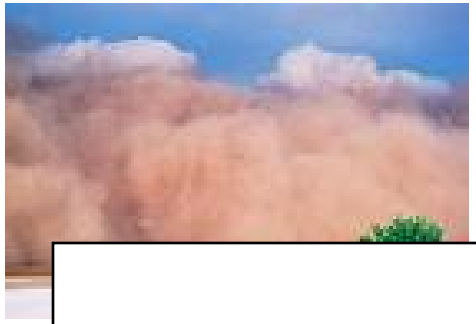


The SANDstorm Hashing Function

Rich Schroeppel, Mark Torgerson, Tim Draelos, Hilarie Orman*,
Mike Collins, Nathan Dautenhahn, Andrea Walker, Sean Malone

Sandia National Laboratories

*Purple Streak Inc

February 2009

Sandia is a multiprogram laboratory operated by Sandia Corporation,
a Lockheed Martin Company, for the United States Department of Energy's
National Nuclear Security Administration under Contract DE-AC04-94AL85000.

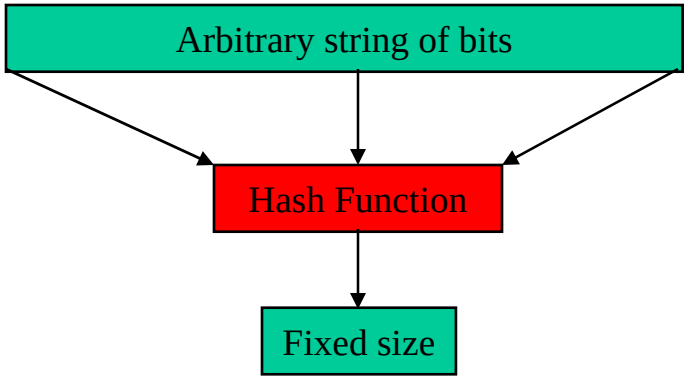
Sandia National Laboratories

Hash Basics

- A Hash Function takes as input an arbitrary string of bits, processes it, and outputs a fixed size digest.

- Used for data integrity checks
- Authenticity of data

- Generic Hashes include
  - Checksums
  - CRCs
  - Cryptographic Hashes



Cryptographic Hash Basics

- A Cryptographic Hashing Function is used for security and has additional properties
  - Collision Resistance
    - Hard to find M and M' so that H(M)=H(M')
  - Second Preimage Resistance
    - Given M and H(M) it is hard to find M' so that H(M)=H(M')
  - Preimage Resistance
    - Given h it is hard to find M so that H(M)=h
- Resistance is based on computational difficulty which is limited by size of output.

Cryptographic Hash Basics

- Given output size n, security goals are
  - Collision requires $2^{n/2}$ operations
  - Second Preimage requires 2^{n} operations
  - Preimage requires 2^{n} operations

Common Hash	Output size
MD5	128
SHA-1	160
SHA-2	224, 256, 384, 512
SANDstorm	224, 256, 384, 512

- MD5 broken
  - Collisions are routine
- SHA-1 theoretically broken
  - Too small for high security
- SHA-2 family is a MD5/SHA-1 derivative

Government and crypto community need an alternative to SHA-2

SANDstorm Basic Features

- Four Members of the Family
- Plug and play with SHA-2 Family

- Truncated Tree Based Mode
- Highly Modified Merkle-Damgard Chaining
- Simple Padding
- Finishing Step

Bits	Word Size	Block Size	Message Size
SANDstorm-224	64	512	2^{128}
SANDstorm-256	64	512	2^{128}
SANDstorm-384	128	1024	2^{128}
SANDstorm-512	128	1024	2^{128}

Strength First with Defense in Depth

Many Desirable Functional and Security Features

SANDstorm Special Features

- Secure Chaining—4x output size state variables
  - Chaining method is Provably Secure against “long message” attacks
  - Resistance to multicollisions, herding etc.
  - Tree forwarding uses 2x size state variables.

Factor of 10K speed-up possible

- Serious Commitment to Parallelism
  - Mode is parallelizable/pipelineable
  - Chaining method is parallelizable/pipelineable
  - Compression function is parallelizable/pipelineable

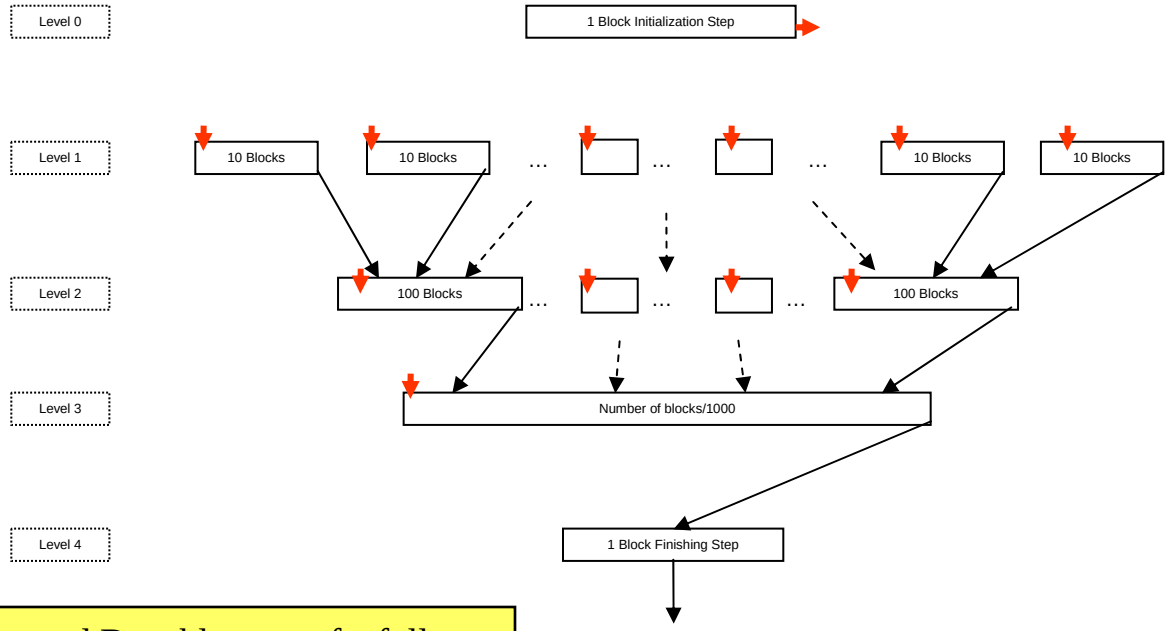
Flexibility in where to apply resources

- Friendly to small changes in large messages
  - Precomputation is an option
- Finishing step prevents length extension attacks

SANDstorm Mode

- Initial & Final compression steps
- 0 - 3 intermediate levels, of 10, 100, and unbounded number of blocks.
  - Early Out when intermediate levels not needed
    - A one block message will call Compress only twice

- Factor of 1000 speed up
- One mode
- 11% serial overhead
- **First block permeates subsequent compressions**
- Double-sized state **passed to next level**



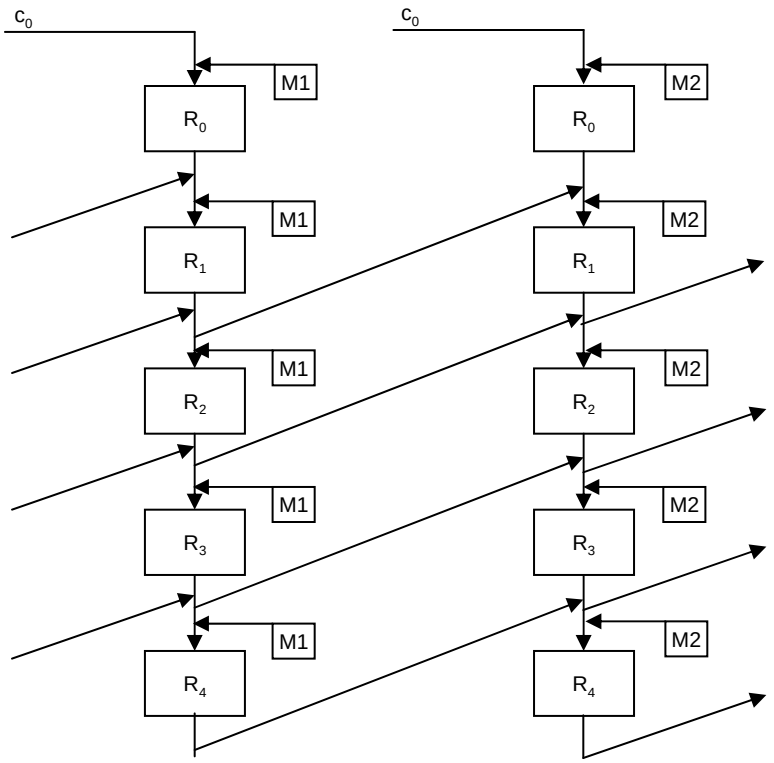
Substantial Parallelism without the Latency and Data blow up of a full tree

SANDstorm Chaining

- 5 rounds, 256-bit state (512 for SS-384/512)
- Provably Secure against long message attacks
- 4x Output State
  - Thwart herding
  - Thwart multicollisions
- Pipeline Friendly
- Reuses SHA-2 constants

There is a lot of mixing distance between the first and the second message input

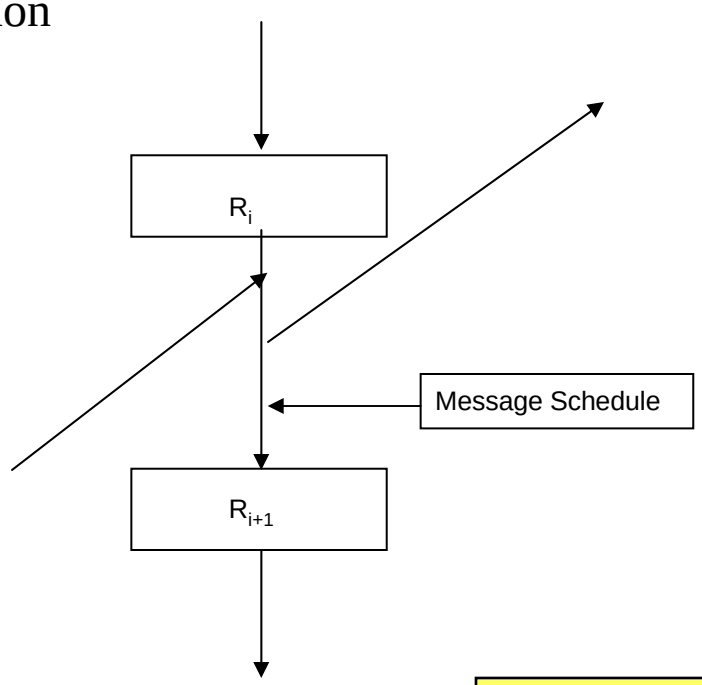
Adjacent message inputs collide iff messages collide



Sandia National Laboratories

SANDstorm Details

- Message Schedule is ~60% of the work
  - Processed independently of round function
  - Brought in all at once
- Round Functions can be pipelined
- Arithmetic
  - Uses Multiply
    - Superb Mixing
    - Kills Differentials
  - Largely parallelizable
  - Minor AES sbox usage
  - Interleaving arithmetic & logic ops
- Tunable Security Parameter



Defense in depth

Message contribution is brought in all at once

SANDstorm Arithmetic

- $Z \Rightarrow X*2^{32} + Y$

- $F(Z) = X^2 + Y^2 \mod 2^{64}$
- $G(Z) = X^2 + Y^2 + ((X+a)(Y+b)<<<32) \mod 2^{64}$

Multiplication has a lot of mixing per unit work

- $Ch(A,B,C) = (A\&B) \oplus (\neg A\&C)$ (choose)
- SB(Z) = Z with low order byte mapped with the AES sbox
- BitMix shuffles bits between 4 words

Defense in Depth

SANDstorm Bit Mix

- J_8 = 8888888888888888
- J_4 = 4444444444444444
- J_2 = 2222222222222222
- J_1 = 1111111111111111

- If A, B, C, D are all 64 bits in length, then

(A', B', C', D') = BitMix(A, B, C, D), where

$A' = (J_8 \,\&\, A) \oplus (J_4 \,\&\, B) \oplus (J_2 \,\&\, C) \oplus (J_1 \,\&\, D)$

$B' = (J_8 \,\&\, B) \oplus (J_4 \,\&\, C) \oplus (J_2 \,\&\, D) \oplus (J_1 \,\&\, A)$

$C' = (J_8 \,\&\, C) \oplus (J_4 \,\&\, D) \oplus (J_2 \,\&\, A) \oplus (J_1 \,\&\, B)$

$D' = (J_8 \,\&\, D) \oplus (J_4 \,\&\, A) \oplus (J_2 \,\&\, B) \oplus (J_1 \,\&\, C)$

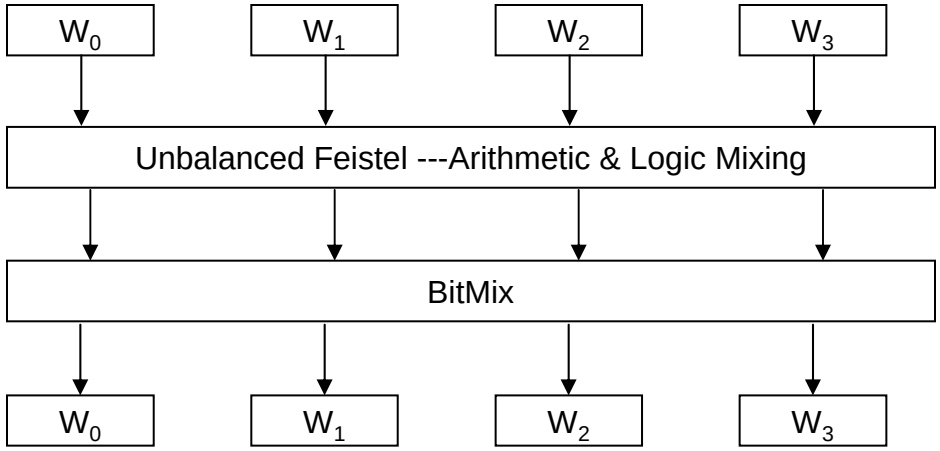
SANDstorm Round Function

- For i from 0 to 3

  $W_i = SB\ (W_i + F(W_{i-1}) + Ch(W_{i-1},W_{i-2},W_{i-3}) + A(r,i) \mod 2^{64}) <<<25$

- $BitMix(W_0, W_1, W_2, W_3)$

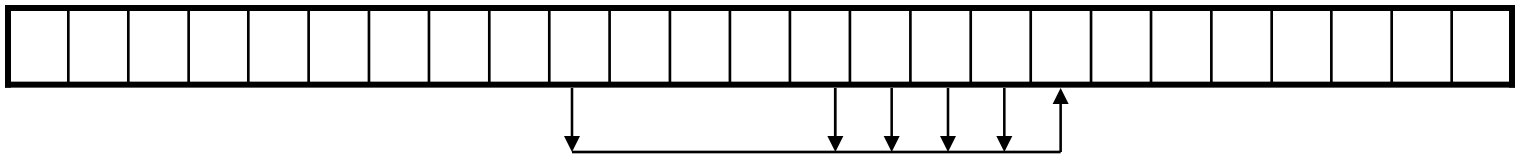
- A(r,i) are the SHA constants



Each byte of output is a function of each and every bit of input

About 40% the work of compress is in the round function

SANDstorm Message Schedule



- For i from 8 to 32

  $d_i = SB(d_{i-8} + G(d_{i-1}) + Ch(d_{i-1}, d_{i-2}, d_{i-3}) + d_{i-4} + B_i \mod 2^{64}) <<< 27$

- Every bit of d_{i+12} is a function of every bit of $d_i \ldots d_{i+7}$
  - A couple weak relationships of d_{i+3}
  - Full mix after 5 steps
  - 25 steps

Multiplication kills differentials

$MS(0,D) = BitMix(ROTL^{19}(d_0) \oplus d_4, ROTL^{19}(d_1) \oplus d_5, ROTL^{19}(d_2) \oplus d_6, ROTL^{19}(d_3) \oplus d_7)$

$MS(1,D) = (d_{14}, d_{15}, d_{16}, d_{17})$

$MS(2,D) = (d_{19}, d_{20}, d_{21}, d_{22})$

$MS(3,D) = (d_{24}, d_{25}, d_{26}, d_{27})$

$MS(4,D) = (d_{29}, d_{30}, d_{31}, d_{32})$

Each entering bit is a function of ALL schedule bits of all 8 words 5 steps back

Sandia National Laboratories

SANDstorm Performance

- Cost, in cycles/byte for one compression operation
  - The assembly version just makes a 32X32→64bit multiply stay that way
  - No optimization effort put into the -384,-512 implementations

	32-bit Machine		64-bit Machine
	Optimized	Assembly	Optimized
SANDstorm -224, -256	71.9	62.5	36.6
SHA-1	18.8		14.5
SHA-256	40.65		39.1
SANDstorm-384, -512	297		95

Reasonably fast serial. Lots of opportunity to parallelize and/or pipeline

SANDstorm Feature Summary

64-bit design (128-bit for SS-384/512)
512-bit blocksize (1024-bit for SS-384/512)
Brick construction – ideally, deleting any single feature is still secure
Multiplication is the best mixer
fewer rounds, kills differentials, but slow C code
Reuse SHA-2 round constants; minor use of AES sbox
Interleave arithmetic & bit mixing operations
Don't dribble in the message – smash it in
60% of the work done in the message schedule -- beefed up, ultra-nonlinear
Serious commitment to parallelism: no separate mode
One tunable security parameter
Ultra-wide pipe: 4x the output size – more intimate block chaining
Double-sized state forwarded to next tree level
Ubiquitous block numbers
Extra compression step for final output

SANDstorm Parallelism

One hash function, with minimum options. (Output size, TSP.)

No separate mode for parallelism.

The mode allows parallelism up to 1100x.

The message schedule can be precomputed (60% of the work).

The round chaining can be pipelined.

Within a round, the arithmetic can be largely parallel.

If the first block is fixed:

A message can be assembled from fixed and variable pieces, and most of the hash for the fixed pieces can be precomputed. Example: A movie plus a variable wrapper.

Why Multiplication?

- Multiplication is far and away the best mixer
- Multiplication kills differentials
- Xor: Each bit depends* linearly on two bits
- Add: Each bit depends linearly on two bits, and non-linearly on two bits
- 32*32 -> 64 Multiply: Each bit on average depends non-linearly on 32 bits
- Cost is only about 3 clocks
- Every operation has substantial overhead gathering inputs and storing results. It makes sense to do as much mixing work as possible per operation
- Another hardware parallelism opportunity

* depends: >= 20% chance that a bit flip changes the output

- Drawback: Hard to express efficiently in C language.

The Future is Parallel

- Moore’s law is moving sideways
- Clock speeds have stalled
- Most submissions don’t have a parallel mode, and should be considered incomplete
- Separate serial and parallel standards is not a **standard**
- SANDstorm has a details-filled-in tree specification, including double-sized state forwarding between levels

Simplicity is Dangerous

- Defense in depth
- Let’s do this once, and get it over with
- We’ve put our multiplication advantage into better mixing

QUESTIONS?

Sandia National Laboratories