Overview

Overview

Casvisor is an open-source platform that provides security log auditing and bastion functionality for managing RDP, VNC, SSH, and databases.

Casvisor Features

Asset Management

- Easily manage and connect to assets using RDP, VNC, and SSH protocols.
- Efficiently handle remote connections to machines.

Security Log Auditing

- Track and monitor remote connections with detailed audit logging.
- Record the start time, duration, and other relevant information of each connection.
- Capture and analyze API logs for Casdoor operations.

Database Management

- Connect and manage databases within Casvisor.
- Efficiently organize and control access to databases.
- Simplify user management and authorization for database resources.

Server Installation

Prerequisite

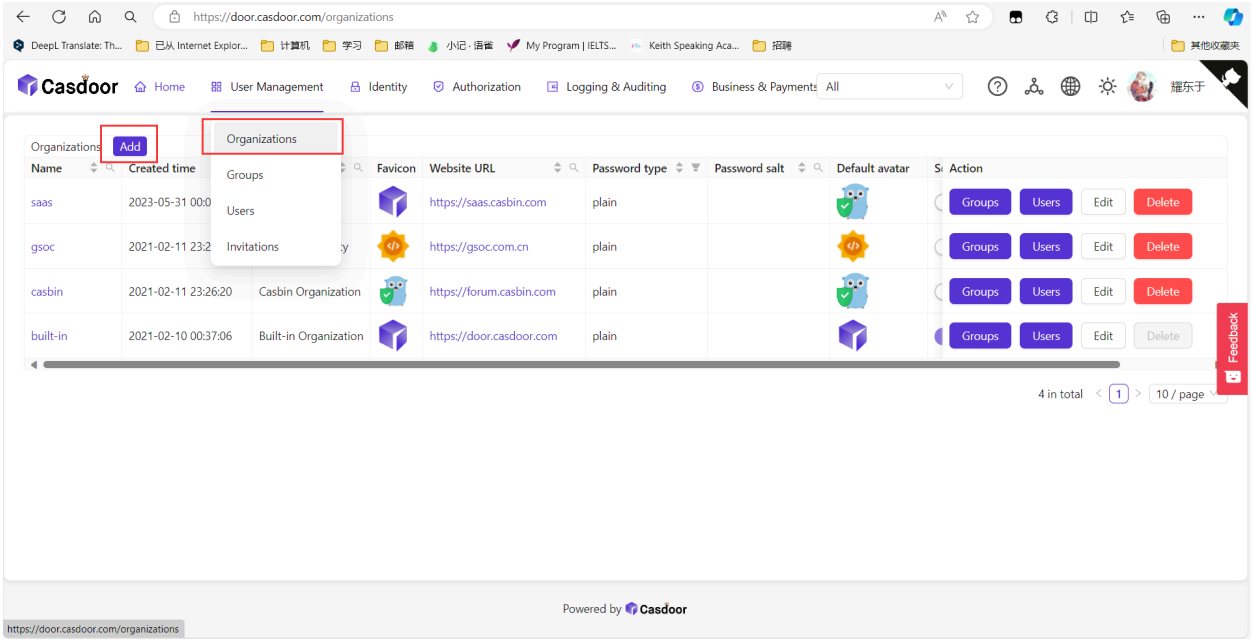
Casvisor server uses Casdoor as the authentication and authorization system. So you need to install Casdoor first. If you haven't installed Casdoor, please refer to Casdoor Installation.

Casdoor

You have installed Casdoor, now you need to do some necessary configuration in Casdoor in order to use Casvisor.

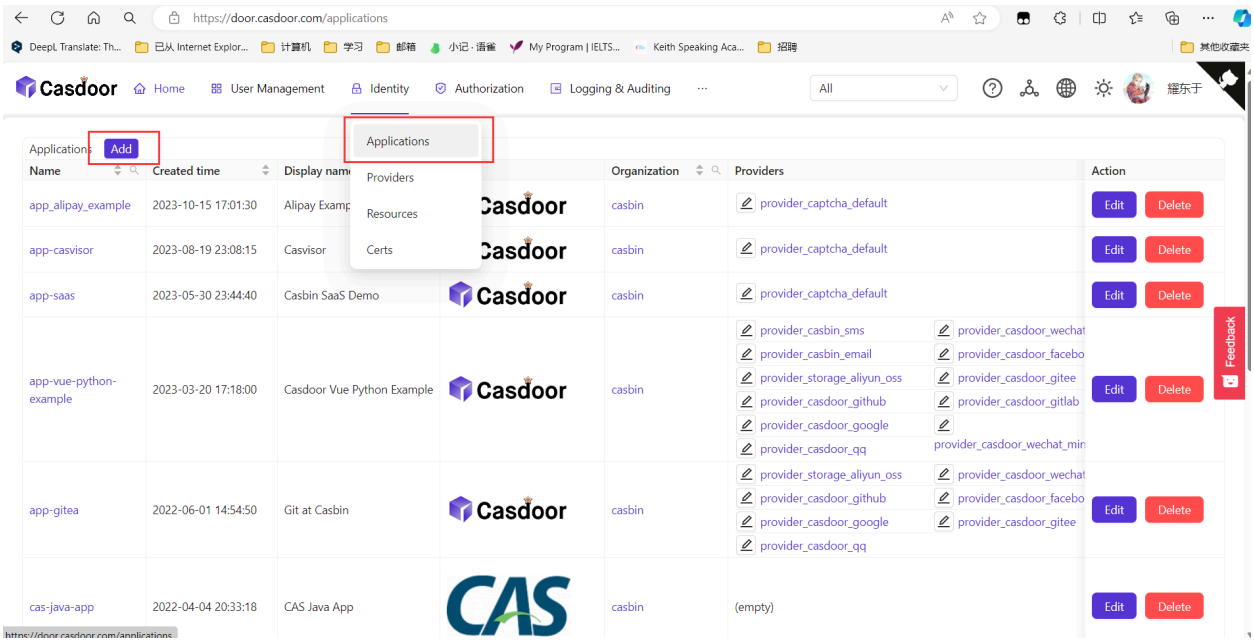
Create an organization

First, you need to create an organization (Except for the `build-in`) in Casdoor. The organization page is at **User Management → Organizations**. And you can create an organization by clicking the `add` button.



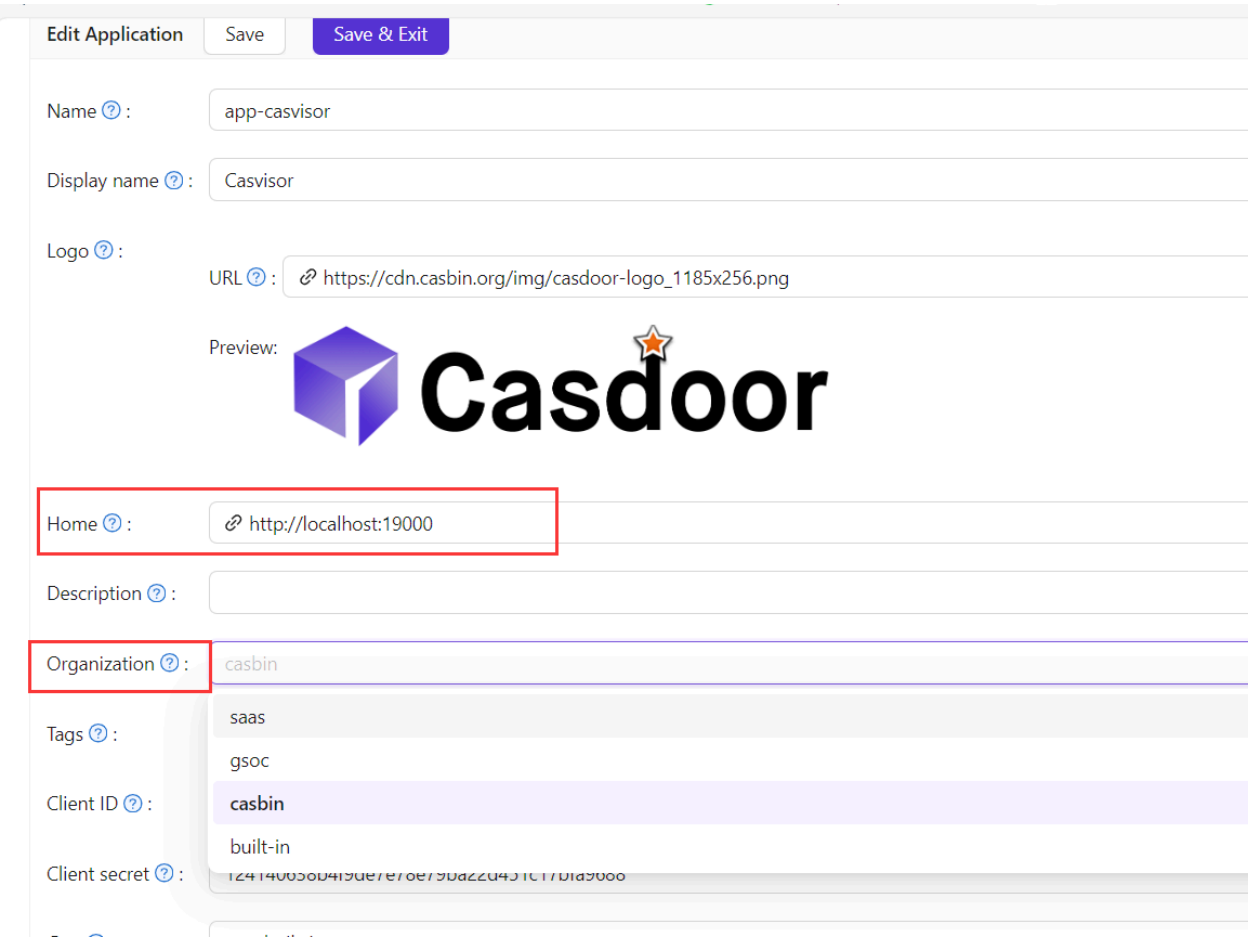
Create an application

You need to create an application for Casvisor in Casdoor. The application page is at **Identity → Applications**. And you can create an application by clicking the `add` button.



Required fields:

1. `Home`: The host of Casvisor server, e.g. `http://localhost:16001`.
2. `Ognization`: The organization you created in the previous step.
3. `Callback URLs`: The callback URL of Casvisor server, e.g. `http://localhost:16001/callback`. You can add multiple callback URLs by clicking the `add` button. These are the urls that is allowed to be redirected after login.



guacd

Casvisor uses guacamole-server to provide remote desktop access. If you want to use this feature, you need to install guacamole-server first. If you haven't installed

guacamole-server, please refer to guacamole-server Installation.

You can also run guacd in docker with the following command:

```
docker run -d --name guacd -p 4822:4822 guacamole/guacd
```

Download

The source code of Casvisor is hosted on GitHub: https://github.com/casvisor/casvisor. Both the Go backend code and React frontend code are contained in a single repository.

Name	Description	Language	Source code
Frontend	Web frontend UI for Casdoor	JavaScript + React	https://github.com/casvisor/casvisor/tree/master/web
Backend	RESTful API backend for Casdoor	Golang + Beego + XORM	https://github.com/casvisor/casvisor

Casvisor supports `Go Modules`. To download the code, simply clone the code using git:

```
git clone https://github.com/casvisor/casvisor
```

Configuration

Backend

The configuration file of Casvisor backend located at `conf/app.conf`. You need to modify the following fields:

Database

Modify `dataSourceName` to your own database connection string. Casvisor will create a database named `casvisor` if it doesn't exist.

```
driverName = mysql
dataSourceName = root:123456@tcp(localhost:3306)/
dbName = casvisor
```

Connect Casdoor

Modify `casdoorEndpoint`, `clientID`, `clientSecret`, `casdoorOrganization` and `casdoorApplication` to your own Casdoor configuration. You can get the `clientID` and `clientSecret` from the application page that you created in the previous step.

```
casdoorEndpoint = http://localhost:8000
clientId = c34fdf145f41313727a8
clientSecret = 615c503d4552d24a40360cf908b6d17e3b7f8832
casdoorOrganization = "casbin"
casdoorApplication = "app-casvisor"
```

Frontend

In `web/src/conf.js`, you need to modify the following fields:

```
    export const AuthConfig = {
    serverUrl: "http://localhost:8000",
    clientId: "c34fdf145f4131b727a8",
    appName: "app-casvisor",
    organizationName: "casbin",
    redirectPath: "/callback",
};
```

Run

Before running Casvisor, make sure Casdoor is running.

Production

In production, you need to build the frontend code first, then run the backend code.

Build frontend

```
cd web
yarn install
yarn build
```

After building successfully, the frontend bundle will be generated in `web/build` directory.

Run backend

```
go build
```

Visit backend server at http://localhost:19000.

Nginx

> TIP
>
> If you use nginx as a reverse proxy, you need to add the following configuration to the nginx configuration file:
>
> ```
> location / {
>     *** your configuration ***
>     proxy_set_header Upgrade $http_upgrade;
>     proxy_set_header Connection "upgrade";
> }
> ```
>
> Because Casvisor uses websocket to communicate with guacd.

Development

In development, you need to run the frontend code and backend code at the same time.

Run frontend

```
cd web
yarn install
```

Run backend

```
go build
```

Visit frontend server http://localhost:16001.

(Optional) Try with Docker

Requirements

Hardware

If you want to build the Docker image yourself, please ensure that your machine has at least **2GB** of memory. Casvisor's frontend is an NPM project of React. Building the frontend requires at least **2GB** of memory. Having less than **2GB** of memory may result in a frontend build failure.

If you only need to run the pre-built image, please ensure that your machine has at least **100MB** of memory.

OS

All operating systems (Linux, Windows, and macOS) are supported.

Docker

You can use **Docker (docker-engine version >= 17.05)** in Linux or **Docker Desktop** in Windows and macOS.

- Docker

Regardless of the operating system, users must ensure that they have **docker-engine version >= 17.05**. This is because we utilize the multi-stage build feature in the docker-compose.yml, which is supported in versions 17.05 and above. For more information, see https://docs.docker.com/develop/develop-images/multistage-build/.

If you are also using docker-compose, please ensure that you have **docker-compose version >= 2.2**. For Linux users, you also need to make sure that docker-compose is installed, as it is separate from docker-engine.

Get the image

We have provided two DockerHub images:

Name	Description	Suggestion
casvisor-all-in-one	Casvisor, MySQL database and guacamole-server are included in the image	This image already includes a toy database and is only for testing purposes
casvisor	Only Casvisor is included in the image	This image can be connected to your own database and used in production

1. casbin/casvisor-all-in-one: This image includes the casvisor binary, a MySQL database and guacamole-server, and all the necessary configurations. It is designed for new users who want to try Casvisor quickly. With this image, you can start Casvisor immediately with just one or two commands, without any complex configuration. However, please note that we **do not recommend** using this image in a production environment.

Casvisor uses Casdoor as the authentication and authorization system. The default configuration of Casvisor is to use the office Casdoor server. If you want to use your own Casdoor server, you need to modify the configuration file `conf/app.conf`.

Option-1: Use the toy database

Run the container with port `19000` exposed to the host. The image will be automatically pulled if it doesn't exist on the local host.

```
docker run -p 19000:19000 casbin/casvisor-all-in-one
```

Visit http://localhost:19000 in your browser.

Option-2: Try with docker-compose

You can use docker-compose.yml in the official repository. Create a `conf/app.conf` directory in the same directory level as the `docker-compose.yml` file. Then, copy app.conf from Casvisor. For more details about `app.conf`, you can see configuration.

```
docker-compose up
```

That's it! 🛩️

Visit http://localhost:19000 in your browser.

Note: If you dig deeper into the docker-compose.yml file, you may be puzzled by the environment variable we created called "RUNNING_IN_DOCKER". When the database 'db' is created via docker-compose, it is available on your PC's localhost but not the localhost of the Casvisor container. To prevent you from running into troubles caused by modifying app.conf, which can be quite difficult for a new user, we provided this environment variable and pre-assigned it in the docker-compose.yml. When this environment variable is set to true, localhost will be replaced with host.docker.internal so that Casvisor can access the database.

Option-3: Try directly with the standard image

MySQL

Mysql is required for Casvisor. If you don't have a MySQL database, you can run it with the following command:

```
docker run  \
       -p 3306:3306 \
       -e MYSQL_ROOT_PASSWORD=123456 \
       -v /usr/local/docker/mysql:/var/lib/mysql \
       mysql:8.0.25
```

Run Casvisor

Create `conf/app.conf`. You can copy it from conf/app.conf in Casvisor. For more details about `app.conf`, you can see configuration.

Then run

```
docker run  -p 19000:19000 -v /folder/of/app.conf:/home/casvisor/
conf casbin/casvisor:latest
```

Anyway, just **mount the folder of app.conf to /home/casvisor/conf** and start the container.

> TIP
>
> If it is not convenient to mount the configuration file to a container, using environment variables is also a possible solution.

example

```
docker run \
  -e driverName=mysql \
  -e dataSourceName='user:password@tcp(x.x.x.x:3306)/' \
  -e casdoorEndpoint=https://door.casdoor.com \
  -e clientId=b108dacba027db36ec26 \
  -e clientSecret=124140638b4f9de7e78e79ba22d451c17bfa9688 \
  -e casdoorOrganization=casbin \
  -e casdoorApplication=app-casvisor \
  -p 19000:19000 \
  casbin/casvisor:latest
```

Run guacd

Casvisor uses guacamole-server to provide remote desktop access. If you want to use this feature, you need to run guacd. You can run guacd with the following command:

```
docker run -d --name guacd -p 4822:4822 guacamole/guacd
```

Visit http://localhost:19000 in your browser.

Assets

Overview

Casvisor Assets Overview

RDP

Casvisor Assets RDP

VNC

Casvisor Assets VNC

Database

Connect to your databases

Intranet

Connect assets in intranet

Overview

Casvisor helps you to manage assets, and connect to your assets remotely, including remote desktop via RDP, VNC, SSH, and databases.

Machine:

- SSH
- RDP
- VNC

Database:

- MySQL
- MariaDB
- PostgreSQL
- Microsoft SQL Server
- Redis
- MongoDB

Every asset has the following basic properties:

- `Organization`: The organization that the asset belongs to.
- `Name`: The unique asset name.
- `Display name`: The display name of the asset.
- `Category`: The category of the asset, including `Machine` and `Database`.
- `Endpoint`: Domain name or IP address.
- `Port`: The port number of the asset.
- `Username`: The username to connect to the asset, such as `root`,

`administrator`, `sa`, etc.

- `Password`: The password to connect to the asset.
- `OS`: The operating system of the asset, including `Windows` and `Linux`, used to classify the asset.
- `Tag`: The tag of the asset, used to classify the asset.

In this chapter, you will learn how to start connecting to your assets.

Let's explore together!

RDP

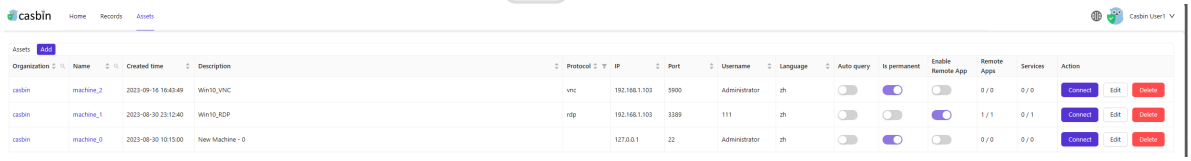
Casvisor Support Connect to your assets via RDP protocol:

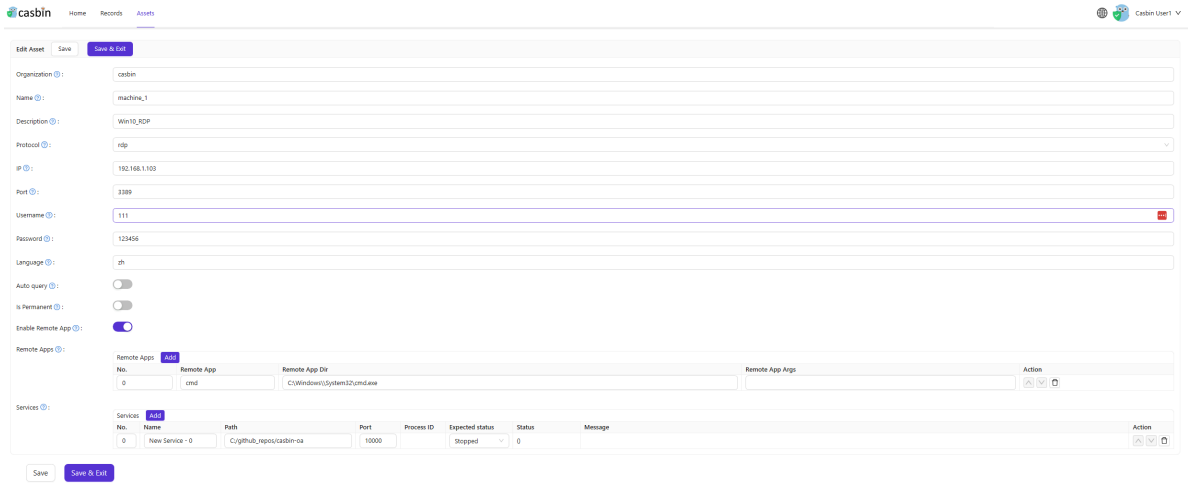
Rdp connection

1. Start Guacamole Server

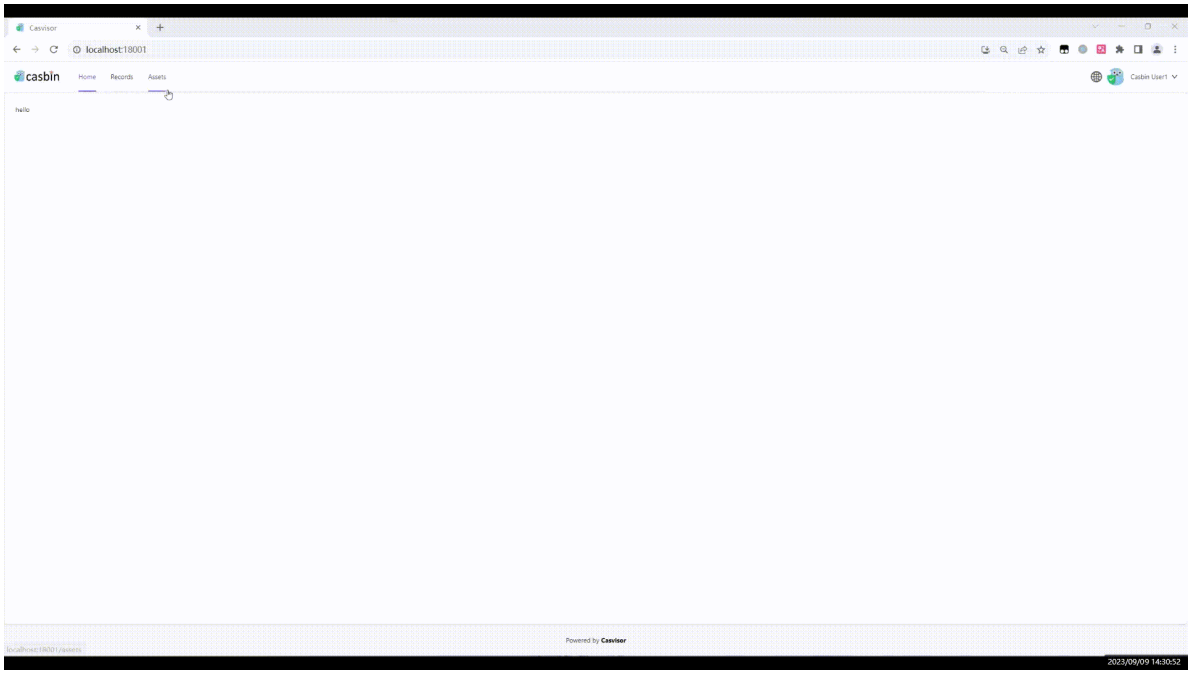
```
docker run --name guacd -d -p 4822:4822 guacamole/guacd
```

2. Add a new asset, set protocol to `rdp`





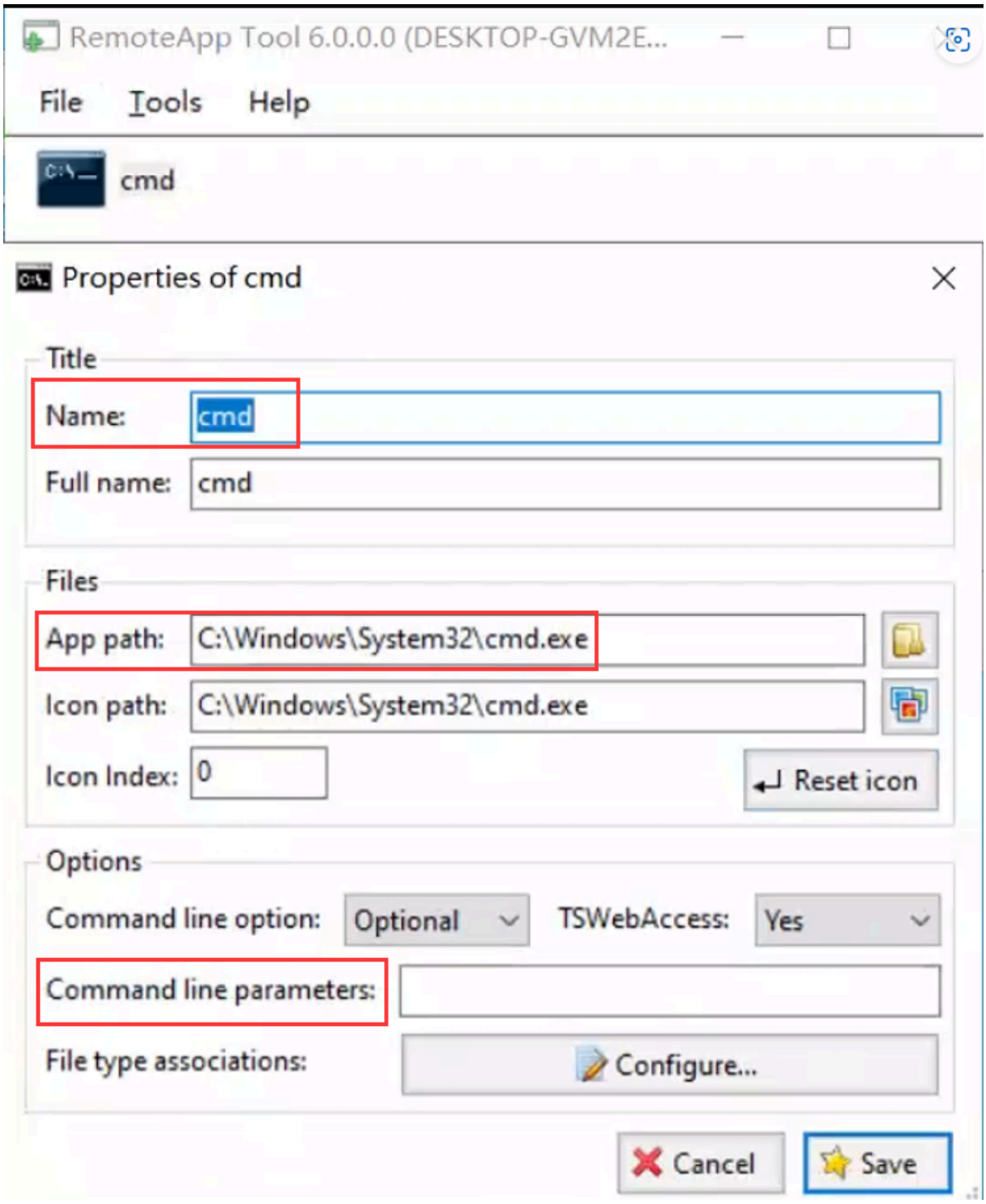
3. Connect to your asset by clicking the `connect` button



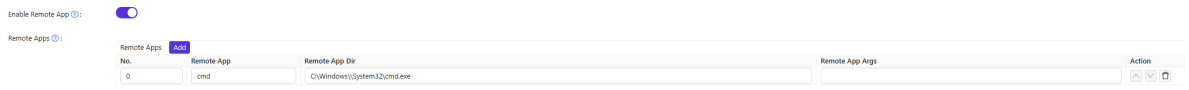
Remote App

We support remote app on Windows assets, you can add remote apps on `Asset Edit` page, and then you can connect to your remote app by clicking the `connect` button.

1. Configure your remote app on the server end.
   You can use RemoteApp Tool to register apps.

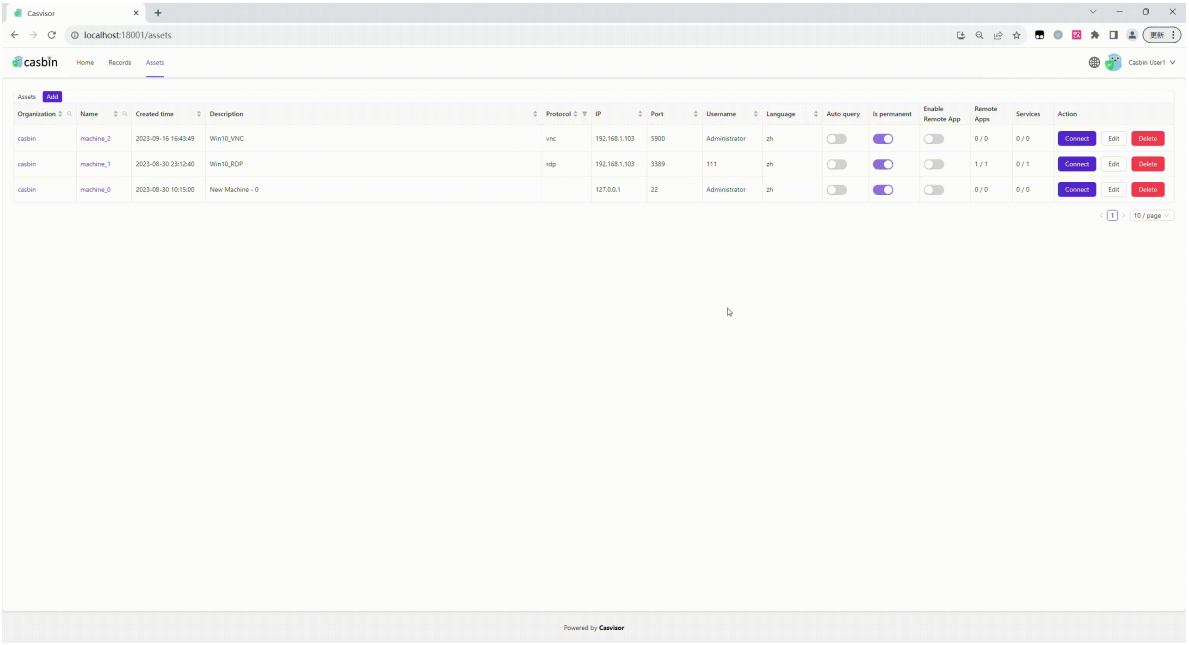


2. Configure the remote app information in the asset edit page according to the server-end configuration. 'remoteAppName', 'remoteAppDir', and 'remoteAppArgs' are required.



refer to Configuring Guacamole — Apache Guacamole Manual v1.5.3

3. Connect to your remote app.



VNC

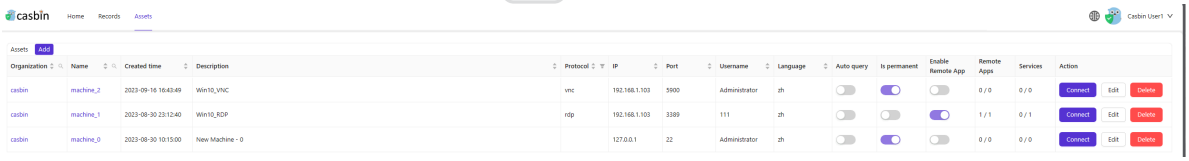
VCN Connect

VCN connection is similar to RDP connections.

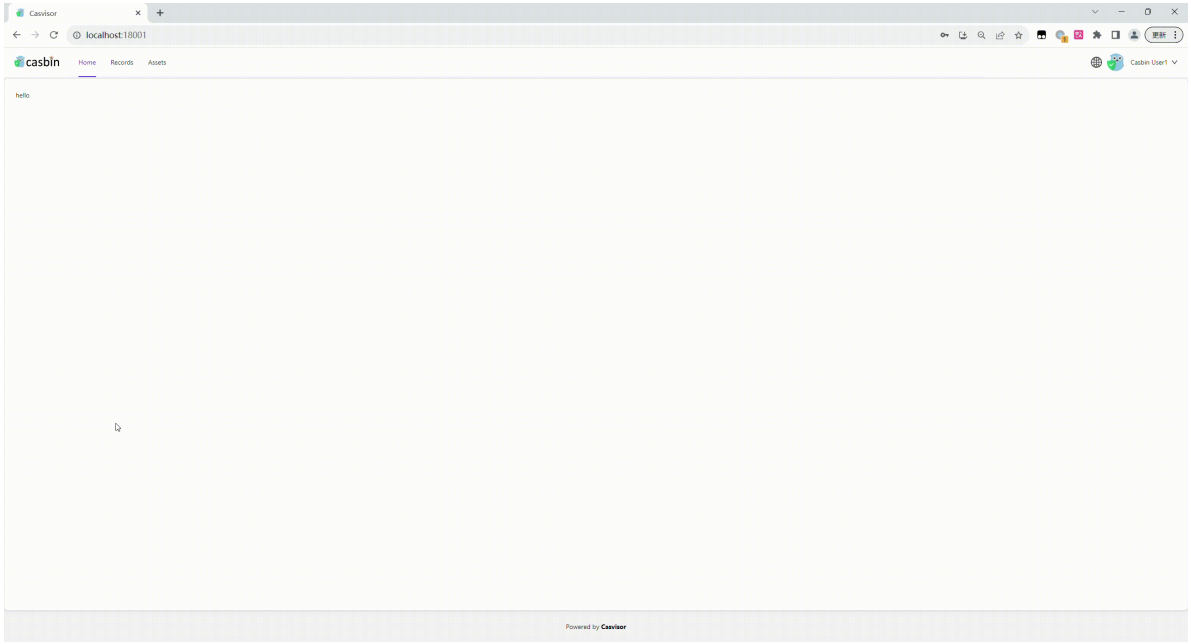
1. Start Guacamole Server

```
docker run --name guacd -d -p 4822:4822 guacamole/guacd
```

2. Add a new asset, set protocol to `vnc`



3. Connect to your asset by clicking the `connect` button

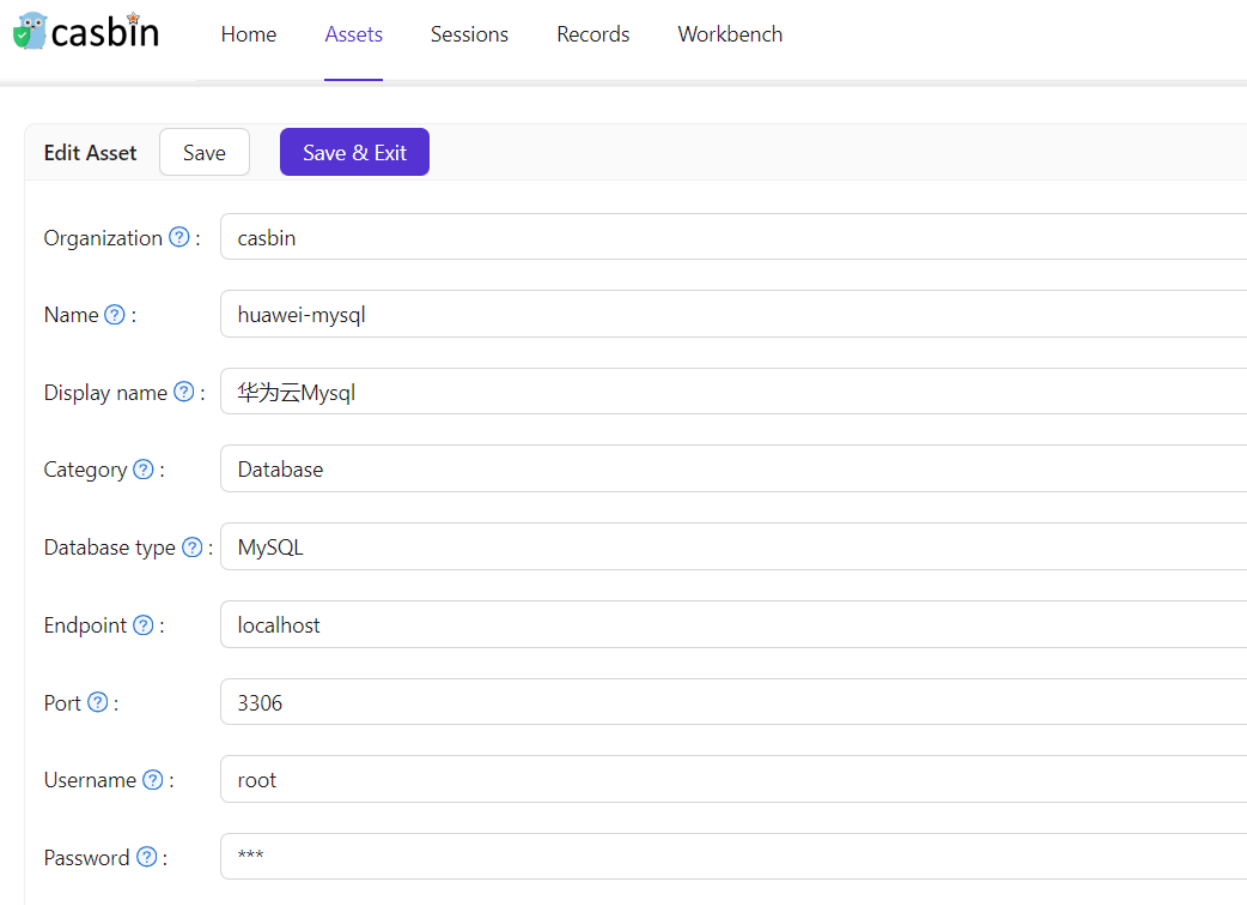


Database

The section will tell you how to add a database asset and connect to your database.

Config database asset

1. In asset list page, click `Add` button to add a new asset.
2. Select `Database` in the `Category` dropdown list.
3. Select the database type in the `Database type` dropdown list, such as `MySQL`, `MariaDB`, `PostgreSQL`, `Microsoft SQL Server`, `Redis`, `MongoDB`.
4. Fill in the required fields that connect to your database.



Connect to database

- In asset list page, click the `Connect` button to connect to your database.
- In the workbench, click the database asset to connect to your database.

Intranet

The section will tell you how to config an intranet asset and connect to your intranet asset. Casvisor uses NAT traversal technology to connect to your intranet asset. You need to deploy a Casvisor agent in your intranet machine. After the agent is connected to the Casvisor server, you can both connect to the intranet RDP asset by Casvisor web UI and other tools like mstsc.

Config Casvisor server

If you want start the NAT traversal service, you need to add `gatewayEndpoint` in `conf/app.conf`.

- The host is your Casvisor server public IP or domain.
- The port is the port that listens to the Casvisor agent connection request.

For example:

```
gatewayEndpoint = door.casvisor.com:19100
```

Config intranet asset

Fill in the required fields that connect to your intranet asset.

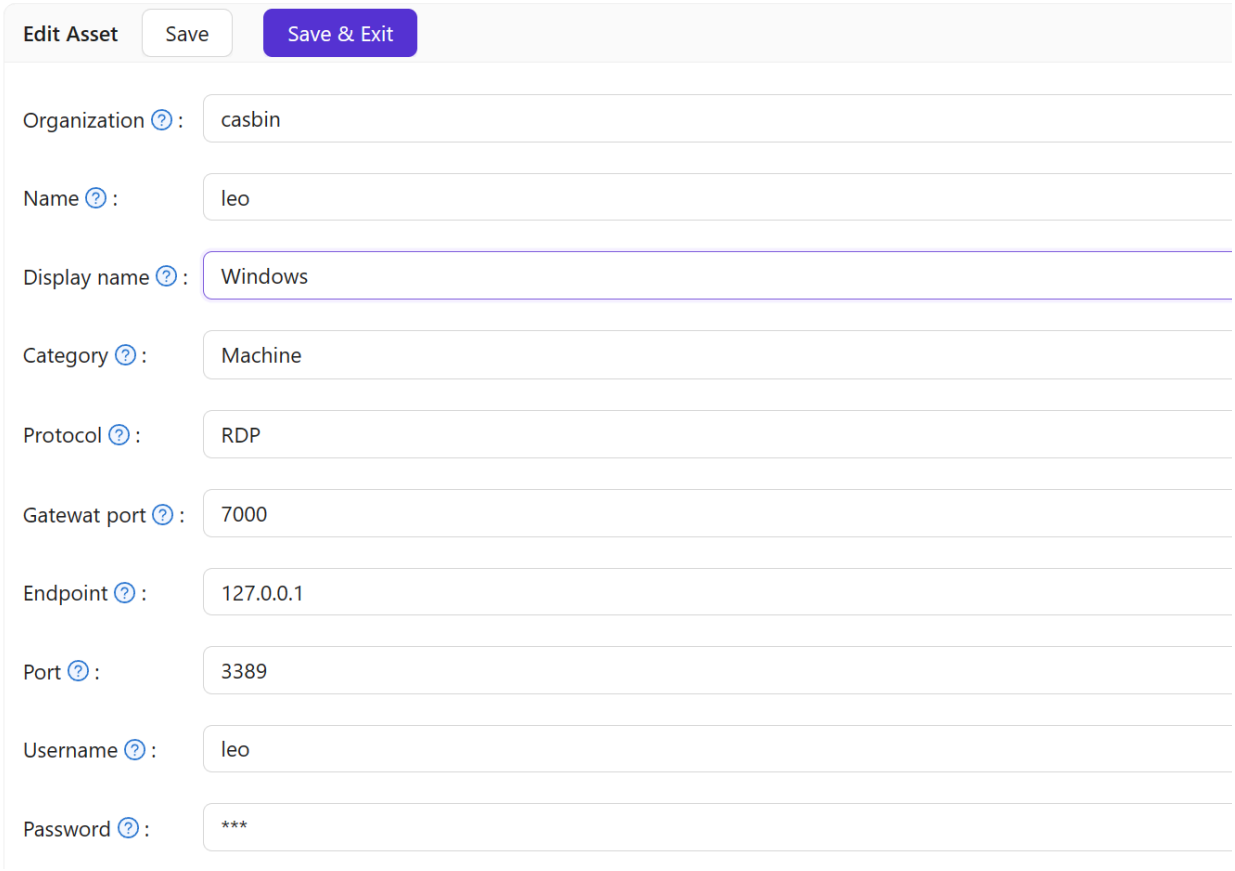
- Name: The **hostname** of the intranet machine. When the casvisor starts, it connects to the database to check whether the current hostname is the name of an asset, and if so, it starts in agent mode. So, make sure to fill in the correct hostname.

- Category: The category of the asset, select `Machine`.
- Protocol: The protocol of the asset, select `RDP`.
- Gateway port: The port in the Casvisor server that listens to the user's connection request.

For example, fill in Gateway port with `7000`. Then the user can connect to the intranet machine by:

```
ssh <Username>@<Casvisor server public IP> -p 7000
```

- Endpoint: Unlike public assets, the endpoint needs to be filled in with the application address of the private network. If you want to use SSH or RDP to connect to the host, enter 127.0.0.1
- Port: The port of the application in the intranet machine. Port 3389 is used by RDP and 22 by SSH by default. If you have a custom port, fill in your custom port.
- Username: The username of the application.
- Password: The password of the application.



Deploy Casvisor agent

You need to deploy a Casvisor on the host to which you are connected in the intranet. As mentioned in the previous chapter, when the casvisor starts, it connects to the database to check whether the current hostname is the name of an asset, and if so, it starts in agent mode. So the must configurations in the agent Casvisor `conf/app.conf` are below:

```
httpport = 19000
driverName =
dataSourceName =
dbName =
gatewayEndpoint = door.casvisor.com:19100
```

It is **recommended** that you can copy the same configuration file `conf/app.conf` in the server Casvisor directly.

Commands

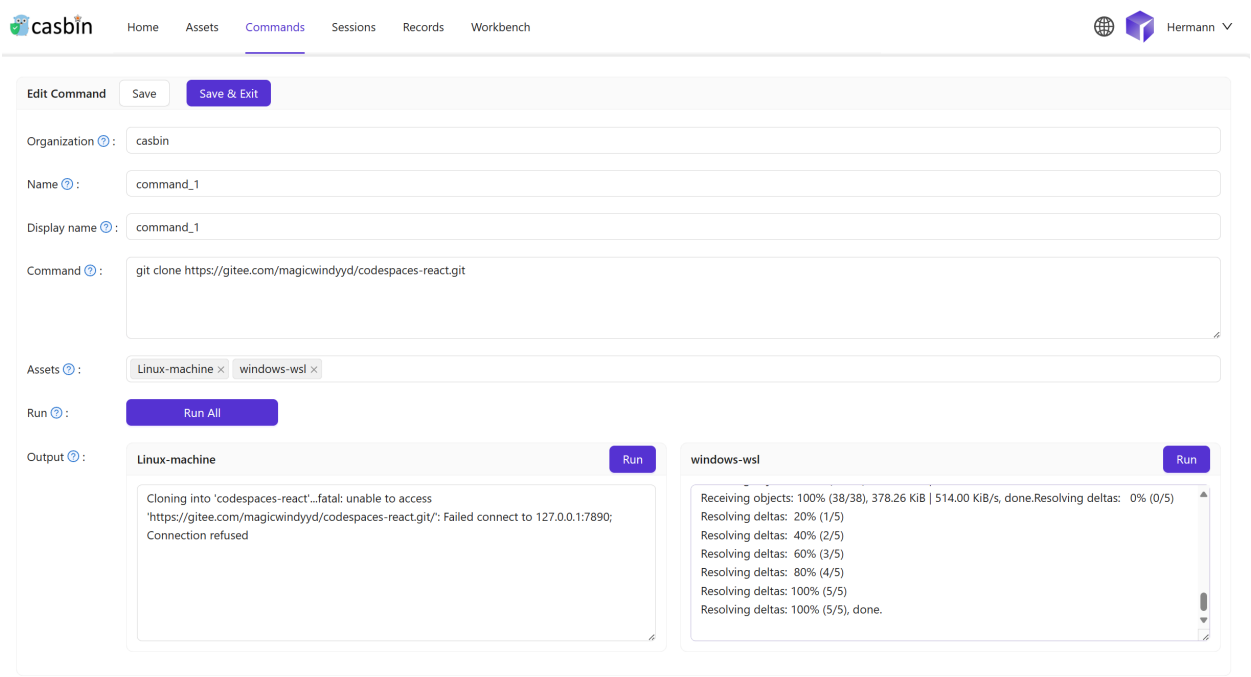
📄 Command

Executing commands on multiple machines via SSH

Command

Overview

Casvisor provides a command feature that allows you to execute commands on multiple machines via SSH. This feature is useful for managing multiple machines at the same time. You can execute commands on multiple machines at the same time, and the results will be displayed in the web UI. Casvisor also exposes API for executing commands.



Usage

1. Click the `add` button in the command list page to add a new command.
2. Input the command in the command input box, if you want to execute multiple commands, you can input commands in multiple lines.

3. Select the assets of SSH protocol that you want to execute the command on.
4. Click the `Run All` button to execute the command on all selected assets. Or click the `Run` button to execute the command on a single asset.

API

Casvisor provides an API for executing commands. The API is as follows:

```
Get /api/get-exec-output
Params:
- id: The id ( owner/name ) of the command
```

Response:

```
{
    "code": 200,
    "msg": "success",
    "data": {
        "<asset name1>": "output",
        "<asset name2>": "output"
    }
}
```