Hand-in 2

antni

October 2022

1 Introduction

For this assignment we were asked to design and implement a protocol that would allow Alice and Bob to do a 6 sided die throw over an insecure network while not trusting each other and without allowing an adversary to see that they are playing dice.

2 Coin Tossing Protocol

To solve Alice and Bob's trust problems I have chosen to use the 'Coin Tossing' protocol by Blum. This allows Alice and Bob to generate a random bit without being able to influence or predict the outcome.

The protocol works as follows:

1. Alice chooses a random bit 'k' and sends a commitment Com(k,r) to Bob where 'r' is a series of random bits appended to 'k'.
2. Bob chooses a random bit 'g' and sends it to Alice.
3. Alice sends (k,r) to Bob, who checks that it hashes to the same commitment as the one she received from Bob.
4. Both Alice and Bob compute the output as $k \oplus g$.

In my implementation I have chosen to send a sample of 3 random bits. I could have chosen a higher sample to make the results more uniform since there's a higher chance of rolling 1 and 2 in my implementation because I apply $\pmod 6 + 1$ to the result.

2.1 Hashing

The commitment used in the 'Coin Tossing' protocol is the result of hashing 'k' appended with 'r'. For this I have chosen the SHA256 hash algorithm which is believed to be secure.

3 TLS

To provides confidentiality, data integrity and authenticity I am using the TLS protocol. The version of the protocol is automatically chosen based on what the client and server supports.

Confidentiality is provided by using one of the encryption algorithms given in the specification for the TLS versions. TLS uses a MAC(Message Authentication Code) to provide data integrity by signing and verifying a message using the same key.

3.1 Certificate

For authenticity I have generated a X.509v3 certificate that uses SHA256 and 4096 bit RSA encryption. This prevents man in the middle attack by verifying the identity of the sender.