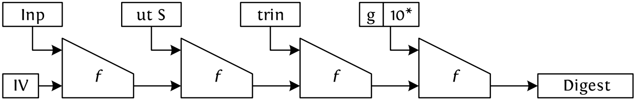
The iterating mode

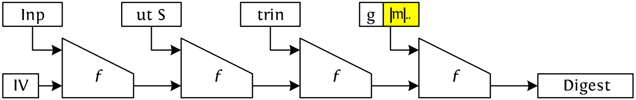
Basic Merkle-Damgård: very simple and elegant



Yes, but can we have collision-resistance preservation?

The iterating mode

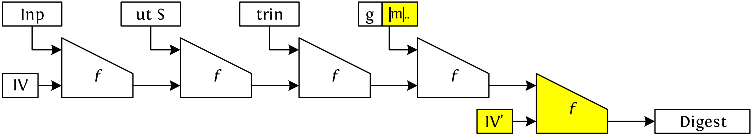
Merkle-Damgård with *strengthening*



Yes, but what about length extension attacks and the like?

The iterating mode

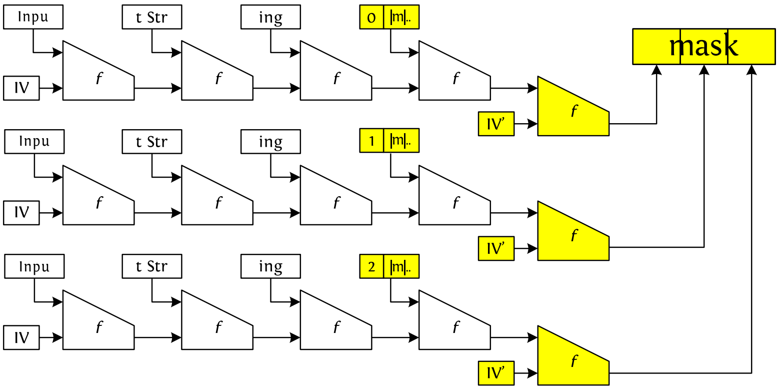
Enveloped Merkle-Damgård



Yes, but we need long output for full-domain hashing (OAEP, RSA-PSS, KEM, etc)?

The iterating mode

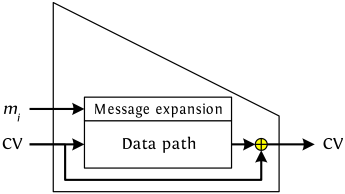
Mask generating function construction



This does what we need!

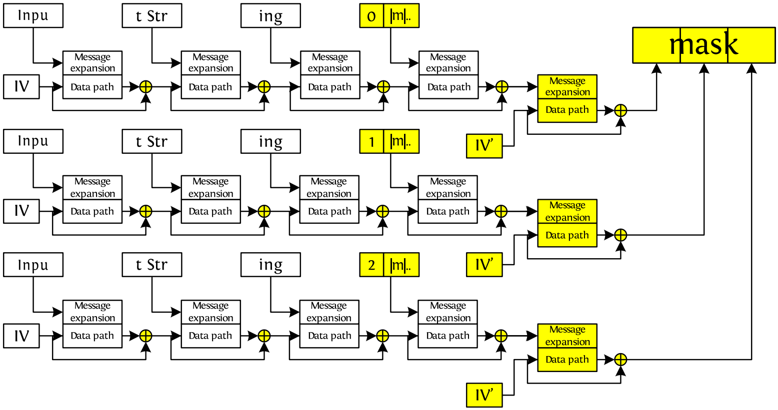
The compression function

Block cipher in Davies-Meyer mode



That's it!

The final solution

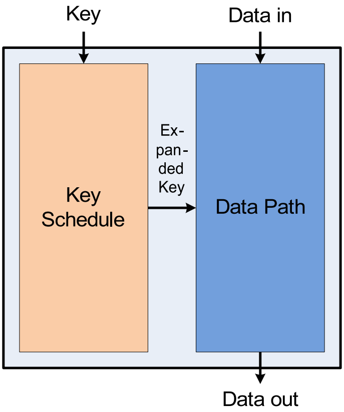


Now we just have to build a suitable block cipher …

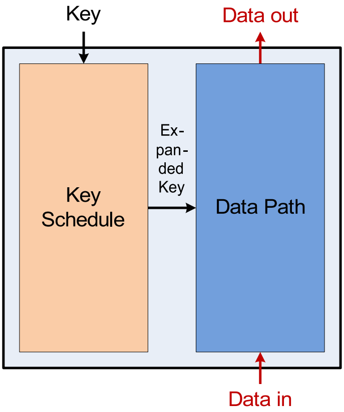
Block-cipher based hashing: time for re-factoring

- Goal: hashing mode that is sound and simple
  - with good level of security against generic attacks
  - calling a block cipher
- Remaining problem: design of a suitable block cipher
  - round function: several good approaches known
  - soundness proofs are typically in *ideal cipher* model
  - key schedule: not clear how to do design good one
- But do we really need a block cipher?

Block cipher operation



Block cipher operation: the inverse



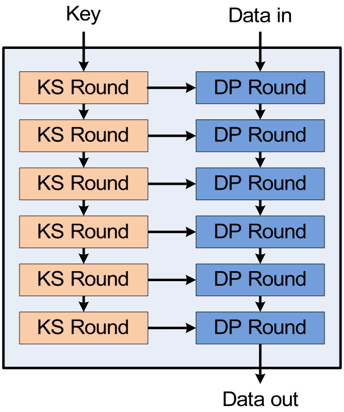
When do you need the inverse?

Indicated in red:

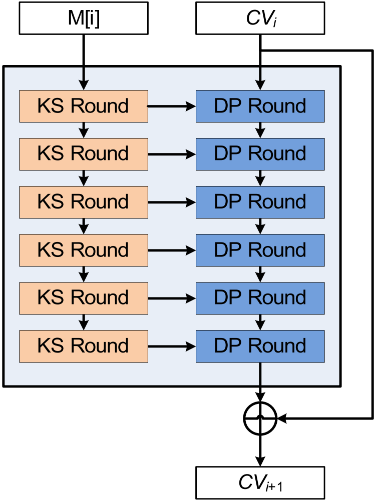
- Hashing and its modes HMAC, MGF1, ...
- Block encryption: ECB, CBC, ...
- Stream encryption:
  - synchronous: counter mode, OFB, ...
  - self-synchronizing: CFB
- MAC computation: CBC-MAC, C-MAC, ...
- Authenticated encryption: OCB, GCM, CCM ...
  - Most schemes with misuse-resistant claims

So for most uses you don't need the inverse!

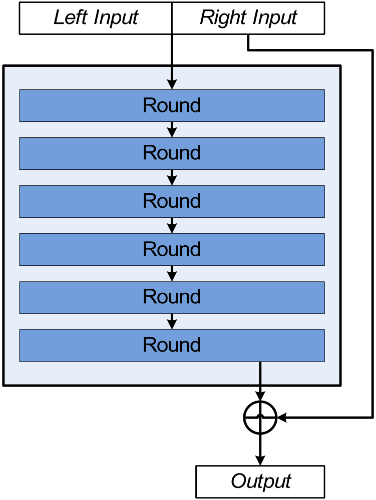
Block cipher internals



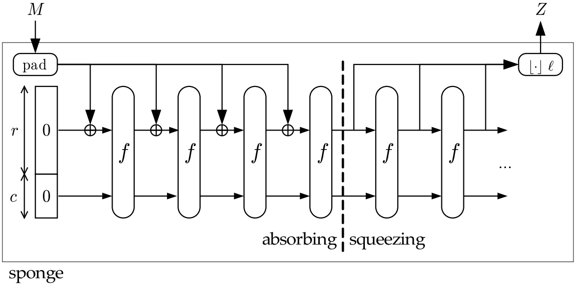
Hashing use case: Davies-Meyer compression function



Simplifying the view: iterated permutation



The result: the sponge construction



- f: a b-bit permutation with $b = r + c$
  - efficiency: processes r bits per call to f
  - security: provably resists generic attacks up to $2^{c/2}$
- Flexibility in trading rate r for capacity c or vice versa

Security strength oriented approach

Security strength	Collision resistance	Pre-image resistance	Required capacity	Relative perf.	SHA-3 instance
$s = 80$	$n \geq 160$	$n \geq 80$	$c = 160$	$\times 1.406$	**SHA3c160**
$s = 112$	$n \geq 224$	$n \geq 112$	$c = 224$	$\times 1.343$	**SHA3c224**
$s = 128$	$n \geq 256$	$n \geq 128$	$c = 256$	$\times 1.312$	**SHA3c256**
$s = 192$	$n \geq 384$	$n \geq 192$	$c = 384$	$\times 1.188$	**SHA3c384**
$s = 256$	$n \geq 512$	$n \geq 256$	$c = 512$	$\times 1.063$	**SHA3c512**
s	$n \geq 2s$	$n \geq s$	$c = 2s$	$\times \frac{1600-c}{1024}$	**SHA3[c=2s]**

s: security strength level [NIST SP 800-57]

- These SHA-3 instances
  - are consistent with philosophy of [NIST SP 800-57]
  - provide a one-to-one mapping to security strength levels
- Higher efficiency

Generic security of the sponge construction

Theorem (Indifferentiability of the sponge construction)

The sponge construction calling a random permutation, $\mathcal{S}'[\mathcal{F}]$, is (t_D, t_S, N, ϵ)-indifferentiable from a random oracle, for any t_D, $t_S = O(N^2)$, $N < 2^c$ and for any ϵ with $\epsilon > f_P(N) \approx \frac{N}{2^{c+1}}$.

[Keccak team, Eurocrypt 2008]

Informally, a random sponge is like a random oracle when $N < 2^{c/2}$.

- Collision-, preimage-resistance, etc., up to security strength $c/2$
- The bound assumes f is a random permutation
  - It covers generic attacks
  - …but not attacks that exploit specific properties of f