Engineering considerations for the SHA-3 hash function

Niels Ferguson
Microsoft

Motivation

- SHA-3 must have good engineering properties
- These are easier to evaluate than security
- Save time by selecting on engineering requirements first
  - Pick candidates with good engineering properties
  - Evaluate those for security

Requirements are subjective

- Engineering properties are mostly objective
  - E.g. Algorithm X runs in N cycles/byte
- Requirements are subjective
  - E.g. SHA-3 should run in < N cycles/byte
- My comments are based on 19 years of professional experience in designing and implementing cryptographic systems
- Other people will have other opinions

Side channel: Table lookup

- A huge security problem
- Memory addresses are not kept secret
  - CPU cache reveals address to other processes
  - Demonstrated attacks on AES on current CPUs
  - Will not be fixed in hardware
    - Fixes cost too much general CPU performance
  - Software fixes are tricky, very slow, and generally not practical
- Recommendation: do not use table lookup

Side channel: AES round

- Some candidates use AES round function
  - Fast and no side channels with AES instruction
- But: requires table lookups on all other CPUs
  - Insecure on CPUs without AES instruction

- Recommendation: avoid AES round function

Background: PC future

- Every PC will have a 64-bit OS
  - Needed to deal with > 4GB RAM
  - 32-bit applications will remain
  - Applications that need speed will be 64-bit
- Prediction: Handhelds will go to 64 bit CPUs
  - When their RAM exceeds 4 GB
  - Currently at 128-256 MB RAM
- Recommendation: optimize for 64-bit (rather than 32-bit) performance

XMM registers

- Some implementations use XMM registers
  - SSE and AES instructions
- Only available on x86-family CPUs
- Very slow in 32-bit Windows Kernel mode
- Recommendation: also evaluate implementation without XMM register

Multi-core

- Crypto is at the bottom of the software stack
  - E.g. DPCs, boot code, initialization & FIPS self-test
- Cannot use multiple cores
  - Necessary thread functions not available
- Two implementations are not practical
  - Choice cannot be automated
  - Existing code & APIs do not support a choice
  - Too many changes required
- Recommendation: ignore multi-core code

Performance: PCs

- SHA-256 (≈20 c/B) is a performance problem
  - SHA-1 is 3x faster (≈7.5 c/B)
  - SHA-256 adoption hindered by low performance
- Performance = power use = battery life
- Recommendation: SHA-3 should be < 20 c/B, preferably < 10 c/B

Performance: embedded systems

- Many different type of embedded systems
- Hardware designed for particular situation
- Any reasonable algorithm is usable
  - Faster is always preferable

Performance: smart cards

- Typically not a problem
- Smart cards process very little data
- Any reasonable algorithm is usable
  - Faster is always preferable

RAM use: servers

- Server can have 10^5 simultaneous connections
- RAM used per connection is important
  - RAM is cheap but slow
  - CPU caches are fast but limited in size
- Minimize RAM use of algorithm
- Recommendation: use < 1 kB, preferably < 300 B

RAM use: smart cards

- RAM is 128-256 bytes and up.
  - More RAM costs more
  - Adding 1 cent to card cost is a *huge* deal
- Hashing is only one of many functions
  - Do not expect to use >50% of memory
- Recommendation: SHA-3-256 uses < 200 B, preferably < 100 B

My recommendation

- Select candidates for round 2 that have the following properties
  - Speed < 20 c/B on x64
  - No table lookup (and thus no AES round)
  - Require < 200 bytes RAM (for 256-bit hash)

Conflict of interest

- I am one of the Skein designers
- We designed Skein to fit these requirements
  - Not the other way around

Questions?