Optimize 25 Testing at Scale: Scaling



Prepared by

**Zachery Olson**

Solution Architect

Contributors

**List of Contributors**

Revision and Signoff Sheet

Change Record

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Author | Version | Change Reference |
| 2/22/2023 | zacolson | 1.0 | Initial draft for review/discussion |
|  |  |  |  |
|  |  |  |  |

Reviewers

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Version Approved | Position | Date |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

Table of Contents

[1 Introduction 5](#_Toc128658043)

[1.1 Test Users 5](#_Toc128658044)

[1.1.1 Test User Example 6](#_Toc128658045)

[1.2 Load Test Engines 7](#_Toc128658046)

[1.3 Service Protection API Limits 7](#_Toc128658047)

[1.3.1 Number of requests 8](#_Toc128658048)

[1.3.2 Execution time 8](#_Toc128658049)

[1.3.3 Number of concurrent requests 8](#_Toc128658050)

[1.4 Resources 9](#_Toc128658051)

1. Introduction

Scaling is a process of increasing a machine or instances capacity to handle a workload, ensure stability, and improve performance. There are two methods of scaling: **Horizontal** and **Vertical**.

**Horizontal scaling** (scaling out) is the addition of nodes to your infrastructure to meet the required demands. An example of horizontal scaling is adding machines to load balance a workload.

**Vertical scaling** (scaling up) is adding additional resources to a single node. An example of vertical scaling is adding more RAM or CPU resources to a server to increase its processing power.

Virtual users are a key concept of load testing. A **virtual user** is the equivalent of an independent test run, and multiple virtual users are configured to simulate concurrent test executions. Apache JMeter refers to virtual users as threads.

When testing components of the power platform at a high scale, it is recommended to scale out the volume of traffic generated by virtual users across a collection of users that are configured specifically for testing.

* 1. Test Users

Creating multiple Test Users is a method of horizontally scaling your test. Test users allow you to test the impact of your security model, and combat service protection limits that could cause errors while a test is executing.

A test user is an identity that is configured to execute the virtual user traffic when testing at scale. A test user must:

* Be licensed appropriately.
* Disabled for multi-factor authentication.
* Follow a persona or specific security role configuration.

Your testing should be accurately based on your security model.  It is possible that different security configurations will have an impact on performance based on the privileges assigned to a user.

Any tests executed in the context of a user that has the system administrator security role will see better performance.  The information returned to a user with the system administrator security role does not have any role-based security filtering applied.

It is best practice to configure enough test users to cover the personas tested in your load test. For high transaction scenarios, you might configure multiple test users with the same personas to distribute the transaction volume and avoid throttling.

* + 1. Test User Example

Test User 1 – Customer Service Agent

* Uses Case, Account, and Contact tables.
* Customer Service security role:
  + Organization level read privileges to Account and Contact
  + User level update privileges to Case table

Test User 2 – Customer Service Manager

* Uses Case, Account, and Contact tables.
* Customer Service Manager security role:
  + Organization level read privileges to Case, Account and Contact
  + Organization level update privileges to Case, Account and Contact

Test User 3 – Quality Control

* Case Report security role:
  + Organization level read privileges to Case, Account and Contact

**Assume the following scenario:**

The Customer Service team consists of:

* 100 Customer Service Agents
* 7 Customer Service Managers
* Some managers also fulfil the Quality Control responsibilities.

The team of customer service agents handle 250 cases on average during peak hours. Customer service managers review and update case information based on specific metrics, averaging 1 case per manager during peak hours. Quality control runs 1 report per day to analyze the previous day’s case metrics.

In the above example, there is more agent transaction volume than manager or quality control volume. Due to the higher volume at peak usage, the Customer Service Agent scenario requires multiple test users to distribute the virtual user volume and avoid throttling.

* 1. Load Test Engines

Azure Load Testing executes a load test using test engines. A **test engine** abstracts the required infrastructure for running a high-scale load test. The test engines run the Apache JMeter script to simulate virtual users simultaneously accessing your application endpoints. JMeter refers to virtual users as threads.

Azure Load Testing recommends that you keep the number of threads in a script below a maximum of 250. When your testing must exceed 250 threads, Azure Load Testing allows you configure the load test by increasing the number of test engine instances to execute the test. Test engines run in parallel.

Consider this formula for the number of virtual users in a load test: VUs = (# threads) \* (# test engine instances)

If your load test calls for simulating user traffic from 1000 virtual users, the following formula could be used:

1000 virtual users = threads (250) \* test engine instances (4)

Refer to the [resources](#Resources) section for more information about high scale load testing.

* 1. Service Protection API Limits

Microsoft Dataverse applies limits to API usage that are intended to ensure consistent availability and performance for everyone. These limits are Service Protection API Limits, and their purpose is to detect when client applications are making extraordinary demands on server resources.

These limits are not intended to impact a normal users’ day-to-day usage of model-driven apps, canvas apps, or power pages. The limits provide a level of protection from random and unexpected increases in request volume that could impact the availability and performance of the platform.

Service protection limits are enforced on each user individually, per web server, and they are based on three different metrics:

* The number of requests sent by a user.
* The combined execution time required to process requests sent by a user.
* The number of concurrent requests sent by a user.

If the platform detects a pattern that exceeds the threshold for one of limits, an error will return indicating that too many requests have been made:

* With the Web API, we return a 429 Too Many Requests error.
* With the Dataverse SDK for .NET, you will get an OrganizationServiceFault error with one of three specific error codes. More information: [Service protection API limit errors returned](https://learn.microsoft.com/en-us/power-apps/developer/data-platform/api-limits?tabs=sdk#service-protection-api-limit-errors-returned)

The specific limits are described below. Refer to the [resources](#Resources) section for more information about Service Protection API Limits, and strategies to avoid the limits.

* + 1. Number of requests

**Description:** The cumulative number of requests made by the user

**Limit per web server:** 6000 within the 5-minute sliding window

* + 1. Execution time

**Description:** The combined execution time of all requests made by the user

**Limit per web server:** 20 minutes (1200 seconds) within the 5-minute sliding window

* + 1. Number of concurrent requests

**Description:** The number of concurrent requests made by the user

**Limit per web server:** 52 or higher

* 1. Resources

Security Roles - <https://learn.microsoft.com/en-us/power-platform/admin/security-roles-privileges>

Azure Load Testing - <https://learn.microsoft.com/en-us/azure/load-testing/overview-what-is-azure-load-testing>

Service Protection API Limits - <https://learn.microsoft.com/en-us/power-apps/developer/data-platform/api-limits?tabs=sdk>

Dataverse API Limits - <https://learn.microsoft.com/en-us/power-apps/maker/data-platform/api-limits-overview>

High-scale load testing - <https://learn.microsoft.com/en-us/azure/load-testing/how-to-high-scale-load>