Cryptography

Symmetric vs Asymmetric

Encryption ciphers are categorized as either symmetric or asymmetric algorithms and this is based on the type of key that they utilize to secure the data. When you're using a symmetric key encryption, you're going to have a single key that's used to encrypt and decrypt the data. With asymmetric encryption, you're going to use two different keys. One key is used to encrypt the data and the second key is used to decrypt it. Beyond the challenge of proving who used the key, there's another challenge with symmetric algorithms and that's the distribution of that shared secret key. So, if you wanted to encrypt emails and send them to your five closest friends, each of you would have to have a set of shared secret keys set up for each of you, so there would be five different pairs of keys.

The second category that we have for encryption ciphers is known as asymmetric algorithms. Unlike symmetric algorithms, asymmetric algorithms do not require a shared secret key. For this reason, they're often referred to as public key cryptography which we're going to discuss in its own lesson. Now, with asymmetric algorithms, two separate keys are used. One is used to encrypt the data and another one is used to decrypt the data.The most commonly used types of asymmetric algorithms are the Diffie-Hellman algorithm, RSA, and ECC.

For example, symmetric algorithms are very popular because they tend to be about 100 to 1000 times faster than an asymmetric algorithm. But asymmetric algorithms allows to overcome the key distribution challenge that we face with symmetric algorithms. Often, though, like most things, implementations are going to use a hybrid approach that combines both of these to get you the best benefits.

Now, in addition to classifying algorithms as symmetric or asymmetric based on their key type, we also categorize an algorithm as a stream or a block cipher based on the mathematical algorithm that they're using to do their encryption and decryption. Stream ciphers perform their computations and encryption a single byte at a time.

Making it a bit by bit process, they utilize a key stream generator to create a bit stream that is mixed with the input plaintext using a mathematical exclusive XOR function and this creates the encrypted cipher text. Because these stream ciphers can perform bit by bit encryption, they are well-suited for securing real-time communication data streams like streaming audio or streaming video. Now, stream ciphers also tend to be symmetric algorithms and they use the same key for encryption and decryption.

A block cipher, on the other hand, is able to break the input into fixed length blocks of data before performing the encryption.Block ciphers are also easily implemented through software solutions where stream ciphers tend to be used in hardware solutions. In fact, most of the algorithms that we're going to talk about in this course are block ciphers, things like DES, 3DES, AES and IDEA.

Symmetric Algorithms

In this lesson, we’re going to cover a little bit of detail about the common symmetric algorithms that you have to know for this Security+ exam. This includes DES, triple DES, IDEA, AES, Blowfish, Twofish, and the Rivest Ciphers, RC4, RC5, and RC6.

DES

The Data Encryption Standard or DES uses a 64-bit key with eight bits of that being used for parity. Therefore, DES only really has an effective key length of 56-bits, which is, you probably guessed by now, means it's not very secured against modern computing power. DES was heavily used in the 1970s and used all the way up until the early 2000s. With DES, each message is broken up in this 64-bit blocks and put through 16 rounds of transposition and substitution to create the cipher text. Due to DES's weakness and its key, a modified version of it known as Triple DES, written as 3DES, was also created. And in this version, there was three 56-bit keys used. The input data was subjected to encryption through the DES algorithm with the first key then decrypted through the algorithm using the second key. Again, this jumbles it up even more and then puts it through the DES algorithm again through another encryption function using that third key. This effectively created an algorithm that had a 112-bit key but it was three times slower than DES because of all those encrypting, decrypting, and encrypting functions.

IDEA

IDEA stands for the International Data Encryption Algorithm and it's another symmetric block cipher which uses a 64-bit block as its input and it uses that to encrypt the data. The key size here is 128-bits and it's faster and harder to break

than DES but it's not as widely used as the more common one, AES, which we're going to talk about in a moment. IDEA is commonly known only because it's really used inside a pretty good privacy suite which we'll talk about a little bit later on in the future lesson. Ultimately, DES and triple DES simply weren't strong enough, though, and so there was this contest held to design a replacement. IDEA was one entrant to the contest but ultimately, it didn't win.

AES

The one that won is known as AES which is the Advanced Encryption Standard. AES was chosen as the replacement for DES and triple DES by the US government. AES can be used with a 128-bit, 192-bit, or 256-bit key, and a matching block size. AES is known as the Rijndael algorithm, as well, which is named after its creator but most people simply call it AES. AES is widely used and it has become the de facto standard in encryption. In fact, it's the encryption standard that's used by the federal government for any encryption of sensitive but unclassified information.

Blowfish

Next, we have Blowfish, which is a block cipher that uses a 32-bit to 448-bit encryption key to encrypt 64 bits of data in blocks at a time. It was originally developed as a replacement for DES but wasn't widely utilized.

Twofish

Another variant, called Twofish, was also developed and this one provides the ability to use 128-bit blocks in its encryption algorithm and use 128-bit, 192-bit, or 256-bit encryption keys. Both Blowfish and Twofish were never patented and they were available for use as open source.

Rivest Ciphers

Another set of symmetric algorithms was created by Ron Rivest, a cryptographer who's created six algorithms under the name RC which stands for the Rivest Cipher. RC1 was never published. RC2 was considered weak originally and skipped over. And RC3 was cracked before it was even released to the public. But RC4, RC5, and RC6 were released and can be found in common use today. RC4 is a stream cipher and it uses a variable key size from 40-bits all the way up to 2048-bits. RC4 is used in both Secure Sockets Layer, SSL and Wired Equivalent Privacy, WEP. Now, RC5 is a block cipher using key sizes up to 2048-bits. And RC6 is based on the RC5 cipher and it was originally considered as the replacement for DES until Rijndael cipher was chosen as the winner and became the Advanced Encryption Standard or AES.

Public Key Cryptography

With asymmetric algorithms, we use a key pair to encrypt and decrypt the data. These two keys are called the public key and the private key. Now, public key cryptography can provide us with confidentiality, integrity, authentication, and non-repudiation for the messages being sent. To provide confidentiality of the data, the data should be encrypted using the receiver's public key. So, if I wanted to send a document to Mary, as you could see here on the screen, I would encrypt that document using Mary's public key. By doing so, only Mary is able to read it, because only Mary is going to have Mary's private key. And this is going to be used to decrypt the contents, ensuring that the message I sent is safe from anybody else's prying eyes.

Now, to provide non-repudiation, the message should be encrypted using the sender's private key, so, in this case, I would use my private key. By doing so, anyone who has access to the sender's public key, which could be anyone in the world, is going to be able to open that message and read it. This isn't going to give us any kind of confidentiality, but I'm not worried about confidentiality right now. Instead, I'm worried about making people know I'm really the person who sent the message. That's the non-repudiation we're working at. This is going to make sure that only I can send it, because only I have my private key.

Digital Signature

We want both of those things. And we also want to add to that list, we want integrity and we want authentication to make sure we have this message being sent where nobody else can read it, and we know who it came from, and that it was never changed in transit. To accomplish all of this with our emails, we often implement a process to create a hash digest based on the message being sent, and then we encrypt that hash digest using the sender's private key. This is known as a digital signature and it provides us with the integrity of the message that's being sent, as well as giving us non-repudiation, because only the sender had access to their private key. Then, we take the message we're sending and we encrypt that using the receiver's public key. This provides us confidentiality, as well. So, now I've got integrity of the message, non-repudiation, as well as confidentiality. To make this whole system of public and private keys work smoothly, there's another concept that we're going to cover in a future section called public key infrastructure, or PKI.

Asymmetric Algorithms

So, in this lesson, we're going to cover the three asymmetric algorithms that you have to know for the Security+ exam. They are Diffie-Hellman, RSA, and the ECC or Elliptic curve cryptography.

Diffie-Hellman

Diffie-Hellman is named for its two inventors. The Diffie-Hellman algorithm is used to conduct key exchanges and secure key distribution. It's used widely when you're setting up VPN tunnels and other encryption tunnels that require a symmetric algorithm's shared secret key, that private key, to be exchanged first before setting up that symmetric tunnel and by using this asymmetric Diffie-Hellman, we can do that. Diffie-Hellman is susceptible to man-in-the-middle attacks, though. So, if you want to secure it, you need to make sure you have some form of authentication, such as requiring a password, or a digital certificate, at the beginning of the exchange process. When you see Diffie-Hellman on the exam, I want you to remember two big things. First, it's an asymmetric algorithm and second, it's used for the key exchange inside of creating a VPN tunnel establishment as part of IPSec.

RSA

Our second asymmetric algorithm is known as RSA. It's also named for its creators, Ron Rivest, Adi Shamir, and Leonard Adleman. RSA is widely used for key exchange, encryption, and digital signatures. The algorithm relies on the difficulty of mathematically factoring large prime numbers and this protects its public and private key pairs. RSA can support key sizes between 1024-bits and 4096-bits. RSA is widely used in organizations around the globe. If you happen to have one of those secure tokens on your key chain, where every 30 to 60 seconds, the six digit number changes and you use that as part of your login and multi factor authentication, well, guess what? You're using RSA! Because that token stores RSA asymmetric one-time use keys.

ECC

ECC is heavily used in mobile devices and it's based on the algebraic structure of elliptical curves over finite fields to define its keys. ECC is very efficient and provides better security than an equivalent RSA key of the same size. In fact, ECC's algorithm is six times more efficient than an RSA algorithm. So, if you're going to have a 256-bit key with ECC, you'd require a 2048-bit key with RSA to be just as secure. For this reason, you're going to see ECC used in a lot of things like tablets, smartphones, and other mobile-based implementations because these devices have much less processing power available than does a standard desktop or laptop. There are a few

variations of ECC, as well, and you might come across these in the field. The first is ECDH, which is the Elliptic Curve Diffie-Hellman, which you might have guessed is an ECC version of our popular Diffie-Hellman key exchange protocol. Another variant is known as the ECDHE, which is the Elliptic Curve Diffie-Hellman Ephemeral protocol, which uses a different key for each portion of the key establishment process inside the Diffie-Hellman key exchange. The final one you might come across is known as the ECDSA, which is the Elliptic Curve Digital Signature Algorithm, which is used as a public key encryption algorithm by the US Government in their digital signatures. For the exam, remember that ECC and all of its variants are most commonly used for mobile devices and low-power computing devices because it gives you equivalent protection to other asymmetric algorithms with a lower key size.

Pretty Good Privacy

Pretty Good Privacy, also known as PGP, is an encryption program that's used for signing, encrypting, and decrypting emails. Over the years, PGP's use has expanded beyond emails, though, and includes an entire suite of protocols that can encrypt emails, files, and even entire hard disks. PGP uses an older algorithm, though, known as IDEA, which you may remember from our symmetric algorithm lesson. Now, you may be wondering, why didn't I just cover PGP all the way back in the symmetric algorithm lesson, then? Well, PGP is actually a hybrid cryptographic tool because it uses a symmetric cipher for the bulk data encryption, but it uses RSA, an asymmetric cipher, to create the digital signatures used in signing its emails and to send the session keys over an untrusted network. This is what I meant before when I said that we often combine both symmetric and asymmetric ciphers to give us a hybrid approach or implementation. Now, PGP uses key sizes of 128 bits or more for symmetric functions and key sizes between 512 bits and 2,048 bits for its asymmetric function. Over time, PGP became an open-source encryption cipher, and as such, it was able to be forked and used for further development. The result of that was GPG. This stands for the GNU Privacy Guard, and it's an implementation of cryptography that's used to provide you with confidentiality in your data by encrypting it like PGP did. Now, GPG is actually a newer version of PGP, or the Pretty Good Privacy encryption suite. The newer GPG uses the more modern AES encryption algorithm instead of that weaker IDEA symmetric algorithm. Now, GPG is a freely-available and non-patented encryption solution that's available for Linux, Windows, and Macintosh operating systems. So, if you're looking for one to try out, you can download GPG and get started today.

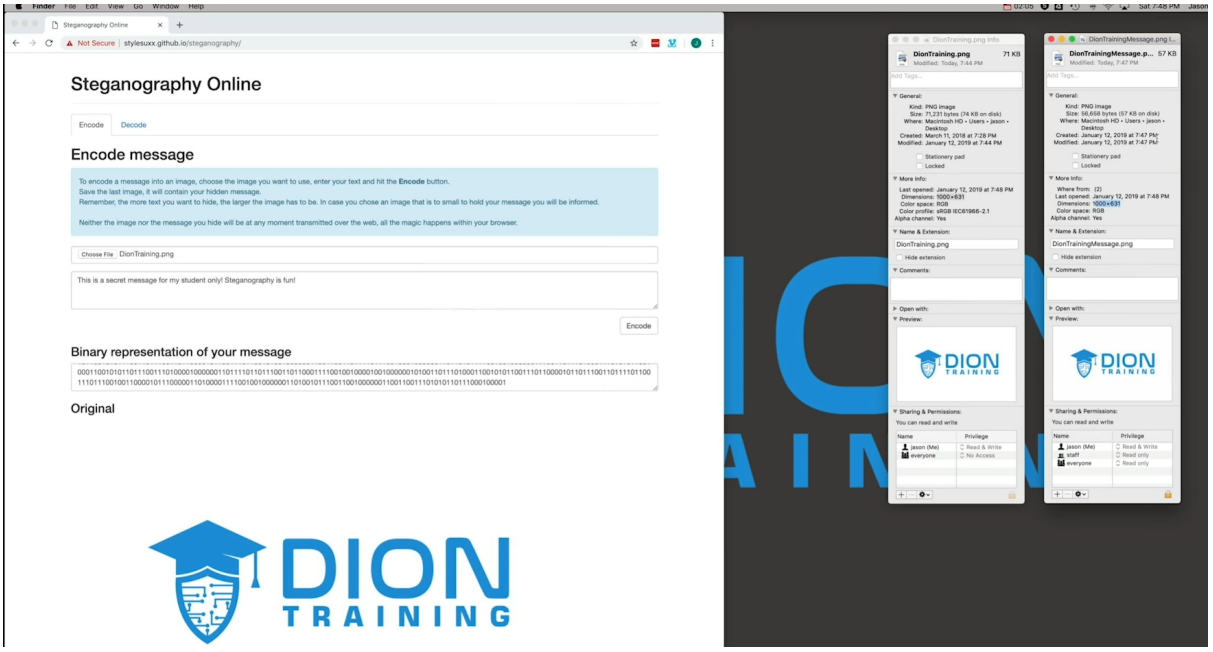
One-Time Pad

This is called a One-Time Pad. Now, a One-Time Pad is a stream cipher that encrypts plaintext information with a secret random key that is the same length as the plaintext input. This secret random key is known as a keystream and it's comprised of a series of random bits. If the keystream is truly random, the only way an attacker could decrypt the information back to the plaintext is if they had access to that randomly created keystream.

Pseudo-Random Number Generator, or PRNG. This is an algorithm that spits out what looks, to you and me, like random numbers, but to other computers, they can figure out what the initial seed was and then that takes away the randomness. These numbers are used for a variety of purposes, including encryption, as well as game development and other things that you might need a random number for. For example, you may wish to have images uploaded by your users and assign a random filename made out of a series of numbers. In truth, though, the numbers aren't purely random, because computer create them based on mathematical functions.

Unfortunately, though, we don’t have a truly random sequence of numbers that we can rely on. So, one-time pads aren't often used. Instead, we use the concept of one-time use passwords. You've seen this discussed inside our multi-factor authentication lesson. For example, you could use a random number that's texted to you by your website whenever you attempt to log in. Or, you might have an RSA secure token that displays pseudo-random numbers that you use to prove you have possession of that token as your second fact of authentication. These are two good examples of where we use pseudo-random numbers in the security of our network.

Demo: Steganography



More

Blockchain

A blockchain is a shared immutable ledger for recording transactions, tracking assets, and building trust. Now, when we talk about the blockchain, you're probably thinking about cryptocurrencies because these are some of the most famous examples out there. For example, Bitcoin has been all the rage for about a decade, and this was really the first commercially available type of thing that was using the blockchain.

Quantum Computing

Now, quantum computing is where we take a computer that uses quantum mechanics to generate and manipulate quantum bits known as qubits in order to access enormous processing power. Now, I know this is a weird definition because in the definition we're actually using both terms inside of it. We're using the word quantum and the word computing, or computer in this case. And I really don't like doing that in a definition, but there's really no better way to explain this. When you think about a classic computer, like the one you're watching this course on right now,

it uses ones and zeros to process information. And the faster you can process those ones and zeros, that means, the faster you can get information done, and that's going to be a faster computer. Well, at a certain point, we can't make our computers really any faster. And we run out of computing capability. So, what we ended up doing was taking single processors and putting in two processors, or we made quad-core, which had four processors, or we made octa-core, which has eight processors. And that's the way we've been able to speed up computers. Well, with quantum computing, it is a completely different ball game. Instead of using ones and zeros, we use these things known as quantum bits or qubits. Now, this can be done in computing or in communications. When we deal with communications, we're talking about quantum communications being a communications network that relies on using qubits made of photons, in our case, light, to send multiple combinations of ones and zeros simultaneously, which will result in tamper-assistant and extremely fast communications.

Qubit

Well, a qubit is really just a quantum bit. It's composed of either electrons or photons, so, it can be electrical or made by light, and it can represent numerous combinations of ones and zeros at the same time using something known as superposition. And this is really the main benefit of using quantum computing, because you're not just having a single one or a zero and you can do multiple combinations of ones and zeros at the same time with this one qubit, you can actually crunch through a wide variety of potential outcomes simultaneously.

Quantum in Cryptography

In fact, asymmetric encryption algorithms, those that are relying on this hard math problem, have been mathematically proven to be broken by quantum computers. Now, the only good thing we have going for us is there's no real quantum computers in use today. The ones they have are only prototypes and they are very small scale, and they have been spending millions and millions of dollars to create these. Right now, the estimate is that we won't have a working quantum computer until at least 2025 or 2030 in some sort of a production environment, maybe even later than that.

Post-Quantum Cryptography

Now, post-quantum cryptography is a new kind of cryptographic algorithm that can be implemented using today's classical computers, but would still be impervious to

attacks from future quantum computers when they are available. Now, there's really two methods that we can use to try to create this post-quantum cryptography. The first method is just to increase our key size, to increase the number of permutations that are needed to be brute-forced. This works well when you're dealing with a symmetric encryption algorithm, something like AES. If I take AES 128 and I increase it to AES 256, for instance, I doubled the key length. But I now have actually squared the number of possible combinations that are going to have to be figured out by the quantum computer. And that extends the time and makes it much stronger and harder to crack. Now, the other way we can do this is by working on other approaches, and researchers are doing this right now. They're looking into things like lattice-based cryptography and super singular isogenic key exchanges.

Ephemeral

The next thing we want to talk about is ephemeral. When we talk about ephemeral cryptography or ephemeral keys, we're talking about a cryptographic key that's generated for each execution of a key establishment process. Essentially, when you hear the word ephemeral, I want you to think about the fact that it is short-lived, it's something we're going to pick for a short period of time and then throw away.

Homomorphic Encryption

Now, homomorphic encryption is an encryption mechanism that allows calculations to be performed on data without decrypting it first.