Microsoft

# Microsoft Defender ATP on Virtual Desktop Infrastructure

## Performance and recommended configuration whitepaper

Iaan D'Souza-Wiltshire

Contributors

*Shweta Jha, Andy Hurren, Yong Rhee*

# Contents

# Introduction

Virtual Desktop Infrastructure (VDI) is the use of dedicated hardware (often servers) that run multiple copies or instances of an operating system. Each instance is called a Virtual Machine (VM) and is generated with a specific set of pseudo-hardware.

See the Windows Virtual Machines Documentation site for more information on using VDI and Windows.

A common consideration when using VDI is how well each VM can perform. Often a single server with actual physical hardware is used to run multiple VMs – together these VMs share that physical hardware. This means that if multiple VMs are running and each performing tasks, they can only take a share of the actual physical hardware that the server is using. In this sense, VMs can sometimes play a zero-sum game, where they are competing for the same resources: there's only one cake, but all the VMs want a slice and so the slices might vary in size. Some VMs don't get much cake.

Performance on VMs can be managed by reducing the installation of various apps and features, and controlling the configuration available for apps and services. However, because antimalware protection is so vitally important, it can be considered a "must-have". Therefore, the performance of an antimalware product is paramount in VDI.

This means that performance of the antimalware component in Microsoft Defender Advanced Threat Protection – Windows Defender Antivirus (AV) – in VDI is paramount to Microsoft, and in this whitepaper we illustrate how important this is by covering:

1. Performance testing results.
2. Configuration and best practice recommendations for Windows Defender AV in VDI.
3. Testing guidelines and instructions to help you test Windows Defender AV performance on your own VDI.
4. Resources for further Microsoft Defender Advanced Threat Protection configuration and information.

# Performance testing

## Methodology and types of tests

In late 2018 Microsoft began a series of tests to measure the performance impact of Windows Defender AV across a number of virtual machine hosting systems. See Appendix A for the methodology and types of tests run.

This section outlines the results of those tests.

## Results

Note that due to legal requirements we are unable to test ourselves against other antivirus products. Microsoft engaged a vendor to perform a number of tests on Windows Defender AV and three other leading AV products and provide non-biased performance results. Those results are described here.

### CPU

During the real-time protection scan, Windows Defender AV peaked at 40% average processor time around 50 seconds into the test (this corresponds to the opening of the Excel file portion of the test). CPU usage then immediately dropped to 3-5% until 300 seconds, at which point it rose to 15% (Hyper-V and VMWare) for 100 seconds (this corresponds to the running of the EICAR copy .bat file portion of the test). It then dropped back to 3-5% for the remainder of the test (Hyper-V and VMWare).

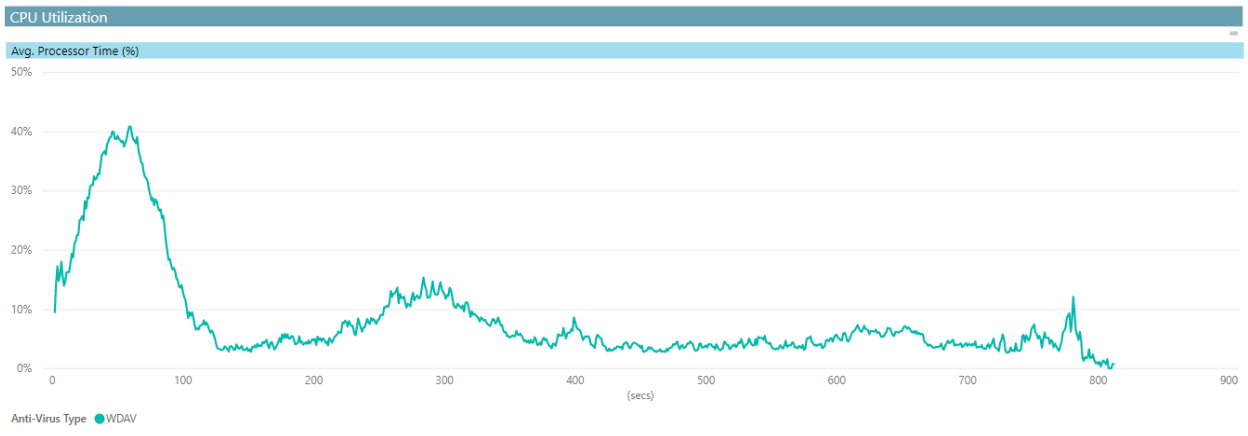



*Figure 1: VMWare CPU usage during real-time protection*

During the quick scan test, CPU usage rose to 30% at around 200 seconds, then tapered off to 2% by 800 seconds.

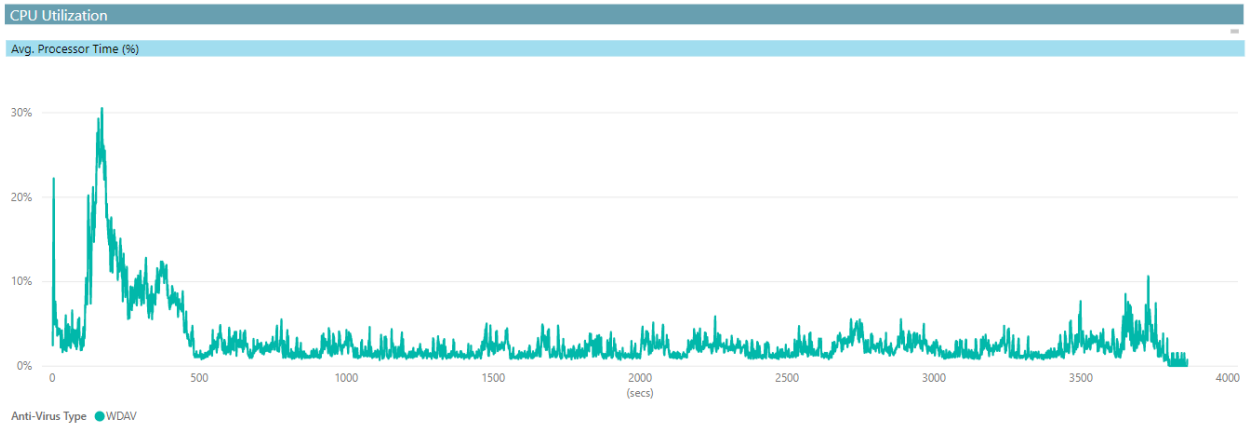



*Figure 2: VMWare CPU usage during quick scan*

## Memory

The quick scan test saw 46% average committed bytes during the entirety of the test. On Hyper-V, Windows Defender AV recorded 50% committed bytes in use for the first 1000 seconds, followed by a peak around the 1000 second mark before the test was completed.

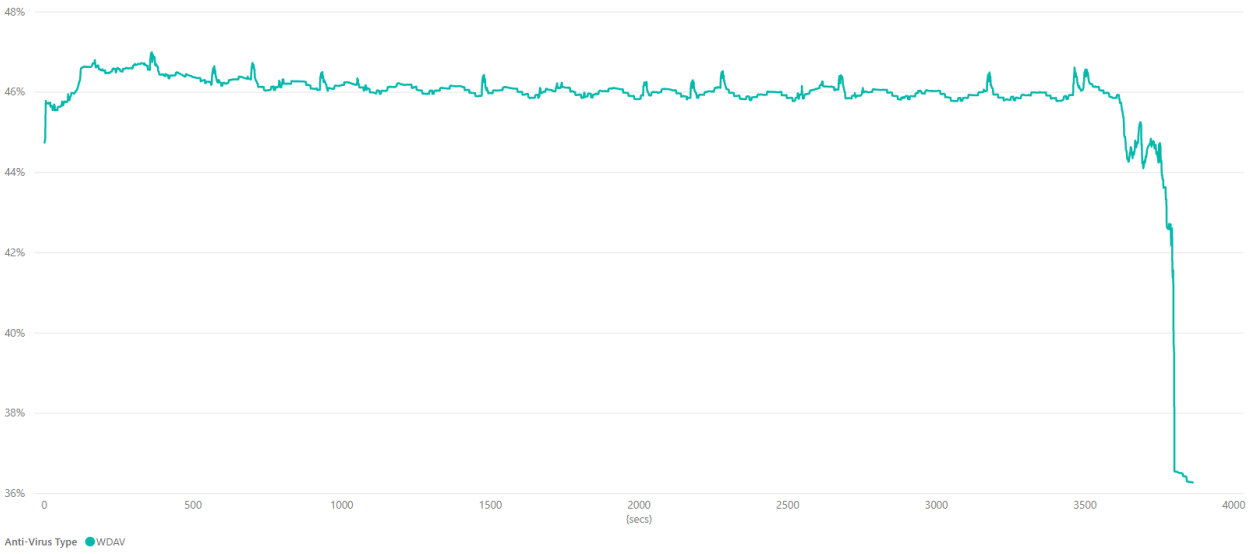



*Figure 3: VMWare Memory usage during quick scan*

During the real-time protection test, memory usage was recorded at 44% 200 seconds into the test (which corresponds with the opening of the .bat file that copied the EICAR file), before tapering down to just above 40% for the remainder of the test.

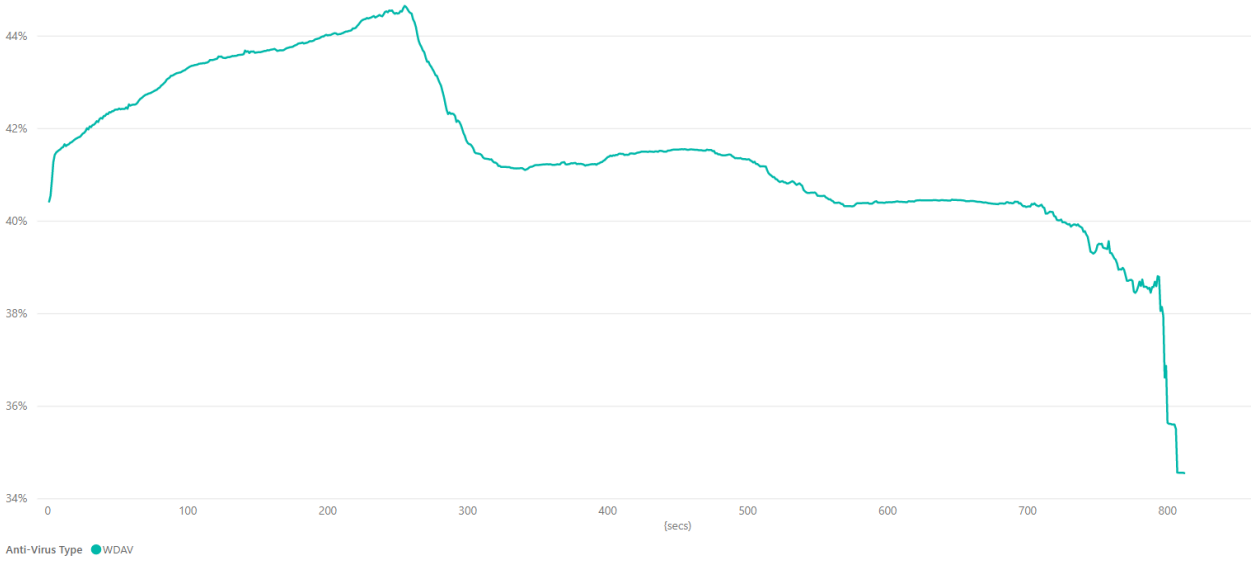



*Figure 4: VMWare Memory usage during real-time protection*

## Read/write

Average disk reads per second were consistently low throughout the quick scan test, initiating at 10% before dropping to a range between 2% and 5% for the remainder of the test.

## Login/startup

On Hyper-V, Windows Defender AV added 6 seconds to the baseline test. On VMWare it added under 100 seconds.

# Configuration and testing recommendations for customers

## Introduction

In the Windows 10, version 1903 release a new management option ("shared security intelligence location") became available that allows enterprises[1] to reduce the CPU and network overhead for installing security intelligence updates (also known in the antimalware industry as "definitions").

The shared security intelligence location feature works by offloading the processing required by an endpoint to unpackage and install security intelligence updates.

In a normal deployment, WSUS, SCCM, or some other management agent is notified of a new Windows Defender AV security intelligence update. It then notifies the endpoints that it is managing that this update is available, and either instructs the endpoint to download the package, or automatically transfers the package from a shared location to each endpoint. This is shown in Figure 5.

[1] A Microsoft Defender ATP license is required. This is typically furnished through the Windows E5, Microsoft 365 E5, or EMS licenses

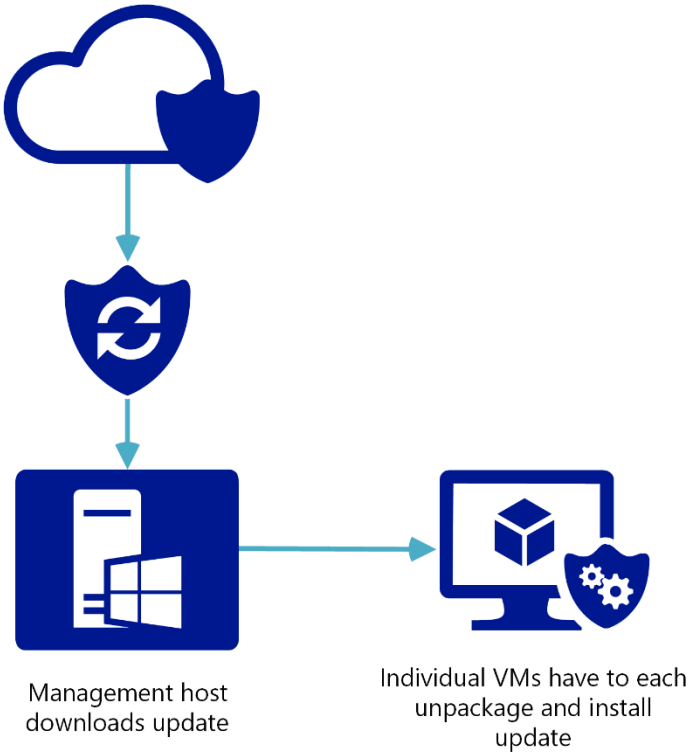



*Figure 5: Security intelligence updates without shared intelligence location*

The security intelligence update is delivered as a compressed binary-similar package. Each individual endpoint must unpackage the update before it can apply it. This requires CPU and memory usage.

With the shared security intelligence feature, the update is instead downloaded and unpackaged by a management machine, which could be running Windows 10, version 1903 or Windows Server 2019. Individual endpoints can then obtain the already-expanded bits and apply them directly to Windows Defender AV. This means the endpoints do not have to perform the CPU and memory cycles normally required to install a security intelligence update. This is shown in Figure 6.

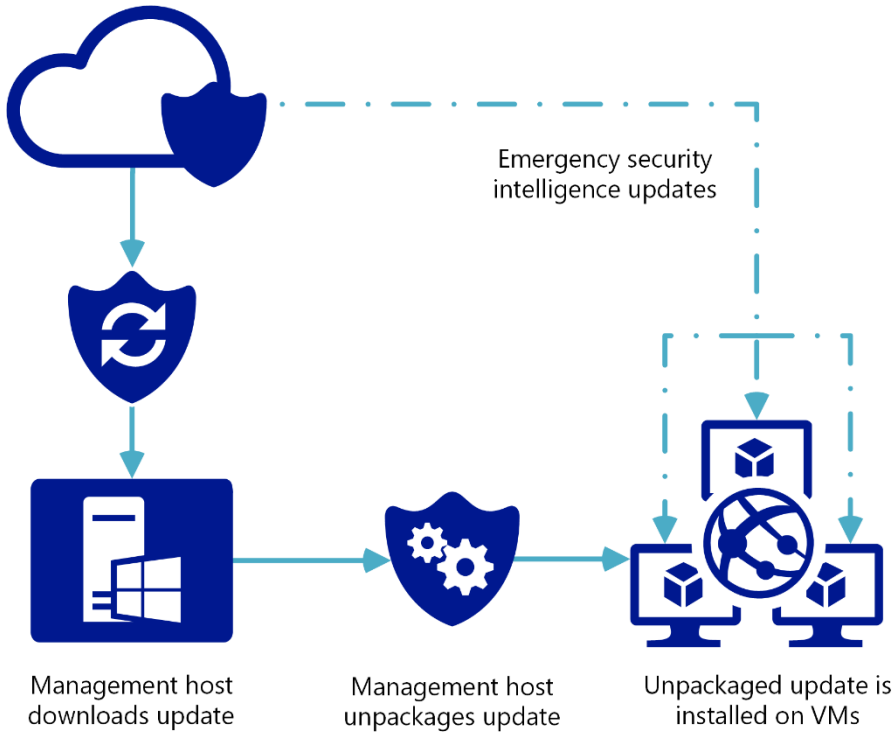



*Figure 6: VMs with the SSU feature*

As part of our release, we'd love for you to test these new improvements and provide feedback. We'll use the feedback to help understand usage, improve further upon our security on virtual machines and in VDI, and address bugs and problems.

To test, you'll need a VDI environment consisting of:

1. At least one management machine running Windows 10 Insider Preview (build 18323 or later) or Windows 10, version 1903
2. At least 20 virtual machines, running Windows 10 Insider Preview (build 18323 or later) or Windows 10, version 1903

There are a few different scenarios that you can test, and we encourage you to use the instructions under Appendix A: Testing methodology on whatever deployment scenario you either already have or want to play around with. The following are some examples:

1. Multiple groups of VMs running on different VM infrastructure, including Hyper-V, Citrix, VMWare, or others
2. Multiple VMs running on single hardware units
3. Individual VMs running on individual hardware

4. VMs that have VPN, proxy, firewalled, or intermittent connections to the management server

However you choose to test, please make sure to identify your deployment when providing verbose feedback.

See the Windows Virtual Machines Documentation site for quick-start guides and details on how to create and provision VMs. If you wish to compare performance, you can use the testing methodology described in Appendix A as a guideline.

## Configure the shared security intelligence feature

First you'll configure your individual VMs to receive intelligence updates through the shared VDI location, then you'll run a PowerShell script that will download and unpackage the update.

Whenever there's a new update that has been unpackaged, the VMs will know to fetch the updates from the management machine.

### Configure the shared security intelligence update

You can do this with Group Policy, PowerShell, or a CSP. You should use whatever you're most familiar with, but if you're not sure which to choose, we recommend CSP as we'll show you how to create a device group for your VMs, configure a policy, and deploy it to the device group.

#### Use Intune to deploy the CSP

Open the Intune management portal either by searching for Intune on https://portal.azure.com or going to https://devicemanagement.microsoft.com and logging in.

First, create groups that you can use to distribute the configuration.

1. To create a group with only the devices or users you specify:
    1. Go to **Groups**. Click **New group**. Use the following values:
        1. Group type: **Security**
        2. Group name: **VDI test VMs**
        3. Group description: *Optional*
        4. Membership type: **Assigned**
    2. Add the devices or users you want to be a part of this test and then click **Create** to save the group. It's a good idea to create a couple of groups, one with VMs running

the latest Insider Preview build and with the shared security intelligence update feature enabled, and another with VMs that are running Windows 10 1809 or earlier versions. This will help when you create dashboards to test the performance changes.

2. To create a group that will include any machine in your tenant that is a VM, even when they are newly created:

    1. Go to **Groups**. Click **New group**. Use the following values:

        1. Group type: **Security**
        2. Group name: **VDI test VMs**
        3. Group description: *Optional*
        4. Membership type: **Dynamic Device**

    2. Click **Simple rule**, and select **deviceModel**, **Equals**, and enter **Virtual Machine**. Click **Add query** and then **Create** to save the group.

Next, create device configuration profiles that contain the configuration settings, and then assign the profiles to the groups you just created.

1. Go to **Device configuration**, then **Profiles**. You can modify an existing custom profile or create a new one. In this demo I'm going to create a new one by clicking **Create profile**.

2. Name it, choose **Windows 10 and later** as the Platform and – most importantly – select **Custom** as the profile type.

3. The **Custom OMA-URI Settings** blade is opened automatically. Click **Add** then enter the following values:

    1. Name: VDI shared sig location
    2. Description: *Optional*
    3. OMA-URI: ./Vendor/MSFT/Defender/SharedSignatureRoot
    4. Data type: **String**
    5. Value: \\<sharedlocation>\wdav-update\

4. Click **Ok** to close the details blade, then **OK** again to close the **Custom OMA-URI Settings** blade. Click **Create** to save the new profile. The profile details page now appears.

5. Click **Assignments**. The **Include** tab is automatically selected. In the drop-down menu, select **Selected Groups**, then click **Select groups to include**. Click the **VDI test VMs** group and then **Select**.

6. Click **Evaluate** to see how many users/devices will be impacted. If the number makes sense, click **Save**. If the number doesn't make sense, go back to the groups blade and confirm the group contains the right users or devices.

7. The profile will now be deployed to the impacted devices. Note that this may take some time.

#### Use Group Policy

1. On your Group Policy management computer, open the Group Policy Management Console, right-click the Group Policy Object you want to configure and click **Edit**.

2. In the **Group Policy Management Editor** go to **Computer configuration**.

3. Click **Administrative templates**.

4. Expand the tree to **Windows components** > **Windows Defender Antivirus** > **Security Intelligence Updates**

5. Double-click **Define security intelligence location for VDI clients** and set the option to Enabled. A field automatically appears, enter \\<*sharedlocation*>*wdav-update*. Click **OK**.

6. Deploy the GPO to the VMs you want to test.

#### Use PowerShell cmdlet

Use the following cmdlet to enable the feature. You'll need to then push this as you normally would push PowerShell-based configuration policies onto the VMs:

```
Set-MpPreference -SharedSignaturesPath \\<shared location>\wdav-update
```

### Download and unpackage the latest updates

Now you can get started on downloading and installing new updates. We've created a sample PowerShell script for you below. This script is the easiest way to download new updates and get them ready for your VMs. You should then set the script to run at a certain time on the management machine by using a scheduled task (or, if you're familiar with using PowerShell scripts in Azure, Intune, or SCCM, you could also use those).

```
$vdmpathbase = 'c:\wdav-update\{00000000-0000-0000-0000-'
$vdmpathtime = Get-Date -format "yMMddHHmmss"
$vdmpath = $vdmpathbase + $vdmpathtime + '}'
$vdmpackage = $vdmpath + '\mpam-fe.exe'
```

```
$args = @("/x")

New-Item -ItemType Directory -Force -Path $vdmpath | Out-Null

Invoke-WebRequest -Uri
'https://go.microsoft.com/fwlink/?LinkID=121721&arch=x64' -OutFile
$vdmpackage

cmd /c "cd $vdmpath & c: & mpam-fe.exe /x"
```

You can set a scheduled task to run once a day so that whenever the package is downloaded and unpacked then the VMs will receive the new update.

We suggest starting with once a day – but you should experiment with increasing or decreasing the frequency to understand the impact.

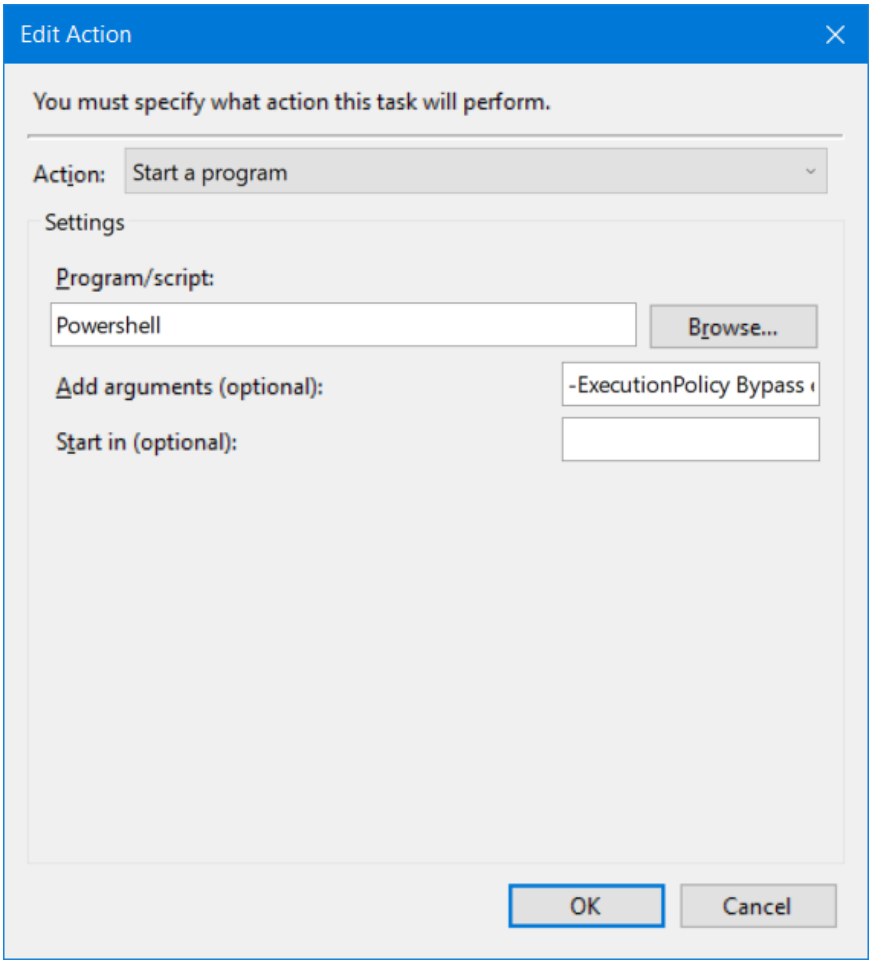
Note that security intelligence packages are typically published once every three to four hours, so setting a frequency shorter than four hours isn't advised as it will increase the network overhead on your management machine for no benefit.

#### Set a scheduled task to run the powershell script

1. On the management machine, open the Start menu and type **Task Scheduler**. Open it and select **Create task...** on the side panel.
2. Enter the name as **Security intelligence unpacker**. Go to the **Trigger** tab. Click **New...** Select **Daily** and click **OK**.
3. Go to the **Actions** tab. Click **New...** Enter **PowerShell** in the **Program/Script** field. Enter

   *-ExecutionPolicy Bypass c:\wdav-update\vdmdlunpack.ps1*

   in the **Add arguments** field. Click **OK**. You can choose to configure additional settings if you wish. Click OK to save the scheduled task.




You can initiate the update manually by right-clicking on the task and clicking **Run**.

**Download and unpackage manually**

If you would prefer to do everything manually, this what you would need to do to replicate the script's behavior:

1. Create a new folder on the system root called *wdav_update* to store intelligence updates, for example, create the folder *c:\wdav_update*
2. Create a subfolder under *wdav_update* with a GUID name, such as *{00000000-0000-0000-0000-000000000000}*; for example *c:\wdav_update\{00000000-0000-0000-0000-000000000000}* (note, in the script we set it so the last 12 digits of the GUID are the year, month, day, and time when the file was downloaded so that a new folder is created each time. You can change this so that the file is downloaded to the same folder each time)

3. Download a security intelligence package from https://www.microsoft.com/en-us/wdsi/definitions into the GUID folder. The file should be named *mpam-fe.exe*.

4. Open a cmd prompt window and navigate to the GUID folder you created. Use the **/X** extraction command to extract the files, for example *mpam-fe.exe /X*.

Note: The VMs will pick up the updated package whenever a new GUID folder is created with an extracted update package or whenever an existing folder is updated with a new extracted package.

## Configure recommended settings for optimal performance

Use the recommended settings as described in the Deployment guide for Windows Defender AV in a VDI. Either add these settings to the profile you created for the shared security intelligence update, or create a new profile with these settings and deploy it to the VMs you want to test.

## Monitor and report on performance

You should monitor for performance during a dedicated period of time, allowing your users to use their VMs as per normal. Take note of perceived and actual differences between using Windows 10, version 1903 and previous releases of Windows 10 (version 1809 and earlier), and between enabling or disabling the shared security intelligence setting.

An example of how to go about this would be:

1. Note the scheduled times for all update and scan actions. For example, take a note of the configuration you have made for scheduled scan times, security intelligence updates, and Windows OS releases (the Windows Defender AV engine updates alongside these monthly releases every "Patch Tuesday" which is the second Tuesday of the month).
2. Use the testing methodology described in Appendix A to define the type and number of VMs and the types of tests you want to run.
3. Use more than the minimum hardware requirements for Windows 10 for each VM. As a general rule, this should be 2GB of RAM or more and 20GB hard disk space or more. Use settings that you would likely want to use for your actual business uses – you should try to replicate the real environment in which you would use VMs as your test environment.
4. Build a dashboard with the Azure Log Analytics tutorial, and take note of the performance metrics described in the following table at the times for scheduled scans,

security intelligence updates, and OS or platform updates. You can also use the Windows Performance Toolkit to measure performance.

5. Compare these between VMs that are running Windows 10, version 1903 versus VMs that are running Windows 10, version 1809 or earlier.

| Use case | Performance to record |
|---|---|
| **Quick scan** | CPU utilization, memory usage, and disk usage (read/writes) |
| **Security Intelligence Updates** | Network bandwidth and CPU utilization when downloading and installing an update |
| **Cloud-delivered protection** | Network bandwidth when encountering a cloud block file (sign in required) |
| **Monthly engine protection updates** | All performance indicators as for quick scan |
| **General performance at startup and during normal usage** | All performance indicators as for quick scan, plus deferred procedure calls and interrupt service routines time, using the Windows Performance Toolkit or manually collecting boot trace logs |

## Send us feedback

As part of your testing, we'd be interested to know what sort of performance you notice on your VMs, and also any comments you have about the experience. You can send feedback to us through the feedback form available on the Microsoft Defender ATP testground site (click **Feedback** in the navigation pane and follow the instructions on the form).

# Appendices

# Appendix A: Testing methodology

First a baseline was created by disabling Windows Defender AV and then running the tests.This was used to compare against the performance noted during the tests when Windows Defender AV was running.

The following are the key points for the test methodology:

1. Test cases were run with 20, 40, and 50 concurrent instances
2. Each VM had Windows Defender AV configured according to the VDI deployment guide.
3. The tests were run in last quarter of 2018 (from August to December), using Windows 10, version 1803.

The following tests were run on each VM:

1. Startup and login – Power On was sent to each VM with a 60 second delay between each. A baseline was also created to compare, where Windows Defender AV was disabled.
2. Idle baseline – A 15 minute window was recorded on VMs with Windows Defender AV disabled, and no instructions were sent to the VM.
3. Quick Scan – A quick scan was initiated on each VM.
4. Real-time protection – An Excel file with junk data is opened on each VM and closed after 200 seconds. After the Excel instance closes, a .bat file is used to copied the EICAR test file from a quarantined directory to the desktop.

The following tests were run on Hyper-V and VMWare. The following installation configuration was used for each:

1. Hyper-V:
    1. Tests were executed on a server running Windows Server 2016 with the Hyper-V Role configured. The Server Manager App was used to create the VDI deployment.
    2. Windows 10 Enterprise 1803 was installed on the template VM with default installation settings and Cortana disabled. Each VM used a registry key to logon to a test account with all the test materials copied to the desktop. This included a folder containing the EICAR test files, which were also excluded from antivirus scans.

Windows Defender AV was also configured for performance in VDI according to the VDI deployment guide.

2. VMWare:
    1. Tests were executed on a server running EXSi 6.5.
    2. VMWare machines were managed by a separate physical machine with Windows Server 2016 installed running VMWare vCenter 6.5.
    3. A separate laptop running Windows Server 2016 was used to run VMWare Horizon, the VDI component that created the desktop pool.
    4. Windows 10 Enterprise 1803 was installed on the template VM with default installation settings and Cortana disabled.
    5. VMware Tools was also installed on each virtual machine. Each VM used a registry key to logon to a test account with all the test materials copied to the desktop. This included a folder containing the EICAR test files, which were also excluded from antivirus scans.
    6. Windows Defender AV was also configured for performance in VDI according to the VDI deployment guide.

# Appendix B: Resources

## General resources

http://aka.ms/wdavtechnet

http://www.microsoft.com/mmpc

https://aka.ms/mmpcblog

**Lifecycle information on both Windows Defender Antivirus and SCEP**

https://support.microsoft.com/lifecycle/search

## Test and deploy Windows Defender AV

**Windows Defender AV compliance mapping whitepaper**

http://download.microsoft.com/download/C/7/7/C778B7BB-0783-42D7-93A9-B86DFB5A7BAD/Coalfire_Branded_Windows_Defender_Whitepaper_EN_US.pdf

**Windows Defender Antivirus & Exploit Guard protection evaluation guide**

https://www.microsoft.com/en-us/download/details.aspx?id=54795

**Deployment guide for Windows Defender Antivirus in a virtual desktop infrastructure (VDI) environment**

https://docs.microsoft.com/en-us/windows/security/threat-protection/windows-defender-antivirus/deployment-vdi-windows-defender-antivirus

**Recommended settings for VDI desktops**

https://docs.microsoft.com/en-us/windows-server/remote/remote-desktop-services/rds-vdi-recommendations

## Additions and changes to security in Windows 10

**What's new in Windows 10, version 1809 for IT Pros - Security**

https://docs.microsoft.com/en-us/windows/whats-new/whats-new-windows-10-version-1809#security

**What's new in Windows 10, version 1803 IT Pro content - Security**

https://docs.microsoft.com/en-us/windows/whats-new/whats-new-windows-10-version-1803#security

**What's new in Windows 10, version 1709 IT Pro content - Security**

https://docs.microsoft.com/en-us/windows/whats-new/whats-new-windows-10-version-1709#security

**What's new in Windows 10, version 1703 IT pro content - Security**

https://docs.microsoft.com/en-us/windows/whats-new/whats-new-windows-10-version-1703#security

**What's new in Windows 10, version 1607 - Security**

https://docs.microsoft.com/en-us/windows/whats-new/whats-new-windows-10-version-1607#security

**What's new in Windows 10, versions 1507 and 1511 - Security**

https://docs.microsoft.com/en-us/windows/whats-new/whats-new-windows-10-version-1507-and-1511#security

## Why WD AV?

**Top scoring in industry tests**

https://docs.microsoft.com/en-us/windows/security/threat-protection/intelligence/top-scoring-industry-antivirus-tests

**Why Windows Defender Antivirus is the most deployed in the enterprise**

https://cloudblogs.microsoft.com/microsoftsecure/2018/03/22/why-windows-defender-antivirus-is-the-most-deployed-in-the-enterprise/

**Antivirus evolved**

https://cloudblogs.microsoft.com/microsoftsecure/2017/05/08/antivirus-evolved/

**The Evolution of Malware Prevention (Machine Learning) whitepaper**

https://info.microsoft.com/Windows-Defender-ML-Whitepaper-Registration.html

**Windows Security Whitepaper - Windows 10 - Windows Defender Antivirus**

http://info.microsoft.com/rs/157-GQE-382/images/Windows%2010%20Security%20Whitepaper.pdf