5 Solution

a. Overview

An overview of solution is shown below. Our solution consists of 3 main components mainly a cloud server, a Raspberry Pi running in the local network and Sync App which runs on the client's Windows computer.

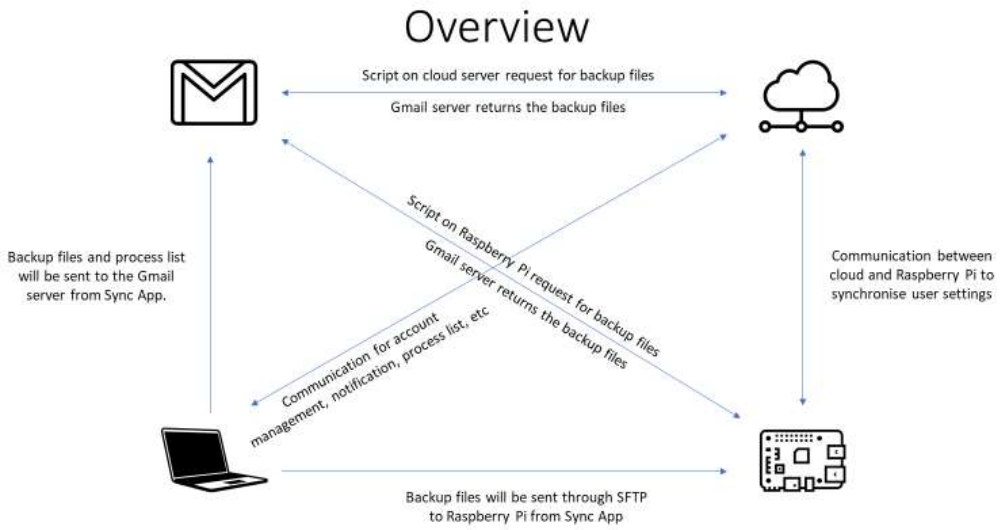

Figure 6: Overview of solution

The Sync App will help the user backup their files automatically using the 3-2-1 backup strategy with file versioning as well as to detect for threats using process and file monitoring against a blacklist and a "Tripwire-Like" detection system.

In the event that a user's computer is infected, the backup will also stop to prevent the infected files from being backed up. The user will be able to login to the WebUI running in the local network or in the cloud to restore his files. Even if the infected files were to be backed up, the file versioning feature will allow users to restore their files to an earlier version which is not infected. Also, our backend is based on Linux, which forms as another line of defence against ransomware. Based on our research, most ransomware only affects one type of operating system.

b. Architecture

We decided to use the 3-2-1 backup strategy as there is a reduced chance of losing files. For example, in the event of a ransomware attack in the local network and the local backup copy was lost, the files can still be restored from the offsite copy. Also, in the event that the tape drive reader is damaged, the files can be restored from the other medium such as the hard drive. Thus, our solution had made use of the 3-2-1 backup strategy as we have 3 copies, one in the user's computer, one in the cloud server and a third copy in the local lab's Raspberry Pi. We also used 2 different mediums as the cloud server would be using hard drives whereas the Raspberry Pi would use a solid-state drive. The cloud server would also serve as an offsite copy.

The Sync App will run on a 64-bit Windows based PC and the cloud WebUI will be running Tomcat8 as well as MySQL database to serve the Sync App, the Raspberry Pi as well as the users. The local WebUI will be running on Django. We chose to use both Tomcat and Django as the Raspberry Pi hardware might not be suitable to run Tomcat.

Gmail is also being used in our backend as email is a reliable way to transfer files. Even if our backend is down, the backups would still be able to continue as they can be stored in Gmail's server. There will also be Python scripts running on both the cloud server and Raspberry Pi to download the files from Gmail to their local storages. All files sent to Gmail will be encrypted twice, first with the user secret which the user specifies during their initial account registration, followed by a secret key that is used throughout our backend that is changed regularly. This would ensure that even if the cloud server or Raspberry Pi were to be compromised, the files will still be unreadable.

All communications in our backend is also done securely. For every set of communication such as creating users on the cloud server, it requires a session key to be retrieved from the cloud. The session key would then be encrypted with our secret key and sent to the cloud before the cloud server would return a response. The protocol of how a user is created in the cloud server from the Sync App is shown below.

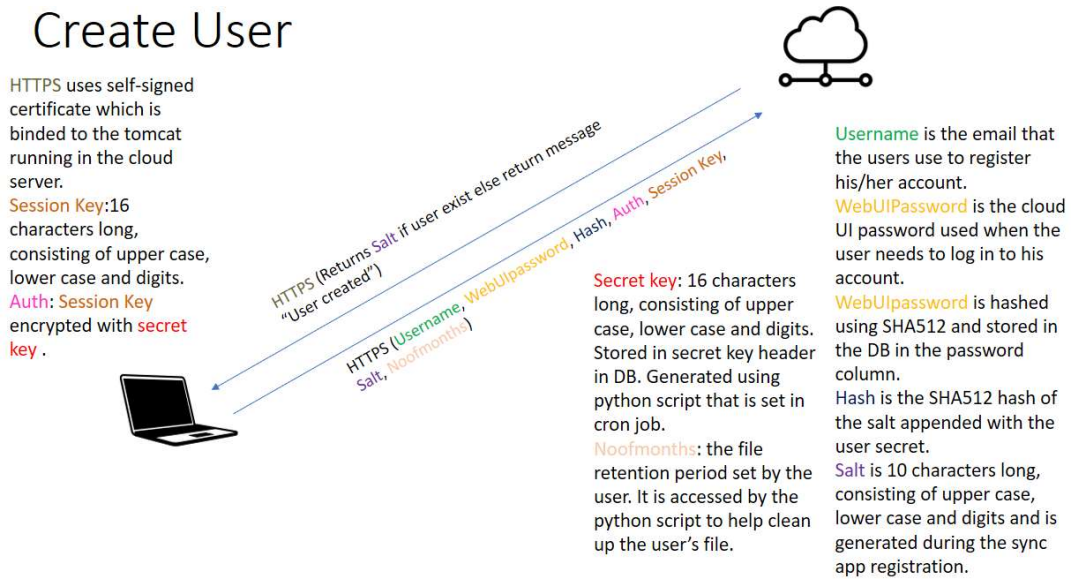

Figure 7: Create user communication process

The request of a new secret key is also done securely. The client would first request for a session key from the cloud server. After that, the client would encrypt the session key with the secret key that it knows. The cloud server would then validate if the secret key used is up to date. It the secret key is not up to date, it would then encrypt the new secret key with the secret key that the client has used and sends it back to the client.

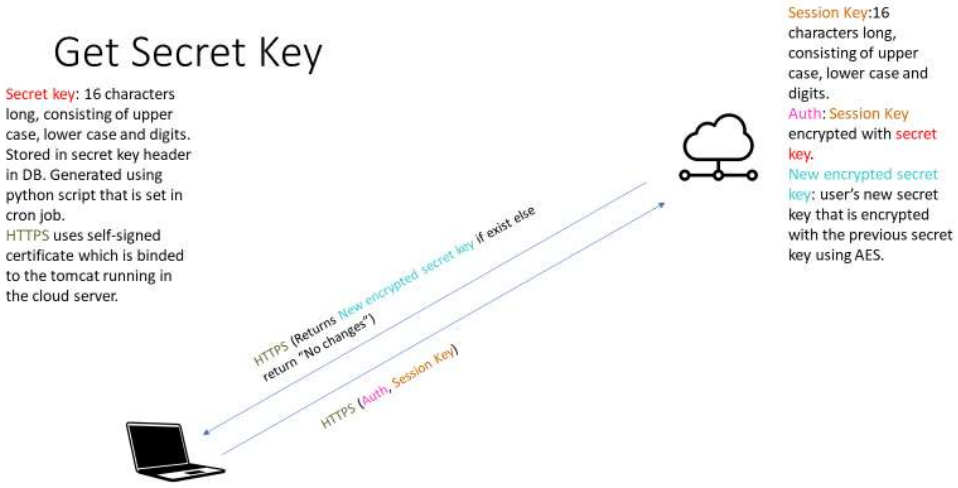

Figure 8: Get secret key communication process

c. Implementation

i. Sync App

The Sync App allows users to select directories, files and which file extensions that they want to backup. It also allows them to pick which files or directories they want to exclude. Users will also be required to pick one local lab that they want to backup their files to. The backups will also support file versioning where the user can restore his file to a previous version. The Sync App would then send the encrypted backups to Gmail as well as to the local lab's Raspberry Pi using Secured File Transfer Protocol (SFTP) if the user is in the local lab.

The user is also allowed to change their preferences such as the backup frequency, local backup directory, backup directory size, file retention period, list of trusted programs, and whether to run Sync when the computers starts. Also, the user can delete all file versions stored in the cloud server and the local lab's Raspberry Pi. When the user has decided to stop using the solution, he can also click delete account to delete his account. The user will also be able to view the amount of storage space used.

A copy of the backup is also stored locally on the user's laptop and users can choose which backup to restore from. The user can also choose to enter a date range of the files and folders to be restored. If the user has downloaded a file or folder from the cloud server or local lab's Raspberry Pi, there is an option in the Sync App for him to decrypt the files.

The Sync App also comes with intrusion detection techniques by doing file and extension monitoring as well as process monitoring. It works by checking for known ransomware file names such as "HELPDECRYPT.txt" and known ransomware extension such as ".epic". For process monitoring, the Sync App would check the processes that are running on the user's computer against a blacklist. If the event a blacklisted process is running, the user will be prompted to shut down his computer.

The user's process will also be sent to our backend to help the administrator decide whether a process is legitimate based on the number of users that have the same process running and whether the user approved or denied the process.

There is also a "Tripwire-Like" detection where a decoy file, '#Donottouch.txt', is placed in the documents folder to help to detect for a ransomware threat. The '#' is used to keep the file at the top of the folder as most ransomware will encrypt the files sequentially. The file is being checked every second and would immediately display a prompt to shut down the computer when the file is being altered or missing. Thus, this would help to detect ransomware at its earlier stages.

Based on research, ransomware attacks would normally encrypt all the files found in the documents folder, thus we think that this is one of the most effective way to detect a ransomware threat.

ii. Cloud WebUI

By using the web application hosted by the cloud server, the user can restore the files from the cloud in the event that the local backup copy in his computer is damaged. The user can login to the WebUI and view the files that he has backed up.

The file explorer navigation style allows the logged in user to easily browse the files and folders that are backed up. The user can also choose to restore a file, folder or restore all his files. When the user has chosen the file to download, he has the option to select the file version that he wants. When the user downloads a folder, he will be able to select a date and time which will be used to sort the version of files in the folder. The restore all feature would allow the user to restore all the files that he had backed up to the server and he is also able to pick a date and time which he wants to restore all his files to.

Users are also able to view the latest ransomware threats and learn more about them in the web application. Recovery of passwords is also available for the user if the user has forgotten his password and he can also login into the cloud WebUI to change his login password.

The cloud WebUI also includes administrative features where the domain expert is able to send announcements to the Sync App users , warn them about potential threats and also give security tips. The domain expert is able to manage ransomware file names and extensions which will be synced to the client's Sync App for detection of ransomware threats. The domain expert will also be able to manage the blacklist and whitelist processes which will also be synced to the client's Sync App for monitoring of application processes.

The domain expert can also view the processes that the users of the Sync App have approved or denied. This list will be compiled so that the domain expert will have a better understanding of what are the trends and choose whether to approve or deny the process.

The domain expert is also able to unlock user accounts that have been locked due to multiple invalid password attempts. By default, if the maximum number of tries for the user account has been reached, the account will be locked, and an email will be sent to the user for the account to be unlocked.

iii. Local lab's WebUI

For the web application that is hosted locally on the Raspberry Pi, it will have most of the user features that the cloud server has. However, it does not have native support for changing password, although there are links for the user to change the password (links will redirect user to the web application on the cloud server). It also does not have the domain expert features.

iv. Python Scripts – Cloud Server

For the files to be downloaded into the cloud server, a python script will be running as a cron job to help to download the emails from Gmail. It will then decrypt the attachments using the secret key and store them in the appropriate folders. It will also help to delete the file versions that are more than 10 days old as well as when the user requested to delete all file versioning. Also, it will help the user to delete his account upon request and helps to count the amount of space that each user has occupied. It will also help to populate the number of users who approved or denied a process.

There is also a separate python script that runs on the cloud server as a cron job which would update the secret key that is being used in our backend.

v. Python Scripts – Local lab's Raspberry Pi

This script is similar to the one running in the Cloud Server, however it does not need to count the amount of space the user occupied as well as to populate the number of users who approved or denied a process. However, it needs to look for files that are only sent to the Raspberry Pi directly and email them out to Gmail.

6 Room for improvements

a. Account lockout

The current solution can be improved by implementing an account lockout feature where the account will be locked up if there are more than 100 file versions in a day as this might be an indication of a suspicious activity.

b. Delete file

The current solution can be improved by implementing a file deletion feature. This allows the user to delete the files in the event that it is accidentally included. In the current solution, if a file is accidentally backed up, the user will have to wait for his file retention period to be up before the file to be automatically deleted from the system.

c. Recovery of user secret

The current solution can be improved by implementing a recovery method for the user secret. In the current solution, the user will be asked to enter his email and user secret when installing the Sync App. Even though the user will not need to use the user secret for backups, he might need it again when he reinstalls the Sync App.

d. Python3 Django

Since the current Django setup is running on Python 2.7, it would be good to use Python 3 for the Django setup so that the call function would not be needed to run the Python 3 script which encrypts the zip file using AES encryption.

e. Sync App could be run as a service

Currently, a user account control (UAC) prompt will be shown every time the user logins to his computer due to the administrative rights that the Sync App requires. The Sync App could be modified to run as a service to prevent the UAC prompt from showing.