VotD: Brute Force Logins

Attack Pattern: See **CAPEC-49** (https://capec.mitre.org/data/definitions/49.html) and it's child the Dictionary-based Password Attack (**CAPEC-16** (https://capec.mitre.org/data/definitions/16.html))

Weakness/Vulnerability: There are several weaknesses that make Brute Forcing a useful attack:

- **CWE-307** (https://cwe.mitre.org/data/definitions/307.html) : Improper Restriction of Excessive Authentication Attempts (most applicable to webapps like DVWA)
- **CWE-521** (https://cwe.mitre.org/data/definitions/521.html) : Weak Password Requirements
- **CWE-259** (https://cwe.mitre.org/data/definitions/259.html) : Use of Hard-coded Password

Description: An attacker repeatedly attempts to log in to a user's account by "brute force" guessing login credentials. This requires guessing:

1. **user name:** 'admin' and 'administrator' are popular to try. Usernames can be inferred from email addresses or black market lists.
2. **password**: The most common form of attack is to use a "dictionary" of passwords, e.g., the RockYou list (be mindful of the sites you download any such lists from).

Login attempts such as this are easy to automate with libraries like MechanicalSoup.

Another type of brute forcing deals with password hashes. Most user's passwords are stored in hashed values in a database, and data breaches often involve the theft of such databases. When hacker's have the hashed password, they can attempt to brute force the hashing algorithm (try multiple passwords and check if the hash matches). In theory, a good hash algorithm makes this intractable. However, so-called "rainbow tables" of pre-computed hash values can be used to quickly look-up the hash values of common passwords.

DVWA and Fuzzer:

- Do not hardcode passwords, such as database credentials, in your code. Place them in external configuration files that are loaded and discarded as quickly as possible from memory.
- You already know one username and password combo: 'admin'/'password'
- There are four other usernames: gordonb, 1337, pablo, and smithy. Try to automate the guessing of their passwords with the help of a password dictionary like the **RockYou** (https://uncw.instructure.com/courses/16302/files/485894/download?wrap=1) list.

Mitigations:

- Do not hardcode Generally speaking, avoid entirely or be careful using these generic "run in the OS" calls. Usually, APIs exist for specific OS calls that can accomplish the same thing without the danger of injection. For example, in Python, **os.listdir()** (https://docs.python.org/3/library/os.html#os.listdir) can give you a list of all files and directories.

- Enforce strong password requirements: a minimum of 8 characters (preferably 12), lowercase and uppercase, one number, one symbol.
- Limit the number of login attempts before locking the account. The downside, of course, is that honest users can forget their passwords, and malicious users can lock out honest users thus denying them service for a time.