Transcript Collision Attacks:

Breaking Authentication in TLS, IKE and SSH

or: MD5 MUST DIE

http://sloth-attack.org

Karthikeyan Bhargavan
Gaëtan Leurent

INVENTORS FOR THE DIGITAL WORLD

Agility vs. Downgrade Attacks

Crypto protocols and applications *evolve*

- SSL v3 → TLS 1.2
- DH-768 → Curve25519
- MD5 → SHA-256

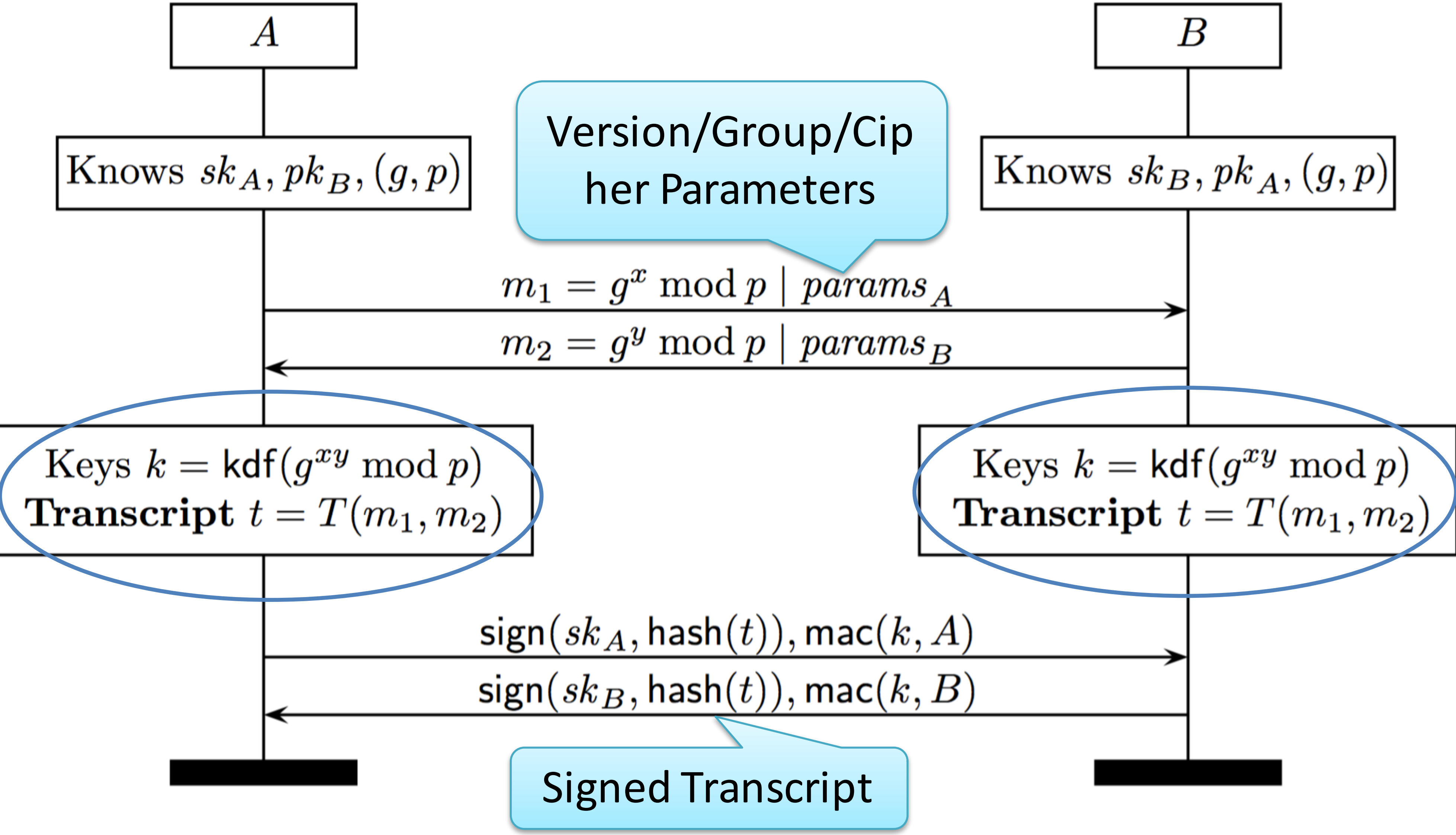
Agility: graceful transition from old to new

- Negotiate best shared version, cipher, DH group

What can go wrong?

- We get lazy and forget to remove weak algorithms
- Downgrade attacks: POODLE, LOGJAM, **SLOTH**

Authenticated DH with Negotiation



What Transcript to Sign?

- Sign the full message trace
  - **sign**(sk_B, **hash**(m_1 | $m2$))
  - *Example*: TLS 1.3, SSH-2, TLS 1.2 client auth

- Sign your ephemerals, MAC the transcript
  - **sign**(sk_B, **hash**($nonce_A$ | $nonce_B$ | g | p | g^y))
  - *Example*: TLS 1.2 server auth

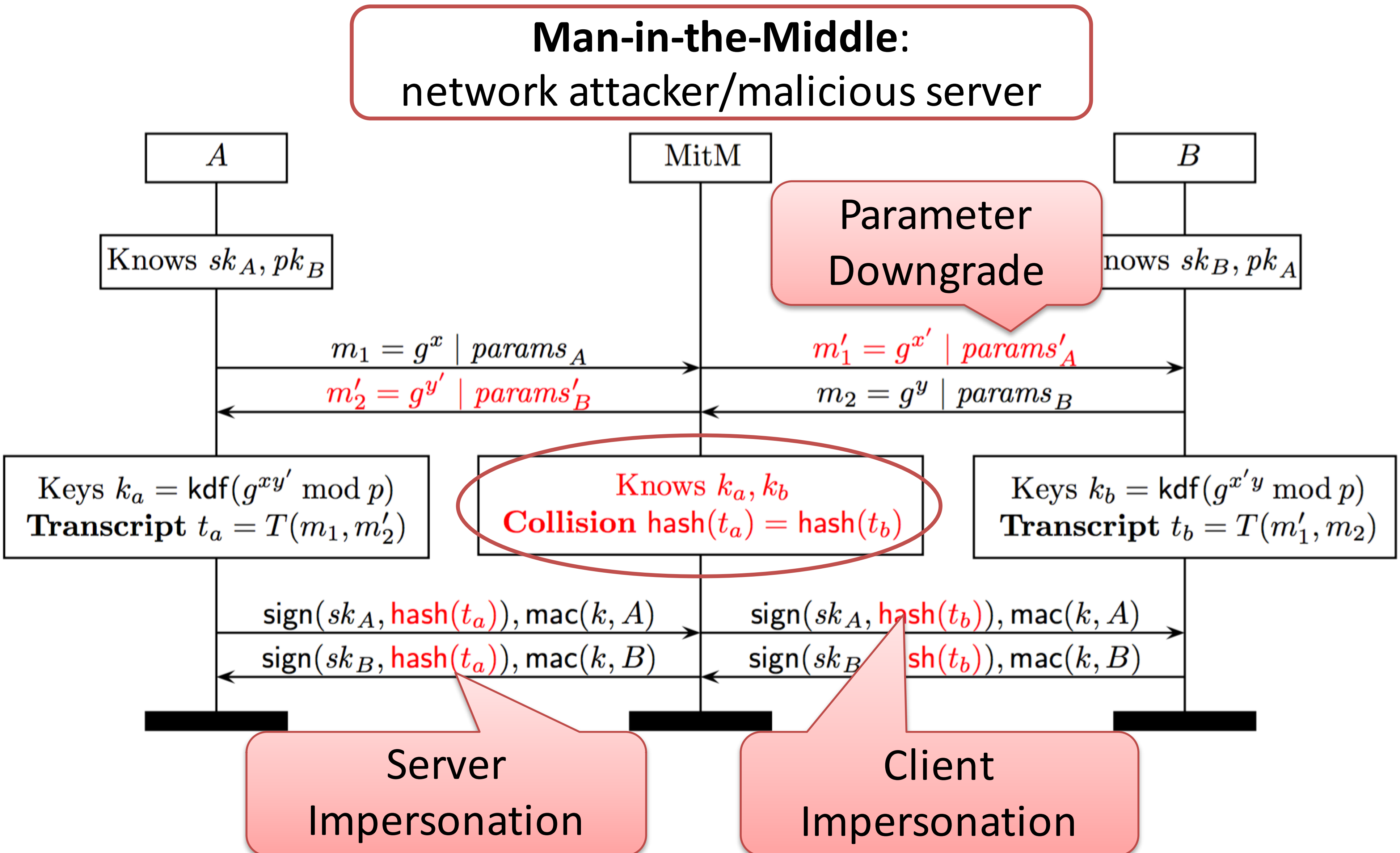
- Sign your own messages and MACed identity
  - **sign**(sk_A, **hash**(m_1 | **mac**(k,A)))
  - **sign**(sk_B, **hash**(m_2 | **mac**(k,B)))
  - *Example:* IKEv2 initiator, responder, EAP auth

Using Weak Hash Functions

- Sign the full transcript
  - **sign**(sk_B, **hash**(m_1 | $m2$))
  - *Example*: TLS 1.3, SSH-2, TLS 1.2 client auth

- How weak can the **hash** function be?
  - do we need collision resistance?
  - do we only need 2nd preimage resistance?
  - Is it still safe to use MD5, SHA-1 in TLS, IKE, SSH?
  - *Disagreement*: cryptographers vs. practitioners (see Schneier vs. Hoffman, RFC4270)

SLOTH: Transcript Collision Attacks



Computing a Transcript Collision

$$\textbf{hash}(m_1 \mid m'_2) = \textbf{hash}(m'_1 \mid m_2)$$

- We need to compute a collision, *not a preimage*
  - Attacker controls parts of both transcripts
  - If we know the black bits, can we compute the red bits?
  - This is usually called a **generic collision**
- If we're lucky, we can set up a **shortcut** collision
  - **Common-prefix**: collision after a shared transcript prefix
  - **Chosen-prefix:** collision after attacker-controlled prefixes

Primer on Hash Collision Complexity

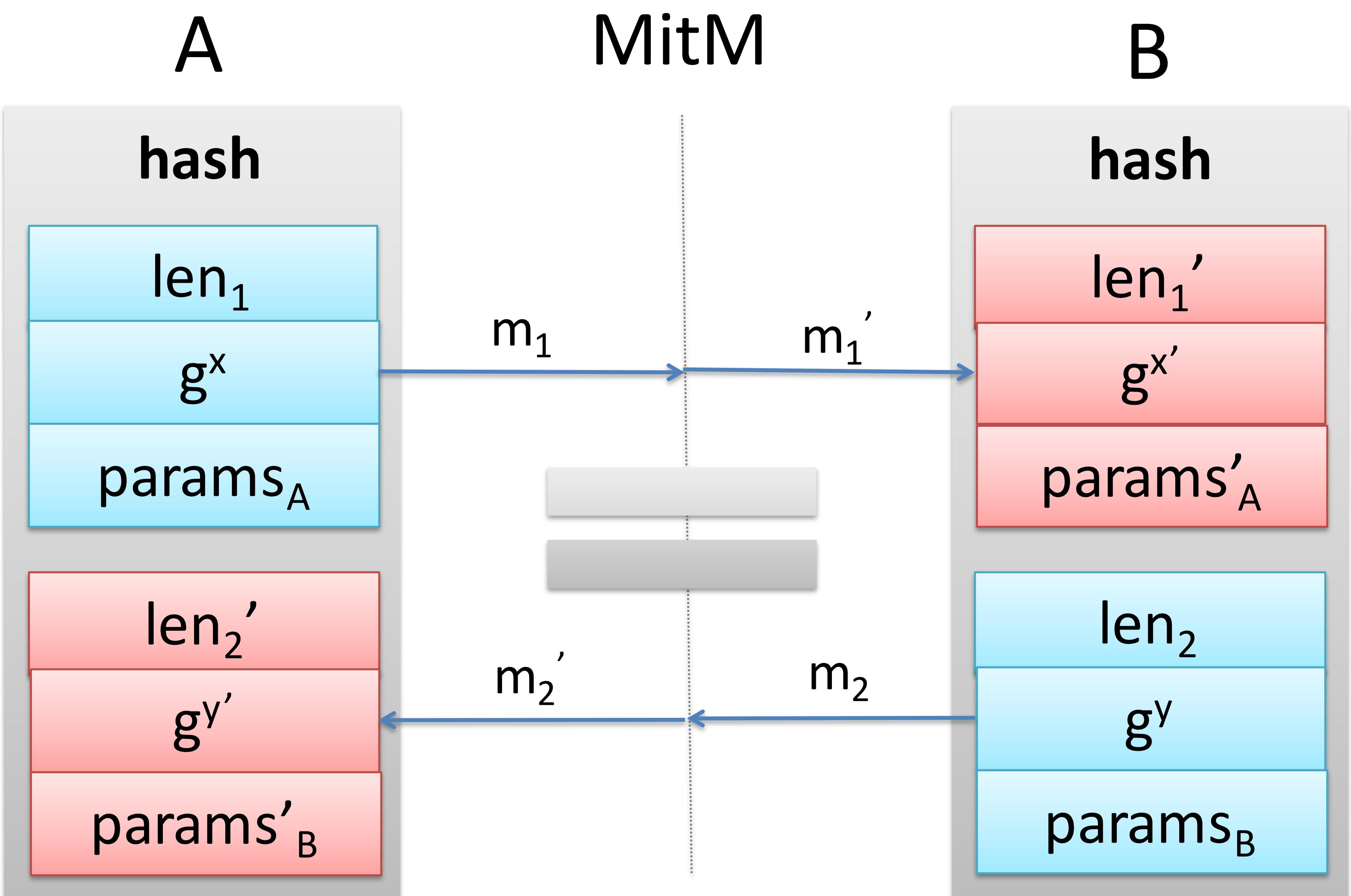
- MD5: known attack complexities
  - **MD5** second preimage 2^{128} hashes
  - **MD5** generic collision: (birthday) 2^{64} hashes
  - **MD5** chosen-prefix collision: 2^{39} hashes (1 hour)
  - **MD5** common-prefix collision: 2^{16} hashes (seconds)

- SHA1: estimated attack complexities
  - **SHA1** second preimage 2^{160} hashes
  - **SHA1** generic collision: (birthday) 2^{80} hashes
  - **SHA1** chosen-prefix collision: 2^{77} hashes (?)

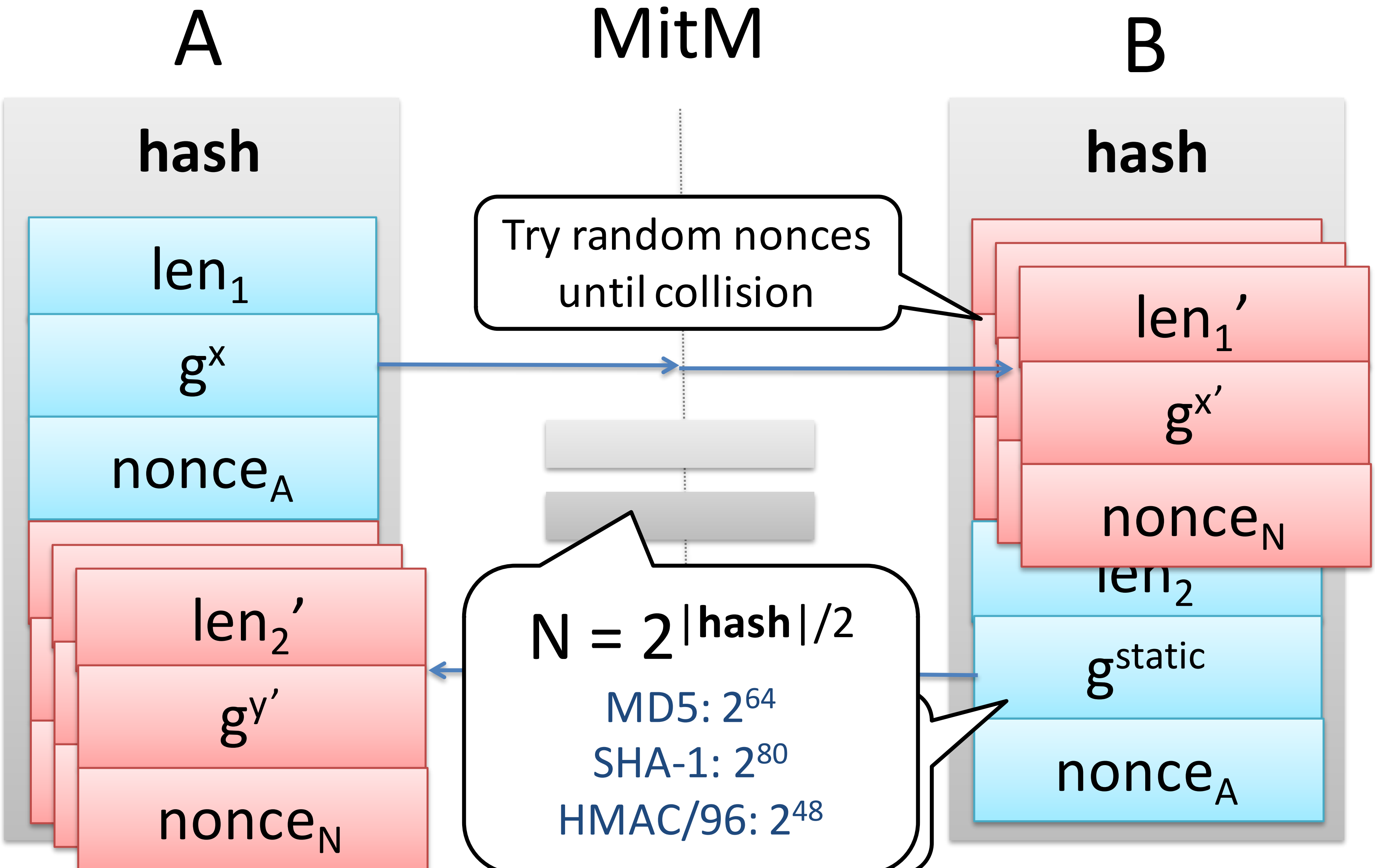
Composite Hash Constructions

- When used as **transcript hash functions** many constructions are not collision resistant
  - **MD5**(x) | **SHA1**(x)
    not much better than SHA1
  - **HMAC-MD5**(k,x)
    not much better than MD5
  - **HMAC-SHA256**(k,MD5(x))
    not much better than MD5
  - **Truncated HMAC-SHA256**(k,x) to N bits
    not much better than a N bit hash function

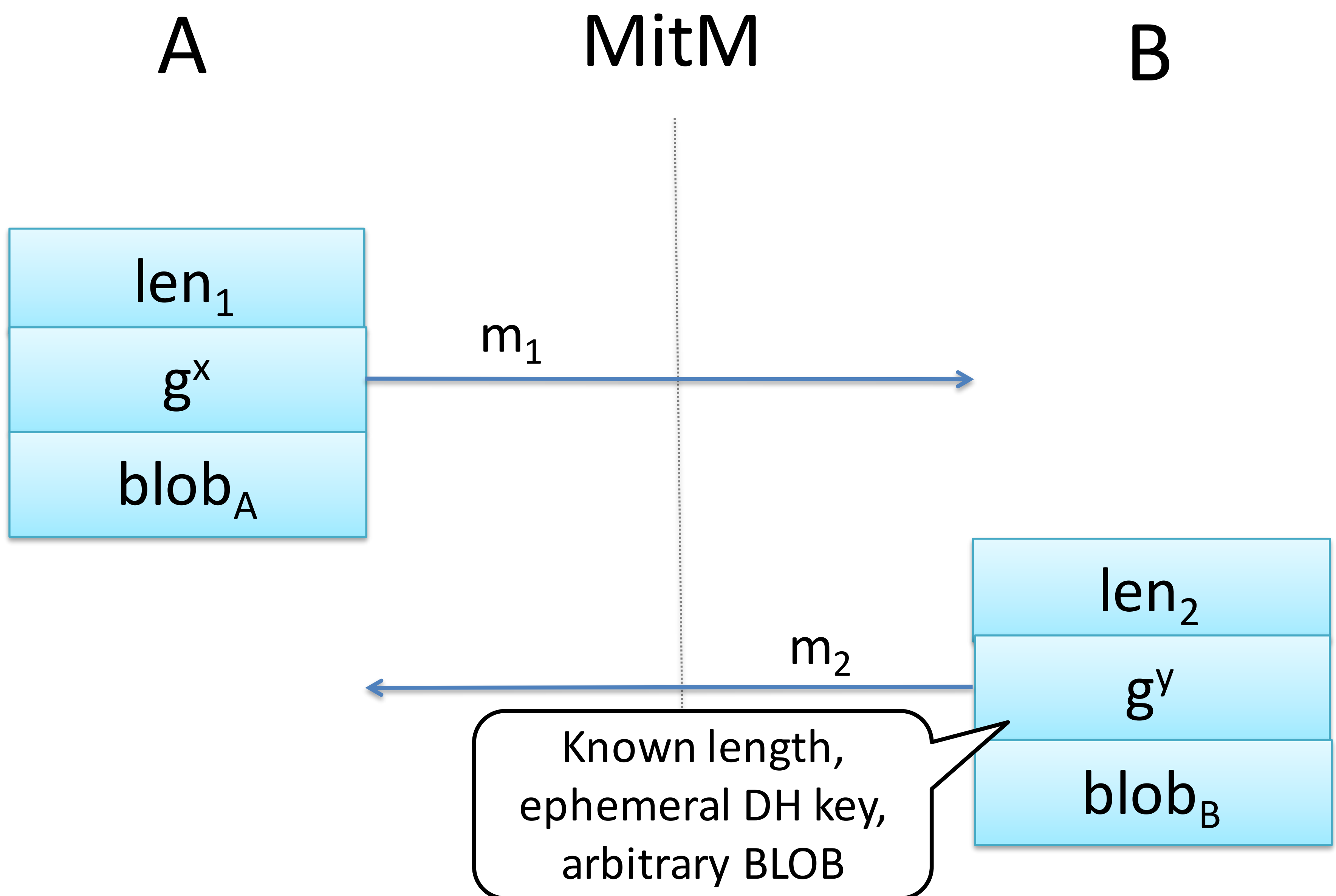
Computing Transcript Collisions

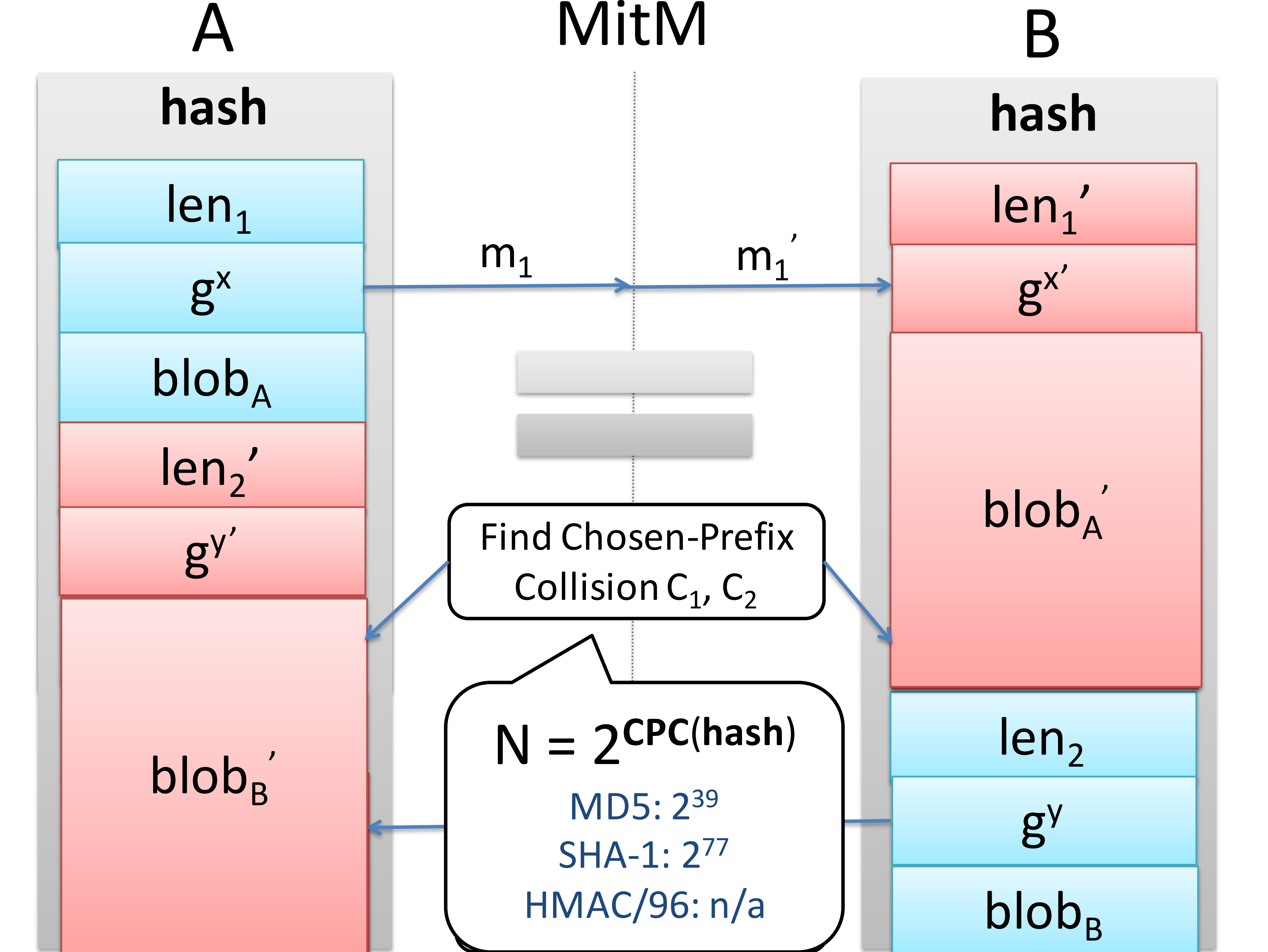


Generic Transcript Collisions



Chosen-Prefix Transcript Collisions




A
MitM
B
hash
len1
gx
blobA
len2'
gy'
blobB'
m1
m1'
Find Chosen-Prefix Collision C1, C2
N = 2^CPC(hash)
MD5: 2^39
SHA-1: 2^77
HMAC/96: n/a
hash
len1'
gx'
blobA'
len2
gy
blobB

SLOTH: Attacking TLS 1.2 Client Auth

- TLS 1.2 upgraded hash functions used in TLS
  - SHA-256 for all handshake constructions
  - New signature algorithms extension: SHA-256/384/512
- TLS 1.2 added support for MD5-based signatures!
  - Even if the client and server prefer **RSA-SHA256**, the connection can be downgraded to RSA-MD5!
- Transcript collisions break TLS 1.2 client signatures
  - Chosen prefix collision attack using flexible formats
  - **Demo:** Takes 1 hour/connection on a 48-core workstation
  - *Not very practical*: connection must be live during attack

SLOTH: Attacking TLS Server Auth

- TLS 1.2 server signatures are harder to break
  - *Irony*: the weakness that enables Logjam blocks SLOTH
  - Needs 2^X prior connections + 2^{128-X} hashes/connection
  - Not practical for academics, as far as we know
- TLS 1.3 server signatures is potentially vulnerable
  - *New*: MD5, SHA-1 sigs now explicitly forbidden in TLS 1.3

Other SLOTH Vulnerabilities

- Reduced security for TLS 1.*, IKEv1, IKEv2, SSH
  - Impersonation attack on TLS channel bindings
  - Exploit downgrades + transcript collisions
  - These are protocol flaws, not implementation bugs
  - Main mitigation is to disable weak hash functions

Protocol	Property	Mechanism	Attack	Collision Type	Precomp.	Work/conn.	Preimage	Wall-clock time
TLS 1.2	Client Auth	RSA-MD5	Impersonation	Chosen Prefix		2^{39}	2^{128}	48 core hours
TLS 1.3	Server Auth	RSA-MD5	Impersonation	Chosen Prefix		2^{39}	2^{128}	48 core hours
TLS 1.0-1.2	Channel Binding	HMAC (96 bits)	Impersonation	Generic		2^{48}	2^{96}	80 GPU days
TLS 1.2	Server Auth	RSA-MD5	Impersonation	Generic	2^X conn.	2^{128-X}	2^{128}	
TLS 1.0-1.1	Handshake Integrity	MD5 | SHA-1	Downgrade	Chosen Prefix		2^{77}	2^{160}	
IKE v1	Initiator Auth	HMAC-MD5	Impersonation	Generic		2^{65}	2^{128}	
IKE v2	Initiator Auth	RSA-SHA-1	Impersonation	Chosen Prefix	2^{77}	0	2^{160}	
SSH-2	Exchange Integrity	SHA-1	Downgrade	Chosen Prefix		2^{77}	2^{160}	

Final Thoughts

- Legacy crypto is strangely hard to get rid of,
  but we have to keep trying to kill broken primitives

  (MD5 MUST DIE)

- Key exchanges in Internet protocols ***do*** rely on collision resistance, question anyone who tells you otherwise!

- **Future**: new downgrade resilient protocols,
  collision-resistant authentication mechanisms

- More details, papers, demos are at:

  http://sloth-attack.org