**Jenkins SSH Setup**

I always get confused and struggle with ssh setup:

could you please explain in detail based on below screen shot?

ChatGPT said:

Absolutely — SSH key setup can be confusing, especially when juggling multiple keys or repos.

Based on your screenshot, here’s a detailed explanation of each file, how SSH works, and how to set it up properly for Jenkins and Bitbucket:

✅ What Each File in Your ~/.ssh Folder Means

File Purpose

id\_rsa Private key (default RSA) – never share this file!

id\_rsa.pub Public key for id\_rsa. Add this to Git services (e.g., Bitbucket).

id\_ed25519 Another private key (newer format, faster, smaller)

id\_ed25519.pub Public key for id\_ed25519

authorized\_keys On a server: stores public keys allowed to connect (used in servers, not needed on client)

config SSH config file — maps hostnames to specific key files, usernames, etc.

known\_hosts Keeps fingerprints of previously accessed hosts to verify identity

known\_hosts.old Backup of the above file

🔐 Basic Concept of SSH Key Auth

You create a keypair: private + public (id\_rsa + id\_rsa.pub)

You add the public key to Bitbucket → Personal Settings → SSH Keys

Git/SSH uses the private key to authenticate when cloning/pulling

✅ How to Configure SSH for Bitbucket Access (Safe + Clean)

Let’s say you want Jenkins to access this repo:

scss

Copy

Edit

git@bitbucket.org:mkeonekey/qa-ios-automation2.git

Follow these steps:

🔧 1. Decide Which Key to Use

You can use either:

id\_rsa (traditional)

id\_ed25519 (modern, preferred)

Let’s assume you want to use id\_ed25519.

🛠️ 2. Configure ~/.ssh/config (The Most Important Part)

Edit the config file and add:

ssh

Copy

Edit

Host bitbucket.org

HostName bitbucket.org

User git

IdentityFile ~/.ssh/id\_ed25519

IdentitiesOnly yes

✅ This tells SSH to use id\_ed25519 whenever connecting to bitbucket.org, ignoring all other identities.

🔐 3. Add Public Key to Bitbucket

Go to Bitbucket → Personal Settings → SSH Keys

Click Add Key

Copy the contents of id\_ed25519.pub and paste it there

bash

Copy

Edit

cat ~/.ssh/id\_ed25519.pub

🧪 4. Test the Connection

From your terminal:

bash

Copy

Edit

ssh -T git@bitbucket.org

Expected output:

pgsql

Copy

Edit

logged in as <your-bitbucket-username>.

You can use git or hg to connect to Bitbucket.

🧩 5. Use Same Key in Jenkins (if needed)

If Jenkins is running on the same machine:

You’re done! It’ll use your ~/.ssh/config and connect via id\_ed25519.

If Jenkins is running on a server:

Copy your private key (id\_ed25519) to Jenkins credentials:

Jenkins → Manage Jenkins → Credentials

Add SSH Username with private key

Username: git

Private key: paste contents of id\_ed25519

ID: bitbucket-ssh (or any ID)

Then in your Jenkins job, use:

groovy

Copy

Edit

checkout([$class: 'GitSCM',

userRemoteConfigs: [[

url: 'git@bitbucket.org:mkeonekey/qa-ios-automation2.git',

credentialsId: 'bitbucket-ssh'

]],

...

])

🧠 Summary (Mental Model)

text

Copy

Edit

Local Machine / Jenkins Agent

│

├── ~/.ssh/

│ ├── id\_ed25519 (private key)

│ ├── id\_ed25519.pub (public key → added to Bitbucket)

│ ├── config (tells SSH which key to use for bitbucket.org)

│

└── You run: git@bitbucket.org:mkeonekey/repo.git

→ SSH uses config → loads id\_ed25519 → Bitbucket authenticates

Would you like a template SSH config file that supports multiple Bitbucket accounts or different repos using different keys?

You said:

yes

ChatGPT said:

Here’s a template ~/.ssh/config file designed to handle:

✅ One or more Bitbucket accounts

✅ Multiple repositories using different keys

✅ Jenkins, local, or personal setup

✅ Easy switching between identities

🧾 SSH Config Template (Multi-Key Bitbucket Setup)

ssh

Copy

Edit

# === Default Bitbucket account (personal or Jenkins) ===

Host bitbucket.org

HostName bitbucket.org

User git

IdentityFile ~/.ssh/id\_ed25519

IdentitiesOnly yes

# === Optional: Second Bitbucket account (e.g., Work) ===

Host bitbucket-work

HostName bitbucket.org

User git

IdentityFile ~/.ssh/id\_rsa

IdentitiesOnly yes

💡 How to Use

What You Want to Do How

Access Bitbucket normally Use: git@bitbucket.org:<user>/<repo>.git

Use second key (e.g., for qa-ios-automation2) Use: git@bitbucket-work:<user>/<repo>.git

Example .git/config:

ini

Copy

Edit

[remote "origin"]

url = git@bitbucket-work:mkeonekey/qa-ios-automation2.git

✅ Folder Permissions (Important)

Make sure these permissions are correct:

bash

Copy

Edit

chmod 700 ~/.ssh

chmod 600 ~/.ssh/id\_ed25519

chmod 600 ~/.ssh/id\_rsa

chmod 644 ~/.ssh/id\_ed25519.pub

chmod 644 ~/.ssh/id\_rsa.pub

chmod 644 ~/.ssh/config

✅ Test SSH Key Selection

bash

Copy

Edit

ssh -T git@bitbucket.org

# or

ssh -T git@bitbucket-work

🧠 Pro Tip

If you only ever use one key for all Bitbucket repos, your config can be very simple:

ssh

Copy

Edit

Host bitbucket.org

HostName bitbucket.org

User git

IdentityFile ~/.ssh/id\_ed25519

IdentitiesOnly yes

But if you're working on different projects (personal vs company), mapping hosts like bitbucket-work makes life easier.

Would you like help adjusting .git/config inside your automation project to match this setup?

You said:

in my screenshot there are "authorized\_keys" and "known\_hosts" files, what are these and how to use please explain

ChatGPT said:

Great question! Let's break down what authorized\_keys and known\_hosts are in your ~/.ssh/ folder (from your screenshot), what they do, and how/when you interact with them.

✅ 1. authorized\_keys — Used by servers to accept SSH connections

🔹 Purpose:

The authorized\_keys file is used on a server to specify which public keys are allowed to SSH into that machine.

🔹 When is it used?

When you are hosting a server (like Jenkins, Git server, remote Linux host)

You copy your public key (id\_rsa.pub, id\_ed25519.pub) into the server's ~/.ssh/authorized\_keys

🔹 Example:

If you want to SSH into a remote machine (e.g., ubuntu@192.168.1.100) without a password:

On your local machine:

bash

Copy

Edit

ssh-keygen -t ed25519 -f ~/.ssh/id\_ed25519

Then:

bash

Copy

Edit

ssh-copy-id -i ~/.ssh/id\_ed25519.pub ubuntu@192.168.1.100

Or manually append:

bash

Copy

Edit

cat ~/.ssh/id\_ed25519.pub | ssh ubuntu@192.168.1.100 "mkdir -p ~/.ssh && cat >> ~/.ssh/authorized\_keys"

Now, your private key (id\_ed25519) lets you in, because the public key is in the server’s authorized\_keys.

✅ 2. known\_hosts — Used by clients to verify SSH servers

🔹 Purpose:

The known\_hosts file keeps a list of server fingerprints you've connected to before — to prevent man-in-the-middle attacks.

When you connect to a remote SSH server for the first time, you’ll see:

vbnet

Copy

Edit

The authenticity of host 'bitbucket.org (18.205.93.1)' can't be established.

RSA key fingerprint is SHA256:xxxxx

Are you sure you want to continue connecting (yes/no)?

If you type yes, the fingerprint is saved in ~/.ssh/known\_hosts.

🔹 Example Line:

bash

Copy

Edit

bitbucket.org,18.205.93.1 ssh-ed25519 AAAAC3Nz...base64...

🔐 Security Role:

If someone tries to spoof Bitbucket's SSH server IP, SSH will warn you:

makefile

Copy

Edit

WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!

This is a very good thing — it means someone may be trying to impersonate the server.

🔁 Summary: Their Roles

File Role You edit it when… Typical Location

authorized\_keys Server side: allows public keys to connect You’re configuring who can log into a server ~/.ssh/authorized\_keys on the server

known\_hosts Client side: verifies SSH servers Rarely. Usually automatic on first connect ~/.ssh/known\_hosts on your machine

🧠 Key Points

If you're setting up your Mac or Jenkins agent to connect to Bitbucket, you usually don't touch authorized\_keys

If you're setting up a remote server (like Jenkins host) to allow login, you put your public key into its authorized\_keys

If you're getting warnings about known\_hosts, you can view or reset it:

bash

Copy

Edit

# View known hosts:

cat ~/.ssh/known\_hosts

# Remove one:

ssh-keygen -R bitbucket.org

Would you like me to diagram how id\_ed25519, authorized\_keys, and known\_hosts all interact visually?

You said:

yes

ChatGPT said:

Here’s a visual diagram + simple explanation of how the SSH key system works — especially how id\_ed25519, authorized\_keys, and known\_hosts interact:

🔐 SSH Key-Based Authentication — Visual Flow

plaintext

Copy

Edit

┌─────────────────────┐ ┌────────────────────────┐

│ Your Local Machine │ SSH Connect │ Remote Server │

│ (Jenkins/Mac/etc) │──────────────▶│ (e.g., Bitbucket, EC2) │

└─────────────────────┘ └────────────────────────┘

│ │

│ │

│ Sends: id\_ed25519 (private key) │

│──────────────────────────────────────▶│

│ │

│ │ Looks inside ~/.ssh/authorized\_keys

│ │ to see if your public key is allowed

│ │

│ ├── If match found → ✅ Access granted

│ └── Else → ❌ Access denied

│

│

▼

Your machine saves fingerprint of server in:

~/.ssh/known\_hosts

🧾 Files and Their Roles

File Location Purpose

id\_ed25519 ~/.ssh/ (local machine) Private key used to prove your identity

id\_ed25519.pub ~/.ssh/ (local machine) Public key shared with server

authorized\_keys ~/.ssh/ on remote server Contains allowed public keys

known\_hosts ~/.ssh/ (local machine) Verifies the server’s identity

📍 Example: Jenkins → Bitbucket

Assume Jenkins wants to pull from Bitbucket using SSH:

Jenkins uses id\_ed25519 private key

Bitbucket already has the matching public key (id\_ed25519.pub)

When Jenkins connects to git@bitbucket.org:

SSH uses the private key

Bitbucket matches it with the stored public key

✅ Access granted

✅ Bitbucket’s SSH fingerprint gets saved in Jenkins agent’s known\_hosts on first connect

⛔ If Bitbucket’s SSH key ever changes (rare), Jenkins will complain:

REMOTE HOST IDENTIFICATION HAS CHANGED!

🧠 Summary Mnemonic

plaintext

Copy

Edit

YOU (Jenkins/Mac)

├── id\_ed25519 ← Secret "key"

├── id\_ed25519.pub ← Shared with server

├── known\_hosts ← Trust store of remote servers

SERVER (e.g., Bitbucket, EC2)

├── authorized\_keys ← Whitelist of who can connect

Would you like a script to automatically:

Generate keypair

Add it to your ~/.ssh/config

Test connection to Bitbucket?

You said:

explain in details about

bitbucket repo--> Security --> Access Keys

and

Settings --> personal bitbucket settings --> SSH Keys

ChatGPT said:

Great! Let's go into detailed comparison and use-cases of the two main SSH key options in Bitbucket:

🔐 Two Places to Add SSH Keys in Bitbucket

Location Purpose Scope Common Use Case

✅ Repository > Settings > Access Keys For automated access to a specific repository Repo-level CI/CD tools like Jenkins

👤 Personal Settings > SSH Keys For personal developer access All repos you have access to Your laptop/dev machine

✅ 1. Repo → Settings → Access Keys

🔸 Where to find it:

Navigate to the specific repo (e.g., qa-ios-automation2) →

Go to ⚙️ Repository Settings → Access keys

🔸 What it does:

Allows read-only or read/write access to that single repository

Tied to a public key, typically from Jenkins or a deployment server

Bitbucket associates the key with the repo, not with any individual user

🔸 Example Use Case:

You want Jenkins to clone this one repo:

scss

Copy

Edit

git@bitbucket.org:mkeonekey/qa-ios-automation2.git

Then:

Generate an SSH keypair on Jenkins (id\_ed25519)

Copy the public key (id\_ed25519.pub)

Paste it into this repo's Access Keys section

plaintext

Copy

Edit

Jenkins → uses private key

Bitbucket repo → trusts matching public key from Access Keys

✅ Pros:

Least privilege: Only applies to that one repo

You don’t need a Bitbucket user account tied to it

Safer for automation

⚠️ Limitations:

Access is repo-specific

You’ll need to repeat for each repo Jenkins needs access to

👤 2. Personal Settings → SSH Keys

🔸 Where to find it:

Go to your avatar (bottom left) → Personal Settings → SSH Keys

🔸 What it does:

Associates a public key with your user account

Gives access to all repositories that your user account has access to (read/write based on your permissions)

🔸 Example Use Case:

You’re a developer working locally and want passwordless access to Bitbucket from your Mac terminal:

bash

Copy

Edit

git clone git@bitbucket.org:mkeonekey/my-repo.git

You:

Generate SSH key on your Mac (~/.ssh/id\_ed25519)

Add the public key to your personal Bitbucket settings

Then Git will use that key to identify you when connecting to Bitbucket.

✅ Pros:

One-time setup for access to all your Bitbucket repos

Convenient for local dev workflows

⚠️ Limitations:

Tied to your user account — if you leave the org or lose access, your jobs break

Not recommended for shared or automation access (e.g., Jenkins)

🧠 Which Should You Use?

For... Use

🔄 CI/CD (e.g., Jenkins) ✅ Access Keys (repo-level)

👨‍💻 Local developer laptop ✅ Personal SSH Keys

🔐 Least-privilege, secure setup ✅ Access Keys

🔓 All-repo developer access Personal SSH Keys

🛠️ Real-World Tip

If you're using Jenkins:

DO NOT use a developer's personal SSH key in Jenkins.

Instead:

Generate a unique keypair on the Jenkins agent

Use the public key in Bitbucket → Repository → Access Keys

Add the private key to Jenkins credentials (SSH Username with private key)

Would you like help generating a key on Jenkins and using it with Access Keys in Bitbucket?

You said:

I copied public key from my jenkins server and with it added access key on bitbucket

on jenkins I added credentials with jenkins server's private key still I see attached issue!

ChatGPT said:

Thanks for the details and screenshot — your setup is almost correct, but the Unauthorized error you're getting means that Jenkins is not actually using the right SSH key when it tries to authenticate with Bitbucket.

Here’s a step-by-step checklist to fix the issue:

✅ Quick Fix Checklist

✅ 1. Verify: SSH key on Jenkins & Access Key on Bitbucket match

On your Jenkins server, run:

bash

Copy

Edit

cat ~/.ssh/id\_ed25519.pub

Then compare this output with the Access Key added to:

Bitbucket → Your Repository → ⚙️ Settings → Access Keys

If they don’t match, remove the incorrect key from Bitbucket and paste the correct one again.

✅ 2. Make sure Jenkins credentials are linked to the correct private key

In Jenkins:

Go to Manage Jenkins → Credentials → (global)

Open the credential imac-shared-one-key-tool-lib

Verify that:

Type = SSH Username with private key

Username = git

Private Key = should match the private part of the key you added as Access Key on Bitbucket

To double-check:

On Jenkins agent, run:

bash

Copy

Edit

cat ~/.ssh/id\_ed25519 | pbcopy

Then paste this into the private key section in Jenkins credentials.

🔐 This private key must be the exact pair of the public key you added to Bitbucket Access Keys.

✅ 3. Ensure the id\_ed25519 file is readable and used by SSH

If the key is on the Jenkins agent (~/.ssh/id\_ed25519), verify:

bash

Copy

Edit

chmod 600 ~/.ssh/id\_ed25519

chmod 644 ~/.ssh/id\_ed25519.pub

And edit ~/.ssh/config on Jenkins if needed:

ssh

Copy

Edit

Host bitbucket.org

HostName bitbucket.org

User git

IdentityFile ~/.ssh/id\_ed25519

IdentitiesOnly yes

✅ 4. Test Jenkins SSH manually from shell

To confirm the Jenkins server can access Bitbucket via the right key, run this from the Jenkins shell (or same environment):

bash

Copy

Edit

ssh -T git@bitbucket.org

You should see:

pgsql

Copy

Edit

logged in as <bitbucket user>.

You can use git or hg to connect to Bitbucket. Shell access is disabled.

If you get Permission denied, then SSH isn’t using the key you expect — and the fix is in your .ssh/config.

✅ 5. Final Tip — Use SSH URL in repo

Confirm the Repository URL in your Jenkins job uses SSH format:

scss

Copy

Edit

git@bitbucket.org:mkeonekey/qa-ios-automation.git

❌ Not HTTPS (like https://bitbucket.org/...)

✅ Summary: Working Setup

Component Value

Bitbucket Access Key Public key (id\_ed25519.pub)

Jenkins Credentials SSH private key (id\_ed25519), username = git

Repository URL git@bitbucket.org:mkeonekey/qa-ios-automation.git

Test from Jenkins ssh -T git@bitbucket.org should work

Optional .ssh/config Ensure correct IdentityFile is mapped to bitbucket.org