What is a digital signature?

A digital signature (not to be confused with a [digital certificate](https://searchsecurity.techtarget.com/definition/digital-certificate)) is a mathematical technique used to validate the authenticity and integrity of a message, software or digital document.

A **digital signature** is basically a way to ensure that an electronic document (e-mail, spreadsheet, text file, etc.) is **authentic**. Authentic means that you know who created the document and you know that it has not been altered in any way since that person created it.

Digital signatures rely on certain types of **encryption** to ensure authentication. Encryption is the process of taking all the data that one computer is sending to another and encoding it into a form that only the other computer will be able to decode. Authentication is the process of verifying that information is coming from a trusted source. These two processes work hand in hand for digital signatures.

There are several ways to authenticate a person or information on a computer:

**Password** - The use of a user name and password provide the most common form of authentication. You enter your name and password when prompted by the computer. It checks the pair against a secure file to confirm. If either the name or password do not match, then you are not allowed further access.

**Checksum** - Probably one of the oldest methods of ensuring that data is correct, checksums also provide a form of authentication since an invalid checksum suggests that the data has been compromised in some fashion. A checksum is determined in one of two ways. Let's say the checksum of a packet is 1 byte long, which means it can have a maximum value of 255. If the sum of the other bytes in the packet is 255 or less, then the checksum contains that exact value. However, if the sum of the other bytes is more than 255, then the checksum is the remainder of the total value after it has been divided by 256. Look at this example:

* Byte 1 = 212
* Byte 2 = 232
* Byte 3 = 54
* Byte 4 = 135
* Byte 5 = 244
* Byte 6 = 15
* Byte 7 = 179
* Byte 8 = 80
* **Total = 1151**. 1151 divided by 256 equals 4.496 (round to 4). Multiply 4 X 256 which equals 1024. 1151 minus 1024 **equals checksum of 127**

**CRC (Cyclic Redundancy Check)** - CRCs are similar in concept to checksums but they use polynomial division to determine the value of the CRC, which is usually 16 or 32 bits in length. The good thing about CRC is that it is very accurate. If a single bit is incorrect, the CRC value will not match up. Both checksum and CRC are good for preventing random errors in transmission, but provide little protection from an intentional attack on your data. The encryption techniques below are much more secure.

**Private key encryption** -Private key means that each computer has a secret key (code) that it can use to encrypt a packet of information before it is sent over the network to the other computer. Private key requires that you know which computers will talk to each other and install the key on each one. Private key encryption is essentially the same as a secret code that the two computers must each know in order to decode the information. The code would provide the key to decoding the message. Think of it like this. You create a coded message to send to a friend where each letter is substituted by the letter that is second from it. So "A" becomes "C" and "B" becomes "D". You have already told a trusted friend that the code is "Shift by 2". Your friend gets the message and decodes it. Anyone else who sees the message will only see nonsense.

**Public key encryption** - Public key encryption uses a combination of a private key and a public key. The private key is known only to your computer while the public key is given by your computer to any computer that wants to communicate securely with it. To decode an encrypted message, a computer must use the public key provided by the originating computer and it's own private key.

The key is based on a hash value. This is a value that is computed from a base input number using a hashing algorithm. The important thing about a hash value is that it is nearly impossible to derive the original input number without knowing the data used to create the hash value. Here's a simple example:

**Input number 10667**

**Hashing Algorithm = Input # x 143**

**Hash Value = 1525381**

You can see how hard it would be to determine that the value of 1525381 came from the multiplication of 10667 and 143. But if you knew that the multiplier was 143, then it would be very easy to calculate the value of 10667. Public key encryption is much more complex than this example but that is the basic idea. Public keys generally use complex algorithms and very large hash values for encrypting: 40-bit or even 128-bit numbers. A 128-bit number has a possible 2128 different combinations. That's as many combinations as there are water molecules in 2.7 million olympic size swimming pools. Even the tiniest water droplet you can image has billions and billions of water molecules in it!

**Digital certificates** - To implement public key encryption on a large scale, such as a secure Web server might need, requires a different approach. This is where digital certificates come in. A digital certificate is essentially a bit of information that says the Web server is trusted by an independent source known as a **Certificate Authority**. The Certificate Authority acts as the middleman that both computers trust. It confirms that each computer is in fact who they say they are and then provides the public keys of each computer to the other.

The **Digital Signature Standard (DSS)** is based on a type of public key encryption method that uses the **Digital Signature Algorithm (DSA)**. DSS is the format for digital signatures that has been endorsed by the US government. The DSA algorithm consists of a private key that only the originator of the document (signer) knows and a public key. The public key has four parts, which you can learn more about at [this page](http://www.tatanka.com/doc/technote/tn0035.htm).

Electronic payment could become the future of currency. Click here to learn how digital signatures could help secure the future of [electronic payment](https://money.howstuffworks.com/personal-finance/online-banking/electronic-payment.htm).

Here are some interesting links:

* [Bionic Buffalo Tech #5: Encryption](http://www.tatanka.com/doc/technote/tn0035.htm)
* [Digital Signature Initiative](http://www.w3.org/DSig/)
* [Links on Law, Cryptography and Electronic Communications](http://www.qmw.ac.uk/~tl6345/)

## **Create a digital signature**

So, you’ve been asked to digitally sign a document? No problem. It’s easy to create digital signatures.

Digital signatures are created online and applied to online documents, rather than using a pen to write your signature physically (also known as a “wet signature”). [Digital signatures provide an extra level of security](https://www.docusign.com/how-it-works/electronic-signature/digital-signature/digital-signature-faq) by using technology that encrypts the signature, verifying the identity of the person signing.

When you receive an email with a link to the document you need to digitally sign, here’s what to do:

1. **Click the link**. Your document should open in an electronic signature tool such as DocuSign.
2. **Agree to electronic signing**. You may be asked to agree to sign. After confirming agreement and if the document was sent via DocuSign, you should see tags with instructions to Start or Sign.
3. **Click each tag** and follow the instructions to add your digital signature.
4. **Verify your identity** and follow the instructions to add your digital signature.