Xây dựng phần mềm sử dụng email hỗ trợ điều khiển quản trị máy tính từ xa

Nhóm 14
Trịnh Quốc Huy - 20120013
Nguyễn Anh Khoa - 20120118
Võ Thị Phước Thảo - 20120191

Ngày 10 tháng 6 năm 2022

Mục lục

1 Lời cảm ơn:

Nhóm 14 chúng em xin được gửi lời cảm ơn đến thầy Đỗ Hoàng Cường, Thầy Lê Hà Minh và Nguyễn Thanh Quân đã hỗ trợ kiến thức nền tảng của môn học để sử dụng trong đồ án. Nhóm cũng xin cảm ơn tư liệu code từ thầy Cường đã giúp đỡ nhóm rất nhiều trong quá trình điều khiển máy tính

2 Thông tin nhóm:

2.1 Thông tin chung:

Nhóm 14 gồm có 3 thành viên:

- Nguyễn Anh Khoa - 20120118
- Trịnh Quốc Huy - 20120013
- Võ Thị Phước Thảo - 20120191

2.2 Bảng phân công:

Công việc	Người thực hiện	Người kiểm tra
Gửi mail	Quốc Huy, Anh Khoa	Phước Thảo
Nhận mail	Quốc Huy, Phước Thảo	Anh Khoa
Các lệnh thực thi	Quốc Huy, Anh Khoa	Phước Thảo
Report	Anh Khoa, Phước Thảo	Quốc Huy
Video	Anh Khoa, Phước Thảo	Quốc Huy
Đồ Họa	Anh Khoa, Quốc Huy	Phước Thảo

Hình 1: Bảng phân công

2.3 Công cụ đồ án:

Nhóm sử dụng ngôn ngữ lập trình Python và thực hiện lập trình trên windown 10, đồ họa được hỗ trợ bởi Tkinter, các thư viên nhóm sử dụng là PIL, numpy, cv2, webbrowser, utils,datetime, os, shutil,uuid,PIL,io,Pickle,psutil,struct,re,winreg,json, email, imaplib, sys, smtplib, logging, pynput.keyboard, threading, keyboard email.mime.(text,image,mutilpart)

3 Giới thiệu:

Chúng ta từng biết đến teamview, một phần mềm giúp người dùng có thể điều khiển một thiết bị A thông qua một thiết bị phụ là B, dựa trên id và password của mật khẩu đó. Vậy thì phần mềm của nhóm em sẽ nâng cấp một chút, chúng

ta chỉ cần sử dụng một tài khoảng google để kết nối và điều khiển giữa các thiết bị với nhau một cách tự động, không cần phải yêu cầu thêm các thao tác trung gian tương tự.

4 Ý tưởng thực thi:

4.1 Danh sách file trong thư mục source code:

- thư mục source chứa hình ảnh cho UI
- UI_login tạo giao diện đăng nhập
- utils chứa cách hàm hỗ trợ cho việc điều khiển(ngoại trừ hàm bắt phím)
- key_log chứa các hàm liên quan đến bắt phím
- sender chứa code liên quan tới người gửi
- reciever chứa code liên quan tới người nhận
- main hàm để thực thi toàn bộ code

4.2 Người gửi:

Để code được gọn gàng và sáng sủa, ta bao hàm code người gửi trong class. Trong hàm khởi tạo, ta cần gán giá trị SMTP_host là 'stmp.gmail.com', gán tên người gửi là username(biến uname), hàm SMTP() nhận vào SMTP_host và trả về object SMTP(hay 1 connection), ta set_debug value cho kết nối vừa tạo, với giá trị True nó sẽ gỡ lỗi cho tất cả tin nhắn nhận và gủi từ máy chủ, còn giá trị 2 hay false thì các thông điệp sẽ kèm theo mốc thời gian. Sau đó, ta tiến hành đăng nhập dựa trên username và password
Ngoài hàm khởi tạo, ta cũng có một hàm để thực thi nhiệm vụ chính, đó là gửi mail. Hàm nhận vào giống cấu trúc của một mail, tên người nhận, tiêu đề(subject) và nội dung mail. Hàm mimetext này trả về 1 đối tượng đặc biệt(có thể xem là đối tượng email), đối tượng này sẽ nhận 2 thông tin người nhận thư và tiêu đề thư từ đầu vào của hàm.
Sau đó ta sẽ tiến hành gửi thư thông qua biến conn(connect), nếu thành công ta hiển thị send success, cuối cùng ta ngắt kết nối, trong trường hợp xảy ra lỗi, ta ném ra ngoại lệ "something wrong"
Trong class này còn có thêm một hàm gửi ảnh, mục đích chính của hàm chính là gửi image đến mail đích. Ta đọc từng dòng của ảnh đó, lưu vào img_data, sau đó tạo một mail object, lưu vào tiêu đề(Subject) và tên người gửi(from). Sau đó thêm đối tượng ảnh và kí tự vào biến msg, sau đó ta gửi toàn bộ đi với msg được chuyển thành chuỗi
Hàm destroy chỉ đơn giản là để ngắt phiên kết nối hiện tại

4.3 Người nhận

Người nhận(reciever) cũng được đặt trong một class, trong hàm khởi tạo, ta tiến hành gán server = 'imap.gmail.com',thông qua hàm imaplib.IMAP4_SSL() tạo lớp thực hiện giao thức IMAP4, nhận vào tham số imaplib.IMAP4. Sau đó, nó tiến hành đăng nhập thông qua username và mật khẩu. Hàm select sẽ thông báo ta sẽ tương tác với thư nằm trong mailbox là "inbox"
Hàm get_mail sẽ lấy ra bức thư mới vừa nhận được, ta tiến hành tìm kiếm trên hộp thư đích, với điều kiện lọc là 'all' tức là lấy toàn bộ thư. Ta lấy bức thư cuối cùng nhờ data[-1].split(), sau đó trích xuất giá trị index của bức thư đó. Thông qua hàm fetch để lấy ra data của bức thư đó. Ta duyệt từng dòng trong data, ta kiểm tra dòng đó có phải tuple hay không, nếu phải ta lấy ra đoạn tin bằng cách chuyển bytes về message. Trích xuất message ta thu được thông tin người gửi và tiêu đề. Ta kiểm tra tin nhắn phải multipart không(tức là có được chia thành nhiều tin nhắn nhỏ hay không) bằng hàm is_multipart(). Nếu đúng như vậy ta duyệt qua từng phần nhỏ rồi tổng hợp lại để được toàn bộ nội dung thư, nếu không phải là multipart, ta chỉ cần tách nó ra từ biến message. Hàm này trả về tiêu đề và nội dung thư

4.4 Chức năng chính:

Đầu tiên, vì mail liên quan đến nhiều chức năng lấy thời gian, nên ta cần phải thực hiện hàm now() để lấy được thời gian gửi mail

4.4.1 MAC address:

MAC address hay còn gọi là "Media Access Control Address". Để lấy được địa chỉ này, ta dùng hàm mac_address() được lưu trong file utils, một cách đơn giản là hàm này sử dụng thư viện uuid để lấy địa chỉ, sau đó chuyển về dạng hexa và trả về. Thông qua hàm send thuộc class sender ở trên, ta gửi trả kết quả đó cho mail đích

4.4.2 capture screen:

Lần này, ta sử dụng hàm capture_screen() để chụp ảnh màn hình hiện tại. Ta sử dụng hàm ImageGrab.grab() từ thư viện PIL, sau đó lưu lại, tiếp theo thông qua hàm image_send ta gửi được ảnh đi đến mail đích.

4.4.3 list app:

Được thực thi nhờ hàm list app, đầu tiên ta tạo 3 list rỗng, sau đó thông qua cmd, dùng proc để lưu lại toàn bộ ứng dụng đang sử dụng, sau đó ta phải tách chúng ra theo từng dòng và loại bỏ đi những dòng trống. Ta chỉ xét lấy thông tin từ dòng thứ 3 trở đi vì 2 dòng đầu chỉ để giới thiệu cho các dòng sau . Tiếp theo ta tạo một array bằng cách tách từng token trong một dòng của tmp, những dòng nào có array[0] là khoảng trắng hoặc rỗng hoặc dòng đó ít hơn 3

phần tử thì ta bỏ qua, khi đó name sẽ là phần tử đầu tiên, còn thread sẽ là phần tử cuối cùng. Sau đó ta tiếp tục lấy ID(phần tử khác 0 kề cuối) và hoàn thành tên của ứng dụng(vì đôi lúc tên của ứng dụng sẽ gồm nhiều token liên tục nên phải thông qua hàm for, for này bắt đầu đi lùi tại vị trí sau khi lấy ID)
Sau khi nhận được kết quả từ hàm, ta format lại rồi dùng hàm send để gửi kết quả đi đến người yêu cầu

4.4.4 Get process:

Hàm get process thì ta chỉ cần thông qua process_iter() để ta có được danh sách các process, cuối cùng thông qua các hàm name(), pid, num_threads ta lấy được thông số cần thiết
Sau khi nhận được kết quả từ hàm, ta format lại rồi dùng hàm send để gửi kết quả đi đến người yêu cầu

4.4.5 Kill:

Ta lấy thông số pid do người dùng nhập vào bằng hàm get_one_content() ở trên
Hàm kill sẽ thông qua pid để tìm được ứng dụng cần tắt. Nếu trả về 0 tức không thành công, trả về 1 nếu tắt thành công. (có thể xài kết hợp get process để lấy pid)
Ta gửi lại người gửi thông điệp thông báo rằng đã kill thành công

4.4.6 Start:

Ta lấy thông số tên ứng dụng do người dùng nhập vào bằng hàm get_one_content() ở trên
Hàm start chỉ đơn giản là khởi động ứng dụng theo tên của nó nhờ hàm os.system
Ta gửi lại người gửi thông điệp thông báo rằng đã start thành công

4.4.7 show tree:

Hàm tree: ta sẽ duyệt từ A đến Z để kiểm tra xem có ổ đĩa tên như vậy trong máy hay không. Nếu có ta thêm vào danh sách trả về
Sau khi nhận được kết quả từ hàm, ta format lại rồi dùng hàm send để gửi kết quả đi đến người yêu cầu

4.4.8 list dir:

Đầu tiên ta sẽ tách lấy đường dẫn mà người dùng nhập vào thông qua hàm get_content_one

Từ đường dẫn đó gọi hàm sendListDir(), hàm này chỉ đơn giản là thông qua hàm listDir của os để liệt kê các file có tại đó. Nếu như không tồn tại đường dẫn nó trả về false và danh sách rỗng, nếu có nó trả về true và danh sách file
Ta nhận các giá trị đó bằng cờ check và danh sách listD nếu như cờ check là false, ta gửi đi "không tồn tại đường dẫn" như vậy, nếu là true, ta format kết quả rồi gửi đi

4.4.9 delete file:

Đầu tiên ta sẽ tách lấy đường dẫn mà người dùng nhập vào thông qua hàm get_content_one
Từ đường dẫn đó gọi hàm delFile(), hàm này chỉ đơn giản là thông qua hàm remove() của os để liệt kê các file có tại đó. Nếu như không tồn tại đường dẫn nó trả về false, nếu có nó trả về true sau khi đã xóa thành công
Ta nhận các giá trị đó bằng cờ check và biến p nếu như cờ check là false, ta gửi đi là "không tồn tại đường dẫn như vậy", nếu là true, ta gửi đi thông điệp đã xóa thành công

4.4.10 copy file:

Ta chia hai hàng đầu vào thành 2 phần tử, sau đó lấy đường dẫn nguồn và đường dẫn đích cho việc copy
Truyền cả hai vào hàm copyFile, hàm này sẽ kiểm tra xem có tồn tại đường dẫn hay không, nếu không trả về false và đường dẫn nguồn, nếu có ta dùng hàm copyfile từ nguồn sang đích, trả về true và đường dẫn đích
Ta nhận các giá trị đó bằng cờ check và biến p nếu như cờ check là false, ta gửi đi là "không tồn tại đường dẫn như vậy", nếu là true, ta gửi đi thông điệp đã copy file thành công

4.4.11 webcam:

Ta dùng hàm webcam để lưu chụp ảnh bằng webcam hiện tại, hàm này sẽ sử dụng videoCapture từ thư viện opencv, sau đó dùng hàm cam.read() và lưu lại thông qua hàm imwrite với tên image.png
Giống với ảnh chụp màn hình, ta dùng hàm send_image để gửi ảnh qua mail

4.4.12 key logger:

Ta cần tạo một đối tượng key control từ đầu cho việc điều khiển bàn phím, việc key logger sẽ được thực thi nhờ hàm key_logger() nằm trong file key_log.py. Hàm này sẽ tạo thêm 1 thread để lắng nghe thao tác phím nhờ hàm listen, trong hàm listen nó sẽ bắt các thao tác nhấn phím từ người dùng thông qua

keylogger. Kết quả của quá trình lắng nghe sẽ lưu trong file outlog.py. Trong lúc chạy hàm key_log() thì ta sẽ gửi thông báo rằng đã bắt đầu bắt phím
Để ngắt quá trình này và gửi kết quả, ta chỉnh flat của đối tượng key control thành false, quá trình bắt phím sẽ ngưng lại, ta đọc kết quả từ file outlog rồi gửi nội dung đi

4.4.13 lock và unlock:

Ta cũng cần phải có sẵn đối tượng key control, sau đó cũng gọi hàm key_logger với string đầu vào bằng "lock" khi đó hàm key_logger sẽ gọi hàm lock và khóa toàn bộ phím cho đến khi ta gửi mail unlocking, hàm key_logger với string đầu vào là "unlock" cũng được gọi và mở khóa các phím

4.4.14 Thiết lập dữ liệu cho registy entry:

Ta lấy dữ liệu theo từng hàng, sau đó trích xuất ra giá trị toàn bộ đường dẫn, giá trị thay đổi, kiểu dữ liệu
Hàm set value sẽ nhờ hàm parse để lấy ra thư mục lớn, đường dẫn đến khóa và name tag cần thiết để chúng ta thay đổi giá trị. Tùy vào mỗi kiểu dữ liệu khác nhau, chúng ta sẽ sử dụng các hàm SetValueEx để thay đổi giá trị, cuối cùng là closeKey. Ta trả về một cờ check để kiểm tra xem đã thay đổi thành công hay chưa, sau đó gửi mail thông báo lại cho người điều khiển

4.4.15 Turn off computer:

Hàm này sẽ gọi hàm TurnOffComputer và sẽ gọi os thực hiện lệnh tắt máy. Lưu ý rằng khi hàm này được gọi, đồng nghĩa với việc ta không thể điều khiển máy tiếp tục được

4.4.16 Restart computer:

Hàm này sẽ gọi hàm Restart và sẽ gọi os thực hiện lệnh khởi động lại máy. Lưu ý rằng khi hàm này được gọi, đồng nghĩa với việc ta không thể điều khiển máy tiếp tục được

4.4.17 quit:

Khi đó, ta chỉ cần kết thúc vòng lặp, chương trình sẽ không bắt mail nữa

4.5 Đồ họa:

Đồ họa được thiết kế nhờ thư viện tkinter của python, gồm có hai class chính

4.5.1 lớp windown:

Lớp có sẵn một cửa sổ Tk() là đối tượng screen, ta tiến hình thiết lập một số thông số cho cửa sổ như kích thước, tên cửa sổ,...Tạo một đối tượng canvas chiếm hết màn hình cửa sổ
Ta viết một hàm destroy() để phá hủy cho canva sau đó đến cửa sổ chính

4.5.2 lớp content mail:

Ban đầu đối tượng nhận vào một parent(tức là lớp ở ngoài), sau đó ta thêm các đối tượng hình ảnh, label và trường input lên trên đó. Cuối cùng, ta thêm hai nút nhấn, nút nhấn đăng ký mở web browser cho việc đăng ký gmail. Nút đăng nhập sẽ yêu cầu ta nhập tài khoản mật khẩu. Khi nhấn nút đăng nhập, từ thông tin nhập vào nó sẽ gọi hàm get_result để lấy thông tin vào, sau đó sử dụng 2 trường thông tin đó tạo hai đối tượng người gửi và người nhận, nếu như sai mật khẩu, một label xuất hiện trên màn hình để thông báo, nếu đúng hai đối tượng được tạo ra và tầm 10s sau, giao diện sẽ biến mất, hàm destroy được gọi và hủy tất cả mọi thứ trên màn hình

4.6 Tiến trình:

Qúa trình đoạn code chạy trong main:

- Tạo đối tượng windown có tên là wd cho giao diện
- Tạo đối tượng content_mail có tên gui_mail
- Gọi hàm mainloop để hiển thị cửa sổ trên màn hình
- Người dùng nhập tên đăng nhập và mật khẩu vào, sau đó nhấn đăng nhập, nếu nhập sai, người dùng nhập lại, nếu nhập đúng, vài giây sau giao diện tắt và phần code dưới bắt đầu chạy
- Tạo một đối tượng key control. Khởi tạo một số đối tượng sử dụng
- Vòng lặp while true bắt đầu, ta lấy ra thời gian gửi mail, nếu 2 mail trùng thời gian gửi với nhau, ta tiến hành bỏ qua mail đó, nếu không, căn cứ vào subject, ta sẽ thực thi những câu lệnh tương ứng

5 Hướng dẫn sử dụng:

5.1 Thiết lập mail:

Hiện tại google đã chặn việc truy cập của các thiết bị kém an toàn, vì vậy ta phải tiến hành thiết lập riêng cho smtp

- Bước 1 - Bật IMAP: Vào gmail -> chọn vào biểu tượng bánh răng ở góc trên bên phải màn hình -> xem tất cả chế độ cài đặt -> chọn tab chuyển tiếp POP/IMAP -> Bật IMAP
- Bước 2 - Tạo xác minh 2 bước: Vào quản lý tài khoản google của bạn -> Bảo mật -> Xác minh hai bước
- Bước 3 - Tạo mật khẩu ứng dụng: Chọn ứng dụng là thư và thiết bị là thiết bị bạn đang sử dụng -> mật khẩu có được là mật khẩu dùng để đăng nhập ứng dụng

Bạn cần làm việc này cho cả mail sender và reciever

5.2 Sử dụng ứng dụng:

Mở file main.py, nhập đầy đủ mail và mật khẩu thu được ở bước trên, cẩn thận nhập đúng. Khi đó ta đã hoàn thành việc kết nối
Bây giờ, bạn chỉ cần đăng nhập vào mail sender và gửi yêu cầu vào mail reciever, máy đang chạy ứng dụng sẽ làm theo điều khiển
Các lệnh có trong chương trình, với subject là tiêu đề mail, content là nội dung mail:

- Subject: mac address. Với thư có tiêu đề như vậy, ta lấy được địa chỉ mac của máy bị điều khiển
- Subject: capture screen. Với thư có tiêu đề như vậy, ta lấy được ảnh chụp màn hình tại thời điểm hiện tại của máy bị điều khiển
- Subject: list dir. Content: path: [đường dẫn] Với tiêu đề như vậy, ta lấy được các file và thư mục con của thư mục hiện tại theo đường dẫn.
- Subject: list app. Với tiêu đề như vậy, ta lấy được danh sách phần mềm đang hoạt động
- Subject: list process. Với tiêu đề như vậy, ta lấy được danh sách tiến trình đang hoạt động
- Subject: kill. Content: pid:[pid của tiến trình]. Ta tắt được tiến trình với pid cho trước
- Subject: start. Content: start:[tên của program]. Ta khởi động một phần mềm
- Subject: show tree. Trả cho người điều khiển toàn bộ cây thư mục có trong file
- Subject: delete file. Content: path:[tên đường dẫn file]. Xóa 1 file bất kì từ xa
- Subject: copy file. Content: path:[tên nguồn](xuống dòng)path:[tên đích]. Copy file từ nguồn sang đích

- Subject: webcam. Chụp hình web cam thời điểm hiện tại rồi gửi cho người điều khiển
- Subject: key start logged. Bắt đầu theo dõi hành trình phím
- Subject: key end. Kết thúc theo dõi hành trình phím và gửi kết quả cho người điều khiển
- Subject: lock keyboard. Vô hiệu hóa bàn phím
- Subject: unlock keyboard. Mở khóa bàn phím
- Subject: update registry. Content: path: [đường dẫn kể từ khóa đầu tiên]<xuống dòng>value: [giá trị mới]<xuống dòng>value_type: [kiểu giá trị]
- Subject: turn off computer. Tắt máy bị điều khiển
- Subject: quit. Thoát phần mềm điều khiển, máy không còn bị kiểm soát
- Subject: restart. Khởi động lại máy

6 Tài liệu tham khảo:

- Chức năng được tham khảo từ code thầy Cường