# New Paradigm

To enable Windows as a Service model, Microsoft has adopted a very different approach to the building and deploying the Windows OS. The core part of the OS is the same, no matter the device or purpose for which it is being deployed. The same core means that the kernel and base platform support binaries of the OS are the same in Windows desktops, Xbox 1, Windows Server, HoloLens, Surface Hub, and IoT devices. This is known as OneCore.



The OS installations differ only in the Platform Services, Registry, and Policy-Based settings that enable the device (i.e., Desktop, Server). Windows 10 desktop installation registry settings can be modified to function the same way as a Windows Server installation and vice versa.

Looking at the differences between Windows 10 1809 and Windows Server 2019, we find that the binaries are identical. However, the registry settings and Platform Services differ. These differences provide features relevant to the function. Server systems are optimized for system throughput as high-performance application servers. The desktop systems are optimized for response time in interactive desktop use.

At boot time, the registry is queried to make critical resource allocation decisions. These include the size and number of OS heaps, internal system workers threads, system data cache, memory management policies, and default length of time slice (thread quantum).

# Windows as a Service

Windows as a service model is focused on continually providing new capabilities and updates while maintaining a high level of hardware and software compatibility. Deploying new versions of Windows is simpler than ever before: Microsoft releases new “feature updates” two to three times per year. This is a significant departure from the traditional upgrade cycle, where new features are only made available every few years. Ultimately, this model replaces the need for conventional “Windows deployment projects,” which can be disruptive and costly. These feature updates spread the required effort into a continuous updating process, reducing the overall effort required to maintain Windows 10 devices.

The change in architecture to a hybrid kernel and implementation of OneCore, makes the update process more fluid. However, the update cadence and the similar naming convention obscure the real changes taking place. For this reason, Microsoft has adopted the use of a “Feature Update Number” that corresponds to the scheduled release date. These feature updates are the same between server and desktop installs. They can be considered as more closely related to service packs in the past versions of windows. However, these are even smaller and come at a faster pace.

The following diagrams give you a visual of how the installations are similar and how the “feature updates” differ.

## Desktop and Server Comparison

Windows 10 1809 was released in the second half of 2018 and is the starting point for the Windows Server 2019 install. The “Platform Service Layer” is the only difference. In the server installation, we find platform services like Active Directory or Credential Guard that do not exist in the desktop install.

Windows 10 1903 is the next feature update following 1809. We find a fundamental change to the kernel or base platform support binaries between these two features updates. These changes show that there are more differences between the feature updates than between install.

## Feature update equivalence

The following tables show the binary equivalence between desktop and server. These tables show that server and desktop installs use the same OneCore components. Installs of server and desktops were compared, and these are the results:

*Note: The comparison consists of a SHA256 hash of the kernel and the supporting binaries that make up OneCore. Identical content (hash) means that the binaries are identical.*

|  |  |
| --- | --- |
| **Feature Update** | **Desktop Hash** |
| **1909** | 5b475fb13d5125cad47b31080e3d696ca67b023a81de7d8e31f7783af4e7c0d8 |
| **1903** | 5b475fb13d5125cad47b31080e3d696ca67b023a81de7d8e31f7783af4e7c0d8 |
| **1809** | e948ddf80cccd7f2d534f19114b4d6fd4241b7e9cc7bafe52c8e55a17110cf08 |
| **1803** | 6bf589702215367c7b0f27b273fd42ac911c51e0be7d5890c7c3dd6f008083ae |
| **1709** | 56ae1f6754940b72faa8d4de2e0b69a06c08cf1350a499957b6c7537a5aa4e1a |
| **1703** | 83a00714ff8f764a4610982fd68f737a368065d45ad7e926f6537e2cfa7c90f3 |

Hash for Windows Desktop installations

|  |  |
| --- | --- |
| **Feature Update** | **Server Hash** |
| **1909** | 5b475fb13d5125cad47b31080e3d696ca67b023a81de7d8e31f7783af4e7c0d8 |
| **1903** | 5b475fb13d5125cad47b31080e3d696ca67b023a81de7d8e31f7783af4e7c0d8 |
| **1809** | e948ddf80cccd7f2d534f19114b4d6fd4241b7e9cc7bafe52c8e55a17110cf08 |
| **1803** | 6bf589702215367c7b0f27b273fd42ac911c51e0be7d5890c7c3dd6f008083ae |
| **1709** | 56ae1f6754940b72faa8d4de2e0b69a06c08cf1350a499957b6c7537a5aa4e1a |
| **1703** | 83a00714ff8f764a4610982fd68f737a368065d45ad7e926f6537e2cfa7c90f3 |

Hash for Windows Server installations

### OneCore Files

|  |  |
| --- | --- |
| File Name | Components |
| Ntoskrnl.exe | Executive and kernel |
| Hal.dll | Hardware Abstraction Layer |
| Win32k.sys | Kernel-mode part of Windows subsystem (GUI) |
| Hvix64.exe | Hypervisor |
| Ntdll.dll | System service dispatch stub to executive functions |
| Kernel32.dll | Core Windows subsystem (I/O, interrupts, memory) |
| Advapi32.dll | Core Windows subsystem (registry, security, users) |
| User32.dll | Core Windows subsystem (GUI API) |
| Gdi32.dll | Core Windows subsystem (video) |
| \*.sys files in drivers folder | Core Drivers |

## Feature update progression

Windows as a Service changes the update process to a continuous update. Two times per year, Microsoft releases “Feature Updates” that build on the prior version. Newer feature updates include previous “features updates” because it is the same code base, enabling the operating system to be highly compatible.

The following table shows the feature releases and operating system versions. Note that the “version” of the kernel is the same, but the build is higher as the feature release is higher.

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **OS build** | **Availability date** | **Latest quality update** |
| **1909** | 10.0.18363.752 | 11/12/2019 | 3/24/2020 |
| **1903** | 10.0.18362.752 | 5/21/2019 | 3/24/2020 |
| **1809** | 10.0.17763.1131 | 3/28/2019 | 3/17/2020 |
| **1803** | 10.0.17134.1399 | 7/10/2018 | 3/17/2020 |
| **1709** | 10.0.16299.1775 | 1/18/2018 | 3/17/2020 |

## Effects on Applications

Applications written in C# that targets .NET Core are not affected by the feature upgrades. The .NET versions and runtimes are immutable and additive. Once shipped they are frozen and newer versions include older versions with no breaking change.

Biotech applications are generally composed of a .NET (C#) and a Python part. The Python part is primarily for analysis while the .NET part is the rest of the system including UI, business logic, and data access. The .NET frameworks are not installed by the feature updates. If your application is built to target a particular version, it probably installs that version.

Python is not installed or updated by any feature update. The interpreters and supporting binaries are built on the Win32 API. This API is complete and Immutable. There are no new versions of the API. Windows 10 has 100% compatibility with Win32 application with one exception: Windows 10 ARM older than 1809.

## Impact on testing

The implementation of “feature updates” spreads the testing effort into a continuous upgrade process. The massive projects to test for compatibility with new versions of windows are unnecessary. The feature updates are much smaller and well documented. Access to early versions is widely available and encouraged. Incompatibilities can be discovered before the public release of new features. Testing can be “targeted” over “indiscriminate”.

### Language Versions

The OneCore model also changes the way that language installations are created. A Chinese “install” is the same kernel, platform services, and framework. OneCore enables the Language to be changed via a “Language Pack.” This pack only affects what is displayed to the user. It includes the default character set and the localization settings. It does not change the OS or its components. The default OS folder structure is all in English US.

When installing a Japanese Language Windows 10, the OS components are the same as the English Windows 10. The only thing that is different is the Language Pack located in the Platform Service Layer

# Conclusion

With Windows 10 and Server, we should change our testing strategy.

* The operating system is identical between Desktop and Server installations. You don’t need to test both.
* Feature updates should be treated cautiously because they may contain changes that could affect your application. Be proactive and work with your IT group to look at the differences while it is in preview.
* Windows 10 has been out since 2015 and support for older operating systems is coming to an end. For your customer’s IT departments, it means that they are experiencing the same change in paradigm and cadence of releases. The adoption of “feature updates” will be faster than with older versions of Windows. Microsoft has reduced the support timeframe for the “releases” to just 18 months instead of 3 to 5 years. Long term support is the exception. Therefore, you should adopt a faster and more tactical approach to version support and testing.
* You should refocus your testing on the “feature updates” rather than the name. I.e., 1809 rather than Windows 10. Refocusing, in this way, ensures that both server and desktop installs are included.
* Language testing should be refocused on character sets rather than Language Installs.

# References

Microsoft. (2020, March 26). ***.NET architectural components***. Retrieved from Microsoft Docs: <https://docs.microsoft.com/en-us/dotnet/standard/components>

Microsoft. (2020, April 19).***.NET Framework versions and dependencies***. Retrieved from Microsoft Docs: <https://docs.microsoft.com/en-us/dotnet/framework/migration-guide/versions-and-dependencies>

Microsoft. (2020, April 19). ***.NET Standard***. Retrieved from Microsoft Docs: <https://docs.microsoft.com/en-us/dotnet/standard/net-standard>

Microsoft. (2020, March 26). ***Compare features in Windows Server versions***. Retrieved from <https://www.microsoft.com/en-us/cloud-platform/windows-server-comparison>

Microsoft. (2020, March 26). ***Optimize Windows 10 update delivery***. Retrieved from Microsoft Docs: <https://docs.microsoft.com/en-us/windows/deployment/update/waas-optimize-windows-10-updates>

Microsoft. (2020, March 26). ***Overview of Windows as a service***. Retrieved from Microsoft Docs: <https://docs.microsoft.com/en-us/windows/deployment/update/waas-overview>

Microsoft. (2020, March 26). ***Quick guide to Windows as a service***. Retrieved from Microsoft Docs: <https://docs.microsoft.com/en-us/windows/deployment/update/waas-quick-start>

Microsoft. (2020, March 26). ***Windows 10 release information***. Retrieved from Microsoft Docs: <https://docs.microsoft.com/en-us/windows/release-information/>

Yosifovich, P., Ionescu, A., Russinovich, M. E., & Solomon, D. A. (2017). ***Windows Internals, Seventh Edition, Part 1****.* Redmond: Microsoft Press.

Hari Pulapaka, Microsoft (2020, March 26). **One Windows Kernel**. Retrieved from Microsoft Tech Community: <https://techcommunity.microsoft.com/t5/windows-kernel-internals/one-windows-kernel/ba-p/267142>

James Smith, AgapeWORKS (2020, October 13). **Windows Desktop vs Server**. Retrieved from The Expert Software Guy: <https://expertdev.blog/testing/windows-desktop-vs-server/>