SECMON1

Information Security Guideline

Application Whitelisting

THIS PAGE INTENTIONALLY LEFT BLANK

Application Whitelisting is a means to limit the number of programs running in your business environment that can potentially pose a danger to the security of your data. It restricts users from installing and accessing applications on their computer or electronic device other than those explicitly allowed by your company. In the SECMON1 blog post 'Security Overview - Information Security Essentials', we spoke about what application whitelisting is and why it is an essential security measure.

In this document, we are going to describe how to implement some basic application whitelisting as well as take you step by step through some ways you can achieve this using your existing technology and resources. Finally, we will give you some interesting and important links where you can educate yourself further on the topic and discuss some other options available to you.

Software Applications

There are some vendor-provided application whitelisting solutions available. It is worthwhile to know that some antivirus and data loss prevention solutions include application whitelisting functionality. Below are a few of them and of course there are others. We suggest to identify what antivirus product is installed in your environment and make enquires as to whether it includes application whitelisting

- Airlock Digital (Application Whitelisting)
- Ivanti (Application Whitelisting)
- Trend Micro (Application Whitelisting)
- McAfee (Antivirus with whitelisting capability)
- Carbon Black (Antivirus with Whitelisting capability)
- Kaspersky (Antivirus with Whitelisting capability)
- Digital Guardian (Data Loss Prevention with Whitelisting capability)

Note: Application whitelisting products may conflict with anti-malware software from a different vendor. Application whitelisting solutions are not meant to be a replacement for antivirus or other security software in place. You can read more on that in the SECMON1 Anti-virus guide. Using multiple security solutions together is an effective defence-in-depth approach to preventing a system compromise.

Implementation Guidance

It is advisable to deploy application whitelisting in phases instead of trying to deploy it to an entire organisation at once. Try on a couple of users first and once any issues are understood and resolved, deploy to the remaining users in the organisation.

Deploying application whitelisting is easier if the organisation has visibility of what software is installed on its computers. Such visibility can be obtained by maintaining a detailed inventory of software. Initially testing application whitelisting in 'audit'/'logging only' mode helps to develop an inventory of installed software, while taking care to avoid including existing malware in the inventory. Once an inventory has been established, application whitelisting can be properly configured in 'enforce' mode to prevent unapproved programs from running. We'll look at this a more detail later in the guide.

AppLocker, available in Microsoft Windows 7-8.1 Enterprise and Ultimate Editions as well as Microsoft Server 2008 and 2012, is an application whitelisting capability.

Device Guard, introduced in Microsoft Windows 10 and Microsoft Windows Server 2016, is an application whitelisting capability.

Further information

Further guidance, including applicability for non-Windows operating systems, is available at ASD Strategies to Mitigate Cyber Security Incidents.

How-To Guide

The following step by step guides are designed to enable you to deploy some fundamental application whitelisting within your organisation. If during this process, or before, you would like to enquire as to SECMON1 providing you a quote to perform this task on your behalf, please direct enquiries to our Cyber Hotline at: http://secmon1.com/cyberhotline

Microsoft Windows 7-8.1, Server 2008 and Server 2012

Starting in Windows 7 Enterprise and Ultimate Editions, AppLocker is a built-in security feature that allows organisations to whitelist applications on workstations.

Note: While AppLocker is only available in the Enterprise and Ultimate Editions of Windows 7-8.1, it is available in all versions of Server 2008 and 2012.

For additional reading about AppLocker, please consult the following resource:

Microsoft Windows AppLocker

How to Get to AppLocker

The policies in the Windows desktop versions are also available in the Server versions. It is accessed through Windows Group Policy either individually (gpedit.msc) or from an enterprise standpoint (gpmc.msc).

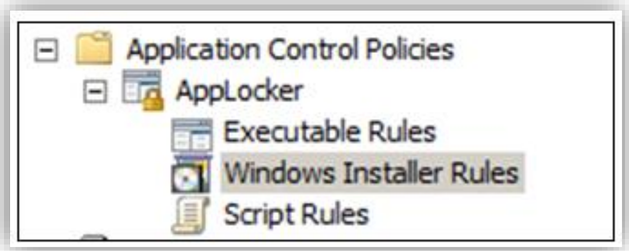
1. Click the **Start** button, then select **All Programs**, then select **Accessories**, and then click **Run**. This will bring up a command prompt.

2. When the command prompt dialog box opens, type **gpedit.msc** or **gpmc.msc** (whichever one applies to your organization).

3. Once your group policy editor is opened, to find AppLocker navigate to the following location: COMPUTER CONFIGURATION -> POLICIES -> **WINDOWS SETTINGS -> SECURITY SETTINGS -> APPLICATION CONTROL POLICIES -> APPLOCKER**

The AppLocker User Interface

The AppLocker interface is designed to be simple, yet powerful.



As depicted above, you have three sections to create your rules against – Executable Rules, Windows Installer Rules and Script Rules.

Executable Rules and Script Rules define which programs or scripts are/are not allowed to execute on the target machine. Similarly, Windows Installer Rules define which installer (MSI) packages can/cannot be installed on a machine.

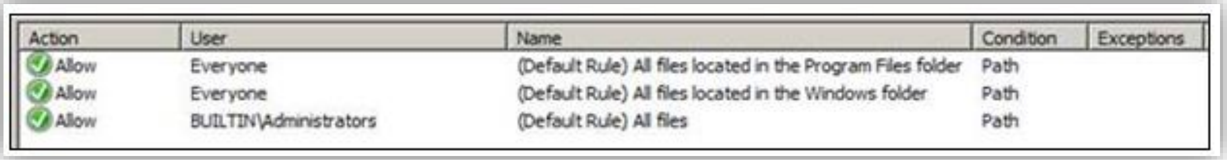
Clicking on each one of them, and then navigating to the whitespace on the screen to the right in the editor, and right-clicking (using the mouse button on the right) in the empty whitespace will bring up a context menu to create a new rule.

The three options we are interested in are at the top of the context menu:

- Create Default Rules
- Create New Rules
- Automatically Generate Rules

Create Default Rules

By choosing this option, AppLocker will create a default rule set for your configuration. This is a great starting point for understanding rules and to get a basic set of rules implemented. The following will be created by default for Executable Rules:



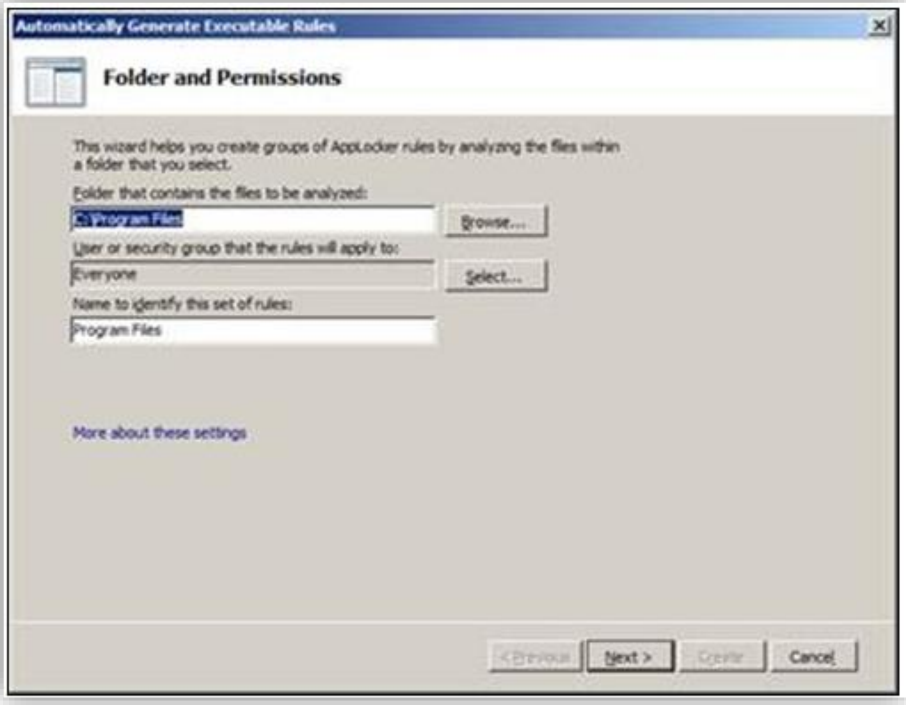
Action	User	Name	Condition	Exceptions
✓ Allow	Everyone	(Default Rule) All files located in the Program Files folder	Path	
✓ Allow	Everyone	(Default Rule) All files located in the Windows folder	Path	
✓ Allow	BUILTIN\Administrators	(Default Rule) All files	Path	

Similar rule sets will be created for Windows Installer Rules and Script Rules.

Automatically Generate Rules

This option will point to a specific location on a machine, and automatically scour the machine for executables, MSIs and or scripts and configure the rules automatically as a base starting point. This option is extremely useful for an Enterprise that is building their base image and wants to quickly generate a rule set that can be applied to the user community.

When starting the wizard this is the first screen you will be presented with. In this screen, you have to choose your path, and an identifier for the configured rules.


Automatically Generate Executable Rules
Browse...
Select...
More about these settings
< Previous
Next >
Create
Cancel

Folder and Permissions

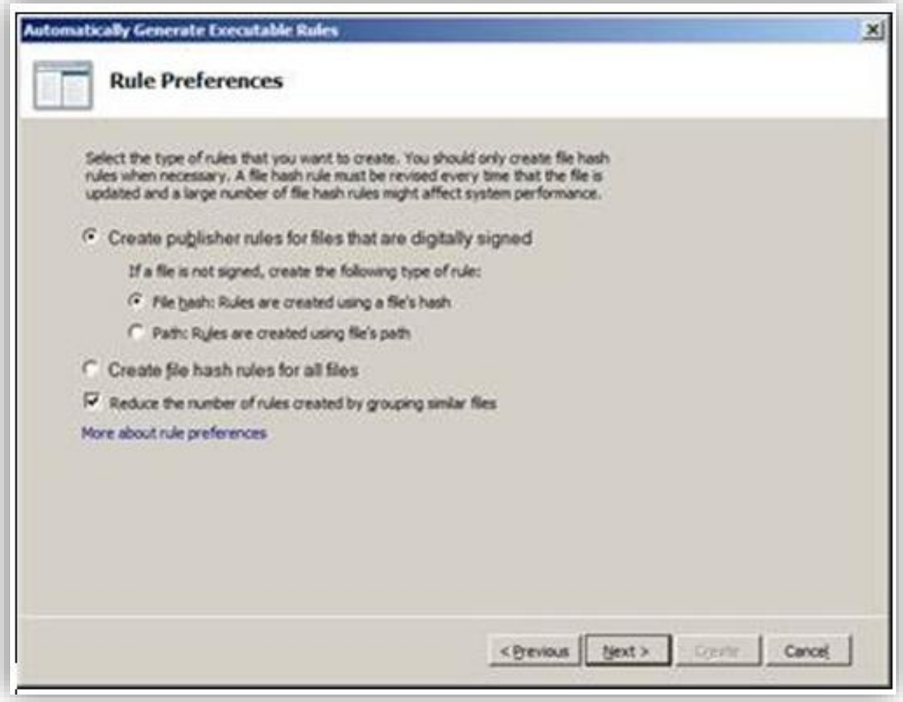
This wizard helps you create groups of AppLocker rules by analyzing the files within a folder that you select.

Folder that contains the files to be analyzed: C:\Program Files

User or security group that the rules will apply to: Everyone

Name to identify this set of rules: Program Files

Clicking on **Next** will show a default rule set screen that will allow you to configure the behavior of the discovery.


Automatically Generate Executable Rules

Rule Preferences

Select the type of rules that you want to create. You should only create file hash rules when necessary. A file hash rule must be revised every time that the file is updated and a large number of file hash rules might affect system performance.

(●) Create publisher rules for files that are digitally signed

If a file is not signed, create the following type of rule:

(●) File hash: Rules are created using a file's hash

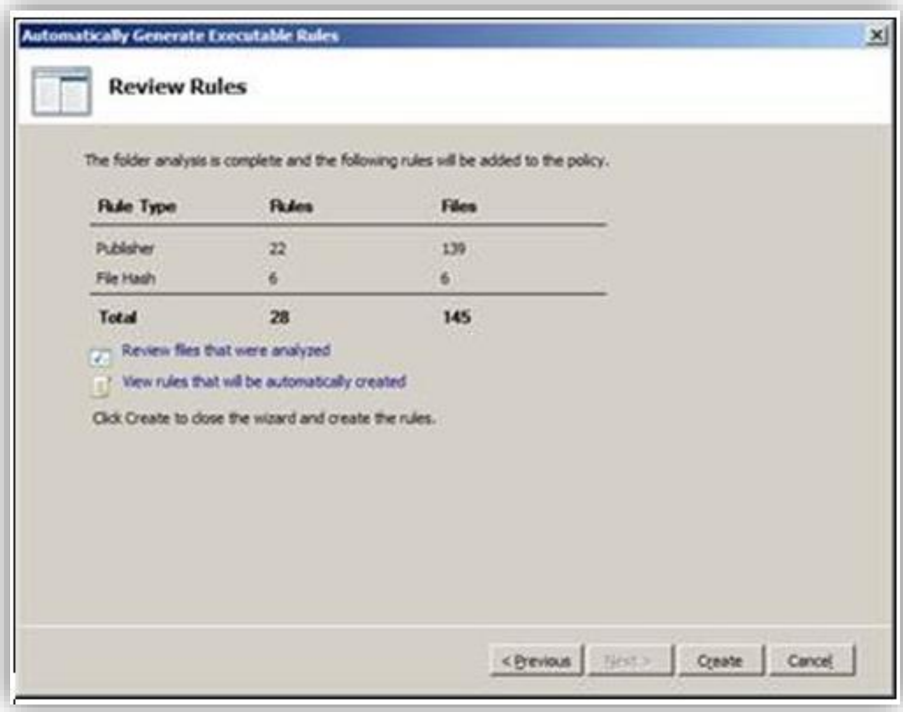
( ) Path: Rules are created using file's path

( ) Create file hash rules for all files

[✓] Reduce the number of rules created by grouping similar files

More about rule preferences

Clicking on **Next** will start the discovery, after which a summary screen will be displayed.


Automatically Generate Executable Rules

Review Rules

The folder analysis is complete and the following rules will be added to the policy.

Rule Type	Rules	Files
Publisher	22	139
File Hash	6	6
Total	**28**	**145**

Review files that were analyzed

View rules that will be automatically created

Click Create to close the wizard and create the rules.

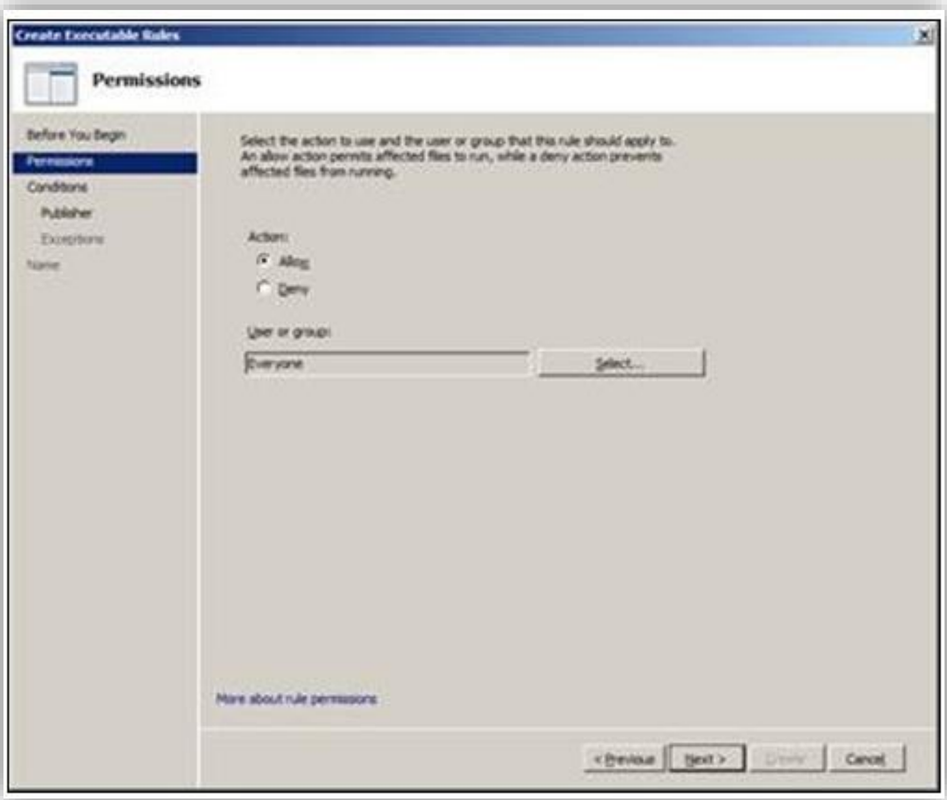
You have the option to view all the files that were examined as part of the search, and the corresponding rules that will be created. Click on Create to create the rules.

Create New Rule

When "Create New Rule..." is chosen, a new rule wizard will open up and will guide you through the process of configuring AppLocker. The following screens are for the Executable Rules wizard, but the other wizards are very similar in design.

The first screen is a welcome screen that can be chosen to skip by default.

The second screen is where the configuration begins. This is where we choose if the behaviour of this rule is going to allow or deny access, and to whom the rule applies.

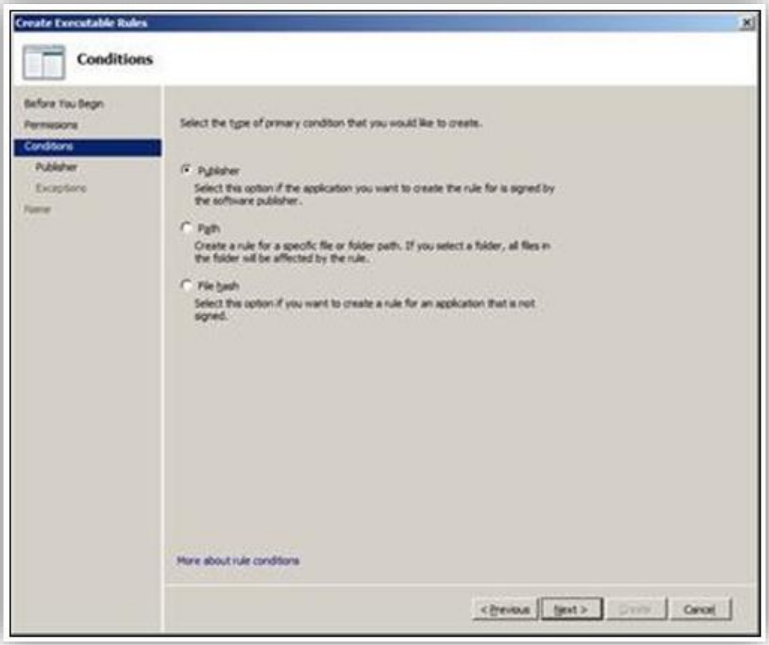


Select the action to use and the user or group that this rule should apply to. An allow action permits affected files to run, while a deny action prevents affected files from running.

Action:
[✓ Allow]
Deny

User or group:
Everyone

After clicking Next, the following screen will have three options. Choose the option that will make the most sense for that particular application.



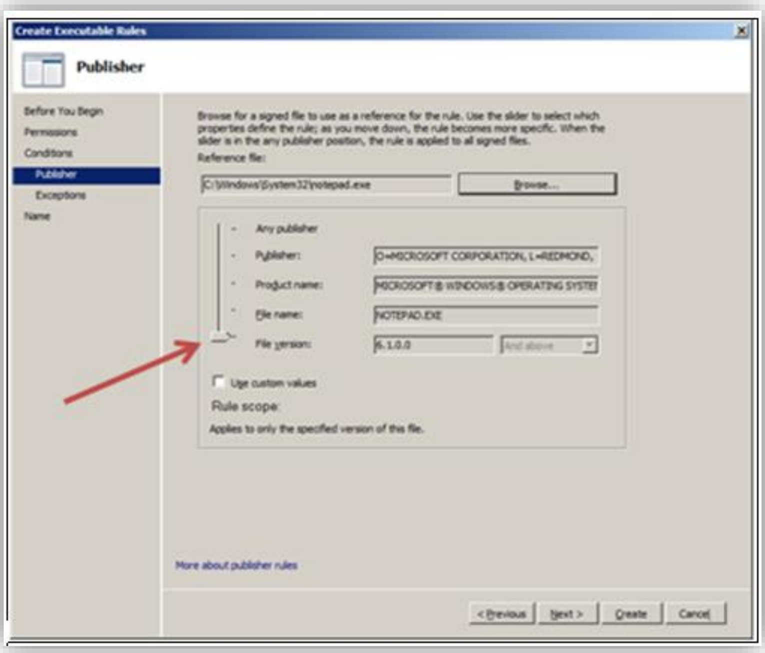
Select the type of primary condition that you would like to create.

(●) Publisher
Select this option if the application you want to create the rule for is signed by the software publisher.

( ) Path
Create a rule for a specific file or folder path. If you select a folder, all files in the folder will be affected by the rule.

( ) File hash
Select this option if you want to create a rule for an application that is not signed.

For this example, "Publisher" was chosen and after clicking on **Next**, you're presented with the following screen. After browsing for your signed executable, all the fields will be populated automatically. The most important and simplest function on this screen is the slide control that is highlighted. This will let you scope the granularity of the rule from most restrictive (File Version) to least restrictive (Any Publisher). Using custom values will allow you to edit the fields that are prepopulated.



Browse for a signed file to use as a reference for the rule. Use the slider to select which properties define the rule; as you move down, the rule becomes more specific. When the slider is in the any publisher position, the rule is applied to all signed files.

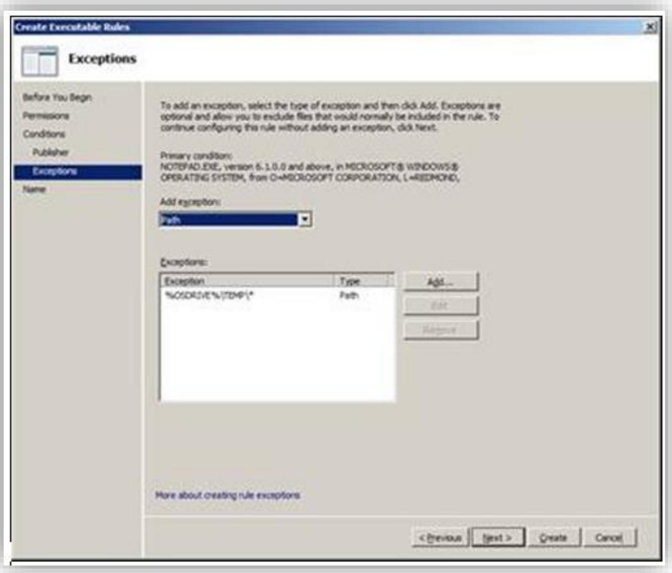
Reference file: C:\Windows\System32\notepad.exe

Any publisher
Publisher: O=MICROSOFT CORPORATION, L=REDMOND,
Product name: MICROSOFT® WINDOWS® OPERATING SYSTEI
File name: NOTEPAD.EXE
File version: 6.1.0.0 And above

[ ] Use custom values

Rule scope:
Applies to only the specified version of this file.

Upon continuing in the wizard, the following screen is the exceptions screen, where you can once again add an exception based on Publisher, Path and/or File Hash. The path %OSDRIVE%\TEMP\* is added as an example exception.



To add an exception, select the type of exception and then click Add. Exceptions are optional and allow you to exclude files that would normally be included in the rule. To continue configuring this rule without adding an exception, click Next.

Primary condition:
NOTEPAD.EXE, version 6.1.0.0 and above, in MICROSOFT® WINDOWS® OPERATING SYSTEM, from O=MICROSOFT CORPORATION, L=REDMOND,

Add exception: Path

Exceptions:

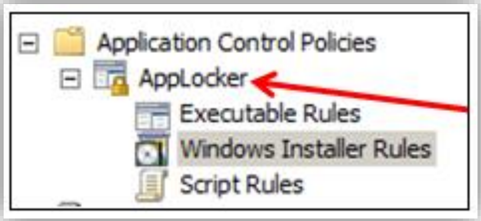
Exception	Type
%OSDRIVE%\TEMP\*	Path

And finally after clicking **Next**, name your rule and choose **Create**.

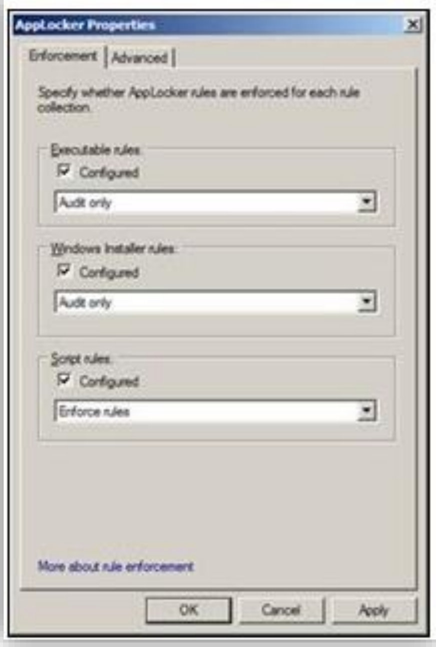
In the example above, an Executable Rules rule was created. Within that rule Notepad.exe is allowed to be run by Everyone, however an exception was added that if Notepad.exe is located in %OSDRIVE%\TEMP\*, it will not be allowed to execute.

Enabling the AppLocker Rules

Right-click on the **AppLocker** top level node (below), and choose properties.



Once you do that, you will be presented with the final configuration step necessary to enable AppLocker.



Specify whether AppLocker rules are enforced for each rule collection.

Executable rules:
✓ Configured
Audit only

Windows Installer rules:
✓ Configured
Audit only

Script rules:
✓ Configured
Enforce rules

On the Enforcement tab, you have to choose which rules to enable.

You will have two options – Audit Only or Enforce Rules.

It is HIGHLY RECOMMENDED that any initial deployment of AppLocker be set to Audit Only. This will create log entries in the event log, located under Application and Services Logs -> Microsoft -> Windows, click AppLocker.

Use this log to view how your rules are behaving before any enforcement is enabled.

Microsoft Windows 10 and Server 2016

Windows 10 introduced a new set of features called Device Guard that enables enterprises to strongly control what is allowed to run in their environment.

As of Windows 10 Enterprise Anniversary Edition, a new type of AppLocker rule can be configured that identifies a specific trusted installation authority, or Managed Installer. Any applications installed by that specified installation authority will be automatically trusted by AppLocker and allowed to run without needing to create any other rules. Applications and software that are installed using any other mechanism will run only if explicitly allowed by another AppLocker rule.

For additional reading about Device Guard and AppLocker, please consult the following resources:

Device Guard Documentation

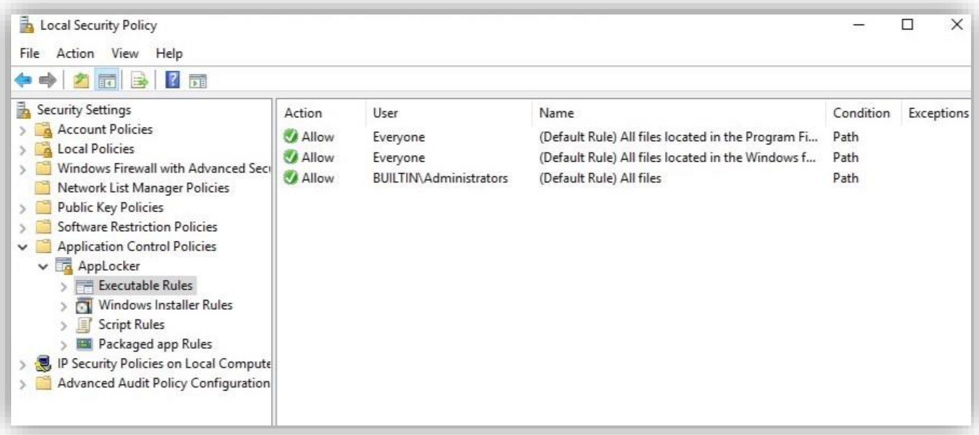
Device Guard Deployment Guide

AppLocker Documentation

Blog: Managing Device Guard Configurable Code Integrity with existing ConfigMgr functionality

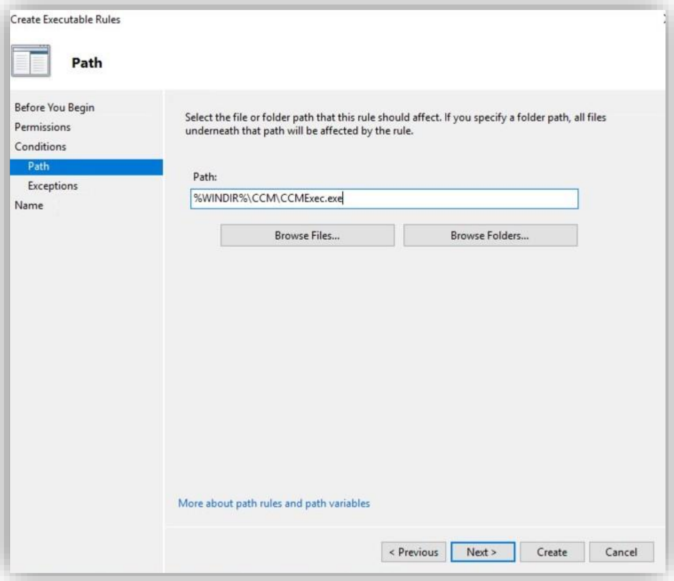
Creating a Custom AppLocker Policy

1. From the Windows Start menu, type **secpol.msc** and then press **Enter** to launch the Local Security Policy MMC snap-in. Once the console opens, navigate to **Application Control Policies > AppLocker > Executable Rules**.


Local Security Policy
File Action View Help
Security Settings
Account Policies
Local Policies
Windows Firewall with Advanced Sec
Network List Manager Policies
Public Key Policies
Software Restriction Policies
Application Control Policies
AppLocker
Executable Rules
Windows Installer Rules
Script Rules
Packaged app Rules
IP Security Policies on Local Compute
Advanced Audit Policy Configuration

Action	User	Name	Condition	Exceptions
Allow	Everyone	(Default Rule) All files located in the Program Fi...	Path	
Allow	Everyone	(Default Rule) All files located in the Windows f...	Path	
Allow	BUILTIN\Administrators	(Default Rule) All files	Path	

2. Right-click (use the mouse button on the right side) **Executable Rules** and create a new rule that allows "Everyone" to run CCMExec.exe based on a condition of your choice. For this example, a File Path condition has been selected (this is the least secure option but it should allow readers to copy the policy used here for basic testing).

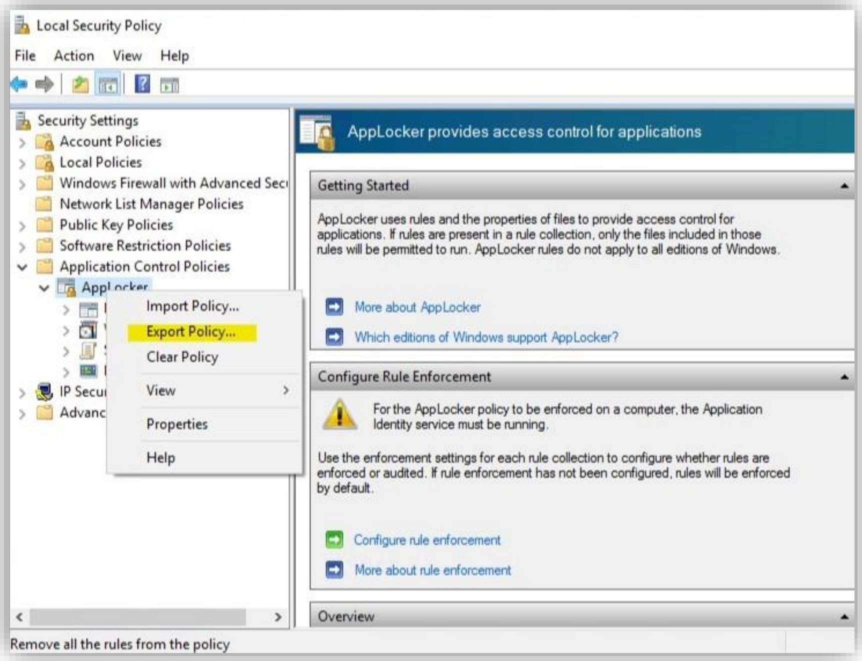

Create Executable Rules

Path

Select the file or folder path that this rule should affect. If you specify a folder path, all files underneath that path will be affected by the rule.

Path: %WINDIR%\CCM\CCMExec.exe

3. Once the rule has been created it will appear in the console. Now, export the policy XML for editing. Right-click **Applocker** in the navigation pane and select **Export Policy…** highlighted below.


Local Security Policy

AppLocker provides access control for applications

Getting Started

AppLocker uses rules and the properties of files to provide access control for applications. If rules are present in a rule collection, only the files included in those rules will be permitted to run. AppLocker rules do not apply to all editions of Windows.

Configure Rule Enforcement

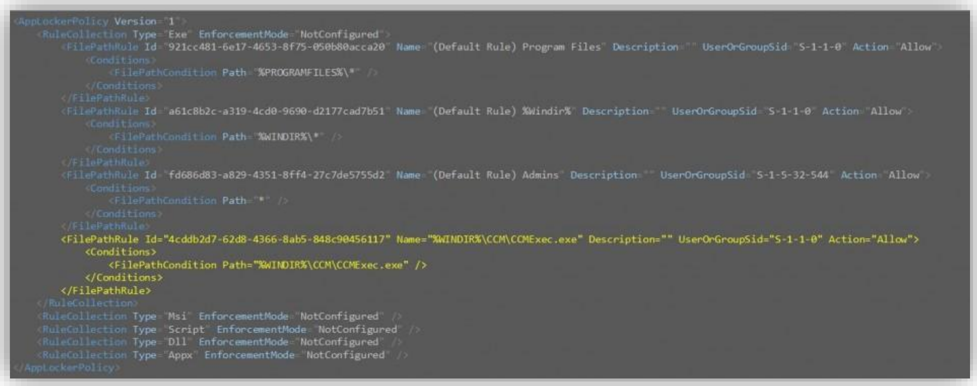
For the AppLocker policy to be enforced on a computer, the Application Identity service must be running.

Use the enforcement settings for each rule collection to configure whether rules are enforced or audited. If rule enforcement has not been configured, rules will be enforced by default.

Remove all the rules from the policy

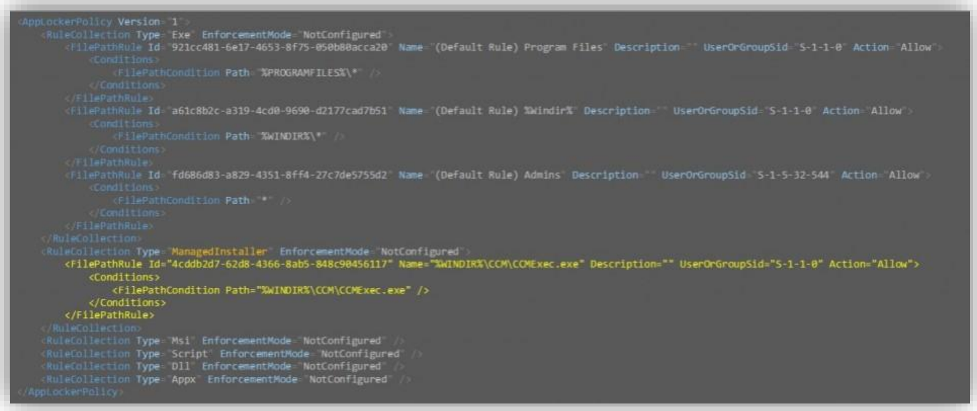
File
Action
View
Help
Security Settings
Account Policies
Local Policies
Windows Firewall with Advanced Seci
Network List Manager Policies
Public Key Policies
Software Restriction Policies
Application Control Policies
AppLocker
IP Secu
Advanc
Import Policy...
Export Policy...
Clear Policy
View
Properties
Help
More about AppLocker
Which editions of Windows support AppLocker?
Configure rule enforcement
More about rule enforcement
Overview

The exported policy XML will look similar to the example below. The new file rule for CCMExec.exe is highlighted in yellow.



```xml
<AppLockerPolicy Version="1">
    <RuleCollection Type="Exe" EnforcementMode="NotConfigured">
        <FilePathRule Id="921cc481-6e17-4653-8f75-050b80acca20" Name="(Default Rule) Program Files" Description="" UserOrGroupSid="S-1-1-0" Action="Allow">
            <Conditions>
                <FilePathCondition Path="%PROGRAMFILES%\*" />
            </Conditions>
        </FilePathRule>
        <FilePathRule Id="a61c8b2c-a319-4cd0-9690-d2177cad7b51" Name="(Default Rule) %Windir%" Description="" UserOrGroupSid="S-1-1-0" Action="Allow">
            <Conditions>
                <FilePathCondition Path="%WINDIR%\*" />
            </Conditions>
        </FilePathRule>
        <FilePathRule Id="fd686d83-a829-4351-8ff4-27c7de5755d2" Name="(Default Rule) Admins" Description="" UserOrGroupSid="S-1-5-32-544" Action="Allow">
            <Conditions>
                <FilePathCondition Path="*" />
            </Conditions>
        </FilePathRule>
        <FilePathRule Id="4cddb2d7-62d8-4366-8ab5-848c90456117" Name="%WINDIR%\CCM\CCMExec.exe" Description="" UserOrGroupSid="S-1-1-0" Action="Allow">
            <Conditions>
                <FilePathCondition Path="%WINDIR%\CCM\CCMExec.exe" />
            </Conditions>
        </FilePathRule>
    </RuleCollection>
    <RuleCollection Type="Msi" EnforcementMode="NotConfigured" />
    <RuleCollection Type="Script" EnforcementMode="NotConfigured" />
    <RuleCollection Type="Dll" EnforcementMode="NotConfigured" />
    <RuleCollection Type="Appx" EnforcementMode="NotConfigured" />
</AppLockerPolicy>
```

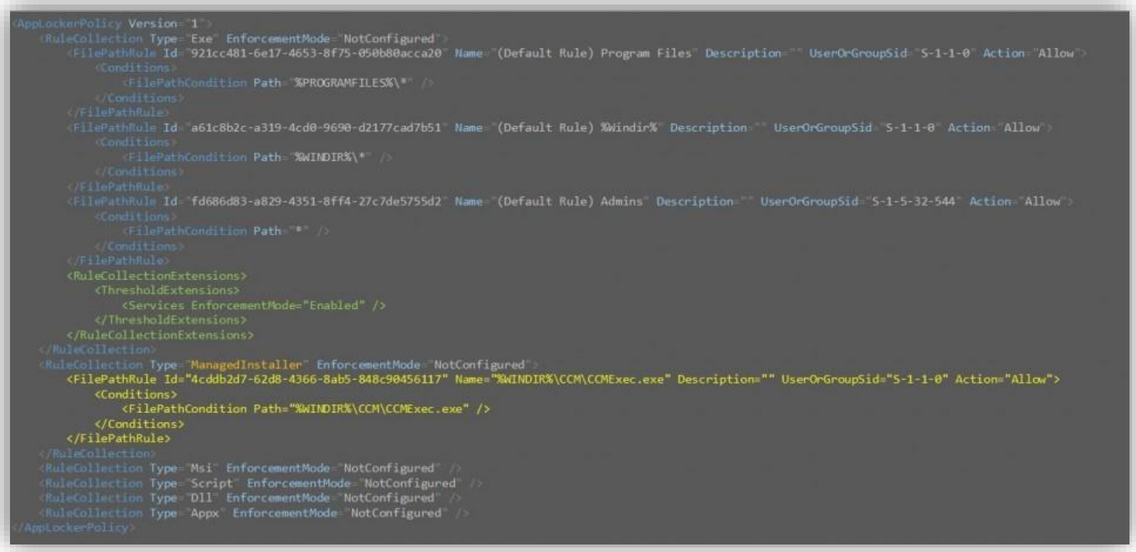
4. Next, duplicate the entire EXE rule collection via copy-paste, and then remove all rules other than the new file path rule in the duplicate version. The original CCMExec.exe file path rule in the EXE rule collection can also be deleted at this point. Change the value of the **Type** attribute on the new rule collection to **ManagedInstaller**. What remains is a new Rule Collection of type **ManagedInstaller** and an EXE rule collection that contains only the original (in this case default) rules.



```xml
<AppLockerPolicy Version="1">
    <RuleCollection Type="Exe" EnforcementMode="NotConfigured">
        <FilePathRule Id="921cc481-6e17-4653-8f75-050b80acca20" Name="(Default Rule) Program Files" Description="" UserOrGroupSid="S-1-1-0" Action="Allow">
            <Conditions>
                <FilePathCondition Path="%PROGRAMFILES%\*" />
            </Conditions>
        </FilePathRule>
        <FilePathRule Id="a61c8b2c-a319-4cd0-9690-d2177cad7b51" Name="(Default Rule) %Windir%" Description="" UserOrGroupSid="S-1-1-0" Action="Allow">
            <Conditions>
                <FilePathCondition Path="%WINDIR%\*" />
            </Conditions>
        </FilePathRule>
        <FilePathRule Id="fd686d83-a829-4351-8ff4-27c7de5755d2" Name="(Default Rule) Admins" Description="" UserOrGroupSid="S-1-5-32-544" Action="Allow">
            <Conditions>
                <FilePathCondition Path="*" />
            </Conditions>
        </FilePathRule>
    </RuleCollection>
    <RuleCollection Type="ManagedInstaller" EnforcementMode="NotConfigured">
        <FilePathRule Id="4cddb2d7-62d8-4366-8ab5-848c90456117" Name="%WINDIR%\CCM\CCMExec.exe" Description="" UserOrGroupSid="S-1-1-0" Action="Allow">
            <Conditions>
                <FilePathCondition Path="%WINDIR%\CCM\CCMExec.exe" />
            </Conditions>
        </FilePathRule>
    </RuleCollection>
    <RuleCollection Type="Msi" EnforcementMode="NotConfigured" />
    <RuleCollection Type="Script" EnforcementMode="NotConfigured" />
    <RuleCollection Type="Dll" EnforcementMode="NotConfigured" />
    <RuleCollection Type="Appx" EnforcementMode="NotConfigured" />
</AppLockerPolicy>
```

5. Now that the Managed installer rule collection has been created, the Services Enforcement extension that was introduced in the first release of Windows 10 must be added. To add the extension that allows for the enforcement of AppLocker policies against Windows Services, paste the below into your policy inside the EXE rule collection. You can see the result highlighted in green in the below. Insert this text:

```
<RuleCollectionExtensions>
    <ThresholdExtensions>
        <Services EnforcementMode="Enabled" />
    </ThresholdExtensions>
</RuleCollectionExtensions>
```



```
<AppLockerPolicy Version="1">
  <RuleCollection Type="Exe" EnforcementMode="NotConfigured">
    <FilePathRule Id="921cc481-6e17-4653-8f75-050b80acca20" Name="(Default Rule) Program Files" Description="" UserOrGroupSid="S-1-1-0" Action="Allow">
      <Conditions>
        <FilePathCondition Path="%PROGRAMFILES%\*" />
      </Conditions>
    </FilePathRule>
    <FilePathRule Id="a61c8b2c-a319-4cd0-9690-d2177cad7b51" Name="(Default Rule) %Windir%" Description="" UserOrGroupSid="S-1-1-0" Action="Allow">
      <Conditions>
        <FilePathCondition Path="%WINDIR%\*" />
      </Conditions>
    </FilePathRule>
    <FilePathRule Id="fd686d83-a829-4351-8ff4-27c7de5755d2" Name="(Default Rule) Admins" Description="" UserOrGroupSid="S-1-5-32-544" Action="Allow">
      <Conditions>
        <FilePathCondition Path="*" />
      </Conditions>
    </FilePathRule>
    <RuleCollectionExtensions>
      <ThresholdExtensions>
        <Services EnforcementMode="Enabled" />
      </ThresholdExtensions>
    </RuleCollectionExtensions>
  </RuleCollection>
  <RuleCollection Type="ManagedInstaller" EnforcementMode="NotConfigured">
    <FilePathRule Id="4cddb2d7-62d8-4366-8ab5-848c90456117" Name="%WINDIR%\CCM\CCMExec.exe" Description="" UserOrGroupSid="S-1-1-0" Action="Allow">
      <Conditions>
        <FilePathCondition Path="%WINDIR%\CCM\CCMExec.exe" />
      </Conditions>
    </FilePathRule>
  </RuleCollection>
  <RuleCollection Type="Msi" EnforcementMode="NotConfigured" />
  <RuleCollection Type="Script" EnforcementMode="NotConfigured" />
  <RuleCollection Type="Dll" EnforcementMode="NotConfigured" />
  <RuleCollection Type="Appx" EnforcementMode="NotConfigured" />
</AppLockerPolicy>
```

6. Finally, select the Enforcement mode for the EXE and Managed Installed rule collections. The possible options are "Notconfigured", "AuditOnly", or "Enabled". They have the following significance:

   NotConfigured – No enforcement or auditing occurs.

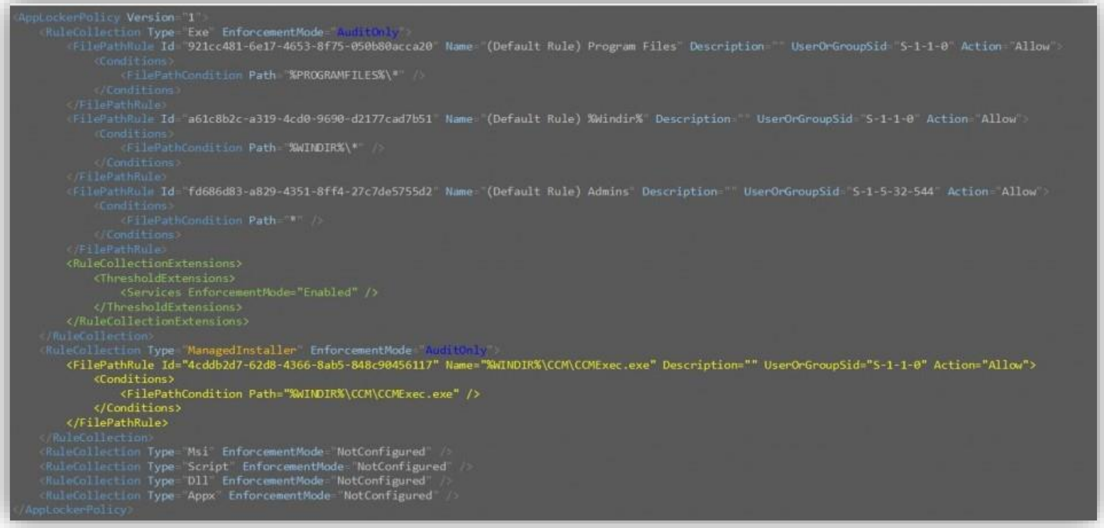
   AuditOnly – Applications and executables are not blocked from running by AppLocker, but logging occurs in the client event logs (visible in Event Viewer under *Applications and Service Logs > Microsoft > Windows > AppLocker*) whenever an application or executable is allowed to run or work have been blocked if enforcement mode had been enabled.

   Enabled – Applications and executables in violation of the AppLocker policies are blocked from running. **Note**: The Managed Installer rule will only be enforced for software that is not allowed by existing EXE rules and that was not already present on the device when the Application Identity service was started (see Configuring Client Devices below)

The recommended way of configuring AppLocker is to set up your policy and first set the enforcement mode to **AuditOnly** and then examine the event logs on the client machine to assess whether the policy is working correctly. Once the correctness of the policy has been adequately verified, then enforcement mode can be changed to **enabled**.

To complete this example, the policy enforcement mode will be changed to **AuditOnly**. The change is highlighted in blue.

With this final change the policy is ready to be saved and subsequently deployed. Once the policy has been validated and client event logs appear to be exhibiting the desired behavior, then the values of **EnforcementMode** highlighted below in blue can be changed to **Enabled** to enforce the new AppLocker policy (the policy must also be redeployed for the changes to take effect).



```
<AppLockerPolicy Version="1">
    <RuleCollection Type="Exe" EnforcementMode="AuditOnly">
        <FilePathRule Id="921cc481-6e17-4653-8f75-050b80acca20" Name="(Default Rule) Program Files" Description="" UserOrGroupSid="S-1-1-0" Action="Allow">
            <Conditions>
                <FilePathCondition Path="%PROGRAMFILES%\*" />
            </Conditions>
        </FilePathRule>
        <FilePathRule Id="a61c8b2c-a319-4cd0-9690-d2177cad7b51" Name="(Default Rule) %Windir%" Description="" UserOrGroupSid="S-1-1-0" Action="Allow">
            <Conditions>
                <FilePathCondition Path="%WINDIR%\*" />
            </Conditions>
        </FilePathRule>
        <FilePathRule Id="fd686d83-a829-4351-8ff4-27c7de5755d2" Name="(Default Rule) Admins" Description="" UserOrGroupSid="S-1-5-32-544" Action="Allow">
            <Conditions>
                <FilePathCondition Path="*" />
            </Conditions>
        </FilePathRule>
        <RuleCollectionExtensions>
            <ThresholdExtensions>
                <Services EnforcementMode="Enabled" />
            </ThresholdExtensions>
        </RuleCollectionExtensions>
    </RuleCollection>
    <RuleCollection Type="ManagedInstaller" EnforcementMode="AuditOnly">
        <FilePathRule Id="4cddb2d7-62d8-4366-8ab5-848c90456117" Name="%WINDIR%\CCM\CCMExec.exe" Description="" UserOrGroupSid="S-1-1-0" Action="Allow">
            <Conditions>
                <FilePathCondition Path="%WINDIR%\CCM\CCMExec.exe" />
            </Conditions>
        </FilePathRule>
    </RuleCollection>
    <RuleCollection Type="Msi" EnforcementMode="NotConfigured" />
    <RuleCollection Type="Script" EnforcementMode="NotConfigured" />
    <RuleCollection Type="Dll" EnforcementMode="NotConfigured" />
    <RuleCollection Type="Appx" EnforcementMode="NotConfigured" />
</AppLockerPolicy>
```

Configuring Client Devices

Four steps are required to configure clients to treat ConfigMgr as a Managed Installer. In this example, a short PowerShell script is used and can be deployed in a package containing both the script and the AppLocker policy XML file (sample one here). The script must be run with Administrative privileges to have the desired result. These commands can be run from any folder except for the step to set the AppLocker policy, which needs to be run from the folder where the policy XML file is located.

1. **Start Windows Application Identity services**
   The PowerShell command (in the Command prompt) to accomplish this is as follows:

   ```
   PS C:\WINDOWS\system32> AppIdtel start -mionly
   ```

   Once this command is run, all software that is already on the device will automatically be trusted, regardless of whether or not that software would be allowed by existing AppLocker EXE rules. This means that the policy used could be much shorter than the simple one provided for this example. Software new to the device will be subject to existing AppLocker rules and if it is not allowed by the existing EXE rules then it will be allowed to run only if it was installed by a Managed Installer.

Note: Managed Installer Functionality will never override an explicit "Deny" AppLocker rule, meaning that if these rules exist, the specified software will still not be allowed to run.

2. **Create a custom DWORD in the client registry**
   To configure the ConfigMgr client to behave as a Managed Installer, the following registry DWORD must be added with a value of "1".

   This can be accomplished using reg.exe that can be executed from PowerShell/Command prompt as follows:

   ```
   PS C:\WINDOWS\system32> reg.exe add HKLM\SOFTWARE\Microsoft\CCM /v
   EnableManagedInstaller /t REG_DWORD /d "1" /f
   ```

3. **Deploy the custom AppLocker policy that was created above**
   AppLocker policies are often deployed via Group Policy, but in this example the policy will be applied using one of the AppLocker PowerShell cmdlets to apply policy from the policy XML file distributed in the same package as the script. The command for this is:

   ```
   PS C:\WINDOWS\system32> set-ApplockerPolicy -XmlPolicy AuditPolicy.xml
   ```

4. **Restart the client SMS Host Agent service (CCMExec), or restart the device**
   The final step to configure clients is to restart the CCMExec service that can be accomplished by executing the net.exe command from Command window as follows:

   ```
   PS C:\WINDOWS\system32> net stop ccmexec
   PS C:\WINDOWS\system32> net start ccmexec
   ```

These four sets of commands can be combined into a simple PowerShell script by copying the lines from above into a text file and naming the file with a .ps1 file extension. The resulting script looks like the below.

```
AppIdtel start -mionly
reg.exe add HKLM\SOFTWARE\Microsoft\CCM /v EnableManagedInstaller /t REG_DWORD /d "1" /fset-
ApplockerPolicy -XmlPolicy AuditPolicy.xml
net stop ccmexec
net start ccmexec
```

The above should be saved to a .ps1 PowerShell script file and that file can then be distributed along with the policy XML file created above to be run on clients using a required package and program. Clients that have run the script will treat ConfigMgr as a Managed Installer.

To validate the policy once it has been deployed, normal application, update, and package deployments should be made to the clients (taking the aforementioned caveats into consideration) and then the local client event logs should be examined to ensure that no trusted software is in violation of both the EXE and Managed Installer AppLocker rules. Software that is allowed by at least one of these rules will be allowed to run. Once the policy has been validated, the AppLocker policy should be edited so that **EnforcementMode** is set to “Enabled”, and then the AppLocker policy deployment step (and only this step) should be re-run to update the policy on the client.

Once this is complete then the original goal has been accomplished! The client has been locked down and only existing software and new software deployed from ConfigMgr will be allowed to run on the client device.

Configuration Manager Resources

Documentation for System Center Configuration Manager Technical Previews
Documentation for System Center Configuration Manager
System Center Configuration Manager Forums
System Center Configuration Manager Support
System Center Configuration Manager Technical Preview 5 (v1603)

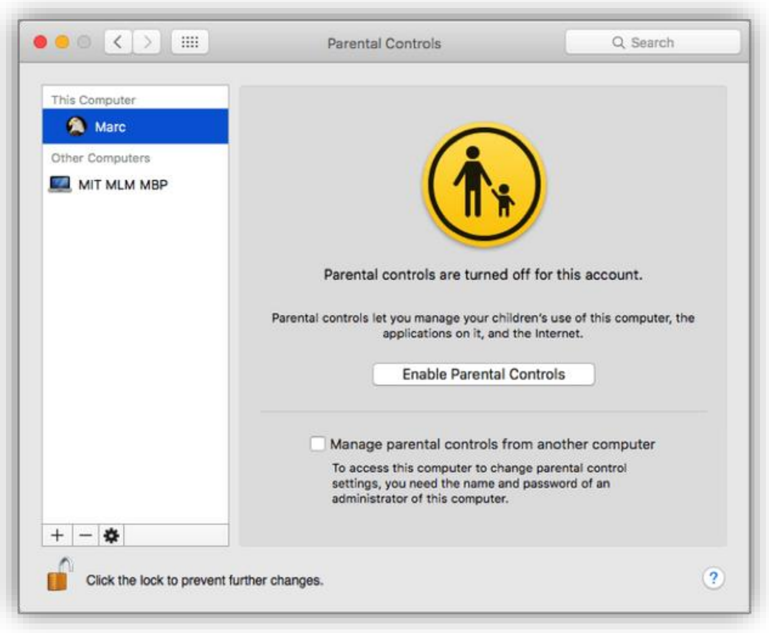
Mac OS 10.5 Leopard

In Mac OS X and above, application whitelisting is done by logging in with a Managed user account with Parental Controls enabled.

To configure your account to have the added security of application whitelisting, the following steps should be taken for all non-administrative accounts on a computer running Mac OS X and above.

1. On the computer hosting the user account to be managed, open Apple menu > System Preferences > Parental Controls.

2. Select the Lock icon to authenticate as an administrator
3. In the sidebar select the target account to manage
4. If you want to manage parental controls from another computer on the same network, enable the Manage parental controls from another computer checkbox.



Parental Controls

This Computer
Marc
Other Computers
MIT MLM MBP

Parental controls are turned off for this account.

Parental controls let you manage your children's use of this computer, the applications on it, and the Internet.

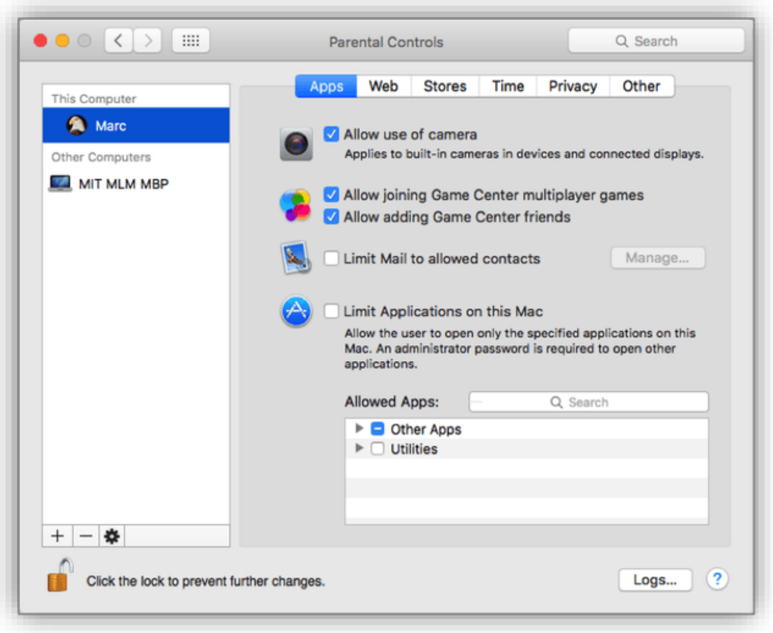
Enable Parental Controls

☐ Manage parental controls from another computer

To access this computer to change parental control settings, you need the name and password of an administrator of this computer.

Click the lock to prevent further changes.

5. Select the Enable Parental Controls button. The Parental Controls System Preference pane opens. Unlock the pane.



Parental Controls

Apps | Web | Stores | Time | Privacy | Other

This Computer
Marc
Other Computers
MIT MLM MBP

☑ Allow use of camera
Applies to built-in cameras in devices and connected displays.

☑ Allow joining Game Center multiplayer games
☑ Allow adding Game Center friends

☐ Limit Mail to allowed contacts — Manage...

☐ Limit Applications on this Mac
Allow the user to open only the specified applications on this Mac. An administrator password is required to open other applications.

Allowed Apps:
Other Apps
Utilities

Click the lock to prevent further changes.

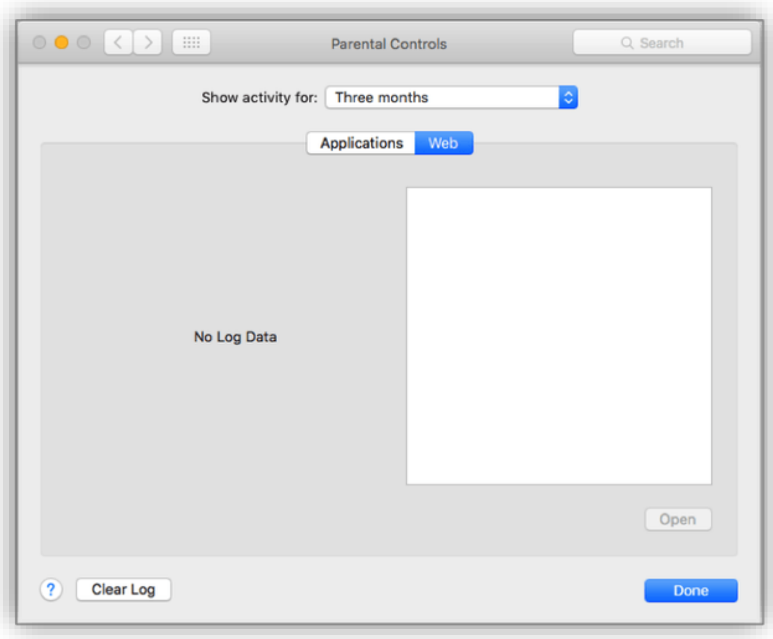
Logs...

-Allow use of camera is self-explanatory
-Allow joining Game Center multiplayer games is self-explanatory (in business, should probably be unchecked, but that is up to the organisation)
-Allow adding Game Center friends is self-explanatory (again, in business, should probably be unchecked, but that is up to the organisation)
-Limit Mail to allowed contacts helps to prevent unknown and unwanted people from exchanging email with the user. Selecting the Manage button opens a configuration window for this option (in business, this option should not be checked)
-Limit Applications activates the application whitelisting. It allows picking which specific application the account will have access to

6. Expand Other Apps. Enable the checkbox for applications this account needs, but do not enable the Other Apps checkbox as this will allow any application to run. Keep in mind we are attempting to prevent unwanted malware from launching.

7. Enable the Utilities checkbox. A non-administrator is not going to create problems accessing these applications.

8. By selecting the Logs button, the administrator is able to view the activities of the managed user. Logs may be viewed from any other computer on the same network.



9. Select the Done button to return to Parental Controls.

If all you want to accomplish is to enable Parental Controls, your job is done! However, if you want to further restrict activities and access to this system, you can return to edit the Web, Stores, Time, Privacy, and Other tabs.

Final Note

SECMON1 are available to review your application whitelisting configurations with you and advise on troubleshooting as well as potential improvements and solutions. We are also available to implement application whitelisting for your company.

Please feel free to visit our website at www.secmon1.com.

You can also reach us by phone at 1300 410 900 or by e-mail at contact@secmon1.com.