digital signature

*noun*

COMPUTING

1. a digital code (generated and authenticated by public key encryption) which is attached to an electronically transmitted document to verify its contents and the sender's identity.

Digital signature

From Wikipedia, the free encyclopedia

A **digital signature** is a mathematical scheme for demonstrating the authenticity of a digital message or documents. A valid digital signature gives a recipient reason to believe that the message was created by a known sender ([authentication](https://en.wikipedia.org/wiki/Authentication)), that the sender cannot deny having sent the message ([non-repudiation](https://en.wikipedia.org/wiki/Non-repudiation)), and that the message was not altered in transit ([integrity](https://en.wikipedia.org/wiki/Data_integrity)).

Digital signatures are a standard element of most [cryptographic protocol](https://en.wikipedia.org/wiki/Cryptographic_protocol) suites, and are commonly used for software distribution, financial transactions, [contract management software](https://en.wikipedia.org/wiki/Contract_management_software), and in other cases where it is important to detect forgery or tampering.

**Contents**

  [hide]

* [1Explanation](https://en.wikipedia.org/wiki/Digital_signature#Explanation)
* [2Definition of Digital Signature](https://en.wikipedia.org/wiki/Digital_signature#Definition_of_Digital_Signature)
* [3History of Digital Signature](https://en.wikipedia.org/wiki/Digital_signature#History_of_Digital_Signature)
* [4How they work](https://en.wikipedia.org/wiki/Digital_signature#How_they_work)
* [5Notions of security](https://en.wikipedia.org/wiki/Digital_signature#Notions_of_security)
* [6Applications of digital signatures](https://en.wikipedia.org/wiki/Digital_signature#Applications_of_digital_signatures)
  + [6.1Authentication](https://en.wikipedia.org/wiki/Digital_signature#Authentication)
  + [6.2Integrity](https://en.wikipedia.org/wiki/Digital_signature#Integrity)
  + [6.3Non-repudiation](https://en.wikipedia.org/wiki/Digital_signature#Non-repudiation)
* [7Additional security precautions](https://en.wikipedia.org/wiki/Digital_signature#Additional_security_precautions)
  + [7.1Putting the private key on a smart card](https://en.wikipedia.org/wiki/Digital_signature#Putting_the_private_key_on_a_smart_card)
  + [7.2Using smart card readers with a separate keyboard](https://en.wikipedia.org/wiki/Digital_signature#Using_smart_card_readers_with_a_separate_keyboard)
  + [7.3Other smart card designs](https://en.wikipedia.org/wiki/Digital_signature#Other_smart_card_designs)
  + [7.4Using digital signatures only with trusted applications](https://en.wikipedia.org/wiki/Digital_signature#Using_digital_signatures_only_with_trusted_applications)
  + [7.5Using a network attached hardware security module](https://en.wikipedia.org/wiki/Digital_signature#Using_a_network_attached_hardware_security_module)
  + [7.6WYSIWYS](https://en.wikipedia.org/wiki/Digital_signature#WYSIWYS)
  + [7.7Digital signatures versus ink on paper signatures](https://en.wikipedia.org/wiki/Digital_signature#Digital_signatures_versus_ink_on_paper_signatures)
* [8Some digital signature algorithms](https://en.wikipedia.org/wiki/Digital_signature#Some_digital_signature_algorithms)
* [9The current state of use – legal and practical](https://en.wikipedia.org/wiki/Digital_signature#The_current_state_of_use_.E2.80.93_legal_and_practical)
* [10Industry standards](https://en.wikipedia.org/wiki/Digital_signature#Industry_standards)
  + [10.1Using separate key pairs for signing and encryption](https://en.wikipedia.org/wiki/Digital_signature#Using_separate_key_pairs_for_signing_and_encryption)
* [11See also](https://en.wikipedia.org/wiki/Digital_signature#See_also)
* [12Notes](https://en.wikipedia.org/wiki/Digital_signature#Notes)
* [13References](https://en.wikipedia.org/wiki/Digital_signature#References)
* [14Further reading](https://en.wikipedia.org/wiki/Digital_signature#Further_reading)

Explanation[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=1" \o "Edit section: Explanation)]

Digital signatures are often used to implement [electronic signatures](https://en.wikipedia.org/wiki/Electronic_signature), a broader term that refers to any electronic data that carries the intent of a signature,[[1]](https://en.wikipedia.org/wiki/Digital_signature" \l "cite_note-1) but not all electronic signatures use digital signatures.[[2]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-2)[[3]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-3) In some countries, including the United States, India, Brazil, Indonesia, Saudi Arabia,[[4]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-4) [Switzerland](https://en.wikipedia.org/wiki/Switzerland) and the countries of the [European Union](https://en.wikipedia.org/wiki/European_Union),[[5]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-Cryptomathic_MajorStandardsDigSig-5)[[6]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-CryptomathicDigSigServicesAshiqJA-6) electronic signatures have legal significance.

Digital signatures employ [asymmetric cryptography](https://en.wikipedia.org/wiki/Asymmetric_key_algorithm). In many instances they provide a layer of validation and security to messages sent through a nonsecure channel: Properly implemented, a digital signature gives the receiver reason to believe the message was sent by the claimed sender. Digital seals and signatures are equivalent to handwritten signatures and stamped seals.[[7]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-7) Digital signatures are equivalent to traditional handwritten signatures in many respects, but properly implemented digital signatures are more difficult to forge than the handwritten type. Digital signature schemes, in the sense used here, are cryptographically based, and must be implemented properly to be effective. Digital signatures can also provide [non-repudiation](https://en.wikipedia.org/wiki/Non-repudiation), meaning that the signer cannot successfully claim they did not sign a message, while also claiming their [private key](https://en.wikipedia.org/wiki/Private_key) remains secret; further, some non-repudiation schemes offer a time stamp for the digital signature, so that even if the private key is exposed, the signature is valid. Digitally signed messages may be anything representable as a [bitstring](https://en.wikipedia.org/wiki/Bitstring" \o "Bitstring): examples include [electronic mail](https://en.wikipedia.org/wiki/Electronic_mail), [contracts](https://en.wikipedia.org/wiki/Contract), or a message sent via some other [cryptographic protocol](https://en.wikipedia.org/wiki/Cryptographic_protocol).

Definition of Digital Signature[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=2" \o "Edit section: Definition of Digital Signature)]

*Main article:*[*Public-key cryptography*](https://en.wikipedia.org/wiki/Public-key_cryptography)

A digital signature scheme typically consists of three algorithms;

* A [*key generation*](https://en.wikipedia.org/wiki/Key_generation) algorithm that selects a *private key* [uniformly at random](https://en.wikipedia.org/wiki/Uniform_distribution_(discrete)) from a set of possible private keys. The algorithm outputs the private key and a corresponding *public key*.
* A *signing* algorithm that, given a message and a private key, produces a signature.
* A *signature verifying* algorithm that, given the message, public key and signature, either accepts or rejects the message's claim to authenticity.

Two main properties are required. First, the authenticity of a signature generated from a fixed message and fixed private key can be verified by using the corresponding public key. Secondly, it should be computationally infeasible to generate a valid signature for a party without knowing that party's private key. A digital signature is an authentication mechanism that enables the creator of the message to attach a code that acts as a signature.

In the following discussion, 1*n* refers to a [unary number](https://en.wikipedia.org/wiki/Unary_numeral_system).

Formally, a **digital signature scheme** is a triple of probabilistic polynomial time algorithms, (*G*, *S*, *V*), satisfying:

* *G* (key-generator) generates a public key, *pk*, and a corresponding private key, *sk*, on input 1*n*, where *n* is the security parameter.
* *S* (signing) returns a tag, *t*, on the inputs: the private key, *sk*, and a string, *x*.
* *V* (verifying) outputs *accepted* or *rejected* on the inputs: the public key, *pk*, a string, *x*, and a tag, *t*.

For correctness, *S* and *V* must satisfy

Pr [ (*pk*, *sk*) ← *G*(1*n*), *V*( *pk*, *x*, *S*(*sk*, *x*) ) = *accepted* ] = 1.[[8]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-8)

A digital signature scheme is **secure** if for every non-uniform probabilistic polynomial time adversary, *A*

Pr [ (*pk*, *sk*) ← *G*(1*n*), (*x*, *t*) ← *AS*(*sk*, · )(*pk*, 1*n*), *x* ∉ *Q*, *V*(*pk*, *x*, *t*) = *accepted*] < [negl](https://en.wikipedia.org/wiki/Negligible_function" \o "Negligible function)(*n*),

where *AS*(*sk*, · ) denotes that *A* has access to the oracle, *S*(*sk*, · ), and *Q* denotes the set of the queries on *S* made by *A*, which knows the public key, *pk*, and the security parameter, *n*. Note that we require any adversary cannot directly query the string, *x*, on *S*.[[9]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-9)

History of Digital Signature[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=3" \o "Edit section: History of Digital Signature)]

In 1976, [Whitfield Diffie](https://en.wikipedia.org/wiki/Whitfield_Diffie) and [Martin Hellman](https://en.wikipedia.org/wiki/Martin_Hellman) first described the notion of a digital signature scheme, although they only conjectured that such schemes existed.[[10]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-10)[[11]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-lysythesis-11) Soon afterwards, [Ronald Rivest](https://en.wikipedia.org/wiki/Ronald_Rivest), [Adi Shamir](https://en.wikipedia.org/wiki/Adi_Shamir" \o "Adi Shamir), and [Len Adleman](https://en.wikipedia.org/wiki/Len_Adleman) invented the [RSA](https://en.wikipedia.org/wiki/RSA_(algorithm)) algorithm, which could be used to produce primitive digital signatures[[12]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-rsa-12) (although only as a proof-of-concept – "plain" RSA signatures are not secure[[13]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-13)). The first widely marketed software package to offer digital signature was [Lotus Notes](https://en.wikipedia.org/wiki/Lotus_Notes) 1.0, released in 1989, which used the RSA algorithm.[[14]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-14)

Other digital signature schemes were soon developed after RSA, the earliest being [Lamport signatures](https://en.wikipedia.org/wiki/Lamport_signature" \o "Lamport signature),[[15]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-15) [Merkle signatures](https://en.wikipedia.org/wiki/Merkle_tree" \o "Merkle tree) (also known as "Merkle trees" or simply "Hash trees"),[[16]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-16) and [Rabin signatures](https://en.wikipedia.org/wiki/Rabin_signature).[[17]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-17)

In 1988, [Shafi Goldwasser](https://en.wikipedia.org/wiki/Shafi_Goldwasser" \o "Shafi Goldwasser), [Silvio Micali](https://en.wikipedia.org/wiki/Silvio_Micali), and [Ronald Rivest](https://en.wikipedia.org/wiki/Ronald_Rivest) became the first to rigorously define the security requirements of digital signature schemes.[[18]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-SJC_17.282.29-18) They described a hierarchy of attack models for signature schemes, and also presented the [GMR signature scheme](https://en.wikipedia.org/wiki/GMR_(cryptography)), the first that could be proved to prevent even an existential forgery against a chosen message attack.[[18]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-SJC_17.282.29-18)

How they work[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=4" \o "Edit section: How they work)]

To create RSA signature keys, generate a RSA key pair containing a modulus, *N*, that is the product of two large primes, along with integers, *e* and *d*, such that *e d* [≡](https://en.wikipedia.org/wiki/Modular_arithmetic) 1 (mod φ(*N*)), where φ is the [Euler phi-function](https://en.wikipedia.org/wiki/Euler%27s_totient_function). The signer's public key consists of *N* and *e*, and the signer's secret key contains *d*.

To sign a message, *m*, the signer computes a signature, σ, such that σ ≡ *md* (mod *N*). To verify, the receiver checks that σ*e* ≡ *m* (mod *N*).

As noted earlier, this basic scheme is not very secure. To prevent attacks, one can first apply a [cryptographic hash function](https://en.wikipedia.org/wiki/Cryptographic_hash_function) to the message, *m*, and then apply the RSA algorithm described above to the result. This approach is secure assuming the hash function is a [random oracle](https://en.wikipedia.org/wiki/Random_oracle_model).

Most early signature schemes were of a similar type: they involve the use of a [trapdoor permutation](https://en.wikipedia.org/wiki/Trapdoor_permutation), such as the RSA function, or in the case of the Rabin signature scheme, computing square modulo composite, *n.* A trapdoor permutation family is a family of [permutations](https://en.wikipedia.org/wiki/Permutation), specified by a parameter, that is easy to compute in the forward direction, but is difficult to compute in the reverse direction without already knowing the private key ("trapdoor"). Trapdoor permutations can be used for digital signature schemes, where computing the reverse direction with the secret key is required for signing, and computing the forward direction is used to verify signatures.

Used directly, this type of signature scheme is vulnerable to a key-only existential forgery attack. To create a forgery, the attacker picks a random signature σ and uses the verification procedure to determine the message, *m*, corresponding to that signature.[[19]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-19) In practice, however, this type of signature is not used directly, but rather, the message to be signed is first [hashed](https://en.wikipedia.org/wiki/Cryptographic_hash_function) to produce a short digest that is then signed. This forgery attack, then, only produces the hash function output that corresponds to σ, but not a message that leads to that value, which does not lead to an attack. In the random oracle model, this [hash-then-sign](https://en.wikipedia.org/wiki/Full_domain_hash) form of signature is existentially unforgeable, even against a [chosen-plaintext attack](https://en.wikipedia.org/wiki/Chosen-plaintext_attack).[[11]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-lysythesis-11)[[*clarification needed*](https://en.wikipedia.org/wiki/Wikipedia:Please_clarify)]

There are several reasons to sign such a hash (or message digest) instead of the whole document.

**For efficiency**

The signature will be much shorter and thus save time since hashing is generally much faster than signing in practice.

**For compatibility**

Messages are typically bit strings, but some signature schemes operate on other domains (such as, in the case of RSA, numbers modulo a composite number *N*). A hash function can be used to convert an arbitrary input into the proper format.

**For integrity**

Without the hash function, the text "to be signed" may have to be split (separated) in blocks small enough for the signature scheme to act on them directly. However, the receiver of the signed blocks is not able to recognize if all the blocks are present and in the appropriate order.

Notions of security[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=5" \o "Edit section: Notions of security)]

In their foundational paper, Goldwasser, Micali, and Rivest lay out a hierarchy of attack models against digital signatures:[[18]](https://en.wikipedia.org/wiki/Digital_signature" \l "cite_note-SJC_17.282.29-18)

1. In a *key-only* attack, the attacker is only given the public verification key.
2. In a *known message* attack, the attacker is given valid signatures for a variety of messages known by the attacker but not chosen by the attacker.
3. In an *adaptive chosen message* attack, the attacker first learns signatures on arbitrary messages of the attacker's choice.

They also describe a hierarchy of attack results:[[18]](https://en.wikipedia.org/wiki/Digital_signature" \l "cite_note-SJC_17.282.29-18)

1. A *total break* results in the recovery of the signing key.
2. A [universal forgery](https://en.wikipedia.org/wiki/Universal_forgery) attack results in the ability to forge signatures for any message.
3. A [selective forgery](https://en.wikipedia.org/wiki/Selective_forgery) attack results in a signature on a message of the adversary's choice.
4. An [existential forgery](https://en.wikipedia.org/wiki/Existential_forgery) merely results in some valid message/signature pair not already known to the adversary.

The strongest notion of security, therefore, is security against existential forgery under an adaptive chosen message attack.

Applications of digital signatures[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=6" \o "Edit section: Applications of digital signatures)]

As organizations move away from paper documents with ink signatures or authenticity stamps, digital signatures can provide added assurances of the evidence to provenance, identity, and status of an electronic document as well as acknowledging informed consent and approval by a signatory. The United States Government Printing Office (GPO) publishes electronic versions of the budget, public and private laws, and congressional bills with digital signatures. Universities including Penn State, [University of Chicago](https://en.wikipedia.org/wiki/University_of_Chicago), and Stanford are publishing electronic student transcripts with digital signatures.

Below are some common reasons for applying a digital signature to communications:

**Authentication**[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=7" \o "Edit section: Authentication)]

Although messages may often include information about the entity sending a message, that information may not be accurate. Digital signatures can be used to authenticate the source of messages. When ownership of a digital signature secret key is bound to a specific user, a valid signature shows that the message was sent by that user. The importance of high confidence in sender authenticity is especially obvious in a financial context. For example, suppose a bank's branch office sends instructions to the central office requesting a change in the balance of an account. If the central office is not convinced that such a message is truly sent from an authorized source, acting on such a request could be a grave mistake.

**Integrity**[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=8" \o "Edit section: Integrity)]

In many scenarios, the sender and receiver of a message may have a need for confidence that the message has not been altered during transmission. Although encryption hides the contents of a message, it may be possible to *change* an encrypted message without understanding it. (Some encryption algorithms, known as [nonmalleable](https://en.wikipedia.org/wiki/Malleability_(cryptography)) ones, prevent this, but others do not.) However, if a message is digitally signed, any change in the message after signature invalidates the signature. Furthermore, there is no efficient way to modify a message and its signature to produce a new message with a valid signature, because this is still considered to be computationally infeasible by most cryptographic hash functions (see [collision resistance](https://en.wikipedia.org/wiki/Collision_resistance)).

**Non-repudiation**[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=9" \o "Edit section: Non-repudiation)]

[Non-repudiation](https://en.wikipedia.org/wiki/Non-repudiation), or more specifically *non-repudiation of origin*, is an important aspect of digital signatures. By this property, an entity that has signed some information cannot at a later time deny having signed it. Similarly, access to the public key only does not enable a fraudulent party to fake a valid signature.

Note that these authentication, non-repudiation etc. properties rely on the secret key *not having been revoked* prior to its usage. Public revocation of a key-pair is a required ability, else leaked secret keys would continue to implicate the claimed owner of the key-pair. Checking revocation status requires an "online" check; e.g., checking a "[Certificate Revocation List](https://en.wikipedia.org/wiki/Certificate_Revocation_List)" or via the "[Online Certificate Status Protocol](https://en.wikipedia.org/wiki/Online_Certificate_Status_Protocol)". Very roughly this is analogous to a vendor who receives credit-cards first checking online with the credit-card issuer to find if a given card has been reported lost or stolen. Of course, with stolen key pairs, the theft is often discovered only after the secret key's use, e.g., to sign a bogus certificate for espionage purpose.

Additional security precautions[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=10" \o "Edit section: Additional security precautions)]

**Putting the private key on a smart card**[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=11" \o "Edit section: Putting the private key on a smart card)]

All public key / private key cryptosystems depend entirely on keeping the private key secret. A private key can be stored on a user's computer, and protected by a local password, but this has two disadvantages:

* the user can only sign documents on that particular computer
* the security of the private key depends entirely on the [security](https://en.wikipedia.org/wiki/Computer_insecurity) of the computer

A more secure alternative is to store the private key on a [smart card](https://en.wikipedia.org/wiki/Smart_card). Many smart cards are designed to be tamper-resistant (although some designs have been broken, notably by [Ross Anderson](https://en.wikipedia.org/wiki/Ross_J._Anderson_(professor)) and his students). In a typical digital signature implementation, the hash calculated from the document is sent to the smart card, whose CPU signs the hash using the stored private key of the user, and then returns the signed hash. Typically, a user must activate his smart card by entering a [personal identification number](https://en.wikipedia.org/wiki/Personal_identification_number) or PIN code (thus providing [two-factor authentication](https://en.wikipedia.org/wiki/Two-factor_authentication)). It can be arranged that the private key never leaves the smart card, although this is not always implemented. If the smart card is stolen, the thief will still need the PIN code to generate a digital signature. This reduces the security of the scheme to that of the PIN system, although it still requires an attacker to possess the card. A mitigating factor is that private keys, if generated and stored on smart cards, are usually regarded as difficult to copy, and are assumed to exist in exactly one copy. Thus, the loss of the smart card may be detected by the owner and the corresponding certificate can be immediately revoked. Private keys that are protected by software only may be easier to copy, and such compromises are far more difficult to detect.

**Using smart card readers with a separate keyboard**[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=12" \o "Edit section: Using smart card readers with a separate keyboard)]

Entering a PIN code to activate the smart card commonly requires a [numeric keypad](https://en.wikipedia.org/wiki/Numeric_keypad). Some card readers have their own numeric keypad. This is safer than using a card reader integrated into a PC, and then entering the PIN using that computer's keyboard. Readers with a numeric keypad are meant to circumvent the eavesdropping threat where the computer might be running a [keystroke logger](https://en.wikipedia.org/wiki/Keystroke_logging), potentially compromising the PIN code. Specialized card readers are also less vulnerable to tampering with their software or hardware and are often [EAL3](https://en.wikipedia.org/wiki/Evaluation_Assurance_Level) certified.

**Other smart card designs**[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=13" \o "Edit section: Other smart card designs)]

Smart card design is an active field, and there are smart card schemes which are intended to avoid these particular problems, though so far with little security proofs.

**Using digital signatures only with trusted applications**[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=14" \o "Edit section: Using digital signatures only with trusted applications)]

One of the main differences between a digital signature and a written signature is that the user does not "see" what he signs. The user application presents a hash code to be signed by the digital signing algorithm using the private key. An attacker who gains control of the user's PC can possibly replace the user application with a foreign substitute, in effect replacing the user's own communications with those of the attacker. This could allow a malicious application to trick a user into signing any document by displaying the user's original on-screen, but presenting the attacker's own documents to the signing application.

To protect against this scenario, an authentication system can be set up between the user's application (word processor, email client, etc.) and the signing application. The general idea is to provide some means for both the user application and signing application to verify each other's integrity. For example, the signing application may require all requests to come from digitally signed binaries.

**Using a network attached**[**hardware security module**](https://en.wikipedia.org/wiki/Hardware_security_module)[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=15)]

One of the main differences between a [cloud](https://en.wikipedia.org/wiki/Cloud) based digital signature service and a locally provided one is risk. Many risk averse companies, including governments, financial and medical institutions, and payment processors require more secure standards, like [FIPS 140-2](https://en.wikipedia.org/wiki/FIPS_140-2) level 3 and [FIPS 201](https://en.wikipedia.org/wiki/FIPS_201) certification, to ensure the signature is validated and secure.[[20]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-20)

**WYSIWYS**[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=16" \o "Edit section: WYSIWYS)]

*Main article:*[*WYSIWYS*](https://en.wikipedia.org/wiki/WYSIWYS)

Technically speaking, a digital signature applies to a string of bits, whereas humans and applications "believe" that they sign the semantic interpretation of those bits. In order to be semantically interpreted, the bit string must be transformed into a form that is meaningful for humans and applications, and this is done through a combination of hardware and software based processes on a computer system. The problem is that the semantic interpretation of bits can change as a function of the processes used to transform the bits into semantic content. It is relatively easy to change the interpretation of a digital document by implementing changes on the computer system where the document is being processed. From a semantic perspective this creates uncertainty about what exactly has been signed. [WYSIWYS](https://en.wikipedia.org/wiki/WYSIWYS) (What You See Is What You Sign) [[21]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-WYSIWYS_SeminalPaper-21) means that the semantic interpretation of a signed message cannot be changed. In particular this also means that a message cannot contain hidden information that the signer is unaware of, and that can be revealed after the signature has been applied. WYSIWYS is a necessary requirement for the validity of digital signatures, but this requirement is difficult to guarantee because of the increasing complexity of modern computer systems. The term WYSIWYS was coined by [Peter Landrock](https://en.wikipedia.org/wiki/Peter_Landrock) and [Torben Pedersen](https://en.wikipedia.org/wiki/Cryptomathic" \o "Cryptomathic) to describe some of the principles in delivering secure and legally binding digital signatures for Pan-European projects.[[21]](https://en.wikipedia.org/wiki/Digital_signature#cite_note-WYSIWYS_SeminalPaper-21)

**Digital signatures versus ink on paper signatures**[[edit](https://en.wikipedia.org/w/index.php?title=Digital_signature&action=edit&section=17" \o "Edit section: Digital signatures versus ink on paper signatures)]

An ink signature could be replicated from one document to another by copying the image manually or digitally, but to have credible signature copies that can resist some scrutiny is a significant manual or technical skill, and to produce ink signature copies that resist professional scrutiny is very difficult.

Digital signatures cryptographically bind an electronic identity to an electronic document and the digital signature cannot be copied to another document. Paper contracts sometimes have the ink signature block on the last page, and the previous pages may be replaced after a signature is applied. Digital signatures can be applied to an entire document, such that the digital signature on the last page will indicate tampering if any data on any of the pages have been altered, but this can also be achieved by signing with ink and numbering all pages of the contract.

What is a Digital Signature?

An introduction to Digital Signatures, by David Youd

|  |  |  |
| --- | --- | --- |
| http://www.youdzone.com/images/sig/face4.gif Bob | http://www.youdzone.com/images/sig/trans_half_inch.gif | http://www.youdzone.com/images/sig/greenkey.GIF (Bob's public key)  http://www.youdzone.com/images/sig/redkey.GIF (Bob's private key) |

Bob has been given two keys. One of Bob's keys is called a Public Key, the other is called a Private Key.

|  |  |  |
| --- | --- | --- |
| Bob's Co-workers: | | |
| http://www.youdzone.com/images/sig/face1.gif | http://www.youdzone.com/images/sig/face2.gif | http://www.youdzone.com/images/sig/face3.gif | http://www.youdzone.com/images/sig/trans_half_inch.gif | http://www.youdzone.com/images/sig/greenkey.GIF Anyone can get Bob's Public Key, but Bob keeps his Private Key to himself |
| Pat | Doug | Susan |  |  |

Bob's Public key is available to anyone who needs it, but he keeps his Private Key to himself. Keys are used to encrypt information. Encrypting information means "scrambling it up", so that only a person with the appropriate key can make it readable again. Either one of Bob's two keys can encrypt data, and the other key can decrypt that data.

Susan (shown below) can encrypt a message using Bob's Public Key. Bob uses his Private Key to decrypt the message. Any of Bob's coworkers might have access to the message Susan encrypted, but without Bob's Private Key, the data is worthless.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| http://www.youdzone.com/images/sig/face3.gif | http://www.youdzone.com/images/sig/trans_half_inch.gif | "Hey Bob, how about lunch at Taco Bell. I hear they have free refills!" | http://www.youdzone.com/images/sig/Encrypt_with_pub.gif | HNFmsEm6Un BejhhyCGKOK JUxhiygSBCEiC 0QYIh/Hn3xgiK BcyLK1UcYiY lxx2lCFHDC/A |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| http://www.youdzone.com/images/sig/face4.gif | http://www.youdzone.com/images/sig/trans_half_inch.gif | HNFmsEm6Un BejhhyCGKOK JUxhiygSBCEiC 0QYIh/Hn3xgiK BcyLK1UcYiY lxx2lCFHDC/A | http://www.youdzone.com/images/sig/Decrypt_with_pri.gif | "Hey Bob, how about lunch at Taco Bell. I hear they have free refills!" |

With his private key and the right software, Bob can put digital signatures on documents and other data. A digital signature is a "stamp" Bob places on the data which is unique to Bob, and is very difficult to forge. In addition, the signature assures that any changes made to the data that has been signed can not go undetected.

|  |  |  |
| --- | --- | --- |
| http://www.youdzone.com/images/sig/text.GIF | http://www.youdzone.com/images/sig/hash.gif | http://www.youdzone.com/images/sig/Message_digest.gif |

|  |  |
| --- | --- |
| http://www.youdzone.com/images/sig/face4.gif | To sign a document, Bob's software will crunch down the data into just a few lines by a process called "hashing". These few lines are called a message digest. (It is not possible to change a message digest back into the original data from which it was created.) |

|  |  |  |
| --- | --- | --- |
| http://www.youdzone.com/images/sig/Message_digest.gif | http://www.youdzone.com/images/sig/Encrypt_with_pri.gif | http://www.youdzone.com/images/sig/signature.gif |

Bob's software then encrypts the message digest with his private key. The result is the digital signature.

|  |  |  |
| --- | --- | --- |
| http://www.youdzone.com/images/sig/signature.gif | http://www.youdzone.com/images/sig/Append.gif | http://www.youdzone.com/images/sig/signed_text.GIF |

Finally, Bob's software appends the digital signature to document. All of the data that was hashed has been signed.

|  |  |  |
| --- | --- | --- |
| http://www.youdzone.com/images/sig/signed_text.GIF | http://www.youdzone.com/images/sig/hash.gif | http://www.youdzone.com/images/sig/Message_digest.gif |
| http://www.youdzone.com/images/sig/Decrypt_with_pub.gif | http://www.youdzone.com/images/sig/Message_digest.gif |

Bob now passes the document on to Pat.

|  |  |
| --- | --- |
| http://www.youdzone.com/images/sig/face1.gif | First, Pat's software decrypts the signature (using Bob's public key) changing it back into a message digest. If this worked, then it proves that Bob signed the document, because only Bob has his private key. Pat's software then hashes the document data into a message digest. If the message digest is the same as the message digest created when the signature was decrypted, then Pat knows that the signed data has not been changed. |

Plot complication...

|  |  |
| --- | --- |
| http://www.youdzone.com/images/sig/face2.gif | Doug (our disgruntled employee) wishes to deceive Pat. Doug makes sure that Pat receives a signed message and a public key that appears to belong to Bob. Unbeknownst to Pat, Doug deceitfully sent a key pair he created using Bob's name. Short of receiving Bob's public key from him in person, how can Pat be sure that Bob's public key is authentic? |

It just so happens that Susan works at the company's certificate authority center. Susan can create a digital certificate for Bob simply by signing Bob's public key as well as some information about Bob.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  | | --- | | Bob Info:      Name      Department      Cubical Number  Certificate Info:      Expiration Date      Serial Number  Bob's Public Key:      http://www.youdzone.com/images/sig/greenkey.GIF | | http://www.youdzone.com/images/sig/trans_half_inch.gif | http://www.youdzone.com/images/sig/sign_data.gif http://www.youdzone.com/images/sig/face3.gif | http://www.youdzone.com/images/sig/trans_half_inch.gif | http://www.youdzone.com/images/sig/certificate.gif |

Now Bob's co-workers can check Bob's trusted certificate to make sure that his public key truly belongs to him. In fact, no one at Bob's company accepts a signature for which there does not exist a certificate generated by Susan. This gives Susan the power to revoke signatures if private keys are compromised, or no longer needed. There are even more widely accepted certificate authorities that certify Susan.

Let's say that Bob sends a signed document to Pat. To verify the signature on the document, Pat's software first uses Susan's (the certificate authority's) public key to check the signature on Bob's certificate. Successful de-encryption of the certificate proves that Susan created it. After the certificate is de-encrypted, Pat's software can check if Bob is in good standing with the certificate authority and that all of the certificate information concerning Bob's identity has not been altered.

Pat's software then takes Bob's public key from the certificate and uses it to check Bob's signature. If Bob's public key de-encrypts the signature successfully, then Pat is assured that the signature was created using Bob's private key, for Susan has certified the matching public key. And of course, if the signature is valid, then we know that Doug didn't try to change the signed content.