Performing a Dictionary Attack Using a High-Powered Server And JavaFX

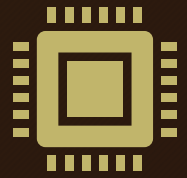
Author: Isaiah Dicristoforo

Advisor: Bill Nicholson, Assistant Professor Educator, IT Program, UC Clermont

Research Question

How can a high-powered server, multithreading, and dictionary attack word lists be used to develop a fast and informative password cracking tool?

Objectives



Create a fast & multithreaded application

Spawn hundreds of threads, deploy application on high powered Linux server, utilize GPU computing power

Provide feedback to the user about cracked passwords

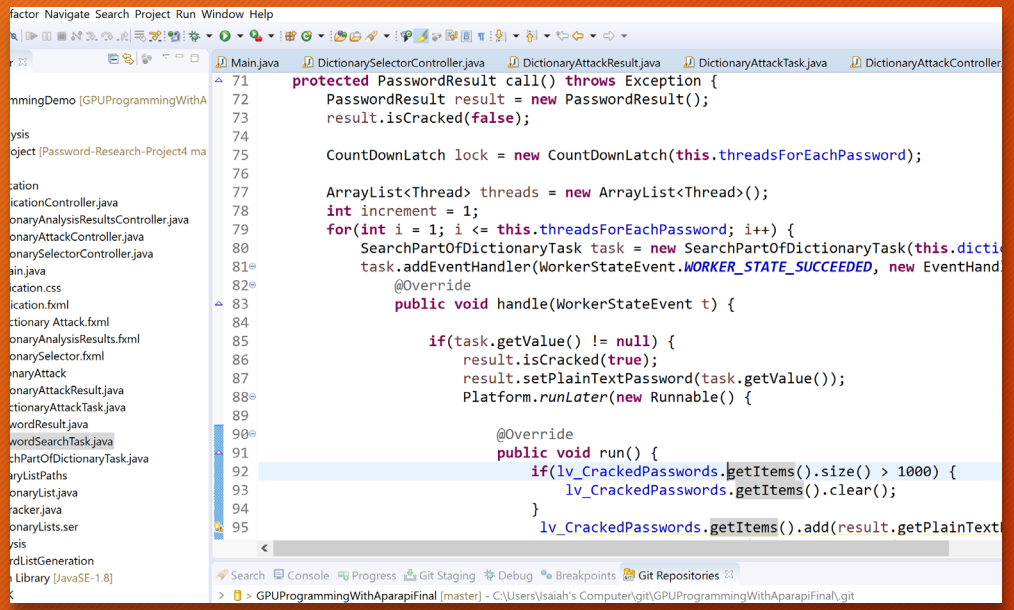
Integrate existing software to analyze password patterns & strength

Design an informative user interface using JavaFX.

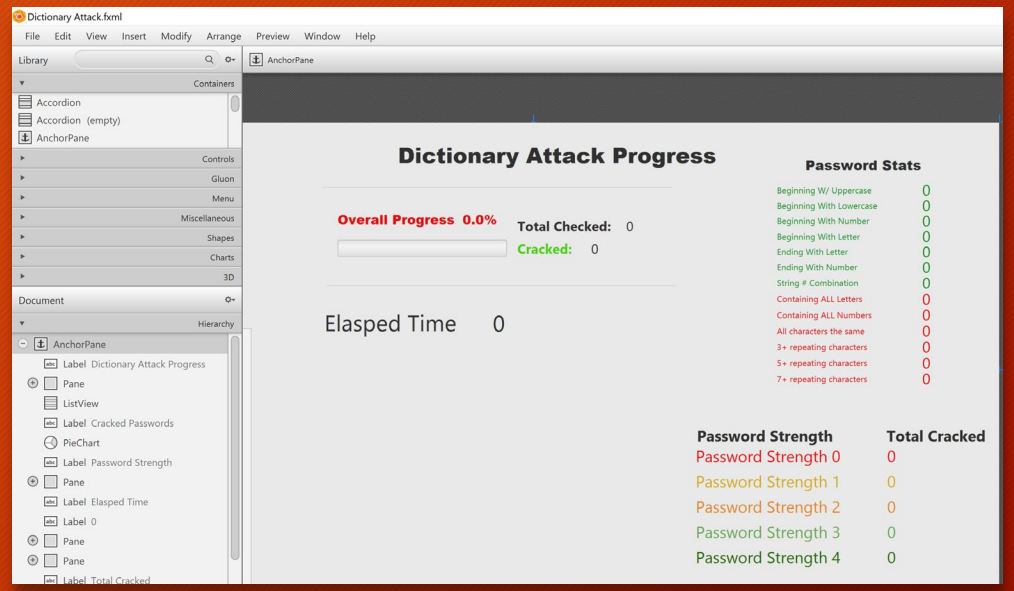
Display charts, program execution time, cracked passwords, etc.

How It Was Made

- JavaFX: Eclipse



- Scene Builder GUI Tool



Dictionary Attack vs. Brute Force Attack

Two different approaches to cracking passwords

Project scope was limited to dictionary attacks.

“Dictionary attacks, with the exception of larger hybrid approaches are much faster than the brute-force method” (Bosnjak et al. 1161).

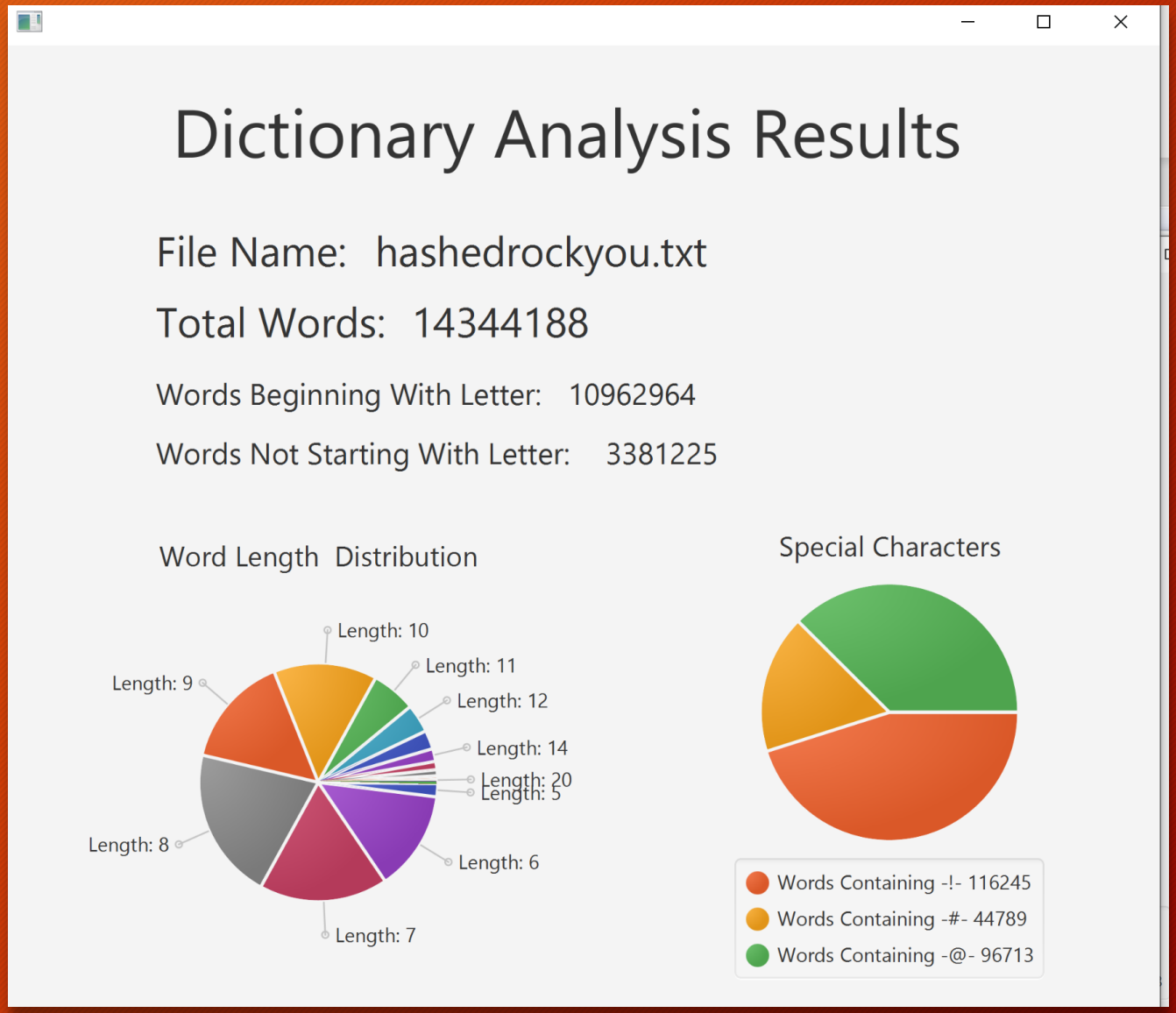
Dictionary Attack Word Lists

- Build a reusable dictionary analysis tool that analyzes any attack dictionary.

Well-Known Dictionary Attack Lists Used

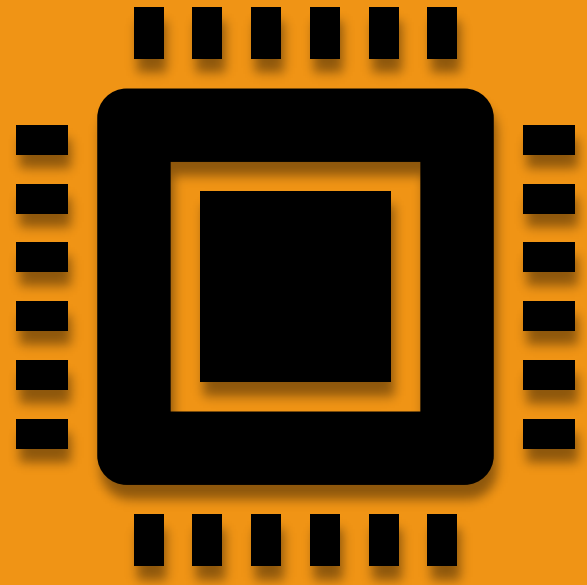
rockyou.txt word list: 14,337,188 words

rocktastic.txt: > 1 billion words



What is Multithreading?

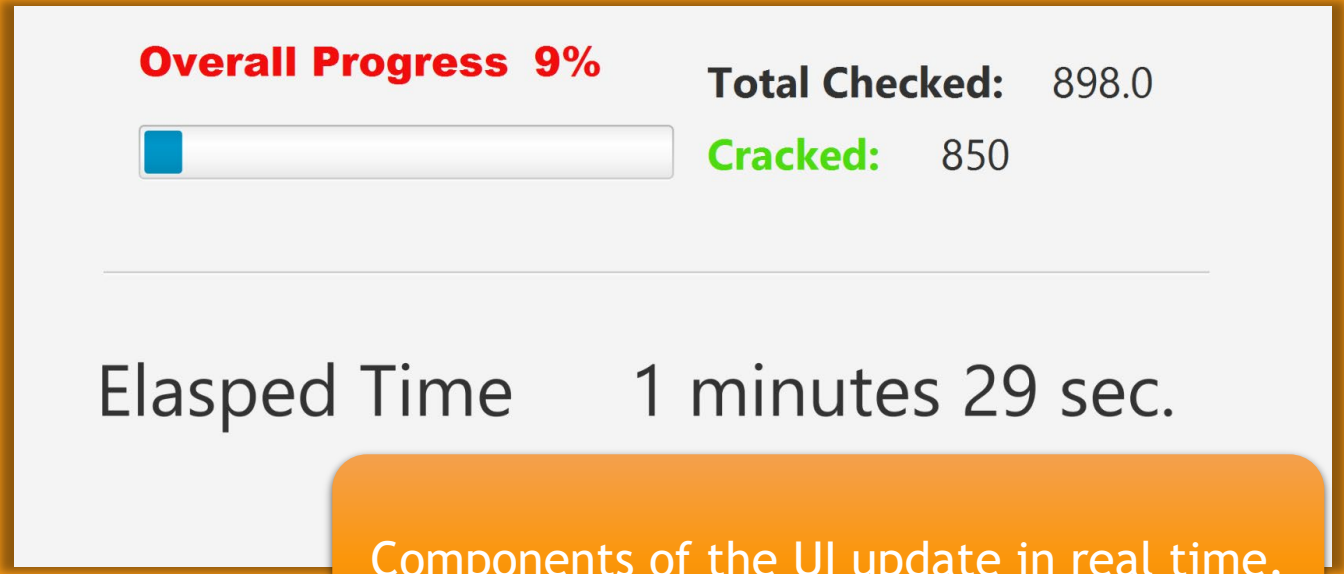
- Multithreading allows you to run different parts of your program at the same time.
  - EX: An operating system can run multiple applications in parallel.
- Today CPUs have multiple cores.
  - Multithreading allows you to utilize the full capacity of a CPU and its cores by allowing you to spawn multiple processes inside them.
- Multithreading was used for this project to crack more than one password at a time, which increases the speed of the program.



Multithreading

- Challenges Faced
  - “The JavaFX scene graph, which represents the graphical user interface of a JavaFX application, is not thread-safe and can only be accessed and modified from the UI thread” – *JavaFX documentation*
  - Data corruption due to race conditions

The UI updates in real time and displays cracked passwords, program duration, overall progress, etc



Components of the UI update in real time.

Running Code On A GPU

Objective: Harness the GPUs on our server to execute GPU code

- The limitations of GPU programming made it infeasible to execute the password cracker on the GPU.
- GPU programming is primarily used for small, mathematical calcuations

Aparapi Framework

- Limited to Java Primitive types
- Used Aparapi to implement a prime number checker that ran on a GPU

Hardware Components

- Laptop: where the application was developed
  - Intel(R) Core(TM) i7-7700HQ CPU @ 2.280GHz
  - 4 CPU cores
  - 16GB RAM
- Linux Server: where the application was deployed
  - Intel(R) Xeon(R) CPU E5-2650 v4 @2.20GHz
  - 28 CPU Cores
  - 26GB Ram

Laptop vs. Server Test

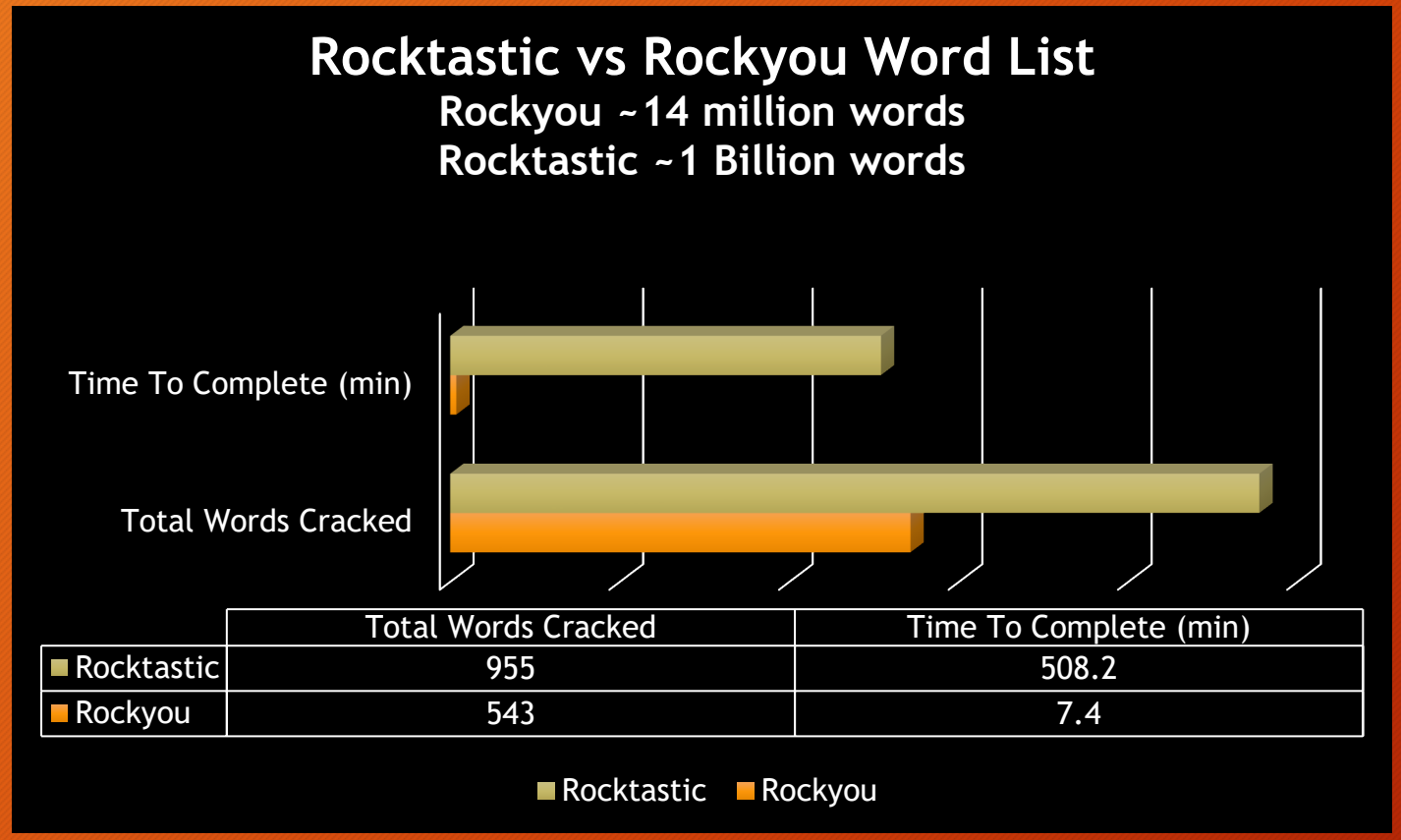
Goal: Compare speed of dictionary attack on each device

- Dictionary attack List: Rockyou (14,344,188 words)
- Password List: *Generated with my password generation tool*
- Total Passwords Generated for cracking: 3500

Laptop	Server
Completion Time: 44 min. 2 seconds **Total Cracked:** 543 (15%) 100 Percent CPU Utilization	**Completion Time:** 7 min 17 sec. **Total Cracked:** 543 (15%) Around 80 Percent CPU Utilization

The server was 83% faster!

Testing Different Word Lists



	Total Words Cracked	Time To Complete (min)
Rocktastic	955	508.2
Rockyou	543	7.4

The larger Rocktastic word list was able to crack 75.87% more passwords than the Rockyou word list.

However…

The test took 6764% longer when using the Rocktastic word list.

The dictionary attack software developed for this project

Password Info

Strength: 1

Estimated Crack Time

less than a second

Patterns Identified

Dictionary

Dictionarys Found In

female_names

Cracked Passwords

bishop1
billie1
florencia
bleach1
darkstar1
narayana
samuel01
popeye1
chennai
55555a
johnlock
400053
myheart
tweety12
montecarlo
caliente
serena1
messiah
dana

Dictionary Attack Progress

Overall Progress 100%

Total Checked: 9715.0

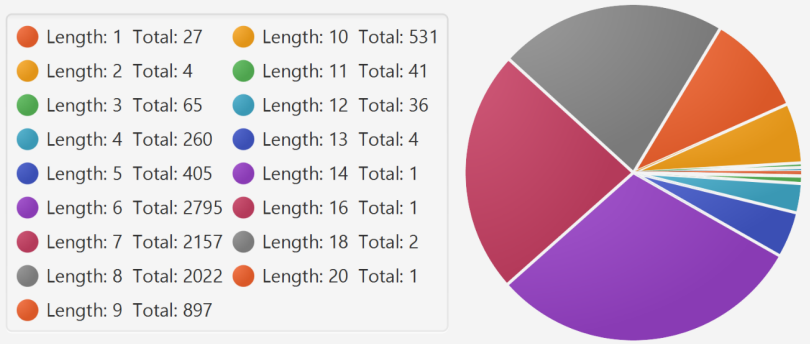
Cracked: 9301

Elasped Time 14 minutes 15 sec.

Password Stats

Beginning W/ Uppercase	51
Beginning With Lowercase	8250
Beginning With Number	988
Beginning With Letter	8301
Ending With Letter	5013
Ending With Number	4245
String # Combination	3481
Containing ALL Letters	4691
Containing ALL Numbers	703
All characters the same	169
3+ repeating characters	249
5+ repeating characters	140
7+ repeating characters	60

Lengths Of Cracked Passwords



Password Strength	Total Cracked
Password Strength 0	2924
Password Strength 1	6277
Password Strength 2	85
Password Strength 3	13
Password Strength 4	2

Conclusion & Takeaways

- The software developed met the original objectives of being *fast* and *informative*.
- Developing a large, multithreaded user-interface can be challenging but deliver a comprehensive result.

- I learned how to utilize the full computing power of CPUs with multithreading
- I learned more about common patterns found in weak passwords.

References

- Bošnjak, L., et al. "Brute-Force and Dictionary Attack on Hashed Real-World Passwords." Brute-Force and Dictionary Attack on Hashed Real-World Passwords - IEEE Conference Publication, 2018, ieeexplore.ieee.org/document/8400211.
- Fedortsova, Irina. “Release: JavaFX 2.1.” Concurrency in JavaFX | JavaFX 2 Tutorials and Documentation, 2 June 2012, docs.oracle.com/javafx/2/threads/jfxpub-threads.htm.

A video demonstration showing how to use the dictionary attack software

- https://www.youtube.com/watch?v=GsvxYYNS3g0&feature=youtu.be