Bcrypt Handout

# Bcrypt Examples

// @route POST encrpyt

// @desc encrypt a value

// @access Public

app.post("/hash", (req, res) => {

payload = {};

bcrypt.genSalt(10, (err, salt) => {

bcrypt.hash(req.body.value, salt, (err, hash) => {

if (err) throw err;

payload.value = hash;

res.json(payload);

});

});

});

Not a real use case – we wouldn’t want to ever return the password to the front end, encrypted or otherwise!

**Gensalt() –** generates the ‘salt’ for bcrypt to use:

* 10 represents the number of ‘rounds’ to generate the salt, more rounds = more secure against brute force attacks, but also more complex to encrypt, 10 is default and normally sufficient.
* Randomly generated using a inbuild **Crypto** module.

**Hash() –** generates a hash for the input using the salt previously generated.

// @route POST decrypt

// @desc check a value

// @access Public

app.post("/check", (req, res) => {

errors = {};  
 const value = req.body.value;

const hashedValue = req.body.hashedValue;

//Check Value

bcrypt.compare(value, hashedValue).then(isMatch => {

if (isMatch) {

res.json({ message: "Success" });

} else {

errors.value = "Incorrect";

return res.status(400).json(errors);

}

});

});

Not a real use case – the ‘hash’ would be stored on the Database and we would just send the plain text (value).

**compare() –** compares the 2 parameters that are passed to it, returns a promise.