# Llaves públicas y privadas

## Introducción

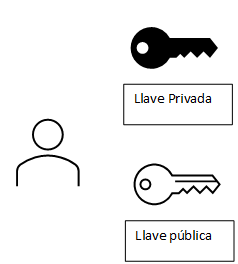
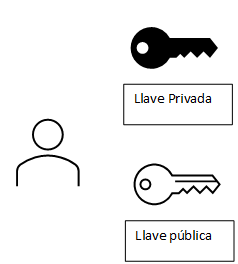
Cifrado asimétrico de un solo camino.

Se vinculan matemáticamente una con la otra, se requiere la llave pública para poder ver lo de la privada.

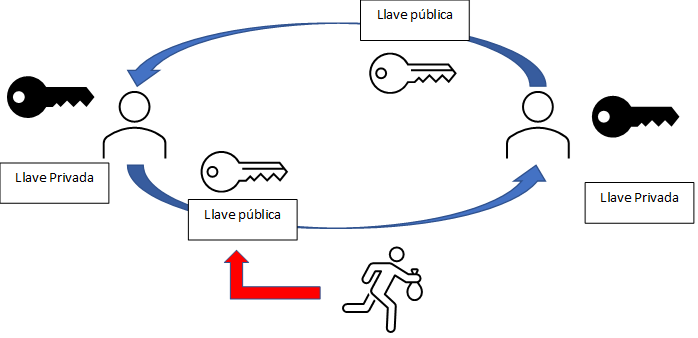
Las llaves públicas y privadas nos ayudan a cifrar y descifrar nuestros archivos de forma que los podamos compartir sin correr el riesgo de que sean interceptados por personas con malas intenciones.

La forma de hacerlo es la siguiente:

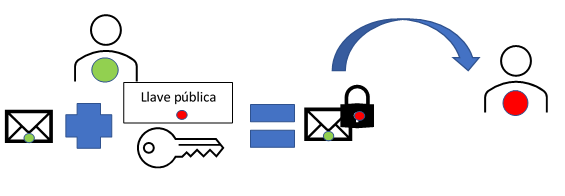
Ambas personas deben crear su llave pública y privada.

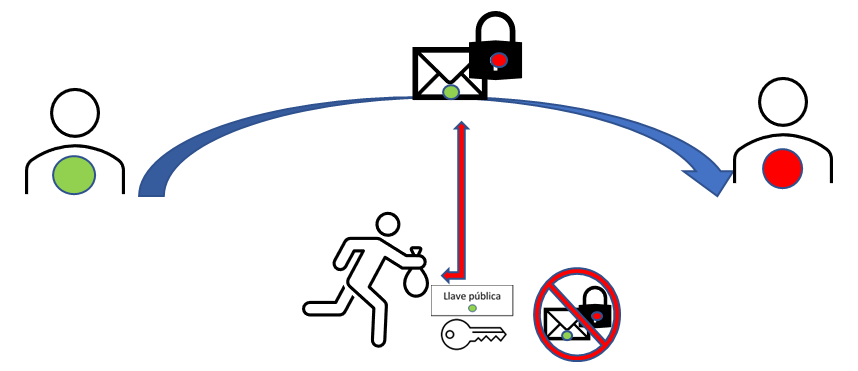


Ambas personas pueden compartir su llave pública a las otras partes (recuerda que esta llave es pública, no hay problema si la “interceptan”).

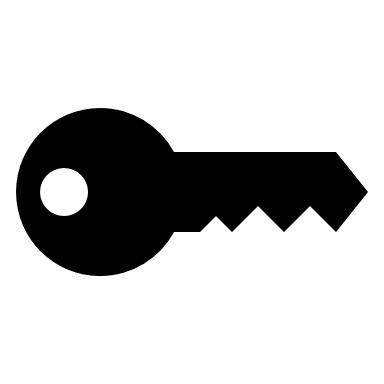
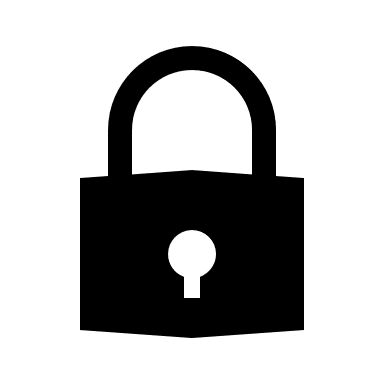


La persona que quiere compartir un mensaje puede usar la llave pública de la otra persona para cifrar los archivos y asegurarse que solo puedan ser descifrados con la llave privada de la persona con la que queremos compartir el mensaje.

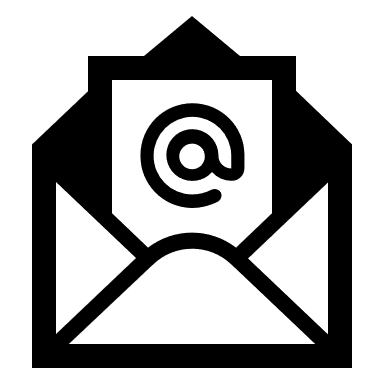
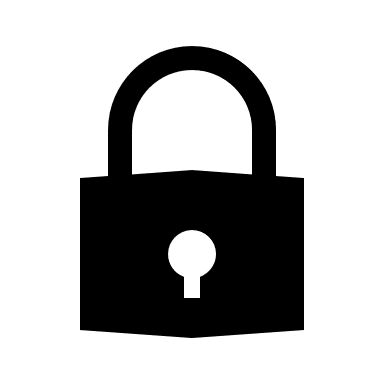


El mensaje está cifrado y puede ser enviado a la otra persona sin problemas en caso de que los archivos sean interceptados.

La persona a la que enviamos el mensaje cifrado puede usar su llave privada para descifrar el mensaje y ver los archivos.



Llave Privada



## Configuración de llave SSH en local

SSH o Secure Shell: Es un protocolo de red que permite acceso remoto seguro a través de una conexión encriptada. Este método de autenticación requiere un passphrase (contraseña) o tambien puede funcionar sin passphrase sobre la clave.

$ git config -l: Muestra la configuración dentro de Git(user y email), Este comando funciona por que Git esta instalado en todo el equipo local.

$ git config --global user.email “nombre\_email\_cambiado” : Se puede utilizar este mismo comando para cambiar el email.

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

ssh-keygen -t rsa -b 4096 -C “[correo@ausar.com](mailto:correo@ausar.com)”

ssh-keygen: Generar llave ssh

-t : especifica el algoritmo para la llave.

Rsa : el algoritmo a utilizar.

-b : Que tan compleja es la llave.

4096: nivel de complejidad solicitado.

-C : correo electrónico al que estará conectada la llave.

Se ejecuta el comando:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Se deja por defecto la ruta mostrada en pantalla, se selecciona contraseña y se escribe la confirmación de esta.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Se realiza la verificaciòn de lo generado: Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Se verifica la ruta en la que se generò :   
Captura de pantalla de computadora

Descripción generada automáticamente

Se requiere adicionar la llave privada a git, para esto se requiere iniciar el agente ssh para poder realizar al validación de estas llaves, utiliza el comando:

eval $(ssh-agent -s)

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Luego se utiliza el símbolo ~ , este es el usado para poder ingresar nuevamente a HOME (donde está la ruta .ssh).

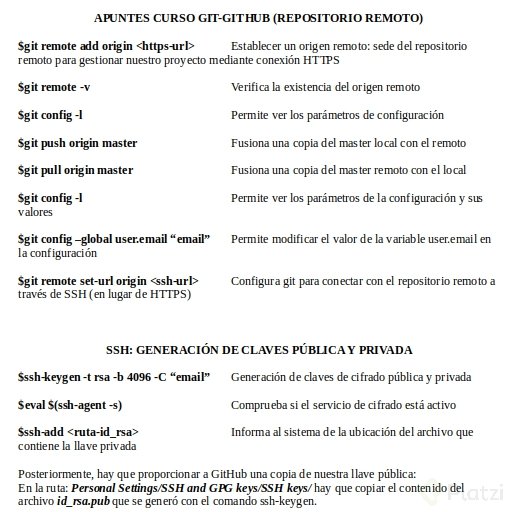
Se debe adicionar ahora la llave privada, por lo que se va a utilizar lo siguiente comando:

ssh-add ruta-donde-guardaste-tu-llave-privada

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

## Conexión a GITHUB con SSH



Para realizar la conexión, lo primero es copiar la llave pública, tener en cuenta que por temas de seguridad, en caso de requerir trabajar en dicho repositorio con diferentes equipos, lo ideal sería que cada equipo tenga su propia llave pública y privada, de esta forma es mucho mas seguro.

Una vez se copia la llave pública, será necesario ingresar al menú de configuración de GITHUB

Captura de pantalla de computadora

Descripción generada automáticamente

Luego de esto buscamos la opción SSH and GPG keys

Captura de pantalla de computadora

Descripción generada automáticamente

Seleccionamos la opción Captura de pantalla de computadora

Descripción generada automáticamente y luego de esto ingresamos la llave pública en el cuadro que se muestra a continuación, para posterior a eso seleccionar “add SSH key”. Captura de pantalla de computadora

Descripción generada automáticamente

esto te solicitará la contraseña, pero una vez ingreses la información correcta, vas a poder observar que se ha adicionado la llave pública sin inconvenientes.

Captura de pantalla de computadora

Descripción generada automáticamente

Ahora podemos actualizar la URL que guardamos en nuestro repositorio remoto, solo que, en vez de guardar la URL con HTTPS, vamos a usar la URL con SSH:

Para esto ingresamos a nuestros repositorios, y en el que vamos a utilizar buscamos la opción de clonar y dentro de esta la opción de ssh.   
Captura de pantalla de computadora

Descripción generada automáticamentePor ultimo usamos el comando:

git remote set-url origin url-ssh-del-repositorio-en-github

## Tags y versiones en Git y GitHub

Los tags o etiquetas nos permiten asignar versiones a los commits con cambios más importantes o significativos de nuestro proyecto.

Comandos para trabajar con etiquetas:

* Crear un nuevo tag y asignarlo a un commit: git tag -a nombre-del-tag id-del-commit.
* Borrar un tag en el repositorio local: git tag -d nombre-del-tag.
* Listar los tags de nuestro repositorio local: git tag o git show-ref --tags.
* Publicar un tag en el repositorio remoto: git push origin --tags.
* Borrar un tag del repositorio remoto: git tag -d nombre-del-tag y git push origin :refs/tags/nombre-del-tag.

Los alias: estos nos sirven para convocar diferentes comandos dentro de la consola, utilizando una única instrucción, para realizarlo debemos tener en cuenta la siguiente nomenclatura:

Alias nombre\_alias=”comandos git”

Ejemplo: alias superlog="git log --all --graph --decorate --oneline".

Interfaz de usuario gráfica

Descripción generada automáticamente

Para crear una etiqueta, se debe hacer :

•git tag -a nombre-del-tag -m “mensaje descripción tag” id-del-commit.

Ejemplo:

$ git tag -a v0.1 -m "Inicio subida de curso GIT" bcba2a5

Para revisar los tags: se requiere utilizar el siguiente comando.   
• git show-ref --tags

Para subir los cambios en los tags, es necesario luego creado los tags, ejecutar el siguiente comando:   
git push origin –tags

Para eliminar los tags que no se van a utilizar se requiere utilizar los siguientes comandos:

$ git tag -d nombre-del-tag

$git push origin :refs/tags/nombre-del-tag

## Manejo de Ramas en Git

Puedes trabajar con ramas que nunca envías a GitHub, así como pueden haber ramas importantes en GitHub que nunca usas en el repositorio local. Lo importante es que aprendas a manejarlas para trabajar profesionalmente.

Crear una rama en el repositorio local:

git branch nombre-de-la-rama o git checkout -b nombre-de-la-rama.

Publicar una rama local al repositorio remoto: git push origin nombre-de-la-rama.

Recuerda que podemos ver gráficamente nuestro entorno y flujo de trabajo local con Git usando el comando gitk.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

## Adicionar colaboradores

Para poder compartir el repositorio y que colaboradores puedan hacer pulls y push a nuestro proyecto, se requiere primero ingresar a github, una vez allá, vamos a buscar el repositorio a compartir, y a seleccionar la opción de “settings”.

Captura de pantalla de computadora

Descripción generada automáticamente

Luego en esta vamos a buscar la opción para adicionar colaboradores, actualmente es “Manage Access”, dentro de esta nueva pantalla vas a observar la opción de invitar un colaborador, únicamente será necesario hacer click y llenar el pequeño formulario que aparecerá luego.

Captura de pantalla de computadora

Descripción generada automáticamente