Encryption

Hunter Jorgensen

What is Encryption

Encryption is the process of turning data into indecipherable text.

Encryption is often done at rest (the data in a database is encrypted) or in flight (data is encrypted as it it passed between client/server).

Encryption takes a key (i.e., password or secret) and combines it with unencrypted data (known as plaintext; such as a user’s password, PIN, social security number, etc.) and, using a conversion method (or algorithm), used to produce the encrypted representation (also known as cipher text.)

The process of converting ciphertext back to plaintext is known as decryption.

Let’s Encrypt!

For this, we will be using a library called bcrypt and demonstrating what it means to have encryption at rest for our users passwords. Looking at our data, if a malicious actor were to access our database, they could very easily get our users passwords and we don’t want that!

bcrypt, aside from the name of the library, is also a famous hashing function meant for passwords on UNIX. This might be a good discussion post explaining how it works!

Setting Up and Thinking Through

To install: **`npm install -s bcryptjs`**

There are 2 points places where we need to use bcrypt:

1. When we insert the data into the database so that it is ENCRYPTED
2. When we pull the data out of the database in order to authenticate a user so that it is DECRYPTED
3. Optional 3rd case: if users update their password, we will need to do both of these!

Note: we're a little careless with our passwords in this course since this is more for understanding. What are some ways we can structure our data or design our APIs so that we hide our password data as much as possible?

Option 1: Encrypt within the Controller

Our first option is to encrypt the within our own logic in the controller. This is a pretty small change!

Demo

Option 2: Encrypt With Mongoose

Our other alternative is to use Mongoose’s ‘hook’ API so that we can add logic to its existing behavior (it *hooks into* the logic). In our Schema, we can use both the ‘pre’ and ‘methods’ API to add our custom behavior:

Demo

Note that Option 1 and 2 are not really compatible (they CAN work together, but they’re not meant to).

What would the benefit of using Option 1 vs Option 2? And vice-versa?