SSH Key

SSH keys are used in portal to access servers. it's not possible to add server membership to users without SSH key.

Once you have your SHH key pair generated, go to your user profile and input **public** part of your key into *Public SSH key* field in *Security management* section.

Recommendations

TDS recommends using EdDSA keys (ed25519):

- as it is more secure than RSA:
  - https://arstechnica.com/security/2023/11/hackers-can-steal-ssh-cryptographic-keys-in-new-cutting-edge-attack
- EdDSA and ECDSA are not only more secure, but also faster than RSA
  - RSA causes slower connection as it has bigger overhead
  - https://goteleport.com/blog/comparing-ssh-keys/
- long term future plans of SSH community are to replace DSA and RSA with at least ECDSA or best with EdDSA which is being widely adopted for several years
- long term TDS plans are to suppress usage of DSA/RSA keys and motivate end users to use EdDSA keys

Generating SSH key on Linux

This step fully follows Recommendations chapter.

Run following commands with properly defining your email in a comment:

```
ssh-keygen -t ed25519 -f ~/.ssh/id_ed25519 -C "pista.bacik@example.com"
cat ~/.ssh/id_ed25519.pub
```

Here we explain used options:

- -t: Specifies the type of key to create, we recommend most secure option currently - Ed25519
- -f: Specify the filename of the generated key file. If you want it to be discovered automatically by the SSH agent, it must be stored in the default `.ssh` directory within your home directory.
- -C: An option to specify a comment. It’s purely informational and can be anything. But it’s usually filled with <login>@<hostname> who generated the key.

Then copy output of cat command and use it in desired application like TDS Portal profile, Gitlab profile, etc...

Generating SSH Key in a Windows

Generating SSH key using Git SCM aka Git Bash

Install Git Bash

Make sure you have Git SCM installed as it contains Git Bash.

- automated installation using winget
  - Open PowerShell as administrator

    - Run following command and wait 1-2 minutes for finishing of installation:

      ```
      winget install --id Git.Git -e --source winget
      ```

      If you are getting error with missing winget tool, install winget using these instructions before, you will use it in future for other stuff.
- manual installation
    - Go to https://git-scm.com/download/win
    - Download relevant installer, 64 bit is recommended
    - Execute installation, simply click next, next... Advanced users can focus on some steps during wizard:
        - Choosing default editor (code, notepad++, vim, nano...)
        - Configuring line ending conventions >> Checkout as-is, commit as-is

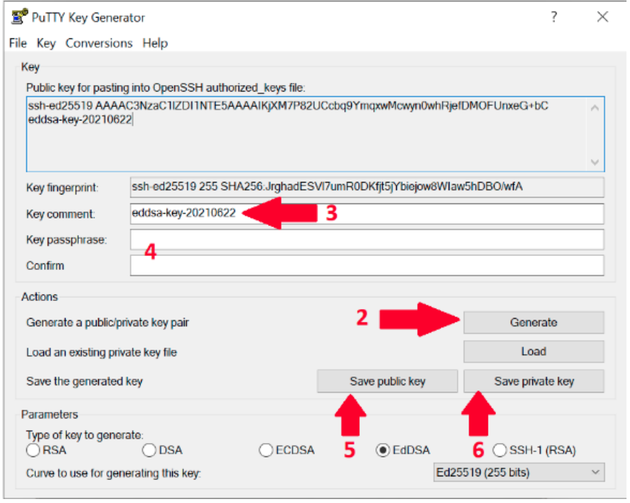
Generate SSH key using Git Bash

Then follow pretty much same instructions as in GeneratingSSHkeyonLinux

Generating SSH key using Putty

Download PuTTY Key Generator (puttygen.exe) from official Putty home page https://www.putty.org/ and start it.

1. Change type to EdDSA, default 256bits is fine. This step fully follows Recommendations chapter.
2. Run the program and click on **Generate** and move your mouse (within the SSH generator window) until key is generated



Public key for pasting into OpenSSH authorized_keys file:

```
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIKjXM7P82UCcbq9YmqxwMcwyn0whRjefDMOFUnxeG+bC
eddsa-key-20210622
```

Key fingerprint: ssh-ed25519 255 SHA256:JrghadESVl7umR0DKfjt5jYbiejow8WIaw5hDBO/wfA

Key comment: eddsa-key-20210622

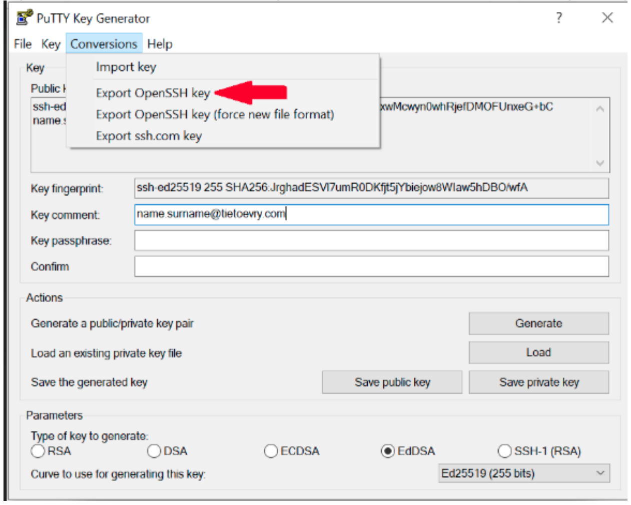
Key passphrase:

Confirm:

Type of key to generate: ○ RSA ○ DSA ○ ECDSA ● EdDSA ○ SSH-1 (RSA)

Curve to use for generating this key: Ed25519 (255 bits)

3. Change *Key comment* to your email address in format like in this example name.surname@example.com
4. Optional: fill *Key passphrase* and confirm passphrase if you need one.
   If you skip this step then you need to confirm that you want to save the keys without passphrase in the next steps.
5. Save public key as ***id_ed25519.pub***
6. Then save your private key in 2 commonly used formats. Please remember to keep private keys as private, never share it with anyone. Only public key can be shared.
    1. Putty format - Click Save private key and store as ***id_ed25519.ppk***
    2. OpenSSH format - From the top menu in Conversions select Export OpenSSH Key and save it as ***id_ed25519***



Key fingerprint: ssh-ed25519 255 SHA256:JrghadESVl7umR0DKfjt5jYbiejow8WIaw5hDBO/wfA

Key comment: name.surname@tietoevry.com

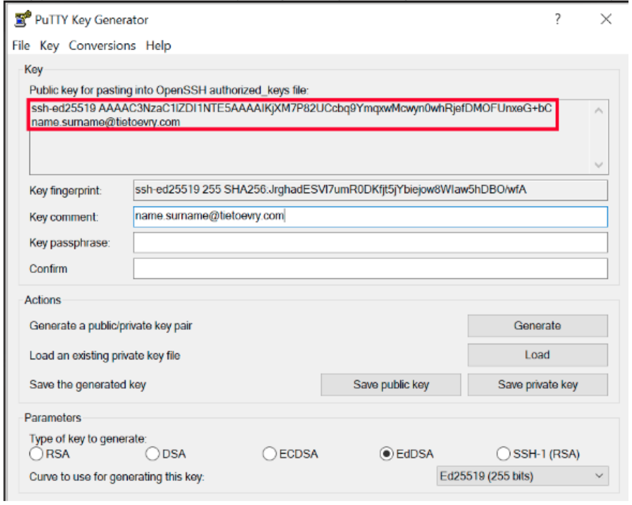
Key passphrase:

Confirm:

Type of key to generate: ○ RSA ○ DSA ○ ECDSA ● EdDSA ○ SSH-1 (RSA)

Curve to use for generating this key: Ed25519 (255 bits)

7. Copy content of "*Public key for pasting into OpenSSH authorized_keys file*"



Public key for pasting into OpenSSH authorized_keys file:

```
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIKjXM7P82UCcbq9YmqxwMcwyn0whRjefDMOFUnxeG+bC
name.surname@tietoevry.com
```

Key fingerprint: ssh-ed25519 255 SHA256:JrghadESVl7umR0DKfjt5jYbiejow8WIaw5hDBO/wfA

Key comment: name.surname@tietoevry.com

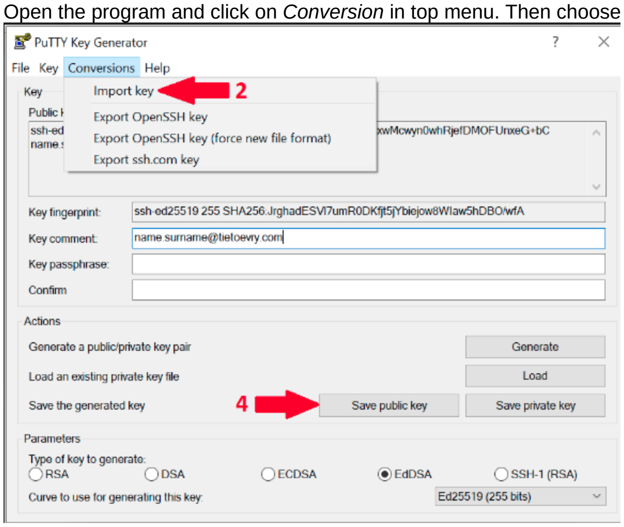
Key passphrase:

Confirm:

8. Go to the TDS Portal and click on user icon located in top right corner of portal page. Then click on your name.
9. In your profile click on "EDIT" button and then enter public part of your SSH key.
10. Paste copied key from the step 7. Click on the **Save** button.
11. Save all your generated SSH key files into folder **~/.ssh** which is the same as **C:\Users\<your_username>\.ssh** folder. Thanks to this standardised location some of your applications will start using SSH key automatically. You will be always able to find it in case of need. Also TDS support team can help you more promptly in case of need.

Converting SSH Keys to PPK Format

1. Download PuTTY Key Generator
2. Open the program and click on *Conversion* in top menu. Then choose *Import key.*



Key fingerprint: ssh-ed25519 255 SHA256:JrghadESVl7umR0DKfjt5jYbiejow8WIaw5hDBO/wfA

Key comment: name.surname@tietoevry.com

Key passphrase:

Confirm:

3. Locate your key in your computer and click open.
4. Once the key is loaded, you can save it as PPK file by clicking on **Save Private Key** or on **Save Public Key** if you want public version of your key.

Troubleshooting

My authentication gets rejected when I try to connect from Windows client to Linux servers via SSH

Recommendation for Windows users - please make sure that your username provided to server is correct one.

For example user "Pišta Báik" with username "bacikpis" with computer in XYZ domain will have username "XYZ+bacikpis" which is not gonna work as server expects just your username.

There are at least 2 ways how to make it work properly

- Use always your username in SSH command during connecting to server as in example:

```
bacikpis@server123...
```

- Or you can configure SSH client to use certain username for any servers by default by configuring this into **~/.ssh/config** file:

  ```
  Host *
    User bacikpis
  ```

Couldn't agree a key exchange algorithm

Symptoms

Otherwise you might experience errors while trying to connect.

```
FATAL ERROR: Couldn't agree a key exchange algorithm (available: curve25519-sha256,curve25519-sha256@libssh.org,
ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-sha2-nistp521,diffie-hellman-group16-sha512,diffie-hellman-group18-
sha512,diffie-hellman-group14-sha256)
```

Root cause

Several old ssh exchange algorithms were removed since CentOS 8 and Ubuntu 20. Most probably you are using old SSH RSA key. Lets resolve it as described in next steps.

Solution

- Make sure to always use latest version SSH client on your side (OpenSSH, Putty or other).
- Make sure you have Ed25519 SSH key generated according to instructions.
- Upload this new PUB key into portal - it will get distributed to your servers automatically, please be patient, it might take some time to replicate.

Workaround

> This is NOT recommended for permanent use, it is only intended for emergency temporary use when you have other serious blockers/obstacles in you way and you are able to accept lower security standards.

If you accept temporarily lowering security standards, you can allow other ciphers in SSH config on server.