An Introduction to Digital Signatures
(and Encryption)

Colin Tuckley

November 2006

Who am I?

- A freelance IT Consultant
- A Chartered Engineer
- Member of the BCS
- Specialist in embedded and realtime systems
- Open Source developer
- An advocate for privacy on the Internet

Introduction

- What are they?
- Why use them?
- How do they work?
- Keeping things confidential
- Why use public key systems?
- Theory & Implementation
- Practical systems
- The “Web of Trust”

What is a Digital Signature?

A digital signature is a term used for marking or signing an electronic document by a process meant to be analogous to paper signatures, but which makes use of a technology known as public-key cryptography.

What is a Digital Signature?

A digital signature is a term used for marking or signing an electronic document by a process meant to be analogous to paper signatures, but which makes use of a technology known as public-key cryptography.

In other words, it's a digital code that can be attached to an electronically transmitted message that uniquely identifies the sender and ensures that the document has not been altered in any way since the sender signed it.

Why Should I Use them?

The processes of creating a digital signature and verifying it accomplish the essential effects desired of a signature for many purposes:

- Signer authentication.
- Message authentication.
- Affirmative act.
- Efficiency.

Why Should I Use them?

- Signer authentication.

If a key is associated with an identified signer, the digital signature attributes the message to the signer. The digital signature cannot be forged unless the signer loses control of the key (the "compromise" of the key), e.g., by divulging it or losing the media or device in which it is contained.

Why Should I Use them?

Message authentication.

The digital signature identifies the signed message with a much greater certainty and precision than paper signatures. Verification reveals any tampering, since the comparison of the hash results (one made at signing and the other made when verifying) shows whether the message is the same as when signed.

Why Should I Use them?

Affirmative act.

Creating a digital signature requires the signer to use the signers private key. This act can perform the "ceremonial" function of alerting the signer to the fact that he is making a transaction with legal consequences.

Why Should I Use them?

Efficiency.

The processes of creating and verifying a digital signature provide a high level of assurance that the digital signature is genuinely the signers. For modern electronic data interchange ("EDI"), the creation and verification processes are capable of being completely automated.

Compared to paper methods such as checking specimen signature cards - methods so tedious and labor-intensive that they are rarely used in practice - digital signatures yield a high degree of assurance without adding greatly to the resources required for processing.

How do they work?

- Digital signatures are made with “keys”. Each key is in two parts, a public key and a private or secret key.
- The public key is distributed widely (made public).
- The private key is kept confidential, if it becomes known to anyone other than it's owner then it becomes useless for security purposes and is “compromised”.
- To ensure that compromised and retired keys can be identified as such a “revocation certificate” can be attached to the public key.

How do they work?

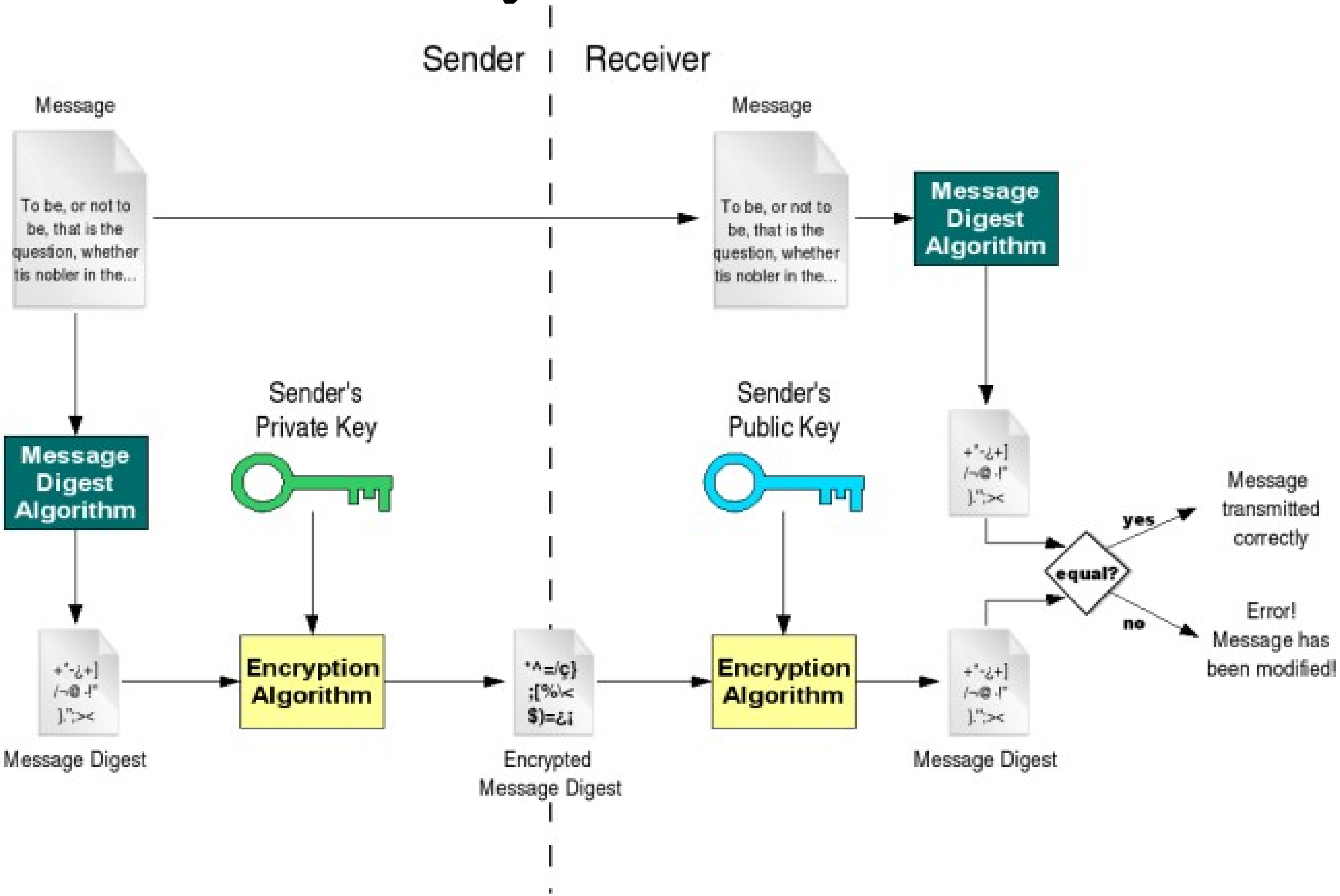
The sender of a message signs it with his private key.

The recipient of a message verifies it with the senders public key.

- confirms source of message.
- ensures contents of message are not tampered with.
- enforces “non-repudiation” (the sender can't deny that he sent the message).

Digital Signatures

How do they work?



Keeping things confidential

Often we wish to keep the contents of our messages confidential. When using the postal service we put our message in an envelope instead of sending a postcard.

Keeping things confidential

Digital messages pass through many computers and other equipment before they reach the intended recipient

Ordinary messages are like postcards, any casual observer can read them.

Keeping things confidential

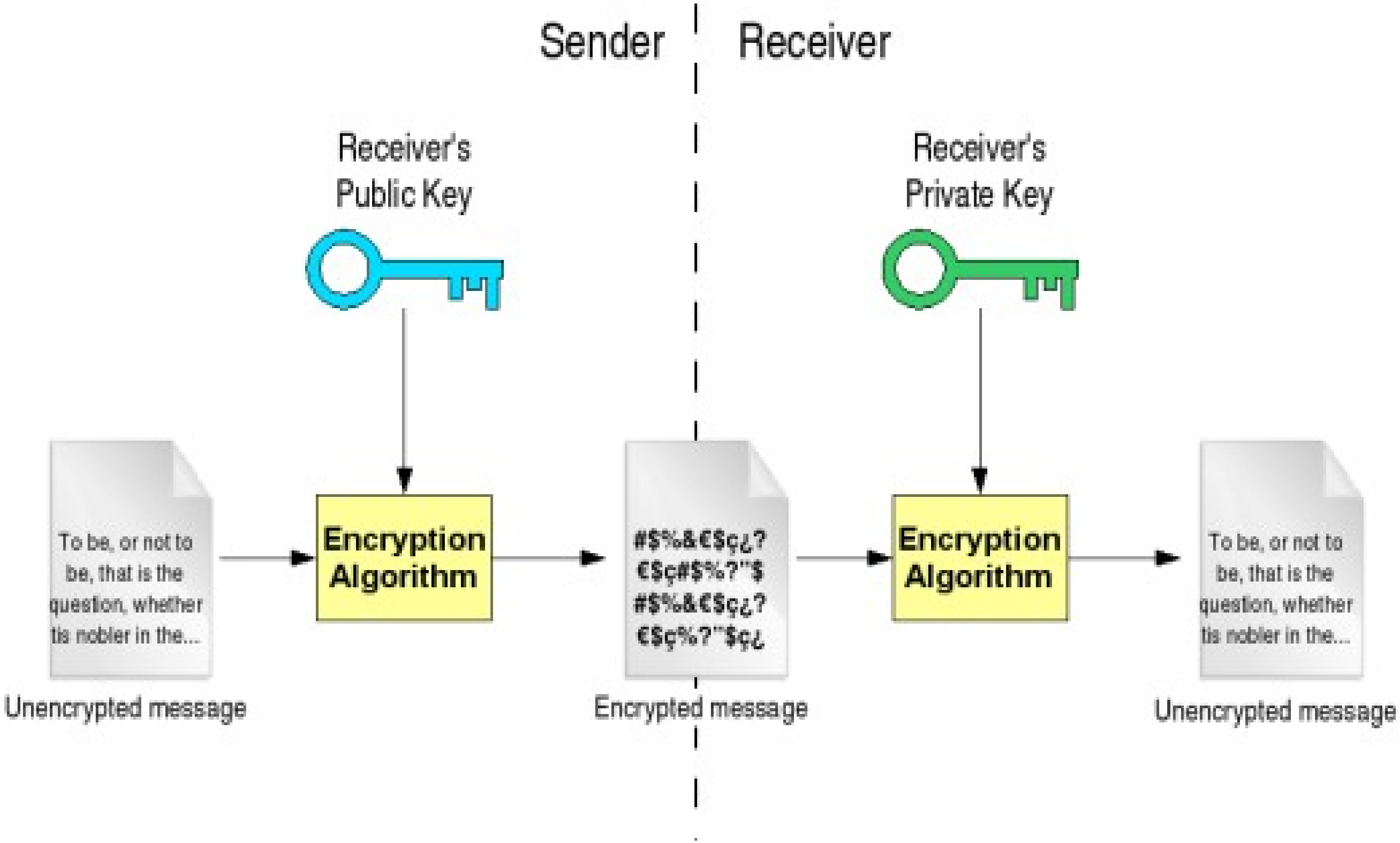
How do we prevent other people from reading our mail?

We can encrypt the message with the public key of the intended recipient. Since the private key is required to decrypt the message only the recipient can do this.

Keeping things confidential

- If we wish to send the message to more than one person then it can be encrypted with more than one public key – each of the matching private keys will be able to decrypt the message.
- We can sign the message as well as encrypting it, thus allowing the recipient to confirm the senders identity.

Keeping things confidential



Why use public-key systems?

Why indeed, when symmetric (common/shared key) systems are much faster.

- Key distribution
- Number of keys required
- The speed problem

Why use public-key systems?

Key distribution

- For a sender and recipient to communicate securely using conventional encryption, they must agree upon a shared key and keep it secret between themselves. If they are in different physical locations, they must trust a courier or some other secure communication medium to prevent the disclosure of the shared key during transmission.

Why use public-key systems?

Key distribution (contd)

- Anyone who overhears or intercepts the key in transit can later read, modify, and forge all information encrypted or authenticated with that key.
- So, from DES to Captain Midnight's Secret Decoder Ring, the persistent problem with conventional encryption is key distribution: how do you get the key to the recipient without someone intercepting it?

Why use public-key systems?

Number of Keys required

- Each pair of people who wish to communicate secretly with symmetric key systems need a unique key.
  - 4 people need 6 keys
  - 10 people need 45 keys!

Why use public-key systems?

The speed problem

Public-key operations are much slower than symmetric-key operations.

- Encryption is 60 times slower
- Decryption is 1000 times slower

Why use public-key systems?

We can use a hybrid method to solve the speed problem.

- generate a symmetric “session” key
- encrypt the (large) message with it
- encrypt the (small) session key with the recipients public key
- bundle both parts together and send them
- the recipient can use his private key to decrypt the session key
- the recipient uses the session key to decrypt the message

Theory & Implementation

Public Key Cryptography codes are based on trap-door mathematical functions.

Like trap doors, it is easy to go in one direction (falling through the trap door), but it is very hard to go in the other direction (climbing out after you've fallen in).

Theory & Implementation

The first system (called RSA after its inventors Ronald **R**ivest, Adi **S**hamir, and Leonard **A**dleman) is based on the fact that multiplication is very easy (especially for a computer), while factoring (determining which two numbers have been multiplied together to produce a given product) is very hard.

Theory & Implementation

For a small number, factoring is relatively easy. If you were asked say, what are the factors of 35, it wouldn't take long to come up with seven and five. But for very large numbers, factoring takes more time than even the fastest computers have. By some estimates, trying all the possible factors of a 150-digit number would take millions of years.

It is an indication of how secure experts believe the system to be, that government agencies have tried to outlaw it's use.

Theory & Implementation

RSA's implementation of Diffie's idea of public-key cryptography is based on "modular exponentiation modulo the product of two large primes."

Practical systems

There are two main systems in common use.

- x509 Certificates & Certificate Authorities
- PGP/GnuPG & the “Web of Trust”

Both use the same underlying public-key cryptography mechanism. The difference between them is the way in which “trust” is determined.

Practical systems

x509 Certificates & Certificate Authorities

This approach, which is most favored by governments and other hierarchical entities, uses formal certificate authorities (or CAs).

To associate a key pair with a prospective signer, a certification authority issues a certificate, an electronic record, which lists a public key as the "subject" of the certificate, and confirms that the prospective signer identified in the certificate holds the corresponding private key.

Practical systems

PGP/GnuPG & the “Web of Trust”

The PGP/GnuPG approach allows anyone to vouch for anyone else's identity. It is up to you to decide who to trust. You must decide who to believe when a statement is made that a key belongs to a certain person.

The “Web of Trust”

If someone you trust introduces someone else by vouching for the authenticity of his key, then you are more inclined to believe it than if you were introduced by a stranger.

The “Web of Trust”

In this approach, one person can sign another person's key, as a statement that the key belongs to the ostensive owner.

If two people sign each others keys then the trust path is bidirectional.

The “Web of Trust”

If person A needs to trust person C but has never met them then it's possible that both have exchanged signatures with a third person B.

A “trust path” in the web of trust then exists between them via person B.

The “Web of Trust”

An example:

```
PGP trust paths : Colin Tuckley -> Richard Stallman

from Colin Tuckley <colin.at.tuckley.org> 1B3045CE

to   Richard Stallman (Chief GNUisance) <rms.at.gnu.org>    135EA668


0  1B3045CE  Colin Tuckley <colin.at.tuckley.org> #826

1  68FD549F  Martin Michlmayr <tbm.at.cyrius.com> #3

2  135EA668  Richard Stallman (Chief GNUisance) <rms.at.gnu.org> #658


0  1B3045CE  Colin Tuckley <colin.at.tuckley.org> #826

1  9D928C9B  Guillem Jover <guillem.at.hadrons.org> #389

2  8BAFCDBD  Neal H Walfield <neal.at.gnu.org> #740

3  135EA668  Richard Stallman (Chief GNUisance) <rms.at.gnu.org> #658
```