fichero .ssh/config

linux, ssh

man page

fichero configuración conexión

para simplificar la conexión vía SSH con diferentes servidores, y en alternativa a crear `alias` de conexiones:

```
alias ssh-local-dns='ssh -v -i /home/mate/.ssh/pi@raspberry pi@192.168.1.60'
```

se puede optar por crear el fichero de configuración de conexión (`~/.ssh/config`) e invocarlo:

```
ssh nombre_host
```

nociones básicas

- uso de `Include <FILE>` para fraccionar el fichero en otros
- uso de `Host *` o `Host *.dominio.com` o `Host 192.168.10.?` para establecer opciones comunes a varios servidores.

```
Host *.co.uk
The following pattern would match any host in the 192.168.0.[0-9] network
range:
Host 192.168.0.?

A pattern-list is a comma-separated list of patterns.  Patterns within
pattern-lists may be negated by preceding them with an exclamation mark (‘!’).
For example, to allow a key to be used from anywhere within an organisation
except from the “dialup” pool, the following entry (in authorized_keys) could
be used:
from="!*.dialup.example.com,*.example.com"
```

- opciones interesantes:
  - `IdentitiesOnly=yes` # only use the authentication identity files configured in the ssh_config files
  - `PreferredAuthentications=publickey` # Specifies the order in which the client should try protocol 2 authentication methods.«gssapi-with-mic, hostbased, publickey, keyboard-interactive, password»
  - `AddressFamily inet` # any, inet (IPv4), inet6 (IPv6)
  - `Protocol 2`
  - `Compression yes`
  - `ServerAliveInterval 60`
  - `ServerAliveCountMax 20`
  - `LogLevel INFO`

estructura ficheros

dentro del fichero `~/.ssh/config`

```
Host dev
    HostName dev.example.com
    Port 22000
    User fooey
```

repositorios git

```
Host github.com
    User git
    HostName github.com
    IdentityFile ~/.ssh/my.key
```

```
git clone git@github.com:orgname/repository.git
```

mismo repositorios git, varios usuarios

```
#user1 account
 Host bitbucket.org-user1
     HostName bitbucket.org
     User git
     IdentityFile ~/.ssh/user1
     IdentitiesOnly yes

 #user2 account
 Host bitbucket.org-user2
     HostName bitbucket.org
     User git
     IdentityFile ~/.ssh/user2
     IdentitiesOnly yes
```

Si se quiere automatizar el pull/push con diferentes cuentas se debe:

- crear esa configuración en el fichero `~/.ssh/config`
- modificar el user/email del proyecto en cuestión
    - `git config user.name «user1»`
    - `git config user.email «user1@example.com»`
- modificar el upstream del proyecto
    - `git remote set-url origin git@bitbucket.org-user1:user1/your-repo-name.git`
- o usar la cadena correcta al clonar/importar:
    - `git clone git@bitbucket.org-user1:user1/your-repo-name.git`

/via: https://developer.atlassian.com/blog/2016/04/different-ssh-keys-multiple-bitbucket-accounts/

tunneling

```
Host tunnel
    HostName database.example.com
```

```
    IdentityFile ~/.ssh/other.key
    LocalForward 9906 127.0.0.1:3306
    User fooey
```

equivaldría a:

```
ssh -f -N -L 9906:127.0.0.1:3306 fooey@database.example.com
```

y se ejecutaría:

```
ssh -f -N tunnel
```

enlaces

- info de este tutorial: https://nerderati.com/2011/03/17/simplify-your-life-with-an-ssh-config-file/
- https://linux.die.net/man/5/ssh_config
    - según el tutorial, se pueden hacer muchas combinaciones, por ejemplo:
        - cambiar el número de conexiones
        - establecer el nivel de log mostrado
        - variables de entorno que se pueden pasar
        - uso de wildcards para los hosts
- https://www.ssh.com/ssh/config/

From:
https://matebcn.eu/ - **miguel angel torres egea**

Permanent link:
https://matebcn.eu/linux:ssh:clientconfig

Last update: **05/05/2026 09:43**