SHA3
Past, Present, and Future

John Kelsey
NIST
CHES 2013
Overview

• Before the competition
• The competition
• Standardizing Keccak as SHA3
• What’s next?
Before the Competition
Hash functions appeared as an important idea at the dawn of modern public crypto.

Many ideas floating around to build hash functions from block ciphers (DES) or mathematical problems.

Ways to build hash functions from compression functions
  - Merkle-Damgaard

Ways to build compression functions from block ciphers
  - Davies-Meyer, MMO, etc.
Merkle-Damgaard

- Used in all widespread hash functions before 2004
  - MD4, MD5, RIPE-MD, RIPE-MD160, SHA0, SHA1, SHA2

Image from Wikipedia
The MD4 Family

- Rivest published MD4 in 1990
- 128-bit output
- Built on 32-bit word operations
- Add, Rotate, XOR, bitwise logical operations
- Fast
- First widely used dedicated hash function
- 48 steps = 3 passes over msg

Image from Wikipedia MD4 Article
MD5

► Several researchers came up with attacks on weakened versions of MD4
► Rivest created stronger function in 1992
► Still very fast
► Same output size
► Some attacks known
  ► Den Boer/Bosselaers
  ► Dobbertin
► 64 steps = 4 passes over msg
SHA-0 and SHA-1

- SHA-0 published in 1993
- 160-bit output
  - (80 bit security)
- NSA design
- Revised in 1995 to SHA-1
  - Round function (pictured) is same
  - Message schedule more complicated
- *Crypto ’98 Chabaud/Joux attack on SHA-0*

- 80 steps = 5 passes over msg

Image from Wikipedia SHA1 Article
SHA-2

► Published 2001
► Three output sizes
  ► 256, 384, 512
  ► 224 added in 2004
► Very different design
► Complicated message schedule

► Still looks strong

► 256 bit output: 64 steps = 4 passes
► 512 bit output: 80 steps = 5 passes

Image from Wikipedia SHA2 Article
As of 2004, we thought we knew what we were doing.

- MD4 was known to be broken by Dobbertin, but still saw occasional use
- MD5 was known to have theoretical weaknesses from Den Boer/Bosselaers and Dobbertin, but still in wide use.
- SHA-0 was known to have weaknesses and wasn’t used.
- SHA-1 was thought to be very strong.
- SHA-2 looked like the future, with security up to 256 bits
- Merkle-Damgaard was normal way to build hashes
Crypto 2004: The Sky Falls
Crypto 2004

• Conference:
  ► Joux shows a surprising property in Merkle-Damgaard hashes
    - Multicollisions
    - Cascaded hashes don’t help security much
  ► Biham/Chen attack SHA-0 (neutral bits)

• Rump Session:
  ► Joux shows attack on SHA-0
  ► Wang shows attacks on MD4, MD5, RIPEMD, some Haval variants, and SHA-0
    - Much better techniques used for these attacks
We found out we didn’t know much about hash functions

► Wang’s techniques quickly extended
  ► Better attacks on MD5 by many people
  ► Claimed attacks on SHA-1 (2005)

► Joux’s multicollisions extended and applied widely
  ► Second preimages and herding
  ► Multicollisions even for multiple passes of hash
  ► Much more
What to do next?

► All widely used hash functions called into question
  ► MD5 and SHA1 were very widespread
  ► SHA-2 and RIPE-MD160, neither one attacked, were not widely used.

► At same time, NIST was pushing to move from 80- to 112-bit security level
  ► Required switching from SHA-1 to SHA-2

► Questions about the existing crop of hash functions
  ► SHA-1 was attacked, why not SHA-2?
Pressure for a Competition

► We started hearing from people who wanted a hash competition
► AES competition had happened a few years earlier, and had been a big success
► This would give us:
  ► Lots of public research on hash functions
  ► A new hash standard from the public crypto community
  ► Everything done out in the open
Hash Workshops

- Gaithersburg 2005
- UCSB 2006

- Encouragement to have competition
- Lots of ideas/feedback about how competition should work.
- Somewhere in here, we decided to have a competition.
2007: Call for Proposals

- We spent a lot of time getting call for proposals nailed down:
  - Algorithm spec
  - Security arguments or proofs
  - Preliminary analysis
  - Tunable security parameter(s)
Security Requirements

▶ Drop-in replacement for SHA-2
  ▶ or even SHA-1 or MD5 with truncation

▶ Security for N-bit Hash
  ▶ N/2 bit collision resistance
  ▶ \textit{N bit preimage resistance}
  ▶ N-K bit second preimage resistance
    ▶ K = \lg(\text{target message length})

▶ Eliminate length-extension property!
▶ Tunable security/performance tradeoffs.
The Competition
## Hash Competition Timetable

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
<th>Candidates Left</th>
</tr>
</thead>
<tbody>
<tr>
<td>11/2/2007</td>
<td>Call for Proposals published, competition began</td>
<td></td>
</tr>
<tr>
<td>10/31/2008</td>
<td>SHA3 submission deadline</td>
<td>64</td>
</tr>
<tr>
<td>12/10/2008</td>
<td>First-round candidates announced</td>
<td>51</td>
</tr>
<tr>
<td>2/25/2009</td>
<td>First SHA3 workshop in Leuven, Belgium</td>
<td>51</td>
</tr>
<tr>
<td>7/24/2009</td>
<td>Second-round candidates announced</td>
<td>14</td>
</tr>
<tr>
<td>8/23/2010</td>
<td>Second SHA3 workshop in Santa Barbara, CA</td>
<td>14</td>
</tr>
<tr>
<td>12/9/2010</td>
<td>SHA3 finalists announced</td>
<td>5</td>
</tr>
<tr>
<td>3/22/2012</td>
<td>Third SHA3 workshop in Washington, DC</td>
<td>5</td>
</tr>
<tr>
<td>10/2/2012</td>
<td>Keccak announced as the SHA3 winner</td>
<td>1</td>
</tr>
</tbody>
</table>
64 ➔ 51

- We started with 64 submissions (10/08)
- 51 were complete and fit our guidelines
- We published those 51 on December 2008
- Huge diversity of designs
About a year and a half—published July 2009
2009 Hash Workshop in Leuven
Many algorithms broken or seriously dented.
AES competition had 15 submissions; we took a year to get down to 14.

BLAKE  BMW  CubeHash  Echo  Fugue  Grostl  Hamsi
JH  Keccak  Luffa  SHABAL  SHAVite  SIMD  Skein
About a year and a half—announced Dec 2010
Second SHA3 Workshop at Santa Barbara
Much harder decisions
  Cryptanalytic results were harder to interpret
  Often distinguishers of no apparent relevance

BLAKE  Gostl  JH  Keccak  Skein
5 → 1

► About two years—final decision Oct 2012
► Third SHA3 Workshop in Washington, DC
► Very tough decisions
► Security, Performance, Complementing SHA3

Keccak
Security

► Nobody knocked out by cryptanalysis
► Different algorithms got different depth of cryptanalysis
► Keccak and Blake had best security margins
► Domain extenders (aka chaining modes) had security proofs
► Grostl had a very big tweak, Skein a significant one
► ARX vs non-ARX designs

• *Keccak looks very strong, and had been analyzed in sufficient depth to give us confidence.*
Performance

► All five finalists have acceptable performance
► ARX designs (BLAKE and Skein) are excellent on high-end software implementations
► JH and Grostl fairly slow in software
► Keccak is very hardware friendly
  ► High throughput per area

• Keccak performs well everywhere, and very well in hardware.
Complementing SHA2

► SHA3 will be deployed into a world full of SHA2 implementations
► SHA2 still looks strong
► We expect the standards to coexist.
► SHA3 should complement SHA2.
  ► Good in different environments
  ► Susceptible to different analytical insights

• Keccak is fundamentally different from SHA2. Its performance properties and implementation tradeoffs have little in common with SHA2.
Wrapup on Selecting a Winner

► Keccak won because of:
  ► High security margin
  ► High quality analysis
  ► Elegant, clean design
  ► Excellent hardware performance
  ► Good overall performance
  ► Design diversity from SHA2
How Did It Work Out?

► The competition brought forth a huge amount of effort by people outside NIST

► The cryptographic community did the overwhelming majority of the work:
  ► Submissions
  ► Analysis
  ► Proofs
  ► Reviews of papers for conferences/journals
  ► Performance benchmarks
  ► Implementations

► NIST's main job was to understand that work and make decisions based on it.
Keccak looks nothing like MD4
Keccak as SHA3
What Will SHA3 Standardize?

• Hash functions (fixed output length)
  – SHA3-224  SHA3-256
  – SHA3-384  SHA3-512

• Sponge functions (variable output length)
  – SHAKE256
  – SHAKE512
SHA3 Fixed-Length Hash Functions

• Drop in replacements for SHA2
• SHA3-224, SHA3-256, SHA3-384, SHA3-512
• Different output lengths are unrelated

\[
\text{SHA3-224}(X) = \text{ABCDEFG} \\
\text{SHA3-256}(X) = \text{HIJKLMNO}
\]

Almost the same security claims as SHA2.
SHAKE256 and SHAKE512

- “Sponge functions”
- Variable length output
- SHA + Keccak
- *Different output lengths give related hashes*

\[
\text{SHAKE256}(X, 224) = \text{ABCDEFG}
\]
\[
\text{SHAKE256}(X, 256) = \text{ABCDEFGH}
\]
Variable-length output is useful

- Lots of protocols and applications need this
  - OAEP, most KDFs, Fix for Vaudenay’s DSA attack

- Better to have it as part of hash definition

- But may be tricky to use correctly:
  - SHAKE256(\(X, 112\)) = K1 K2
  - SHAKE256(\(X, 168\)) = K1 K2 K3
SHAKE256 and SHAKE512

Image from Rene Peralta
Under the hood, they’re all sponges

- **Hash functions**: (SHA3-x)
  - Restricted to fixed length
  - Padding: different outputs for different lengths
- **Sponge functions**: (SHAKE-c)
  - Variable length
  - We don’t know output length till output’s done
From Keccak to SHA3:
Preliminaries
Collision and Preimage Resistance

• Collision:
  – Find $X, Y$ so that
    \[
    \text{hash}(X) == \text{hash}(Y)
    \]
  – $n$-bit output $\Rightarrow$ collisions with $2^{n/2}$ work

• Preimage:
  – Given $Y$, find $X$ so that
    \[
    \text{hash}(X) == Y
    \]
  – $n$-bit output $\Rightarrow$ preimages with $2^n$ work
Security Levels

• Convenient to assign each algorithm a security level

• Algorithm with 128-bit security level promises to resist attacks up to about $2^{128}$ computations.

• SHA256: 128-bit security level
  – But claims no preimages up to $2^{256}$ work!
  – Natural—that’s the limit for n-bit hash functions
A sponge has collision and preimage resistance of C/2 bits.
Finding a collision or preimage is equally hard
Bigger C = slower hashing
Sponges vs Merkle-Damgaard

• Most MD hashes: n bit output means
  – n bits preimage resistance
  – n/2 bits collision resistance

• Sponges: C bit capacity means
  – C/2 bit security level
  – Variable output size
From Keccak to SHA3
Keccak SHA3 Submission

• Had four versions, each with a different capacity
  – Keccak-224, -256, -384, -512
  – Hard to see why we needed four
• Guaranteed n-bit preimage resistance by making capacity huge.
• Suffered big performance hit to get this preimage resistance.
  – Hard to see why this made sense.
One security level for each function
Only two capacities in SHA3

- **SHA3-224***  
  128 bits of security
  against everything  
  ( C = 256 )

- **SHA3-256**  
  256 bits of security
  against everything  
  ( C = 512 )

- **SHAKE256**

- **SHA3-384***

- **SHA3-512**

- **SHAKE512**
A sponge has collision and preimage resistance of C/2 bits.
Finding a collision or preimage is equally hard
Bigger C = slower hashing
Security level determined by hash function internals, not output size

► 128-bit security level
  ► SHA3-224
  ► SHA3-256
  ► SHAKE256

► 256-bit security level
  ► SHA3-384
  ► SHA3-512
  ► SHAKE512
Summary of Keccak $\rightarrow$ SHA3 Changes

• Changed padding scheme
  – Sakura scheme from Keccak designers
  – Supports fixed-length hashes and sponges
  – Supports tree hashing
• Only two capacities (256 and 512)
• Preimage strength = collision strength
  – Using tunable parameter to make performance/security tradeoff
  – But this is a pretty big change from the submission
What next?
Getting the FIPS Out

• This should be FIPS 202
• Draft for public comment around end of October 2013.
• The FIPS process can be slow
  ...and a lot of it is outside our control
  – The final FIPS document goes to the Secretary of Commerce for approval
Authenticated Encryption

• Keccak specified a duplex mode for authenticated encryption
• We plan to standardize this in a special publication
• Hope to have draft for public comment next year
PRF

• Keccak specifies a dedicated PRF
  – Can be used in place of HMAC
  – Perhaps also for randomized hashing

• We also plan to standardize this in a special publication.

• Hope to have a draft out next year.
Tree Hashing

• We are also working on a standard for tree hashing
  – Will incorporate Keccak team’s Sakura padding scheme where possible
  – Will support tree-hashing with SHA3 and SHA2
• Hope to have a draft out next year.
Random Number Generation

• Keccak Duplex mode can be used for cryptographic random number generation
• We are considering adding another DRBG for SP 800-90A based on SHA3 in duplex mode
• No timetable or commitment to this yet
Further in the Future

• We are interested in analysis of Keccak with smaller permutation sizes
  – Could be really nice for constrained devices
  – Currently not a lot of published analysis
• What else can be done with sponge functions?
• What else can be done with duplex mode?
2014 NIST Hash Workshop

• Colocated with Crypto 2014
  – Friday and Saturday
• Workshop on all things SHA2 and SHA3
  – Keccak with smaller permutations
  – Cryptanalysis and differential/linear trail bounds
  – Tree hashing
  – Generic hash-based authenticated encryption
  – Clever applications for sponges or duplex mode

Thank You!

• This whole thing would have been impossible without the help of the community
• The amount of work done for free to choose a new SHA3 was incredible
• We really appreciate it

• Questions?