GUARANTEED MESSAGE AUTHENTICATION FASTER THAN MD5 (ABSTRACT)

DANIEL J. BERNSTEIN

Let $r_0, r_1, r_2, r_3, x_0, x_1, x_2, x_3, m_0, m_1, m_2, \ldots, m_{n-1}$ be integers in $[-2^{31}, 2^{31}-1]$. Define $r = 2^{96}r_3 + 2^{64}r_2 + 2^{32}r_1 + r_0$, $x = 2^{96}x_3 + 2^{64}x_2 + 2^{32}x_1 + x_0$, and

$$s = (r^{n+1} + r^n m_0 + r^{n-1} m_1 + \cdots + r m_{n-1} + x) \bmod (2^{127} - 1).$$

I can compute s in about $330 + 19n$ Pentium cycles, or $470 + 26n$ UltraSPARC cycles, after a precomputation depending only on r.

Applications. Here's one way to mathematically guarantee the authenticity of a single message $m = (m_0, m_1, \ldots, m_{n-1})$. The sender and receiver share a secret uniform random pair (r, x). The sender computes s as above and sends (m, s). The receiver verifies (m, s) by recomputing s. An attacker, given (m, s), has a negligible probability of successfully forging a different message.

This system is faster than yesterday's MD5-based systems: $s = \mathrm{MD5}(r, m, x)$, for example, or $s = \mathrm{MD5}(r, \mathrm{MD5}(x, m))$. It provides essentially the same level of security against today's attacks; unlike MD5, it is also guaranteed secure against tomorrow's attacks. It can easily be extended to handle multiple messages.

The same method can be used to reduce a multiprecision integer modulo a big secret prime. One application is to check ring equations involving large integers—for example, the equation $s^2 = tn + fh$ in my variant of the Rabin-Williams public-key signature system—with a negligible chance of error.

Method. The following comments apply to the Pentium.

The fastest way to add and multiply small integers is with the help of the floating-point unit. For example, it takes just one cycle to compute an exact product of two 32-bit integers in floating-point registers. There is an old trick to split the result into low bits and high bits with a few floating-point additions and subtractions; one need not retrieve integers from the floating-point unit.

In multiprecision arithmetic one should generally use a radix below 2^{32} so that more useful work can be done between carries. I precompute small integers $c_{i,j}$ such that r^i is congruent to $c_{i,0} + 2^{26}c_{i,1} + 2^{52}c_{i,2} + 2^{78}c_{i,3} + 2^{104}c_{i,4}$ modulo $2^{127} - 1$. If n is not too large then the dot products $\sum_i c_{n-i,0} m_i$, $\sum_i c_{n-i,1} m_i$, etc. fit safely into floating-point registers. Handling large n is not much more difficult.

The `gcc -O6` optimizer does a poor job of instruction scheduling and register allocation. I use `gcc -O1` and schedule instructions manually. The speeds reported above are still not optimal.

Date: 19990404.

The author was supported by the National Science Foundation under grant DMS–9600083.

History. There are many previous message authentication systems that reduce various rings modulo big secret random prime ideals. This is the first high-security system to break the MD5 speed barrier. The direct ancestor of this system is Shoup's analogous system in characteristic 2.

Department of Mathematics, Statistics, and Computer Science (M/C 249), The University of Illinois at Chicago, Chicago, IL 60607–7045

E-mail address: `djb@pobox.com`