# NIST Special Publication 800-106
# Randomized Hashing for Digital Signatures

**Quynh Dang**

**Computer Security Division**
**Information Technology Laboratory**

## C O M P U T E R  S E C U R I T Y

**February 2009**


**U.S. Department of Commerce**
*Otto J. Wolff, Acting Secretary*
**National Institute of Standards and Technology**
*Patrick D. Gallagher, Deputy Director*

## Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analysis to advance the development and productive use of information technology. ITL's responsibilities include the development of management, administrative, technical, and physical standards and guidelines for the cost-effective security and privacy of non-national security-related information in Federal information systems. This special publication 800-series reports on ITL's research, guidelines, and outreach efforts in information system security, and its collaborative activities with industry, government, and academic organizations.

## Abstract

NIST approved digital signature algorithms require the use of an approved cryptographic hash function in the generation and verification of signatures. Approved cryptographic hash functions and digital signature algorithms can be found in [FIPS 180-3] and [FIPS 186-3], respectively. The security provided by the cryptographic hash function is vital to the security of a digital signature application. This Recommendation specifies a method to enhance the security of the cryptographic hash functions used in digital signature applications by randomizing the messages that are signed.


KEY WORDS: digital signature, cryptographic hash function, hash function, collision resistance, randomized hashing.

## Acknowledgements

The author, Quynh Dang of the National Institute of Standards and Technology (NIST) gratefully appreciates the contributions and comments from Elaine Barker, William E. Burr, Shu-jen Chang, Donna F. Dodson, John Kelsey, Ray Perlner, W. Timothy Polk and Andrew Regenscheid from NIST, Rich Davis from NSA, and Hugo Krawczyk during the development of this Recommendation.

Table of Contents

# 1. Introduction

This Recommendation provides a technique to randomize messages that are input to a cryptographic hash function during the generation of digital signatures using the Digital Signature Algorithm (DSA), Elliptic Curve Digital Signature Algorithm (ECDSA) and RSA. Approved cryptographic hash functions for Federal government use are specified in Federal Information Processing Standard (FIPS) 180-3, the Secure Hash Standard (SHS) [FIPS 180-3]. Digital Signatures **shall** be generated as specified in FIPS 186-3, the Digital Signature Standard [FIPS 186-3].

Collision resistance is a required property for the cryptographic hash functions used in Digital Signature Applications. The intent of this randomization method is to strengthen the collision resistance provided by the cryptographic hash functions in digital signature applications without any changes to the core cryptographic hash functions and digital signature algorithms. A message will have a different digital signature each time it is signed if it is randomized with a different random value. The randomization method specified in this Recommendation is a NIST-approved randomized hashing method.

This Recommendation is based on the work of Shai Halevi and Hugo Krawczyk [Randomizing].

## 1.1 Authority

This document has been developed by the National Institute of Standards and Technology (NIST) in furtherance of its statutory responsibilities under the Federal Information Security Management Act (FISMA) of 2002, Public Law 107-347.

NIST is responsible for developing standards and guidelines, including minimum requirements, for providing adequate information security for all agency operations and assets, but such standards and guidelines **shall not** apply to national security systems. This recommendation is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130, Section 8b(3), Securing Agency Information Systems, as analyzed in A-130, Appendix IV: Analysis of Key Sections. Supplemental information is provided in A-130, Appendix III.

This Recommendation has been prepared for use by Federal agencies. It may be used by non-governmental organizations on a voluntary basis and is not subject to copyright. Nothing in this document should be taken to contradict standards and guidelines made mandatory and binding on Federal agencies by the Secretary of Commerce under statutory authority. Nor should this Recommendation be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of OMB, or any other Federal official.

Conformance testing for randomized hashing implementations, as specified in this Recommendation, will be conducted within the framework of the Cryptographic Module Validation Program (CMVP), a joint effort of NIST and the Communications Security Establishment of the Government of Canada. The requirements of this Recommendation are indicated by the word "**shall**".

## 1.2 Audience and Assumptions

This Recommendation is targeted at Federal agencies and implementers of digital signature applications. Readers are assumed to have a working knowledge of cryptography, especially the cryptographic hashing algorithms specified in [FIPS 180-3] and their use by the digital signature algorithms specified in [FIPS 186-3].

## 1.3 Glossary

| | |
|---|---|
| Approved | FIPS-approved and/or NIST-recommended. An algorithm or technique that is either:<br><br>1) Specified in a FIPS or NIST Recommendation,<br><br>2) Adopted in a FIPS or NIST Recommendation, or<br><br>3) Specified in a list of NIST-approved security functions. |
| Byte | A string of eight bits. |
| Collision | An event in which two different messages have the same message digest. |
| Collision resistance | An expected property of a cryptographic hash function whereby it is computationally infeasible to find a collision, See "Collision". |
| Cryptographic hash function | A function that maps a bit string of arbitrary length to a fixed length bit string and is expected to have the following three properties:<br><br>1) Collision resistance (see Collision resistance),<br><br>2) Preimage resistance (see Preimage resistance) and<br><br>3) Second preimage resistance (see Second preimage resistance).<br><br>Approved cryptographic hash functions are specified in [FIPS 180-3]. |
| Digital signature | The output that results from the successful completion of a digital signature algorithm operating on data (e.g., a message) that is to be signed. When used appropriately, a digital signature can provide assurance of data integrity, origin authentication, and signatory non-repudiation. See [FIPS 186-3] for details. |

| | |
|---|---|
| Hashed | The process whereby data (e.g., a message) was input to a cryptographic hash function (see Cryptographic hash function) to produce a hash value (see Hash value). |
| Hashing algorithm | A sequence of steps to execute a cryptographic hash function (see Cryptographic hash function). |
| Hash value | The result of applying a cryptographic hash function to data (e.g., a message). Also known as a "message digest". |
| Message digest | See "Hash value". |
| Preimage | A message *Ms* that produces a given message digest when it is processed by a hash function. |
| Preimage resistance | An expected property of a cryptographic hash function such that, given a randomly chosen message digest, *message_digest*, it is computationally infeasible to find a preimage of the *message_digest*, See "Preimage". |
| Random bit | A bit for which an attacker has exactly a 50% probability of success of guessing the value of the bit as either zero or one. |
| Random value | A sufficient entropy bit string. |
| Randomized hashing | A technique for randomizing the input to a cryptographic hash function. |
| Randomized message | A message that has been modified using a random value. |
| Salt | A bit string generated during digital signature generation using the *RSA Signature Scheme with Appendix - Probabilistic Signature Scheme (RSASSA-PSS RSA)* [PKCS#1 v2.1]. |
| Second preimage | A message *Ms'*, that is different from a given message *Ms* , such that its message digest is the same as the known message digest of *Ms*. |
| Second preimage resistance | An expected property of a cryptographic hash function whereby it is computationally infeasible to find a second preimage of a known message digest, See "Second preimage". |

## 1.4 Abbreviations and Terms

DSA Digital Signature Algorithm

| | |
|---|---|
| ECDSA | Elliptic Curve Digital Signature Algorithm |
| FIPS | Federal Information Processing Standard |
| PKCS | Public Key Cryptography Standard |
| RSA | Rivest-Shamir-Adleman |
| RSASSA-PSS | RSA Signature Scheme with Appendix - Probabilistic Signature Scheme (RSASSA-PSS) as specified in Public Key Cryptography Standard (PKCS) #1 [PKCS#1 v2.1] |
| SP | Special Publication |

## 1.5 Symbols

| | |
|---|---|
| $M$ | The randomized message. |
| $Ms$ | The (original) message prior to randomization. |
| *padding* | A string consisting of a single "1" bit, followed by zero or more "0" bits. |
| $rv$ | The random value that is combined with the message $Ms$ to produce the randomized message $M$. |
| *rv_length_indicator* | A 16-bit binary representation of the length (in bits) of $rv$. |
| $(r, s)$ | Digital signature for DSA or ECDSA. |
| $\|x\|$ | The length (in bits) of the bit string $x$. For example, $\|01100100\| = 8$. |
| $+$ | Addition. For example, $5 + 4 = 9$. |
| $\oplus$ | Bitwise logical "exclusive-or", where $0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$, and $1 \oplus 1 = 0$. For example: $01101 \oplus 11010 = 10111$. |
| $\lfloor a \rfloor$ | The floor of a non-negative number $a$: the greatest integer that is smaller than or equal to $a$. For example, $\lfloor 5 \rfloor = 5$, and $\lfloor 5.9 \rfloor = 5$. |
| $\|\|$ | Concatenation; e.g. $A \| \| B$ is the concatenation of bit strings $A$ and $B$. |
| $0^x$ | A string of $x$ zero bits. For example, $0^5 = 00000$. |
| $rv[0\ldots b]$ | For bit string $rv$, $rv[0\ldots b]$ is a substring consisting of the leftmost $b$+1 bit(s) of $rv$, where $b \geq 0$. |
| $S$ | The desired security strength for a digital signature. |
| *sig* | A digital signature of a randomized message. |

| | |
|---|---|
| *sig'* | The received digital signature of a randomized message. |
| *x mod n* | The unique remainder *r,* $0 \leq r \leq (n-1)$, when integer *x* is divided by positive integer *n.* For example, 23 mod 7 = 2. |

## 2. Scope and Applicability of Randomized Hashing

Randomized hashing is designed for situations where one party, the message preparer, generates all or part of a message to be signed by a second party, the message signer. If the message preparer is able to find cryptographic hash function collisions (i.e., two messages producing the same hash value), then she might prepare meaningful versions of the message that would produce the same hash value and digital signature, but with different results (e.g., transferring $1,000,000 to an account, rather than $10). Cryptographic hash functions have been designed with collision resistance as a major goal, but the current concentration on attacking cryptographic hash functions may result in a given cryptographic hash function providing less collision resistance than expected. Randomized hashing offers the signer additional protection by reducing the likelihood that a preparer can generate two or more messages that ultimately yield the same hash value during the digital signature generation process – even if it is practical to find collisions for the hash function. However, the use of randomized hashing may reduce the amount of security provided by a digital signature when all portions of the message are prepared by the signer.

In randomized hashing, a quantity, called a "random value," or *rv,* that the preparer cannot predict, is used by the signer to modify the message. This modification occurs after the preparer commits to the message (i.e., passes the message to the signer), but before the signer computes the hash value. The technique specified in this Recommendation does not require knowledge of the specific cryptographic hash function; the same randomization process is used regardless of the cryptographic hash functions used in the digital signature applications.

NIST does not require the use of randomized hashing. Protocol and application designers **should** select cryptographic hash functions believed to be collision resistant, and then consider the use of the randomized hashing in the design of their protocol or application whenever one party prepares a full or partial message for signature by another party.

The randomization method specified in this Recommendation is an approved method for randomizing messages prior to hashing. The method will enhance the security provided by the approved cryptographic hash functions in certain digital signature applications. Other randomization techniques may be used, but NIST makes no claims regarding their effect on security.

## 3. Randomized Hashing

## 3.1 Randomizing a Message Prior to Hashing

When using one of the approved digital signature algorithms [FIPS 186-3] to generate a digital signature for a message, the message must first be processed using one of the approved hash algorithms [FIPS 180-3]. The technique specified in this Recommendation is used to randomize the original message prior to hashing. The randomizing technique specified below assumes only that the cryptographic hash function processes messages in the usual left-to-right order and can be used with all digital signature algorithms specified in FIPS 186-3.

## 3.2 Message Randomization

A random value *rv* is obtained to randomize a message *Ms*. In this Recommendation, the primary function used to randomize the message with the random value is the bitwise logical "exclusive-or" operation, $\oplus$, which is defined in Section 1.5. The message *Ms* is encoded by padding with a single "1" bit, followed by zero or more "0" bits, and the resulting encoded message is then randomized as specified below.

Inputs to message randomization method:
*Ms*: an input message.
*rv*: a random bit string as described in Section 3.3 below.

Output from the message randomization method:
*M*: a randomized message.

Message Randomization (*Ms*, *rv*):
{

1. If ($|Ms| \geq (|rv| - 1)$)
   {
   1.1 $padding = 1$.
   }

   Else
   {
   1.2 $padding = 1 \,||\, 0^{(|rv| - |Ms| - 1)}$.
   }

2. $m = Ms \,||\, padding$.

3. $n$ is a positive integer, and $n = |rv|$.

4. If ($n > 1024$) then stop and output an error indicator (see Section 3.3).

5. $counter = \lfloor |m| / n \rfloor$.

6. $remainder = (|m| \bmod n)$.

7. Concatenate *counter* copies of the *rv* to the *remainder* left-most bits of the *rv* to get *Rv,* such that $|Rv| = |m|$.

$$Rv = \underbrace{rv \,||\, rv \,||\, \ldots \,||\, rv}_{counter \text{ times}} \,||\, rv[0 \ldots (remainder - 1)].$$

8. Convert *n* to a 16-bit binary string *rv_length_indicator* using the *rv_length_indicator_generation* function specified in the Appendix.

   *rv_length_indicator = rv_length_indicator_generation (n).*

9. $M = rv \,||\, (m \oplus Rv) \,||\, rv\_length\_indicator$ (Figure 1).

}

Output: *M*

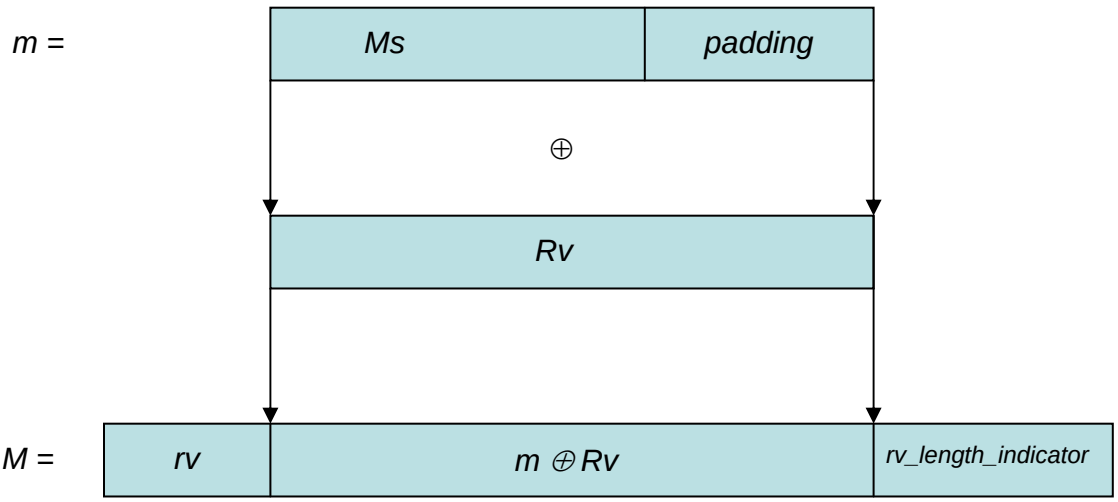


*Figure 1: Message Randomization*

## 3.3 The Random Value

The random value *rv* **shall** be a message-independent bit string of at least 80 bits, but no more than 1024 bits. The *rv* **shall** have sufficient randomness to meet the desired security strength for the digital signature application.

The *rv* **shall** be either:

1. The output from an approved random bit generator (RBG) (see [SP 800-90]),
2. Formed as the concatenation of several copies of the output from the RBG or
3. Generated as specified below when using DSA or ECDSA.

The random bit generator **shall** meet or exceed the desired security strength *S* for the digital signature application. It is important to note that the length of the output must be at least *S* bits in order to provide *S* bits of security for the *rv*.

In DSA and ECDSA, there is an existing integer value *r* that is a part of a digital signature (*r, s*), and is a message-independent value. The value of *r* is derived from an output of an approved random bit generator that meets or exceeds the required security strength *S* for the digital signature application; details can be found in [FIPS 186-3]. [FIPS 186-3] allows this value to be pre-computed prior to performing a digital signature operation. When *r* is pre-computed, *r* may be converted to a bit string (see FIPS 186-3 for the integer to bit string conversion) that is used to determine *rv* as follows:

- If the length of the bit string is at least 2*S* bits, but no more than 1024 bits, then the bit string is used as *rv*.
- If the length of the bit string is greater than 1024 bits, then the rightmost 1024 bits of the converted bit string are used as the *rv*.
- If the bit string is less than 2*S* bits in length, then one or more zero bits **shall** be prepended to the string to make a bit string of 2*S* bits that is then used as *rv*.

For DSA and ECDSA signature applications, *rv* **shall** be provided to the signature verifier along with the digital signature (*r*, *s*) unless it is agreed that *rv* will be derived from *r* as specified above.

In the RSA Signature Scheme with Appendix - Probabilistic Signature Scheme (RSASSA-PSS) digital signature scheme specified in [PKCS#1 v2.1], there is an existing bit string *Salt* that is generated during digital signature generation. The *Salt* is a message-independent bit string. The *Salt* may be used as *rv* if the following conditions are met: 1) the *Salt* is available before a message is randomized, 2) the *Salt* is output from an approved random bit generator at a security strength that meets or exceeds the desired security strength *S* for the digital signature application, and 3) the Salt is at least 80 bits, but no more than 1024 bits in length. However, for many existing applications, one or more of these conditions are not met. Note that modifications to the existing applications may cause some compatibility issues.

*rv* **shall** be kept secret until a digital signature using this value *rv* is generated. An *rv* **shall not** be used to randomize a message provided to the signer by the message preparer if the *rv* is exposed (potentially known to the preparer).

In the following cases, a single *rv* may be used by a signer to randomize multiple messages, provided that the *rv* is kept secret from the message preparer(s) until every message has been committed[1] by those preparer(s):

- When *rv* is used for message randomization as specified in this Recommendation prior to generating RSA digital signatures or
- When the *rv* is not generated from the *r* component of a DSA or ECDSA digital signature.

However, it is not recommended that a single *rv* be used for multiple messages.

When *rv* is generated from the *r* component of a DSA or ECDSA digital signature, this *rv* **shall not** intentionally be used for message randomization in any other DSA or ECDSA digital signatures.

### 3.4 Hashing

The randomizing technique in Section 3.2 **shall** be performed before the cryptographic hash function is invoked in order to produce a randomized message. This transformation requires no changes to the cryptographic hash function specified in [FIPS 180-3], since the randomized message, *M*, is hashed instead of the original message, *Ms*. However, the random value *rv* must be provided to the digital signature verifier for signature verification, i.e., the signature verifier must have *rv, Ms* and the digital signature *sig* on the randomized message *M*.

## 4. Digital Signatures Using Randomized Hashing

To accommodate the randomizing of a message during digital signature generation and verification, additional operations are needed as specified below.

**Signature Generation:**

1. Obtain the *rv* as described in Section 3.3.
2. Randomize the message *M*s with *rv* as described in Section 3.2, obtaining the randomized message *M*.
3. Generate a digital signature *sig* on the randomized message *M* as specified in [FIPS 186-3].
4. Provide the original message *Ms*, the random value *rv* and the digital signature *sig* for signature verification.

**Signature Verification:**

1. The digital signature verifier receives the signature *sig'*, the message *Ms'* and the random value *rv'* [2].

[1] A message is committed when the message preparer no longer has any control over its contents.
[2] *Ms'* , *rv'* and *sig'* are purportedly *Ms, rv* and *sig*, respectively.

2. *Ms′* is randomized with *rv′* as described in Section 3.2 (*Ms′* is used as *Ms,* and *rv′* is used as *rv* in the randomization process). Let the randomized result be *M′*.
3. *M′* and the signature *sig′* are used in the signature verification and validation process as specified in [FIPS 186-3].

## 5. References


[Randomizing] Shai Halevi and Hugo Krawczyk, Strengthening Digital Signatures via Randomized Hashing, Advances in Cryptology, CRYPTO 2006 Proceedings.

[FIPS 186-3] Federal Information Processing Standard 186-3, Digital Signature Standard (DSS), Draft November 08.

[FIPS 180-3] Federal Information Processing Standard 180-3, Secure Hash Standard (SHS), October 2008.

[SP 800-90] NIST Special Publication (SP) 800-90, Recommendation for Random Number Generation Using Deterministic Random Bit Generators, June 06.

[SP 800-57] NIST Special Publication (SP) 800-57, Part 1, Recommendation for Key Management: General, August 2005.

[PKCS#1 v2.1] RSA Laboratories, PKCS#1 v2.1: RSA Cryptographic Standard, June 14, 2002.

[SP 800-107] NIST Special Publication (SP) 800-107, Recommendation for Applications Using Approved Hash Algorithms, February 2009.

# 6. Appendix: *rv_length_indicator* generation

The following *rv_length_indicator_generation* function converts $n$, an integer in the range $80 \le n \le 1024$ in this Recommendation, to a 16-bit string $b_0 \| b_1 \| b_2 \cdots \| b_{14} \| b_{15}$, where each $b_i$ is a single bit. This function returns the 16-bit string as the *rv_length_indictor*.

*rv_length_indicator_generation ( n)*

1. $A = n$. Comment: $A$ is an integer.
2. $B = A \ mod \ 2$. Comment: $B$ is an integer, and the integer equivalent of bit $b_{15}$.
3. If ($B = 0$), then

   $b_{15}$ = "0". Comment: $b_{15}$ is a single "0" bit.

   Else

   $b_{15}$ = "1". Comment: $b_{15}$ is a single "1" bit.
4. For (integer $i = 14$ down to 0)

   4.1 $A = \lfloor A/2 \rfloor$.

   4.2 $B = A$ mod 2. Comment: B is the integer equivalent of bit $b_i$.

   4.3 If ($B = 0$), then

   $b_i$ = "0". Comment: $b_i$ is a single "0" bit.

   Else

   $b_i$ = "1" . Comment $b_i$ is a single "1" bit.
5. *rv_length_indicator* = $b_0 \| b_1 \| b_2 \cdots \| b_{14} \| b_{15}$.
6. Return *rv_length_indicator*.