* **Exam Essentials:**
  + Understand the goals of cryptography.
  + Explain the differences between symmetric and asymmetric encryption.
  + Explain how digital signatures provide nonrepudiation.
  + Understand the purpose and use of digital certificates.
  + Demonstrate familiarity with emerging issues in cryptography.
* **An Overview of Cryptography:**
  + **Cryptography (cryptology)** is the practice of encoding information in a manner that it cannot be decoded without access to the required decryption key.
    - Consists of two main operations:
      * **Encryption**, which transforms plain-text information into ciphertext using an encryption key.
      * **Decryption**, which transforms the ciphertext back into plain text using the decryption key.
    - Cryptography has 4 main goals, and two that align with the CIA triad:
      * **Confidentiality** – Confidentiality (CIA)
      * **Integrity** – Integrity (CIA)
      * **Authentication:**
        + Refers to uses of encryption to validate the identity of individuals.
      * **Nonrepudiation**:
        + Ensures that individuals can prove to a third party that a message came from its purported sender.
* **Historical Cryptography:**
  + Before computers, cryptography didn’t depend on math but on scrambling text.
  + A **Cipher** is a method used to scramble or obfuscate characters to hide their value.
    - **Ciphering** is the process of using a cipher to do that type of scrambling to a message.
  + The two primary types of nonmathematical cryptography (ciphering methods):
    - **Substitution Cipher**
      * A type of ciphering system that changes one character or symbol into another.
        + **Caesar Cipher** is one where you just shift every letter up by 3 letters in the alphabet (A -> D)
        + **ROT13** is the same but shifts up 13 letters.
        + These methods are very simple to solve, however.
    - **Polyalphabetic Substitution:**
      * The problem with normal substitution ciphers is that they do not change the underlying letter and work frequency of the text.
      * **Polyalphabetic Substitutions** instead have multiple substitution alphabets for the same message.
        + For example, you might have:

First letter shift up by 3.

Second letter shift up down by 1.

Third letter shift up by 20.

And so on.

* + - * + **Vigenère Cipher** is a popular historical example that runs a message along a chosen word. The word will then determine how each letter in the message gets encrypted through a Vigenère cipher table.

S E C R E T M E S S A G E (message)

M E O W M E O W M E O W M (word is meow)

* + - **Transposition Cipher:**
      * Involves transposing or scrambling the letters in a certain manner.
        + Typically, a message is broken into blocks of equal size, and each block is then scrambled.

An example can be taking a message and shifting the position of each word(including spaces) 3 to the right.

(Moon Beams -> on Mo amesBe)

* + - * **Columnar transposition** is a classic example of a transposition cipher.
        + You choose the number of rows in advance, which will be your encryption key.
        + Then you write your message by placing successive characters in the next row until you get to the bottom of a column.

A message like “MeetMeInTheStore”

M M T T

E E H O

E I E R

T N S E

The cipher text is then the matrix read by row:

MMTTEEHOEIERTNSE

To decrypt the message, you need to know how many rows there are initially, and form the matrix, then read it by the columns.

* **The Enigma Machine:**
  + A machine created by the Germans in WWII to provide secure communications between military and political units.
  + Used different codes every day.
  + Implemented polyalphabetic substitution, changing the substitution for each character in the message.
* **Steganography:**
  + The art of using cryptographic techniques to embed secret messages within another file.
  + Steganographic algorithms work by making alterations to the least significant bits of the many bits that make up image files.
    - The changes are so minor that there is no appreciable effect on the viewed image.
    - This allows communicating parties to hide messages in plain sight.
      * Such as hiding a message within an illustration on an innocent webpage.
  + Messages are typically embedded in images, video files, or audio files, because these files are often so large that the secret message would easily be missed by even the most observant inspector.
    - Used in illegal activities like espionage and child pornography.
    - Also used in legit activities like digital watermarks to protect intellectual property.
  + Its very simply technology to use and tools like OpenStego can take in a .txt file to hide inside a chosen image.
* **Goals of Cryptography:**
  + **Confidentiality:**
    - Ensures that data remains private in three different situations:
      * When it is in rest, transit, or use.
    - This is the most widely cited goal of cryptosystems.
    - Two main types of cryptosystems enforce confidentiality:
      * **Symmetric Cryptosystems** use a shared secret key available to all users of the cryptosystem.
      * **Asymmetric Cryptosystems** use individual combinations of public and private keys for each user of the system.
    - **Obfuscation** is a concept closely related to confidentiality, in which it is the practice of making it intentionally difficult for humans to understand how the code works.
      * Used to hide the inner workings of software and other sensitive intellectual property.
  + **Integrity:**
    - Ensures that data is not altered with authorization.
      * Mechanisms or checks can ensure that stored data was not altered between the time it was created and the time it was accessed.
    - This can be enforced through **digital signatures** which are created upon the transmission of a message.
    - Integrity can be enforced by both public and secret key cryptosystems.
  + **Authentication:**
    - Verifies the claimed identity of system users and is a major function of cryptosystems.
      * This can be done with **challenge-response authentication** where a user may want access to a system, so the system asks the user to encrypt some word like “Apple”.
        + If the user is authorized, the encrypted word should match whatever the system is trying to recognize.
  + **Nonrepudiation:**
    - Provides assurance to the recipient that the message was originated by the sender and not someone pretending to be the sender.
    - Also prevents the sender from claiming that they never sent the message in the first place (also known as **repudiating** the message)
    - Secret keys or symmetric keys, cryptosystems such as simple substitution ciphers, do not provide this guarantee of nonrepudiation.
      * Nonrepudiation is offered only by public key, or asymmetric cryptosystems.
* **Cryptographic Concepts:**
  + **Plain-text** messages are messages before they are put into code form.
    - Represented by a **P**
  + **Cipher-text** messages are plain-text messages that have been **encrypted** using a cryptographic algorithm.
    - Represented by a **C**
  + **Cryptographic Keys (Cryptovariables):**
    - All cryptographic algorithms rely on **keys** to maintain their security.
      * **Keys** are usually just a large binary number.
    - Every algorithm has a specific **key space** which is the range of values that are valid for use as a key for a specific algorithm.
      * Key spaces are defined by its **key length** which is the number of binary bits in the key.
        + The key space is the range between the key that has all 0s and the key that has all 1s.

A key with 8 bits will have a key space of 0 to 255.

* + Keep keys private and unfound cause if they are discovered, the encryption doesn’t work.
  + **The Kerchoff Principle:**
    - A principle that a cryptographic system should be secure even if everything about the system, except the key, is public knowledge.
      * While it may be true a deployed system will be more secure if it keeps the workings of the system private as well, Kerchoff believes this is bad to follow, because if a system can be broken, it should be replaced.
* **Cryptography:**
  + The art of creating and implementing secret codes and ciphers.
  + **Cryptanalysis:**
    - The study of methods to defeat codes and ciphers.
  + **Cryptology:**
    - The combination of cryptography and cryptanalysis.
  + **Cryptosystems:**
    - Specific implementations of a code or cipher in hardware and software.
* **Ciphers:**
  + Algorithms used to perform encryption and decryption operations.
  + **Cipher Suites** are the sets of ciphers and key lengths supported by a system.
  + Modern ciphers fit into two major categories, describing their operation:
    - **Block Ciphers:**
      * Operate on “chunks” or blocks of a message and apply the encryption algorithm to an entire message block at the same time.
        + Transposition ciphers are an example.
      * Most modern encryption algorithms use some type of block cipher.
    - **Stream Ciphers:**
      * Operate on one character or bit of a message (or data stream) at a time.
        + The Caesar Cipher is an example.
      * Stream ciphers can also function as a type of block cipher, where there is a buffer that fills up to real-time data that is then encrypted as a block and transmitted to the recipient.
* **Modern Cryptography:**
  + **Cryptographic Secrecy:**
    - Back then, one predominant principle of cryptography was “security through obscurity” which meant it was best to hide the encryption algorithm from outsiders to keep it secure.
      * However, designing a system around this principle makes the system more reliant on secrecy of itself then how good it is at protecting data.
        + Thus, modern systems are very public in order to receive feedback from people to ensure any security problem can be found and fixed.
    - Now, most secrecy in modern cryptography surrounds the key and not the system.
      * **Columnar Transposition** is a good example of this, yet still contains flaws which make it inadequate for modern cryptography.
    - Overall, the length of a key is an extremely important factor in determining the strength of a system and reducing the likelihood that the encryption will be compromised.
      * The advancement of computing power makes it so that keys will need to get continuously larger.
        + Before, 56-bit keys used to suffice, but now 128-bit are required.
        + With quantum computers, this length will need to increase.
  + **Symmetric Key Algorithms:**
    - These rely on a “shared secret” encryption key that is distributed to all members who participate in the communication.
      * Key is used by all parties to encrypt and decrypt messages.
    - When large-sized keys are used, symmetric encryption is difficult to break.
    - It is primarily employed to perform bulk encryption and provides only for the security service of confidentiality.
    - **Symmetric Key Cryptography** can be **called Secret Key Cryptography** and **Private Key Cryptography**.
    - **Private Key:**
      * The term itself always means the private key from the key pair of public key cryptography (aka asymmetric).
      * However, both private key cryptography and shared private key refer to symmetric cryptography.
        + The meaning of the word private is stretched to refer to two people sharing a secret that they keep confidential.
      * The true meaning of private is that only a single person has a secret that’s kept confidential.
    - **Symmetric key cryptography has several weaknesses:**
      * **Key distribution is a major problem** since parties must have a secure method of exchanging the secret key before establishing communications with a symmetric key protocol.
        + If a secure electronic channel is not available, an offline key distribution method must often be used (that is, out-of-band exchange)
      * **Nonrepudiation is not implemented** because any communicating party can encrypt and decrypt messages with the shared secret key.
        + Thus, there is no way to prove where the message originated.
      * **The algorithm is not scalable.** It is extremely difficult for large groups to communicate using symmetric key cryptography.
        + Secure private communications between individuals in the group could be achieved only if each possible combination pair of users forms a private key.
      * **Keys must be regenerated often**, because if a participant leaves the group, all keys known by that participant must be discarded.
    - The strength of symmetric key cryptography comes in its high operating speeds which makes it 1000 to 10000 times faster than asymmetric algorithms.
      * Hardware implementations also make it faster as well.
  + **Asymmetric Key Algorithms (Public Key Algorithms):**
    - Provide a solution to the weaknesses of symmetric key encryption.
    - In this system, each user has two keys; a public key which is shared with all users, and a private key which is kept secret and known only to the owner of the keypair.
      * However, opposite and related keys must be used in tandem to encrypt and decrypt.
        + This means if the public key encrypts the message, then only the corresponding private key can decrypt it.

Example:

If I encrypt Skippy’s message with his public key, he can only decrypt the message with his private key. Because of this, I can’t even decrypt his message.

* + - Asymmetric key algorithms also provide support for digital signature technology.
      * I could send a message to Skippy that contains a message digest. The digest could be encrypted with my private key, then Skippy could decrypt it with my public key. Doing this ensures that it was me and only me who created and sent the message.
    - Asymmetric key cryptography has major strengths:
      * **The addition of new users requires the generation of only one public-private key pair.** This means that the algorithm is extremely scalable for any cryptosystem.
      * **Users can be removed far more easily.** Asymmetric cryptosystems provide key revocation mechanisms that allows a key to be canceled, effectively removing a user from the system.
      * **Key regeneration is required only when a user’s private key is compromised.** Unlike symmetric key systems, if a user leaves the system, there is no need to change anything but remove the user’s keys. This is because the user never had any critical information related to other users.
      * **Asymmetric key encryption can provide integrity, authentication, and nonrepudiation.** If a user does not share their private key with other individuals, a message signed by that user can be shown to be accurate and from a specific source and cannot be later repudiated.
      * **Key distribution is a simple process.** Users who want to participate in the system simply make their public key available to everyone, that is all.
      * **No preexisting communication link needs to exist.** Two individuals can begin communicating securely from the start of their communication session, but a preexisting relationship is not required to provide a secure mechanism for data exchange.
    - **The major weakness of public key cryptography is its slow speed of operation.**
      * For this reason, many applications that require the secure transmission of large amounts of data use public key cryptography to establish a connection and then exchange a symmetric secret key for the data itself.
  + **Hashing Algorithms:**
    - Message digests are summaries of a message’s content produced by a hashing algorithm.
      * It’s extremely difficult, if not impossible, to derive a message from an ideal hash function, and it’s very unlikely that two message will produce the same hash value.
        + **Collisions** are occurrences when two different messages produce the same hash value and is the reason why hashing algorithms are not popularly viewed.
* **Symmetric Cryptography:**
  + **Data Encryption Standard (DES):**
    - Published by the U.S. government in 1977 as a proposed standard cryptosystem for all government communications.
      * Contains flaws that made it considered unsecure.
    - DES is a 64-bit block cipher that has fives modes of operation:
      * Electronic Codebook (ECB) mode,
      * Cipher Block Chaining (CBC) mode,
      * Cipher Feedback (CFB) mode,
      * Output Feedback (OFB) mode,
      * Counter (CTR) mode
    - All modes operate on 64 bits of plaintext at a time to generate 64-bit blocks of ciphertext.
      * The key used by DES is 56 bits long.
        + Technically, the key is 64 bits long, but the 8 bits are sometimes but rarely used as parity information.

Thus, it best to assume 56-bit keys.

* + - * Uses a longer series of exclusive or (XOR) operations to generate the ciphertext.
        + Process is repeated 16 times for each encryption/decryption operation.
        + Each repetition is called a **Round** of encryption.
  + **Electronic Codebook Mode (ECB Mode):**
    - Most simple yet unsecure mode of DES.
    - Each time the algorithm processes a 64-bit block, it simply encrypts the block using the chosen secret key.
      * This means if the algorithm encounters the same block multiple times, it will produce the same encrypted block.
        + Attackers could thus build a “code book” of all the possible encrypted values and after enough blocks have been gathered, it would be very possible to decipher encrypted data.
    - This vulnerability means ECB mode is impractical for all but the shortest of transmissions.
      * Today, ECB is only used for exchanging small amounts of data such as keys and parameters used to initiate other DES modes as well as the cells in a database.
  + **Cipher Block Chaining Mode (CBC Mode):**
    - Each block of unencrypted text is combined with the block of ciphertext immediately preceding it before it is encrypted using the DES algorithm.
      * The decryption process simply decrypts the ciphertext and reverses the encryption operation.
    - CBC uses an **Initialization Vector (IV)**, which is a randomly selected value that is used to start the encryption process.
      * CBC takes the IV and combines it with the first block of the message using an operation known as the exclusive or (XOR), producing a unique output every time the operation is performed.
      * The IV must be sent to the recipient, perhaps by tacking the IV onto the front of the completed ciphertext in plain form or by protecting it with ECB mode encryption using the same key used for the message.
    - One important consideration when using CBC mode is that errors propagate – if one block is corrupted during transmission, it becomes impossible to decrypt that block and the next block as well.
  + **Cipher Feedback Mode (CFB Mode):**
    - The streaming cipher version of CBC.
      * This means CFB operates against data produced in real time.
      * However, instead of breaking a message into blocks, it uses memory buffers of the same block size.
      * As the buffers become full, it is encrypted and then sent to the recipients.
        + Then the system waits for the next buffer to be filled as the new data is generated before it is in turn encrypted and then transmitted.
    - Other than the change from the preexisting data to real-time data, CFB operates in the same fashion as CBC.
  + **Output Feedback Mode (OFB Mode):**
    - In OFB, DES operates in almost the same fashion as it does in CFB mode.
      * However, instead of XORing an encrypted version of the previous block of ciphertext, DES XORs the plain text with a seed value.
      * For the first encrypted block, an initialization vector is used to create the seed value.
        + Future seed values are derived by running the DES algorithm on the previous seed value.
      * The major advantages of OFB mode are that there is no chaining function and transmission errors do not propagate to affect the decryption of future blocks.
  + **Counter Mode (CTR Mode):**
    - Uses a stream cipher similar to that used in CFB and OFB modes.
      * However, instead of creating the seed value for each encryption/decryption operation from the results of the previous seed values, it uses a simple counter that increments for each operation.
    - CTR mode allows you to break an encryption or decryption into multiple independent steps.
      * This makes it well suited in parallel computing.
    - As with OFB mode, errors do not propagate in CTR mode.
  + **Triple DES (3DES):**
    - Because DES’s 56-bit key is no longer considered adequate in the face of modern cryptanalytic techniques and supercomputing, alternatives are needed.
      * 3DES is a solution that utilizes the same algorithm as DES yet is more secure.
    - There are four versions of 3DES plus a not-really-used version:
      * **DES-EEE3:** Encrypts the plaintext three times using three different keys: K1, K2, and K3.
        + Can expressed as **E(K1, E(K2, E(K3, P)))**

E is an encrypt method and P is the plaintext

* + - * + Has an effective key length of 168 bits.
      * **DES-EDE3:** Uses three keys but replaces the second encryption operation with a decryption operation.
        + **E(K1, D(K2, E(K3, P)))**
      * **DES-EEE2:** Uses only two keys, K1 and K2, but still encrypts a total of 3 times.
        + **E(K1, E(K2, E(K1, P)))**
        + Has an effective key length of 112 bits.
      * **DES-EDE2:** Uses only two keys but uses a decryption operation in the middle.
        + **E(K1, D(K2, E(K1, P)))**
        + Has an effective key length of 112 bits.
      * **DES-EDE1:** Uses only one cryptographic key and thus has the same algorithm standard as DES, making it unacceptably weak for most applications.
        + It only exists for backward-compatibility reasons.
    - It is believed that all 4 main versions of 3DES are equally secure.
  + **Advanced Encryption Standard (AES):**
    - In 2000, NIST announced that the Rijndael (rhine-doll) block cipher would replace DES.
    - 2001, NIST released FIPS 197 which mandated the use of AES/Rijndael for the encryption of all Sensitive but Unclassified data (SBU) by the U.S. government.
      * The AES cipher allows the use of three key strengths:
        + 128 bits
        + 192 bits
        + 256 bits
      * AES only allows the processing of 128-bit blocks, but Rijndael exceeded this specification allowing cryptographers to use a block size equal to the key length.
        + The number of encryption rounds depends on the key length chosen:

128-bit keys require 10 rounds.

192-bit keys require 12 rounds.

256-bit keys require 14 rounds.

* + **Symmetric Key Management:**
    - Because cryptographic keys contain information essential to the security of the cryptosystem, it requires measures to protect it.
      * These security measures are collectively known as **Key Management Practices**.
        + This includes safeguards surrounding the creation, distribution, storage, destruction, recovery, and escrow of secret keys.
    - **Creation and Distribution of Symmetric Keys:**
      * **Key exchange** is one of the major problems underlying symmetric encryption algorithms, as it is the secure distribution of the secret keys required to operate the algorithm.
      * Three main methods used to exchange secret keys securely:
        + **Offline Distribution:**

The most technically simple method where one party provides the other party with a sheet of paper or piece of storage media containing the secret key.

Literally just a physical means to convey the key, which has its set of problems like if the media containing the key is intercepted or if the media is thrown away then found.

* + - * + **Public Key Encryption:**

Faster method, as once the link is successfully established and the parties are satisfied as to each other’s identity, they exchange a secret key over the secure public key link.

They switch communications from the public key algorithm to the secret key algorithm and enjoy the increased processing speed.

In general, secret key encryption is thousands of times faster than public key encryption.

* + - * + **Diffie-Hellman Key Exchange Algorithm:**

In some cases, neither public key nor offline distribution is sufficient.

This method works to solve this in steps:

1. The communicating parties Jon and Doe agree on two large numbers:

P (prime number)

G (integer)

1 < G < P

2. Jon chooses a random large integer r and performs the following:

R = Gr mod P

3. Doe chooses a random large integer s and performs the following:

S = Gs mod P

4. Jon sends R to Doe and Doe sends S to Jon.

5. Jon performs the following:

K = Sr mod P

6. Doe performs the following:

K = Rs mod P

At this point, Jon and Doe have the same value K, and can use this for secret key communication between the two parties.

* + - **Storage and Destruction of Symmetric Keys:**
      * A major challenge with symmetric key cryptography is that all keys in the system must be secured.
      * Best practices with handling keys include:
        + Never storing keys on the same system where encrypted data resides.
        + For sensitive keys, consider providing two individuals with half of the key.

**Split Knowledge** is the principle that there must be a collaboration to re-create the entire key.

* + - * However, the difficulty comes when a user leaves an organization.
        + You will have to re-encrypt all material with a new key.
    - **Key Escrow and Recovery:**
      * There are efforts by governments to implement **Key Escrow Systems** that pretty much breaks the encryption of any system just in case they have content they need to break.
      * There are two major approaches to key escrow that have been proposed:
        + **Fair Cryptosystems:** The secret keys used in a communication are divided into two or more pieces, each of which is given to an independent third party.

Each of these pieces is useless on its own but may be recombined to obtain the secret key.

When the government obtains legal authority obtain a particular key, they present the court order to each third party.

* + - * + **Escrowed Encryption Standard:** Provides the government with a technological means to decrypt ciphertext.

This is the basis behind the Skipjack algorithm.

* **Asymmetric Cryptography:**
  + This uses a public key and a private key. It has its pros and cons. Can provide safe communications between two parties that are initially unknown to each other.
    - However, it is a lot more complex and requires keys that are longer in length in order to be as powerful as symmetric cryptography.
* **RSA Public Key Algorithm:**
  + A famous public key algorithm that is widely used for secure communications.
  + The algorithm depends on the computational difficulty inherent in factoring large prime numbers.
    - Each user of the cryptosystem generates a pair of public and private keys using the algorithm.
    - While the exam does test on specifics, you must know that it is based on the complexity of factoring large prime numbers.
  + **Importance of Key Length:**
    - This is perhaps the most important security parameter that can be set at the discretion of the security administrator.
    - It’s important to understand the capabilities of your encryption algorithm and choose a key length that provides an appropriate level of protection.
      * This judgement can be made by weighing the difficulty of defeating a given key length (measured in required processing time)
    - It is important to take into account the rapid growth of computing power, as Moore’s law suggests that computing power doubles approximately every two years.
      * There are also cloud computing serves which can give Hackers access to supercomputers for a rented amount of time.
    - The strength of the length of a key also depends on the cryptosystem.
      * 1024-bit RSA is as secure as 160-bit ECC.
    - While it might be obvious to just use very large keys, the longer the key, the more overhead is required.
* **Elliptic Curve Cryptography (ECC)**:
  + Any elliptic curve can be defined by y^2 = x^3 + ax + b.
    - All numbers are real numbers.
    - Each elliptic curve has a corresponding **elliptic curve group** made up of the points on the elliptic curve along with the point O, located at infinity.
    - Two points within the same elliptic curve group (P and Q) can be added together with an elliptic curve addition algorithm.
      * P + Q
    - This problem can be extended to involve multiplication by assuming that Q is a multiple of P, meaning the following.
      * Q = xP
  + Elliptic curve discrete logarithm problem is the basis of elliptic curve cryptography, as finding the x value is super hard even if P and Q are known.
    - It is believed to be harder to solve than RSA and standard discrete logarithm problem.
* **Hash Functions:**
  + Take a potentially long message and generate a unique output value derived from the content of the message.
    - This value is also known as a **Message Digest**.
      * This can be generated by the sender of a message and transmitted to the recipient along with the full message for two reasons.
        + First, the recipient can use the hash function to recompute the message digest from the full message.

They can then compare the computed message digest to the transmitted one to ensure that the message sent by the originator is the same one received by the recipient.

If the message does not match, it was modified during transit.

Overall, just a slight change in a message will make the digest value very different, even if there is an added space.

* + - * + Second, the message digest can be used to implement a digital signature algorithm.
      * The term message digest is used interchangeably with a wide variety of synonyms, including **hash, hash value, hash total, CRC, fingerprint, checksum,** and **digital ID.**
    - There are five basic requirements for a cryptographic hash function:
      * They accept an input of any length.
      * They produce an output of a fixed length, regardless of the length of the input.
      * The hash value is relatively easy to compute.
      * The hash function is one-way (meaning it is extremely hard to determine the input when provided the output)
      * The hash function is **collision free**, meaning it is extremely hard to find two messages that produce the same hash value.
  + **Secure Hash Algorithm (SHA):**
    - SHA has different versions including SHA-1 to SHA-3.
      * They are government standard hash functions promoted by the NIST and are specified in the **Secure Hash Standard (SHS)**, also known as **Federal Information Processing Standard (FIPS) 180**.
    - SHA-1 takes an input of virtually any length (in reality, there is an upper bound of about 2 million terabytes) and produces a 160-bit message digest.
      * Processes messages in 512-bit blocks.
        + If message is not a multiple of 512 bits, additional data will be added to pad the length.
      * Cryptanalytic attacks demonstrated that there are weaknesses in the SHA-1 algorithm.
    - SHA-2 was thus created which has four variants:
      * SHA-256 produces a 256-bit message digest using a 512-bit block size.
      * SHA-224 uses a truncated version of the SHA-256 hash to produce a 224-bit message digest using a 512-bit block size.
      * SHA-512 produces a 512-bit message digest using 1024-bit block size.
      * SHA-384 uses a truncated version of the SHA-512 hash to produce a 384-bit digest using a 1024-bit block size.
        + Even though SHA-2 is really secure, it still suffers from the same problems as the SHA-1 algorithm.
    - The **Keccak Algorithm** was released as the **SHA-3** standard hashing function which supports the same features and variants as SHA-2 but is a lot more secure.
  + **MD5:**
    - A message digest algorithm that processes 512-bit blocks of messages but uses four distinct rounds of computation to produce the digest of the same length as earlier MD2 and MD4 algorithms (128 bits).
    - Security features are implemented that reduce the speed of message digest production significantly.
      * Cryptanalytic attacks demonstrated that the MD5 protocol is subject to collisions, preventing its use for ensuring message integrity.
* **Digital Signatures:**
  + Once you have chosen a cryptographically sound hashing algorithm, you can use it to implement a **digital signature** system.
  + Digital signature infrastructure has two goals:
    - Assure the recipient that the message truly came from the claimed sender.
      * This enforces nonrepudiation (a claim from the sender that the message is a forgery).
    - Assure the recipient that the message was not altered while in transit between the sender and recipient.
      * Protects against both malicious modification (from a third party) and unintentional modification (electrical interferences).
  + Digital signature algorithms rely on a combination of public key cryptography and hashing functions.
    - If Alice wants to digitally sign a message she’s sending to Bob, she will:
      * 1. Generate a message digest of the original plaintext message using a good hashing algorithm like SHA3-512.
      * 2. Encrypt only the message digest using her private key. This encrypted message digest is the digital signature.
      * 3. Append the signed message digest to the plaintext message.
        + When Bob receives the signed message, he reverses the procedure:

1. Decrypts the digital signature with Alice’s public key.

2. Uses the same hashing function to create a message digest of the full plaintext message received from Alice.

3. Compare the decrypted message digest he received from Alice with the message digest he computed himself to ensure everything is good.

* + These Digital Signatures can also be used by Software Vendors to authenticate code distributions.
    - Notice however that digital signing does not provide any privacy.
      * It only ensures that the cryptographic goals of integrity, authentication, and nonrepudiation are met.
    - For the example above, Alice could encrypt her entire message + digest with Bob’s public key.
      * Thus, only Bob is able to decrypt the entire message with his private key and reverse the digest.
        + To summarize:

If you want to encrypt a message, use the recipient’s public key.

If you want to decrypt a message sent to you, use your private key.

If you want to digitally sign a message you are sending, use your private key.

If you want to decrypt a signed message sent to you, use the sender’s public key.

* + **Hashed Message Authentication Code (HMAC):**
    - An algorithm that implements a partial digital signature—it guarantees the integrity of a message during transmission but does not provide for nonrepudiation.
    - HMAC can be combined with any standard message digest generation algorithm, such as SHA-3, by using a shared secret key.
      * Therefore, only communicating parties who know the key can generate or verify the digital signature.
    - While not being able to provide any nonrepudiation functionality, it operates in a more efficient manner than other digital signature standard processes and may be more suitable for applications in which symmetric key cryptography is appropriate.
      * Thus, HMAC represents a halfway point between unencrypted use of a message digest algorithm and computationally expensive digital signature algorithms based on public key cryptography.
  + **Digital Signature Standard (DSS):**
    - NIST specifies the digital signature algorithms acceptable for federal government use in Federal Information Processing Standard (FIPS) 186-4, also known as **Digital Signature Standard (DSS).** This document specifies that all federally approved digital signature algorithms must use the SHA-3 hashing functions.
      * DSS also specifies the encryption algorithms that can be used to support a digital signature infrastructure. Three are currently approved:
        + The **Digital Signature Algorithm (DSA)**
        + The **Rivest, Shamir, Adleman (RSA)**
        + The **Elliptic Curve DSA (ECDSA)**
* **Public Key Infrastructure (PKI):**
  + The major strength of public key encryption is its ability to facilitate communications between parties previously unknown to each other.
    - This is made possible by **public key infrastructure (PKI)** hierarchy of trust relationships.
      * These trusts permit combining asymmetric cryptography with symmetric cryptography along with hashing and digital certificates, giving us hybrid cryptography.
  + **Certificates:**
    - **Digital Certificates** provide communicating parties with the assurance that the people they are communicating with truly are who they claim to be.
      * These are essentially endorsed copies of an individual’s public key.
    - When users verify that a certificate was signed by a trusted **certificate authority (CA),** they know that the public key is legitimate.
    - Digital certificates contain specific identifying information, and their construction is governed by an international standard—X.509.
      * Certificates conforming to X.509 contain:
        + Version of X.509.
        + Serial number (from the certificate creator).
        + Signature algorithm identifier (specifies the technique used by the certificate authority to digitally sign the contents of the certificate).
        + Issuer name (identification of the certificate authority that issued the certificate).
        + Validity period (specifies the date and time the certificate began and the data and time it will expire).
        + Subject’s Common Name (CN) that clearly describes the certificate owner (e.g., “certmike.com”).
        + While optional, Subject Alternative Names (SAN) can allow you to specify additional items (IP addresses, domain names, and so on) to be protected by the certificate.
        + Subject’s public key (the meat of the certificate—the actual public key the certificate owner used to set up secure communications).
      * The current version of X.509 (version 3) supports certificate extensions—customized variables containing data inserted into the certificate by the certificate authority to support tracking of certificates and various applications.
    - Certificates may be issued to provide assurance for the public keys of:
      * Computers/machines
      * Individual users
      * Email addresses
      * Developers (code-signing certificates)
    - The subject of a certificate may include a wildcard in the certificate name, indicating that the certificate is good for subdomains as well. The wildcard is designated by an asterisk character.
      * Example for \*.certmike.com:
        + Certmike.com
        + [www.certime.com](http://www.certime.com)
        + Mail.certmike.com
        + Secure.certmike.com
      * Wildcard certificates are only good for one level of subdomain.
        + Therefore, the \*.certmike.com certificate would not be valid for the [www.cissp.certmike.com](http://www.cissp.certmike.com) subdomain.
  + **Certificate Authorities (CAs):**
    - The glue that binds the public key infrastructure together.
    - These neutral organizations offer notarization servers for digital certificates.
      * You must provide your identity to the satisfaction of the reputable CA to get a digital certificate.
      * Organizations include:
        + Amazon Web Services
        + GlobalSign
        + Certum
        + GoDaddy
        + Entrust
        + Trustwave
    - Nothing is preventing any organization from simply setting up shop as a CA.
      * However, the certificates issued by a CA are only as good as the trust placed in the CA that issued them.
        + Thus, if you don’t recognize the name of a CA that issued the certificate, you shouldn’t place any trust in the certificate at all.
      * PKI relies on hierarchy of trust relationships.
        + If you configure your browser to trust a CA, it will automatically trust all of the digital certificates issued by that CA.
        + Preconfigured search browsers typically trust major CAs to avoid placing the burden on the user.
    - **Registration Authorities (RAs)** assist CAs with verifying users’ identities prior to issuing digital certificate, but don’t issue the certificates themselves.
    - Certificate authorities must carefully protect their own private keys to preserve their trust relationships.
      * They do this with an **Offline Certificate Authority** to protect **Root Certificates**, the top-level certificate for their entire PKI.
        + This offline CA is disconnected from networks and powered down until it is needed.
        + The offline CA uses the root certificate to create subordinate **Intermediate CAs** that serve as the **Online CAs** used to issue certificates on a routine basis.
      * In the CA trust model, the use of a series of intermediate CAs is known as **Certificate Chaining**.
        + To validate a certificate, the browser verifies the identity of the intermediate CA(s) first and then traces the path of trust back to a known root CA, verifying the identity of each link in the chain of trust.
    - Certificate authorities do not need to be third-party service providers.
      * Many organizations operate internal CAs that provide **self-signed certificates** for use inside an organization.
        + Their certificates won’t be trust by the browsers of external users, but internal systems may be configured to trust the internal CA. saving the expense of obtaining certifications from a third part CA.
  + **Certificate Generation and Destruction:**
    - **Enrollment:**
      * First, you must prove your identity to a CA in some manner.
      * Then, you provide your public key in the form of a **Certificate Signing Request (CSR).**
      * Then, the CA creates an X.509 digital certification containing your identifying information and a copy of your public key.
        + The CA digitally signs the certificate using the CA’s private key and provides you with a copy of your signed digital certificate.

You may then safely distribute this certificate to anyone you want to communicate securely with.

* + - * CAs issue different types of certificates depending upon the level of identity verification that they perform.
        + **Domain Validation (DV) Certificates** are the most simple and common where the CA only verifies that the certificate subject has control of the domain name.
        + **Extended Validation (EV) Certificates** provide a higher level of assurance, and the CA takes steps to verify that the certificate owner is a legitimate business.
    - **Verification:**
      * When you receive a digital certificate from someone with whom you want to communicate, you verify the certificate by checking the CA’s digital signature using the CA’s public key.
        + Then, you must check that the certificate was not revoked using a **Certificate Revocation List (CRL)** or the **Online Certificate Status Protocol (OCSP)**.

At this point, it is safe to assume the public key listed in the certificate is authentic assuming:

The digital signature of the CA is authentic.

You trust the CA.

The certificate is not listed on a CRL.

The certificate actually contains the data you are trusting.

* + - * One last important point is that before you trust an identifying piece of information about someone, be sure that it is actually contained within the certificate.
        + If a certificate contains the email ([billjones@foo.com](mailto:billjones@foo.com)), you can not make assumptions about the actual identity of the email.

However, if the certificate contains the name (Bill Jones) along with the email and even phone number, then their identity is proven.

* + - * + Most web browsers handle this for you though.
      * **Certificate Pinning** approaches instruct browsers to attach a certificate to a subject for an extended period of time.
        + When sites use certificate pinning, the browser associates that site with their public key.

This allows users or admins to notice and intervene if a certificate unexpectedly changes.

* + - **Revocation:**
      * Certificates can get revoked because:
        + The certificate was compromised (i.e., the private key was accidentally given away)
        + The certificate was erroneously issued (i.e., the CA mistakenly issued a certificate without proper verification).
        + The details of the certificate changed (i.e., the subject’s name changed).
        + The security association changed (i.e., the subject is no longer employed by the organization sponsoring the certificate).
      * The **revocation request grace period** is the maximum response time within which a CA will perform any requested revocation.
        + **The Certificate Practice Statement (CPS)** defines this and states the practices a CA employs when issuing or managing certificates.
      * Three techniques to verify the authenticity of certificates and identify revoked certificates:
        + **Certification Revocation Lists (CRLs)**: Maintained by CAs and tell what certificates have been revoked and when.

Downside is that these lists must be periodically downloaded and cross-referenced, so you won’t always have the most updated list.

* + - * + **Online Certificate Status Protocol (OCSP)**: This eliminates the latency inherent in the use of CRLs by providing real-time certification verification.

When a client receives a certificate, it sends an OCSP request to the CA’s OCSP server, to which a valid, invalid, or unknown response will be given.

The primary issue with OCSP is that it places a significant burden on the OCSP servers operated by CAs. These servers must process requests from every single visitor to a website or other user of a digital certificate.

* + - * + **Certificate Stapling:** An extension to OCSP that relieves some of the burden placed upon CAs by the original protocol.

Instead of the user contacting the OCSP server, the website does it instead. The website will receive a digital signature and time stamp from the OCSP server which will be attached or stapled to the digital certificate sent to the user.

Thus, the user can authenticate the signature and how recently it was signed.

The website can reuse this stapled certificate with other users assuming the time stamp is recent enough too.

Typically, timestamps will be valid for 24 hours, reducing the burden on OCSP servers greatly.

* + - **Certificate Formats:**
      * Digital certificates are stored in files and those files come in different formats, both binary and text-based:
        + **Distinguished Encoding Rules (DER)** format is the most common binary format where DER certificates are normally stored in files with the **.DER**, **.CRT**, or **.CER** extension.
        + **Privacy Enhanced Mail (PEM)** certificate format is an ASCII text version of the DER format, usually with the **.PEM** or **.CRT** extension.

Thus, you cannot tell whether a CRT file is a binary or text file without looking at it.

* + - * + **Personal Information Exchange (PFX)** format is commonly used by Windows systems where PFX certificates may be stored in binary form using either **.PFX** or **.P12** file extensions.
        + **P7B Certificates** are stored in ASCII text format and used for Windows systems.
* **Asymmetric Key Management:**
  + When working within the public key infrastructure, it’s important to comply with several best practice requirements to maintain the security of your communications.
    - First, your encryption system wisely, as systems in the public domain are constantly tested by experts and are thus better.
    - Second, choose your keys in an appropriate manner. These should have the balance between security requirements and performance considerations. These keys should also be truly random if possible.
    - Next, don’t share private keys in public systems.
      * Also, retire and rotate keys after they have been used for a good amount of time.
        + Backup your keys too.
  + **Hardware Security Modules (HSMs)** provide an effective way to manage encryption keys by storing them and managing them in a way where no human needs to work directly with the keys.
    - Can be as simple as a USB to cloud services.
* **Cryptographic Attacks:**
  + **Brute Forces:**
    - Simply involves trying every possible key.
      * It is guaranteed to work but will take a very long time.
        + DES which is considered weak has 2^56 different possible keys.
  + **Frequency Analysis:**
    - Involves looking at the blocks of an encrypted message to determine if any common pattern exists.
      * Only works for historical ciphers and not modern algorithms.
  + **Known Plain Text:**
    - This attack relies on the attacker having pairs of known plain text along with the corresponding ciphertext.
      * This gives attackers a place to start attempting to derive the key.
        + With modern ciphers, this will require billions of combinations to have a chance at cracking the cipher.

This was used to crack the Enigma, however.

* + **Chosen Plain Text:**
    - The attacker obtains the ciphertexts corresponding to a set of plain texts of their own choosing.
      * This allows the attacker to attempt to derive the key used and thus decrypt other messages encrypted with that key.
        + Difficult but not impossible.
    - Advanced methods such as differential cryptanalysis are types of chosen plaintext attacks.
  + **Related Key Attack:**
    - This is like a chosen plain-text attack, except the attacker can obtain cipher texts encrypted under two different keys.
      * This is useful if you can obtain the plain text and matching ciphertext.
  + **Birthday Attack:**
    - An attack on cryptographic hashes, based on the **Birthday Theorem**.
      * Based on the idea of “How many people would you need to have in a room to have a strong likelihood that two would have the same birthday (month and day, but not year).
        + 367 to be 100% sure, but only 23 to be 51% sure.
    - Using this concept, you can try to find a collision of hashes, and to get a 51% chance that two plaintexts collide when hashing, you need a lot of combinations of keys. However, this is still way faster than brute force.
  + **Downgrade Attack:**
    - Used against secure communications such as TLS in an attempt to get the user or system to inadvertently shift to less secure cryptographic modes.
  + **Rainbow Tables, Hashing, and Salting:**
    - **Rainbow Table** attacks attempt to reverse hashed password values by precomputing the hashes of common passwords and trying to match them.
      * **Salting** however prevents this by adding randomly generated values to each password prior to hashing.
    - **Key Stretching** is used to create encryption keys from passwords in a strong manner.
      * Algorithms include **Password Based Key Derivation Function v2 (PBKDF2)**, which use thousands of iterations of salting and hashing to generate encryption keys that are resilient to attacks.
  + **Exploiting Weak Keys:**
    - Scenarios where a good cryptographic algorithm is being used like AES, but it is implemented in a weak manner like using weak key generation.
  + **Exploiting Human Error:**
    - Human error is one of the major causes of encryption vulnerabilities.
      * If an email is sent using an encryption scheme, someone else may send it **In the Clear (Unencrypted)**.
        + If both encrypted and unencrypted messages are obtained by a cryptanalyst, the process of decoding future messages will be simplified.

Code keys give insight into what the key consists of, leading to many accidents.

* + - * Another error is to use weak or deprecated algorithms with found flaws or are just redundant with modern computing power.
* **Emerging Issues in Cryptography:**
  + **Tor and the Dark Web:**
    - **The Onion Router (TOR)** provides a mechanism for anonymously routing traffic across the internet using encryption and a set of relay nodes.
      * Relies on **Perfect Forward Secrecy** where layers of encryption prevent nodes in the relay chain from reading anything other than the specific information they need to accept and forward the traffic.
        + By using this with 3 or more relay nodes, TOR allows for anonymous browsing on the standard internet and the hosting of anonymous sites on the Dark Web.
  + **Blockchain:**
    - A distributed and immutable public ledger. It can store records in a way that distributes those records among many different systems located around the world and do so in a manner that prevents anyone from tampering with those records.
      * Creates a data store that nobody can tamper with or destroy.
    - A major application is **Cryptocurrency**, whether blockchains like Bitcoin allow the tracking of transactions without a central authority, and all transactions are distributed among all participants.
      * But can be applied to other types of technology where data needs to be transparent and have guaranteed integrity.
  + **Lightweight Cryptography:**
    - Specialized hardware and software that is optimized to minimize power consumption and sometimes processing time.
      * Used on systems that run on limited power supply and need to get their cryptography done quick.
  + **Homomorphic Encryption:**
    - Technology that allows the encrypting of data in a way that preserves the ability to perform computations on that data.
      * If encrypted data is processed into something else, that something else should be able to be decrypted into exactly what the data should be if it was unencrypted and processed.
      * This helps protect the privacy of individuals while allowing calculations to be done on their data.
  + **Quantum Computing:**
    - An emerging field that attempts to use quantum mechanics to perform computing and communication tasks.
      * Can potentially defeat cryptographic algorithms that depend on factoring large prime numbers.
      * Can be used to potentially create need cryptographic algorithms way stronger and secure then what we use now.