9.6.2 Asymmetric Encryption Facts

Asymmetric encryption, also known as public key encryption, uses two keys that are mathematically related. Both keys together are called the key pair. Consider the following asymmetric encryption facts:

Consideration	Description
Functionality	Asymmetric encryption functions in the following manner: • The public key is made available to anyone; the private key is kept secret. • One key encrypts and the other key decrypts. For example, if data is encrypted with the public key, the private key is used to decrypt the data. • The strength of an asymmetric encryption system lies in the secrecy and security of its private keys. If the private key is ever discovered, a new key pair must be generated. • Both private and public keys are created on a local machine by a local security authority (the security kernel) and a cryptographic service provider (CSP). • Asymmetric key ciphers are two associated algorithms that are inverses of each other. Both of the algorithms are easy to compute. • It is computationally infeasible to derive the second algorithm from the first without the private key.
Uses	Asymmetric key encryption can provide confidentiality (encryption), strong authentication, and non-repudiation. Asymmetric encryption is used for: • Data encryption to secure data. • Digital signing to confirm the integrity of the message and the authenticity of the sender. • Key exchange to ensure keys are secure during transit. Asymmetric encryption is often used to securely exchange symmetric keys.
Hybrid Cryptography	Operating systems, applications, and other components of information systems typically use a hybrid cryptography system. A hybrid cryptography system combines the strengths of both the symmetric and asymmetric cryptography systems (meaning that symmetric systems can process large amounts of data relatively fast, and asymmetric systems can securely distribute keys). A hybrid cryptography system works as follows: 1. A plaintext message is encrypted into ciphertext with a symmetric session key. 2. The symmetric session key is then encrypted with asymmetric cryptography using the public key of the recipient. 3. The encrypted symmetric session key and the ciphertext are sent to the recipient. 4. The recipient decrypts the symmetric session key with asymmetric cryptography (using the recipient's private key). 5. The ciphertext is then decrypted into plaintext with the decrypted session key.
Implementations	Asymmetric encryption is used with the following protocols: • SSL/TLS • IPsec • VPNs (PPTP, L2TP, SSTP) • S/MIME and PGP for email security • SSH tunnels
Management Considerations	Management considerations in implementing asymmetric key cryptography include the following: • Keys can be easily distributed with no prior relationship required. Only the public key needs to be distributed, and it does not matter who has access to the public key. The private key is always kept secure and private. • Asymmetric cryptography is scalable for use in very large, expanding environments where data is frequently exchanged between different communication partners. The number of keys required is two per user. • Key space typically starts around 1,000 bits and goes as high as 32,000 bits. • Processing speeds are much slower (about 1000 times slower) than symmetric key cryptography. • Two mechanisms are used that determine how long a cryptographic key remains in use: ◦ Ephemeral keys are generated every time the key establishment process is executed and only exist for the lifetime of a specific communication session. As such, these keys have a relatively short lifespan. ◦ Static keys can be reused by multiple communication sessions. As such, these keys remain in use for a relatively long period of time.
Perfect Forward Secrecy	Perfect forward secrecy can be implemented in public-key cryptography systems so that: • Random public keys are generated for each session. • No deterministic algorithm is used when generating the public keys. As a result, when perfect forward secrecy is used, there is no single value that can be used to compromise multiple messages. For example, the compromise of one encrypted message cannot result in other messages being compromised.

All asymmetric cryptographic systems are based on a trapdoor function creating a value that is easy to produce in one direction but difficult to reverse. The following table describes common asymmetric key cryptography systems.

System	Characteristics
Diffie-Hellman Key Exchange	The Diffie-Hellman Key Exchange was the first asymmetric algorithm. It was developed by Whitfield Diffie and Martin Hellman in 1976. It is a key agreement protocol that generates symmetric keys simultaneously at sender and recipient sites over non-secure channels. The Diffie-Hellman key exchange: - Provides for key distribution and does not provide any cryptographic services - Is based on calculating discreet logarithms in a finite field - Is used in many algorithms and standards such as DES - Is subject to man-in-the-middle attacks and requires strong authentication to validate the end points The Diffie-Hellman model uses a formula with the following values to create the public and private keys: - Y = large number < P (a typical large number is 301 digits). - P = large prime number (a prime number is a whole number, greater than or equal to one, with only two natural divisors, one and the number itself). - mod = modulus (the remainder resulting from dividing two numbers). Diffie-Hellman does not use authentication; however, it is used as a base for many other authenticated protocols. It is also used to provide perfect forward secrecy by TLS when operating in ephemeral mode (referred to as EDH or DHE).
ElGamal	ElGamal is based on a discrete logarithm problem. It was described by Taher Elgamal in 1984. ElGamal: - Extends Diffie-Hellman for use in encryption and digital signatures - Is used in the free GNU Privacy Guard software, recent versions of PGP, and other cryptosystems - Is considered to be very slow when used to create digital signatures The Digital Signature Algorithm is a variant of the ElGamal signature scheme.
Elliptic Curve Cryptography (ECC)	Elliptic curve cryptography is based on groups of numbers in an elliptical curve. The use of elliptic curves in cryptography was suggested independently by Neal Koblitz and Victor S. Miller in 1985. Elliptic curve: - Is a more efficient algorithm than other algorithms - Is used in conjunction with other methods to reduce the key size - Is suitable for small amounts of data for small devices, such as smart phones and PDAs - Produces a key of 160 bits that is equivalent to 1024-bit RSA key, which means less computational power and memory requirements. Elliptic curve Diffie-Hellman (ECDH) is an implementation of the Diffie-Hellman key exchange protocol using elliptic curve cryptography. It allows two parties, each having their own elliptic curve public/private key pair, to generate symmetric keys simultaneously over a non-secure channel.
Merkle-Hellman Knapsack	The Merkle-Hellman (MH) Knapsack is based on the subset sum problem: given a list of numbers and a sum, determine the subset used to create the sum. It was one of the earliest public key cryptosystems and was invented by Ralph Merkle and Martin Hellman in 1978. Knapsack: - Was broken by Adi Shamir. - Was regenerated in 1988; however, that version was also broken.
Rivest, Shamir, Adleman (RSA)	RSA is based on factoring large numbers into their prime values. It was developed by Rivest, Shamir, and Adleman in 1977. RSA: - Is one of the most popular and secure asymmetric cryptosystems. - Is based on the difficulty of factoring N, a product of two large prime numbers (201 digits). - Has key-length ranges from about 512 bits to 8,000 bits (2401 digits).