Creating a high availability replica

Increase the fault tolerance of your instance
4 of 5 in learning path

Next: **Using GitHub Enterprise Server with a load balancer**

In this article

In an active/passive configuration, the replica appliance is a redundant copy of the primary appliance. If the primary appliance fails, high availability mode allows the replica to act as the primary appliance, allowing minimal service disruption.

Note: There is a maximum of 8 high availability replicas (both passive and active/geo replicas) allowed for GitHub Enterprise Server.

Creating a high availability replica

1. Set up a new GitHub Enterprise Server appliance on your desired platform. The replica appliance should mirror the primary appliance's CPU, RAM, and storage settings. We recommend that you install the replica appliance in an independent environment. The underlying hardware, software, and network components should be isolated from those of the primary appliance. If you are a using a cloud provider, use a separate region or zone. For more information, see "Setting up a GitHub Enterprise Server instance."
2. Ensure that the new appliance can communicate with all other appliances in this high availability environment over ports 122/TCP and 1194/UDP. For more information, see "Network ports."
3. In a browser, navigate to the new replica appliance's IP address and upload your GitHub Enterprise license.
4. Set an admin password that matches the password on the primary appliance and continue.
5. Click **Configure as Replica**.
6. Under "Add new SSH key", type your SSH key.
7. Click **Add key**.

8. Connect to the replica appliance's IP address using SSH.

```
ssh -p 122 admin@REPLICA_IP
```

9. To generate a key pair for replication, use the `ghe-repl-setup` command with the primary appliance's IP address and copy the public key that it returns.

```
ghe-repl-setup PRIMARY_IP
```

10. To add the public key to the list of authorized keys on the primary appliance, browse to `https://PRIMARY-HOSTNAME/setup/settings` and add the key you copied from the replica to the list.

11. To verify the connection to the primary and enable replica mode for the new replica, run `ghe-repl-setup` again.

```
ghe-repl-setup PRIMARY_IP
```

12. To start replication of the datastores, use the `ghe-repl-start` command.

```
ghe-repl-start
```

> **Warning:** `ghe-repl-start` causes a brief outage on the primary server, during which users may see internal server errors. To provide a friendlier message, run `ghe-maintenance -s` on the primary node before running `ghe-repl-start` on the replica node to put the appliance in maintenance mode. Once replication starts, disable maintenance mode with `ghe-maintenance -u`. Git replication will not progress while the primary node is in maintenance mode.

13. To verify the status of each datastore's replication channel, use the `ghe-repl-status` command.

```
ghe-repl-status
```

Creating geo-replication replicas

This example configuration uses a primary and two replicas, which are located in three different geographic regions. While the three nodes can be in different networks, all nodes are required to be reachable from all the other nodes. At the minimum, the required administrative ports should be open to all the other nodes. For more information about the port requirements, see "Network ports."

For high availability, the latency between the network with the active nodes and the network with the replica nodes must be less than 70 milliseconds. We don't recommend configuring a firewall between the two networks. If latency is more than 70 milliseconds, we recommend cache replica nodes instead. For more information, see "Configuring a repository cache."

1. Create the first replica the same way you would for a standard two node configuration by running `ghe-repl-setup` on the first replica.

```
(replica1)$ ghe-repl-setup PRIMARY_IP
```

```
(replica1)$ ghe-repl-start
```

2. Create a second replica and use the `ghe-repl-setup --add` command. The `--add` flag prevents it from overwriting the existing replication configuration and adds the new replica to the configuration.

```
(replica2)$ ghe-repl-setup --add PRIMARY_IP
(replica2)$ ghe-repl-start
```

3. By default, replicas are configured to the same datacenter, and will now attempt to seed from an existing node in the same datacenter. Configure the replicas for different datacenters by setting a different value for the datacenter option. The specific values can be anything you would like as long as they are different from each other. Run the `ghe-repl-node` command on each node and specify the datacenter.

   On the primary:

```
(primary)$ ghe-repl-node --datacenter [PRIMARY DC NAME]
```

   On the first replica:

```
(replica1)$ ghe-repl-node --datacenter [FIRST REPLICA DC NAME]
```

   On the second replica:

```
(replica2)$ ghe-repl-node --datacenter [SECOND REPLICA DC NAME]
```

   Tip: You can set the `--datacenter` and `--active` options at the same time.

4. An active replica node will store copies of the appliance data and service end user requests. An inactive node will store copies of the appliance data but will be unable to service end user requests. Enable active mode using the `--active` flag or inactive mode using the `--inactive` flag.

   On the first replica:

```
(replica1)$ ghe-repl-node --active
```

   On the second replica:

```
(replica2)$ ghe-repl-node --active
```

5. To apply the configuration, use the `ghe-config-apply` command on the primary.

```
(primary)$ ghe-config-apply
```

Configuring DNS for geo-replication

Configure Geo DNS using the IP addresses of the primary and replica nodes. You can also create a DNS CNAME for the primary node (e.g. `primary.github.example.com`) to access

the primary node via SSH or to back it up via `backup-utils`.

For testing, you can add entries to the local workstation's `hosts` file (for example, `/etc/hosts`). These example entries will resolve requests for `HOSTNAME` to `replica2`. You can target specific hosts by commenting out different lines.

```
# <primary IP>      HOSTNAME
# <replica1 IP>     HOSTNAME
<replica2 IP>     HOSTNAME
```

Further reading

- "About high availability configuration"
- "About geo-replication"

Previous
About high availability configuration

Next
Using GitHub Enterprise Server with a load balancer

Legal

© 2023 GitHub, Inc. Terms Privacy Status Pricing Expert services Blog