Om Khadka, U51801771
Sources: Cryptographic hash function properties
Collabs N/A

Challenge 4

The hack I did exploited a flawed custom hash function that was implemented as a replacement for MD5. The original programmers essentially made a flawed cryptographic hashing function.

ID'ing the hash's problem

The hash function did the following:

- Truncates input to first 48 bits (6 bytes)
- XORs each byte with 0xA5
- Stores the result as a hexadecimal string

This can easily be reversed, completely missing the point of hashes being pre-image resistant.

Attack

Everything was done under a script. The script does the following:

1. Used SQL injection on the /search endpoint.
    a. " UNION SELECT password_hash FROM members WHERE username='admin'--
2. Store the **admin_password_hash** from that resulting query.
3. Reverse **admin_password_hash** to its original string
    a. Firstly, hex string → bytes
    b. Then, XOR the bytes with/ 0xA5
    c. Finally, bytes → string
        i. This is now the **true_password**
4. Log in fromthe main endpoint with the following form
    username : admin
    name : om khadka
    password : **true_password**

```
omimahomie@LAPTOP-CEUFRM7P:~/cs357/loginLab$ python3 ch4.py
Admin pswd hash: f6e3e2efe9e8
Hash as actual pswd: SFGJLM
Logged in as admin.
Completion Hash: 33220d7624e6a3cdf23439b1cc65161232473fc73f6825a52553f115b89a478f

================================================
Pswd: SFGJLM
Completion Hash: 33220d7624e6a3cdf23439b1cc65161232473fc73f6825a52553f115b89a478f
```

Hash: 33220d7624e6a3cdf23439b1cc65161232473fc73f6825a52553f115b89a478f