Hash Functions

Hash Functions

- A hash function H provides a way to deterministically map a long input string to a shorter output string sometimes called a digest.
- The primary requirement is that it should be infeasible to find a collision in H: namely, two inputs that produce the same digest.
- A collision is a pair of distinct elements x and x^0 for which $H(x) = H(x^0)$;

Collision Resistance

- A function H is collision resistant if it is infeasible for any probabilistic polynomial-time algorithm to find a collision in H

DEFINITION 6.1 *A* hash function *(with output length $\ell(n)$) is a pair of probabilistic polynomial-time algorithms* (Gen, H) *satisfying the following:*

- Gen *is a probabilistic algorithm that takes as input a security parameter 1^n and outputs a key s. We assume that n is implicit in s.*
- *H is a deterministic algorithm that takes as input a key s and a string $x \in \{0,1\}^*$ and outputs a string $H^s(x) \in \{0,1\}^{\ell(n)}$ (where n is the value of the security parameter implicit in s).*

If H^s is defined only for inputs x of length $\ell'(n) > \ell(n)$, then we say that (Gen, H) *is a* fixed-length hash function for inputs of length $\ell'(n)$. *In this case, we also call H a* compression function.

Collision-finding experiment

The collision-finding experiment $\mathsf{Hash\text{-}coll}_{\mathcal{A},\mathcal{H}}(n)$:

1. *A key s is generated by running* $\mathsf{Gen}(1^n)$.
2. *The adversary $\mathcal{A}$ is given s, and outputs x, x'. (If $\mathcal{H}$ is a fixed-length hash function for inputs of length $\ell'(n)$, then we require $x, x' \in \{0,1\}^{\ell'(n)}$.)*
3. *The output of the experiment is defined to be* 1 *if and only if $x \neq x'$ and $H^s(x) = H^s(x')$. In such a case we say that $\mathcal{A}$ has found a collision.*

The definition of collision resistance states that no efficient adversary can find a collision in the above experiment except with negligible probability.

Collision resistant

DEFINITION 6.2 *A hash function* $\mathcal{H} = (\mathsf{Gen}, H)$ *is* collision resistant *if for all probabilistic polynomial-time adversaries* $\mathcal{A}$ *there is a negligible function* negl *such that*

$$\Pr\left[\mathsf{Hash\text{-}coll}_{\mathcal{A},\mathcal{H}}(n) = 1\right] \leq \mathsf{negl}(n).$$

For simplicity, we sometimes refer to H or H^s as a "collision-resistant hash

Unkeyed hash functions

- Cryptographic hash functions used in practice are generally unkeyed and have a fixed output length
- The hash function is just a fixed, deterministic function $H : \{0,1\}^* \rightarrow \{0,1\}^\ell$.

Requirements for Hash Functions

1. can be applied to any size message M
2. produces a fixed-length output h
3. is easy to compute $h=H(M)$ for any message M
4. given h is infeasible to find x s.t. $H(x)=h$
   - one-way property
5. given x is infeasible to find y s.t. $H(y)=H(x)$
   - weak collision resistance
6. is infeasible to find any x, y s.t. $H(y)=H(x)$
   - strong collision resistance

Notions of security

- *Second-preimage resistance:* Informally, a hash function is said to be second-preimage resistant if given s and a uniform x it is infeasible for a PPT adversary to find $x' \neq x$ such that $H^s(x') = H^s(x)$.

- *Preimage resistance:* Informally, a hash function is preimage resistant if given s and $y = H^s(x)$ for a uniform x, it is infeasible for a PPT adversary to find a value x' (whether equal to x or not) with $H^s(x') = y$.

“Birthday” attacks

- Compute $H(x_1), \ldots, H(x_k)$
  - What is the probability of a collision?

- Related to the so-called *birthday paradox*
  - How many people are needed to have a 50% chance that some two people share a birthday?

Message Authentication Using Hash Functions

- Hash-and-MAC
  - Collision-resistant hash functions can be used for message authentication codes
- We can authenticate an arbitrary-length message m by using the MAC to authenticate the hash of m

Hash-and-MAC

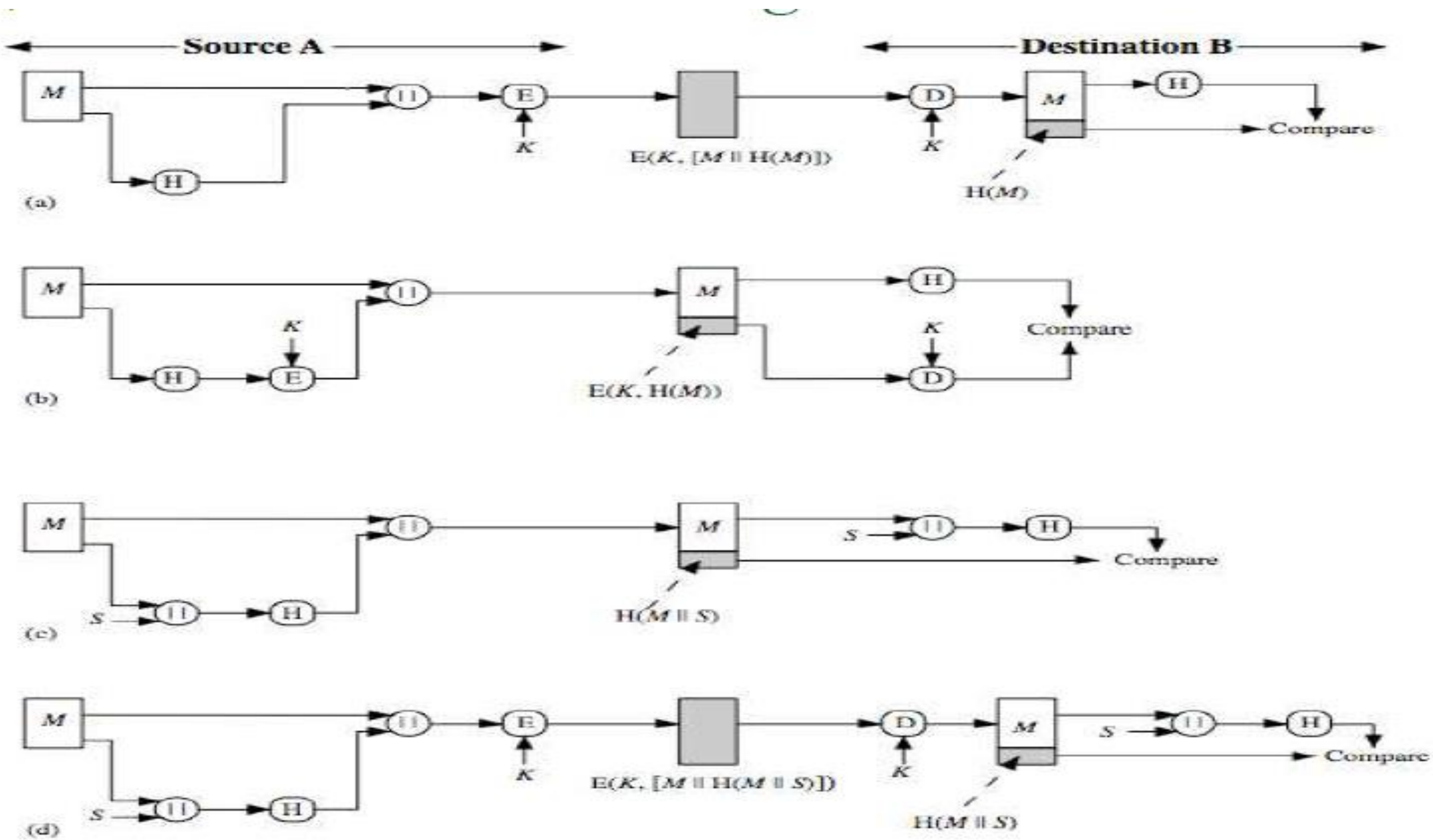
CONSTRUCTION 6.5

Let $\Pi = (\mathsf{Mac}, \mathsf{Vrfy})$ be a MAC for messages of length $\ell(n)$, and let $\mathcal{H} = (\mathsf{Gen}_H, H)$ be a hash function with output length $\ell(n)$. Construct a MAC $\Pi' = (\mathsf{Gen}', \mathsf{Mac}', \mathsf{Vrfy}')$ for arbitrary-length messages as follows:

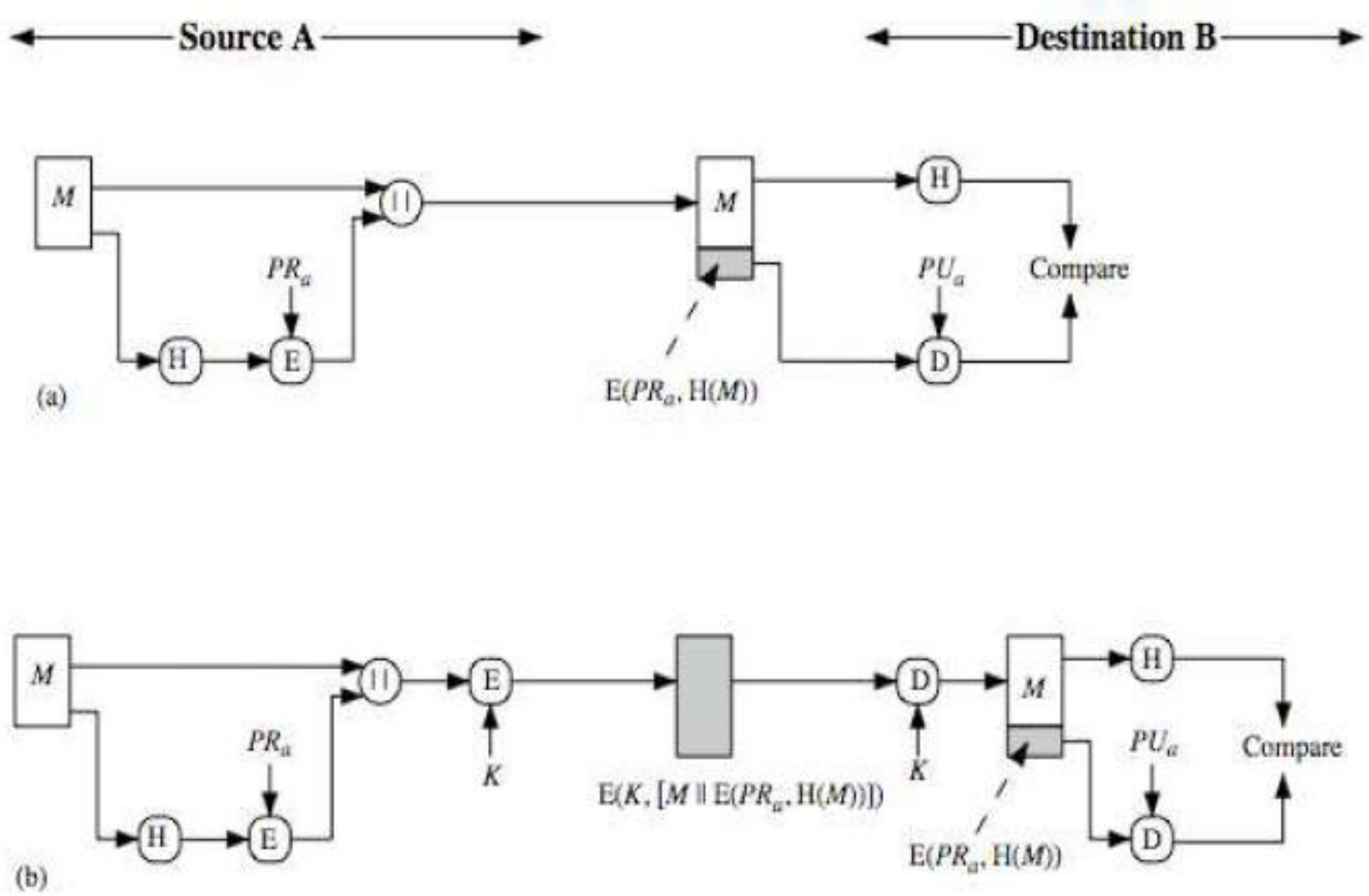
- Gen': on input 1^n, choose uniform $k \in \{0,1\}^n$ and run $\mathsf{Gen}_H(1^n)$ to obtain s; output the key (k, s).
- Mac': on input a key (k, s) and a message $m \in \{0,1\}^*$, output $t \leftarrow \mathsf{Mac}_k(H^s(m))$.
- Vrfy': on input a key (k, s), a message $m \in \{0,1\}^*$, and a tag t, output 1 if and only if $\mathsf{Vrfy}_k(H^s(m), t) \stackrel{?}{=} 1$.

The hash-and-MAC paradigm.

Hash Functions & Message Authentication



Hash Functions & Digital Signatures



Summary

Discussed about

- Hash function
- Requirements of hash function
- MAC and Message encryption