Symmetric ciphers – use same key for encryption and decryption.

2 types

Transposition ciphers -diffuse data in the plaintext.

Substitution ciphers – replace data in the plaintext.

Transposition cipher – rearrange characters in plaintext to form the ciphertext.

Ex – rail fence cipher

Writing plaintext in 2 rows down then across

Reading across then down.

H L O

E L

Substitution cipher – replace characters in plaintext to form ciphertext

Intro to crypto

You might wonder at the fact that we're now moving from, let's face it, a very theoretical

part of the subject, the part with policies, ACMs and the like, into what surely is a very,very concrete part, the cryptography section.

Why do I prefer to go straight to cryptography?

Well, that's because cryptography, you see, is such a ubiquitous tool for the security professional.

The things that we understand about cryptography, we will be able to apply them to things like

authentication and access control, and even other things we'll touch upon during the course.

Okay, having said that - Cryptography. It has been the domain, traditionally they say, over the ages, the domain of warriors and lovers, because those are the two groups of people who've needed to keep secrets from other people while sharing them with some.

Otherwise, well, let's start with the idea that I want to keep a secret.

I want to be able to share it with certain groups of people, or indeed I just want to remember it, because I can keep secrets in my head, but sometimes we might need support from our memory.

So what we're talking about is in some way externalising my secrets, and at the same time keeping them secret from others.

In the Second World War, you see, the American troops, they would employ the services of members of the Navajo tribe.

And the Navajo language, you see, is so very obscure, so very few speakers of it, that they were more or less guaranteed that none of the enemy would ever be able to decipher what the Navajo Indians were saying.

So there we have it. All we have to do is create a new language, a language that other people won't be able to understand within reasonable time, and teach a few people to speak that language, and yeah, simple method.

I'm being sarcastic.

Already with this example, I think I'd like to introduce the idea that there's a matter

of cost, as there is in all matters of security, and here we're looking at the cost of creating

a new language, and the likelihood that this new language will not be able to be analysed

by an opponent.

So nevertheless, this doesn't seem to be a very effective method.

It's not generally the way we go about things, let's face it.

Rather than creating a brand new language, what we tend to do is to take a

language we all understand, and in some way transpose it, change it, mix it up, so that the mixing up is secret.

We use a mixing up that allows people to unmix them if they're amongst our little group of people who can share the secret.

So now we're getting to the point of the lecture, which is cryptography, and I'm just making

the point here that crypto comes from the Greek, meaning more or less hidden, and graphy for writing.

So that's what it's about.

We're trying to write things down, or communicate things, and at the same time hide them. OK, so can I think of a better method to go about keeping secrets than inventing a whole new language?

Yeah.

So let me suggest a method for doing this.

This method has a name, Caesar Cipher.

Let's say that I have a secret.

It's a clandestine meeting with somebody called Maggie.

It's at Thursday at 7pm in the evening, and I want to make a note of this, but I don't

want my wife to find out about it.

If I were to apply the Caesar Cipher, it would look something like this.

I now take a copy of the alphabet in which I express my messages.

I then take the first three letters, shift them to the end of the alphabet, and voila,

I have a mapping from my clear text.

This is my clear text.

I take the letters of the clear text and simply map them in the Caesar Cipher.

Taking my message, applying this method, that is what I get.

A secret.

A secret from my wife.

Right.

So, now it's time to start considering important things like, is this a good method or not?

I'm going to ask you to consider all the reasons you think why this might not be a very good method.

What makes Caesar cipher weak

Now for goodness sake don't imagine that the Caesar Cipher is a real proper method that we could use today.

There are possible uses but in general if you bring up the Caesar Cipher as a real method at the exam you're on very thin ice, unless you can motivate it well.

We will see that Caesar Cipher is horribly weak for a number of reasons and we'll go into that.

But it's clear that we need a way to improve on this kind of basic cipher and that's why I introduce it.

If we can take small steps now then maybe we can understand what is important to having a good cipher.

We go from a bad cipher, Caesar Cipher, in small steps to see how far it can get us.

Now the first step you could say, Caesar Cipher by the way is reputedly invented by Julius Caesar himself.

And if you know your Roman history you'll know that the person who succeeded Julius Caesar, that was Augustus Caesar.

And it is said then that Augustus Caesar also invented a cipher.

It was remarkably similar whereas the Caesar Cipher did a shift of three letters in the alphabet to get the mapping.

The Augustus Cipher is said to have just been a shift of one letter.

Okay. Is this much better? Probably not, right?

But nevertheless a very interesting thing happened between the two.

Because as I say the two methods are very similar and I'll identify the similarity for you.

It's that what we're doing is we're taking the Roman alphabet and we're shifting it a certain number of steps in order to create a new mapping.

Okay. I'd like to call this the alphabet shift method.

It's not something you'll find clearly written in the books but between us let's call this method the alphabet shift.

So we have an alphabet shift method where Caesar used three steps and Augustus used one step.

The important thing that's happened is we've extracted a part of a general method and boiled it down to a key.

Okay. The key for how many shifts you have to do for Caesar was three.

The key for how many shifts you had to do for Augustus was one.

And the idea of having a general method that could be used time and time again but extracting a part of it, and that's the part that you keep secret, that's going to become a very powerful idea.

So, powerfully enough for me to now give you some terminology.

We've seen cryptography and now we're going to introduce the idea of key and lots of associated ideas.

Right. Because this is such a powerful idea I've built a machine to help me use it.This is my alphabet shift machine.

It works like this.

The key tells me how far I should shift the inner circle.

So I'm shifting it anti-clockwise.

For the Caesar cipher then.

One, two, three.

To encrypt a message what I do is I follow from the outside in.

I take an A and map it to a D.

Decrypting is the opposite.

I take a letter from the inside and map it to the one outside to decrypt.

Now we have two important concepts to encrypt or indeed to encipher and to decrypt.

To decipher.

The differences are subtle and we probably won't need to go into that during the course.

OK.

Note how the key is the same in both encryption and decryption.

It's the methods that's kind of difference differs a little bit.

In one case going one way and in the other.

OK.

I can already now show you a little interesting game we can play.

Let's say that the key here is 13.

There we go.

One, two, three, four, five, six, seven, eight, nine, ten, eleven, twelve, thirteen.

OK.

This is a special case of the alphabet shift method.

If you'll notice.

Now if I encrypt then I map an A to an N.

OK.

Normally I would decrypt by going the opposite direction.

But look at this.

In this particular case I can do the decrypt mapping by going from the outside in.

N goes to A.

That's maybe kind of fun.

Maybe it's kind of profound.

At least I thought I'd throw it in there for you.

This happens to be the method that is called ROT13.

Which you would have seen 25 years ago used quite a lot.

A simple way to hide a text is to use an alphabet shift with a key of 13.

Open design of ciphers

Now, why is this idea of keys so important to us? Think about that. It definitely is

the case that with the Navajo language, imagine that somebody did manage to once start deciphering

messages sent in Navajo. Imagine that one member of the Navajo tribe was bought over

by the enemy and now all these messages could be decrypted by that individual. What's happened

now is that we've had a very expensive method developed and with one little chink in its

armour we can completely wipe it out as a useful method. So that's a very dangerous and expensive way to go.

If we instead make a key to be the vital part of our secret,

the vital part of our algorithm to encrypt and decrypt, then maybe we needn't worry so much about the algorithm or even the secrecy of the algorithm. All we need to worry about is the key and if the key were to be discovered all is not lost because we can still use the same algorithm but just use a different key, hopefully.

And that starts to be a very economical idea and a powerful idea. Moreover, and this is an important

point, if the secrecy of our encryption is not in the algorithm but in the key, maybe we can allow the actual algorithm, how it's done, to be entirely in the open.

We can make it open source ,if you like. We can allow the algorithm, in this case alphabet shift, we can allow that to be used

all over the world. We'll say, I've used alphabet shift. The point being that if you don't know the

key, then you don't know how to decrypt it. But moreover, everybody knows the algorithm and that's important for several ideas.

The first being that we can standardise cryptography because so long as we're all trying to make our own secret methods, then we don't want to share the method with

anybody else.

So to talk to somebody else beyond our group and still have secrets with them, we will

have to introduce another method and another method. So in order to avoid an explosion of methods,

let's try and make an algorithm where the actual workings of the algorithm aren't part of the

secret. It's the key that's the secret. Now this is kind of a weird and interesting idea because I'm going to call this idea openness. And the big surprise is that I'm talking about security and to improve security, there are cases where it's an advantage to me to be open about them.

In this case, we like to be open about the algorithm. There's another reason to consider beyond the possibilities for standardisation.

If I write an algorithm for cryptography, then I'm relying entirely on my own expertise to make sure that it's not easily crackable. If I instead were to publish my algorithm in all sorts of scientific journals or conferences,

then I can at least hope that other friendly academic types, mathematicians, will find it interesting to analyse my algorithm and let me know if they find fault in it. So we can get community effort around making stronger and stronger algorithms.

There's a name for this principle as well. We can call this, or it is often called,

the many eyes principle. That something can be more secure if it's not secret in terms of the algorithm, because

many people can and hopefully will look at it, analyse it and find fault with it and make sure that the algorithm

becomes stronger.

Keyspace

So, in the midst of our discussion about cryptography, we got into interesting issues of openness

or as we'll call it later on, the principle of open design and the many eyes principle.

And indeed, cryptographers generally speaking will say that an open algorithm is a better algorithm for the reasons we've given. Nevertheless, I think we can realise that we haven't got

terribly far, even with the idea of a key and an open algorithm. It's still a weak method,

a horribly weak method and one of the main reasons for that now is something that I will

call keyspace.

The point being that we can have keys that alter the way the algorithm works, but in our alphabet shift method, we have a very limited set of different keys to choose from.

Our keyspace is going to be small. How small? I'll let you think about that for a moment.

How small or large is the keyspace for alphabet shift methods that we've seen with the Roman alphabet?

Well, if you said 26 different values, then that was good going. Yeah. Why 26? Well, because my machine has 26 different values.

However, there's one small nasty point that I should bring to your attention and that is that in the alphabet shift method,

a value of 0 or 26, depending on how you like to look at it, doesn't do us any good because if I encrypt with this,

I get the same thing as I started with and my wife will be able to read it straight off, understanding the message entirely,

even though I've encrypted it. So, strangely enough, there's one key amongst all of those that I could have picked that I should avoid using.

So, apart from that one key, I think we can say that we have now a keyspace in this method of 25.

Now, what that means is, one of the important ways for me to, for somebody else to discover my key is really just to test different values one after the other.

So, if I find an encrypted message, I first start with this mapping and try decrypting it and I'll see if it seems to make sense.

And if it doesn't, then I go on like that and so on, until all of a sudden the letters pop out at me and seem to be a language that I understand,

like English, and it says Maggie, Thursday, 7pm. OK? In general, let's say it might take me up to a second to do one check.

I do like that and quickly check the first letters and, no, it obviously wasn't that, and I go on and do that. OK?

We see that with a keyspace this small, and even with a method that is practically speaking this slow, it's not going to take me a long time to decrypt things.

On average, it's going to take me, what, well, around about 13 seconds, with one second to try.

Sometimes I'll be lucky and get it quicker, sometimes I'll be unlucky and I'll have to go all the way round.

But now we come to the idea that I can do a, sorry, what I'm going to call a brute force search through all the possible key values.

Try them out in the algorithm and then see if they work.

Right. The keyspace is so critically small in this method, I don't think we can save it.

Maybe we could still use this in some circumstances. We still have this idea of a trade-off between cost and risk and how powerful our methods are.

And, you know, if I were to write this message in my diary, maybe my wife would see it and just think, oh, that's just Alan playing silly games again, and she wouldn't even put in the effort to going through all the key, different key values.

So maybe it'll work, it depends on the situation. But if we're looking for good general methods, I think it's time to forget this alphabet, that shift method.

Improving keyspace

So now I've introduced the idea of the alphabet shift method and then very

quickly afterwards I said it's not a method we can use because it's never

going to be good and something interesting that happens at that point is

that very often a lot of my students will be putting their hands up saying

but wait a minute wait a minute can't you just do this do that can you if you do like that doesn't it become much stronger so now if that's the case this

is the time for a word of warning. I do not recommend that you try and devise

your own encryption algorithms.

It might help to play the game and consider can I

make it stronger but unless you're a very good mathematician please don't try to invent new cryptographic methods

cryptanalysts, they're very clever people they have

generations of experience and an awful lot of machine power so you can have a hard time outsmarting these kinds of people.

Having said that let's consider the idea of what we possibly could do to make it stronger and the important point

we can expand on the number of choices we're definitely going to make it more

work. If there were only 25 choices and it took a second each choice, but why not

make a hundred thousand different choices in our keyspace and then on

average it would take 50,000 seconds to break that key. But why stop there maybe

we should just keep on expanding expanding on the keyspace and that is

what we do in modern cryptography because we have computer machines we

have to be assured that there are no computing machines today in existence

that will be able to run through all of my keyspace that is if I want to be

really secure. So how feasible is this? How large keyspaces can we find?

Well

you can have a look at the algorithms that Bishop talks about in the book you could have a look at some of the common ones out on the net.

Wikipedia is a good source in that respect and you should be able to get an

idea of how large the keyspaces are.

There's a very closely related concept that I should introduce and that is that of key length. The longer the key is for

more combinations it allows and therefore the more different versions we

will have I we're growing the keyspace so these are very close ideas.

You expand the key length then generally speaking, not always but generally speaking, you'll

find that that has pretty direct correspondence to the size of the keyspace.

Scytale

Let's take a little diversion from the track we've been following so far. I'd like to introduce another cryptographic method, a nice simple toy method again,

not one to be used in real life. Although this is a method that is said to be used,

to have been used thousands of years ago. It's an old Greek method and forgive me

if I try my Greek. I believe the Greek name for this is "scytale". At least when it's

written out and an Englishman pronounces it, we can call it a "sky tail". And here it

is.

You'll see it's basically a staff and around this staff I've wound a piece of long strip of paper. In ancient Greece it would apparently be a long strip of leather.

Right and on this scytale or scytale I've written a message. Okay and I can read it

by turning my staff around. It seems to be an important message. I hope I'm not giving

anything away by showing it to you. But yeah, the way this method now works is that I write

my message like this and then I dismantle my staff. Thus, so now I have a strip of paper

or a strip of leather or something thin with a lot of jumbled letters on it. So this is

cryptography again. I've taken a message and I've transformed it in some way so you can't

read it. Now this is actually a fundamentally different method from the one we just saw

with the alphabet shift because what I was doing with the alphabet shift was I was taking

letters from my language and substituting them for another letter. Substituting for

them for something else. What I substituted them with was fairly arbitrary. I could have

had little pictures instead of letters of the same alphabet just shifted along. Okay

but that's the basic principle to take something and swap it out for something else. With this

method you see I've not changed these letters at all. They're still there. They're still

in there. Okay the difference is that I've jumbled them up. I've swapped the order of them.

Okay so now we've got two more concepts and I've lost my pen.

There we go.

The first method I introduced I can summarise as being a kind of

substitution. Could I get this right? Substitutional method.

I call it that because I'm substituting letters from my real message with other kinds of letters

or symbols or whatever. Okay what I've done here is keep the letters but jumble them up

and there are several names for this method but I'm going to call it a transpositional

method. Forgive my bad handwriting. Okay and this is quite as valid a encryption method

as the earlier one then. You will notice that this is also a method that has a key to it.

Can you consider for a moment while I wind this up what's the key to this method?

Okay well you have to consider that when I'm going to send this message to somebody

I take a staff like this and wind it around, wind the message around. If I'm sending this

as a message I'm presupposing that somebody can wind this thing around a staff of exact equal

diameter. Diameter must be the key in some way. If you don't have the same diameter staff then

you won't get the same combination of letters. Okay another way to see this if I unwind this again

you'll see, do it the quick way,

actually it's kind of a key of four because I had room for four rows around here so you see

if I pick each fourth letter I would get p one two three r one two three o and I start to spell

r one two three o and I start to spell out problem. Okay but that's the basic principle

and we can see that it is a keyed method. Once again not a very good one, easy to discover,

easy to try staffs of different sizes and see which one seems to make sense when I start to wrap

the strip around. So we've looked at these toy methods but I have another one.

I'm sorry I don't have one with me but I'd like you to imagine a Rubik's Cube. I hope we've all

seen them and know what they look like. Maybe I'll try and see if I can find mine to bring it

to class sometime. But you know I have a Rubik's Cube. Now imagine a Rubik's Cube

where I peel off all of the colours first. I then stick white labels all over the cube

and then I write my message one letter. I might even swap around the angle of the letters

so you don't know which way I wrote it. But I wrote one letter on each square of my cube

so that as I hold it up I can read it by turning the cube around.

Imagine that and then imagine that my key is that I turn parts of the cube,

certain number of turns in a certain angle like that, for just say five or six turns.

And I write that down as the key for my Rubik's Cube twist method.

Now cleverer mathematicians than I have worked out that the number of combinations for a Rubik's Cube

is, I'll write this down,

43.25 quintillion. And that's the international quintillion. That is to say 10 to the power

30. Sorry, 13. Yeah that's right. 43 with 13. No what am I saying? Quintillion 18.

I get it right. Get it right. Yes 18. Here we go. So after the point here we'll have 18 places.

After the 25 we'll have 16 more zeros as an approximation. Okay and that's a simple device

like twisting a Rubik's Cube. So with that simple example I think we're getting into the realms of

very healthy key spaces because if you consider how long it would take you to go through all the

combinations of a Rubik's Cube by trial and error, by brute force, then you'll find that

these kinds of numbers will take you up into more than your lifetime. Some of these key spaces that

we are developing mean that if you, in order to make your way through the whole key space, you

might have to spend more time than the length of the universe, the projected length of the universe.

These are numbers that are so vast they're equivalent to more atoms than are known in the

known universe. So we are getting to literally astronomical numbers. The good thing is that we

have computers these days so we can develop algorithms that allow us to use key spaces of

these enormous sizes. Now let's go back to the idea of cost.

You remember that I said about my very simple Caesar Cipher or simple alphabet shift that maybe

I don't have to develop something more powerful for my wife's sake. Maybe it's good enough as it is.

Does the same apply with these things? Do we need to separate out our algorithms to the weaker ones

and to the stronger ones and because the stronger ones we can expect to cost more? Well the interesting

thing there is that computers are so good at this and the methods that we've developed are so good

that it doesn't cost a lot more to use a fairly weak method, sorry to use a fairly strong method

compared to a fairly weak method. The cost difference isn't that great. You possibly will

notice it when you're working with cryptography that sometimes creating keys will take a lot

longer as the first step. You might find that some algorithms take a little bit longer so you start

to get a little bit edgy and isn't it going to be done soon kind of thing. But it's certainly not

something that would spoil your day. So in that case, in this cost-benefit calculation,

you should understand that we now have the possibility for everybody, even me,

keeping secrets from my wife, everybody can afford to use methods that are in principle

safe for the projected lifetime of the universe.