Symmetric Algorithms

Symmetric Encryption

- Symmetric, or secret key, encryption is the most commonly used form of cryptography, because the shorter key length increases the speed of execution.
  - Symmetric key algorithms are based on simple mathematical operations that can easily be accelerated by hardware.
  - Symmetric encryption is often used for wire-speed encryption in data networks and to provide bulk encryption when data privacy is required, such as to protect a VPN.

Symmetric Key Management

- Key management can be a challenge since the encryption and decryption keys are the same.
- The security of a symmetric algorithm rests in the secrecy of the symmetric key.
  - By obtaining the key, anyone can encrypt and decrypt messages.
  - Sender and receiver must exchange the secret key using a secure channel before any encryption can occur.
Symmetric Key Management

- Well-known encryption algorithms that use symmetric keys including:
  - DES
  - 3DES
  - AES
  - Software Encryption Algorithm (SEAL)
  - Rivest ciphers (RC) series (RC2, RC4, RC5, and RC6)

- Other symmetric encryption algorithms include Blowfish, Twofish, Threefish, and Serpent.

Symmetric Encryption Algorithms

<table>
<thead>
<tr>
<th>Symmetric Encryption Algorithm</th>
<th>Key length</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DES</td>
<td>56</td>
<td>Designed at IBM during the 1970s and adopted as the NSA standard until 1997. Although considered outdated, DES remains readily in use. Although it is recommended that it be replaced by other algorithms, it is still used by some organizations. DES was designed to be implemented only in hardware, and no efficient software implementation is available.</td>
</tr>
<tr>
<td>3DES</td>
<td>112 and 168</td>
<td>Based on using DES three times which means that the input data is encrypted three times and the three keys combined using DES. It is considered much stronger than DES, which means that it is safe for use in today's computing environment.</td>
</tr>
<tr>
<td>AES</td>
<td>128, 192, and 256</td>
<td>AES is often used in hardware and software. It is relatively easy to implement, and requires little memory.</td>
</tr>
<tr>
<td>Software Encryption Algorithm (SEAL)</td>
<td>160</td>
<td>SEAL is an alternative algorithm to DES, 3DES, and AES. It uses a 160-bit encryption key and has a lower impact to the CPU when compared to other software-based algorithms.</td>
</tr>
<tr>
<td>The RC series</td>
<td></td>
<td>RC algorithms use a set of symmetric-key encryption algorithms created by Ron Rivest. RC2 and RC6 are widely used, while RC4 is the one used by the Internet Engineering Task Force. RC5 is an RC algorithm for which a public key version, called RC5E, was an AES finalist developed in 1997.</td>
</tr>
</tbody>
</table>

Symmetric Encryption Techniques

- There are two types of encryption method used:
  - Block Ciphers
  - Stream Ciphers
Block Ciphers

- Block ciphers transform a fixed-length block of plaintext into a common block of ciphertext of 64 or 128 bits.
  - Block size refers to how much data is encrypted at any one time.
  - The key length refers to the size of the encryption key that is used.
  - This ciphertext is decrypted by applying the reverse transformation to the ciphertext block, using the same secret key.
- Common block ciphers include:
  - DES with a 64-bit block size
  - AES with a 128-bit block size
  - RSA with a variable block size

Stream Ciphers

- Stream ciphers encrypt plaintext one byte or one bit at a time.
  - Think of it like a block cipher with a block size of one bit.
  - The Vigenère cipher is an example of a stream cipher.
  - Can be much faster than block ciphers, and generally do not increase the message size.
- Common stream ciphers include:
  - A5 used to encrypt GSM cell phone communications.
  - RC4 cipher.
  - DES can also be used in stream cipher mode.

How to Choose an Encryption Algorithm?

- Is the algorithm trusted by the cryptographic community?
  - Algorithms that have been resisting attacks for a number of years are preferred.
- Does the algorithm adequately protect against brute-force attacks?
  - With the appropriate key lengths, these attacks are usually considered unfeasible.
- Does the algorithm support variable and long key lengths?
- Does the algorithm have export or import restrictions?
How to Choose an Encryption Algorithm?

<table>
<thead>
<tr>
<th>DES</th>
<th>3DES</th>
<th>AES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is the algorithm trusted by the cryptographic community?</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Does the algorithm adequately protect against brute-force attacks?</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Data Encryption Standard (DES)

- The most popular symmetric encryption standards.
  - Developed by IBM
  - Thought to be unbreakable in the 1970s
  - Shared keys enable the encryption and decryption
- DES converts blocks of 64-bits of clear text into ciphertext by using an encryption algorithm.
  - The decryption algorithm on the remote end restores ciphertext to clear text.

DES Scorecard

<table>
<thead>
<tr>
<th>Description</th>
<th>Data Encryption Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeline</td>
<td>Standardized 1976</td>
</tr>
<tr>
<td>Type of Algorithm</td>
<td>Symmetric</td>
</tr>
<tr>
<td>Key size (in bits)</td>
<td>56 bits</td>
</tr>
<tr>
<td>Speed</td>
<td>Medium</td>
</tr>
<tr>
<td>Time to crack (assuming a computer and 2^32 keys per second)</td>
<td>Days (0.4 days by the COPACABANA machine, a specialized cracking device)</td>
</tr>
<tr>
<td>Resource Consumption</td>
<td>Medium</td>
</tr>
</tbody>
</table>
Because of its short key length, DES is considered a good protocol to protect data for a very short time.  
3DES is a better choice to protect data because it has an algorithm that is very trusted and has higher security strength.

Recommendations:
- Change keys frequently to help prevent brute-force attacks.
- Use a secure channel to communicate the DES key from the sender to the receiver.

3DES is 256 times stronger than DES.
It takes a 64-bit block of data and performs three DES operations in sequence:
- Encrypts, decrypts, and encrypts.
- Requires additional processing time.
- Can use 1, 2, or 3 different keys (when used with only one key, it is the same as DES).
3DES software is subject to US export laws.

**3DES Scorecard**

<table>
<thead>
<tr>
<th>Description</th>
<th>Triple Data Encryption Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeline</td>
<td>Standardized 1977</td>
</tr>
<tr>
<td>Type of Algorithm</td>
<td>Symmetric</td>
</tr>
<tr>
<td>Key size (in bits)</td>
<td>112 and 168 bits</td>
</tr>
<tr>
<td>Speed</td>
<td>Low</td>
</tr>
<tr>
<td>Time to crack</td>
<td>4.6 Billion years with current technology</td>
</tr>
<tr>
<td>Resource Consumption</td>
<td>Medium</td>
</tr>
</tbody>
</table>

Andreas Ahrens  
July 2013, Vilnius, Lithuania
Although 3DES is very secure, it is also very resource intensive and for this reason the AES encryption algorithm was developed.

- AES has proven to be as secure as 3DES, but with much faster results.

AES is an extremely secure Federal Information Processing Standard (FIPS)-approved cryptographic algorithm.
- Based on the Rijndael ("Rhine dahl") algorithm.
- It uses keys with a length of 128, 192, or 256 bits to encrypt blocks with a length of 128, 192, or 256 bits.
  - All 9 combinations of key length and block length are possible.
- AES is now available in the latest Cisco router images that have IPSec DES/3DES functionality.
**AES Scorecard**

<table>
<thead>
<tr>
<th>Description</th>
<th>Advanced Encryption Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeline</td>
<td>Official Standard since 2001</td>
</tr>
<tr>
<td>Type of Algorithm</td>
<td>Symmetric</td>
</tr>
<tr>
<td>Key size (in bits)</td>
<td>128, 192, and 256</td>
</tr>
<tr>
<td>Speed</td>
<td>High</td>
</tr>
<tr>
<td>Time to crack</td>
<td>149 Trillion years</td>
</tr>
<tr>
<td>Resource Consumption</td>
<td>Low</td>
</tr>
</tbody>
</table>

**AES Example**

In this example, the **SECRETKEY** key and plaintext are entered.

They are now encrypted using 128-AES.

An attempt at deciphering the text using a lowercase, and incorrect key.

A second attempt at deciphering the text using the correct key displays the original plaintext.

**AES**

- AES was chosen to replace DES for a number of reasons:
  - The key length of AES makes the key much stronger than DES.
  - AES runs faster than 3DES on comparable hardware.
  - AES is more efficient than DES and 3DES on comparable hardware, usually by a factor of five when it is compared with DES.
  - AES is more suitable for high-throughput, low-latency environments, especially if pure software encryption is used.
- However, AES is a relatively young algorithm and the golden rule of cryptography states that a mature algorithm is always more trusted.
- 3DES is therefore a more trusted choice in terms of strength, because it has been tested and analyzed for 35 years.
The Software-optimized Encryption Algorithm (SEAL) is an alternative algorithm to software-based DES, 3DES, and AES.
- Designed in 1993, it is a stream cipher that uses a 160-bit encryption key.
- Because it is a stream cipher, data to be encrypted is continuously encrypted and, therefore, much faster than block ciphers.
- However, it has a longer initialization phase during which a large set of tables is created using SHA.
- SEAL has a lower impact on the CPU compared to other software-based algorithms.
- SEAL support was added to Cisco IOS Software Release 12.3(7)T.

SEAL Scorecard

<table>
<thead>
<tr>
<th>Description</th>
<th>Software-Optimized Encryption Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeline</td>
<td>First published in 1994, Current version is 3.0 (1997)</td>
</tr>
<tr>
<td>Type of Algorithm</td>
<td>Symmetric</td>
</tr>
<tr>
<td>Key size (in bits)</td>
<td>160</td>
</tr>
<tr>
<td>Speed</td>
<td>High</td>
</tr>
<tr>
<td>Time to crack</td>
<td>Unknown but considered very safe</td>
</tr>
<tr>
<td>Resource Consumption</td>
<td>Low</td>
</tr>
</tbody>
</table>

RC Algorithms

- The RC algorithms were designed all or in part by Ronald Rivest, who also invented MD5.
- The RC algorithms are widely deployed in many networking applications because of their favorable speed and variable key-length capabilities.
- There are several variation of RC algorithms including:
  - RC2
  - RC4
  - RC5
  - RC6
Ron’s Code or Rivest Codes Scorecard

<table>
<thead>
<tr>
<th>Description</th>
<th>RC2</th>
<th>RC4</th>
<th>RC5</th>
<th>RC6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of Algorithm</td>
<td>Block Cipher</td>
<td>Stream Cipher</td>
<td>Block Cipher</td>
<td>Block Cipher</td>
</tr>
<tr>
<td>Key size (bits)</td>
<td>8/94</td>
<td>6 to 2040</td>
<td>4 to 1024</td>
<td>128, 192, or 256</td>
</tr>
<tr>
<td>Use</td>
<td>Fast block cipher that is designed to be a drop-in replacement for DES.</td>
<td>Fast block cipher that uses a variable key size.</td>
<td>Fast block cipher that is designed to be a drop-in replacement for DES.</td>
<td>Fast block cipher that uses a variable key size.</td>
</tr>
<tr>
<td>Key size (in bits)</td>
<td>40 and 64</td>
<td>4 to 2040</td>
<td>4 to 1024</td>
<td>128, 192, or 256</td>
</tr>
</tbody>
</table>

Asymmetric Algorithms

- DH is an asymmetric cryptographic protocol that allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel.
  - This key can then be used to encrypt subsequent communications using a symmetric key cipher.
- Published by Whitfield Diffie and Martin Hellman in 1976.

Diffie-Hellman (DH)

- DH is an asymmetric cryptographic protocol that allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel.
  - This key can then be used to encrypt subsequent communications using a symmetric key cipher.
- Published by Whitfield Diffie and Martin Hellman in 1976.
DH

- DH is commonly used when data is exchanged using an IPsec VPN, data is encrypted on the Internet using either SSL or TLS, or when SSH data is exchanged.
- It is not an encryption mechanism and is not typically used to encrypt data because it is extremely slow for any sort of bulk encryption.
- This is why it is common to encrypt the bulk of the traffic using a symmetric algorithm and use the DH algorithm to create keys that will be used by the encryption algorithm.

DH Scorecard

<table>
<thead>
<tr>
<th>Description</th>
<th>Diffie-Hellman Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeline</td>
<td>1976</td>
</tr>
<tr>
<td>Type of Algorithm</td>
<td>Asymmetric</td>
</tr>
<tr>
<td>Key size (in bits)</td>
<td>512, 1024, 2048</td>
</tr>
<tr>
<td>Speed</td>
<td>Slow</td>
</tr>
<tr>
<td>Time to crack</td>
<td>Unknown but considered very safe</td>
</tr>
<tr>
<td>Resource Consumption</td>
<td>Medium</td>
</tr>
</tbody>
</table>

Diffie-Hellman Algorithm

Performs authenticated key exchange

Private value
Public value

\[ Y_A = g^{x_A} \mod p \]
\[ Y_B = g^{x_B} \mod p \]

\[ (Y_A^{x_B}) \mod p = k \]
\[ (Y_B^{x_A}) \mod p = k \]
**Alice and Bob DH Key Exchange**

**Alice**

<table>
<thead>
<tr>
<th>Shared</th>
<th>Secret</th>
<th>Calc</th>
</tr>
</thead>
<tbody>
<tr>
<td>5, 23</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

5^6 mod 23 = 8

**Bob**

<table>
<thead>
<tr>
<th>Shared</th>
<th>Secret</th>
<th>Calc</th>
</tr>
</thead>
<tbody>
<tr>
<td>5, 23</td>
<td>15</td>
<td></td>
</tr>
</tbody>
</table>

5^{15} mod 23 = 19

- Bob and Alice agree to use a base number g=5 and prime number p=23.
- Alice chooses a secret integer a=6.
- Alice sends Bob (g^a mod p) or 5^6 mod 23 = 8.
- Meanwhile Bob chooses a secret integer b=15.
- Bob sends Alice (g^b mod p) or 5^{15} mod 23 = 19.
- Alice computes (g^b mod p) or 5^{15} mod 23 = 19.
- Bob computes (g^a mod p) or 5^{6} mod 23 = 8.

**Modulo**

- In computing, the modulo operation finds the remainder of division of one number by another.
- Given two numbers, X and Y, a modulo N (abbreviated as a mod N) is the remainder, on division of a by N.
- For instance:
  - “8 mod 3” would evaluate to 2.
  - “9 mod 3” would evaluate to 0.
Alice and Bob DH Key Exchange

- The result (\(7\)) is the same for both Alice and Bob.
- They will now use this as the secret key for encryption.

Alice and Bob DH Key Exchange

- BTW:
  - The initial secret integer used by Alice (6) and Bob (15) are very, very large numbers (1024 bits).
  - 8 bits = 10101010
  - 1024 bits =

RSA Cryptosystem

- Martin Hellman and Whitfield Diffie published their landmark public key paper in 1976
- Asymmetric RSA cryptosystem (Ronald Rivest, Adi Shamir and Leonard Adleman, 1977)
- Up to now, RSA is the most widely used asymmetric cryptosystem
- RSA is mainly used for two applications
  - Transport of (i.e., symmetric) keys
  - Digital signatures
RSA Cryptosystem

- RSA operations are done over the integer ring $\mathbb{Z}_n$ (i.e., arithmetic modulo $n$), where $n = p \cdot q$, with $p, q$ being large primes
- Encryption and decryption are simply exponentiations in the ring

**Encryption and Decryption**
- Given the public key $k_{pub} = (n, e)$ and the private key $k_{priv} = d$ we write $(x, y \in \mathbb{Z}_n)$
  
  $y = e_{k_{pub}}(x) \equiv x^e \mod n$

  
  $x = d_{k_{priv}}(y) \equiv y^d \mod n$

- We call $e_{k_{pub}}()$ the encryption and $d_{k_{priv}}()$ the decryption operation.
- In practice $x, y, n$ and $d$ are very long integer numbers ($\geq 1024$ bits).
- The security of the scheme relies on the fact that it is hard to derive the ‘private exponent’ $d$ given the public-key $(n, e)$.

**Key Generation**
- Like all asymmetric schemes, RSA has a set-up phase during which the private and public keys are computed
- Algorithm: RSA Key Generation

1. Choose two large primes $p, q$
2. Compute $n = p \cdot q$
3. Compute $\Phi(n) = (p-1) \cdot (q-1)$
4. Select the public exponent $e \in \{1, 2, \ldots, \Phi(n)-1\}$ such that $\gcd(e, \Phi(n)) = 1$
5. Compute the private key $d$ such that $d \cdot e \equiv 1 \mod \Phi(n)$
6. Result: public key $k_{pub} = (n, e)$ and private key $k_{priv} = d$

**Remarks:**
- Choosing two large, distinct primes $p, q$ (in Step 1) is non-trivial
- $\gcd(e, \Phi(n)) = 1$ ensures that $e$ has an inverse and, thus, that there is always a private key $d$

**Example**

ALICE | Bob
---|---

**Message $x = 4$**

1. Choose $p = 3$ and $q = 11$
2. Compute $n = p \cdot q = 33$
3. $\Phi(n) = (3-1) \cdot (11-1) = 20$
4. Choose $e = 3$
5. $d \equiv e^{-1} \equiv 7 \mod 20$

$k_{pub} = (n, e) = (33, 3)$


\[ y = x^e \equiv 4^3 \equiv 31 \mod 33 \]

\[ y^e = 31^3 \equiv 4 \equiv x \mod 33 \]
Public Key Cryptography

Public-key Algorithms

- Public-key algorithms are asymmetric algorithms based on the use of two different keys instead of one.
  - Private key: This key must be known only by its owner.
  - Public key: This key is known to everyone (it is public).
- The key that is used for encryption is different from the key that is used for decryption.
  - However, the decryption key cannot, in any reasonable amount of time, be calculated from the encryption key and vice versa.
- Public-key systems have a clear advantage over symmetric algorithms:
  - There is no need to agree on a common key for both the sender and the receiver.

Fundamental Concept

- Either key can be used for encryption but the complementary matched key is required for decryption.
  - If a public key encrypts data, the matching private key decrypts data.
  - If a private key encrypts data, the matching public key decrypts data.
Sender encrypts the message using the receiver’s public key.
- Remember that this key is known to everyone.

The encrypted message is sent to the receiving end, who will decrypt the message with his private key.
- Only the receiver can decrypt the message because no one else has the private key.

Process

This process enables asymmetric algorithms to achieve:
- Confidentiality
- Integrity
- Authentication

Authentication = Private Key (Encrypt) + Public Key (Decrypt)
Confidentiality = Public Key (Decrypt) + Private Key (Encrypt)

Authentication

Authentication is achieved when the encryption process is started with the private key.
- The corresponding public key must be used to decrypt the data.
- Since only one host has the private key, only that host could have encrypted the message, providing authentication of the sender.
Asymmetric Algorithms for Authentication

Private Key (Encrypt) + Public Key (Decrypt) = Authentication

1. Alice encrypts a message with her private key.
2. Alice transmits the encrypted message to Bob.
3. To verify that the message actually came from Alice, Bob requests and acquires Alice's public key.
4. Bob uses the public key to successfully decrypt the message and authenticate that the message did, indeed, come from Alice.

Confidentiality

- Confidentiality is achieved when the encryption process is started with the public key.
- When the public key is used to encrypt the data, the private key must be used to decrypt the data.
  - Only one host has the private key guaranteeing confidentiality.

Asymmetric Algorithms for Confidentiality

Public Key (Encrypt) + Private Key (Decrypt) = Confidentiality

1. Alice asks Bob for his public key and Bob sends it to her.
2. Alice uses Bob’s public key to encrypt a message using an agreed-upon algorithm.
3. Alice sends the encrypted message to Bob.
4. Bob uses his private key to decrypt and reveal the message.
To provide confidentiality, authentication, and integrity, the combination of two phases is necessary.
- Phase 1 - Confidentiality
- Phase 2 - Authentication

1. Alice encrypts a message using Bob’s public key.
2. Alice encrypts a hash of the message using her private key.
3. Bob uses Alice’s public key to decrypt and reveal the hash.
4. Bob uses his private key to decrypt and reveal the message.

Well-known asymmetric key algorithms:
- Diffie-Hellman
- Digital Signature Standard (DSS), which incorporates the Digital Signature Algorithm
- RSA encryption algorithms
- ElGamal
- Elliptical curve techniques
### Asymmetric Encryption Algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Key length (in bits)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diffie-Hellman (DH)</td>
<td>512, 1024, 2048</td>
<td>Public key algorithm invented in 1976 by Whitfield Diffie and Martin Hellman that allows two parties to agree on a key to use for encrypting messages. Security depends on the ease of computing a discrete logarithm.</td>
</tr>
<tr>
<td>Digital Signature Standard (DSS) and Digital Signature Algorithm (DSA)</td>
<td>512, 1024</td>
<td>Created by NIST and specifies DSA as the algorithm for digital signatures. Combination of RSA encryption and Schnorr signature. Signing and verifying a message with a DSA key pair is slow. RSA algorithm is used to sign the hash of the message.</td>
</tr>
<tr>
<td>RSA encryption algorithms</td>
<td>512 to 4096</td>
<td>Developed by Ron Rivest, Adi Shamir, and Leonard Adleman at MIT in 1977. It is an algorithm for public-key cryptography based on the difficulty of factoring large integers. It is the first algorithm known to be suitable for signing as well as encryption, and one of the first great advances in public key cryptography. Widely used in electronic commerce protocols, and is believed to be secure given sufficiently long keys and the use of up-to-date implementations.</td>
</tr>
<tr>
<td>Elliptical curve techniques</td>
<td>160</td>
<td>Elliptic curve cryptography was invented by Neil Koblitz in 1987 and by Victor Miller in 1986. Can be used to adapt many cryptographic algorithms, such as Diffie-Hellman. The main advantage of elliptical curve cryptography is that the key sizes can be smaller.</td>
</tr>
</tbody>
</table>

Although the mathematics differ with each algorithm, they all share one trait in that the calculations required are complicated.

- Design is based on factoring extremely large numbers or computing discrete logarithms of extremely large numbers.
  - As a result, computation takes more time for asymmetric algorithms.
  - Can be up to 1,000 times slower than symmetric algorithms.
- Because they lack speed, they are typically used in low-volume cryptographic mechanisms.

### Key Lengths

- Typical key lengths for asymmetric algorithms range from 512 to 4096 bits.
  - Key lengths >= 1024 bits Considered to be trustworthy
  - Key lengths < 1024 bits Considered unreliable
- Do not compare asymmetric and symmetric algorithms because they underlying designs differ greatly.
  - For example:
    - 2048-bit encryption key of RSA is roughly equivalent to a 128-bit key of RC4 in terms of resistance against brute-force attacks.
Digital Signatures

Digital Signatures Security Services

- Authenticity of digitally signed data:
  - Digital signatures authenticate a source, proving that a certain party has seen and signed the data in question.
- Integrity of digitally signed data:
  - Digital signatures guarantee that the data has not changed from the time it was signed.
- Nonrepudiation of the transaction:
  - The recipient can take the data to a third party, and the third party accepts the digital signature as a proof that this data exchange did take place.
  - The signing party cannot repudiate that it has signed the data.

Digital Signatures

- Digital signatures are often used in the following situations:
  - To provide a unique proof of data source, which can only be generated by a single party, such as contract signing in e-commerce environments.
  - To authenticate a user by using the private key of that user and the signature it generates.
  - To prove the authenticity and integrity of PKI certificates.
  - To provide nonrepudiation using a secure timestamp and a trusted time source.
  - Each party has a unique, secret signature key, which is not shared with any other party, making nonrepudiation possible.
1. Bob creates a hash of the document.
2. Bob encrypts the hash with the private key.
3. The encrypted hash, known as the signature, is appended to the document.
4. Alice accepts the document with the digital signature and obtains Bob’s public key.
5. Alice decrypts the signature using Bob’s public key to unveil the assumed hash value.
6. Alice calculates the hash of the received document, without its signature, and compares this hash to the decrypted signature hash and if the hashes match = document is authentic.

Code Signing

- Digital signatures are commonly used for code signing:
  - Provide assurance of the authenticity and integrity of software codes.
  - The executable files, or possibly the entire installation package of a program, are wrapped with a digitally signed envelope, which allows the end user to verify the signature before installing the software.

Digital Signing

- Well-known asymmetric algorithms, such as RSA or Digital Signature Algorithm (DSA), are typically used to perform digital signing.
- In 1994, the U.S. NIST selected the DSA as the Digital Signature Standard (DSS).
  - DSA is based on the discrete logarithm problem and can only provide digital signatures.
- A network administrator must decide whether RSA or DSA is more appropriate for a given situation.
  - DSA signature generation is faster than DSA signature verification.
  - RSA signature verification is much faster than signature generation.
### DSA Scorecard

<table>
<thead>
<tr>
<th>Description</th>
<th>Digital Signature Algorithm (DSA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeline</td>
<td>1994</td>
</tr>
<tr>
<td>Type of Algorithm</td>
<td>Provides digital signatures</td>
</tr>
<tr>
<td>Advantages</td>
<td>Signature generation is fast</td>
</tr>
<tr>
<td>Disadvantages</td>
<td>Signature verification is slow</td>
</tr>
</tbody>
</table>

### RSA Scorecard

<table>
<thead>
<tr>
<th>Description</th>
<th>Ron Rivest, Adi Shamir, and Len Adleman</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeline</td>
<td>1977</td>
</tr>
<tr>
<td>Type of Algorithm</td>
<td>Asymmetric algorithm</td>
</tr>
<tr>
<td>Key size (in bits)</td>
<td>512 - 2048</td>
</tr>
<tr>
<td>Advantages</td>
<td>Signature verification is fast</td>
</tr>
<tr>
<td>Disadvantages</td>
<td>Signature generation is slow</td>
</tr>
</tbody>
</table>