ECE 8813 -ACS

Date: 9/29/2025

Team Members: Shuyi Li, Raj Patel

Approach:

Attack Tree: Targeted Password Guessing
├── **1. Data**
│   ├── 1.1 Collect user:password dumps such as 51k_training_set.txt in this case
│
│
├── **2. Preprocessing & Filtering**
│   ├── 2.1 Parse raw dumps (user:password)
│   ├── 2.2 Deduplicate → build per-user password sets
│   ├── 2.3 Select candidate users (e.g., users with ≥2 passwords)
│   └── 2.4 Filter entries (ASCII, length >=4, length <=32)
│
├── **3. Rule Analysis & Generation**
│   ├── 3.1 Evaluate rule usage counts within filtered per-user password sets
│   ├── 3.2 Prioritize rules in rule file
│
│
├── **4. Guess Generation**
│   ├── 4.1 Inject domain-specific common password guesses
│   ├── 4.2 Apply rules to existing user passwords to form new ones
│   └── 4.3 Special-case injections (weird sets)
│
├── **5. Output**
│   ├── 5.1 Prioritize users (users with digits in their passwords, shorter passwords)
│   ├── 5.2 BFS round-robin write to output to keep correct guessing index low

The dataset required to run this notebook is in the zip file. If you need to change the input dataset to the notebook, please change the path in the part shown below:

```
assert RAN_FIRST_CELL, "Please run the first cell before running this cell"
from tqdm.auto import tqdm

path = "./51k_training_set.txt" # CHANGE PATH AS NEEDED

stage_info = [] # format (stage_name, num_users, num_passwords)

# Read in the data, handling impoperly formatted data
```

Run the following command on your system before using the notebook: **pip install pandas tqdm rapidfuzz plotly**

All required files such as datasets, rule files, python scripts are in the zip file. Output will be written to **final_output.txt**. All cells of the python notebook must be run sequentially to reproduce our approach.

** Copilot was used in this assignment for automatic code completion and correcting syntax. ChatGPT was used in this assignment to manually parse best_64_unparsed.rule file to obtain a rule file compatible with hashcat. The parsed file is called best64_sorted.rule.