GPG Telegram Bot for Debian Linux

This project is a Telegram bot that provides GPG encryption capabilities. It allows users to create temporary GPG key pairs, encrypt messages, and send them securely.

Features

- Create temporary GPG keys (valid for 1 day)
- Import public keys from other users
- Encrypt messages using GPG
- Send encrypted messages to a designated recipient
- Decrypt encrypted messages
- List available public keys

Prerequisites

- Debian 12 Linux PC
- Python 3.9+
- GPG installed
- Internet connection

Setup Instructions

1. **Clone or download this repository**
2. **Install dependencies**

   ```
   # Make the installation script executable
   chmod +x install_dependencies.sh

   # Run the installation script
   ./install_dependencies.sh
   ```

3. **Configure the bot tokens**
   - Edit `sendMessage.py` and update `BOT_TOKEN` and `RECIPIENT_CHAT_ID`
   - Edit `telegram_getid.py` and update `BOT_TOKEN`
4. **Get your Telegram Bot Token (if you don't already have one)**
   - Start a chat with @BotFather on Telegram
   - Send `/newbot` and follow the instructions to create a new bot
   - Copy the token provided by BotFather
5. **Get the recipient chat ID**
   - Use the `telegram_getid.py` script to find out your chat ID:

     ```
     # Activate the virtual environment
     source gpg_telegram_env/bin/activate

     # Run the script
     ./telegram_getid.py
     ```

   - Start a chat with your bot and send the `/start` or `/chatid` command
   - The bot will reply with your chat ID. Use this ID as the `RECIPIENT_CHAT_ID` in `sendMessage.py`

Running the Bot

```
# Activate the virtual environment (if not already activated)
source gpg_telegram_env/bin/activate

# Run the main bot
./sendMessage.py
```

Bot Commands

- `/createkey <name> <email>` - Create a temporary GPG key pair (expires in 1 day)
- `/importkey` - Import a public key (attach the key file to this command)
- `/encrypt <fingerprint> <message>` - Encrypt a message and send it to the recipient
- `/decrypt` - Decrypt a message (reply to an encrypted message with this command)
- `/listkeys` - List all available public keys

Directory Structure

```
GPG/
├── cfg/
├── msg/
├── src/
│   ├── sendMessage.py          # Main bot script
│   └── telegram_getid.py       # Utility script to get chat IDs
├── install_dependencies.sh     # Script to install dependencies
└── README.md                   # This file
```

Security Considerations

- This bot stores GPG keys in the default GPG home directory (`~/.gnupg`)
- Messages are forwarded to a single hardcoded recipient
- The bot token is stored in plain text in the script
- Consider using proper key management and secure storage for production use

Troubleshooting

If you encounter issues with the GPG integration:

1. Make sure GPG is properly installed: `gpg --version`
2. Check permissions on the ~/.gnupg directory: `ls -la ~/.gnupg`
3. Try creating a GPG key manually to ensure GPG is working: `gpg --gen-key`