JumpServer HA-cluster configuration

JumpServer (including Community Edition) fully supports HA clustering without any restrictions. In this article, I will show how it works.

Why is an HA cluster needed for JumpServer

An HA (High Availability) cluster for JumpServer is necessary to ensure high system availability and minimize downtime. It enables:

1. **Avoiding downtime**: If one cluster node fails, another continues to handle user requests.
2. **Ensuring fault tolerance**: Clustering allows the system to automatically switch to available nodes in case of errors.
3. **Improving performance**: Load is distributed across multiple nodes, enhancing system responsiveness under high user demand.
4. **Increasing data storage reliability**: Using shared resources like MySQL and Redis with clustering support minimizes the risk of data loss.
5. **Scalability**: The cluster can be easily expanded by adding more nodes to handle more users and tasks.

This architecture is particularly important for organizations where JumpServer is used as a critical access and control system.

Components of the JumpServer Cluster

Nodes/JumpServer Nodes - Core cluster nodes with JumpServer installed. Each server does not store "useful" data, making it possible to clone, copy, delete, or add nodes as needed.

Database MySQL/PostgreSQL - The main DBMS for storing all JumpServer data, including system settings, device parameters, user accounts, and passwords for target systems. By default, it also stores session text logs such as SSH commands, SQL queries, and keyboard input in RDP sessions.
By default, JumpServer creates and uses PostgreSQL within a container on the same server where JumpServer is installed.

Redis Database - An auxiliary database for caching. It can be a shared database for the entire cluster or separate databases for each cluster node.
By default, JumpServer creates and uses Redis within a container on the same server where JumpServer is installed.

Video Recordings Storage - By default, it stores session recordings in the folder **$folder/core/data/media**, where **$folder** is specified in the main configuration file (default: **VOLUME_DIR=/data/jumpserver**). The product's web interface allows setting up external video storage: **SFTP, S3, Ceph, MinIo, and others.**

Command Logs Storage - Logs are stored in the main database by default. The web interface allows configuring log storage in **Elasticsearch**.

Load Balancer - Usually based on HAProxy, but other options can be used.

Cluster Architecture

Typically, a JumpServer cluster consists of two or more cluster nodes that:

- Are connected to a shared MySQL/PostgreSQL database (or cluster).
- Are connected to a shared Redis database (or each has its own).
- Share a common storage for video recordings:
  - A common folder **$folder/core/data/** (usually implemented via an NFS server).
  OR
  - External storage for session recordings (e.g., SFTP, S3, Ceph, MinIo).
- Share a common storage for command logs:
  - Based on a shared database (default).
  OR
  - Based on Elasticsearch.
- Use a load balancer (optional).

Example of creating a JumpServer HA Cluster with two nodes

Example of creating a JumpServer cluster
- With a shared MySQL database
- With a shared Redis database
- With a shared folder for storing video recordings **$folder/core/data/** using an NFS server.

For this, we need:

1. **Server with NFS, MySQL, Redis**:
   - 4 CPUs, 8 GB of RAM.
2. **JumpServer Node1**:
   - 4 CPUs, 8 GB of RAM, 100 GB of free disk space.
3. **JumpServer Node2**:
   - 4 CPUs, 8 GB of RAM, 100 GB of free disk space.
4. **HAProxy Server** (or another load balancer).

---

1. Preparing the server with NFS, MySQL, and Redis

- **Server**: Ubuntu 22.04, IP: `10.10.50.10`

Installing and configuring NFS

Commands may vary for different Linux versions, but generally, you need to create a shared folder:

```
sudo apt install nfs-kernel-server
sudo mkdir -p /data
```

```
sudo chown -R nobody:nogroup /data/
sudo chmod 777 /data/
sudo nano /etc/exports
```

Add the following line to the `/etc/exports` file:

```
/data 10.10.50.10/24(rw,sync,no_subtree_check)
```

Apply the settings and restart the NFS service:

```
sudo exportfs -a
sudo systemctl restart nfs-kernel-server
```

Installing and configuring MySQL

Instructions depend on the OS version. To create a database and user, run the following commands:

```
mysql -uroot
mysql> create database jumpserver default charset 'utf8';
mysql> set global validate_password_policy=LOW;
mysql> create user 'jumpserver'@'%' identified by 'KXOeyNgDeTdpeu9q';
mysql> grant all on jumpserver.* to 'jumpserver'@'%';
mysql> flush privileges;
mysql> exit;
```

Don’t forget to configure the firewall to open the MySQL port (`3306`).

Installing and configuring Redis

Instructions depend on the OS version. After installing Redis, run the following commands:

```
sed -i "s/bind 127.0.0.1/bind 0.0.0.0/g" /etc/redis.conf
sed -i "561i maxmemory-policy allkeys-lru" /etc/redis.conf
sed -i "481i requirepass KXOeyNgDeTdpeu9q" /etc/redis.conf
```

This will allow access to Redis with the password `KXOeyNgDeTdpeu9q`. Make sure to use a unique password for your server. Open the port `6379` in the firewall.

2. Installing JumpServer

Installing the first JumpServer node

Mounting the NFS Directory

Install the NFS client, mount the folder, and configure automatic mounting at startup:

```
sudo apt install nfs-common
mkdir -p /opt/jumpserver/core/data
```

```
mount -t nfs 10.10.50.10:/data /opt/jumpserver/core/data
echo "10.10.50.10:/data /opt/jumpserver/core/data nfs defaults 0 0" >> /etc/fstab
```

Configuring JumpServer

Edit the `config-example.txt` file in the installer directory:

```
# Modify the following parameters, leave others as default.
# IMPORTANT: SECRET_KEY must match on all JumpServer nodes, or the data will not decrypt.

VOLUME_DIR=/opt/jumpserver

SECRET_KEY=
BOOTSTRAP_TOKEN=
LOG_LEVEL=ERROR
SESSION_EXPIRE_AT_BROWSER_CLOSE=True

# MySQL

DB_HOST=10.10.50.10
DB_PORT=3306
DB_USER=jumpserver
DB_PASSWORD=KXOeyNgDeTdpeu9q
DB_NAME=jumpserver

# Redis

REDIS_HOST=10.10.50.10
REDIS_PORT=6379
REDIS_PASSWORD=KXOeyNgDeTdpeu9q

# KoKo Lion
SHARE_ROOM_TYPE=redis
REUSE_CONNECTION=False
```

Run the installation:

```
./jmsctl.sh install
```

After the installation is complete, you will receive the following values:

```
SECRET_KEY=kWQdmdCQKjaWlHYpPhkNQDkfaRulM6YnHctsHLlSPs8287o2kW
BOOTSTRAP_TOKEN=KXOeyNgDeTdpeu9q
```

Installing the Second JumpServer Node

Install the NFS client and mount the folder just as on the first node.

When editing the JumpServer configuration file, **fill in the values for `SECRET_KEY` and `BOOTSTRAP_TOKEN`** obtained after installing the first node:

```
VOLUME_DIR=/opt/jumpserver

SECRET_KEY=kWQdmdCQKjaWlHYpPhkNQDkfaRulM6YnHctsHLlSPs8287o2kW
BOOTSTRAP_TOKEN=KXOeyNgDeTdpeu9q
LOG_LEVEL=ERROR
SESSION_EXPIRE_AT_BROWSER_CLOSE=True

# MySQL

DB_HOST=10.10.50.10
DB_PORT=3306
DB_USER=jumpserver
DB_PASSWORD=KXOeyNgDeTdpeu9q
DB_NAME=jumpserver

# Redis

REDIS_HOST=10.10.50.10
REDIS_PORT=6379
REDIS_PASSWORD=KXOeyNgDeTdpeu9q

# KoKo Lion
SHARE_ROOM_TYPE=redis
REUSE_CONNECTION=False
```

And run the installation:

```
./jmsctl.sh install
```

Result

After completing the setup, you will have two JumpServer nodes sharing one MySQL/Redis server and NFS storage. You can use any of the nodes to access target devices or configure HAProxy to automatically redirect users to an active node.

Версия #1
Сергей Попцов создал 23 января 2025 11:52:28
Сергей Попцов обновил 23 января 2025 12:03:42