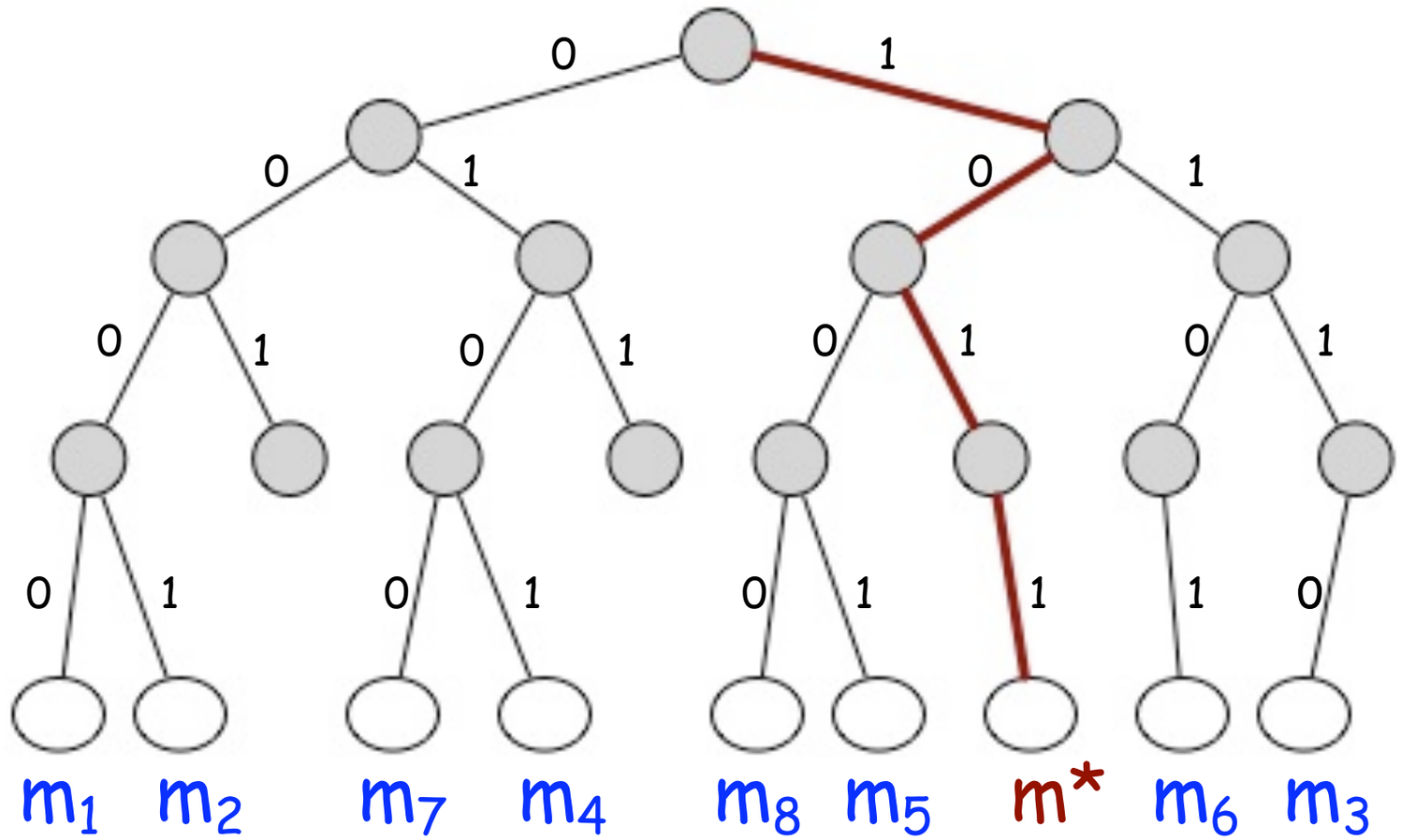
What about m*?


Weak Security
$m_1, \ldots, m_q$
PK
$s_1, \ldots, s_q$
m^*, s^*

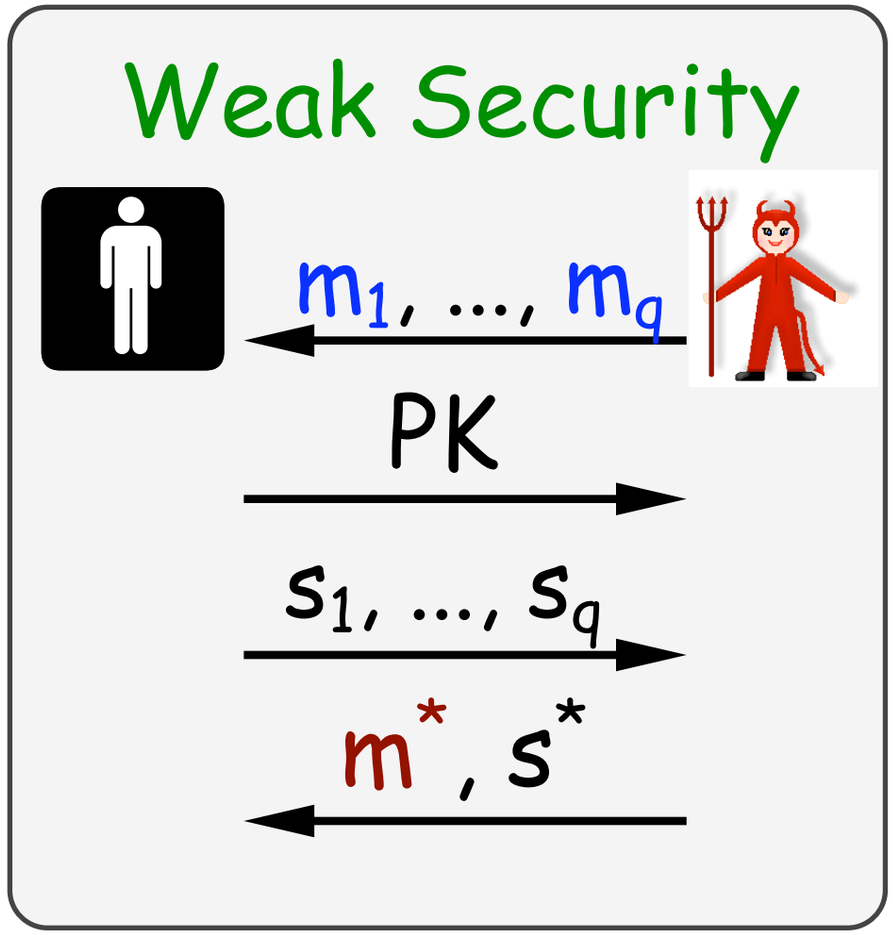

0
1
m_1 m_2 m_7 m_4 m_8 m_5 m_6 m_3

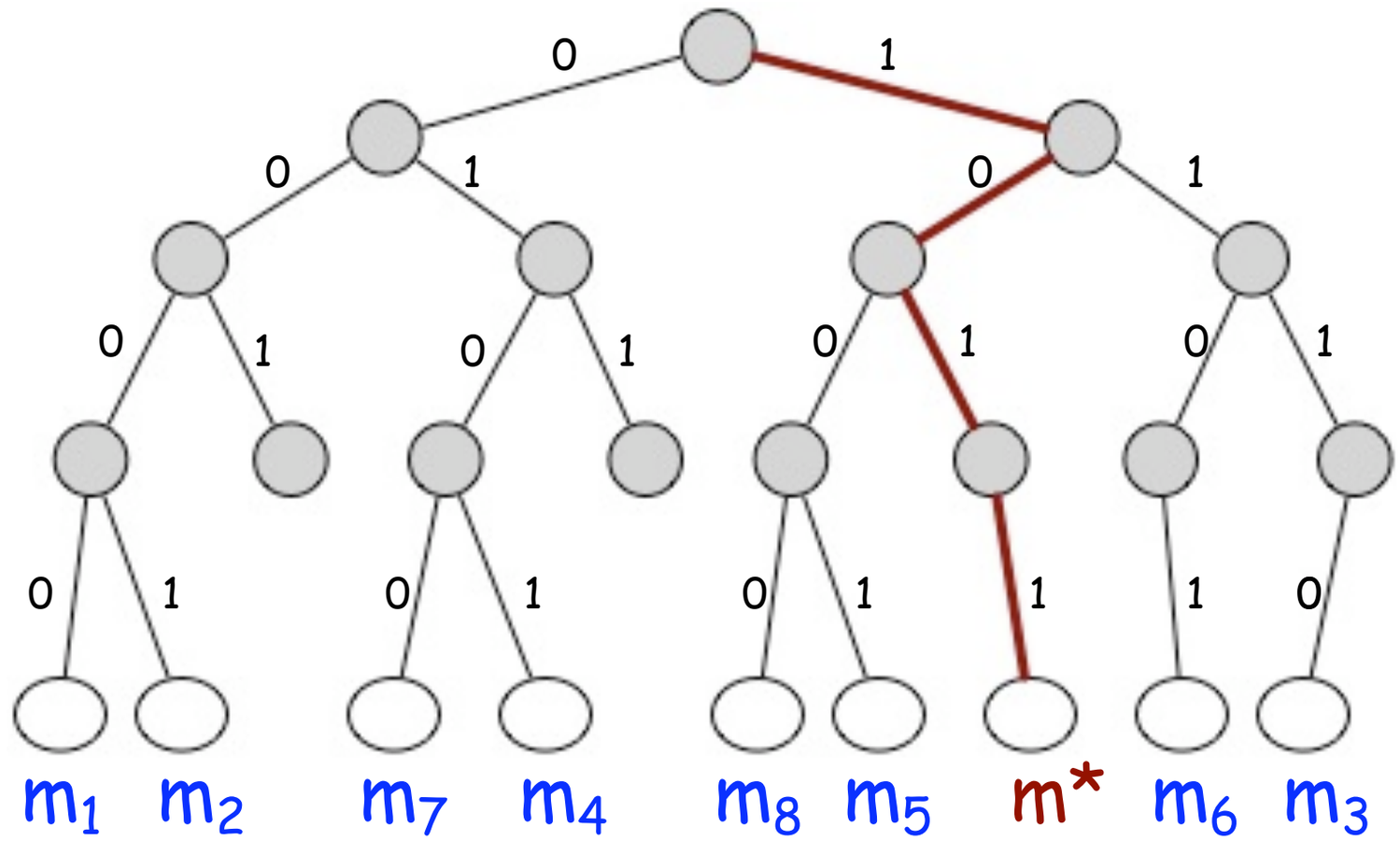
What about m*?


Weak Security
$m_1, \ldots, m_q$
PK
$s_1, \ldots, s_q$
m^*, s^*


m_1 m_2 m_7 m_4 m_8 m_5 m^* m_6 m_3

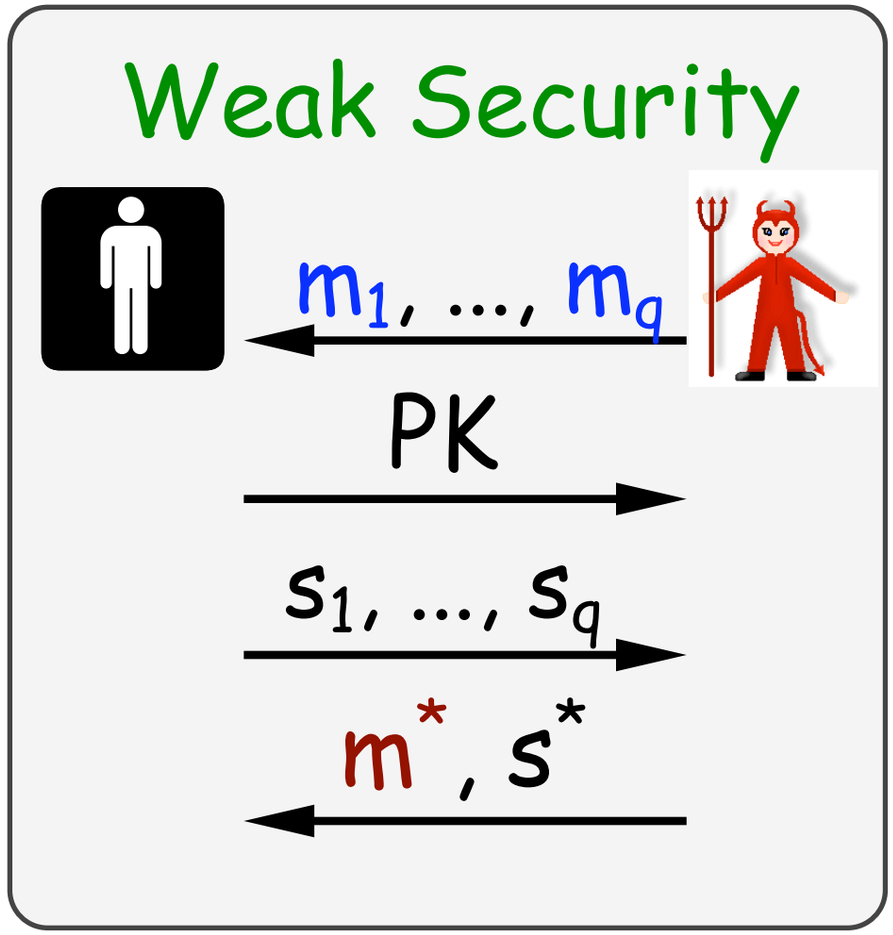
What about m*?

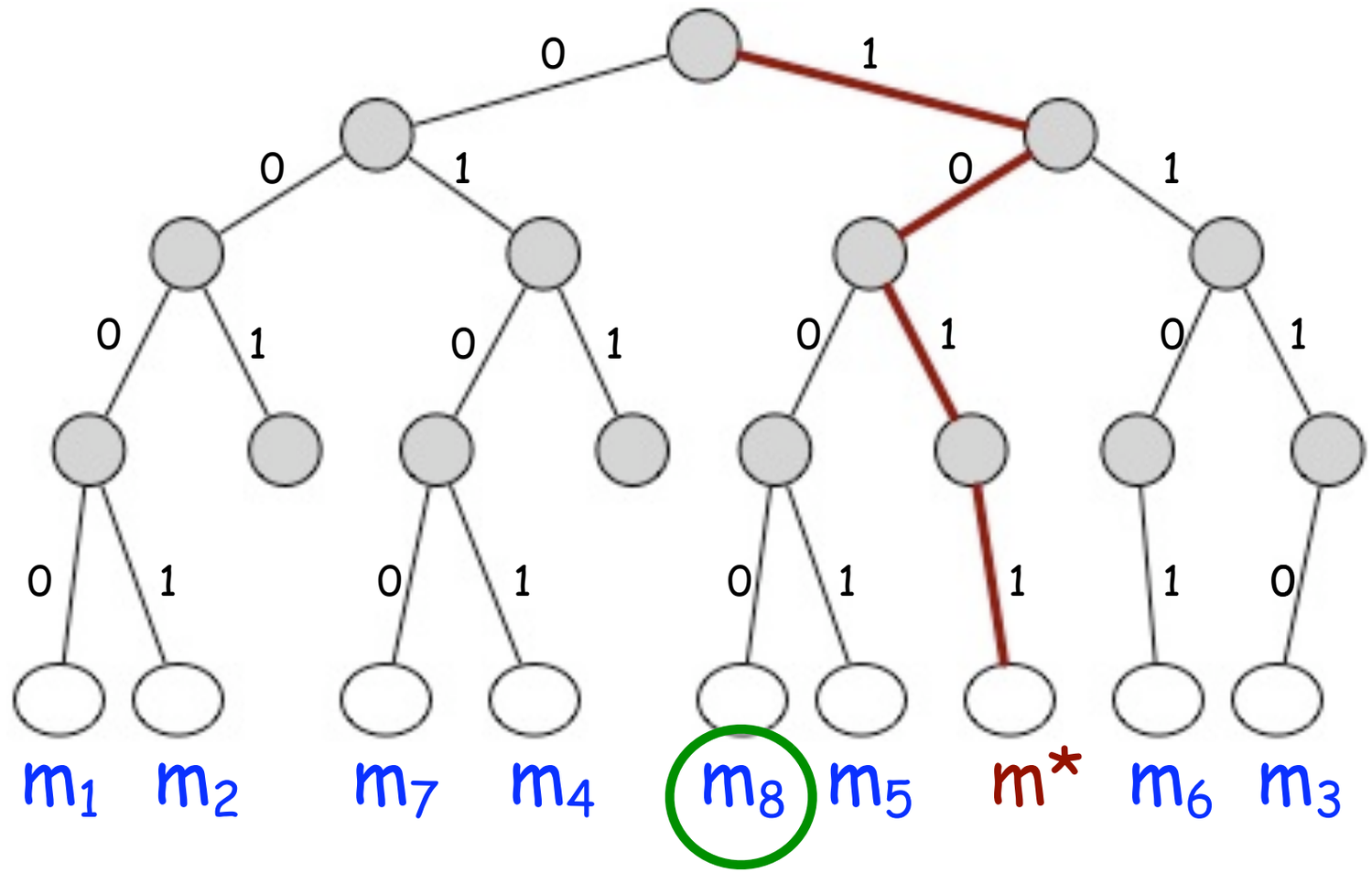




Shortest unique prefix of m^* = 101.

What about m*?

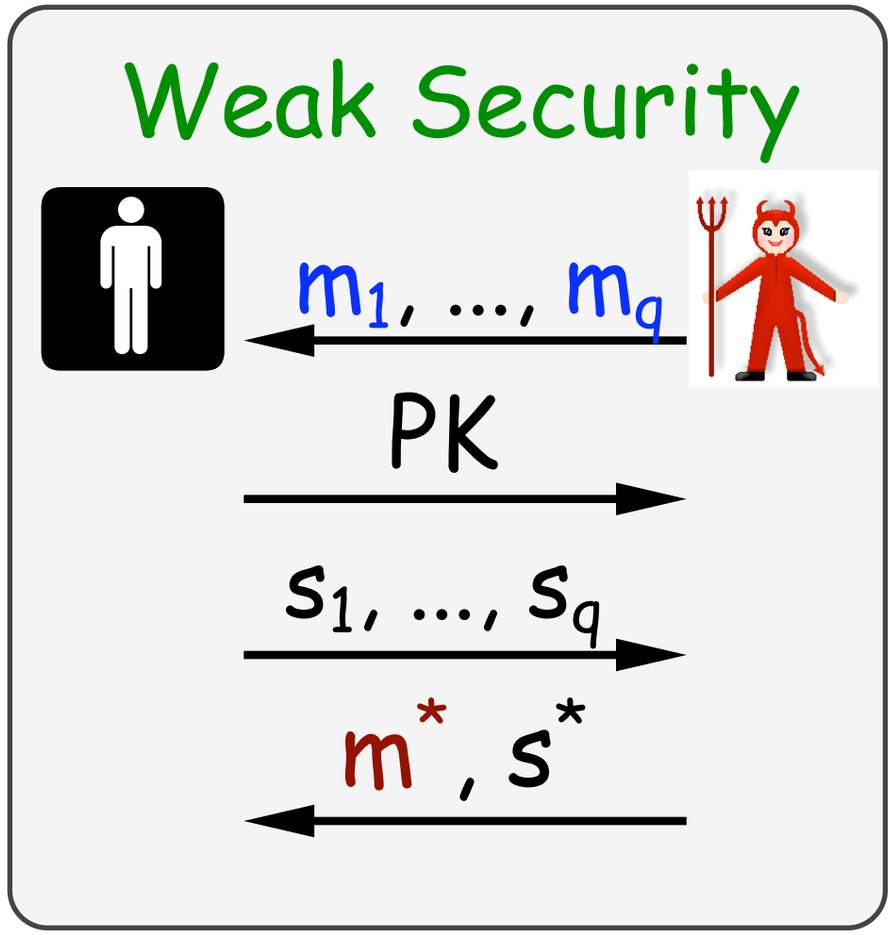


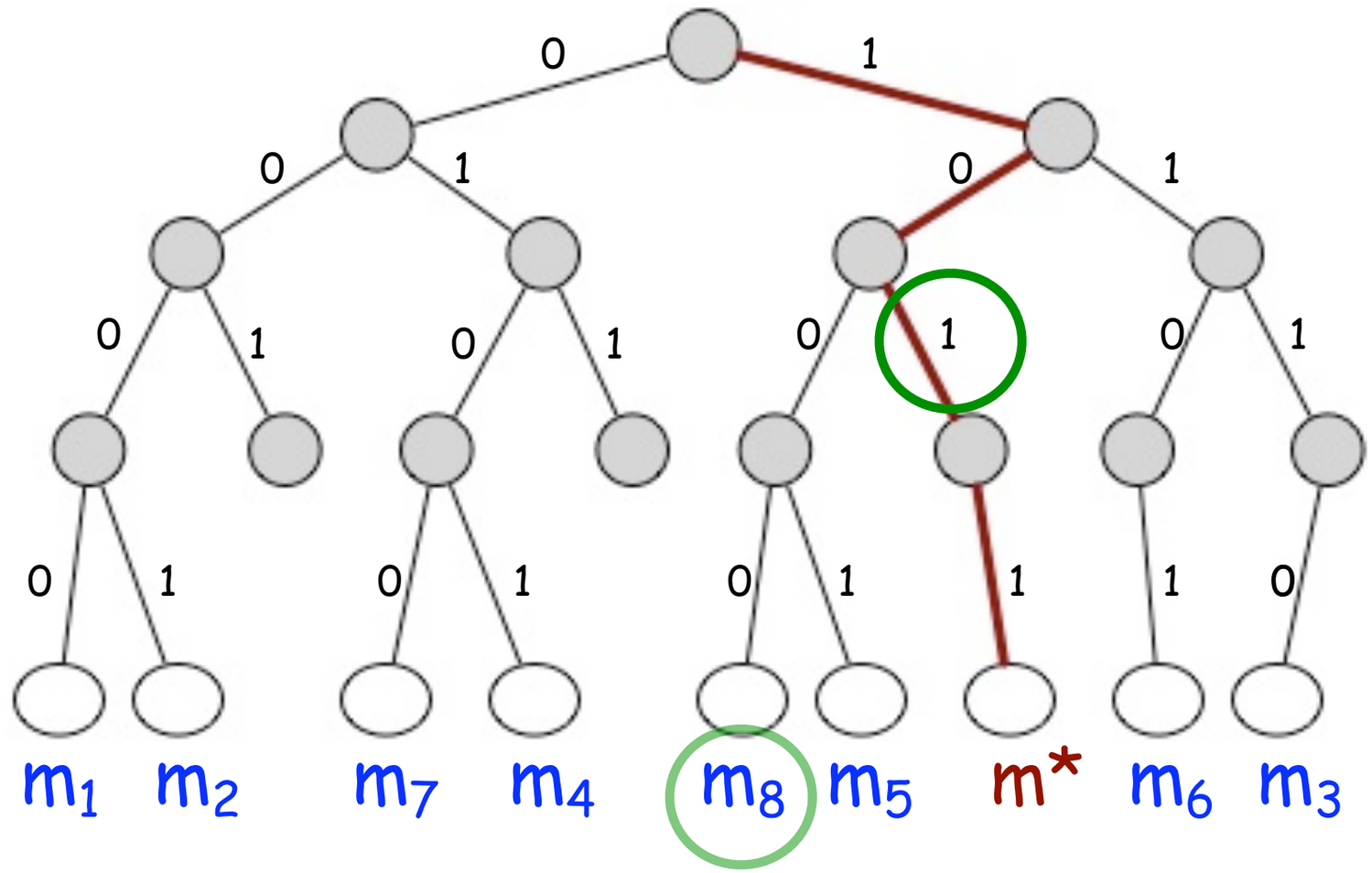


Shortest unique prefix of m^* = 101.

IDEA: Guess this prefix (before seeing m*).
-- guess m_i which m* follows longest: >= 1/q chance.

What about m*?

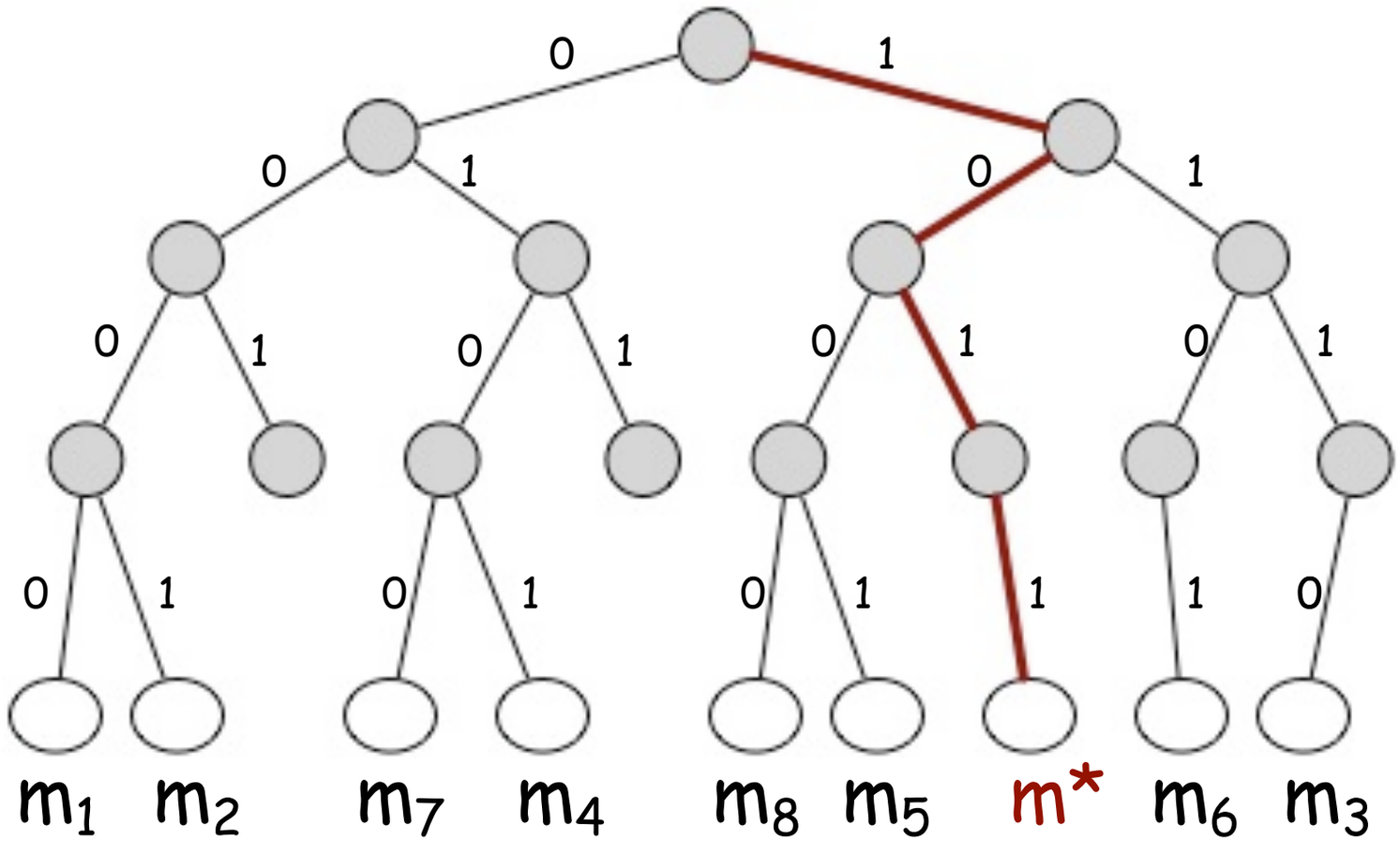




Shortest unique prefix of m^* = 101.

IDEA: Guess this prefix (before seeing m*).
-- guess m_i which m* follows longest: >= 1/q chance.
-- guess first bit where m* differs: >= 1/n chance.

What to do with observation?



Shortest unique prefix of m* = 101

1. Design signature using all prefixes of message.
2. Guess unique prefix of m* to embed challenge.

RSA Construction

Public Key: N, h, and H: {0,1}* -> primes.

Sign: Let M^i := first i bits of M.

$s := h^{1/e_1 e_2 \ldots e_n} \bmod N$, where $e_i := H(M^i)$.

Verify: Accept iff $h = s^{e_1 e_2 \ldots e_n} \bmod N$, where $e_i := H(M^i)$.

GHR: $s := h^{1/H(M)} \bmod N$

RSA Construction

Public Key: N, h, and H: {0,1}* -> primes.

Sign: Let M^i := first i bits of M.

$s := h^{1/e1\ e2 \ldots en} \bmod N$, where $e_i := H(M^i)$.

Verify: Accept iff $h = s^{e1\ e2 \ldots en} \bmod N$, where $e_i := H(M^i)$.

RSA: Given (N,y,e), find the x s.t. e >1 and $y = x^e \bmod N$.

RSA Construction

Public Key: N, h, and H: {0,1}* -> primes.

Sign: Let M^i := first i bits of M.

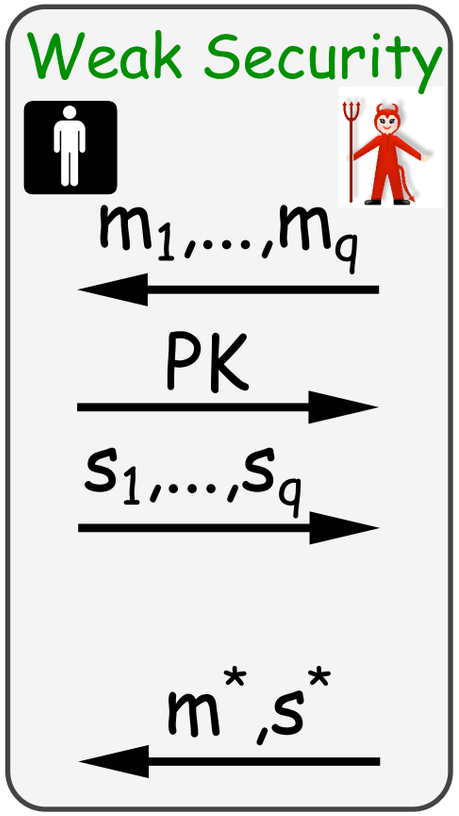
$s := h^{1/e_1 e_2 \ldots e_n} \bmod N$, where $e_i := H(M^i)$.

Verify: Accept iff $h = s^{e_1 e_2 \ldots e_n} \bmod N$, where $e_i := H(M^i)$.

RSA: Given (N,y,e), find the x s.t. e >1 and $y = x^e \bmod N$.

<u>Proof sketch</u>. Adversary gives $M_1, \ldots, M_q$.

1. Guess w* as shortest unique prefix of M*.
2. Choose H so that H(w*) = e.
3. $h := y^{(\text{product of hash of all prefixes of } M_1,\ldots,M_q)}$.
4. Sign for $M_1, \ldots, M_q$ by omit from product.
5. Extract x from M* forgery by Shamir's Trick.



Performance

Public Key: O(1) elements (N, h, hash descriptions)

Signature: 1 element in Z_N^*, 1 integer

Signing: 1 exp. E(primality tests) = nk.

Verification: n exp. E(primality tests) = nk.

n = length of message, k = security parameter

Performance

Public Key: O(1) elements (N, h, hash descriptions)

Signature: 1 element in Z_N^*, 1 integer

Signing: 1 exp. E(primality tests) = nk.

Verification: n exp. E(primality tests) = nk.

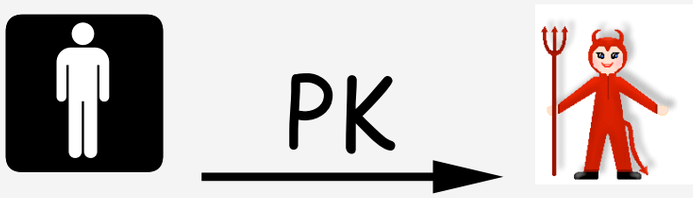
n = length of message, k = security parameter

Optimizations?

1. online/offline - do signing work offline [ST01].
2. use larger alphabet for prefixes
   - v bit chunk => n/v primes, but security loss of $1/(2^v - 1)$.
3. longer signature to speed up hash
   - adding ~n log(k) bits => E(primality tests) = n.
3. hash to smaller primes?
   - good idea, slightly changes the RSA assumption.

Definitions of Security

Full Security



$\xrightarrow{PK}$

$\xleftarrow{m_1}$

$\xrightarrow{s_1}$

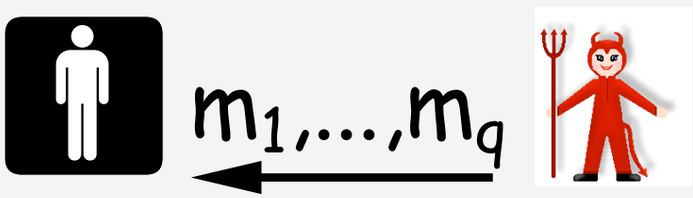
$\xleftarrow{m_2}$

$\xrightarrow{s_2}$

...

$\xleftarrow{m^*,s^*}$

Weak Security



$\xleftarrow{m_1,\ldots,m_q}$

$\xrightarrow{PK}$

$\xrightarrow{s_1,\ldots,s_q}$

$\xleftarrow{m^*,s^*}$

+

Chameleon Hash

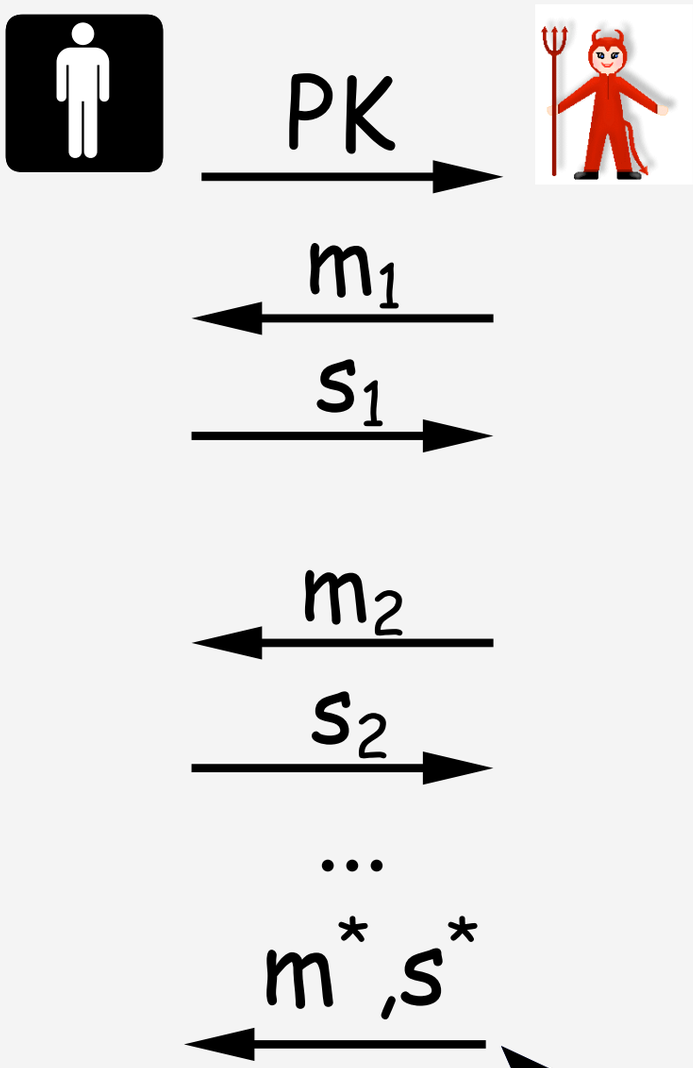
exist under

- -- factoring
- -- RSA
- -- discrete log

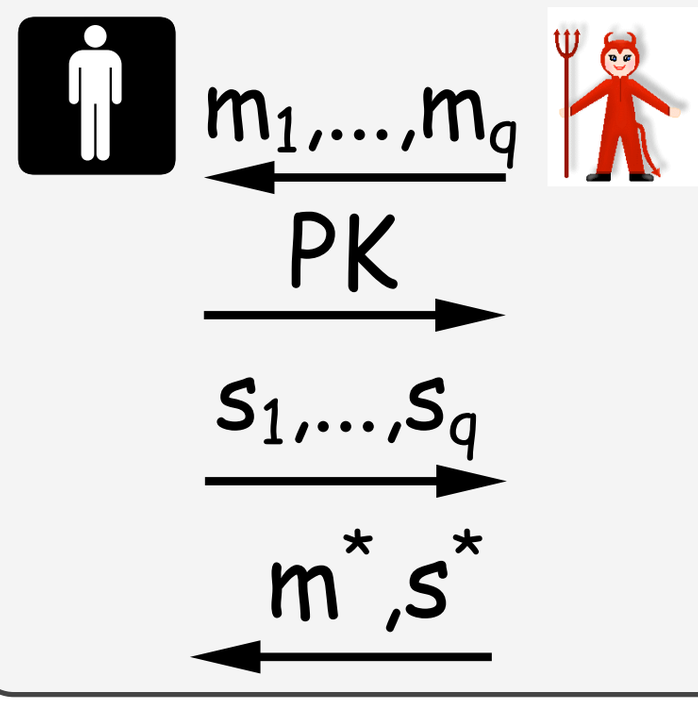
Theorem [ST01]: Full Signatures <= Chameleon Hash + Weak Signatures.

Definitions of Security

Full Security



Weak Security



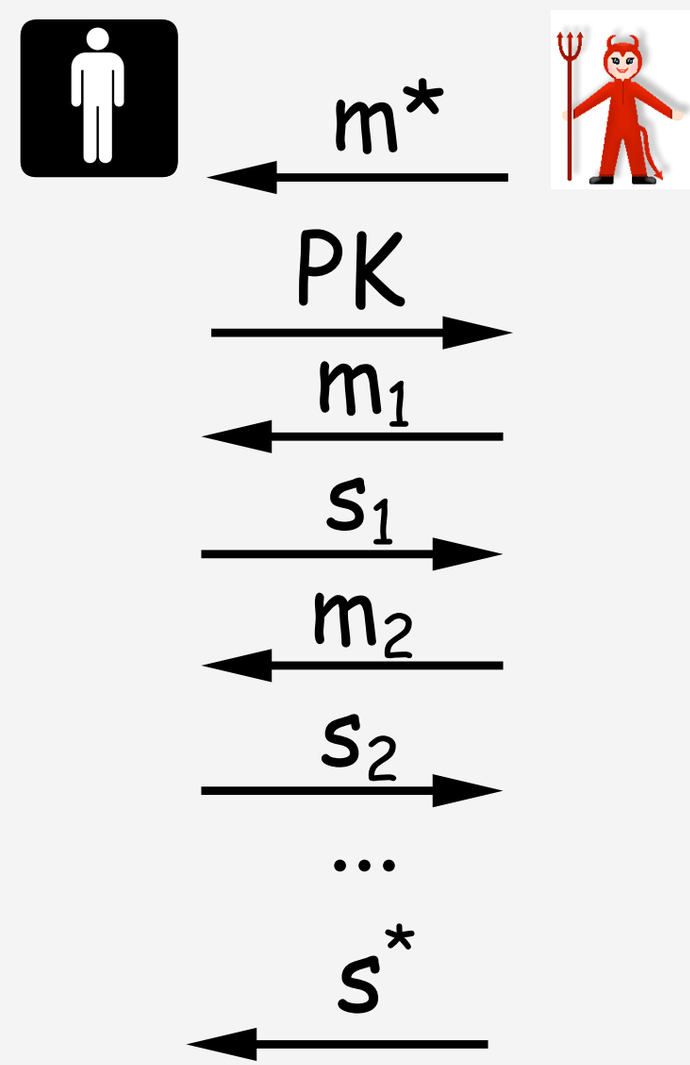
+

Chameleon Hash

exist under

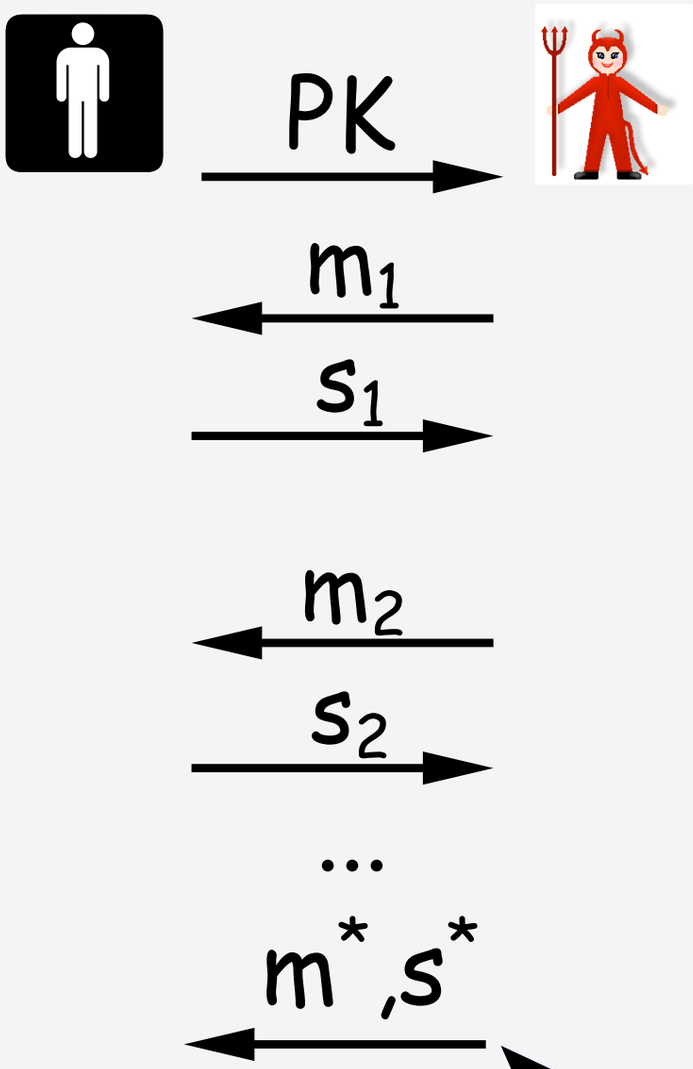
- -- factoring
- -- RSA
- -- discrete log

Selective Security

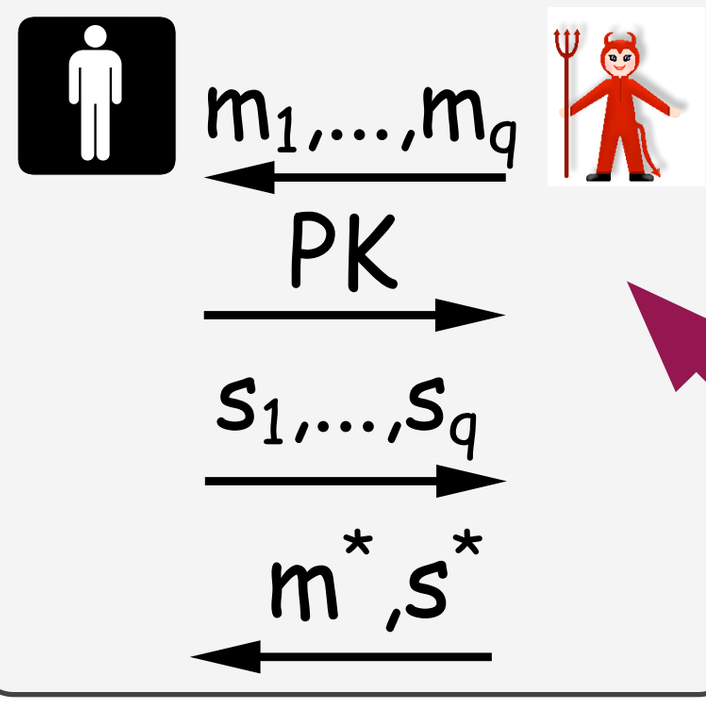


Definitions of Security

Full Security



Weak Security



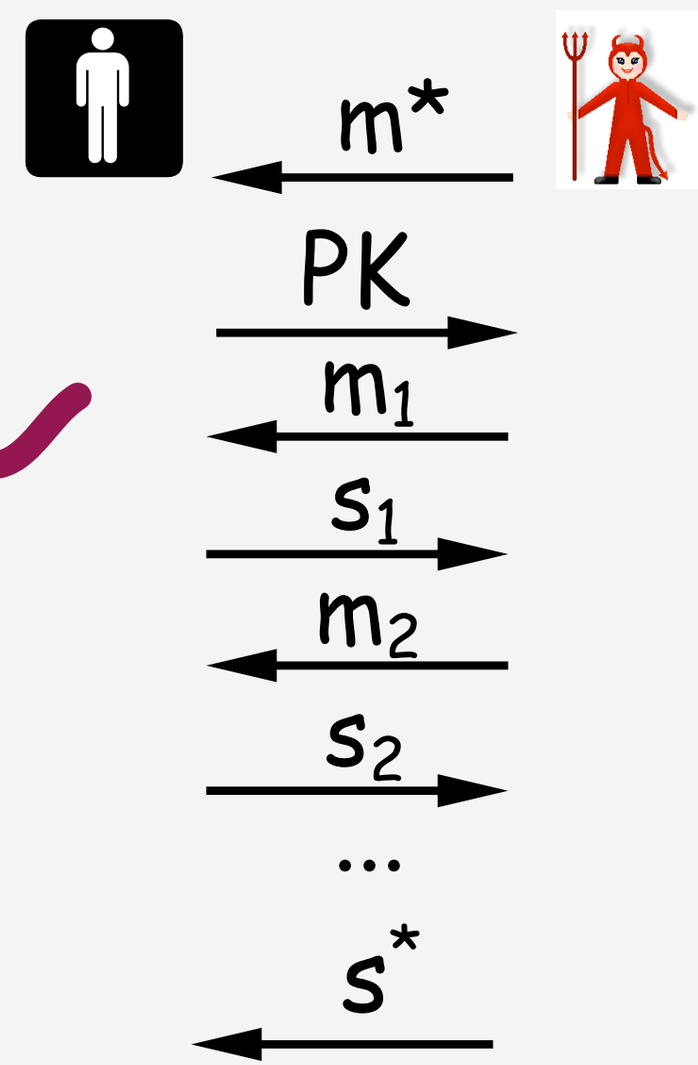
+

Chameleon Hash

exist under

- -- factoring
- -- RSA
- -- discrete log

Selective Security



[HW09b]: a non-generic technique for selective to weak security.

Wider Application of Technique

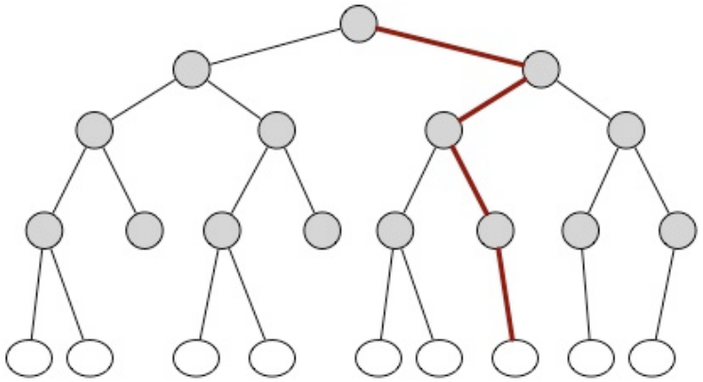
The prefix technique has applications beyond RSA.

-- We also show new proof for Waters signatures under CDH in bilinear groups.

-- Recently, [Cash-Hofheinz-Kiltz, Peikert, Agarwal-Boyen] present lattice analog of the Canetti-Halevi-Katz selectively-secure IBE.

Admits selectively-secure signatures [Naor].

Apply our techniques to realize full signatures, as explicitly done by [Peikert].



Open Directions

1. Better performance under RSA.

2. Generalize selective to full security.

3. Short, standard model signatures from
   -- discrete logarithm
   -- CDH <u>without</u> bilinear groups

4. Standard model/assumptions for:
   -- anonymous credentials
   -- electronic cash
   -- aggregate signatures
   -- etc.