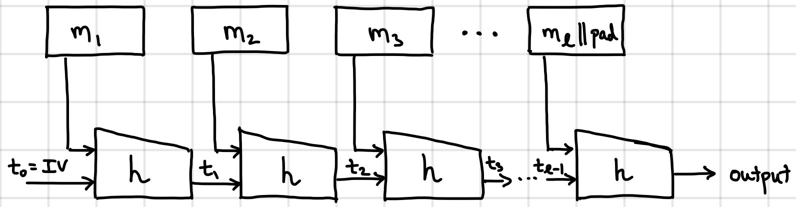
Constructing CRHFs:

Many cryptographic hash functions (e.g., MD5, SHA-1, SHA-256) follow the Merkle-Damgård paradigm: start from hash function on <u>short</u> messages and use it to build a collision-resistant hash function on a long message:

1. Split message into blocks
2. Iteratively apply <u>compression function</u> (hash function on short inputs) to message blocks



h: compression function

$t_0, \dots, t_\ell$: chaining variables

padding introduced so last block is multiple of block size

↳ <u>must</u> also include an encoding of the message length: typically of the form $100\cdots0 \| \langle s \rangle$ where $\langle s \rangle$ is a <u>fixed-length</u> binary representation of message length in blocks

<u>Recall</u>: $100\cdots0$ padding was used in the ANSI standard

if not enough space to include the length, then extra block is <u>added</u> (similar to CBC encryption)

Hash functions are <u>deterministic</u>, so IV is a fixed string (defined in the specification) — can be taken to be all-zeroes string, but usually set to a custom value in constructions

for SHA-256:
$\mathcal{X} = \{0,1\}^{256} = \mathcal{Y}$

<u>Theorem</u>. Suppose $h : \mathcal{X} \times \mathcal{Y} \to \mathcal{X}$ be a compression function. Let $H : \mathcal{Y}^{\leq \ell} \to \mathcal{X}$ be the Merkle-Damgård hash function constructed from h. Then, if h is collision-resistant, H is also collision-resistant.

<u>Proof</u>. Suppose we have a collision-finding algorithm A for H. We use A to build a collision-finding algorithm for h:

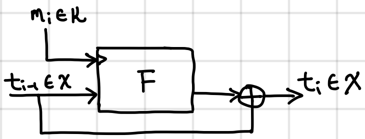
1. Run A to obtain a collision M and M' ($H(M) = H(M')$ and $M \neq M'$).
2. Let $M = m_1 m_2 \cdots m_u$ and $M' = m_1' m_2' \cdots m_v'$ be the blocks of M and M', respectively. Let $t_0, t_1, \dots, t_u$ and $t_1' t_2' \cdots t_v'$ be the corresponding chaining variables.
3. Since $H(M) = H(M')$, it must be the case that
$$H(M) = h(t_{u-1}, m_u) = h(t_{v-1}', m_v') = H(M')$$
If either $t_{u-1} \neq t_{v-1}'$ or $m_u \neq m_v'$, then we have a collision for h.
Otherwise, $m_u = m_v'$ and $t_{u-1} = t_{v-1}'$. Since m_u and m_v' include an encoding of the length of M and M', it must be the case that $u = v$. Now, consider the second-to-last block in the construction (with output $t_{u-1} = t_{u-1}'$):
$$t_{u-1} = h(t_{u-2}, m_{u-1}) = h(t_{u-2}', m_{u-1}') = t_{u-1}'$$
Either we have a collision or $t_{u-2} = t_{u-2}'$ and $m_{u-1} = m_{u-1}'$. Repeat down the chain until we have collision or we have concluded that $m_i = m_i'$ for all i, and so $M = M'$, which is a contradiction.

<u>Note</u>: Above constructing is <u>sequential</u>. Easy to adapt construction (using a <u>tree</u>) to obtain a parallelizable construction.

Sufficient now to construct a <u>compression function</u>.

Typical approach is to use a block cipher.

<u>Davies-Meyer</u>: Let $F: \mathcal{K} \times \mathcal{X} \to \mathcal{X}$ be a block cipher. The Davies-Meyer compression function $h: \mathcal{K} \times \mathcal{X} \to \mathcal{X}$ is then



$$h(k, x) := F(k, x) \oplus x$$

Many other variants also possible: $h(k,x) = F(k,x) \oplus k \oplus x$ [used in Whirlpool hash family]

Need to be careful with design!

- $h(k,x) = F(k,x)$ is <u>not</u> collision-resistant: $h(k,x) = h(k', F^{-1}(k', F(k,x)))$
- $h(k,x) = F(k,x) \oplus k$ is <u>not</u> collision-resistant: $h(k,x) = h(k', F^{-1}(k', F(k,x) \oplus k \oplus k'))$

<u>Theorem.</u> If we model F as an ideal block cipher (i.e., a truly random permutation for every choice of key), then Davies-Meyer is collision-resistant.

<u>Conclusion</u>: Block cipher + Davies-Meyer + Merkle-Damgård $\Longrightarrow$ CRHFs

<u>Examples</u>: SHA-1: SHACAL-1 block cipher with Davies-Meyer + Merkle-Damgård $\leftarrow$ no longer secure [first collision found in 2017!]

birthday attack run-time: $\sim 2^{80}$
attack ran in time $\sim 2^{64}$ (100,000x faster)

January, 2020: chosen-prefix collision in $\sim 2^{63.4}$ time!

SHA-256: SHACAL-2 block cipher with Davies-Meyer + Merkle-Damgård

- SHA-1 extensively used (e.g., git, svn, software updates, PGP/GPG signatures, certificates) $\longrightarrow$ attacks show need to transition to SHA-2 or SHA-3

Why not use AES?

- Block size too small! AES outputs are 128-bits, not 256 bits (so birthday attack finds collision in 2^{64} time)
- Short keys means small number of message bits processed per iteration.
- Typically, block cipher designed to be fast when using <u>same</u> key to encrypt many messages
  ↳ In Merkle-Damgård, <u>different</u> keys are used, so alternate design preferred (AES key schedule is <u>expensive</u>)

<u>Recently</u>: SHA-3 family of hash functions standardized (2015)

↳ Relies on different underlying structure ("sponge" function)

↳ Both SHA-2 and SHA-3 are believed to be secure (most systems use SHA-2 – typically much faster)

Back to building a secure MAC from a CRHF — can we do it more <u>directly</u> than using CRHF + small-domain MAC? (or even better, a large-domain PRF)

↳ Main difficulty seems to be that CRHFs are <u>keyless</u> but MACs are <u>keyed</u>

<u>Idea</u>: include the key as part of the hashed input

By itself, collision-resistance does not provide any "randomness" guarantees on the output

↳ For instance, if H is collision-resistant, then $H'(m) = m_0 \| \cdots \| m_{10} \| H(m)$ is also collision-resistant even though H' also <u>leaks</u> the first 10 bits/blocks of m

↳ Constructing a PRF/MAC from a hash function will require more than just collision resistance

- <u>Option 1</u>: Model hash function as an "ideal hash function" that behaves like a fixed <u>truly random</u> function (modeling <u>heuristic</u> called the random oracle model — will encounter later in this course)
- <u>Option 2</u>: Start with a concrete construction of a CRHF (e.g., Merkle-Damgård or the sponge construction) and reason about its properties
  ↳ We will take this approach

How long does the output of a CRHF have to be?

__Birthday attack on CRHFs.__ Suppose we have a hash function $H : \{0,1\}^n \to \{0,1\}^\ell$. How might we find a collision in H (without knowing anything more about H)

__Approach 1:__ Compute $H(1), H(2), \ldots, H(2^\ell + 1)$ ← size of hash output space

↳ By Pigeonhole Principle, there must be at least __one__ collision — runs in time $O(2^\ell)$

__Approach 2:__ Sample $m_i \xleftarrow{R} \{0,1\}^n$ and compute $H(m_i)$. Repeat until collision is found.

How many samples needed to find a collision?

__Theorem (Birthday Paradox).__ Take any set S where $|S| = n$. Suppose $r_1, \ldots, r_\ell \xleftarrow{R} S$. Then,

$$\Pr[\exists i \neq j : r_i = r_j] \geq 1 - e^{-\frac{\ell(\ell-1)}{2n}}$$

__Proof.__

$$\begin{aligned}
\Pr[\exists i \neq j : r_i = r_j] &= 1 - \Pr[\forall i \neq j : r_i \neq r_j] \\
&= 1 - \Pr[r_2 \notin \{r_1\}] \cdot \Pr[r_3 \notin \{r_2, r_1\}] \cdot \cdots \cdot \Pr[r_\ell \notin \{r_{\ell-1}, \ldots, r_1\}] \\
&= 1 - \frac{n-1}{n} \cdot \frac{n-2}{n} \cdot \cdots \cdot \frac{n-\ell+1}{n} \\
&= 1 - \prod_{i=1}^{\ell-1} \left(1 - \frac{i}{n}\right) \\
&\geq 1 - \prod_{i=1}^{\ell-1} e^{-i/n} \quad \text{since } 1 + x \leq e^x \text{ for all } x \in \mathbb{R} \quad \Big[e^x = 1 + x + \underbrace{\frac{x^2}{2} + \frac{x^3}{6} + \cdots}_{\text{positive for all } x>0} \Big] \\
&= 1 - e^{\sum_{i=1}^{\ell-1} -i/n} = 1 - e^{-\frac{1}{n}\sum_{i=1}^{\ell-1} i} \\
&= 1 - e^{-\frac{(\ell-1)\ell}{2n}}
\end{aligned}$$

automatically holds for $x \leq -1$

dominant term when $|x| < 1$

When $\ell \geq 1.2\sqrt{n}$, $\Pr[\text{collision}] = \Pr[\exists i \neq j : r_i = r_j] > \frac{1}{2}$. [For birthdays, $1.2\sqrt{365} \approx 23$] ← number of people in a room to have a common birthday

↳ Birthdays not uniformly distributed, but this only __increases__ collision probability. [Try proving this!]

For hash functions with range $\{0,1\}^\ell$, we can use a birthday attack to find collisions in time $\sqrt{2^\ell} = 2^{\ell/2}$ — can even do it with __constant__ space! [via Floyd's cycle finding algorithm]

↳ For 128-bit security (e.g., 2^{128}), we need the output to be 256-bits (hence __SHA-256__)

↳ Quantum collision-finding can be done in $2^{\ell/3}$ (cube root attack), though requires more space