Understanding and Avoiding Performance Pitfalls with SharePoint online

Sean P. McDonough, Bitstream Foundry LLC
Scott Stewart, Microsoft
Sean P. McDonough
CTO and Owner, Bitstream Foundry LLC
Senior Program Manager at Microsoft

Scott Stewart
OneDrive and SharePoint Engineering
Our Agenda

- SPO’s current implementation
- Service Improvements
- What contributes to poor performance?
- Strategies for good performance
- Gathering diagnostic data and Tools
- Q & A
SPO’s current implementation
Welcome to the farm!

- A farm is the basic unit of SharePoint deployment
- SharePoint Online has Tenants within a Farm i.e. Multi-Tenant
- SPO contains most of the components you’d expect to find on-premises
- Don’t make any assumptions beyond this point ...
Big Differences with SPO

- MinRole deployments
- WFEs – not two or three, but 100’s on average
- High availability (multiple farms)
- Redundancy (across datacenters)
- Bottom line: extremely large scale versus on-premises
Load Testing

- Load testing is futile.
- Any load testing numbers you might get are temporary.
- If you attempt to load test, Microsoft will detect it and throttle your load generators.
Service improvements
Direct Egress to Microsoft

Network devices identify Office 365 traffic

Local egress to Microsoft Network at every location

Bypass network security stack with trusted SaaS traffic
Bringing SharePoint Online to you

Globally distributed edge nodes provide low latency connectivity to the Microsoft Network.
Connectivity

- United States
  - US User
  - edge node
- United Kingdom
  - UK User
  - edge node
- Long haul TCP Connection
- Long haul on the Microsoft Network using pre-existing connections
- Short TCP Connection
Using SharePoint CDN’s

- Tenant level control with folder level (origins) opt-in
- Auto origins for Publishing
- HTTP2
- Automatic URL rewrite
- 15min update worldwide
- SharePoint referrer

<table>
<thead>
<tr>
<th>Public (Site Assets)</th>
<th>Private (Content)</th>
<th>Public Common</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anonymous access</td>
<td>User cookie on first access</td>
<td>&lt;script src=<a href="http://ajax.aspnetcdn.com/ajax/">http://ajax.aspnetcdn.com/ajax/</a> jquery-2.1.1.js&gt; &lt;/script&gt;</td>
</tr>
<tr>
<td>SharePoint</td>
<td>60 min lifespan</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Slower than Public ~100ms</td>
<td></td>
</tr>
</tbody>
</table>
CDN Command set

Get-Command *TenantCdn*

Set-SPOTenantCdnEnabled -Enable $true -CdnType <Both, Private, Public>
(without any params will configure all default settings including both public and private and default origins)
If you don’t want the default origins you can add the -NoDefaultOrigins switch

Get-SPOTenantCdnEnabled

Add-SPOTenantCdnOrigin
Example: Add-SPOTenantCdnOrigin -CdnType Public -OriginUrl "*/masterpage"

Get-SPOTenantCdnOrigins
Remove-SPOTenantCdnOrigin

Get-SPOTenantCdnPolicies
Set-SPOTenantCdnPolicy
CDN’s and SPFx

Set-SPOTenantCdnEnabled -CdnType Public

Warnings:
- This is a feature built on a 3rd party application with privacy and compliance standards that differ from the commitments outlined by the Microsoft Office 365 Trust Center. Any data cached through this service does not conform to the Microsoft Data Processing Terms (DPT) and is outside of the Microsoft Office 365 Trust Center compliance boundaries.
- Files of type CSS, EOT, GIF, ICO, JPEG, JP, JP2, JPAG, PNG, SVG, TFF, TTF, WebP, and XML are not cached in the locations configured to serve as Public CDN origins. Public CDN origins will not be served and cached in Content Delivery Network (CDN). Such content will then be accessible by everyone anonymously not monitored nor governed by Microsoft content policies.

Public CDN enabled locations:
- /_masterpage
- /_styleLibrary
- /_components/assets (configuration pending)

Sources | Content scripts | Snippets
-------|----------------|--------
- top
- sppnp.sharepoint.com
  - layouts/15
  - sites/Group
- outlook.office365.com
- portal.office.com
- publiccdn.sharepointonline.com
- sppnp.sharepoint.com/sites/apps/ClientSideAssets/2500e6ac-1986-4464-b294-2014180be2ca
  - hello-world-web-part_48906344c_95d4b42a29d1b10f2e5.js
- 1rules.office365.com
- sppprod-a.akamaiah.net
- static2.sharepointonline.com
- onePageResourceLoader (about:blank)
Public non-Microsoft CDNs

Customer accessing the asset from their own library

<table>
<thead>
<tr>
<th>URL</th>
<th>Received</th>
<th>Taken</th>
<th>Initiator</th>
</tr>
</thead>
<tbody>
<tr>
<td>/_catalogs/masterpage/javascript/jquery-2.1.1.min.js</td>
<td>82.98 KB</td>
<td>1.51 s</td>
<td>&lt;script&gt;</td>
</tr>
<tr>
<td><a href="https://cdn.sharepointonline.com/12413/_layouts/15/16">https://cdn.sharepointonline.com/12413/_layouts/15/16</a></td>
<td>18.98 KB</td>
<td>156 ms</td>
<td>&lt;script&gt;</td>
</tr>
<tr>
<td>/ScriptResource.axd?d=M1vNi_a6A2vtkOenP45i9-peGfx</td>
<td>100.80 KB</td>
<td>2.04 s</td>
<td>&lt;script&gt;</td>
</tr>
</tbody>
</table>

Using the publicly published version

<table>
<thead>
<tr>
<th>URL</th>
<th>Received</th>
<th>Taken</th>
<th>Initiator</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-2.1.1.min.js">https://ajax.aspnetcdn.com/ajax/jQuery/jquery-2.1.1.min.js</a></td>
<td>82.74 KB</td>
<td>469 ms</td>
<td></td>
</tr>
</tbody>
</table>
What contributes to poor performance?
The Good, the Bad and the Ugly

Structural Navigation
Security Trimming
  • Further impacted by broken Role Inheritance
CDN’s not configured

Large Images
CBQ vs CSWP
MUI
Large Taxonomy Hierarchy
SharePoint Online is a shared service

How to avoid being Throttled?
- Do not load test
- Don’t build code that utilizes a single service account
- Optimize custom code
- Follow recommended guidelines

User throttling is rare

Background tasks
- Migration
- Backup
- Search
- Sync
Classic or Modern

Why Classic Publishing?
Team sites – why publishing?

Why not Modern?
Strategies for good performance
Avoid “too many, too big”

On-prem

- Low-latencies
- High bandwidth

SPO

- Minimize file sizes
- Reduce file counts.
- Number of calls back to SPO
Avoid “too many, too big”

- Minify files
- Resize images
- Compress images
- Use sprite sheets
Avoid “too many, too big”

- Use SharePoint’s Image Rendition service.
- Leverage a toolkit like Font Awesome in place of individual icons and similar files.
Caching

- Shared Multi-Tenant
- Bottom line: You cannot use Object Cache
- Use Browser Cache
- Blob Cache
  Cachability headers (24 hours)
Leverage Search

- Replace Content Query Web Part (CBQ)
- Use Content Search Web Part (CSWP)
  *if you have SPO Plan 2 or an E3/G3/etc. plan.
Navigation

- Structural navigation (Not Recommended) It is extremely inefficient
- Use Managed navigation without security trimming
- Use Search-driven navigation
- Use a Custom provider
Site Structure and scalability

Traditional design
Top Parent Site Collection
-----→Root Site (Rootweb)
-----→Subsites (Subwebs)

Modern design
Site Collection
-----→Root Site (Rootweb)
Web Parts

- Are you a web part junkie? If so, it’s time to talk rehab.
- Make sure you don’t have closed web parts on pages (Contains=1)
- Switch to client-side code (e.g. SPFx) instead of using server-side processing.
Async code

• Use good asynchronous development techniques.
• For long-running operations, ensure users have a visual indicator (e.g. a spinner) to let them know something is going on.
• For users, appearing slow and being slow are no different.
Know your JS libraries

- Lots of legacy JS code out there.
- Much of it works with SPO, but should you use it?
- Profile libraries to understand how they work to ensure you aren’t introducing slowdowns.
- SPServices example ...
Know your JS libraries

- `$.SPServices.SPGetCurrentUser()`
- Necessary in 2007, but no longer necessary (i.e. we have UPA to supply the information).
- `SPGetCurrentUser()` does “screen scraping” of a rendered profile page and is dead-slow!
Gathering diagnostic tools and data
Tools Overview

• Page Diagnostics for SharePoint *New
• Browser (F12)
• Fiddler
• Wireshark
• Message Analyzer
Demo
Page Diagnostics for SharePoint
Scott
Browser (F12)

- **Strengths**
  - Easy to use
  - All major browsers have debug tools ...
  - ... so present on all systems
  - Wide variety of inspectors and analyses
  - Can export results for further analysis

- **Weaknesses**
  - Limited ability to shape data and requests
  - Goes fairly deep in some areas, but not many

- **Sum-Up**
  - Excellent first-line tool for determining where problems may exist.
Fiddler

- Strengths
  - Relatively easy to use
  - Includes front-end web dev tools
  - Extensible; includes FiddlerScript engine
  - Can import traces from other tools

- Weaknesses
  - SSL analysis requires additional setup
  - More difficult to learn and use

- Sum-Up
  - Great “next step” tool beyond browser F12 capabilities
• **Weaknesses**
  - Need to develop some experience to use effectively
  - Knowledge of network protocols a must (although the tool can lend a hand)
  - As much an art as a science

• **Sum-Up**
  - Great “next step” tool beyond browser F12 capabilities
Microsoft Message Analyzer

- **Strengths**
  - Evolution of NetMon
  - Tremendous capabilities ...
  - OMS and other support
  - Very little it can’t do

- **Weaknesses**
  - Tough tool to learn
  - Overkill for most purposes

- **Sum-Up**
  - The network tool to rule them all ...
Helpful Links

O365 Performance Tuning
https://aka.ms/tune

Tune SharePoint Online Performance
https://go.microsoft.com/fwlink/?linkid=873107

Navigation options for SharePoint Online
https://go.microsoft.com/fwlink/?linkid=873247

Using Content by Search Web Part
https://go.microsoft.com/fwlink/?linkid=873245

SharePoint Developer Community
https://docs.microsoft.com/en-us/sharepoint/dev/community/community

SharePoint PnP Solution Guidance

How to avoid being throttled in SharePoint Online?
https://docs.microsoft.com/en-us/sharepoint/dev/general-development/how-to-avoid-getting-throttled-or-blocked-in-sharepoint-online