NUMBER 2 — JANUARY 23, 1996

RSA Laboratories' **Bulletin**

News and advice from RSA Laboratories

Timing Attacks on Cryptosystems

Dr. Burt Kaliski

RSA Laboratories

Summary

A new class of "timing attacks" published recently by Paul Kocher, an independent cryptography consultant (see J. Markoff, "Secure digital transactions just got a little less secure," New York Times, December 11, 1995), poses a potential risk to a variety of cryptosystems, including RSA, DSA, Diffie-Hellman and RC5.

The concept of timing attacks has been known for years, but Kocher's results are new and significant in that they can recover complete key information given only the running time of an operation. Previous attacks could only recover partial key information or required timing information on the individual steps within a cryptographic operation.

Of greatest concern are algorithms with a key-dependent correlation between input and running time, including RSA, DSA, Diffie-Hellman, and RC5. Algorithms without such correlations are unaffected. In general, DES, DESX, triple-DES, RC2, RC4, and standard message-digest algorithms (for which the "key information" might be the input message) are not affected, although in special cases, according to Kocher, there may be some minor leakage of key information.

For one of the new attacks to succeed, it must be possible to measure the running time of cryptographic operations. Thus operations in an interactive protocol such as SSL, or in a cryptographic module in the attacker's possession, such as a smart card, are at the greatest risk. This is true even if there is some "noise" in the measurement, such as transmission delays, since the attacker can factor out the noise by averaging enough measurements. Operations performed privately and without external feedback, on the other hand, such as off-line digital signatures or file encryption, are unaffected.

Burt Kaliski is chief scientist at RSA Laboratories. He can be contacted at burt@rsa.com

Impact on RSA Data Security Products

The timing attacks potentially affect the implementations of the RSA and DSA algorithms in BSAFE and RSAREF, which, like many other implementations, are optimized for performance, and hence take an amount of time that can potentially be correlated with the input. (TIPEM and other products based on BSAFE are similarly affected.)

The attacks do not affect the implementation of Diffie-Hellman key agreement in BSAFE and RSAREF, as the Diffie-Hellman exponent is not a fixed key, but instead varies from one call to another. They also do not affect the implementation of RC5 in BSAFE on standard platforms as the RC5 rotations take a constant amount of time on those platforms and so do not give opportunity for correlation. However, RC5 may be affected on other platforms.

Other algorithms in BSAFE and RSAREF, including DES, DESX, triple-DES, RC2, RC4, and the message-digest algorithms, as noted above, are in general not affected.

Countermeasures

A simple way to prevent timing attacks, regardless of algorithm, is to ensure that all operations with a given algorithm take the same amount of time by "quantizing" the operations into a fixed time period. This approach is highly dependent on the environment, and may degrade performance, but it requires no modification to the algorithm implementations. "Quantizing" code was recently offered by Matt Blaze of AT&T Bell Laboratories at *ftp://research.att.com/dist/mab/quantize.shar*.

For RC5, it is sufficient to ensure that the rotations in RC5 take a constant amount of time, which is the case on all platforms supported by BSAFE 3.0.

For RSA, one can prevent the attacks by introducing what is called "blinding" into the cryptographic operations, without changing the underlying implementation. This approach, suggested by Ron Rivest, is being incorporated into BSAFE 3.0 for RSA. (Similar algorithmic could be incorporated for countermeasures for DSA and Diffie-Hellman.) Specifically, the RSA private-key operation $y := x^d \bmod n$, where x is the input, y is the output, n is the RSA modulus and d is the private exponent, is implemented as follows:

1. Generate a secret random number r between 0 and $n - 1$.
2. Compute $x' := x r^e \bmod n$, where e is the public exponent.
3. Compute $y' := (x')^d \bmod n$ with the ordinary RSA implementation.
4. Compute $y := y' r^{-1} \bmod n$. It is easy to see that the result is as expected by noting that $r^{ed} \equiv r \pmod{n}$.

Since step 3 involves a random, secret x', its running time cannot be correlated with the input x, hence the term "blinding." Consequently, a timing attack cannot obtain any information about the private key.

There is still the theoretical concern that information about the random value r may be revealed from the time for steps 2 and 4, but this does not seem to be a practical issue provided the value of r varies from one RSA operation to the next.

A practical way to generate the r value in step 1, suggested by Steve Burnett, is to compute a one-way function on the input x and the RSA private key; this makes the r value vary from one input value to another, and also keeps the r value unknown. This is being done in BSAFE 3.0 following techniques based on MD5 similar those of Krawcyzk et al. for message authentication (see "Keyed-MD5 for message authentication" submitted as an Internet-Draft in November 1995, and M. Bellare, R. Canetti, and H. Krawczyk, "Keyed hash functions and message authentication," to be presented at the 1996 RSA Data Security Conference). In particular, the seed is computed as

$$\text{seed} := \text{MD5}\,(p \,||\, \text{pad}_p \,||\, \text{MD5}\,(q \,||\, \text{pad}_q \,||\, c))$$

where p and q are the RSA primes, least significant byte first, pad_p and pad_q are strings of zero bytes sufficient to extend p and q to lengths that are multiples of MD5's block size of 64 bytes, and $||$ denotes concatenation. The random r value is generated from the seed with BSAFE's MD5Random pseudorandom generator.

For more information on this and other recent developments in cryptography, contact RSA Laboratories at one of the addresses below.

RSA Laboratories
100 Marine Parkway, Suite 500
Redwood City, CA 94065 USA
415/595-7703
415/595-4126 (fax)
rsa-labs@rsa.com
http://www.rsa.com/rsalabs/