JUMPCLOUD

**1.0**

Password Hashing Application

**Document Revision 1.0**

**August 8, 2019**

**Primary Author:   
Christian Mueller**

Table of Contents

[1. Overview 3](#_Toc15840466)

[2. Assumptions for this Feature 3](#_Toc15840467)

[3. Limitations of this Feature 3](#_Toc15840468)

[4. Requirements 3](#_Toc15840469)

[5. Test Approach 4](#_Toc15840470)

**Version History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Author** | **Date** | **Description** |
| 1.0 | Christian Mueller | August 4, 2019 | Initial Test Plan |
|  |  |  |  |
|  |  |  |  |

# Overview

JumpCloud has implemented a password hashing application in Golang

and we have intentionally left bugs in it. The assignment is to write the test cases needed to

test the application, explain your choices for coverage, execute the test cases and report the

bugs you find. Deliverables should be submitted along with a README in a GitHub repo that

you share with us. The style, depth, scope and type of tests and bug reports you write are up to you. Use them to demonstrate your style and strengths

# Assumptions for this Feature

The application works as described but has been left with bug to be found through testing.

Application is available at <https://s3.amazonaws.com/qa-broken-hashserve/broken-hashserve.tgz>

The broken-hashserve.tgz archive contains binaries for Linux, Windows & Mac OS X

operating systems.

You must set a PORT environment variable before executing the application.

Supported following operating systems:

* Ubuntu 16.04
* Mac OS X - Sierra, High Sierra
* Windows

# Limitations of this Feature

* Not all Curl commands are supported
* Port must be set
* Does not answer on other or unspecified ports

# Requirements

* When launched, the application should wait for http connections.
* It should answer on the PORT specified in the PORT environment variable.
* It should support three endpoints:

○ A POST to /hash should accept a password. It should return a job identifier

immediately. It should then wait 5 seconds and compute the password hash.

The hashing algorithm should be SHA512.

○ A GET to /hash should accept a job identifier. It should return the base64

encoded password hash for the corresponding POST request.

○ A GET to /stats should accept no data. It should return a JSON data structure

for the total hash requests since the server started and the average time of a hash

request in milliseconds.

● The software should be able to process multiple connections simultaneously.

● The software should support a graceful shutdown request. Meaning, it should allow any

in-flight password hashing to complete, reject any new requests, respond with a 200 and

shutdown.

● No additional password requests should be allowed when shutdown is pending.

# Test Approach

The tests to be executed were developed to exercise the requirements and limitations of the product.

Negative tested was added in some instance to ensure the product will behave accordingly.

Manual testing and scripts were used to exercise the features and functionality of the product.