C2 Security Application

Introduction

This project is a Command and Control (C2) Security Application developed primarily for Windows environments. It allows monitoring and managing endpoint devices by tracking OS information, installed applications, timestamps, and geolocation (if available). The application operates in two parts: the server, which handles device data and interactions, and the client, which runs on the endpoint devices to send information to the server and receive commands for execution.

Prerequisites

Before building or running the application, ensure that the following software and tools are installed: - Operating System: Preferably Windows 10 or higher - Python: Version 3.9 or higher - Flask: Used for the server's web framework. - Requests: For handling HTTP requests between client and server. - PyInstaller: To create a standalone binary executable for the client.

To install the required Python libraries, run the following command in your terminal:

```
pip install -r requirements.txt
```

Building the Software

1.Server

- The server is simply a Python application that can be run directly. It is located in the server/ directory and can be executed as is. To run the server:
- Open your command prompt.
- Navigate to the server/ directory:

```
cd server/
```

- Run the server using the following command:

```
python server.py
```

The server will run by default on http://localhost:5050. If you want to change this, you can modify the port or host settings in the server.py file.

2.Client

- To run the client as a standalone .exe binary (especially on Windows), you'll need to use PyInstaller to bundle it into a single executable file. This allows the client to run on systems without requiring Python to be installed.
- 1. Install PyInstaller:

```
pip install pyinstaller
```

- 2. Bundle the client:

```
pyinstaller --onefile client.py
```

This will generate a single .exe file for Windows under the dist/ directory.

```
dist/client.exe
```

Running the Software

Server

To run the server, navigate to the server/ directory and use the following command:

```
cd server
python server.py
```

The server will start on http://localhost:5050 by default. To access the web interface open your browser and go to that address.

Client

For running the client, ensure that the server is already up and running. You can run the client from the command line by navigating to the client/ directory:

```
cd client
python client.py
```

If you've created a binary (as described above), simply run the .exe file from the dist/ folder.

Key Functionalities

1. Device Registration:

```
-The client collects information such as OS name, version, installed applications, and geolocation.
-This data is sent to the server upon registration, and the device is added to the watchlist.
```

2. Heartbeat Mechanism:

```
-The client regularly sends heartbeats to the server to confirm it is online. If the device is removed from the watchlist, the heartbeat stops, and the device is marked as offline.
```

3. File Operations:

```
-Upload: The client can upload files to the server.
-Download: The client can download files from the server.
```

4. Command Execution:

```
-The server can send commands to the client, which are executed locally on the client’s machine, and the result is returned to the server.
```

5. Interaction History:

```
-The server logs interactions, such as file uploads, downloads, and command execution results. The client can view this history if permitted.
```

6.Watchlist Management:

```
-Devices can be manually added or removed from the watchlist using the server UI. The client will be notified if it is removed.
```

7. Permissions:

```
-Admins can manage view permissions for device information and interaction history through the server UI.
```

License