CS 240: Symmetric Key Encryption Transcript

[00:00:00] First, let's talk about symmetric key encryption.

[00:00:03] Now again, a symmetric key encryption algorithm is one where the, the same key is used to both encrypt and decrypt the data.

[00:00:13] So you can see here in this diagram on this, this slide here that, that Bob wants to send some data to Alice.

Start visual description. The professor demonstrates a diagram showing Bob sending data to Alice using symmetric key encryption. The data "Hello, Alice" is encrypted with a secret key, resulting in random, gibberish-like encrypted data. This encrypted data is then sent to Alice, who decrypts it using the same secret key to retrieve the original message. End visual description.

[00:00:21] And so um so hello, Alice is the data. He's encrypting. So, he passes that data through the encryption algorithm and one of the inputs to the algorithm is the key, the secret key.

[00:00:34] And so there's two inputs to the algorithm, the data itself and then the key and then the output of the encryption algorithm is the encrypted data that looks very random and it's just gibberish.

[00:00:46] So that's the output. And then he would send the encrypted data over to, to Alice.

[00:00:52] And then she would decrypt it by running it through um the algorithm again in this case, running it in decryption mode.

[00:01:00] And she would use the same key to decrypt it as Bob used to encrypt it.

[00:01:06] And then the output of the decryption algorithm would be the original data.

[00:01:09] Hello Alice.

[00:01:11] Now, the reason they sometimes call this secret key encryption is that it's very important that this, this key that's used to encrypt and decrypt is, is kept secret.

[00:01:20] It's important that only Bob and Alice know what that is because if everyone else knows what that is, then there's no security here at all.

[00:01:26] So they're going to have to be really careful to keep that, that key secret.

[00:01:32] All right. So that's, that's the basic idea.

[00:01:34] So what are the algorithms that people use for symmetric encryption? Um Historically, we had the data encryption standard or des and that algorithm had a 56-bit key and back in the day, which was pretty secure.

Start visual description. The professor discusses the evolution of symmetric encryption algorithms, starting with the Data Encryption Standard (DES) and moving to Triple DES, which runs the data through the DES algorithm three times with different keys. The professor then introduces the Advanced Encryption Standard (AES), the current preferred algorithm, and explains its key sizes and initialization vector. End visual description.

[00:01:49] But as, as computers have become more and more powerful, um you can trivially break the DES algorithm with, with a 56-bit key. And so, um once DES was compromised, then they moved to triple des or triple des and triple des basically just runs the data through the DES algorithm three times with three different 56-bit keys.

[00:02:12] And so that was um an early effort to, to strengthen DES has just run the data through it three times.

[00:02:19] And eventually, you know, that became so that it wasn't secure enough.

[00:02:23] And so now we have uh more modern algorithms that we actually currently use.

[00:02:27] Probably the most commonly used uh symmetric key algorithm is AES, which stands for the advanced encryption standard.

[00:02:36] So this, this algorithm is blessed by the, the national security agency in the USA.

[00:02:41] And what you'll see in this realm is that um NSA, one of the things they're responsible for is developing security algorithms and sometimes they run contests where different researchers who developed their own algorithms can, can basically enter their algorithm into the contest.

[00:02:59] And what NSA will do is they will, they will select the winner.

[00:03:02] So whoever wins the context basically becomes the blessed algorithm.

[00:03:06] So currently the uh the uh certified symmetric key algorithm is, is AES that doesn't mean you can't use other algorithms.

[00:03:16] It just means that this is the one that they, they prefer.

[00:03:19] Now AES can be used with different key sizes.

[00:03:24] You can have 100 and 28 bits key or a 192-bit key or a 256-bit key. Of course, using a 256-bit key is going to be more secure.

[00:03:32] So if you're interested in maximizing your security, you probably want to use a 256-bit key.

[00:03:40] AES also has another uh parameter that you have to pass in.

[00:03:45] It's, it's almost like a second key, it's called an initialization vector.

[00:03:49] So actually when you run AES, you need to have a key, but you also need to have a, a 128-bit initialization vector.

[00:03:56] And you can think of both of these, these values as just random arrays of bytes.

[00:04:02] And um so to run AES you actually have to pass in your key and your initialization vector.

[00:04:11] Another um algorithm that people commonly use is called Blowfish.

[00:04:16] And it has a key size anywhere between 32 bits and 448 bits.

[00:04:21] You can kind of pick your key size.

[00:04:22] And so that's nice that you can adjust the key size to whatever you, you need for security.

[00:04:27] So that's another algorithm. It's not the uh you know, the, the current favorite of the NSA, but it, it's quite um secure and I, I don't know of any uh attacks that people have been able to make on it that have really compromised it much.

[00:04:43] There's another algorithm sometimes you'll hear about it's called two fish.

[00:04:47] And uh you know, if you get on the internet, there's, there's tons of, of algorithms that are out there, but these are probably some of the most commonly used ones.

[00:04:56] AES by far.

[00:04:57] That's kind of the, the default go to algorithm that, that you would probably use.

[00:05:02] Now let's move over to a code example.

Start visual description. The professor transitions to a code example, demonstrating a Java program that uses the AES algorithm to encrypt and decrypt data. The program encrypts and decrypts its own source code, showing the process of passing data through the AES algorithm with a secret key and initialization vector. End visual description.

[00:05:11] So there's an example called symmetric key encryption demo. So, um again, we're making heavy use of java's built in cryptography and security libraries.

[00:05:22] Java X dot Crypto Java dot security.

[00:05:29] OK.

[00:05:30] So this, this is a program that shows how to use java to use the AES algorithm to encrypt and decrypt some data.

[00:05:38] Now, what data are we going to encrypt and decrypt? Well, I decided that we would have this program encrypt and decrypt itself.

[00:05:45] And so the input or the data that we're going to encrypt is the source code for this program.

[00:05:52] And so that's our data.

[00:05:54] And if we scroll to the bottom here, you can see what the code looks like to run AES on some data.

[00:06:01] So this little run AES um method uh First parameter is cipher mode.

[00:06:07] So you have to tell this method whether it's encrypting or decrypting.

[00:06:10] So there's two modes, you're either encrypting or you're decrypting.

[00:06:13] So just pass one of those in the input stream has the data contains the data that you're encrypting or decrypting.

[00:06:20] Then the output stream would receive the data, the output of the encryption or decryption algorithm.

[00:06:26] And then uh the next parameter is the secret key.

[00:06:29] So that would be in this case uh for AES, it would be the, the 256-bit key for example.

[00:06:37] And then we have the initialization vector for AES.

[00:06:39] So that's the 128-bit initialization vector.

[00:06:42] So like I said, kind of like it has 22 keys instead of one and um to run AES on the data.

[00:06:50] And the first thing you do is you call java's cipher class. So, cipher has a static method called get instance and you just pass it a string that tells it what algorithm you want an instance of.

[00:07:01] So in this case, we're going to say AES.

[00:07:04] And after it says AES, then we pass in some other parameters, there's different parameters you can set on AES.

[00:07:10] So this string would, would indicate um what flavor of AES we want.

[00:07:16] And then once we've got our cipher, which is our algorithm, then we just initialize it with the mode.

[00:07:22] We talk, if we're encrypting or decrypting, we pass in the key and the initialization vector and now it's ready to go.

[00:07:28] And so once we've initialized the algorithm, then all we have to do is read all the data from the input stream and pass it through the cipher.

[00:07:38] And so you'll see in this little loop down here that we just read chunks of data from the input stream. So, we call input stream dot read, we get back an array of bytes.

[00:07:49] So we're going to read 64 bytes at the time it looks like.

[00:07:52] So we read the next 64 bytes and then we take those bytes, and we run them through the cipher or the encryption algorithm. We call update, pass it the, the new bytes that we have and then it will pass us the encrypted uh bites back.

[00:08:09] And then we just take the encrypted bytes, and we append those to our output stream, and we just keep doing that until we run out of data.

[00:08:17] And then at the very end, there's a last step where you give the, the cipher an opportunity to do any uh finalization that it needs to do any steps that it needs to do to complete its algorithm and that may return some more output bytes.

[00:08:31] So we pin those to the output stream as well and that's pretty much it. And then we uh then we uh yeah, we have our data.

[00:08:41] So all the encrypted data in that case would be in the output stream.

[00:08:44] Now, it is important to remember that you can run this method in encryption or decryption mode.

[00:08:49] I kind of described it as if we're encrypting data, but you can also pass in decrypt as the mode and the whole thing runs the same except it's, it's decrypting instead of encrypting.

[00:09:00] And so you can see how easy that is to um incorporate that encryption into your, your application.

Start visual description. The professor explains how to generate keys and initialization vectors using Java's built-in key generation algorithms. The professor demonstrates creating a 256-bit key and a 16-byte initialization vector using Java's secure random number generator, emphasizing the importance of secure random numbers in encryption. End visual description.

[00:09:08] So I've got another couple of methods here.

[00:09:10] I've got an AES encrypt method and AES decrypt, but they just call a run AES method with either encrypt or decrypt. So, they're just for convenience.

[00:09:22] Now, the next thing to focus in on here is where do these keys come from? We said we need a 256-bit key, for example, and we also need an initialization vector.

[00:09:31] Well, how do we create those? Where do they come from? Well, Java also has built in key generation algorithms. So, you don't need to generate your own keys.

[00:09:41] You can just call java's um key generator and um it'll generate suitable keys um for whatever algorithm you're dealing with. And so, in this case, um we have to create AES key method.

[00:09:56] And Java has a class named key generator which has a static method.

[00:10:00] Get instance, we just pass at the algorithm for which we're generating keys, get back a key generator.

[00:10:07] And then we ask the key generator to initialize itself.

[00:10:10] We're, we're going to basically say we want a 256-bit key.

[00:10:14] And then we ask the generator to generate the key, and we get back a secret key.

[00:10:18] So it's just that easy to, to generate a key.

[00:10:23] And then it's kind of similar for the initialization vector except um really the initialization vector is nothing more than a random number.

[00:10:33] That's a big random number. It's a 16-byte random number. So, to generate an initialization vector, what we'll do here is we'll use Java secure random number generator.

[00:10:45] Now, what's, what's a secure random number generator? Well, it's, it's kind of like just a regular random number generator except it's, it's um it's harder to crack, it's harder to hack.

[00:10:55] It's more secure, it's more random than a regular random number generator.

[00:11:00] And so all we do is we create a secure random object, and we ask it to generate us a 16-byte random number and that becomes our initialization vector. So that, that's pretty easy.

[00:11:12] So a lot of security algorithms require generating secure random numbers.

[00:11:17] So that's a useful class and so once I've got the keys, then it, it's trivial to, to call uh the AES algorithm.

[00:11:25] All I have to do is generate my keys, my initialization vector and then I can encrypt and decrypt the data.

[00:11:32] And so you kind of get the idea there.

[00:11:34] So it's not hard to do.