Assignment 1 | **Tyler Wilding**
COSC 4426 | Due Date: 18/09/16
Prof. Biocchi | Source Code
TA: – | Live Demo

Web-App Description

The web-app I made is relatively simple, it was developed (almost) entirely using the Elm Language which has amazing performance compared to other similar tools like Ember, React, or Angular. It is essentially MVC within a single file.

It allows you to enter some plaintext and a passphrase which are then used to create all the hashes for the encryption algorithms I chose. If you leave the fields blank nothing will be added. And lastly, it uses local storage (HTML5 this time, not IndexedDB) to store the objects, which can also be deleted.

Encryption Methods

In my assignment I made use of the CryptoJS library which supports: AES, DES, Triple DES, Rabbit, and RC4 encryption, among others, but those are the ones I chose to utilize and they are the ones I will explain.

AES

The Advanced Encryption Standard (AES), originally known as Rijndael, is the current specification for encryption by the National Institute of Standards and Technology, superseeding the previous encryption method Data Encryption Standard (DES). AES now used world-wide and is a symmetric-key algorithm which means that the same key is used for encryption and decryption.

Without getting too much into detail of how AES works, AES involves 4 main steps: Key Expansions, InitialRound, Rounds, Final Round.

Key Expansions

In this stage of encryption, the Rijndael's key schedule is used to determine the round keys to be eventually used by using the cipher key. The key schdule utilizes many operations, including XOR, a 16-bit 2D grid referred to as Rijndael's S-box, as well as Rcon which is the exponentiation of 2 to a user-specific values using Rijndael's finite field, among other steps.

$$rcon(i) = b^{i-1} mod(x^8 + x^4 + x^3 + x + 1)$$

Because AES at a minimum requires 128bits, only 11 round keys are required, for larger forms of encryption more round keys are required. Once these round keys are determined we can move onto the next step.

InitialRound

Now each byte of the 4x4 column-major-order matrix (called the state) is combined with the respective byte of the block of the round key using an exclusive or operation. This is referred to as an AddRoundKey operation.

Rounds

This step requires 4 operations. The first is called SubBytes, we substitute bytes in the state according to a seperate lookup table. This step is required to provide the non-linearity of the cipher, avoiding simple algebraic attacks. The next operation is called ShiftRows, the last three rows of the state are rotated to the left a certain amount of times, each row is not rotated the same amount of times. Thirdly, we perform MixColumns, each column of the state is multiplied with a fixed polynominal amount using exclusive or. And finally we perform an AddRoundKey operation again.

Final Round

This step is the same as the previous except we do not perform an AddRoundKey operation.

Final Word on AES

AES is the current standard for encryption so it should definitely be the one used, however even though AES is still not practical to crack, it should not be used in it's original specification. Modern use of AES utilizes an Initialization Vector (and usually a Salt), where the output is fed through AES again several times. This is called Cipher-block Chaining. A nicer explanation to AES can be found here Estimates put cracking an AES-128 key at around 2^{126} complexity

DES

Data Encryption Standard, first designed by IBM in 1975 and derived from the older Lucifer cipher, is now superseeded by AES and has been withdrawn from NIST due to it's vulnerabilities. It is vulnerable to a bruteforce and DES keys have been cracked within a day by distributed.net and the EFF. As well as it's theoretical cracking complexity is many several orders of magnitude smaller than AES's at 2^{43}

To keep things brief, DES involves multiple steps. First 16 subkeys are created using an 8 byte key, the results of which are shifted, then permuted over a random bit sequence array. Next the message can be encrypted at a rate of 8 characters at a time using the permutation sequence again. After several XOR operations and shifts the ciphertext is generated.For an in-depth look at DES encryption click here

Final Word on DES

Clearly, DES should not be used anymore because it has been shown to be broken quite easily, not data encrypted in DES is truly secure as a result. However one other thing that should be obvious is how slow DES is. By comparison to AES, the steps are much less involved and can be performed in parallel, DES however is much more difficult to do so with, so it has two problems going against it.

Triple DES

The original solution to DES being vulnerable, it is essentially doing the same steps in DES but doing it three times, this means that the key is 24 bytes (three times as large). Triple DES will encrypt and decrypt the data three times before being added to the result. While Triple DES does solve the problem of brute-force attacks, it creates a new problem of the extremely slow nature of the algorithm, and was the primary motivator for developing other encryption methods like AES, and should not be used either.

Rabbit

A relatively new encryption cipher, open-sourced back in 2008, it uses a 128-bit key and 64-bit initialization vector. Rabbit is a very fast an compact cipher in hardware, encrypting 128 bits per iteration. All operations are basic arithmetical operations, no lookup tables or S-boxes like in AES are required.

The security of Rabbit is actually very good, there is a very small bias in Rabbit's output that results in a 2^{158} complexity. Due to it's ease of use and speed, I would say that Rabbit is a very good option and may even be better than the AES standard.

RC4

RC4 is another stream cipher like Rabbit, known for its simplicity and speed. However there have been multiple vulnerabilities discovered and is considered insecure. Some of these vulnerabilities are when the output of the keystream is not discarded, or the key is not random, or if many similar keys are used together. The insecurity of RC4 has led to insecurity of other protocols that use it, such as the WiFi protocol WEP.

RC4 consists of two parts, Key Scheduling, and Pseudo Random Byte Generation. Key scheduling involves taking an array of 256 elements of 0-255, and using a secret key to randomly re-arrange the array. Next, the Pseudo Random Byte Generation stage will further distort this array. Because a secret key is used for every single element, as long as the key is not known the result can also not be known. However, this is also its main downfall, not everything is truly random and if portions of the output can give a strong indication of the true secret key.

The algorithm is so easy that it can be shown in under 20 lines of recognizable code:

```
KSA:
for(i=0; i<256; i++) S[i] := i; // initialize the array
j := 0
for(i=0; i<256; i++){
    j := (j + S[i] + key[i mod keylength]) mod 256;
   swap(S[i], S[j]);
}

PRGA:
i := 0; j := 0;
do{
   i := (i + 1) mod 256;
   j := (j + S[i]) mod 256;
  swap(S[i], S[j]);
 K := S[(S[i] + S[j]) mod 256];
     output K;
}while(required);
```

Summary

It is important to note that when looking at the complexity of breaking an encryption method that one looks at the key size. If the keysize for an encryption method is 128 bits, than this means that the brute force complexity for the key is 2^{128}, meaning that as long as the complexity to break the algorithm exceeds this, it is not a security threat.

For this reason, AES or Rabbit would be perfectly fine to choose as an encryption method, however the other algorithms that can be cracked in less than brute-force time, actually make it easier to crack the passwords than brute forcing the password, assuming the password was stored as 128bits. Definitely algorithms like DES or especially RC4 should not be used, but Triple DES is a gray area, there are definitely better encryption methods but using Triple DES would probably not compromise your security. In fact, many companies such as Mozilla and Microsoft and outright prohibited using it.

References

[1] S. Sanadhya. (2014). *Cryptography: What is an intuitive explanation of the RC4 encryption algorithm and its weaknesses?* [Online]. https://www.quora.com/Cryptography-What-is-an-intuitive-explanation-of-the-RC4-encryption-algorithm-and-its-weaknesses?srid=Gp4A

[2] *DES Explanation* [Online]. http://www.tero.co.uk/des/explain.php

[3] *Data Encryption Standard* Online. https://en.wikipedia.org/wiki/Data_Encryption_Standard

[4] *Advanced Encryption Standard* [Online]. https://en.wikipedia.org/wiki/Advanced_Encrypt ion_Standard

[5] *Rijndael key schedule* [Online]. https://en.wikipedia.org/wiki/Rijndael_key_schedule

[6] *RC4* [Online]. https://en.wikipedia.org/wiki/RC4