# The RONALD BAT

Daniel J. Bernstein *

djb@cr.yp.to

RONALD is a sample signing encrypting BAT (Benchmarkable Asymmetric Tool). RONALD uses the popular OpenSSL library, `http://www.openssl.org`, to generate keys, encrypt messages, decrypt messages, sign messages, and verify messages. Other BAT implementors may find RONALD useful as an illustration of the ease of writing BATs.

Beware that the speed and security of RONALD can be improved in many ways. RONALD is *not* meant as a state-of-the-art implementation of public-key cryptography.

## Signature system

RONALD implements the RSA public-key signature system. The inventors of RSA are Ron Rivest, Adi Shamir, and Len Adleman.

Actually, "RSA" is a misnomer. The system that was actually published by Rivest, Shamir, and Adleman in 1978 didn't hash messages, so it was trivially breakable. Furthermore, it used large random exponents rather than a small exponent, so it didn't have fast verification.

Rabin published a signature system in January 1979 fixing these deficiencies. Rabin's system featured hashing, finally producing a public-key signature system that was hard to break; Rabin pointed out that a broad class of attacks "does not seem a serious threat because of the hashing." Rabin's signature system remains unbroken today. Rabin's system also featured a small exponent; Rabin pointed out that his system was "several hundred times faster" than the system published by Rivest, Shamir, and Adleman. Both of these contributions from Rabin are essential parts of what is now commonly called "RSA."

Rabin does receive widespread credit for another improvement he introduced in 1979, namely exponent 2. There have been many other improvements in "RSA-type" signature systems; for further discussion see my paper "Proving tight security for standard Rabin-Williams signatures."

Anyway, RONALD version 1's `signedmessage` function uses exponent 3 with PKCS #1 padding to sign the MD5 hash of a message. RONALD rejects messages longer than 1000000000 bytes to protect OpenSSL's MD5 API against overflows. The use of MD5 in RONALD should not be interpreted as any sort of endorsement of MD5; again, RONALD is not meant as a state-of-the-art implementation of public-key cryptography!

RONALD version 2's `signatureofshorthash` function uses exponent 3 with PKCS #1 padding to sign a short message. Version 2 has less code here than

* Date of this document: 2006.06.14. 

version 1, because it relies on BATMAN to automatically build `signedmessage` and `messagesigned` from `signatureofshorthash` and `verification`.

RONALD version 3's `signedshortmessage` function uses exponent 3 with PKCS #1 padding to sign a short message with message recovery. Version 3, like version 2, has less code here than version 1, and has the advantage of shorter signed messages.

RONALD's signing times, as reported by eBATS, are almost double the times reported by OpenSSL's speed-testing utility. Each RONALD signature ends up recomputing OpenSSL's undocumented internal `BN_MONT_CX` values; this time is included in the eBATS measurements but is skipped by OpenSSL's speed-testing utility. Of course, one could speed up RONALD by saving OpenSSL's undocumented internal values as part of RONALD's secret keys.

### Encryption system

RONALD also implements the RSA public-key encryption system. As above, calling this system "RSA" is actually giving too much credit to Rivest, Shamir, and Adleman.

RONALD version 1's `ciphertext` function uses exponent 3 with PKCS #1 padding to encrypt a 16-byte key; that key is then used to encrypt a message using RC4 (with no protection against malleability). The use of RC4 in RONALD should not be interpreted as any sort of endorsement of RC4; again, RONALD is not meant as a state-of-the-art implementation of public-key cryptography!

RONALD version 2's `shortciphertext` function uses exponent 3 with PKCS #1 padding to encrypt a short message. Version 2 has less code here than version 1, because it relies on BATMAN to automatically build `ciphertext` and `plaintext` from `shortciphertext` and `shortplaintext`.

RONALD version 3 has the same `shortciphertext` function as RONALD version 2.

### Security estimates

RONALD version 3 includes `distinguishingchance` and `forgerychance` and so on. These functions simply call BATMAN's `factorizationchance` function. BATMAN's `factorizationchance` function is currently rather primitive but hopefully will be improved later.

RONALD version 3 also returns 0 from `ccattacks`, asserting that adaptive chosen-ciphertext attacks are no better than chosen-plaintext attacks. I have no idea whether this is actually true for OpenSSL's RSA padding.