SHA

The SHA, or Secure Hash Algorithm, is best described in FIPS 180–2 with Change Notice, available at the NIST website mentioned last time.

SHA–1 begins with a message $M$ of length $l < 2^{64}$. The message schedule consists of 80 words of length 32; denote these $W_0, \ldots, W_{79}$. There are five auxiliary variables $a$, $b$, $c$, $d$, and $e$. The hash value is five 32-bit words; in other words, the length is 160 bits, so the security against birthday attacks is $2^{80}$.

The message is padded to make its length a multiple of 512. Append a ‘1’, then enough bits to make the current length congruent to 448 mod 512. This leaves 64 bits; put the length (as a binary digit) there.

| message | \ldots | 10 \cdot 0 | length |

Now, the message is parsed into 512-bit blocks $M^{(i)}$. Each of these is represented as sixteen 32-bit words, or 64 hexadecimal characters.

The initial hash value is set to

\[
\begin{align*}
a &= H_0^{(0)} &= 67452301 \\
 b &= H_1^{(0)} &= efcdab89 \\
 c &= H_2^{(0)} &= 98badcfe \\
 d &= H_3^{(0)} &= 10325476 \\
 e &= H_4^{(0)} &= c3d2e1f0
\end{align*}
\]

These are not random. The first two are big-endian representations of the hex digits 0 through f, with each pair interchanged. the second two come from a similar rule, while the last mixes items from the beginning of the hex count with its end.

This concludes the preprocessing. The hash uses 80 binary functions $f_i$ of the 32-bit words $x$, $y$, and $z$. The notation is $\land$ for AND, $\oplus$ for XOR, $\neg$ for NEGATE, and $\lor$ for OR.

\[
\begin{align*}
f_i(x, y, z) &= (x \land y) \oplus (\neg x \land z) & \text{if } 0 \leq i < 20 \\
f_i(x, y, z) &= x \oplus y \oplus z \ [\text{parity}] & \text{if } 20 \leq i < 40 \\
f_i(x, y, z) &= (x \land y) \oplus (x \land z) \oplus (y \land z) \ [\text{majority}] & \text{if } 40 \leq i < 60 \\
f_i(x, y, z) &= x \oplus y \oplus z \ [\text{parity, } \text{[bis]}] & \text{if } 60 \leq i < 80
\end{align*}
\]

The actual hash starts with the message schedule; $ROTL^j$ is a left circular shift by $j$ bits.
\[ W_t = \begin{cases} M_t^{(i)} & 0 \leq t \leq 15 \\ ROTL^1(W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) & 16 \leq t \leq 79 \end{cases} \]

The working variables \( a, \ldots, e \) are initialized to \( H_{0}^{(i-1)}, \ldots, H_{4}^{(i-1)} \), respectively.

Each round consists of the following set of operations. \( K_t \) is a constant determined as the fractional part of the square root of the \( t \)th prime number.

\[
T = ROTL^5(a) + f_t(b, c, d) + e + K_t + W_t; \\
e = d; d = c; \\
c = ROTL^{30}(b); \\
b = a; a = T.
\]

Finally,

\[
H_{0}^{(i)} = a + H_{0}^{(i-1)}, \text{ etc.}
\]

The final hash is the concatenation of \( H_{0}^{(80)}, \ldots, H_{4}^{(80)} \).