LOADBALANCER

MANUALS

Configuring Endurance using Bruno

Version 1.0.0 Revision 1.0.2

Table of Contents

1. About this Guide

This guide provides an introduction to configuring load balanced services using Bruno. It demonstrates how API requests can be used to create the simple layer 7 test environment presented in the Endurance Quick Start Guide.

2. Introduction

API calls can be formed and sent using any RESTful compliant application or API test client. This enables Endurance to be configured without using the WebUI.

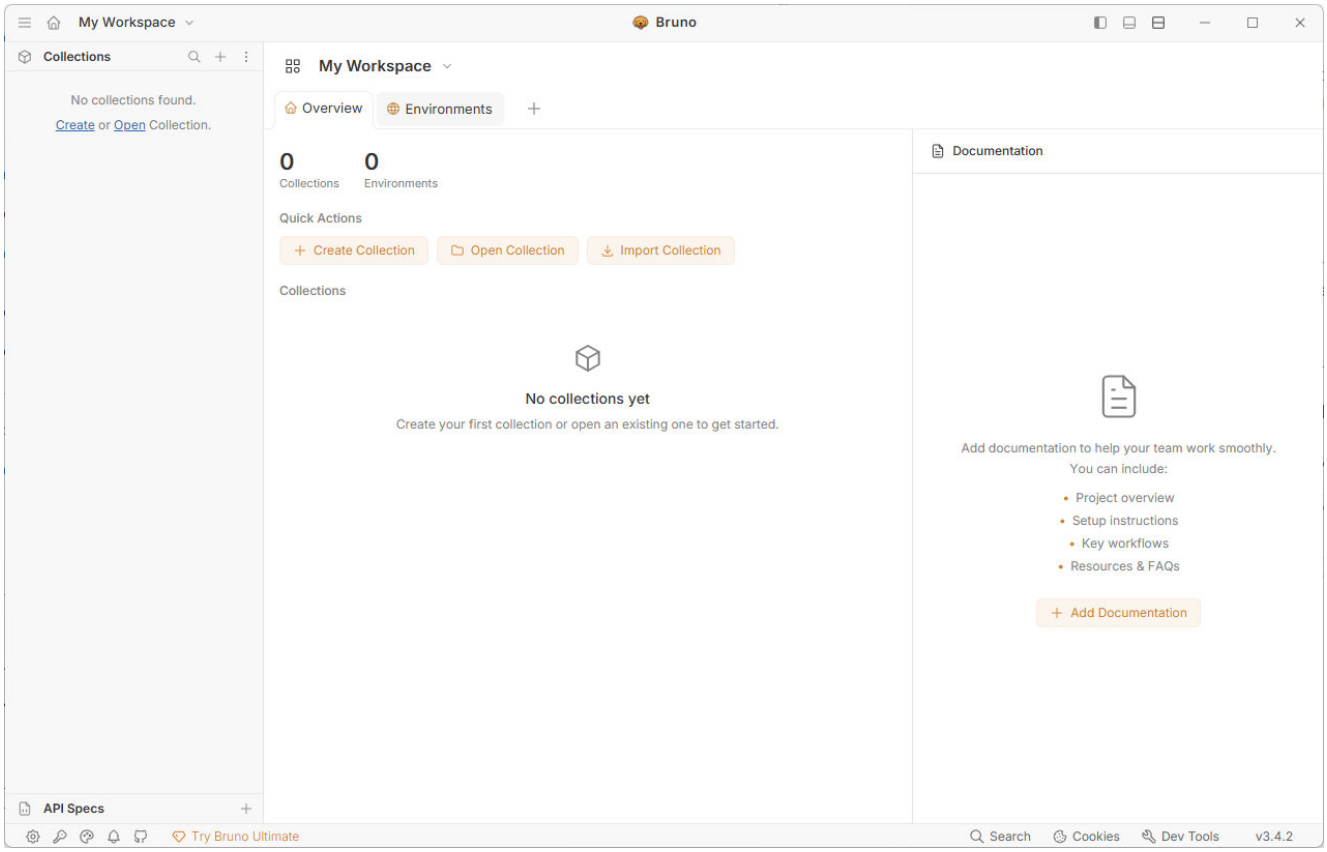
3. Getting Started with Bruno

Bruno is a Git-integrated, fully offline open-source API client that is available for Linux, Windows and MAC and can be downloaded from here. Bruno uses a "collection" to define the API environment and store the API requests. Collections must be run in order to execute the requests.

3.1. Download & Import the Bruno Collection for the Simple Test Environment

The Bruno collection for the test environment can be downloaded here.

Once downloaded, run Bruno (the screenshots below show Bruno v3.4.2).



Under *Quick Actions*, select *Import Collection*.

In the pop-up, select **choose file(s)**, then browse to and select the file downloaded previously and click **Open**.

Under *Location*, click **Browse**, then choose a location to store the new collection and click **Select Folder**.

Click **Import** to import the collection.

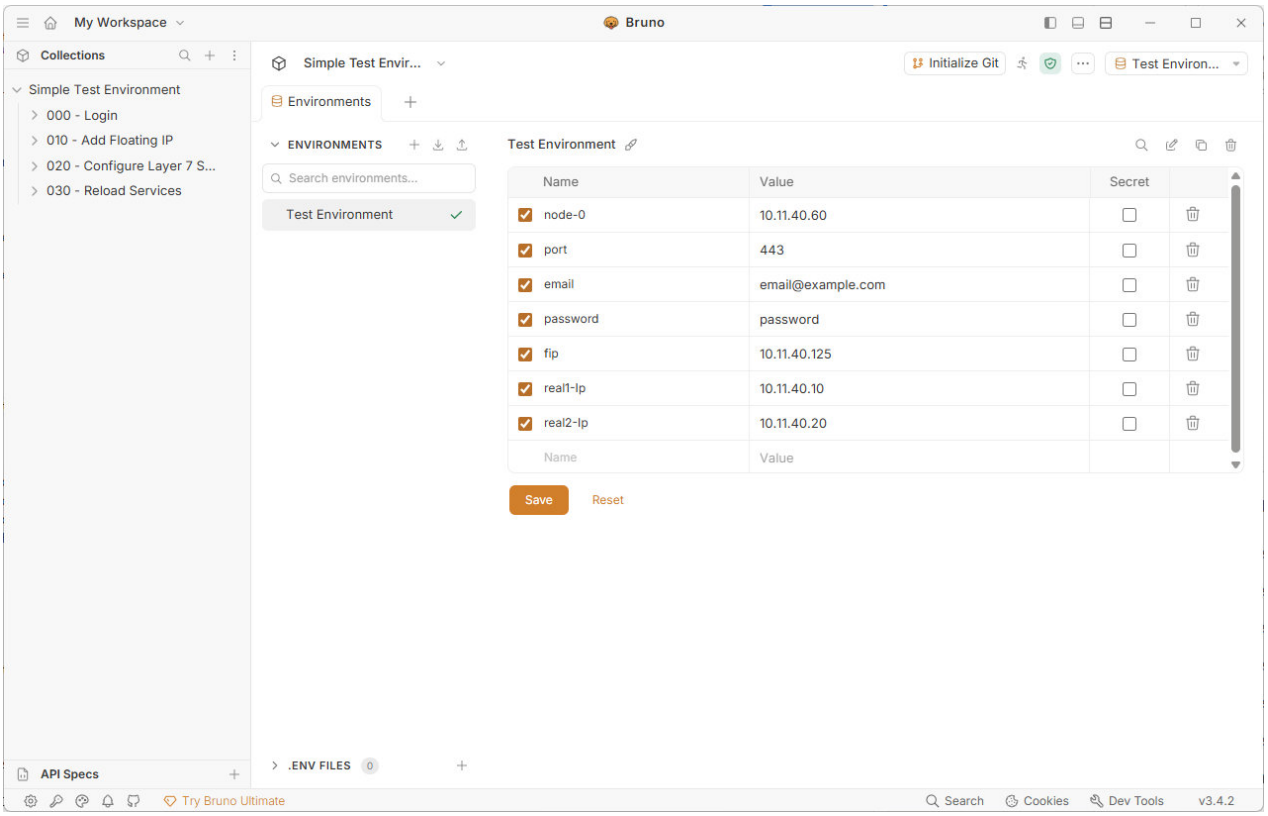
Under *Collections*, click **Simple Test Environment** to select the collection.

3.2. Environment Variables

The Test Environment collection uses several environment variables in the API requests, these must be modified to suit your environment.

To edit the environment variables:

1. Click on *No Environment* in the top right corner, then select **Test Environment**.
2. Select *Test Environment* in the top right corner and select **Configure**.



- Change the IP address for *node-0* to match your appliance.
- Change *email* to match the email address you use to login to your appliance.
- Change *password* to match the password you use to login to your appliance.
- Change the IP address for *fip* to the IP address you'd like to use for the Virtual Service.
- Change the IP address for *real1-lp* to the address of your first Real Server.
- Change the IP address for *real2-lp* to the address of your second Real Server.

- Click **Save**.

3.3. Command Overview

The table below provides a summary of the API requests that are used to configure the simple test environment:

Command Group	Request	Purpose
Login	Login	Login and obtain access token
Add Floating IP	Get Management Group	Determine the groupId for the management subnet
	Add Floating IP	Add the floating IP to the management subnet
Configure Layer 7 Service	Add Real Server 1	Add first real server
	Add Real Server 2	Add second real server
	Add Backend	Add the backend for the layer 7 service
	Add Default ACL	Add the default ACL to link frontend and backend
	Add Frontend	Add the frontend for the layer 7 service
Reload Services	Reload Required Services	Restart services to apply new settings (HAProxy and Nftables)

3.4. Command Details

The following sections provide full details of all API requests.

3.4.1. Login

Request

```
POST https://{{node-0}}:{{port}}/api/v1/token
```

Body

```
{
  "email": "{{email}}",
  "password": "{{password}}"
}
```

Post Response Script

```
bru.setEnvVar("token", res.body.data.token);
bru.setEnvVar("localnodeid", res.body.data.appliance_id);
```

3.4.2. Get Management Group

Request

```
GET https://{{node-0}}:{{port}}/api/v1/ipgroups/
```

Auth

```
Bearer token : {{token}}
```

Post Response Script

```
var groupId = ""
for (let i = 0; i < res.body.data.length; i++) {
    if (res.body.data[i].name == "Management") {
        groupId = res.body.data[i].id
    }
}
bru.setEnvVar("groupId", groupId);
```

3.4.3. Add Floating IP

Request

```
POST https://{{node-0}}:{{port}}/api/v1/fips
```

Auth

```
Bearer token : {{token}}
```

Post Response Script

```
bru.setEnvVar("fipId", res.body.data.id);
```

3.4.4. Add Real Server 1

Request

```
POST https://{{node-0}}:{{port}}/api/v1/haproxy/rips
```

Body

```
{
  "name": "RS-01",
  "ipDomain": "{{real1-Ip}}",
  "port": 80,
  "weight": 100,
  "minconns": 0,
  "maxconns": 0,
  "enabledCount": false
}
```

Auth

```
Bearer token : {{token}}
```

Post Response Script

```
bru.setEnvVar("real1-Id", res.body.data.id);
```

3.4.5. Add Real Server 2

Request

```
POST https://{{node-0}}:{{port}}/api/v1/haproxy/rips
```

Body

```
{
  "name": "RS-02",
  "ipDomain": "{{real2-Ip}}",
  "port": 80,
  "weight": 100,
  "minconns": 0,
  "maxconns": 0,
  "enabledCount": false
}
```

Auth

```
Bearer token : {{token}}
```

Post Response Script

```
bru.setEnvVar("real2-Id", res.body.data.id);
```

3.4.6. Add Backend

Request

```
POST https://{{node-0}}:{{port}}/api/v1/haproxy/backends
```

Body

```
{
  "name": "BE-01",
  "mode": "tcp",
  "ripIds": ["{{real1-Id}}","{{real2-Id}}"],
  "balanceMode": "leastconn",
  "persistenceMode": "none",
  "healthCheck": "connectToPort",
  "healthCheckPort": 80,
  "healthCheckRequestToSend": "/",
  "healthCheckResponseExpected": "",
  "encryptToBackend": false,
  "enableGzipCompression": false,
  "headers": "",
  "httpPipeline": "httpKeepAlive",
  "httpPretendKeepalive": false,
```

```
  "httpAcceptInvalidResponse": false,
  "httpRequestTimeout": true,
  "tunnelTimeout": "1h",
  "TCPKeepalive": false,
  "connectionRedispatch": true,
  "abortOnClose": false,
  "slowStartTime": "3s",
  "backendSourceAddress": "",
  "enableServerConnTimeout": true,
  "serverTimeout": "43s",
  "enableFeedbackAgent": false,
  "feedbackAgentPort": 3333,
  "feedbackAgentInterval": "2s",
  "fallbackPersistence": false,
  "enableTProxy": false,
  "sendProxyProtocol": "off"
}
```

Auth

```
Bearer token : {{token}}
```

Post Response Script

```
bru.setEnvVar("backend-id", res.body.data.id);
```

3.4.7. Add Default ACL

Request

```
POST https://{{node-0}}:{{port}}/api/v1/haproxy/acls
```

Body

```
{
  "conditionType":"default",
  "condition":"eq",
  "redirectType":"backend",
  "redirectLocation":"{{backend-id}}"
}
```

Auth

```
Bearer token : {{token}}
```

Post Response Script

```
bru.setEnvVar("acl-default-id", res.body.data.id);
```

3.4.8. Add Frontend

Request

```
POST https://{{node-0}}:{{port}}/api/v1/haproxy/frontends
```

Body

```
{
  "name": "Web-Cluster",
  "ports": [80],
  "mode": "tcp",
  "fipID": "{{fipId}}",
  "aclIds": ["{{acl-default-id}}"],
  "SSLEnable": false,
  "SSLTLSv130rttEnable": false,
  "tcpKeepalive": false,
  "acceptProxyProtocol": false,
  "SSLMinVer": "TLSv1.2",
  "SSLMaxVer": "TLSv1.3",
  "httpsForceToRedirect": false,
  "httpsRedirectCode": 301,
  "hstsEnable": false,
  "hstsTimeout": 6,
  "sslCerts": [],
  "maximumConnections": 40000,
  "enableClientConnTimeout": false,
  "clientTimeout": "42s",
  "httpPipeLine": "httpKeepAlive",
  "httpAcceptInvalidRequest": false,
  "httpRequestTimeout": true,
  "httpPretendKeepalive": false,
  "setXForwardedforHeader": false
}
```

Auth

```
Bearer token : {{token}}
```

Post Response Script

```
bru.setEnvVar("frontend-id", res.body.data.id);
```

3.4.9. Reload Required Services

Request

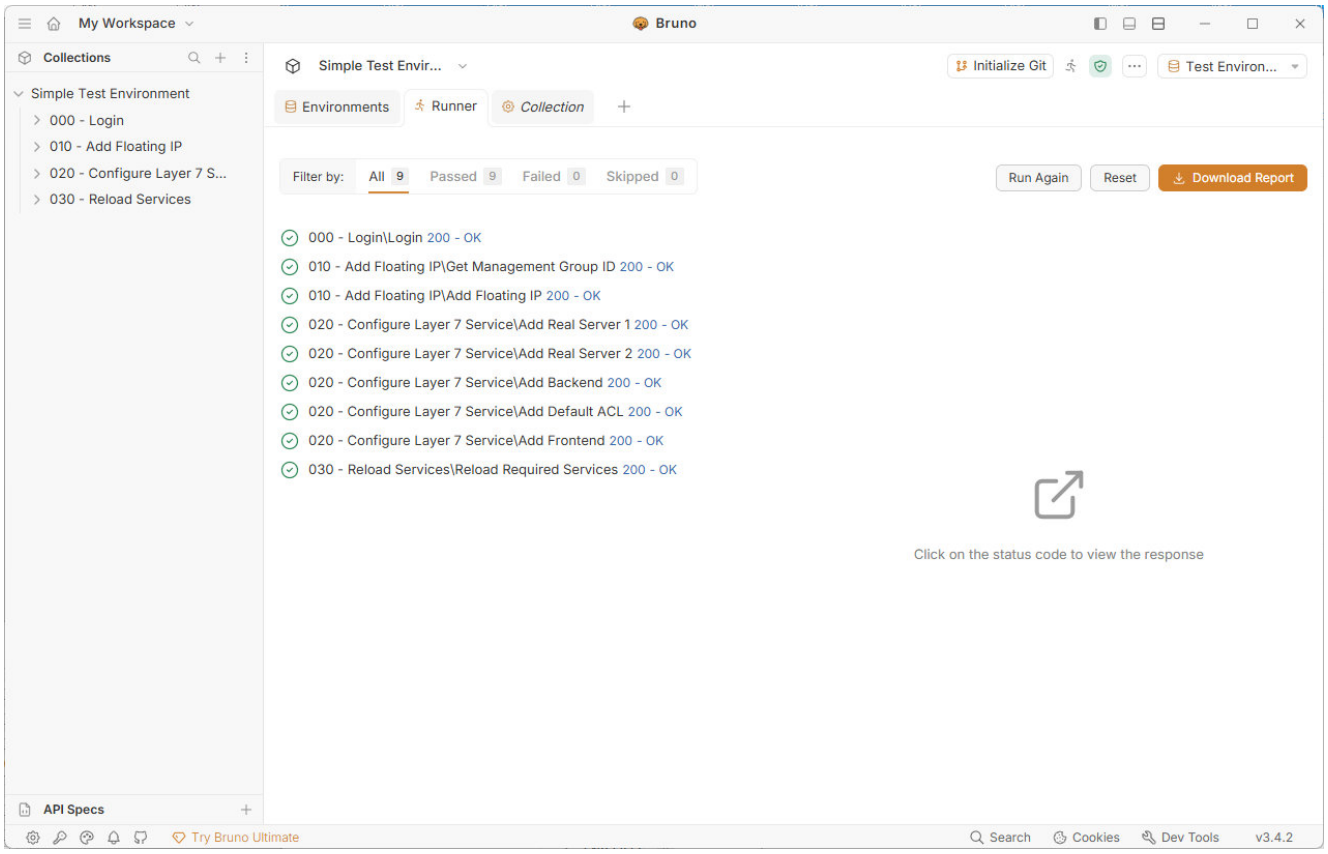
```
PUT https://{{node-0}}:{{port}}/api/v1/{{localnodeid}}/services/apply
```

Auth

```
Bearer token : {{token}}
```

3.5. Run the Collection

Under *Collections*, right click **Simple Test Environment**, select **Run** then click **Run 9 Requests**.



The above image shows that all requests have completed successfully. This confirms that the simple test environment has been created.

4. Viewing & Testing the Configuration

For details on viewing/modifying and testing the configuration, please refer to the Endurance Quick Start Guide.

5. Technical Support

If you need assistance or have any questions, please don't hesitate to contact our highly experienced Support Engineers. They are on hand to help 24 hours a day, 365 days a year.

Unlimited support is available for Loadbalancer.org products as follows:

- During any Beta product testing program.
- During the cover period of any active appliance support agreement. To purchase a support package, please contact: sales@loadbalancer.org.
- During the Enterprise VA free trial period. To download a trial, please go to Virtual Appliance Trial.
- As specified in the relevant ADC Portal subscription level. For details, please refer to the Loadbalancer Portal Product Datasheet.

5.1. Accessing Support

If you need assistance or have any questions, don't hesitate to contact us.

5.1.1. Using Online Chat from Endurance

Click the Intercom icon in the bottom right-hand corner to use the online chat function to communicate with our support team.

5.1.2. Create a Support Ticket from Endurance

To create a support ticket:

1. Navigate to *Support > Contact us*.
2. Click the raise support ticket option.
3. Enter the required details and click **Open Ticket**.

5.1.3. Email Us

To raise a support ticket via email, send a message to the support team : support@loadbalancer.org.

5.2. Technical Support Download (TSD)

The TSD is a compressed archive containing all log files and configuration files for the appliance. This can be attached with your email when first requesting support - this will help to minimize the time it takes to resolve your issue.

To create a TSD:

1. Navigate to *Support > Technical support download*.
2. Click **Download**.
3. Download the TSD from your browser.

> **Note** If the TSD is too large to send via email, our Support Engineers can supply details of our upload server.

5.3. Remote Support

When required, a remote session can be arranged to assist with trouble shooting. **Zoom** is the preferred tool and is used where possible.

LOADBALANCER

Visit us: www.loadbalancer.org

Phone us: +44 (0)330 380 1064

Phone us: +1 833 274 2566

Email us: info@loadbalancer.org

Follow us: @loadbalancer.org

About Loadbalancer.org

Loadbalancer.org's mission is to ensure that its clients' businesses are never interrupted. The load balancer experts ask the right questions to get to the heart of what matters, bringing a depth of understanding to each deployment. Experience enables Loadbalancer.org engineers to design less complex, unbreakable solutions - and to provide exceptional personalized support.