Figure 1: BBrowserX

A GPU-accelerated single-cell platform by Bio-Turing®

BBrowserX - GPU enterprise version installation guide

This edition of the installation guide describes the installation process of BioTuring® Browser X (BBrowserX) for supported: container runtime (Docker/Containerd), K8S, and standalone Linux machine.

1. Introduction

BBrowserX is a GPU-accelerated single-cell platform developed by BioTuring®. It enables dramatic increases in single-cell analysis computing performance by harnessing the power of the graphics processing unit (GPU).

1.1. Pre-Installation Requirements

Before installing the BBrowserX on Linux/K8S, some pre-installation steps are required: - Container runtime (Docker, Containerd) or K8s - The system has one or multiple NVIDIA GPU(s) (at least 16 GB memory per GPU) - SSL certificate and a domain name for users to securely access the platform on the web browser - Token obtained from Bioturing

1.2. OS Support Policy

- BBrowserX support for Ubuntu 18.04.x, Ubuntu 20.04.x, Ubuntu 22.04.x, RHEL 7.x, RHEL 8.x, RHEL 9.x , CentOS 7.x. and K8S
- BBrowserX supports a single and latest Debian release version - as this is the only version of Debian that has CUDA support. For Debian release timelines, visit DebianReleases.

Refer to the support lifecycle for these supported OSes to know their support timelines and plan to move to newer releases accordingly.

2. Self-Signed CA Certificate installation (optional):

Adding the self-signed certificate as trusted to your proxy agent/server

`For Ubuntu OS:`

```
bash ./cert/ubuntu.sh

For Redhat/Centos OS:

bash ./cert/rhel.sh
```

3. Prebuilt binary files (in standalone machine or K8S node)

This section includes instructions for deploying BBrowserX on supported Ubuntu 20.04.x using prebuilt binary files.
For other OSes installations, please contact us.

Note: We suggest starting from scratch to avoid package/driver conflicts.

0. Pre-installation:

```
sudo apt update && sudo apt upgrade -y
sudo apt install build-essential curl gnupg lsb-release ca-certificates xfsprogs -y
```

1. Install NVIDIA CUDA Toolkit 11.7.

Run the commands below to install NVIDIA CUDA Toolkit 11.7 on Ubuntu 20.04.x:

```
wget https://developer.download.nvidia.com/compute/cuda/11.7.1/local_installers/cuda_11.7.1_
sudo sh cuda_11.7.1_515.65.01_linux.run

List all the NVIDIA GPUs in the system:
nvidia-smi -L
```

2. Install NVIDIA Container Toolkit

Make sure you have installed the NVIDIA driver and Docker engine for your Linux distribution (or K8s NODE).

```
Check for NVIDIA Container Toolkit at: https://github.com/NVIDIA/nvidia-docker
```

3. Patch container engines (Docker, Containerd)

Check for Nvidia cloud native at: https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/install-guide.html

Check container engines (Docker, Containerd)

```
For microk8s :
microk8s kubectl describe no | grep Runtime

For vanilla :
kubectl describe no | grep Runtime
```

If container engine is Containerd Add these lines to : /etc/containerd/config.toml

```
privileged_without_host_devices = false
base_runtime_spec = ""
[plugins."io.containerd.grpc.v1.cri".containerd.runtimes.runc.options]
    SystemdCgroup = true
[plugins."io.containerd.grpc.v1.cri".containerd.runtimes.nvidia]
    privileged_without_host_devices = false
    runtime_engine = ""
    runtime_root = ""
    runtime_type = "io.containerd.runc.v1"
    [plugins."io.containerd.grpc.v1.cri".containerd.runtimes.nvidia.options]
    BinaryName = "/usr/bin/nvidia-container-runtime"
    SystemdCgroup = true
[plugins."io.containerd.grpc.v1.cri".cni]
bin_dir = "/opt/cni/bin"
conf_dir = "/etc/cni/net.d"
```

```
sudo systemctl restart containerd
sudo nvidia-container-cli --load-kmods info
```

If container engine is Docker Add these lines to : /etc/docker/daemon.json

```
{
    "default-runtime": "nvidia",
    "runtimes": {
        "nvidia": {
            "path": "nvidia-container-runtime",
            "runtimeArgs": []
        }
    }
}
```

```
sudo systemctl restart docker
sudo nvidia-container-cli --load-kmods info
```

4. BioTuring Docker Hub

`Check for BioTuring docker images at: https://registry.bioturing.com/`

5. Docker Installation:

We support container runtime: Docker, Containerd.

Note: The ideal system that we recommend for most companies is AWS g5.8xlarge

1. Simple installation:

`For Ubuntu OS:`

```
bash ./install.docker.sh ubuntu

For Redhat/Centos OS:

bash ./install.docker.sh rhel
```

2. Manual Installation:

```
docker container run -d -t -i \
    -e WEB_DOMAIN='CHANGE THIS TO YOUR DOMAIN' \
    -e BIOTURING_TOKEN='USE TOKEN OBTAINED FROM BIOTURING' \
    -e ADMIN_USERNAME='admin' \
    -e ADMIN_PASSWORD='CHANGE YOUR PASSWORD IF NECESSARY' \
    -p 80:80 \
    -p 443:443 \
    -v '/path/to/persistent/storage/':/data/user_data \
    -v '/path/to/stateful/storage/':/data/app_data \
    -v '/path/to/ssl/storage/':/config/ssl \
    --link bioturing-ecosystem:latest \
    --name bioturing-ecosystem
```

6. Kubernetes installation

Kubernetes, also known as K8s, is an open-source system for automated deployment, scaling, and management of containerized applications.

We support all k8s engines: GKE (Google Kubernetes Engine), EKS (Amazon Elastic Kubernetes Service), AKS (Azure Kubernetes Service), MicroK8s, and vanilla K8S.

1. Ensure that helm (version 3) is installed.

First, check the Helm version

```
Example :

microk8s enable helm3

microk8s helm3 version
```

2. Add BioTuring Helm charts

```
Example:

For Vanilla K8s:

helm repo add bioturing https://registry.bioturing.com/charts/

For Microk8s:
```

```
microk8s helm3 repo add bioturing https://registry.bioturing.com/charts/
```

3. Simple Installation: use BioTuring installation script

```
For Vanilla k8s:

bash ./install.k8s.sh vanilla

For Micro8s:

bash ./install.k8s.sh microk8s
```

4. Check pods information

```
microk8s kubectl get all

microk8s kubectl get pods
microk8s kubectl get services --all-namespaces
microk8s kubectl get services
microk8s kubectl get pvc

microk8s kubectl logs bioturing-ecosystem-0

microk8s.kubectl -n ingress get pods
microk8s.kubectl -n ingress logs <your pod name here> | grep reload
```

5. Check secrets

- bioturing-ecosystem-tls
- bioturing-ecosystem
- bioturingregred

```
microk8s kubectl edit secrets mysecret

Example:
microk8s kubectl edit secrets bioturing-ecosystem-tls
```

6. Helm chart Values

Kubernetes: `>=1.19.0-0`

Key	Type	Default	Description
image.tag	string	`"1.0.11"`	image tag
secret.data.domain	string	`"bbrowserx.com"`	your domain
secret.data.bbtoken	string	`""`	bioturing access token
secret.admin.username	string	`admin`	username
secret.admin.password	string	`turing2022`	password
secret.server.useletsencrypt	string	`"false"`	
secret.server.certificate	string	`""`	CRT base64 string
secret.server.key	string	`""`	KEY base64 string

Key	Type	Default	Description
service.type	string	`ClusterIP`	
service.ports.http.port	int	80	
service.ports.https.port	int	443	
persistence.dirs.app.size	string	`5Gi`	APP size
persistence.dirs.user.size	string	`5Gi`	USER size
persistence.storageClass	string	`""`	
ingress.enabled	bool	`true`	
ingress.className	string	`""`	
ingress.annotations	object	`{}`	
ingress.tls.enabled	bool	`true`	
resources	object	`{}`	
autoscaling	object	`{}`	
nodeSelector	object	`{}`	
tolerations	object	`{}`	
affinity	object	`{}`	
podAnnotations	object	`{}`	
podSecurityContext	object	`{}`	
securityContext	object	`{}`	
serviceAccount.name	string	`""`	

7. Manual Installation.

Please replace paths to your certificate, key, admin password, and other helm chart values of your choice.

```
BBTOKEN="USE TOKEN OBTAINED FROM BIOTURING"
SSLCRT="base64 -w 0 ./bioturing.com.crt" # <- (REPLACE THIS WITH A PATH TO YOUR CRT CERTFICA
SSLKEY="base64 -w 0 ./bioturing.com.key" # <- (REPLACE THIS WITH A PATH TO YOUR KEY)
ADMIN_USERNAME="admin"
ADMIN_PASSWORD="admin" # <- (CHANGE YOUR PASSWORD IF NECESSARY)
USELETSENCRYPT="false"
SVHOST="k8stest.bioturing.com" # <- (CHANGE THIS TO YOUR K8S INGRESS DOMAIN)

For Microk8s:

microk8s helm3 repo update
microk8s helm3 upgrade --install --set secret.data.bbtoken="${BBTOKEN}" \
 --set secret.data.domain="${SVHOST}" \
 --set secret.server.certificate="${SSLCRT}" \
 --set secret.server.key="${SSLKEY}" \
 --set secret.server.useletsencrypt="${USELETSENCRYPT}" \
 --set secret.admin.username="${ADMIN_USERNAME}" \
 --set secret.admin.password="${ADMIN_PASSWORD}" \
bioturing bioturing/ecosystem --version 1.0.11
```

```
For Vanilla k8s:

helm repo update
helm upgrade --install --set secret.data.bbtoken="${BBTOKEN}" \
 --set secret.data.domain="${SVHOST}" \
 --set secret.server.certificate="${SSLCRT}" \
 --set secret.server.key="${SSLKEY}" \
 --set secret.server.useletsencrypt="${USELETSENCRYPT}" \
 --set secret.admin.username="${ADMIN_USERNAME}" \
 --set secret.admin.password="${ADMIN_PASSWORD}" \
bioturing bioturing/ecosystem --version 1.0.11
```

7. Notices

7.1. Security

- The BBrowserX platform uses HTTPS protocol to securely communicate over the network.
- All of the users need to authenticate using a BioTuring account or the company's SSO to access the platform.
- We highly recommend setting up a private VPC network for IP restriction.
- The data stays behind the company firewall.
- The BBrowserX platform does not track any usage logs.

7.2. Data visibility

- Data can be uploaded to Personal Workspace or Data Sharing group.
- In the Personal Workspace, only the owner can see and manipulate the data she/he uploaded.
- In the Data Sharing group, only people in the group can see the data.
- In the Data Sharing group, only people with sufficient permissions can manipulate the data.