Cluster network configuration

In this article

About networking for a GitHub Enterprise Server cluster
Network considerations
Configuring a load balancer
Handling client connection information
DNS requirements

A GitHub Enterprise Server cluster requires proper DNS name resolution, load balancing, and communication between nodes.

GitHub determines eligibility for clustering, and must enable the configuration for your instance's license. Clustering requires careful planning and additional administrative overhead. For more information, see "About clustering."

About networking for a GitHub Enterprise Server cluster

Each node in your GitHub Enterprise Server cluster must be able to communicate with all of the other nodes in the cluster over the network. You can review the required ports and protocols for end users, administration, and communication between nodes. To distribute traffic among front-end nodes, GitHub recommends that you configure an external load balancer.

Network considerations

The simplest network design for clustering is to place the nodes on a single LAN. If a cluster must span subnetworks, we do not recommend configuring any firewall rules between the networks. The latency between nodes should be less than 1 millisecond.

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.

- Application ports for end users
- Administrative ports
- Cluster communication ports

Application ports for end users

Application ports provide web application and Git access for end users.

Port	Description	Encrypted
22/TCP	Git over SSH	✓
25/TCP	SMTP	Requires STARTTLS

25/TCP	SMTP	Requires STARTTLS
80/TCP	HTTP	✕ When SSL is enabled this port redirects to HTTPS
443/TCP	HTTPS	✓
9418/TCP	Simple Git protocol port (Disabled in private mode)	✕

Administrative ports

Administrative ports are not required for basic application use by end users.

Port	Description	Encrypted
ICMP	ICMP Ping	✕
122/TCP	Administrative SSH	✓
161/UDP	SNMP	✕
8080/TCP	Management Console HTTP	✕ When SSL is enabled this port redirects to HTTPS
8443/TCP	Management Console HTTPS	✓

Cluster communication ports

If a network level firewall is in place between nodes, these ports will need to be accessible. The communication between nodes is not encrypted. These ports should not be accessible externally.

Port	Description
1336/TCP	Internal API
3033/TCP	Internal SVN access
3037/TCP	Internal SVN access
3306/TCP	MySQL
4486/TCP	Governor access
5115/TCP	Storage backend
5208/TCP	Internal SVN access
6379/TCP	Redis
8001/TCP	Grafana
8090/TCP	Internal GPG access
8149/TCP	GitRPC file server access

8300/TCP	Consul
8301/TCP	Consul
8302/TCP	Consul
9000/TCP	Git Daemon
9102/TCP	Pages file server
9105/TCP	LFS server
9200/TCP	Elasticsearch
9203/TCP	Semantic code service
9300/TCP	Elasticsearch
11211/TCP	Memcache
161/UDP	SNMP
8125/UDP	Statsd
8301/UDP	Consul
8302/UDP	Consul
25827/UDP	Collectd

Configuring a load balancer

We recommend an external TCP-based load balancer that supports the PROXY protocol to distribute traffic across nodes. Consider these load balancer configurations:

- TCP ports (shown below) should be forwarded to nodes running the `web-server` service. These are the only nodes that serve external client requests.
- Sticky sessions shouldn't be enabled.

Warning: When terminating HTTPS connections on a load balancer, the requests from the load balancer to GitHub Enterprise Server also need to use HTTPS. Downgrading the connection to HTTP is not supported.

Handling client connection information

Because client connections to the cluster come from the load balancer, the client IP address can be lost. To properly capture the client connection information, additional consideration is required.

If your load balancer can support it, we strongly recommend implementing the PROXY protocol. When no PROXY support is available, it is also possible to load balance the HTTP and HTTPS ports using the `X-Forwarded-For` header.

Security Warning: When either PROXY support or HTTP forwarding is enabled, it is critical that no external traffic can directly reach the GitHub Enterprise Server appliances. If external traffic is not properly blocked, the source IP addresses can be forged.

Enabling PROXY support on GitHub Enterprise Server

We strongly recommend enabling PROXY support for both your instance and the load balancer.

> **Note:** GitHub Enterprise Server supports PROXY Protocol V1, which is incompatible with AWS Network Load Balancers. If you use AWS Network Load Balancers with GitHub Enterprise Server, do not enable PROXY support.

- For your instance, use this command:

  ```
  ghe-config 'loadbalancer.proxy-protocol' 'true' && ghe-cluster-config-apply
  ```

- For the load balancer, use the instructions provided by your vendor.

PROXY protocol TCP port mappings

Source port	Destination port	Service description
22	23	Git over SSH
80	81	HTTP
443	444	HTTPS
8080	8081	Management Console HTTP
8443	8444	Management Console HTTPS
9418	9419	Git

Enabling X-Forwarded-For support on GitHub Enterprise Server

Use the X-Forwarded-For protocol **only** when the PROXY protocol is unavailable. The `X-Forwarded-For` header only works with HTTP and HTTPS. The IP address reported for Git connections over SSH will show the load balancer IP.

To enable the `X-Forwarded-For` header, use this command:

```
ghe-config 'loadbalancer.http-forward' 'true' && ghe-cluster-config-apply
```

Protocol TCP port mappings for use without PROXY support

Source port	Destination port	Service description
22	22	Git over SSH
25	25	SMTP
80	80	HTTP
443	443	HTTPS
8080	8080	Management Console HTTP

8443	8443	Management Console HTTPS

Configuring health checks

Health checks allow a load balancer to stop sending traffic to a node that is not responding if a pre-configured check fails on that node. If a cluster node fails, health checks paired with redundant nodes provides high availability.

Configure the load balancer to check the following URL.

```
http(s)://HOSTNAME/status
```

The endpoint will return status code `200` (OK) if the node is healthy and available to service end-user requests. For more information, see "Monitoring a high-availability configuration."

> **Note:** When the appliance is in maintenance mode, the `https://HOSTNAME/status` URL will return status code `503` (Service Unavailable). For more information, see "Enabling and scheduling maintenance mode."

DNS requirements

DNS lookups for the GitHub Enterprise Server hostname should resolve to the load balancer. We recommend that you enable subdomain isolation. If subdomain isolation is enabled, an additional wildcard record (`*.HOSTNAME`) should also resolve to the load balancer. For more information, see "Enabling subdomain isolation."