Main.rs

Overview

This code represents an entry point for a program that performs certain actions based on command-line arguments. It imports and utilizes modules `c2`, `encryption`, and `system` for various functionalities related to interacting with a C2 API, encryption, and system information.

The code relies on the following dependencies:

```
mod c2;
mod encryption;
mod system;
use base64::{engine::general_purpose, Engine as _};
use serde_json::json;
use std::{
    env,
    fs::{read_to_string, File, OpenOptions},
    io::Write,
    process::exit,
};
```

Make sure to add these dependencies to your project's `Cargo.toml` file.

Usage

The code checks the command-line arguments and performs different actions based on the number of arguments.

Case 1: No Arguments

If no arguments are provided, the code checks if a debugger or sandbox environment is detected using the `sandbox` module from the `system` module.

If no debugger or sandbox is detected, the code proceeds with the following steps:

1. Creates an instance of `C2API` from the `c2` module.
2. Retrieves public IP information using the `get_public_ip_info` method of `C2API`.
3. Checks if an error occurred during the retrieval of public IP information. If so, it prints the error message and exits.
4. Retrieves system information such as hostname and username using the `info` module from the `system` module.
5. Constructs a JSON body containing system information and public IP details.
6. Sends a POST request with the JSON body to the C2 API endpoint `/agent/new` using the `post` method of `C2API`.
7. Checks if an error occurred during the POST request. If so, it prints the error message and exits.
8. Creates an agent tag using the received data from the API response.
9. Encodes the agent tag using Base64 encoding.
10. Cleans the received public key and assigns it to `private_public_key`.
11. Performs file and disk encryption using methods from the `encryption` module.
12. Writes a message containing the recovery instructions to a file named `HELP_RECOVER_ALL_MY_FILES.txt`.
13. Deletes shadow copies using the `delete_shadow_copies` method from the `file` module in the `system` module.

Case 2: One Argument

If one argument is provided, the code assumes it is a path to a private key file.

The code performs the following steps:

1. Reads the contents of the private key file.
2. Performs file and disk encryption using methods from the `encryption` module.

Limitations

- The code assumes the usage of the `tokio` runtime for asynchronous operations.
- The code relies on specific modules and their implementations in the `c2`, `encryption`, and `system` files. Ensure these files are present and contain the required functionality.
- The code depends on specific C2 API endpoints and response formats. Modify the code if using a different API or endpoints.

Examples

Example usage of the code:

```
#[tokio::main]
async fn main() {
    // ... Code from the original main function
}
```

Ensure that you have the required dependencies, modules, and files in your project before running the code.

Revision #3
Created 3 July 2023 10:28:26 by Makito
Updated 3 July 2023 10:28:55 by Makito