Orchestration - Servers

Server Deployment

Servers

A server is a virtual machine running in Tieto DevOps Space with installed and configured TDS supported operating system.

All Servers are managed by its project's users with proper rights.

Users with Project Owner rights have full responsibility for his/her project's Applications, servers management and security protection.

Servers page

Servers page can be found in the main tab under SaaS and represents a list of virtual servers in projects. Each server in the list has information about server name, URL, status and Applications (if any).

The user also can add the server to Favorites by clicking on Star icon in the up-right corner for each server.

Click to server name redirects to Detail page:

Detail	Basic details about the server including date of creation, authentication type, OS information, a username of the creator of the server
Connections	Information about connecting to the server, commonly via SSH or LDAP
Applications (if applicable)	List of applications installed on the server
Users	Users and their privileges on the server are managed from this page
Logs	List of events listed by date
Usage	Graphs about the usage of the CPU, RAM and Disk capacity
Backups	List of backups, user can enable/disable regular backups of the server
Settings	A page for changing server status and capacity. Admin can also Delete server from this page

How to deploy new server

1. **step** - Login into TDS portal.
2. **step** -  **Button -** allows adding a new server from the Store. For creating new server simply Select desired server type from the list. Each item in the list represents a single server.
3. **step** - Next page named Store Config displays basic information about the server, estimated price and several checkboxes which can be filled.

   User can change:

   - hostname (at least 3 characters with at least 1 letter and 1 hypfen "-", maximum characters is 15).
   - authentication method (login to the server via ssh)
   - the capacity of the server (CPU, RAM and disk capacity)

4. **step** - After all information is filled, clicking on the Order button will start creating the server. There are several types of information which are already pre-filled but can be changed.

Only some information needs to be filled.

SSH key management

SSH Key

Certificates management

Enabling certificates automatically creates /data/ssl with relevant files:

- ca-bundle.crt - chain of root and intermediate certificates that signed server certificate
- server.crt
- server.key

Self-managed PaaS applications (Gerrit, Jenkins, SonarQube) from TDS are automatically configured to use those certificate files.

You can also create hooks if you want to execute some commands after new certificate is deployed. Typically it is restart of some service. Also you can use hooks which can be automatically executed after each new certificate deploy - look for more in Certificatedeploymenthooks chapter.

CA certificates management

How to make new Lets Encrypt CA certificates trusted

Certificate deployment hooks

Hooks shall be bash scripts made executable and placed in folder /data/ssl/hooks folder. It will be automatically executed every time when new certificate is deployed.

Nginx certificates hook example

For Nginx web server it is recommended to have server certificate and intermediate certificates bundled in file configured by "ssl_certificate" directive: http://nginx.org/en/docs/http/ngx_http_ssl_module.html#ssl_certificate

Example of setting of correct certificate path in nginx files:

```
# Example of configuring recommended path to complete chain
grep 'ssl_certificate /' /etc/nginx/sites-available/*
sed -i 's#ssl_certificate /.*#ssl_certificate /data/ssl/fullchain.crt;#' /etc/nginx/sites-available/*
sed -i 's#ssl_certificate_key /.*#ssl_certificate_key /data/ssl/server.key;#' /etc/nginx/sites-available/*
grep 'ssl_certificate /' /etc/nginx/sites-available/*
```

This is recommended setup verified by users:

```
# Preparing hook:
mkdir -p /data/ssl/hooks/
touch /data/ssl/hooks/nginx.sh
chmod +x /data/ssl/hooks/nginx.sh
echo '#!/bin/sh
cat /data/ssl/server.crt > /data/ssl/fullchain.crt
cat /data/ssl/ca-bundle.crt >> /data/ssl/fullchain.crt
systemctl restart nginx' > /data/ssl/hooks/nginx.sh
cat /data/ssl/hooks/nginx.sh

 # Finally executing the hook to verify that it works
/data/ssl/hooks/nginx.sh
```

Logs

Logs