

## Re: This Week's Finds in Mathematical Physics (Week 226)

---

*Source:* <http://sci.tech-archive.net/Archive/sci.physics.research/2006-02/msg00195.html>

---

- *From:* Stephen Riley <steve@xxxxxxxxxxxxxxxxxxxxxx>
  - *Date:* Mon, 13 Feb 2006 22:36:01 +0000 (UTC)
- 

In message <dsn1ji\$sfj\$1@xxxxxxxxxxxxxx>, John Baez <baez@xxxxxxxxxxxxxxxxxxxxxxxxxx> writes

"I think it's usually credited to a small set of coinventors,  
and I think Ralph Merkle is a coinventor either of MD5 or one  
of its immediate ancestors."

I'd like more information on this.

The book "Applied Cryptography" by Bruce Schneier goes into detail on these algorithms and has loads of references. Schneier credits Rivest as the designer of MD4, saying Bert den Boer and Antoon Bosselaers successfully cryptanalysed the last of the algorithms three rounds, while Ralph Merkle successfully attacked the first two rounds. Others are credited (with attacks and analysis on MD4) too. Schneier credits Rivest as strengthening MD4 with the result being MD5. While Rivest himself outlined the improvement of MD5 over MD4 in

"The MD5 Message Digest Algorithm", RFC 1321, Apr 1992

If you want more references, Schneier's book has another 1652 of them :)

A quick google also turns up citations like "At Crypto '91 Ronald L. Rivest introduced the MD5 Message Digest Algorithm as a strengthened version of MD4, differing from it on six points".

Of course, if deliberate tampering is what you fear, you have  
to send the digest by a different channel than the original file,  
or use some other trick.

By coincidence my Masters thesis, soon due for submission, was on the use of one of those other tricks, in business: digital signatures. Nothing interesting mathematically, it was only as interesting a subject as I could manage to get a degree in IT to be :)

But if you prove that P \*does\* equal NP, you might make more money by  
breaking cryptographic hash codes and setting yourself up as the Napoleon

Re: This Week's Finds in Mathematical Physics (Week 226)

of crime.

For a bit of fun I once wrote a program that would generate random programs, spitting out and testing something like 50,000 a minute. It would quickly output programs for polynomial time problems, like finding greatest common divisors or whatever, but the only ones it spat out for NP problems like factoring were polynomial solutions, such as iterating through odd numbers until the square root of  $n$ . As I say it was nothing scientific and it would probably be unlikely to spit out anything useful anyway, just too many combinations. Not sure if that was a new idea, but probably not.

—

Stephen Riley

.