Signed and Secure Communications

Allen Baker

February 5, 2004

How can you trust files that you download from the Internet? How do you know a file really is coming from who it says it's coming from? How can you be sure that the file has not been altered maliciously between the time it was created and the time you download it? These are particularly important questions when you are downloading software to run on your computer and which will have access to the computer's file system and other resources. This paper describes the most common method of making the Internet a safe and reliable source for software distribution.

Users frequently encounter situations where they need to download software or applications. For example, a user might not be planning to download software but is unexpectedly advised by a program they are using or the site they are visiting, that they need to upgrade, patch or augment their program. They are asked to make a spot decision: "Run" or "Don't Run."



In these situations, "run/don't run" asks the user whether or not to run the downloaded code. How does a user decide? How does a user or user agent (usually a "browser") know whether or not to trust the software?

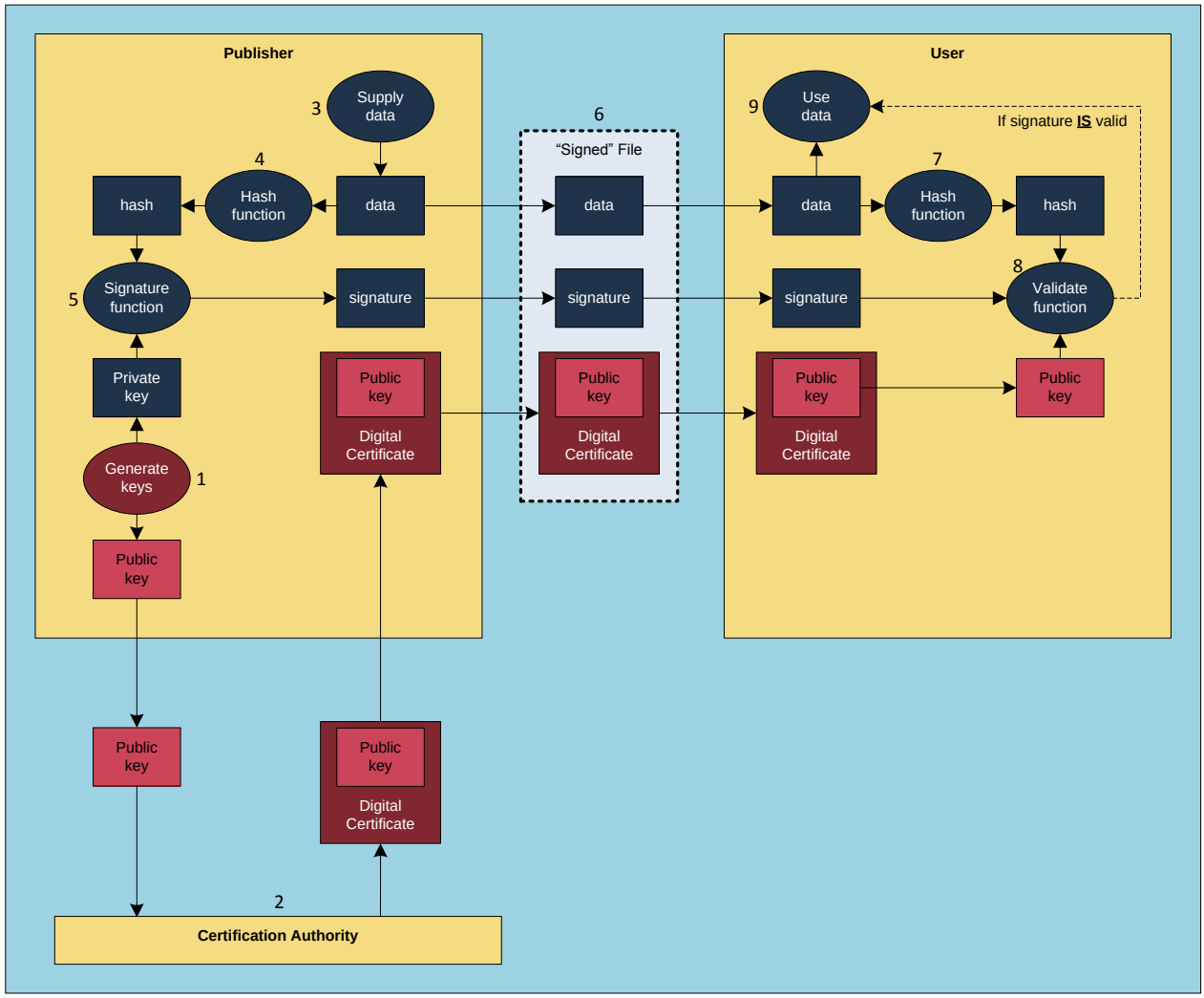
A number of techniques are in use to address these questions. The most secure technique in common use on the Internet is called "code signing". Code signing does two things to help ensure the safety of downloaded software files:

Ensures authenticity	Assures users that they know where the software download came from.
Ensures integrity	Verifies that the software download has not been tampered with since its publication.

Although code signing, like any security technique, cannot guarantee that the software you download will not harm your computer, it makes it very difficult for nefarious Internet actors to corrupt the software provided by honest software publishers or to distribute dangerous software of their own. It identifies the publisher so they can be held accountable and it ensures that the software you receive is an exact copy of the software that the publisher published. Signed code, which is backed by a certificate issued by a Certificate Authority acting as a trusted third party, is granted greater trust than unsigned code, which should generally not be trusted.

How It Works

As you can see from this diagram, code signing involves several steps both at the publisher when they make the software available for download and on your computer when you download the software.



The process begins on the Publisher's side:

1. The Publisher generates a public-private key pair using a public-key algorithm such as the RSA public-key cipher and sends the public key to a Certification Authority (CA).

The private key is known only to its owner, while the public key can be available to anyone. Public-key algorithms are designed so that if one key is used for encryption, the other is necessary for decryption. Furthermore, the decryption key cannot be reasonably calculated from the encryption key.

2. The CA verifies the identity of the Publisher, authenticates the Publisher's certificate request, creates a digital certificate containing the Publisher's public key, and delivers the certificate back to the Publisher.

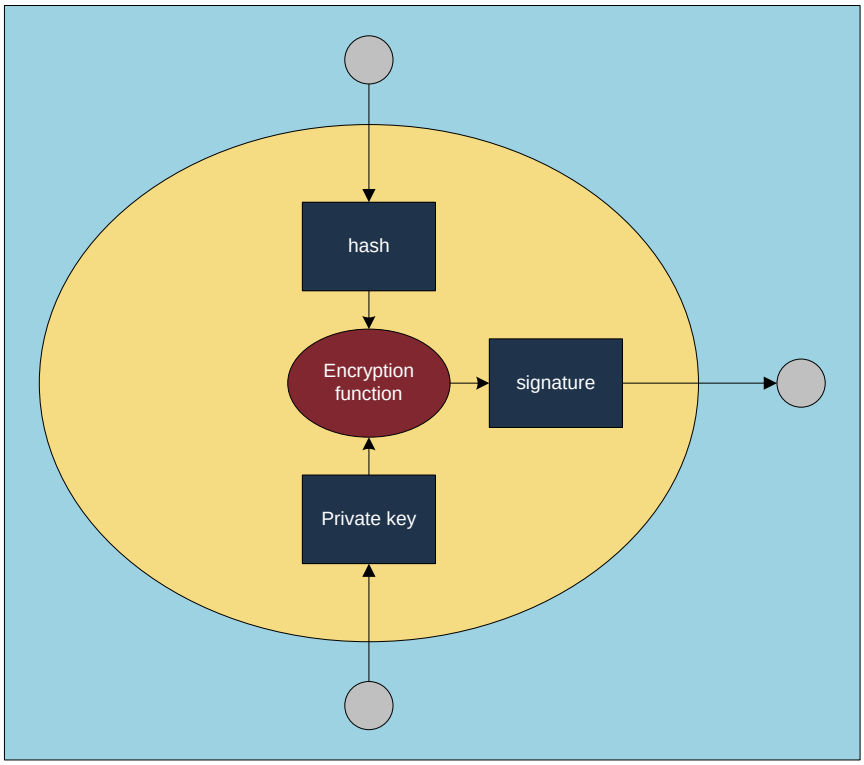
   CAs are trustworthy persons or organizations that issue certificates to applicants whose identity has in some way been verified by the certification authority. Certificates are verified through a hierarchy of these certification authorities. Each certificate is linked to the certificate of the certification authority that signed it. By following this hierarchy, or verification path, to a known, trusted certification authority, you can be assured that a certificate is valid.

3. Now the Publisher is ready to begin publishing software for distribution over the Internet. The Publisher produces the data (in this case, the software).

4. The Publisher generates a one-way hash of the software file.

   A one-way hash of the file is a cryptographic digest of the file's contents. It is a cryptographically unique representation of the file and since it is one-way, it is only reproducible using the unaltered file and the hashing algorithm that was used to create the hash. It is actually the hash that is signed instead of the file itself.

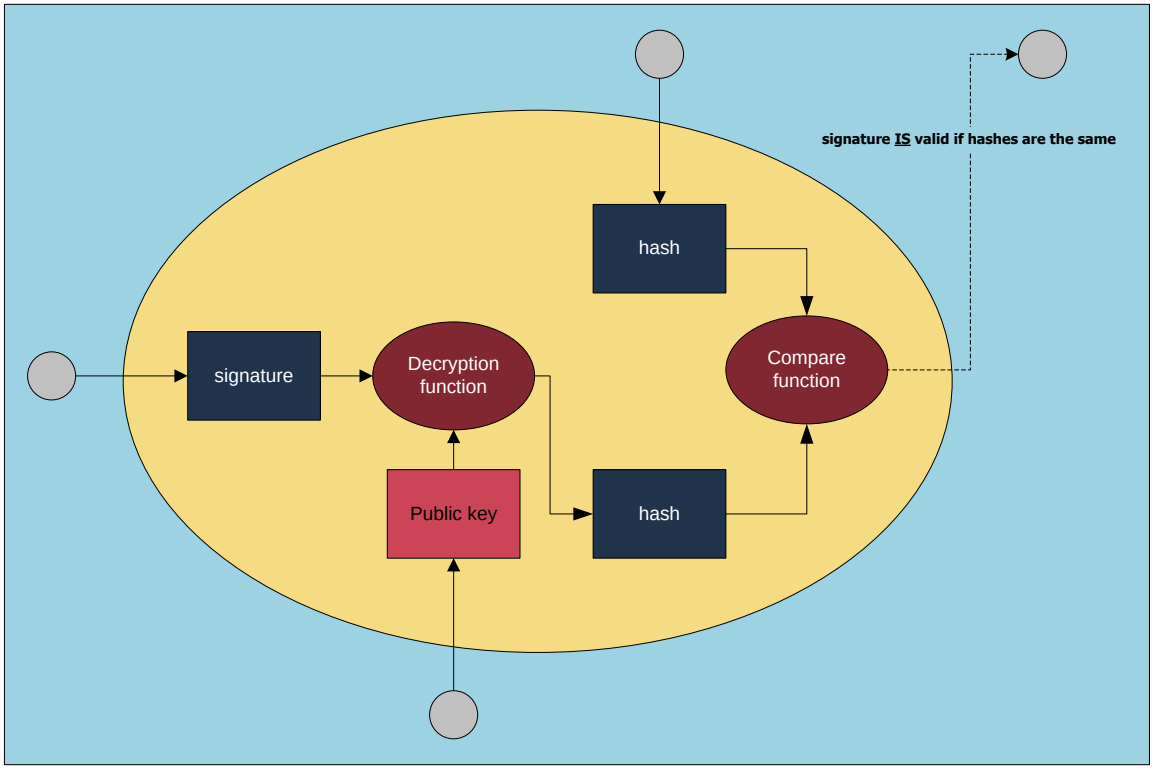
5. The Publisher signs the hash using their private key as the encryption key as shown here.



6. The Publisher then generates a signed file by placing the data (the software in this case), the signature, and a copy of the digital certificate containing the Publisher's public key into a single file. This is the file that the user downloads over the Internet. The three components inside the signed file are extracted into three separate files on the user's computer.

7. Using the exact same one-way hash algorithm as was used by the Publisher to create a hash of the software file, the User's computer generates a one-way hash of the software file it extracted from the signed file.

8. The User's computer validates the signature it extracted from the signed file using the Publisher's public key from the certificate in the signed file and the hash the User's computer generated in step 7 above. It decrypts the signature using the Publishers public key as the decryption key. This recreates the hash that the Publisher had created in step 4 above. If the recreated hash is identical to the hash that the User's computer generated in step 7, then the software is deemed safe to use.



9. The User can now run the software, confident that it really came from the Publisher and that it was not tampered with after the Publisher made it available on the Internet.

The value of code signing is knowing who authored the file to be downloaded, and that the bits have not been altered from the time the code was signed to the present. It provides assurances of authenticity and origin. When a Publisher uses code signing, the most important step for establishing end-user trust is that the signed code is backed by a code signing certificate issued by a trusted Certification Authority.