

Mechanics: Encryption

In this lesson, we'll continue our discussion of HTTPS mechanics with how encryption works.

We'll cover the following

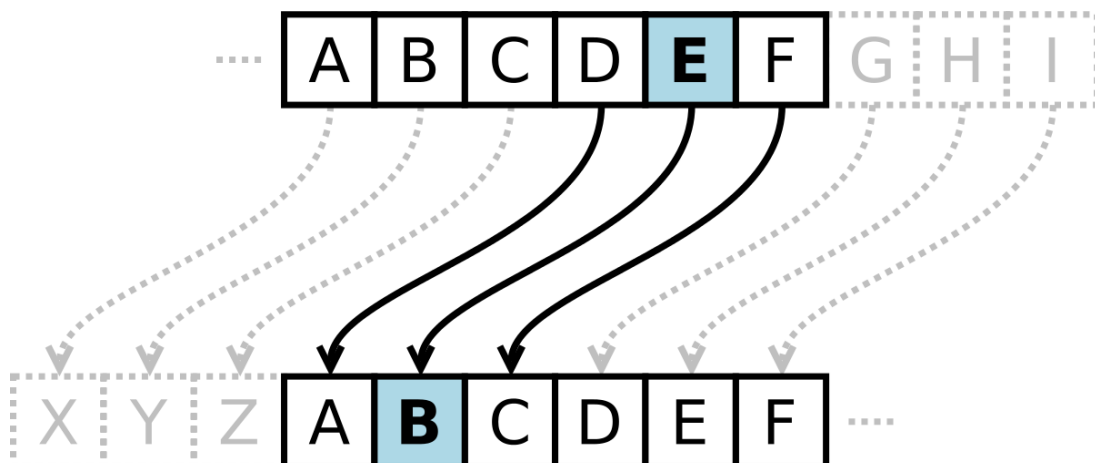
- Why encryption?
- Encryption via shared secret
- Try out encryption!
- Secure exchange of the secret

Why encryption?

We're halfway to securing the communication between you and your partner. Now that we've solved authentication (verifying the identity of the caller) we need to make sure we can communicate safely without others eavesdropping in the process. As I mentioned, you're right in the middle of a meeting and need to spell your online banking password. You need to find a way to encrypt your communication so that only you and your partner will be able to understand your conversation.



Encryption via shared secret

You can do this by establishing a shared secret between the two of you, and encrypt messages through that secret. You could, for example, decide to use a variation of the [Caesar cipher](#) based on the date of your wedding.



Try out encryption!

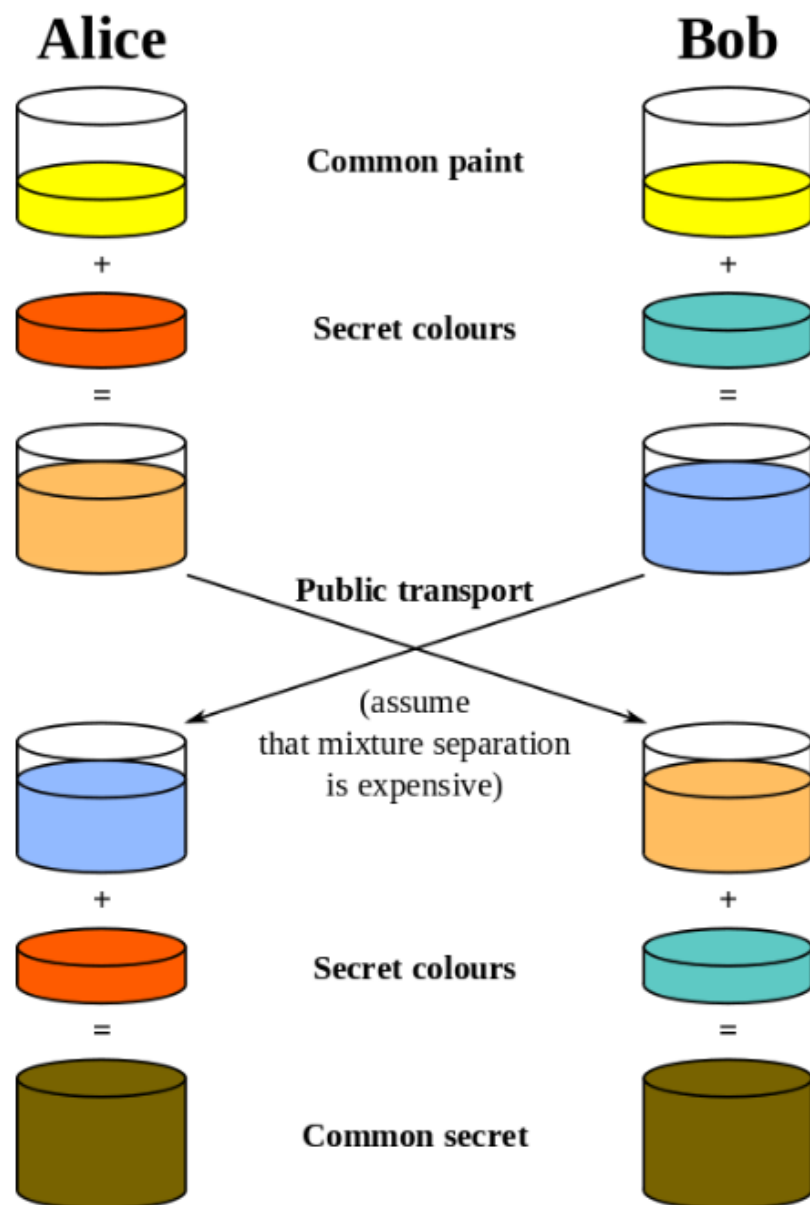
Try encrypting your first name using the Caesar Cipher app below!

Output
JavaScript
HTML
CSS (SCSS)
<div><div>clear text</div><div></div><div>shift</div><div></div><div>calculate</div><div>Cipher text</div></div> <div><div></div><div></div></div>

This would work well if both parties have an established relationship, like you and your partner, as they can create a secret based on a shared memory no one else has knowledge of. Browsers and servers cannot use the same kind of mechanism as they have no prior knowledge of each other.

Secure exchange of the secret

Variations of the [Diffie-Hellman key exchange protocol](#) are used instead. This ensures parties without prior knowledge can establish a shared secret without anyone else being able to sniff it. This involves [using a bit of math](#), an exercise left to the reader.



An illustration of the Diffie-Hellman exchange protocol, where a common secret is established over a public channel

Once the secret is established, a client and a server can communicate without having to fear that someone might intercept their messages. Even if attackers do so, they will not have the common secret that's necessary to decrypt the messages.

i More on Public Key exchange algorithms

For more information on HTTPS and Diffie-Hellman, I would recommend reading "[How HTTPS secures connections](#)" by Hartley Brody and "[How does HTTPS actually work?](#)" by Robert Heaton. In addition, "[Nine Algorithms That Changed The Future](#)" has an amazing chapter that explains Public-key encryption, and I highly recommend it to Computer Science geeks interested

in ingenious algorithms.

In the next lesson, we'll study HTTPS more closely.