ENEE459B: Reverse Engineering Lab
Project #2: Binary Formats

Scenario

Your boss has come to you with a new problem. He says there was a very **old program** that was used to **track emails for the help desk**. They would store them in **some kind of password-protected database** that was written by an intern many years ago.

The problem is they **need to retrieve data from the database,** and the original source code is gone. To make matters worse, the **passwords used to access the database are gone** as well. All that is known is that **some kind of JSON-based input is used** to fill the database.

“TheBoss” provides you with:

- A copy of the binary (program)
- Their existing database (bin.db)

Assignment

You have been provided with the files ‘**program**’ (executable) and ‘**bin.db**’ (the existing database).

Tasks

Please answer the following questions/perform the following tasks:

1. (**10 pts**) Give a top-level description of how the binary works
    a. What **command line parameters** are used?
       Does the binary operate in **different modes**?
    b. Give an **overall description** of how the binary would be ***used by a user and/or administrator.***
    c. How are users **managed**?
       Can new users and passwords be added?
       Are there any **limits**?
2. (**5 pts**) What are the **authentication** mechanisms?
    a. **Identify the routines**.
    b. What **encryption algorithm** is used for passwords?
    c. Where are users/passwords stored?
3. (**5 pts**) The input to the binary is a simple **JSON format**.
   What variables in the JSON object are **parsed**?
   Show an example that you can **add your own entries** to the database.
4. (**5 pts**) **How** is the help desk **email data stored**?
   It appears to be **encrypted/obfuscated** somehow. What is the **algorithm** used?
5. (**20 pts**) Describe the **overall binary format** of the ‘**bin.db**’ file. ***Be as specific as possible***.
   You should have proper **sizes/offsets/types in your description**.
    - Your description should be ***complete enough*** *to allow someone to write a program* to read all the data. This includes a **description of any obfuscation algorithms used**.
6. (**5 pts**) Find at least **one vulnerability** that provides administrator access to the database.

Turn in

- A **written report detailing all findings** – Be as complete as possible.
  Please use **screenshots** to describe **important code sections**.
  *The code should have **variables** and **functions properly renamed** and **labeled**.*
- A copy of your **annotated Ghidra database**.
- A **dump of the database provided** with some of the **emails extracted**.

Tips

- Stick to your goals.
  - I have intentionally used complicated code that is unnecessary to reverse to reach your goals. Don’t get bogged down in reverse engineering functions that do not lead you to your goals.
- Don’t be afraid to write small C programs (or even Python scripts if you are more comfortable, but the code you turn in MUST be C) to crunch the data you find
- Give me as much information about your thought processes as you can, especially if you are stuck – I can’t give partial credit unless I can see what you are thinking