Might I Get Pwned: A Second Generation Compromised Credential Checking Service

Bijeeta Pal, *Cornell University;* Mazharul Islam, *University of Wisconsin–Madison;* Marina Sanusi Bohuk, *Cornell University;* Nick Sullivan, Luke Valenta, Tara Whalen, and Christopher Wood, *Cloudflare;* Thomas Ristenpart, *Cornell Tech;* Rahul Chatterjee, *University of Wisconsin–Madison*

https://www.usenix.org/conference/usenixsecurity22/presentation/pal

This artifact appendix is included in the Artifact Appendices to the Proceedings of the 31st USENIX Security Symposium and appends to the paper of the same name that appears in the Proceedings of the 31st USENIX Security Symposium.

August 10–12, 2022 • Boston, MA, USA

978-1-939133-31-1

A Artifact Appendix

A.1 Abstract

MIGP (Might I Get Pwned) is a next-generation password breach altering service to prevent users from picking passwords that are very similar to their prior leaked passwords; such credentials are vulnerable to credential tweaking attacks.

In summary, we are providing guidlines to evaluate the following results.

- [Figure 2]: Our proposed secure protocol for MIGP.
- Security simulation:
  - [Figure 8]: Simulation of attacker's success rate for different query budgets compared to traditional breach-altering service
  - [Figure 9]: Comparison of attack success rate for 'Das-R' and 'wEdit' for different query budgets.
- Performance simulaiton:
  - [Figure 12]: Average latency for different C3 services.
- Similarity simulation
  - [Figure 4]
  - [Figure 5]
  - [Figure 6]

A.2 Artifact check-list (meta-information)

- **Data set:** Since the files required to run the experiments are sensitive password leaks from 2019, if you need access to datasets please write to us. After downloading them, put the downloaded compressed file inside `path_to_MIGP/security_simulation/data_files` folder and then unzip it. For the `models.zip` file download it and put it inside the `similarity_simulation/artifact` folder.

  [Warning]. The zipped file is around 4.25 GB for the data files and 5.84 GB for the model files.
- **Software environment:** We have provided the required packages in `requirement.txt` file. We encourage the reviewers to use 'conda' or 'virtualenv' to create virtual environments and use pip to install them. We have used Python version 3.8.

  Before that, you will need to install the following three software packages.
  - `petlib` from here (For Figure 2, 12). Instructions are already in the link on how to install it.
  - Install argon2-cffi from here Installation - argon2-cffi 21.3.0 documentation.
  - GO (version 1.15) to run the WR19 and WR20 protocols in Figure 12. Make sure `GOPATH` variable points to `'path_to_MIGP_folder/performance_simulation/WR-19-20'`. To install GO version 1.15, we refer to the instructions from this link How To Install Go 1.14 on CentOS 8 | CentOS 7 | ComputingForGeeks. Additionally go the the `'path_to_MIGP_folder/performance_simulation/WR-19-20/src/` folder and run `'go get github.com/willf/bloom'`.
- **Hardware:** Our experiments were run on an Intel Xeon Linux machine with 56 cores and 125 GB of memory. You do not need any special hardware. But some security simulations may need large memory. Please let us know if you encounter such a memory error. We already provide the trained models.
- **Execution/compilation:** We have provided bash scripts to generate the figures. See section A.5.
- **Security, privacy, and ethical concerns:** Please DO NOT share this Google Drive link of the datasets with others as it contains leaked password datasets. Although these leaks are "publicly available", we request the reviewers to do so to safeguard against any problem. We also share the minimum version of the full leaked dataset that is required to evaluate the artifact. Moreover we request to delete the downloaded leaked dataset files after the evaluation is complete from the permanent storage.
- **How much disk space required: approximately** $\leq$ 13 GB

A.3 Description

A.3.1 How to access

Available at `https://github.com/islamazhar/MIGP_python/releases/tag/artifact_eval`. You can either clone or download the zipped source code.

A.3.2 Hardware dependencies

The security simulation for Figure 8 may require large memory. Please let us know if you encounter memory error while running those experiments.

A.3.3 Software dependencies

Already specified in A.2 software environment paragraph.

A.3.4 Data sets

Since the files required to run the experiments are sensitive password leaks from 2019, if you need access to datasets please write to us. We can grant access to datasets after properly reviewing the request.

A.3.5 Models

Already provided in the Google Drive link above.

A.3.6 Security, privacy, and ethical concerns

Already mentioned in A.2 in the "Security, privacy, and ethical concerns" paragraph.

A.4 Installation

Follow the “Software environment" paragraph in A.2.

A.5 Experiment workflow

A.5.1 Figure 2

- Expected time: 2-3 mins after installing the required software
- Required packages: petlib, argon2, all packages in requirements.txt
- Compilation: Go to ‘performance_simulation‘ folder and run the following commands.
  - In one terminal run the server using ‘`python3 MIGP_server.py`‘.
  - In another terminal run the query the server using ‘`python3.8 post_client_MIGP.py -username <username> -password <password>`‘
- How to evaluate: If you issue the following comands the expected outcome will be the following.

```
python3.8 MIGP_client.py --username Alice --
    ↪ password 123456 #will give exact
    ↪ password matching.
python3.8 MIGP_client.py --username Alice --
    ↪ password 123456$ # will give
    ↪ similar password matching
python3.8 MIGP_client.py--username Alice --
    ↪ password deercrossing # or any
    ↪ other password, will give not
    ↪ present in the leak
```

A.5.2 Figure 8

- Expected time: for budget $qc = 10, 100$‘. It takes less time but for $qc = 10^3$ expect 1-2 hours for $n = 10$ and 3-4 hours $n = 100$ depending on the memory and number of threads being run.
- Required packages: all packages in requirements.txt
- Compilation: We simulate the security simulation in three steps.
  - ‘`bash script_step_1.sh`‘. // This will create password variations. You can skip this one as we already provide the variations file inside ‘data_files‘ folder
  - ‘`bash script_step_2.sh`‘. // This create the top 10^3 guess ranks. We have also generated the guess ranks and balls of each password in the ‘data_files‘ folder. [skip if you want]
  - Finally, run "`bash script_step_3.sh <n> <qc>`" to generate the row corresponding to row with value ‘n‘ and ‘qc‘ in Figure 8. The results will be saved in ‘results/security_sumulation.tsv‘ file. This part may long time as for n =100 and $qc = 10^3$ it took us 12 hours to complete the simulation.
- How to evaluate: The results of each run will be saved at ‘results/security_simulation.tsv‘. Run ‘python3.8 Figure_9.py‘ to generate the Figure 8. If some values for ‘n‘ and ‘qc‘ the values has not been generated it will show blank.

A.5.3 Figure 9

- Expected time: 2-3 minutes
- Required packages: None
- Compilation: : Go to ‘`security_simulation/Figure_9` and run ‘python3.8 Figure 9.py‘
- How to evaluate: Inspect the generated ‘Figure_9.jpg‘ it should correspond to the paper presented in the paper (was just drawn using pgfplot for our paper)

A.5.4 Figure 12

We run the experiments on two EC2 instances as mentioned in the paper. But they can be tested on localhost as well. Make sure go (version 1.15) is installed and ‘GOPATH‘ points to path_to_MIGP_folder/performance_simulation/WR-19-20‘.

- Expected time: 1-2 hours. Basically, WR-19 and WR-20 take a lot of time.
- Required packages: specified in ‘requirements.txt‘ file
- Compilation:
  - In one terminal run the servers using ‘bash script_run_server.sh‘ and wait for some time for the servers to finish the precomputation.
  - On another terminal run ‘bash script.sh‘ and you will see the Figure on the terminal.
- How to evaluate: Since it is running on localhost the values may NOT exactly correspond to the values reported in the paper. But should follow a similar trend shown reported on the paper. Such As ‘IDB‘ protocols are the fastest ones. WR-19 and WR-20 are very expensive. MIGP_Hybrid should have low latency.

A.5.5 Figure 4,5,6

- Required packages: Training the Pass2Path models is computationally expensive. Therefore, we train these models in GPU and generated the prediction files for required test_files, to run the experiments fast. The code for training the Pass2Path models is in https://github.com/Bijeeta/credtweak/tree/master/credTweakAttack/. We also stored the sorted list of rules for Das-R and EDR models, ranked based on the breached dataset. Also make sure to install the packages mentioned in ‘requirements.txt‘
- Compilation: Go to “artifact” folder. Download the models.zip and copy the “models” folder here. Go to “artifact/src” and
  - For Fig-4, run "bash fig4.sh"
  - For Fig-5, run "bash fig5.sh"
  - For Fig-6, run "bash fig6.sh"
- How to evaluate: The values should match the ones in the figure.

A.6 Evaluation and expected results

Expected results are already mentioned for each of the Figures in A.5 section.

A.7 Notes

Please contact us via hotcrp if you face any problems or have any questions. Thanks for reviewing our artifact.

A.8 Version

Based on the LaTeX template for Artifact Evaluation V20220119.