CubeHash

http://cubehash.cr.yp.to

D. J. Bernstein
University of Illinois at Chicago

NSF ITR–0716498

CubeHash is the *smallest* high-security SHA-3 proposal.

Several meanings of “smallest”:

- Smallest memory use.
- Smallest description.
- Smallest code size.
- Smallest vector-code size.
- Smallest area in hardware.

Also good security/speed tradeoff. Default parameters are extremely conservative, but faster-than-MD5 parameters are still resisting all attacks.

Memory use

CubeHash-512 (512-bit output) fits into 128 bytes of RAM.

Closest competitor:
Keccak; 200 bytes of RAM.

Skein-512: 224 bytes of RAM?

SHA-512: 256 bytes of RAM (plus counter if necessary).
64 for current state,
64 for beginning-of-block state,
128 for transformed block.

Most submissions: Much larger.

CubeHash b-byte message block is xor'ed into first b state bytes. Does *not* take extra space.

Standard generic attacks: $2^{512-4b+\text{overhead}}$ bit operations to find preimages etc.

Default parameter: $b = 1$; $2^{508+\text{overhead}}$ bit operations.

$b = 8$ is $\approx 8\times$ faster; $2^{480+\text{overhead}}$ bit operations.

$b = 32$ is $\approx 32\times$ faster; $2^{384+\text{overhead}}$ bit operations.

Attack details: See submission.

Code size

After b-byte message block, CubeHashr/b transforms state (as 32 little-endian 4-byte words) through r identical rounds.

Finalization: Flip one bit; $10r$ extra identical rounds; output first 64 state bytes.

Initialization: Similarly easy.

SHA-3 proposal: CubeHash8/1; 8 rounds after each byte.

Faster: CubeHash8/2, CubeHash8/4, CubeHash8/8, etc.

First half of a round:

```
for (i = 0;i < 16;++i)
  x[i + 16] += x[i];
for (i = 0;i < 16;++i)
  y[i ^ 8] = x[i];
for (i = 0;i < 16;++i)
  x[i] = ROTATE(y[i],7);
for (i = 0;i < 16;++i)
  x[i] ^= x[i + 16];
for (i = 0;i < 16;++i)
  y[i ^ 2] = x[i + 16];
for (i = 0;i < 16;++i)
  x[i + 16] = y[i];
```

Second half of a round:

```
for (i = 0;i < 16;++i)
  x[i + 16] += x[i];
for (i = 0;i < 16;++i)
  y[i ^ 4] = x[i];
for (i = 0;i < 16;++i)
  x[i] = ROTATE(y[i],11);
for (i = 0;i < 16;++i)
  x[i] ^= x[i + 16];
for (i = 0;i < 16;++i)
  y[i ^ 1] = x[i + 16];
for (i = 0;i < 16;++i)
  x[i + 16] = y[i];
```

Reduced-round cryptanalysis

Traditional confidence-building: see how many rounds survive third-party cryptanalysis.

Culmination of attacks by Aumasson, Brier, Dai, Khazaei, Meier, Naya-Plasencia, Peyrin: collision in CubeHash3/64; 2^{231} for CubeHash5/64.

Do you have an improved attack? `http://cubehash.cr.yp.to/prizes.html`

Speed: cycles/byte from eBASH

CubeHash8/32	CPU
4.42	64 Core i7 920
6.03	64 Phenom 9550
6.29	64 Core 2 Duo 6f6
6.44	32 Core 2 Duo 6f6
7.32	32 Phenom 9550
9.00	64 Atom 330
9.47	32 Atom 330

SHA-512	CPU
12.41	64 Core i7 920
9.92	64 Phenom 9550
13.09	64 Core 2 Duo 6f6
116.61	32 Core 2 Duo 6f6
20.31	32 Phenom 9550
17.58	64 Atom 330
59.92	32 Atom 330