Challenges in Cryptography

Tristan Erney

September 30, 2021

Contents

Abstract

In this document is the accumulation of all the knowledge I have obtained from doing the exercises and challenges provided by the Cryptopals and Mystery Twister Cryptography Challenge. This document will be divided into the sections in which I took for each of the challenges and my thoughts and experiences while completing them. All of these notes have been collected in real time to create an air of legitimacy. I'm not an expert in the realm of cryptography, but I will give this my best shot and feel that this will be a good challenge.

Expectations

While I am taking the direction of my studies towards Computer Security Engineer, I am an avid programmer and love taking up different challenging projects to learn new and clever techniques. My interests lye in systems level programming with the C/C++ languages with the former being my go to. For the purposes of these challenges I will be using the C programming language. While many would argue that C++ would be a better choice due to it being more modern, I prefer the simplicity that C offers to the programmer.

To begin preparing for this project, I'm taking notes on the different topics which I will need to know in order to complete these challenges. At the end, I should have an extensive knowledge bank of information to pull from to help create implementations for each challenge.

Set One

First Challenge

For the first challenge we are to create a conversion from hexadecimal string to base64. At first I thought to look up libraries in order to do this task. Had I done this, I would have been done this task in a few hours. Instead, to get the full experience and know how of hexadecimal and base64, I decided to roll my own conversion header which has been placed in the file conversions.h. The key to converting between different formats like hexadecimal and base64 is to have an intermediate value to convert between. For this, we can transform each of the formats to a byte string. In our conversions.h header file we have a type declared just for this use. Along with our new type we have a couple of dedicated functions which allow us to create and manipulate our byte strings. This will be the key for doing arithmetic and conversions throughout this challenge.

Inside of conversions.h, we have a bunch of different declarations. Most of them are functions, but a few of them are just static instances of tables or constants which help during the course of converting formats, most of them for base64. We have an encoding and decoding table for base64, which is nothing more than a string of possible values for encoding and a table of what each value represents between ASCII code 0 - 127. Additionally, we have our padding symbol as well. To be honest, I don't use it at all but it's more of a place holder to allow the user to realize what that symbol means if they have never looked at base64 before. It certainly helped me. Other than that is a handful of functions we use fairly regularly in this project.

Writing functions for the hexadecimal transformations was the least complicated part section of the first set. Although there was one hiccup when transforming the hexadecimal to bytes initially, but that problem was quickly resolved. Where the real growing pains came from was implementing a base64 conversion on my own.

References

Base64: https://www.base64encoder.io/learn/
Base64: https://datatracker.ietf.org/doc/html/rfc4648
PKCS#7: https://datatracker.ietf.org/doc/html/rfc2315