



- 8 -

# Defender for Office 365

---

## In This Chapter

- Introduction
- General MDO Cmdlets
- Email & Collaboration Policies
- Tenant Allow / Block List Items
- Advanced Delivery
- Enhanced Filtering
- ORCA
- Configuration Analyzer
- Defender Evaluation

---

## Introduction

Microsoft released Advance Threat Protection (ATP) as a premium feature for their Office 365 product line. With time comes change as ATP has now morphed into the newly (as of 2020) renamed Microsoft Defender for Office 365 (MDO) which is part of the larger Microsoft Defender Suite that was introduced at Ignite in 2020. One thing to keep in mind is that it is designed to do many things. It was not designed to just protect Exchange, but was intended to be a more holistic product to protect the suite of products that is Office 365. A whole slew of products have been released to bolster this approach:

- Safe Links
- Safe Attachments
- Anti-Phish
- ... and more ...

In the background of all of these features, there are PowerShell cmdlets to help us configure, manage and utilize these features of Office 365. In this section we'll start with some overarching reporting cmdlets and then dive into each feature that MDO provides and that we have PowerShell for. Note that some features that are part of the MDO licensing are NOT going to be covered. This is because either they are in another workload (Security and Compliance Center) or there are no relevant PowerShell cmdlets at the moment in the Exchange Online module.

One thing to note when it comes to MDO and Office 365, there is specific licensing that needs to be in place in order to expose the features and thus the PowerShell cmdlets in the module. Make sure to read this page about licensing requirements and what features are exposed depending on what license is purchased:

<https://docs.microsoft.com/en-us/office365/servicedescriptions/office-365-advanced-threat-protection-service-description>

## General MDO Cmdlets

While the naming of the product has changed, the background PowerShell cmdlet have not. All cmdlets for base

configurations have ATP in them still. The Exchange Online PowerShell module includes some general reporting cmdlets that may be helpful in getting an idea of what is configured, what is flowing through the tenant and more. Let's see what cmdlets we have available to explore this aspect of an Exchange Online tenant.

Make sure to connect with the Exchange Online v2 PowerShell Module first:

```
Connect-ExchangeOnline
```

Cmdlets: (We need to specify the source as other cmdlets get displayed that are not relevant.)

```
Get-Command *atp* | Where {$_.Source -eq 'tmp_ielfbchc.2s1'}
```

We get a list of cmdlets:

Add-ATPEvaluation	Disable-ATPProtectionPolicyRule
Enable-ATPProtectionPolicyRule	Get-AdvancedThreatProtectionDocumentDetail
Get-AdvancedThreatProtectionDocumentReport	Get-AdvancedThreatProtectionTrafficReport
Get-ATPEvaluation	Get-AtpPolicyForO365
Get-ATPProtectionPolicyRule	Get-ATPTotalTrafficReport
Get-MailDetailATPReport	Get-MailTrafficATPReport
New-ATPProtectionPolicyRule	Remove-ATPEvaluation
Remove-ATPProtectionPolicyRule	Set-AtpPolicyForO365
Set-ATPProtectionPolicyRule	

Now that we have a list of cmdlets we can work with, let's dive right in.

### Get-AtpPolicyForO365

This cmdlet will display the current configuration for Office 365 and its MDO features: (below are defaults):

```
AdminDisplayName      :
TrackClicks           : False
AllowClickThrough     : False
EnableSafeLinksForClients : False
EnableSafeLinksForWebAccessCompanion : False
EnableSafeLinksForO365Clients : True
BlockUrls             : {}
EnableATPForSPOTeamsODB : False
EnableSafeDocs       : False
AllowSafeDocsOpen    : False
Identity              : Default
Id                    : Default
IsValid               : True
ExchangeVersion      : 0.20 (15.0.0.0)
Name                  : Default
```

**NOTE:** There aren't a lot of parameters with the Get-AtpPolicyForO365 cmdlet. So just run Get-AtpPolicyForO365 as is to get the results above.

Once we have turned on the features present for ATP, we would see something like below:

```
AdminDisplayName      :
TrackClicks          : True
AllowClickThrough     : False
EnableSafeLinksForClients : True
EnableSafeLinksForWebAccessCompanion : True
EnableSafeLinksForO365Clients : True
BlockUrls             : {newsweek.ga}
EnableATPForSPOTeamsODB : True
EnableSafeDocs       : False
AllowSafeDocsOpen    : False
Identity             : Default
Id                   : Default
IsValid              : True
ExchangeVersion      : 0.20 (15.0.0.0)
Name                  : Default
```

How do we configure those settings? Use the `Set-AtpPolicyForO365` cmdlet.

### Set-AtpPolicyForO365

From our above defaults, it looks like there are quite a few features that are turned off by default.

**AllowClickThrough:** If a URL is blocked, setting this value to `$True` allows a user to still click through to the original blocked URL.

**BlockUrls:** Specify one or more URLs to be blocked by Safe Links.

**EnableATPForSPOTeamsODB:** `$True` enables ATP for SharePoint Online, OneDrive for Business, and Microsoft Teams.

**EnableSafeLinksForClients:** Enable or disable Safe Links for Office 365 Pro Plus Clients.

**TrackClicks:** Set to `$True` to track the clicks a user makes on blocked URLs.

These options seem pretty straight forward. For a sample configuration, we would like to enable Safe Links for email, track any clicks that may occur and make sure to block any click-throughs for blocked URLs. We can do that with this one-liner:

```
Set-AtpPolicyForO365 -AllowClickThrough $False -EnableSafeLinksForClients $True -TrackClicks $True
```

Which we can validate by running `Get-AtpPolicyForO365`:

```
TrackClicks          : True
AllowClickThrough     : False
EnableSafeLinksForClients : True
EnableSafeLinksForWebAccessCompanion : False
EnableSafeLinksForO365Clients : True
BlockUrls             : {}
EnableATPForSPOTeamsODB : False
EnableSafeDocs       : False
AllowSafeDocsOpen    : False
Identity             : Default
```

We could have also enabled Safe Links for OneDrive, SharePoint and Teams if we wanted to protect those as well.

## ATP Protection Policy Rule

By default there are no ATP Protection Policy Rules in place. These rules can be created in two ways, one is to apply either the Standard or Strict settings Microsoft documents [ [here](#) ] or we can create them with PowerShell. In order to apply the Standard or Strict settings, we first need to visit the Security and Compliance Center [ [here](#) ] and browse to the Security and Compliance Center --> Threat Management --> Policy --> Preset security policies:

Home > Policy > Preset security policies

### Preset security policies

A preset security policy is compilation of settings for these security policies: anti-spam, anti-malware, anti-phishing, Safe Links, and Safe Attachments.

When multiple security policies are applied a user, the Strict policy overrides the Standard policy and any custom policies. The Standard policy overrides custom policies. [Learn more](#)

<b>Standard protection</b>	<b>Strict protection</b>
Apply a baseline protection profile that's suitable for most users.	Apply a more aggressive protection profile for selected users.

From this interface we can apply Standard or Strict policies based on a set of conditions that include verified tenant domains, groups or individual users.

### ATP protections apply to

The following policies are considered ATP protections: ATP a [about the best-practices for configuring EOP and Office 365](#)

^ The recipients domains are

Any of these

powershellgeek.com X

Once a policy is applied, we can see this in Exchange Online PowerShell with the `Get-ATPProtectionPolicyRule` cmdlet. We can also see the mapped policies to the rule:

```
PS C:\> Get-SafeLinksPolicy |ft Identity
Identity
-----
Damian@PracticalPowerShell.com
brian@practicalpowershell.com
Standard Preset Security Policy
Strict Preset Security Policy

PS C:\> Get-SafeAttachmentPolicy
Name                                     Action      Enable IsDefault
----
Company Wide Safe Attachments           Block       True   False
Standard Preset Security Policy         Block       True   False
Strict Preset Security Policy           Block       True   False
Safe Attachments - Practical PowerShell DynamicDelivery False      False

PS C:\> Get-ATPProtectionPolicyRule
RunspaceId                               : b681c214-bbfc-4b31-a2e1-d0495c5c5cbe
SafeAttachmentPolicy                      : Standard Preset Security Policy
SafeLinksPolicy                           : Standard Preset Security Policy
State                                      : Enabled
```

These Safe Links Policies and Safe Attachment Policies are pre-built and ready to be assigned like the above ATP

Protection Policy Rule.

### New-ATPProtectionPolicyRule

We can reverse engineer the settings by looking at the ATP Protection Rule we created in the Preset Security wizard. We see we have a name, a Safe Links Policy and a Safe Attachments Policy. We can use this information to create a one-liner:

```
New-ATPProtectionPolicyRule -Name 'Standard Preset Security Policy' -RecipientDomains
'PowerShellGeek.com' -SafeAttachmentPolicy 'Standard Preset Security Policy' -SafeLinksPolicy
'Standard Preset Security Policy'
```

Which creates the Policy for us:

```
RunspaceId      : b681c214-bbfc-4b31-a2e1-d0495c5c5cbe
SafeAttachmentPolicy : Standard Preset Security Policy
SafeLinksPolicy  : Standard Preset Security Policy
State           : Enabled
Priority        : 0
Comments       :
Description     : If the message:
                  recipients's address domain portion belongs to any of these domains:
                  'PowerShellGeek.com'
                  Take the following actions:
                  Apply safe attachment policy "Standard Preset Security Policy"., Apply safe
                  links policy "Standard Preset Security Policy".

RuleVersion     : 15.0.5.2
SentTo         :
SentToMemberOf :
RecipientDomainIs : {PowerShellGeek.com}
```

The change is also presented in the Security and Compliance Center under Preset Security Policies:

**Advanced Threat Protection applies to**

If the recipient domain is:  
PowerShellGeek.com

What other options do we have for creating a new ATP policy here?

**SentToMembersOf:** Specify a group of mailboxes to protect with this policy.

**SentTo:** Specify individual mailboxes to be protected.

**SafeAttachmentPolicy:** Specify the Standard or Strict Preset Security Policy.

**SafeLinksPolicy:** Specify the Standard or Strict Preset Security Policy.

**ExceptIfRecipientDomainIs:** Exclude a particular SMTP domain from the policy settings.

**ExceptIfSentTo:** Exclude individual mailboxes from the policy settings

**ExceptIfSentToMemberOf:** Exclude groups from the policy settings.

**Comments:** Add appropriate comments about the new Policy Rule.

### Scenario

Let's say that we have a company whose primary domain is PowerShellGeek.com and they deal in custom PowerShell application for their Office 365 customers. They want to set up two layers of policies where all users get at least the Standard Protection Policy, but select groups get the Strict Protection Policy. How would we layer these policies so that we don't create conflicts? We can create one Standard policy for all users and use the 'ExceptIfSentToMemberOf' to block the policy for those select groups. Then we create a second policy using the 'SentToMemberOf' and choose the Strict Protection Policy.

**PowerShell code:****Standard Policy**

```
New-ATPProtectionPolicyRule -Name 'Corp-Standard-ATP-Policy' -RecipientDomainIs 'PowerShellGeek.com' -SafeAttachmentPolicy 'Standard Preset Security Policy' -SafeLinksPolicy 'Standard Preset Security Policy' -ExceptIfSentToMemberOf 'Excluded-ATP-Standard-Group'
```

**Strict Policy**

```
New-ATPProtectionPolicyRule -Name 'Corp-Strict-ATP-Policy' -RecipientDomainIs 'PowerShellGeek.com' -SafeAttachmentPolicy 'Strict Preset Security Policy' -SafeLinksPolicy 'Strict Preset Security Policy' -SentToMemberOf 'Excluded-ATP-Standard-Group'
```

We now have two complementing policies with no overlap: *[note the group mentioned in each policy]*

```
SafeAttachmentPolicy      : Standard Preset Security Policy
SafeLinksPolicy           : Standard Preset Security Policy
State                     : Enabled
SentTo                    :
SentToMemberOf            :
RecipientDomainIs         : {PowerShellGeek .com}
ExceptIfSentTo            :
ExceptIfSentToMemberOf    : {Excluded-ATP-Standard-Group@PowerShellGeek.com}
ExceptIfRecipientDomainIs :
```

```
SafeAttachmentPolicy      : Strict Preset Security Policy
SafeLinksPolicy           : Strict Preset Security Policy
State                     : Enabled
SentTo                    :
SentToMemberOf            : {Excluded-ATP-Standard-Group@PowerShellGeek.com}
RecipientDomainIs         : {PowerShellGeek .com}
ExceptIfSentTo            :
ExceptIfSentToMemberOf    :
ExceptIfRecipientDomainIs :
```

**Exchange Online Protection (EOP) Protection Policy Rule**

Exchange Online Protection is Microsoft's messaging hygiene solution. By default there are no EOP Protection Policy Rules in place. These rule can be created in two ways, one is to apply either the Standard or Strict settings Microsoft documents [ [here](#) ] or we can create them with PowerShell. In order to apply the Standard or Strict settings, we first need to visit the Security and Compliance Center [ [here](#) ] and browse the Security and Compliance Center --> Threat Management --> Policy --> Preset security policies:

[Home](#) > [Policy](#) > Preset security policies

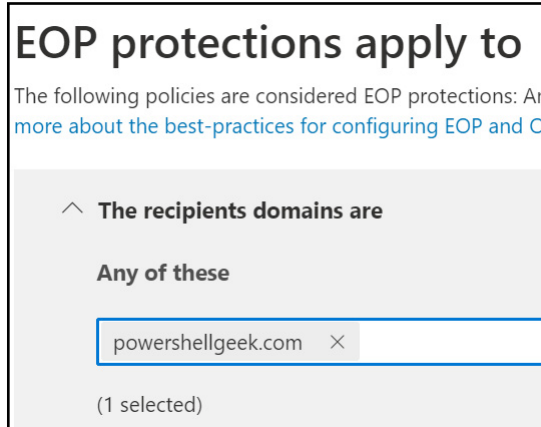
## Preset security policies

A preset security policy is compilation of settings for these security policies: anti-spam, anti-malware, anti-phishing, Safe Links, and Safe Attachments.

When multiple security policies are applied a user, the Strict policy overrides the Standard policy and any custom policies. The Standard policy overrides custom policies. [Learn more](#)

<b>Standard protection</b>	<b>Strict protection</b>
Apply a baseline protection profile that's suitable for most users.	Apply a more aggressive protection profile for selected users.

From this interface we can apply Standard or Strict policies based on a set of conditions that include verified tenant domains, groups or individual users.



Once a policy is applied, we can see this in Exchange Online PowerShell with the `Get-EOPProtectionPolicyRule` cmdlet. We can also see the mapped policies to the rule:

```

PS C:\> Get-HostedConnectionFilterPolicy
Name      Identity
-----
Default   Default

PS C:\> Get-AntiPhishPolicy |ft name
Name
----
Executive Anti-Phishing Protection
Standard Preset Security Policy
Strict Preset Security Policy
Office365 AntiPhish Default

PS C:\> Get-MalwareFilterPolicy
Name
----
Default
Standard Preset Security Policy
Strict Preset Security Policy

RunspaceId          : 982600c0-dddb-453e-a720-5b0929daf3ba
HostedContentFilterPolicy : Standard Preset Security Policy
AntiPhishPolicy      : Standard Preset Security Policy
MalwareFilterPolicy  : Standard Preset Security Policy
State                 : Enabled
  
```

Notice in the screenshot of the EOP Protection Rule that the Hosted Content Filter Policy is listed as 'Standard Present Security Policy' but there is no policy with that name, there is only the Default Hosted Content Filter. For the AntiPhish and Malware policies, there are matching ones we can list with PowerShell. The question is, can we assign all three with PowerShell or will it complain that there is no Hosted Content Filter Policy? Let's try an EOP rule that mimics the ATP rule we created previously, like so:

```

New-EOPProtectionPolicyRule -Name 'Standard Preset Security Policy' -HostedContentFilterPolicy
'Standard Preset Security Policy' -AntiPhishPolicy 'Standard Preset Security Policy' -MalwareFilterPolicy
'Standard Preset Security Policy' -RecipientDomains 'PracticalPowerShell.Com'
  
```

Works!

```

RunspaceId          : 81878ff8-ec42-438e-a2cc-eb81bc3f67c1
HostedContentFilterPolicy : Standard Preset Security Policy
AntiPhishPolicy      : Standard Preset Security Policy
MalwareFilterPolicy  : Standard Preset Security Policy
State                 : Enabled
Priority              : 0
Comments             :
  
```

```

Take the following actions:
    Apply hosted content filter policy '
    AntiPhish policy "Standard Preset Security Policy"
    "Standard Preset Security Policy".

RuleVersion      : 15.0.5.2
SentTo           :
SentToMemberOf   :
RecipientDomainIs : {PracticalPowerShell.Com}

```

Similar to the ATP Rule we have options for SentTo, SentToMemberOf, RecipientDomainIs, ExceptIfSentTo, ExceptIfSentToMemberOf, ExceptIfRecipientDomainIs and Comments.

## Scenario

Taking our example from the ATP section again where we have a company whose primary domain is PracticalPowerShell.com and they deal in custom PowerShell application for their Office 365 customers. They want to set up two layers of EOP policies where all users get at least the Standard Protection Policy, but select groups get the Strict Protection Policy. How would we layer these policies so that we don't create conflicts? We can create a Standard policy for all users and use the 'ExceptIfSentToMemberOf' to block the policy for those select groups.

### Standard Policy

```

New-EOPProtectionPolicyRule -Name 'Corp-Standard-EOP-Policy' -HostedContentFilterPolicy 'Standard Preset Security Policy' -AntiPhishPolicy 'Standard Preset Security Policy' -MalwareFilterPolicy 'Standard Preset Security Policy' -RecipientDomainIs 'PracticalPowerShell.Com' -ExceptIfSentToMemberOf 'Excluded-ATP-Standard-Group'

```

### Strict Policy

```

New-EOPProtectionPolicyRule -Name 'Corp-Strict-EOP-Policy' -HostedContentFilterPolicy 'Strict Preset Security Policy' -AntiPhishPolicy 'Strict Preset Security Policy' -MalwareFilterPolicy 'Strict Preset Security Policy' -RecipientDomainIs 'PracticalPowerShell.Com' -SentToMemberOf 'Excluded-ATP-Standard-Group'

```

We now have two complementing policies with no overlap: *[note the group mentioned in each policy]*

```

HostedContentFilterPolicy : Strict Preset Security Policy
AntiPhishPolicy           : Strict Preset Security Policy
MalwareFilterPolicy       : Strict Preset Security Policy
State                     : Enabled
SentTo                    :
SentToMemberOf            : {Excluded-ATP-Standard-Group@PracticalPowerShell.Com}
RecipientDomainIs         : {PracticalPowerShell.Com}
ExceptIfSentTo            :
ExceptIfSentToMemberOf    :
ExceptIfRecipientDomainIs :

```

```

HostedContentFilterPolicy : Standard Preset Security Policy
AntiPhishPolicy           : Standard Preset Security Policy
MalwareFilterPolicy       : Standard Preset Security Policy
State                     : Enabled
SentTo                    :
SentToMemberOf            :
RecipientDomainIs         : {PracticalPowerShell.Com}
ExceptIfSentTo            :
ExceptIfSentToMemberOf    : {Excluded-ATP-Standard-Group@PracticalPowerShell.Com}
ExceptIfRecipientDomainIs :

```



## Email & Collaboration Policies

### Introduction

In addition to the global MDO settings, we also have our individual policies for Spam, Phishing, and other message hygiene components. In this section we will break down each feature and PowerShell that can be used to configure it.

### Anti-Phishing

Social engineering has been around since way before computers were invented. People have been trying to convince people to do things for them before technology made this a bit more automated. Now with email and the Internet we have a phenomenon called Phishing where one party is trying to convince another party to do something - click a link, provide a password or some other task in order to retrieve some information. Microsoft realized this was an issue when it built out the Anti-Phishing features that are present in Office 365. Anti-Phishing is also a feature that is part of the ATP set from Microsoft. Same licensing applied to Anti-Phishing as it does to any other portion of the ATP suite.

Behind the scenes Microsoft is using machine learning combined with impersonation algorithms in order to detect these types of messages. An incoming message is analyzed against a series of tests on it's way to the recipients mailbox. What happens to the message is determined by a combination of Microsoft's built-in filters and your own Anti-Phishing policies that are configured.

For the part that we can configure in Office 365, let's explore the PowerShell side of the configuration.

### PowerShell cmdlets

First, we need a list of cmdlets to work with to see what we can do in PowerShell:

```
Get-Command *AntiPhish*
```

This reveals the following list of cmdlets:

Disable-AntiPhishRule	Enable-AntiPhishRule
Get-AntiPhishPolicy	Get-AntiPhishRule
New-AntiPhishPolicy	New-AntiPhishRule
Remove-AntiPhishPolicy	Remove-AntiPhishRule
Set-AntiPhishPolicy	Set-AntiPhishRule

First, we will start with an Anti-Phish Policy. If we run the Get-AntiPhishPolicy we see that there is a default policy already in place called 'Office365 AntiPhish Default'. This policy is applied when no other policies match. In fact, we cannot even create a rule that applies to this policy as it isn't allowed:

```
Policy "Office365 AntiPhish Default" is marked as the default. Creating a rule to apply the
default policy is not allowed. The default policy is always applied when none of the rules match.
+ CategoryInfo          : InvalidOperation: (:) [New-AntiPhishRule], OperationNotAllowedExcept
ion
+ FullyQualifiedErrorId : [Server=DM5PR2201MB1051,RequestId=29f16c65-cec0-4d2c-b34a-30d242d001
88,TimeStamp=1/21/2020 12:17:55 AM] [FailureCategory=Cmdlet-OperationNotAllowedException] 7A94
4598,Microsoft.Exchange.Management.SystemConfigurationTasks.NewAntiPhishRule
+ PSComputerName       : ps.outlook.com
```

Let's go ahead and create our own policy using the New-AntiPhishPolicy cmdlet. What parameters and switches do we have available to us: (... it is a very long list ...)

**AuthenticationFailAction:** If composite authentication (\*) fails, then an action is performed, MoveToJmf (Move to Junk Mail Folder) is the default, while we also have the option to put the message into 'Quarantine' as well.

**EnableAntiSpooofEnforcement:** \$True - Enable anti-spoofing protection (default and recommended setting) / \$False - Disable antispoofing protection.

**EnableAuthenticationSafetyTip:** \$True - Tip is displayed if a message fails composite authentication (\*) (default and recommended setting) / \$False - Tip is not displayed.

**Enabled:** \$True - Policy is enabled / \$False - Policy is disabled.

**EnableMailboxIntelligence:** \$True - Use mailbox intelligence (\*\*) for domain and user impersonation / \$False - Do not use mailbox intelligence (\*\*) in this case (default value).

**EnableMailboxIntelligenceProtection:** \$True - Enables protection based on Mailbox Intelligence / \$False - disables this protection (default value).

**EnableOrganizationDomainsProtection:** \$True - Enables domain impersonation protection for all registered domains in Office 365 / \$False - disables this feature (default value).

**EnableSimilarDomainsSafetyTips:** \$True - Enables safety tips for domain impersonation detections / \$False - disables these safety tips (default value).

**EnableSimilarUsersSafetyTips:** \$True - Enables safety tips when a recipient for user impersonation detections / \$False - disables these safety tips (default).

**EnableTargetedDomainsProtection:** \$True - Enables domain impersonation protection for domains specified in the TargetedDomainsToProtect parameter / \$False - does not use targeted domain protection (default).

**EnableTargetedUserProtection:** \$True - Enables impersonation protection for users specified in the 'TargetedUsersToProtect' Parameter / \$False - disables this feature (default).

**EnableUnauthenticatedSender:** \$True - Adds a '?' to a user's Outlook contact card if it fails authentication checks / \$False - no '?' will be added with authentication failure.

**EnableUnusualCharactersSafetyTips:** \$True - Display safety tip if there is an unusual character in user or domain impersonation / \$False - No safety tip displayed (default).

**ExcludedDomains:** Domains that are excluded from the scanning process.

**ExcludedSenders:** Senders who are excluded from the scanning process.

**ImpersonationProtectionState:** Valid values are automatic, manual and off. Manual is the default value.

**MailboxIntelligenceProtectionAction:** When a message fails mailbox intelligence, what action is taken to the message - NoAction (default), BccMessage, Delete, MoveToJmf, Quarantine and Redirect.

**MailboxIntelligenceProtectionActionRecipients:** If the previous parameter is set to Redirect or BccMessage, then a recipient needs to be specified to handle this message.

**PhishThresholdLevel:** Machine learning level used in scanning: (1) Standard (default), (2) Aggressive, (3) More Aggressive and (4) Most Aggressive. So values of 1, 2, 3 or 4 are valid.

**PolicyTag:** This value is not specified in the Online help for the cmdlet, but it is a valid parameter that can be chosen when creating a policy. No documentation out there at the moment.

**SimilarUsersSafetyTipsCustomText:** Create your own custom text for User based impersonation.

**TargetedDomainActionRecipients:** Works with the TargetedDomainProtectionAction parameter so specify users to receive BccMessage or Redirected email. Can be one or more addresses.

**TargetedDomainProtectionAction:** When there is a detected domain impersonation, what action is taken to the message - NoAction (default), BccMessage, Delete, MoveToJmf, Quarantine and Redirect.

**TargetedDomainsToProtect:** List of domains to protect. Used in conjunction with the EnableTargetedDomainsProtection parameter.

**TargetedUserActionRecipients:** Used in conjunction with the 'TargetedUserProtectionAction' parameter to

specify which users should be targeted by this policy.

**TargetedUserProtectionAction:** When there is a detected user impersonation, what action is taken to the message - NoAction (default), BccMessage, Delete, MoveToJmf, Quarantine and Redirect.

**TargetedUsersToProtect:** Specifies users to protect when the 'EnableTargetedUserProtection' parameter is set to \$True.

**TreatSoftPassAsAuthenticated:** \$True (default) respect soft pass results / \$False - makes anti-spoofing more restrictive and could result in false positives.

**UnusualCharactersSafetyTipsCustomText:** Create your own custom text for unusual character detection based impersonation.

(\*) **Composite Authentication** is a reference to various Anti-Spam checks - DMARC, SPF, DKIM and other factors that determine if a message has authenticated. See here for more details:

<https://docs.microsoft.com/en-us/microsoft-365/security/office-365-security/anti-spam-message-headers>.

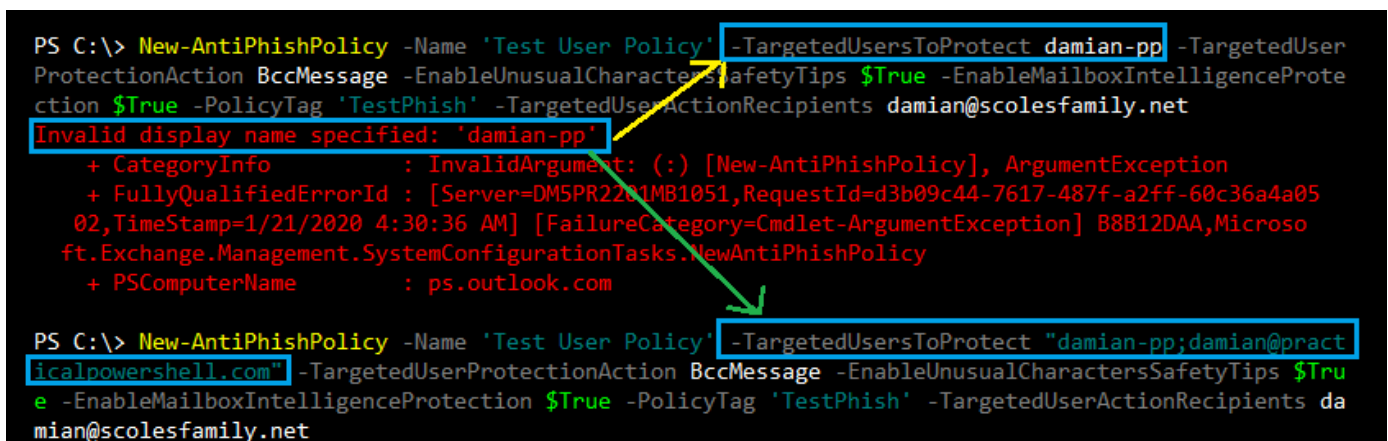
(\*\*) **Mailbox Intelligence** is understanding a users habits and personal contacts. A map of these contacts and their habits is built and used to intelligently make decisions for incoming messages. See here for more details

<https://docs.microsoft.com/en-us/microsoft-365/security/office-365-security/atp-anti-phishing>.

With all of these parameters, we are looking at quite a long list of options to choose from. How do we know what options to pick? What values should be used and is there any guidance on this? Well, we can use the default policy for some guidance as well as the defaults that are provided. For this policy, we will single out one user and apply some settings for testing - Mailbox Intelligence, unusual character safety tips and then user a target action to BCC the message to an IT person for awareness.

```
New-AntiPhishPolicy -Name 'Test User Policy' -TargetedUsersToProtect 'damian@practicalpowershell.com' -TargetedUserProtectionAction BccMessage -EnableUnusualCharactersSafetyTips $True -EnableMailboxIntelligenceProtection $True
```

Now this leads to two different errors. Both could have been avoided with a better examination of documentation. First, the TargetedUsersToProtect value is in the wrong format, hence this error:



```
PS C:\> New-AntiPhishPolicy -Name 'Test User Policy' -TargetedUsersToProtect damian-pp -TargetedUserProtectionAction BccMessage -EnableUnusualCharactersSafetyTips $True -EnableMailboxIntelligenceProtection $True -PolicyTag 'TestPhish' -TargetedUserActionRecipients damian@scolesfamily.net
Invalid display name specified: 'damian-pp'
+ CategoryInfo          : InvalidArgument: (:) [New-AntiPhishPolicy], ArgumentException
+ FullyQualifiedErrorId : [Server=DM5PR2201MB1051,RequestId=d3b09c44-7617-487f-a2ff-60c36a4a0502,TimeStamp=1/21/2020 4:30:36 AM] [FailureCategory=Cmdlet-ArgumentException] B8B12DAA,Microsoft.Exchange.Management.SystemConfigurationTasks.NewAntiPhishPolicy
+ PSComputerName        : ps.outlook.com

PS C:\> New-AntiPhishPolicy -Name 'Test User Policy' -TargetedUsersToProtect "damian-pp;damian@practicalpowershell.com" -TargetedUserProtectionAction BccMessage -EnableUnusualCharactersSafetyTips $True -EnableMailboxIntelligenceProtection $True -PolicyTag 'TestPhish' -TargetedUserActionRecipients damian@scolesfamily.net
```

As we can see, the correct value is “<displayname>;<emailaddress>”. Next, we forgot that when we specify a Redirect or BccMessage, we also need to specify the recipient of that message. Hence we have this error message and need to find the correct parameter. The error specified that the RedirectToRecipients is incorrect. Unfortunately this is the correct value.

However, if we review the list of parameters, we realize that we needed to set the ‘TargetedUserActionRecipients’:

```
PS C:\> New-AntiPhishPolicy -Name 'Test User Policy' -TargetedUsersToProtect 'damian@practicalpowershell.com' -TargetedUserProtectionAction BccMessage -EnableUnusualCharactersSafetyTips $True -EnableMailboxIntelligenceProtection $True -PolicyTag 'TestPhish'
TargetedUserProtectionAction: The Redirect action requires the RedirectToRecipients value to be set.
+ CategoryInfo          : NotSpecified: (scoles.onmicrosoft.com\Test User Policy:ADObjectId) [New-AntiPhishPolicy], DataValidationException
+ FullyQualifiedErrorId : [Server=DMSPR2201MB1051,RequestId=7c435246-1dce-42f8-985b-8cb843a6c180,TimeStamp=1/21/2020 4:28:02 AM] [FailureCategory=Cmdlet-DataValidationException] C5E01A0C,Microsoft.Exchange.Management.SystemConfigurationTasks.NewAntiPhishPolicy
+ PSComputerName        : ps.outlook.com

PS C:\> New-AntiPhishPolicy -Name 'Test User Policy' -TargetedUsersToProtect 'damian@practicalpowershell.com' -TargetedUserProtectionAction BccMessage -EnableUnusualCharactersSafetyTips $True -EnableMailboxIntelligenceProtection $True -PolicyTag 'TestPhish' -TargetedUserActionRecipients john@domain.com
```

This would provide a long one-liner, fixing all errors:

```
New-AntiPhishPolicy -Name 'Test User Policy' -TargetedUsersToProtect "damian-pp;damian@practicalpowershell.com" -TargetedUserProtectionAction BccMessage -EnableUnusualCharactersSafetyTips $True -EnableMailboxIntelligenceProtection $True -PolicyTag 'TestPhish' -TargetedUserActionRecipients john@domain.com
```

For this next example, we will protect an entire domain from every setting we can set, we will turn up the aggressiveness to 3 and use all the custom safety tip text that we can get away with:

```
New-AntiPhishPolicy -Name 'Protect Practicalpowershell.com' -TargetedDomainProtectionAction MoveToJmf -TargetedDomainsToProtect 'practicalpowershell.com' -EnableUnusualCharactersSafetyTips $True -EnableSimilarDomainsSafetyTips $True -EnableTargetedDomainsProtection $True -PhishThresholdLevel 3 -UnusualCharactersSafetyTipsCustomText 'Suspicious domain name detected - unusual characters' -SimilarUsersSafetyTipsCustomText 'Suspicious name detected - Similar users'
```

## New-AntiPhishRule

We can use this cmdlet along with the previous New-AntiPhishPolicy to create an operating Anti-Phish mechanism to protect users in our Exchange Online tenant. First, let's see what parameters and switches are available:

**AntiPhishPolicy:** Specify the Anti-Phish Policy that is to be associated with this Rule

**Comments:** Add any relevant comments, possibly a creation date, purposes and/or who created the Rule

**Enabled:** If we choose, we can enable the rule upon creation or disable it upon creation if we do not wish the Rule to be active.

**ExceptIfRecipientDomainIs:** Exception made for a specified domain.

**ExceptIfSentTo:** Exception made for a specified recipient.

**ExceptIfSentToMemberOf:** Exception made for a specified group.

**Priority:** Decide the order in which multiple Safe Attachment Rules are applied.

**RecipientDomainIs:** Exception made for a specified domain.

**SentTo:** Rule applies to specified recipients.

**SentToMemberOf:** Rule applies to specified members of the specified group.

Using the parameters above as well as one of the new Policies, we can create the new Rule to apply to one domain,

perhaps a test domain, like this:

```
New-AntiPhishRule -Name 'Practical PowerShell Anti-Phish Rule' -AntiPhishPolicy 'Protect Practicalpowershell.com' -Enabled $True -RecipientDomains 'practicalpowershell.com'
New-AntiPhishRule -Name 'Test User Practical PowerShell Anti-Phish Rule' -AntiPhishPolicy 'Test User Policy' -Enabled $True -SentTo: 'damian@practicalpowershell.com'
'Test User Policy'
```

### Modify Existing Policies and Policies

With the previous Safe Links and Safe Attachments sections, we could disable the created rules by changing the 'Enable' value to \$False. For Anti-Phishing Rules, we have a pair of cmdlets that we can use to disable rules as well.

*Adding a comment:*

```
Set-AntiPhishRule 'Test User Practical PowerShell Anti-Phish Rule' -Comments 'This Rule applies to test users for validating Anti-Phish scenarios.'
```

*Changing the SentTo is easy as one line as well:*

```
Set-AntiPhishRule 'Test User Practical PowerShell Anti-Phish Rule' -SentToMemberOf brian@practicalpowershell.com
```

### Remove Existing Policies and Policies

We can also remove the Anti-Phishing Rules and Policies, just like Safe Link/Attachment Rules and Policies. The code is very similar to the previous examples:

```
PS C:\> Get-AntiPhishRule | Remove-AntiPhishRule

Confirm
Are you sure you want to perform this action?
Removing AntiPhish rule "Damian Scoles".
[Y] Yes [A] Yes to All [N] No [L] No to All [?] Help (default is "Y"): a
PS C:\> Get-AntiPhishPolicy | Remove-AntiPhishPolicy
```

```
Confirm
Are you sure you want to perform this action?
Remove AntiPhish policy "Test User Policy".
[Y] Yes [A] Yes to All [N] No [L] No to All [?] Help (default is "Y"): a
```

```
The default AntiPhish Policy cannot be removed.
+ CategoryInfo          : InvalidOperation: (Office365 AntiPhish Default:AntiPhishPolicy) [Remove-AntiPhishPolicy], OperationNotAllowedException
+ FullyQualifiedErrorId : [Server=DMS5PR2201MB1051,RequestId=ef88c6fa-98de-4dea-bfd6-9bdd8f38c50a,TimeStamp=1/21/2020 5:04:51 AM] [FailureCategory=Cmdlet-OperationNotAllowedException] C11ED0E9,Microsoft.Exchange.Management.SystemConfigurationTasks.RemoveAntiPhishPolicy
+ PSComputerName        : ps.outlook.com
```

**NOTE:** We cannot remove the default Anti-Phish Policy. We will receive the above error message.

### Anti-Phishing Further Reading

[Anti-phishing protection in Office 365](#)

[Tune anti-phishing protection in Office 365](#)

[ATP anti-phishing capabilities in Office 365](#)

[Set up Office 365 ATP anti-phishing and anti-phishing policies](#)

[How Office 365 validates the From address to prevent phishing](#)

## Anti-Spam

Beyond the external control above, we have an additional layer of protection called Anti-Spam Policies. Some are configurable in your specific tenant, while many others are inherent to the service and are not exposed for configuration purposes. Similar to how other third-party providers will scan messages as they arrive at their servers, will scan messages looking for known payloads, attack patterns and more. In the Anti-Spam section of the Security Center, we see these types of filtering protections:

- **Connection Filters** - Consists of IP Allow and IP Block lists for controlling who can send email to you.
- **Inbound Spam** - Provides quite a few configurable options like Bulk Spam controls, domain/email block and allow lists, international SPAM controls, as well as a set of more advanced controls.
- **Outbound Spam** - Configure notifications if outbound Spam is detected.

Now that we know the available configurable options are, how do we modify or configure these settings with PowerShell?

### Connection Filters

Connection Filters are used to specifically block or allow certain email server IPs from connecting to your Office 365 tenant. These types of filters are good for either allowing a trusted source, like a partner organization, to connect or to block a server that is either currently sending Spam to your tenant or has done so in the past. Microsoft already has extensive lists for blocked IPs due to its own internal parameters. You are more likely to use an allowed IP connections than block when using EOP.

### PowerShell

In order to configure these settings, we first need to find the cmdlets needed in PowerShell that will allow us to do so. We'll use the noun phrase 'ConnectionFilter' to find the relevant cmdlets:

```
Get-Command *ConnectionFilter*
```

```
Name
----
Get-HostedConnectionFilterPolicy
New-HostedConnectionFilterPolicy
Remove-HostedConnectionFilterPolicy
Set-HostedConnectionFilterPolicy
```

First, if we run the `Get-HostedConnectionFilterPolicy` we see that the default policy is not configured with any settings:

```
Name: Get-HostedConnectionFilterPolicy
IsDefault : True
IPAllowList : {}
IPBlockList : {}
EnableSafeList : False
DirectoryBasedEdgeBlockMode : Default
Identity : Default
Id : Default
IsValid : True
```

Now let's review some examples for configuring this filter:

## Get-Help Set-HostedConnectionFilterPolicy -Examples

```

----- Example 1 -----
Set-HostedConnectionFilterPolicy "Contoso Connection Filter Policy" -IPAllowList
192.168.1.10,192.168.1.23 -IPBlockList 10.10.10.0/25,172.17.17.0/24

----- Example 2 -----
Set-HostedConnectionFilterPolicy "Contoso Connection Filter Policy" -IPAllowList
@{Add="192.168.2.10", "192.169.3.0/24", "192.168.4.1-192.168.4.5";Remove="192.168.1.10"}

```

**Example**

Engineers on the Messaging Team have been tracking the IP connections made by SMTP servers and were able to connect Spam and Malware attacks by these bad IPs. On the reverse side the Messaging Team also has a list of known good SMTP servers from clients, partners and other sources. They are now in the process of preparing EOP as their new protection service before they move mailboxes to their new Exchange Online tenant.

In order to configure the IP Allow and IP Block lists, we need to run the Set-HostedConnectionFilterPolicy cmdlet in order to do so. Here is a sample of what they would need to configure:

```
Set-HostedConnectionFilterPolicy Default -IPAllowList 64.34.23.43,23.100.100.1 -IPBlockList
64.34.23.44,23.100.100.2
```

We can verify the settings have been applied:

```
Get-HostedConnectionFilterPolicy Default
```

```

RunspaceId           : 792cf214-33a1-464d-bf35-3f50
AdminDisplayName     :
IsDefault            : True
IPAllowList          : {23.100.100.1, 64.34.23.43}
IPBlockList          : {64.34.23.44, 23.100.100.2}
EnableSafeList       : False
DirectoryBasedEdgeBlockMode : Default
Identity             : Default

```

After applying these settings, one of the engineers realized they had their lists mixed, how can they reverse the settings? First we'll wipe out all settings to ensure no duplicates or odd values are retained:

```
Set-HostedConnectionFilterPolicy Default -IPAllowList $Null -IPBlockList $Null
```

```
Get-HostedConnectionFilterPolicy Default
```

```

IsDefault            : True
IPAllowList          : {}
IPBlockList          : {}
EnableSafeList       : False
DirectoryBasedEdgeBlockMode : Default
Identity             : Default

```

```
Set-HostedConnectionFilterPolicy Default -IPBlockList 64.34.23.43,23.100.100.1 -IPAllowList
64.34.23.44,23.100.100.2
```

```
Get-HostedConnectionFilterPolicy Default
```

```

AdminDisplayName     :
IsDefault            : True
IPAllowList          : {23.100.100.2, 64.34.23.44}
IPBlockList          : {64.34.23.43, 23.100.100.1}
EnableSafeList       : False
DirectoryBasedEdgeBlockMode : Default
Identity             : Default

```

**Spam Filter**

The Spam Filters in Exchange Online are quite varied and can be a bit confusing. Trying to find PowerShell cmdlets for configuring these settings can be confusing as well. That is because there is no cmdlet that is listed as

a Spam Filter cmdlet. How do we find the cmdlet for it? We use the ‘Show Command Logging’ feature that we covered in Appendix B. By using that and changing one setting in the Spam Filter we see that changing the Spam Filter requires the ‘Set-HostedContentFilterPolicy’ cmdlet. Let’s see what other cmdlets we have that share the same noun phrase:

```
Get-Command *HostedContentFilter*
```

```
Name
-----
Disable-HostedContentFilterRule
Enable-HostedContentFilterRule
Get-HostedContentFilterPolicy
Get-HostedContentFilterRule
New-HostedContentFilterPolicy
New-HostedContentFilterRule
Remove-HostedContentFilterPolicy
Remove-HostedContentFilterRule
Set-HostedContentFilterPolicy
Set-HostedContentFilterRule
```

Looks like we have a lot of configuration cmdlets to choose from to make adjustments to the Spam Filter. Similar to the Malware Policies and Rules we covered earlier in the chapter, the Spam Filter has Rules and Policies as well. So we can configure more Rules and Policies depending on our need. Let’s start out with the default Spam Filter Policy to see what is configured and what we can configure with PowerShell.

```
Get-HostedContentFilterPolicy
```

Name	SpamAction	HighConfidenceSpamAction	IsDefault
Default	MoveToJmf	MoveToJmf	True

So we have one Spam Policy that we can configure called ‘Default’. Let’s review this in more detail:

```
Get-HostedContentFilterPolicy | Fl
```

```
AdminDisplayName           :
AddXHeaderValue           :
ModifySubjectValue       :
RedirectToRecipients      : <>
TestModeBccToRecipients  : <>
FalsePositiveAdditionalRecipients : <>
QuarantineRetentionPeriod : 15
EndUserSpamNotificationFrequency : 3
TestModeAction           : None
IncreaseScoreWithImageLinks : On
IncreaseScoreWithNumericIps : Off
IncreaseScoreWithRedirectToOtherPort : Off
IncreaseScoreWithBizOrInfoUrIs : Off
MarkAsSpamEmptyMessages  : Off
MarkAsSpamJavaScriptInHtml : Off
MarkAsSpamFramesInHtml   : Off
MarkAsSpamObjectTagsInHtml : Off
MarkAsSpamEmbedTagsInHtml : Off
MarkAsSpamFormTagsInHtml : Off
MarkAsSpamWebBugsInHtml  : Off
MarkAsSpamSensitiveWordList : Off
MarkAsSpamSpfRecordHardFail : Off
MarkAsSpamFromAddressAuthFail : Off
MarkAsSpamBulkMail       : On
MarkAsSpamNdrBackscatter : Off
IsDefault                 : True
LanguageBlockList        : <>
RegionBlockList          : <>
HighConfidenceSpamAction : MoveToJmf
SpamAction                : MoveToJmf
EnableEndUserSpamNotifications : False
DownloadLink             : False
EnableRegionBlockList    : False
EnableLanguageBlockList  : False
EndUserSpamNotificationCustomFromAddress :
EndUserSpamNotificationCustomFromName :
EndUserSpamNotificationCustomSubject :
EndUserSpamNotificationLanguage : Default
EndUserSpamNotificationLimit : 0
BulkThreshold            : 7
AllowedSenders           : <>
AllowedSenderDomains     : <>
BlockedSenders           : <>
BlockedSenderDomains     : <>
ZapEnabled               : True
InlineSafetyTipsEnabled  : True
BulkSpamAction           : MoveToJmf
PhishSpamAction          : MoveToJmf
Identity                 : Default
Id                       : Default
```



As we can see from the screenshot above of a brand new tenant, there are a lot of settings that are off and not set. These are off for a reason as turning them on could potentially block legitimate emails if you do not understand what the settings do. However, there is a provided mechanism for testing settings as some of the parameters have possible values of On, Off and Test. This last one is important if there is some uncertainty as to what the setting will do in production. Microsoft provides a good description of these options and their effects on filtering here:

<https://docs.microsoft.com/en-us/microsoft-365/security/office-365-security/configure-your-spam-filter-policies>

There are a couple of these settings in the Spam Filter that show 'MoveToJmf':

```
HighConfidenceSpamAction : MoveToJmf
SpamAction                : MoveToJmf
BulkSpamAction            : MoveToJmf
PhishSpamAction           : MoveToJmf
```

The 'MoveToJmf' setting is just an abbreviated 'Move To Junk Mail Folder'; though it isn't clear why this had to be abbreviated for this property.

### Example

Let's say we want to configure our Spam Filter to test a few parameters. To help analyze the impact of the change, we need to set the 'TestModeAction' parameter to either 'AddXHeader' or 'BccMessage Redirect'. This will allow us to either tag the message with a X-Header we can track or to have a copy of the message sent to a certain address for analysis. For this example we will use an X-Header and change the MarkAsSpamFormTagsInHtml, MarkAsSpamFramesInHtml and MarkAsSpamEmbedTagsInHtml parameters all to 'Test'.

```
Set-HostedContentFilterPolicy Default -MarkAsSpamFormTagsInHtml Test -MarkAsSpamFramesInHtml
Test -MarkAsSpamEmbedTagsInHtml Test -TestModeAction BccMessage -TestModeBccToRecipients
Damian@PracticalPowerShell.Com
```

Now if a message matches any of these settings, a copy will be BCC'd to the email address specified in the 'TestModeBccToRecipients' parameter in our cmdlet.

### Outbound Spam

Exchange Online Protection has an interesting feature that looks for Outbound Spam for your Exchange Online tenant. While the feature is useful, there isn't a lot we can configure. We can notify some admins of the emails that are outbound Spam or we can add some BCC recipients to the message. There isn't much else to this particular feature.

### PowerShell

There aren't a lot of configuration settings and thus not a lot of PowerShell cmdlets available to configure it. Let's take a look at what we can do:

```
Get-Command *HostedOutbound*
```

```
Name
----
Get-HostedOutboundSpamFilterPolicy
Set-HostedOutboundSpamFilterPolicy
```

So, we can Get and we can Set our Outbound Spam settings. First we can check to see what is set by default on the Outbound Spam configuration:

```
Get-HostedOutboundSpamFilterPolicy | Fl
```

```
RunspaceId           : f81f6f03-f259-40bc-9665-1927b37e23
AdminDisplayName     :
NotifyOutboundSpamRecipients : {}
BccSuspiciousOutboundAdditionalRecipients : {}
BccSuspiciousOutboundMail : False
NotifyOutboundSpam   : False
ExchangeVersion     : 0.20 (15.0.0.0)
Name                 : Default
```

Now that we see what our defaults are, let's look to configure this feature with the Set cmdlet that is available to us:

```
Get-Help Set-HostedOutboundSpamFilterPolicy -Examples
```

```
----- Example 1 -----
Set-HostedOutboundSpamFilterPolicy Default -NotifyOutboundSpam $true
-NotifyOutboundSpamRecipients chris@contoso.com
```

### Sample Scenario

Take for example an organization that is worried about Spam being sent outbound from their organization and would like have the administrators of Exchange Online notified if outbound Spam does get sent. For us to do so, we need to configure two parameters:

- NotifyOutboundSpam - This is by default set to 'False', so no notifications are sent.
- NotifyOutboundSpamRecipients - This property is blank because the first one is set to False.

We just need a recipient for notifications and then we can configure the Outbound Spam settings:

```
Set-HostedOutboundSpamFilterPolicy Default -NotifyOutboundSpam $True
-NotifyOutboundSpamRecipients 'Damian@PracticalPowerShell.Com'
```

Use a recipient that is little used as it could lead to potential confusion when email notifications for emails that are sent by this recipient. We can run the 'Get-HostedOutboundSpamFilterPolicy | Fl' one-liner to verify our configuration worked:

```
RunspaceId           : f81f6f03-f259-40bc-9665-1927b37e23
AdminDisplayName     :
NotifyOutboundSpamRecipients : <Damian@PracticalPowerShell.Com>
BccSuspiciousOutboundAdditionalRecipients : {}
BccSuspiciousOutboundMail : False
NotifyOutboundSpam   : True
ExchangeVersion     : 0.20 (15.0.0.0)
Name                 : Default
```

### Using Transport Rules for Spam Protection

In addition to using EOP and other features in Exchange Online, a more rudimentary method still exists for blocking unwanted emails - Transport Rules. While this may seem like an inelegant solution, using Transport Rules to control SPAM is used by many corporations today. These Transport Rules could be simple or complex, depending on the desired results. As a starting point we can use an article written by Microsoft on the subject:

<https://docs.microsoft.com/en-us/microsoft-365/security/office-365-security/use-transport-rules-to-configure-bulk-email-filtering>

The above article provides us with some examples of what can be done, however these are done in the EAC and not in PowerShell. With Exchange Online we can use Transport Rules and construct some rules to control SPAM using techniques like RegEx.

## Anti-Malware

Another useful feature of MDO is Malware protection which consists of protecting mailboxes from malicious attachments as well as blocking well known filetypes that could include executable content. Let's review what we have cmdlet-wise for Malware Policies / Rules:

First off, we'll start with the Malware Filter configuration:

```
Get-Command *MalwareFilter*
```

```
Name
----
Disable-MalwareFilterRule
Enable-MalwareFilterRule
Get-MalwareFilterPolicy
Get-MalwareFilterRule
New-MalwareFilterPolicy
New-MalwareFilterRule
Remove-MalwareFilterPolicy
Remove-MalwareFilterRule
Set-MalwareFilterPolicy
Set-MalwareFilterRule
```

In Exchange Online, the Malware configuration consists of two parts: Malware Filter Policy and Malware Filter Rules. The Rules roll up and can be associated with a Policy in order to be applied. There are no requirements to use the Malware Filter Rules as we can see that none are provided in a brand new tenant.

## Malware Filter Policies

In a brand new tenant, there are no defined Malware Rules in EOP. If we were to run 'Get-MalwareFilterRule', no results would be returned. However, there is a default policy, the same one you see in the Exchange Admin Center, and the 'Get-MalwareFilterPolicy | Fl' cmdlet will display that configuration:

```
CustomAlertText           :
AdminDisplayName          :
CustomInternalSubject     :
CustomInternalBody        :
CustomExternalSubject     :
CustomExternalBody        :
CustomFromName            :
CustomFromAddress         :
InternalSenderAdminAddress :
ExternalSenderAdminAddress :
BypassInboundMessages     : False
BypassOutboundMessages    : False
Action                    : DeleteMessage
IsDefault                  : True
CustomNotifications       : False
EnableInternalSenderNotifications : False
EnableExternalSenderNotifications : False
EnableInternalSenderAdminNotifications : False
EnableExternalSenderAdminNotifications : False
EnableFileFilter          : False
FileTypes                  : {ace, ani, app, docm...}
ZapEnabled                 : True
ExchangeVersion           : 0.20 (15.0.0.0)
Name                       : Default
```

We can configure this Malware Filter Policy using PowerShell, so let's review some examples of this configuration:

#### Get-Help Set-MalwareFilterPolicy -Examples

```
----- Example 1 -----
Set-MalwareFilterPolicy -Identity "Contoso Malware Filter Policy" -Action DeleteMessage
-EnableInternalSenderAdminNotifications $true -InternalSenderAdminAddress admin@contoso.com

----- Example 2 -----
$FileTypesAdd = Get-MalwareFilterPolicy -Identity Default | select -Expand FileTypes
$FileTypesAdd += "com","bat"
Set-MalwareFilterPolicy -Identity Default -EnableFileFilter $true -FileTypes $FileTypesAdd
```

### Example

Management has decided they would like to change the way malware messages are processed by EOP. First they would like to have Administrators notified when a malware messages is detected. Next, if the sender is internal, they would like to notify that user if a malware message was sent from their account. Lastly they would like to notify an external sender if their message contained malware as well. How can we do this? First, we would need to review the full Get-Help for the Set-MalwareFilterPolicy cmdlet.

#### Parameters we can use:

First step is to allow custom notifications, set a From address and a display name for that address:

```
-CustomNotifications $True
-CustomFromAddress ITAdmins@PracticalPowerShell.Com
-CustomFromName 'Big Box IT Admins'
```

Next, we have the internal message notifications for any messages sent by your Exchange Online mailboxes. We will enable notifications to Administrators, the sender (internal), which Administrator's email address to use and a custom subject and body for the notification.

```
-EnableInternalSenderAdminNotifications $True
-EnableInternalSenderNotifications $True
-InternalSenderAdminAddress ITAdmins@PracticalPowerShell.Com
-CustomInternalBody 'Dear user, you have recently sent out a message that contains malware. Please
contact the IT Department.'
-CustomInternalSubject 'Malware detected!'
```

Next, we have the external message notifications for any messages sent to your Exchange Online mailboxes. We will enable notifications to administrators, the sender (external), which admin email address to use and a custom subject and body for the notification.

```
-EnableExternalSenderAdminNotifications $True
-EnableExternalSenderNotifications $True
-ExternalSenderAdminAddress ITAdmins@PracticalPowerShell.Com
-CustomExternalBody 'Dear sender, you have recently sent out a message that contains malware.
Please contact our IT Department @ ITAdmins@PracticalPowerShell.Com'
-CustomExternalSubject 'Malware detected!'
```

Combining all of the options makes for one very long one-liner:

```
Set-MalwareFilterPolicy Default -CustomNotifications $True -CustomFromAddress
ITAdmins@PracticalPowerShell.Com -CustomFromName 'Big Box IT Admins'
-EnableInternalSenderAdminNotifications $True -EnableInternalSenderNotifications $True
-InternalSenderAdminAddress ITAdmins@PracticalPowerShell.Com -CustomInternalBody
'Dear user, you have recently sent out a message that contains malware. Please contact the IT
Department.' -CustomInternalSubject 'Malware detected!' -EnableExternalSenderAdminNotifications
$True -EnableExternalSenderNotifications $True -ExternalSenderAdminAddress ITAdmins@
PracticalPowerShell.Com -CustomExternalBody 'Dear sender, you have recently sent out a message
that contains malware. Please contact our IT Department @ ITAdmins@PracticalPowerShell.Com'
-CustomExternalSubject 'Malware detected!'
```

These changes should then be verified:

```
AdminDisplayName :
CustomInternalSubject : Malware detected!
CustomInternalBody : Dear user, you have recently sent out a message that
contains malware. Please contact the IT Department.
CustomExternalSubject : Malware detected!
CustomExternalBody : Dear sender, you have recently sent out a message that
contains malware. Please contact our IT Department @
ITAdmins@PracticalPowerShell.Com
CustomFromName : Big Box IT Admins
CustomFromAddress : ITAdmins@PracticalPowerShell.Com
InternalSenderAdminAddress : ITAdmins@PracticalPowerShell.Com
ExternalSenderAdminAddress : ITAdmins@PracticalPowerShell.Com
BypassInboundMessages : False
BypassOutboundMessages : False
Action : DeleteMessage
IsDefault : True
CustomNotifications : True
EnableInternalSenderNotifications : True
EnableExternalSenderNotifications : True
EnableInternalSenderAdminNotifications : True
EnableExternalSenderAdminNotifications : True
EnableFileFilter : False
FileTypes : {ace, ani, app, docm...}
```

## Malware Filter Rules

To enhance the Malware Policies there are Malware Rules that can add to the Policies. Rules can be used like Access Control Lists (ACLs) and determine who is affected by a particular Malware Policy by blocking or allowing particular accounts or groups.

Get-Help New-MalwareFilterRule -Examples

```
----- Example 1 -----
New-MalwareFilterRule -Name "Contoso Recipients" -MalwareFilterPolicy "Contoso Malware Filter
Policy" -RecipientDomainIs contoso.com
```

### Example

In the previous example we configured notifications for emails that were detected as malware. The notifications were configured for any internal or external message and there were no restrictions placed on which mailboxes this would apply to and thus all mailboxes are affected. What if we wanted to restrict this policy to the IT department only, possibly for testing before putting the Malware Policy into production. If we review the full Get-Help for the New-MalwareFileRule we see two options that we can use:

```
-SentToMemberOf
-MalwareFilterPolicy
```

This enabled us to create the rule, assign it to the policy and then apply it only to the IT Admins group (with the provided SMTP address):

```
New-MalwareFilterRule -Name "IT Admin Testing Rule" -SentToMemberOf ITAdmins@
PracticalPowerShell.Com -MalwareFilterPolicy Default
```

**NOTE:** You cannot apply a Malware Filter Rule to the Default Malware Filter Policy.

## Safe Attachments

Safe Attachments is part of Microsoft's Advance Threat Protection suite of products and will protect your users from malicious email attachments. It also is not enabled by default. Make sure you have the correct licensing for this product:

Licensing - <https://docs.microsoft.com/en-us/microsoft-365/security/office-365-security/atp-safe-attachments>

### Required Rights:

- Office 365 Global Admin
- Security Administrator - Security and Compliance Center
- Exchange Online Management

### Overall cmdlets

```
Disable-SafeAttachmentRule
Enable-SafeAttachmentRule
Get-SafeAttachmentPolicy
Get-SafeAttachmentRule
New-SafeAttachmentPolicy
New-SafeAttachmentRule
Remove-SafeAttachmentPolicy
Remove-SafeAttachmentRule
Set-SafeAttachmentPolicy
Set-SafeAttachmentRule
```

## How to Configure

In order to configure the Safe Attachment option in Exchange Online, we need to work with a pair of ideas - Rules and Policies. We need to configure a Safe Attachment Policy and then reference this policy in a new Safe Attachment Rule.

### Examples

```
New-SafeAttachmentPolicy
```

```
----- Example 1 -----
New-SafeAttachmentPolicy -Name "Marketing Block Attachments" -Enable $true -Redirect $true
-RedirectAddress admin@contoso.com
```

## New-SafeAttachmentRule

```
----- Example 1 -----
New-SafeAttachmentRule -Name "Research Department Attachment Rule" -SafeAttachmentPolicy
"Research Block Attachments" -SentToMemberOf "Research Department" -ExceptIfSentToMemberOf
"Research Department Managers"
```

We first need to construct a Safe Attachment Policy with which we'll choose some options. Then with the new Policy we will create a Rule that will apply the Policy in a way that handles attachments. Before we create a Rule, let's see what options we have and what makes sense to apply and why:

**Action:** This is a crucial part of the Policy. Here we can decide to allow the document through, block it all together or replace it with another file that might contain some sort of information or warning text for the recipient.

**ActionOnError:** If the attachment scan fails, this determines if the 'Action' above is applied. \$True means the action will apply on scan failure and \$False (default) the 'Action' is not applied upon failure.

**Enable:** \$True - the Policy is enforced / \$False - the Policy is not enforced.

**Redirect:** \$True - detected malware is redirected to another address / \$False (default) - emails are not redirected.

**RedirectAddress:** The address where malware is redirected if \$Redirect is \$True.

For a quick sample Policy, we want to Replace the file if Malware is detected, Enable the Rule and not apply any other changes. We need to make sure we also provide a name.

```
New-SafeAttachmentPolicy -Name "Replace Malware With Warning" -Action Replace -Enable $True
```

One warning message will occur with this Policy. The warning states that if you have an action applied, but do not have a redirect, there is a possibility for data loss:

```
WARNING: The 'Enable redirect' option should be selected when 'Apply the above selection if malware scanning for attachments times out or error occurs' is selected. Not enabling redirect could result in loss of messages.
```

In order to fix this, we would need to change the Policy and add both the Redirect and the RedirectAddress:

```
Set-SafeAttachmentPolicy "Replace Malware With Warning" -Redirect $True -RedirectAddress
damian@practicalpowershell.com
```

Now we can verify our settings:

```
Get-SafeAttachmentPolicy "Replace Malware With Warning" | Fl
```

```
RedirectAddress      : damian@practicalpowershell.com
Redirect             : True
Action               : Replace
ScanTimeout          : 30
ConfidenceLevelThreshold : 80
OperationMode        : Delay
Enable               : True
ActionOnError         : True
IsDefault             : False
AdminDisplayName     :
EnableOrganizationBranding : False
ExchangeVersion      : 0.20 (15.0.0.0)
Name                 : Replace Malware With Warning
```

Don't forget that these Policies can be updated using the Set-SafeAttachmentPolicy cmdlet. With this cmdlet we can change pretty much the same properties we created the Policy with - Action, ActionOnError, Enable, Redirect and RedirectAddress. So if for some reason a change needs to be made it can be done post Policy creation. An example of this is if a new administrator is hired and is put in charge of ATP, we can redirect the messages to this person:

```
Set-SafeAttachmentPolicy "Replace Malware With Warning" -RedirectAddress sally@domain.com
```

If for some reason the Policy needs to be decommissioned/retired, it is recommended to disable the Policy first:

```
Set-SafeAttachmentPolicy "Replace Malware With Warning" -Enable $False
```

Then once this configuration change has been tested for some period of time, then the Policy should be removed:

```
Remove-SafeAttachmentPolicy "Replace Malware With Warning"
```

Now that we have a Safe Attachment Policy, we need to create the corresponding Safe Attachment Rule. This new rule will reference the previously created Safe Attachment Policy as well. Available parameters:

**SafeAttachmentPolicy:** Reference the Safe Attachment Policy created before the rule

**Comments:** Any notations that need to be made to inform admins of the Rule's purpose

**Enabled:** \$True - the Rule is enforced / \$False - the Rule is not enforced

**ExceptIfRecipientDomainIs:** Exception made for a specified domain

**ExceptIfSentTo:** Exception made for a specified recipient

**ExceptIfSentToMemberOf:** Exception made for a specified group

**Priority:** Decide the order in which multiple Safe Attachment Rules are applied

**RecipientDomainIs:** Exception made for a specified domain

**SentTo:** Rule applies to specified recipients

**SentToMemberOf:** Rule applies to specified members of the specified group

We have a policy already created called 'Replace Malware With Warning', so we can use this for our example. In this scenario we want to create a specific rule that applies to the corporate office. People in this office are in a group called 'Corporate'.

```
New-SafeAttachmentRule -Name "Corporate Attachment Rule" -SafeAttachmentPolicy "Replace Malware With Warning" -SentToMemberOf "Corporate"
```

For this next scenario we have a different Safe Attachment Policy called 'Company Wide Safe Attachments'.

```
New-SafeAttachmentPolicy -Name "Company Wide Safe Attachments" -Action Block -Enable $True
```

With this policy, we need to create a rule that will apply it to all mailboxes, except those from the previous Safe Attachment Rule which was applied to those in the 'Corporate' group:

```
New-SafeAttachmentRule -Name "Company Wide Attachment Rule" -SafeAttachmentPolicy "Company Wide Safe Attachments" -ExceptIfSentToMemberOf "Corporate" -RecipientDomainIs PracticalPowerShell.com
```

Now we have two valid Safe Attachment Polices which are used by two Safe Attachment Rules:

```
PS C:\> Get-SafeAttachmentPolicy
```

Name	Action	Enable	IsDefault
Replace Malware With Warning	Replace	True	False
Company Wide Safe Attachments	Block	True	False



```
PS C:\> Get-SafeAttachmentRule
```

Name	State	Priority	SafeAttachmentPolicy	Comments
Corporate Attachment Rule	Enabled	0	Replace Malware With Warning	
Company Wide Attachment Rule	Enabled	1	Company Wide Attachment Rule	

Once the rules are in flight, we can still modify and/or remove them just like we can with the Safe Attachment Policies. We can change the exceptions or change the targets (Send To) for the Rule as well as disable it.

Set-SafeAttachmentRule "Company Wide Attachment Rule" -ExceptIfSentToMemberOf 'Executives'

We can also remove all Safe Attachment Rules and Policies:

```
PS C:\> Get-SafeAttachmentPolicy | Remove-SafeAttachmentPolicy
```

Confirm  
Are you sure you want to perform this action?  
Remove safe attachment policy "Replace Malware With Warning".  
[Y] Yes [A] Yes to All [N] No [L] No to All [?] Help (default is "Y"): a

```
PS C:\> Get-SafeAttachmentRule | Remove-SafeAttachmentRule
```

Confirm  
Are you sure you want to perform this action?  
Removing safe attachment rule "Corporate Attachment Rule".  
[Y] Yes [A] Yes to All [N] No [L] No to All [?] Help (default is "Y"): a

## Safe Links

Office 365 Safe Links is designed to protect those in Exchange Online from malicious links that may be present in emails or Office documents. These links will be blocked if there is an appropriate configuration of the Safe Links feature.

Licensing - <https://docs.microsoft.com/en-us/microsoft-365/security/office-365-security/atp-safe-links>

## Required Rights

- Office 365 Global Admin
- Security Administrator - Security and Compliance Center
- Exchange Online Management

## How to Configure

In order to configure the Safe Link option in Exchange Online, we need to work with a pair of ideas - Rules and Policies. We need to create a Safe Link Policy and then reference this Policy in a new Safe Link Rule.

## Examples

New-SafeLinksRule

```
----- Example 1 -----  
  
New-SafeAttachmentRule -Name "Research Department URL Rule" -SafeAttachmentPolicy "Research Block URL" -SentToMemberOf "Research Department" -ExceptIfSentToMemberOf "Research Department Managers"
```

## New-SafeLinksPolicy

```
----- Example 1 -----
New-SafeLinksPolicy -Name "Marketing Block URL" -IsEnabled $true -TrackClicks $true
```

## Overall Cmdlets

Disable-SafeLinksRule  
 Enable-SafeLinksRule  
 Get-SafeLinksPolicy  
 Get-SafeLinksRule  
 New-SafeLinksPolicy  
 New-SafeLinksRule  
 Remove-SafeLinksPolicy  
 Remove-SafeLinksRule  
 Set-SafeLinksPolicy  
 Set-SafeLinksRule

We first need to construct a Safe Links Policy with which we'll choose some options. Then with the new Policy we will create a rule that will apply the Policy in a way that handles links. Before we create a rule, let's see what options we have and what makes sense to apply and why: (Microsoft Internal parameters have been removed)

**DeliverMessageAfterScan:** <unknown - no help info>.

**DoNotAllowClickThrough:** \$True - blocks a user from clicking to the original URL / \$False (default) - Allows click through to the original URL.

**DoNotRewriteUrls:** Specify one or more URLs that skip Safe Link scanning. (Use this in place of ExcludedUrls)

**DoNotTrackUserClicks:** \$True (default) - do not track user clicks / \$False - track user clicks.

**EnableForInternalSenders:** \$True - Will scan internal sent emails as well, \$False (default) only scans external senders.

**ExcludedUrls:** Specify one or more URLs that can be skipped from scanning. (**Parameters id deprecated**)

**IsEnabled:** (Different from Get-Help which shows 'Enabled') - \$True - enables the policy / \$False - disabled policy.

**ScanUrls:** \$True - scans links in emails / \$False (Default) - does not scan links in emails.

Let's dive in by creating a Policy that scans links in emails (I mean, this is for Exchange Online PowerShell ...), we want to enable scanning for internal and external senders, exclude the corporate Intranet (www.bigboxhomepage.com) and blocks clicking through to original links.

```
New-SafeLinksPolicy -Name 'Corporate Safe Links Policy' -DoNotTrackUserClicks $False
-EnableForInternalSenders $True -IsEnabled $True -DoNotRewriteUrls 'www.bigboxhomepage.com'
-ScanUrls $True
```

If we were to use 'ExcludedUrls' instead of the now recommended 'DoNotRewriteUrls', we would receive this error message:

```
The WhitelistedUrls and ExcludedUrls params are deprecated. Use the DoNotRewriteUrls param
instead. If you have any scripts that use the WhitelistedUrls and ExcludedUrls params, update them
to use the DoNotRewriteUrls param.
```

Don't forget that these Policies can be updated using the Set-SafeLinksPolicy cmdlet. With this cmdlet we can change pretty much the same properties we created the Policy with - DoNotTrackUserClicks, EnableForInternalSenders, DoNotRewriteUrls, ScanUrls and more. So if for some reason a change needs to be made it can be done post Policy creation. For example, if we want to take the Policy above and stop scanning links in emails we can run this one-liner:

```
Set-SafeLinksPolicy "Corporate Safe Links Policy" -ScanUrls $False
```

If for some reason the Policy needs to be decommissioned/retired, it is recommended to disable the Policy first:

```
Set-SafeLinksPolicy "Corporate Safe Links Policy" -Enable $False
```

Then once this configuration change has been tested for some period of time, then the Policy should be removed:

```
Remove-SafeLinksPolicy "Corporate Safe Links Policy"
```

Now that we have a Safe Links Policy, we need to create the corresponding Safe Links Rule. This new rule will reference the previously created Safe Links Policy as well. Available parameters:

**Comments:** Any notations that need to be made to inform admins of the Rule's purpose

**Enabled:** \$True - the Rule is enforced / \$False - the Rule is not enforced

**ExceptIfRecipientDomainIs:** Exception made for a specified domain

**ExceptIfSentTo:** Exception made for a specified recipient

**ExceptIfSentToMemberOf:** Exception made for a specified group

**Priority:** Decide the order in which multiple Safe Attachment Rules are applied

**RecipientDomainIs:** Exception made for a specified domain

**SafeLinksPolicy:** Reference the Safe Links Policy created before the Rule

**SentTo:** Recipient of the message being scanned

**SentToMemberOf:** Group recipient of the message being scanned

We have a policy already created called 'Corporate Safe Links Policy', so we can use this for our example. In this scenario we want to create a specific rule that applies to the corporate office. People in this office are in a group called 'Corporate'.

```
New-SafeLinksRule -Name "Corporate Office Links Rule" -SafeLinksPolicy "Corporate Safe Links Policy"
-SentToMemberOf "Corporate" -Enable $True
```

Let's say that as part of a cleanup process for a tenant, the IT Manager decided that all Safe Links Rule need to have comments on creation date and which IT policy it was created for, we can do so post creation:

```
Set-SafeLinksRule "Corporate Office Links Rule" -Comment "Created 1/1/2020 - Per IT Policy #54321"
```

Now let's say we need to create a second Policy and apply it to other groups in the organization:

```
Set-SafeLinksPolicy "Company Wide Safe Links Policy" -Enable $False -EnableForInternalSenders $True
-ScanUrls $True -DoNotTrackUserClicks $False
```

What we find is that we can only use the -EnableForInternalSenders once between all Safe Link Policies: (Error message is deceptive)

```
Set-SafeLinksPolicy : Cannot bind parameter because parameter 'EnableForInternalSenders' is
specified more than once. To provide multiple values to parameters that can accept multiple
values, use the array syntax. For example, "-parameter value1,value2,value3".
At line:1 char:65
+ ... te Safe Links Policy" -Enable $True -EnableForInternalSenders $True - ...
+ ~~~~~
```

This means we need to reconstruct the rule like so, removing the parameter we received an error for:

```
New-SafeLinksPolicy "Company Wide Safe Links Policy" -Enable $False -ScanUrls $True
-DoNotTrackUserClicks $False
```

After creating this Policy, we can create a new Safe Links Rule to go with it:

```
New-SafeLinksRule -Name "Company Wide Safe Links Rule" -SafeLinksPolicy "Company Wide Safe
Links Policy" -ExceptIfSentToMemberOf "Corporate" -RecipientDomainIs: "practicalpowershell.com"
-Enable $True
```

Now we have two valid Safe Attachment Polices which are used by two Safe Attachment Rules:

```
PS C:\> Get-SafeLinksPolicy |ft -Auto
```

RunspaceId	IsEnabled	TrackClicks	DoNotTrackUserClicks	AllowClickThrough	DoNotAllowClick
efb1852e-6c34-4b57-a1e4-21b5b12aaf35	True	True	False	True	True
efb1852e-6c34-4b57-a1e4-21b5b12aaf35	False	True	False	True	True

```
PS C:\> Get-SafeLinksRule | ft -Auto
```

RunspaceId	SafeLinksPolicy	State	Priority	Comments	Description
efb1852e-6c34-4b57-a1e4-21b5b12aaf35	Corporate Safe Links Policy	Enabled	0		If the message:...
efb1852e-6c34-4b57-a1e4-21b5b12aaf35	Company Wide Safe Links Policy	Enabled	1		If the message:...

Once the rules are in flight, we can still modify and/or remove them just like we can with the Safe Link Policies. We can change the exceptions or change the targets (Sent To) for the Rule as well as disable it.

```
Set-SafeLinksRule "Corporate Office Links Rule" -SentToMemberOf 'Executives'
```

We can also remove all Safe Attachment Rules and Policies:

```
PS C:\> Get-SafeLinksPolicy | Remove-SafeLinksPolicy
```

```
Confirm
Are you sure you want to perform this action?
Remove safe links policy "Corporate Safe Links Policy".
[Y] Yes [A] Yes to All [N] No [L] No to All [?] Help (default is "Y"): a
PS C:\> Get-SafeLinksRule | Remove-SafeLinksRule
```

```
Confirm
Are you sure you want to perform this action?
Removing safe links rule "Corporate Office Links Rule".
[Y] Yes [A] Yes to All [N] No [L] No to All [?] Help (default is "Y"): a
```

Now that we are done with Safe Links and Safe Attachments we'll move on to the Anti-Phishing feature that is also part of the ATP feature set.

## MDO Best Practices

Microsoft has also released a set of recommendations and best practices for configuring options like MDO. These recommendations should be used as general guidelines and not hard requirements of what you absolutely should have configured. What we find is that all organizations of all sizes have such varied requirements for how they operate, how they process email and what end results they would like to see that almost everyone will find something that will not fit with the organization.

As these recommendations are subject to change over time, use the link below to see the current recommendations by Microsoft:

<https://docs.microsoft.com/en-us/microsoft-365/security/office-365-security/recommended-settings-for-eop-and-office365-atp>

## Tenant Allow / Block List Items

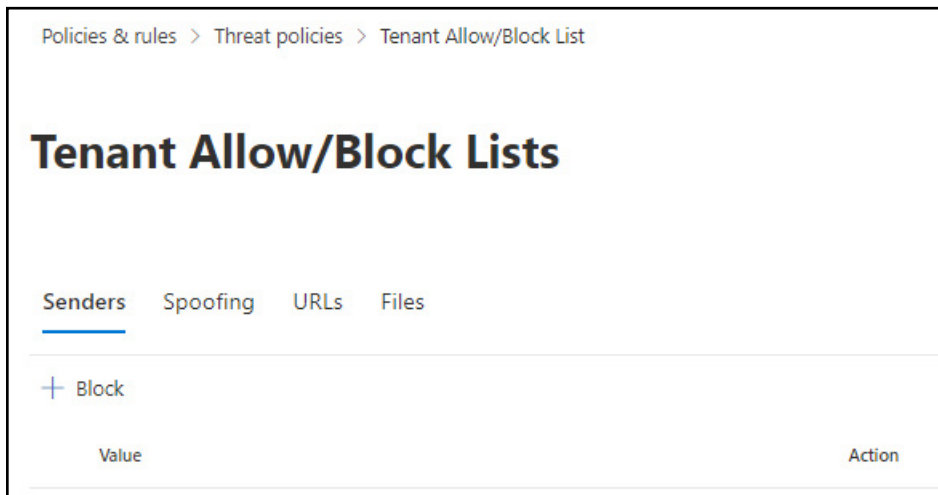
Microsoft recently released this as a Public Preview feature for Microsoft Defender for O365, which is now in production. What the Tenant Allow/Block allows an organization to do is override a particular EOP filtering verdict when it comes to URLs and files that could be blocked or allowed by EOP. In the Security and Compliance Center, this can be found here:

<https://security.microsoft.com/tenantAllowBlockList>

Which can be found here in the GUI:



Clicking on this brings us to the new GUI interface for the Tenant Allow/Block Lists feature:



### Permissions

- To add and remove values from the Tenant Allow/Block List, you need to be a member of the Organization Management or Security Administrator role groups.
- For read-only access to the Tenant Allow/Block List, you need to be a member of the Global Reader or Security Reader role groups.

### Licensing Required

- EOP - Exchange Online Plan 1 and Plan 2 include EOP by default.

- Defender for Office 365 Plan 1 or 2

### Conditions

- Files can be specified by using the SHA256 Hash of the file.
- URL Rules
  - Punycode can be used, but not Unicode
  - Hostnames should be x.x at the very least
  - Subpaths not included by default [www.contoso.com/Signup]
- Wildcards can be used for paths as well as DNS names
- Max of 500 URLs and 500 File hashes in a tenant.
- The maximum number of characters for each entry is:
  - File hashes = 64 / URL = 250
- An entry should be active within 30 minutes.
- Entries expire in 30 days by default, but can be set to expire differently or not at all.

### PowerShell

Now that we've gone over the basics and the GUI interface in the Security and Compliance Center, we can now look at the PowerShell cmdlets that will allow us to manage these settings:

```
Get-Command *AllowBlock*
```

This produces a short list of cmdlets:

Get-TenantAllowBlockListItems	Get-TenantAllowBlockListSpoofItems
New-TenantAllowBlockListItems	New-TenantAllowBlockListSpoofItems
Remove-TenantAllowBlockListItems	Remove-TenantAllowBlockListSpoofItems
Set-TenantAllowBlockListItems	Set-TenantAllowBlockListSpoofItems

What you will find with these cmdlets is that there really is very little help at the moment and even fewer examples to help us understand what to do:

```
PS C:\> Get-Help New-TenantAllowBlockListItems -Examples

NAME
    New-TenantAllowBlockListItems

ALIASES
    None

REMARKS
    None
```

### New-TenantAllowBlockListItems

With this cmdlet we can add these new block lists. No examples exist for this cmdlet in Exchange Online, however, we can find them on the Microsoft Docs page for the cmdlet - <https://docs.microsoft.com/en-us/powershell/module/exchange/new-tenantallowblocklistitems>.

### Example 1

```
New-TenantAllowBlockListItems -ListType Url -Action Block -Entries "contoso.com"
```

```
PS C:\> New-TenantAllowBlockListItems -ListType Url -Action Block -Entries ~contoso.com~
Creating a new Remote PowerShell session using Modern Authentication for implicit remoting of "New-TenantAllowBlockListItems" command ...
A parameter cannot be found that matches parameter name 'Action'.
+ CategoryInfo          : InvalidArgument: (:) [New-TenantAllowBlockListItems], ParameterBindingException
```

## Example 2

```
New-TenantAllowBlockListItems -ListType FileHash -Action Allow -Entries
"768a813668695ef2483b2bde7cf5d1b2db0423a0d3e63e498f3ab6f2eb13ea3",
"2c0a35409ff0873cfa28b70b8224e9aca2362241c1foed6f622fef8d4722fd9a" -NoExpiration
```

Also, in the list of available parameters Allow is missing and Block appears to be the only 'action' available:

```
PS C:\> New-TenantAllowBlockListItems -ExpirationDate
ExpirationDate NoExpiration ErrorAction
Block          Entries PipelineVariable
Notes          ListType Verbose
OutputJson     AsJob OutVariable
```

**NOTE:** The examples, with the parameters shown above as well as the help above do not match what is available as of the writing of this edition of the book. It appears that we are also missing parameters that are specified in the Get-Help for the cmdlet. For example, there is no 'Action' parameter available for the cmdlet:

From the two provided examples, we see that URLs are relatively straight forward process, but we do need to follow some basic rules. These rules are spelled out here from Microsoft [[here](#)]. Now, for File Hashes we need to perform an extra step because the value specified is a SHA256 hash value for the file. For example, if we have a file called BlockThis.Txt which is stored in the c:\Source folder and we needed the hash, we can retrieve the appropriate value. This can be obtained by using a CMD prompt or PowerShell window to run this one-liner:

```
certutil.exe -hashfile "C:\Source\BlockThis.txt" SHA256
```

```
C:\>certutil.exe -hashfile "C:\Source\BlockThis.txt" SHA256
SHA256 hash of C:\Source\BlockThis.txt:
5c1fa593f95d544b661096b92a60a6f1fde4f6c17c90b77844fb68a855c30437
CertUtil: -hashfile command completed successfully.
```

The hash value provided can be used in PowerShell now:

```
5c1fa593f95d544b661096b92a60a6f1fde4f6c17c90b77844fb68a855c30437.
```

Using this has value, a Block could be created:

```
New-TenantAllowBlockListItems -ListType FileHash -Block
-Entries '5c1fa593f95d544b661096b92a60a6f1fde4f6c17c90b77844fb68a855c30437' -NoExpiration
```

```
PS C:\> New-TenantAllowBlockListItems -ListType FileHash -Block -Entries '5c1fa593f95d544b661096b92a60a6f1fde4f6c17c90b77844fb68a855c30437' -NoExpiration

RunspaceId      : 90f4a31e-13e4-43bc-a206-a3d00393e9c6
Error           :
Identity        : RgAAAAA4fqs4PIeBQbzo-yGiHPbFBwB-cCadBE1nQJBr5gm8mpxLAABo1StSAAB-cCadBE1nQJBr5gm8mpxLAABo1TEsAAAA
                0
Value           : 5c1fa593f95d544b661096b92a60a6f1fde4f6c17c90b77844fb68a855c30437
Action          : Block
Notes           :
LastModifiedDateTime : 3/21/2021 4:54:17 AM
ExpirationDate   :
ObjectState     : New
```

What available parameters do we have to create this Block or Allow?

**Allow:** Not available yet, but would set the file or URL to be allowed by EOP

**Block:** Block this file or URL

**Entries:** List the file hashes or URLs

**ExpirationDate:** When this Allow or Block expires

**ListType:** URL or FileHash

**NoExpiration:** Allow or Block never expires

**Notes:** Add comments or notes to the Allow or Block

**OutputJSON:** JSON is shown for the addition

(Sample JSON Output):

```
PS C:\> New-TenantAllowBlockListItems -ListType FileHash -Block -Entries '5c1fa593f95d544b661096b92a60a6f1fde4f6c17c90b77844fb68a855c30437' -NoExpiration -OutputJson

RunspaceId      : d47d5764-d8dd-4508-92d7-30c97ef5df79
Error           :
Identity        :
Value           : [{"Error":null,"Identity":"RgAAAAA4fqs4PIeBQbzo-yGiHPbfBwB-cCadbElnQJBr5gm8mpxLAABolStSAAB-cCadbElnQJBr5gm8mpxLAABolTEuAAAA0","Value":"5c1fa593f95d544b661096b92a60a6f1fde4f6c17c90b77844fb68a855c30437","Action":"Block","Notes":null,"LastModifiedDate":"2021-03-21T05:19:10.9766197Z","ExpirationDate":null,"ObjectState":"New"}]
Action          : 0
Notes           :
```

## Set-TenantAllowBlockListItems

Just like the New-TenantAllowBlockListItems cmdlet, no examples exist in PowerShell and we need to pull them from the Microsoft Docs page for the cmdlet - <https://docs.microsoft.com/en-us/powershell/module/exchange/set-tenantallowblocklistitems>.

### Example 1

```
Set-TenantAllowBlockListItems -ListType Url -Ids "RgAAAAAI8gSyl_NmQqzeh-HXJBywBwCqfQNJY8hBTbdIKFkv6BcUAAAI_QCZAACqfQNJY8hBTbdIKFkv6BcUAAAI_oSRAAAA" -ExpirationDate (Get-Date "5/30/2021 9:30 AM").ToUniversalTime()
```

### Example Usage

We can take all Tenant Block lists that were set to no expiration and put a 365 day expiration on it, in order to force a review of the Allows. We would do this as periodic reviews of these Allows is required by IT Management and the Security Group within IT. As such, we will calculate the end date of 365 days from now and apply it to all Block Rules. First, see which have no expiration date:

```
Get-TenantAllowBlockListItems -ListType FileHash -NoExpiration
```

```
RunspaceId      : 90f4a31e-13e4-43bc-a206-a3d00393e9c6
Error           :
Identity        : RgAAAAA4fqs4PIeBQbzo-yGiHPbfBwB-cCadbElnQJBr5gm8mpxLAABolStSAAB-cCadbElnQJBr5gm8mpxLAABolTEAAAA0
Value           : 3efcf43e546f1f38f7b21a7e4cedcbcad912799854fe8369b1ee8b7398116ec0
Action          : Block
Notes           :
LastModifiedDate : 3/21/2021 4:56:36 AM
ExpirationDate   :
ObjectState     : Unchanged

RunspaceId      : 90f4a31e-13e4-43bc-a206-a3d00393e9c6
Error           :
Identity        : RgAAAAA4fqs4PIeBQbzo-yGiHPbfBwB-cCadbElnQJBr5gm8mpxLAABolStSAAB-cCadbElnQJBr5gm8mpxLAABolTEsAAAA0
Value           : 5c1fa593f95d544b661096b92a60a6f1fde4f6c17c90b77844fb68a855c30437
Action          : Block
Notes           :
LastModifiedDate : 3/21/2021 4:54:17 AM
```



Next, we get the date for 365 days in the future:

```
$Date = Get-Date
$365Days = $Date.AddDays(365)
```

Then we use the Set cmdlet to change the expiration to 365 days in the future:

```
$NoExpirationBlocks = (Get-TenantAllowBlockListItems -ListType FileHash -NoExpiration).Identity
ForEach ($NoExpirationBlock in $NoExpirationBlocks){
    Set-TenantAllowBlockListItems -Ids $NoExpirationBlock -ExpirationDate $365Days -ListType FileHash
}
```

```
PS C:\> ForEach ($NoExpirationBlock in $NoExpirationBlocks){ Set-TenantAllowBlockListItems -Ids $NoExpirationBlock -ExpirationDate $365Days -ListType FileHash }
```

```
RunspaceId      : 90f4a31e-13e4-43bc-a206-a3d00393e9c6
Error           :
Identity        : RgAAAAA4fqs4PIeBQbzo-yGiHPbfBwB-cCadBElnQJBr5gm8mpxLAABolStSAAB-cCadBElnQJBr5gm8mpxLAABolTEtAAAA
Value           : 3efcf43e546f1f38f7b21a7e4cedcbcad912799854fe8369b1ee8b7398116ec0
Action          : Block
Notes           :

ExpirationDate  : 3/21/2022 12:00:00 AM
ObjectState     : Changed

RunspaceId      : 90f4a31e-13e4-43bc-a206-a3d00393e9c6
Error           :
Identity        : RgAAAAA4fqs4PIeBQbzo-yGiHPbfBwB-cCadBElnQJBr5gm8mpxLAABolStSAAB-cCadBElnQJBr5gm8mpxLAABolTEsAAAA
Value           : 5c1fa593f95d544b661096b92a60a6f1fde4f6c17c90b77844fb68a855c30437
Action          : Block
Notes           :
LastModifiedDateTime : 3/21/2021 5:01:04 AM
ExpirationDate  : 3/21/2022 12:00:00 AM
ObjectState     : Changed
```

If we rerun this one-liner below, no results should now show:

```
Get-TenantAllowBlockListItems -ListType FileHash -NoExpiration
```

## Remove-TenantAllowBlockListItems

Just like the New-TenantAllowBlockListItems cmdlet, no examples exist in PowerShell and we need to pull them from the Microsoft Docs page for the cmdlet - <https://docs.microsoft.com/en-us/powershell/module/exchange/remove-tenantallowblocklistitems>.

### Example 1

Using hashes from the previous examples, we can also remove them via PowerShell:

```
Remove-TenantAllowBlockListItems -ListType Url -Ids "RgAAAAA18gSyl_NmQqzeh-HXJBywBwCqfQNJY8hBTbdIKFkv6BcUAAA1_QCZAACqfQNJY8hBTbdIKFkv6BcUAAA1_oSPAAAAoI"
```

Similar to the Set cmdlet, we need a listtype and the Ids for the items to be removed. If we decided that a Block is no longer needed, we can get the Id and then remove that Block:

```
Remove-TenantAllowBlockListItems -ListType FileHash -Ids RgAAAAA4fqs4PIeBQbzo-yGiHPbfBwB-cCadBElnQJBr5gm8mpxLAABolStSAAB-cCadBElnQJBr5gm8mpxLAABolTEtAAAAo
```

We get a result that the Block was removed:

```
PS C:\> Remove-TenantAllowBlockListItems -ListType FileHash -Ids RgAAAAA4fqs4PIeBQbzo-yGiHPbfBwB-cCadBE1nQJBr5gm8mpxLAAo1StSAAB-cCadBE1nQJBr5gm8mpxLAABo1TEtAAAA0_

RunspaceId      : d47d5764-d8dd-4508-92d7-30c97ef5df79
Error           :
Identity        : RgAAAAA4fqs4PIeBQbzo-yGiHPbfBwB-cCadBE1nQJBr5gm8mpxLAAo1StSAAB-cCadBE1nQJBr5gm8mpxLAABo1TEtAAAA0_
Value           :
Action          : 0
Notes          :
LastModifiedDate :
ExpirationDate  :
ObjectState     : Deleted
```

In addition, we can see in the console that there are two new files being blocked. Even though the creation allowed for a single 'rule', two actual blocks are created, as shown here:

Policies & rules > Threat policies > Tenant Allow/Block List

## Tenant Allow/Block Lists

Specify the URLs and files that are always allowed or blocked by your tenant in Office 365. [Learn more.](#)

Senders Spoofing URLs Files

+ Block Edit Delete 1 of 2 selected Group Search Refresh Filter

Applied filters:

Value	Action	Last updated date	Expiration Date
5c1fa593f95d544b661096b92a60a6f1fde4f6c17c...	Block	Mar 21, 2021 12:19 AM	Never expire
3efcf43e546f1f38f7b21a7e4cedcbcad912799854...	Block	Mar 21, 2021 12:21 AM	Never expire

## Client View

If we attempt to send an email with a file that matches this hash, we see this:

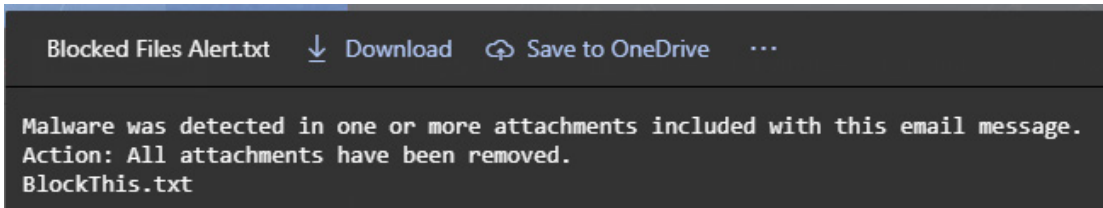
**File Hash test for Tenant Block rules**

**DS** **Damian R. Scoles**  
Sun 3/21/2021 12:22 AM  
To: Damian Scoles

 **Blocked Files Alert.txt**  
556 bytes

Test 1-2-3

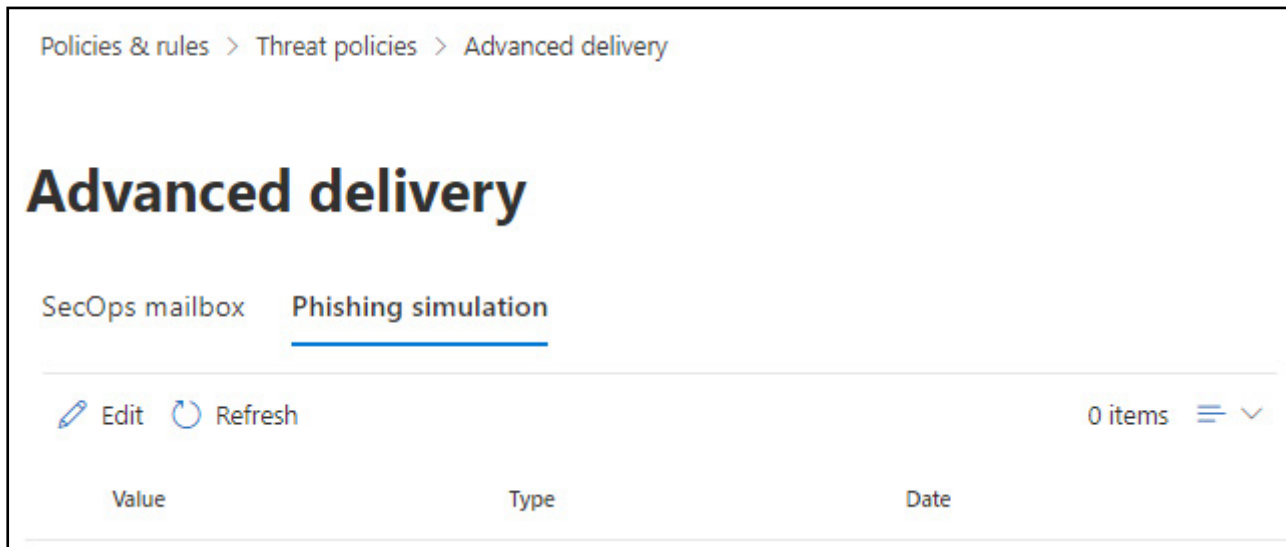
Blocked Files Alert txt file content:



## Advanced Delivery

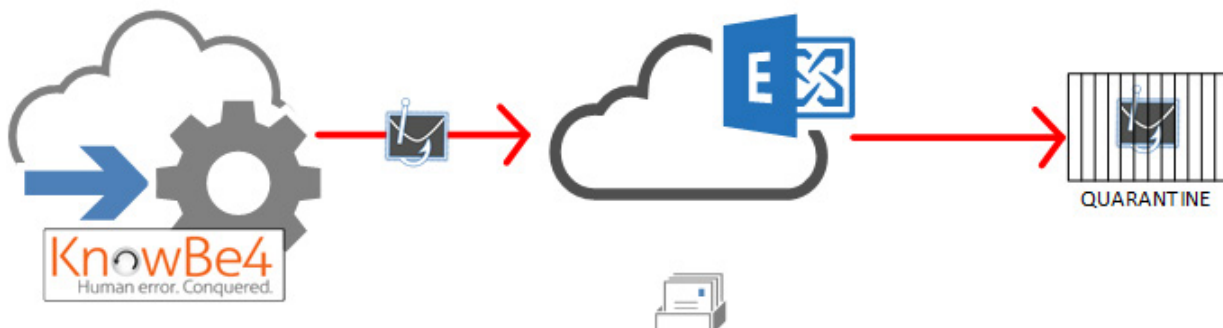
Recently added is a new feature in the Security Center called Advanced Delivery which is focused on two aspects of mail flow control. One of these is Phish Simulation Overrides which deal with emails that are created externally and sent to users for training users how to recognize and respond to Phish emails. The other is creating an override specifically for Security Operations mailboxes that handle alerting and examining emails in their RAW state.

This new feature can be found in the **Security Center > Policies & Rules > Threat Policies > Advanced Delivery**.

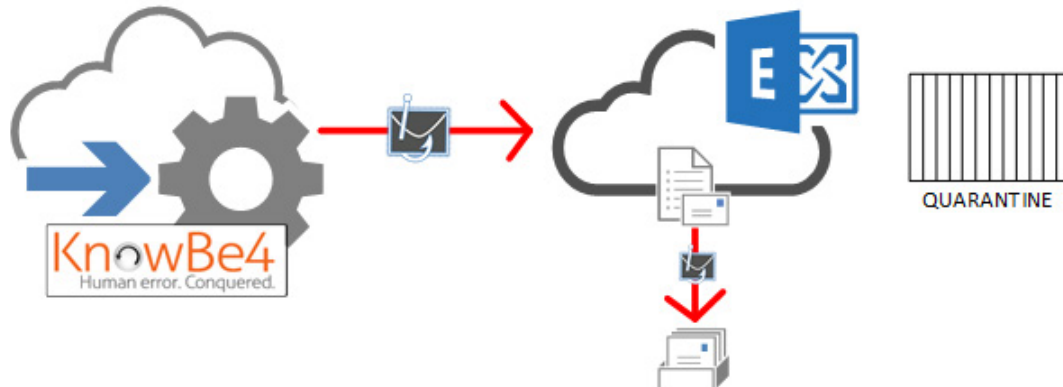


## Phishing Simulation Overrides

With the rise of more persistent Phishing attacks against organizations, the effort and expense of training end users has also gained a foothold. As such organizations are now utilizing resources to simulate these attacks whether with native Microsoft Attack Simulation training or with a third-party service which send simulated emails to users in an organization. The problem is that Exchange Online Protection is designed to block these types of emails as shown below:



Traditionally organizations follow the direction of their vendor and create Transport Rules to allow these emails through, skipping EOP's natural protection and the Phishing email will be delivered to the user's mailbox:



With the Phish Simulation Override feature, EOP is provided with the well known source IPs of the third-party service and allow their Phishing emails to pass. Now, just like any other manually configured services, this configuration should be reviewed often as it could be forgotten and if the vendor goes out of business, the IPs may eventually pass to another company or if the third-party removes IP addresses, the configuration would then also be open to compromise. Consider deploying these feature carefully. Also, if you have any existing Transport Rules, remove those as well prior to making this change to allow for replication to occur in the service.

## PowerShell

Like the rest of this book, what would configuring these features be without a little PowerShell. Microsoft has indeed provided us with the coding tools to do that with PowerShell.

Get-Command \*PhishSim\*

This provides us with a list of cmdlets we can use to manage this feature:

```
Get-PhishSimOverridePolicy
Get-PhishSimOverrideRule
New-PhishSimOverridePolicy
New-PhishSimOverrideRule
Remove-PhishSimOverridePolicy
Remove-PhishSimOverrideRule
Set-PhishSimOverridePolicy
Set-PhishSimOverrideRule
```

## Example

In our sample scenario we have an organization that was using a third-party Phish simulation cloud service prior to migrating to Office 365. Now the organization is having problems with the simulation emails making it to their end user's mailboxes and are getting caught in Exchange Online Protection (EOP). Previously the organization would follow guidance from the vendor and created special Transport Rules to skip EOP processing. However, these did not appear to be working 100% and tweaks are constantly needed. Currently the organization has two domains: BigBox.Com and MediumBox.Com

Microsoft has just recently released a series of cmdlets and the IT department has decided to move forward with these. The current Phish vendor is KnowBe4. With a bit of research we can confirm their sending IP addresses,

which can be used in the creation of these features.

<https://support.knowbe4.com/hc/en-us/articles/203645138-Whitelisting-Data-and-Anti-Spam-Filtering-Information#whitelist>

First, we start with the New Policy and New Rule cmdlets:

```
New-PhishSimOverridePolicy -Name 'KnowBe4PhishOverridePolicy'
```

After creating the Policy, this needs to be applied with a Rule:

```
New-PhishSimOverrideRule -Name 'KnowBe4PhishOverridePolicy' -Policy 'KnowBe4PhishOverridePolicy'
-SenderDomains BigBox.Com,MediumBox.Com -SenderIpRanges 147.160.167.0/26,
23.21.109.197,23.21.109.212
```

Once in place emails sent from KnowBe4 will pass-through and not be scanned by EOP.

### Verifying a Configuration

Get cmdlets are well known for displaying the current configuration of an object or feature. In the case of the PhishSim Override cmdlets, we need to do a bit more legwork. Settings such as IP addresses of the third-party sender as well as the sending domain are easy enough to find as we can run these two cmdlets:

```
Get-PhishSimOverridePolicy
Get-PhishSimOverrideRule
```

What we see is results like so, relevant configurations are highlighted:

```
Get-PhishSimOverridePolicy
```

```
RunspaceId      : df584627-c07c-43bd-893f-9563161c2018
Mode            : Enable
Type            : PhishSim
Workload        : Exchange
Priority         : 0
ObjectVersion   : 196ed3ea-3a31-493d-7a64-08d96196961b
CreatedBy       : Damian Scoles
LastModifiedBy  : Damian Scoles
ReadOnly        : False
ExternalIdentity : 203e881b-9175-4e22-9e98-15e6b248f66c
Comment         :
Enabled         : True
DistributionStatus : Pending
DistributionResults :
LastStatusUpdateTime :
ModificationTimeUtc :
CreationTimeUtc  :
Identity        : PhishSimOverridePolicy
Id              : PhishSimOverridePolicy
IsValid         : True
ExchangeVersion : 0.20 (15.0.0.0)
Name            : PhishSimOverridePolicy
```

## Get-PhishSimOverrideRule

```

RunspaceId      : df584627-c07c-43bd-893f-9563161c2018
SenderIpRanges  : {52.56.150.127, 35.177.22.237}
SenderDomainIs  : {tacklephishing.com}
RuleXml         : <rule name="_Exe:PhishSimOverr:71dccd3a-725c-4641-aa4b-aedbd51207f3" enabled="True" comments=""
mode="Enforce" whenChangedUtc="2021-08-17 18:05:31Z"><version
requiredMinVersion="1.0.67.0"><condition><and><is property="Item.SharedFromIPAddress" type="Microsoft.Office.CompliancePolicy.IPAddressRange"><value>52.56.150.127</value><value>35.177.22.237</value></
is><is property="Item.SharedByDomains"
type="System.Collections.Generic.List`1[Microsoft.Office.CompliancePolicy.Domain]"
Workload="Exchange"><value>tacklephishing.com</value></is></and></condition><action
name="ApplyOverride" /></version></rule>
ManagementRuleId :
ReadOnly          : False
ExternalIdentity  : d2fb7242-0b15-4295-8e7e-ac2e0ce91e98
ImmutableId      : 00000000-0000-0000-0000-000000000000
Priority          : 0
Workload         : Exchange
Policy          : 203e881b-9175-4e22-9e98-15e6b248f66c
Comment         :
Disabled        : False
Mode            : Enforce
ObjectVersion    : 17b392d2-8316-4718-f62b-08d961a99a87
CreatedBy       : Damian Scoles
LastModifiedBy  : Damian Scoles
Guid           : ad029cfe-b657-444a-a244-5085c2c2e5db
Identity        : _Exe:PhishSimOverr:71dccd3a-725c-4641-aa4b-aedbd51207f3
Id             : _Exe:PhishSimOverr:71dccd3a-725c-4641-aa4b-aedbd51207f3
IsValid        : True
ExchangeVersion : 0.20 (15.0.0.0)
Name           : _Exe:PhishSimOverr:71dccd3a-725c-4641-aa4b-aedbd51207f3
  
```

However, when reviewing the configured settings we can see that no URL is displayed. The URL referred to here is the one being used as a simulation URL, specified in the portal here:

### Edit third party phishing simulations


Phishing simulations are attacks orchestrated by your security team and used for training and learning. Simulations can help identify vulnerable users and lessen the impact of malicious attacks on your organization.

Third party phishing simulations require at least 1 entry for **Sending domain** and at least 1 entry for **Sending IP** categories below. Simulations URLs to allow is an optional field. Specify URLs here to not block or detonate on for your phishing simulation.

Sending Domain (1 items) ∨

Sending IP (2 items) ∨

Simulation URLs to allow (1 items) ⓘ ∧

 www.practicalpowershell.com ×

In order to get this URL, we first need to connect to Exchange Online PowerShell:

```
Connect-ExchangeOnline
```

... and then we can run this cmdlet:

```
Get-TenantAllowBlockListItems -ListType URL -ListSubType AdvancedDelivery
```

This then reveals the URL:

```

Error          :
Identity       : RgAAAAA4fqs4PIeB0bzo-yGiHPbfBwB-cCadBElnQ
Value         : www.practicalpowershell.com
Action        : Allow
Notes         :
SubmissionID   : Manual
ListSubType    : AdvancedDelivery
SysManaged    : False
LastModifiedDate : 8/18/2021 3:52:35 PM
ExpirationDate :
ObjectState    : Unchanged

```

What is confusing is that this same cmdlet exists in the Security and Compliance Center PowerShell module, however it is missing the ListSubType parameter. In Exchange Online, this same parameter has three possible values:

**Submission:** Reveals URLs that have been submitted by Admins or end users for analysis.

**AdvancedDelivery:** Reveals URLs that are part of Advanced Delivery configurations.

**Tenant:** Reveals URLs specified in the Tenant Allow/Block Lists.

## Removing Rules and Policies

```
Remove-PhishSimOverrideRule -Identity bb154945-fe02-49bd-809b-d6d89837135f
```

```

PS C:\> Remove-PhishSimOverrideRule -Identity bb154945-fe02-49bd-809b-d6d89837135f

Confirm
Are you sure you want to perform this action?
Removing Compliance Rule 'bb154945-fe02-49bd-809b-d6d89837135f'.
[Y] Yes [A] Yes to All [N] No [L] No to All [?] Help (default is "Y"): y
PS C:\>

```

Then we can remove the Policy that goes with it:

```
Remove-PhishSimOverridePolicy -Identity 8337c875-802a-476d-bdc2-28a0c845ccbc
```

```

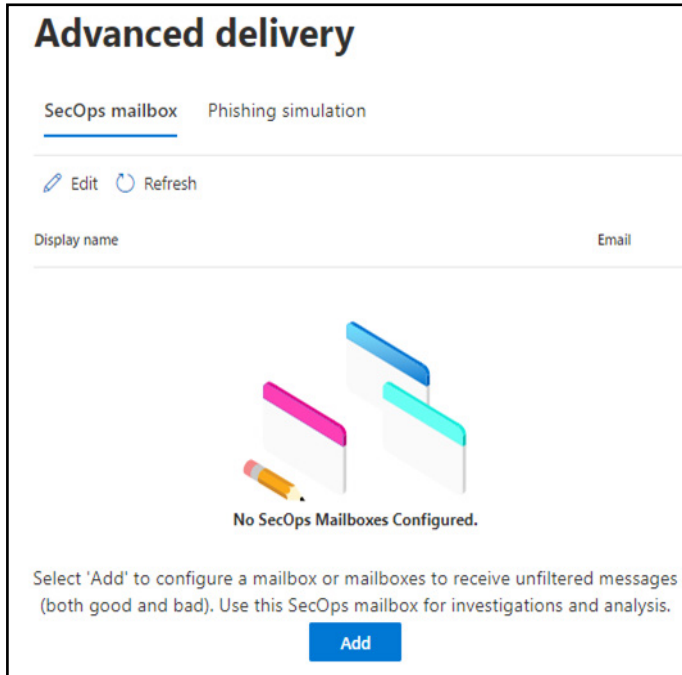
PS C:\> Remove-PhishSimOverridePolicy -Identity 8337c875-802a-476d-bdc2-28a0c845ccbc

Confirm
Are you sure you want to perform this action?
Removing Compliance Policy '8337c875-802a-476d-bdc2-28a0c845ccbc'.
[Y] Yes [A] Yes to All [N] No [L] No to All [?] Help (default is "Y"): y
PS C:\>

```

## SecOps Mailbox Overrides

A SecOps mailbox is a special mailbox that is set up, once configured in PowerShell / GUI, to receive unfiltered emails from EOP or other sources. These emails can then be analyzed by an organizations Security Operations Team, hence the concept name. We can configure this in the GUI:



Or with PowerShell. Like the rest of this book, what would configuring these features be without a little PowerShell. Microsoft has indeed provided us with the coding tools to do that with PowerShell.

Get-Command \*SecOps\*

This provides us with a list of cmdlets we can use to manage this feature:

```
Get-SecOpsOverridePolicy
Get-SecOpsOverrideRule
New-SecOpsOverridePolicy
New-SecOpsOverrideRule
Remove-SecOpsOverridePolicy
Remove-SecOpsOverrideRule
Set-SecOpsOverridePolicy
Set-SecOpsOverrideRule
```

Keep in mind that in order to use this feature, a mailbox needs to be dedicated as a Sec Ops mailbox that can be used for this analysis work. It is highly advised that a user mailbox is not used, but instead a service mailbox should be used. For example, creating a new mailbox called SecurityTeam, designate it as a Shared mailbox and then remove the license.

```
New-Mailbox -Shared -Name 'SecurityTeam' -DisplayName 'Security Team' -Alias SecurityTeam
```

### Example

```
New-SecOpsOverridePolicy -Name SecTeamSecOpsOverridePolicy -SentTo SecurityTeam@powershellgeek.com
```



Which generates this output:

```

Mode                : Enable
SentTo              : {SecurityTeam@powershellgeek.onmicrosoft.com}
Type               : SecOps
Workload           : Exchange
Priority            : 0
ObjectVersion      : 83fdafb8-c8a2-470f-c819-08d954a69dc1
CreatedBy          : Damian Scoles
LastModifiedBy    : Damian Scoles
ReadOnly           : False
ExternalIdentity   :
Comment            :
Enabled            : True
DistributionStatus : Pending
DistributionResults:
LastStatusUpdateTime :
ModificationTimeUtc : 8/1/2021 4:41:23 AM
CreationTimeUtc    : 8/1/2021 4:41:23 AM
Identity           : FF0.extest.microsoft.com/Microsoft Exchange Hosted Organizations/powershellgeek.onmicrosoft.com
Id                 : FF0.extest.microsoft.com/Microsoft Exchange Hosted Organizations/powershellgeek.onmicrosoft.com
IsValid            : True
ExchangeVersion    : 0.20 (15.0.0.0)
Name               : SecOpsOverridePolicy
DistinguishedName  : CN=SecOpsOverridePolicy,CN=Configuration,CN=powershellgeek.onmicrosoft.com,OU=Exchange Hosted Organizations,DC=FF0,DC=extest,DC=microsoft,DC=com
  
```

This mailbox will now be displayed in the portal:

The screenshot shows the 'Advanced delivery' interface in the Microsoft Defender for Office 365 portal. It displays a 'SecOps mailbox' used for a 'Phishing simulation'. The mailbox contains 1 item, 'Security Team', with the email address SecurityTeam@powershellgeek.com. The interface includes options to 'Edit' and 'Refresh' the mailbox, and a dropdown menu for the item list.

Creating a new Rule:

```
New-SecOpsOverrideRule -Name SecOpsOverrideRule -Policy SecOpsOverridePolicy
```

```

RunspaceId        : 8d6f6db5-28f7-483d-a1a5-2a8d0a2ee24b
RuleXml           : <rule name="SecOpsOverrideRule75ffe4ba-aba6-4046-9fee-a667bdfb08b4" enabled="True" comments=""
mode="Enforce" whenChangedUtc="2021-08-01 04:47:32Z"><version
requiredMinVersion="1.0.67.0"><condition><and><is property="Item.SharedWithUserAddresses" type=
"System.Collections.Generic.IEnumerable`1[System.String]"><value>SecurityTeam@powershellgeek.on
microsoft.com</value></is></and></condition><action name="ApplyOverride" /></version></rule>
SentTo            : {SecurityTeam@powershellgeek.onmicrosoft.com}
ReadOnly          : False
ExternalIdentity  :
ImmutableId       : 00000000-0000-0000-0000-000000000000
Priority           : 0
Workload          : Exchange
Policy            : 84cec863-dd8f-4d48-8c69-37711434d8e8
Comment           :
  
```

```

Disabled           : False
Mode               : Enforce
ObjectVersion      : b433d76c-f093-45ec-2e46-08d954a77994
MaximumBlobRuleLength : 0
CreatedBy          : Damian Scoles
LastModifiedBy    : Damian Scoles
Guid               : e1779826-a7cb-453c-93ec-862310f7b293
Identity           : FF0.exetest.microsoft.com/Microsoft Exchange Hosted Organizations/powershellgeek.onmicrosoft.com
                   : /Configuration/SecOpsOverrideRule75ffe4ba-aba6-4046-9fee-a667bdfb08b4
Id                 : FF0.exetest.microsoft.com/Microsoft Exchange Hosted Organizations/powershellgeek.onmicrosoft.com
                   : /Configuration/SecOpsOverrideRule75ffe4ba-aba6-4046-9fee-a667bdfb08b4
IsValid            : True
ExchangeVersion    : 0.20 (15.0.0.0)
Name               : SecOpsOverrideRule75ffe4ba-aba6-4046-9fee-a667bdfb08b4

```

**NOTE:** Specifying a name for either the Rule or the Policy is essentially a non-relevant option. Whatever you enter will be ignore by PowerShell. As we can only have one SecOps Override Policy and one Rule, naming is completely system controlled. The Policy name is SecOpsOverridePolicy and the Rule name is SecOpsOverrideRule followed by a GUID.

## Enhanced Filtering

Exchange Online Protection behaves like we would expect any mail hygiene product to do, if an email is received from an external source, the source is examined to determine if it is trustworthy. When the two mail servers, sending and receiving, communicate, the servers perform SMTP authentication where the receiving server validates the sending server information. For when EOP initiates connections with a sending SMTP server, a series of checks are made on the sender and information in the email being sent. The main component is the concept of Email Authentication in which Microsoft utilizes multiple DNS records to help validate information on a sending SMTP server. Email Authentication takes place when a senders domain is checked for SPF, DMARC and DKIM DNS records. While these records are a common way to provide domain protection, they are not consistently applied by all domains. Not all administrators have put these records into place.

Enhanced filtering is a specific configuration for a specific mail flow scenario. This scenario comes into play when an organization has a mail hygiene or other mail filtering service in front of Exchange Online such as ProofPoint, Mimecast, MessageLabs and others. Why is this different? First, let's review some email flow scenarios:

**MX points to EOP:** In this scenario, email will flow from an external recipient to EOP and then to the user's mailbox after it has been properly scanning. EOP uses it's full breadth of protection which includes things like anti-phishing, connection filtering, DNS lookups and other email authentication verification information. All of these are combined to return the proper signals on the email to be delivered. Or not!

**MX points to On-premises in Hybrid:** In this scenario, an external email will first be routed to an organizations on-premises Exchange Servers which will process the email and then route it to Office 365. Because Office 365 should trust an organizations own on-premises Exchange Servers, less analysis is done and the email does not pass through as many filters since the server is a trusted source.

**MX points to third-party service:** In this scenario, the email flows first to an service external to EOP which will scan the email, apply its own X-Headers and so on before handing the email off to EOP. When EOP receives the email, it might scan the email again for malicious content and then deliver the email message. The issue here is that EOP cannot properly perform email authentication as the email has already been intercepted and thus the

authentication details in the header will not match the details of the sending server.

Enhanced filtering is best put in place for the last scenario. This is also where Implicit Authentication comes into play. Implicit Authentication is an extension of the normal Email Authentication and Microsoft deploys their own custom algorithm to analyze these (per Microsoft):

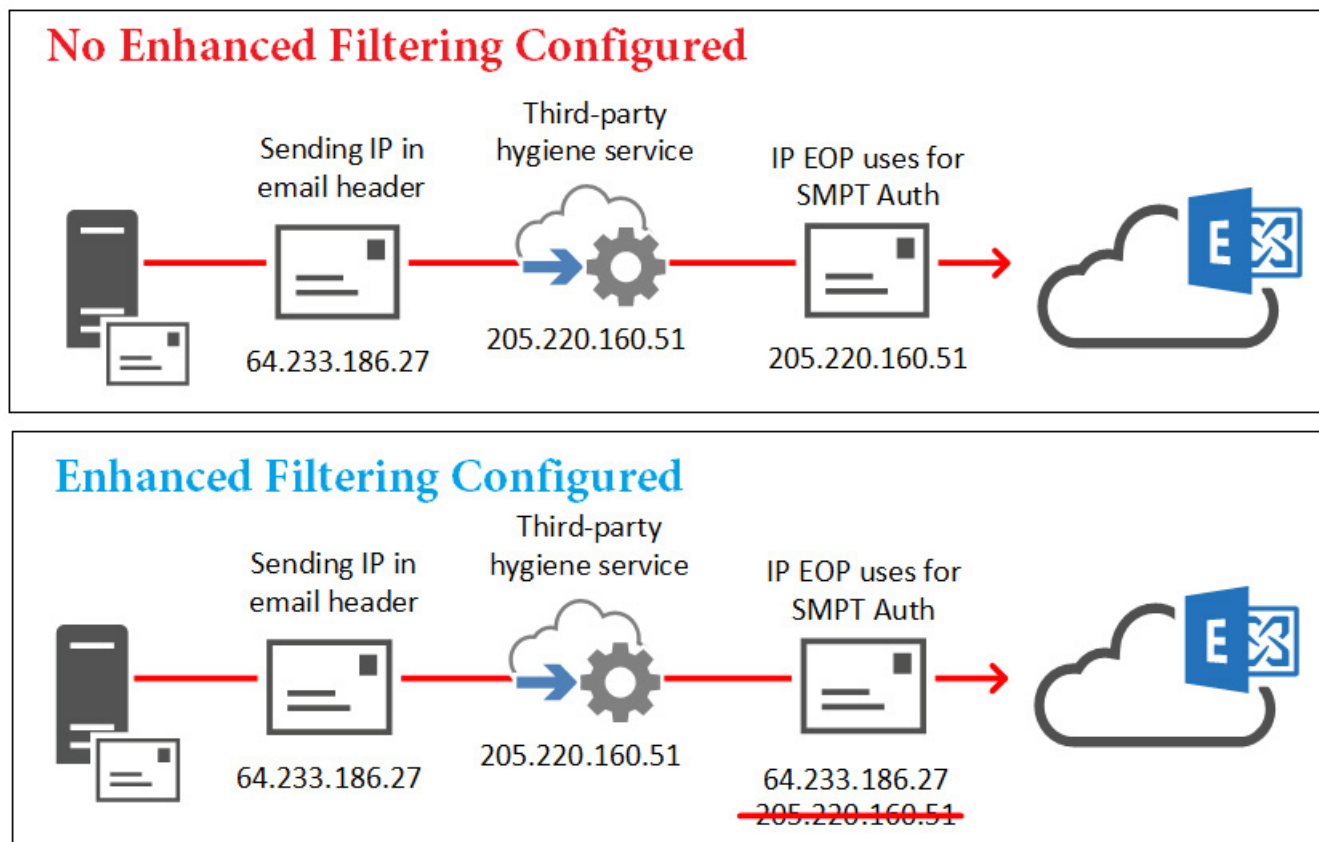
- Sender reputation
- Sender history
- Recipient history
- Behavioral analysis
- Other advanced techniques

Put together, Microsoft constructs what is known as Composite Authentication result if the Implicit Authentication checks that it performs. This result will show up as a header value in emails sent to someone in Exchange Online.

### Real World Header Sample:

```
was verified) header.d=mstechdiscussions.onmicrosoft.com;practicalpowershell.com;
dmarc=pass action=none header.from=microsoft.com;compauth=pass reason=100
Received-SPF: Pass (protection.outlook.com: domain of mstechdiscussions.com
```

Possible values for the Compauth header value are none, pass, fail and softpass. Additionally, a reason code is stamped into the same header, mostly for Microsoft internal processes. Why does this matter? Simply put, if there is another hygiene service between the sending server and EOP, the SMTP authentication results could be wrong and misinterpreted. See below for the ways in which these are interpreted with EOP:



There are two way to handle this scenario. One we can create a Transport Rule to allow this connection in to

bypass spam detection rules or two, we can use the Enhanced Filtering Feature found in the Security Center:

Policies & rules > Threat policies > Enhanced Filtering for Connectors

Enhanced Filtering for Connectors allows you to filter email based on the actual source of messages that arrive over the connector. Enhanced Filtering skips the source IP addresses of the connector and looks back in the routing path to determine the actual source of the incoming messages. Learn more at [Enhanced Filtering for Connectors](#).

Refresh 3 items Search

Connector Name	Enhanced filtering	Applies To
Inbound from 7df5bdad-2b70-...	● Off	

To configure this with PowerShell, we need to connect to Exchange Online PowerShell (Connect-ExchangeOnline) and then take a look at cmdlets that work with Inbound Connectors. First, what settings are enabled by default on an Inbound Connector:

```
Get-InboundConnector | ft EF*
```

```
PS C:\> Get-InboundConnector |ft EF*
EFTestMode EFSkipLastIP EFSkipIPs EFSkipMailGateway EFUsers
-----
False      False {}          {}          {}
```

**NOTE:** 'EF' was chosen because the values we need to review are for Enhanced Filtering (EF).

Thus we see that no value are configured, Test and Skip Last IP are set to False and other values are blank. Let's take the example from the before with the hygiene service located at an IP of 205.220.160.51 and we configure skip listing/Enhanced Filtering for this service with PowerShell. First option we can choose is to allow EOP to automatically detect the last IP address using the header, which in this case should be the service we've placed in front of EOP, and just enable Enhanced Filtering like so:

```
Set-InboundConnector -Identity "Third-party hygiene service" -EFSkipLastIP $True
```

Otherwise we can specify IP addresses of the provider and add that into this cmdlet as well:

```
Set-InboundConnector -Identity "Third-party hygiene service" -EFSkipLastIP $False -EFSkipIPs
205.220.160.51
```

Configured so, all recipients in your tenant will receive the same treatment. However, if this needs to be scoped, say for testing or specific scenarios, we can also add a list of email addresses to have the settings apply to just those users:

```
Set-InboundConnector -Identity "Third-party hygiene service" -EFSkipLastIP $True -EFUsers damian@practicalpowershell.com,dave@practicalpowershell.com
```

```
EFTestMode      : False
EFSkipLastIP    : True
EFSkipIPs       : {}
EFSkipMailGateway : {}
EFUsers         : {damian@practicalpowershell.com, dave@practicalpowershell.com}
```

Note that if configuring EFSkipIPs, the EFSkipLastIP value must be False.

The EFSkipMailGateway property is an interesting feature that allows us to specify a well-known provider. Current options are:

proofpoint, symanteccloud, barracuda, mimecast, sophos, trendmicro, ciscoironportcloud, ironport

However, this property is only for Microsoft Internal use only.

## Office 365 Advanced Threat Protection Recommended Configuration Analyzer (ORCA)

One of the challenges of managing an email system in the cloud is knowing what best practices should be applied. If a tenant has E5 licenses, then an additional set of features are available to configure, named the Advance Threat Protection suite. Microsoft field techs have created a PowerShell module to help with assessing your configuration and recommending appropriate actions. This module is known as ORCA or Office [atp] Recommended Configuration Analyzer. We are provided with two versions of this module. One I would consider is the Stable/tested version and the other is a Preview version if you want to be on the bleeding edge of the testing. Let's start by reviewing these modules.

### Preview vs Stable Versions

One key thing to be aware of with ORCA is that there are two different versions of the PowerShell module. There is the stable version and then there is a Preview version for testing. Which one you use is up to the operator, you. From testing so far I have not seen any issues with either, but your mileage may vary.

Production (Stable) – <https://www.powershellgallery.com/packages/ORCA>

Preview (Test) – <https://www.powershellgallery.com/packages/ORCAPreview>

### How to Install

Like any other modern PowerShell module / modern PowerShell version, we can use the Install-Module cmdlet to install this new module on our computer:

ORCA or ORCAPreview. Sample syntax:

```
Install-Module -Name ORCA
```

```
Install-Module -Name ORCAPreview
```

Once the install has completed, we can load the module again and run the module.

## How to Update

The ORCA module is a downloadable module that comes from a PowerShell repository and because of this, the module is an update-able module. Periodically this version of ORCA and ORCAPreview will increment as additional features and bug fixes are added. These PowerShell modules can be updated like any other PowerShell module that has been installed from a PSRepository:

```
Update-Module -Name ORCA
Update-Module -Name ORCAPreview
```

Once the update has been performed, we can load the module again and run the module.

**NOTE:** The below example will work for either the Preview or the Stable version of the module. Simply replace ORCAPreview with ORCA and run the cmdlets.

## How to check your version vs what is available

Let's take for example that you currently have the 1.6.3 version of the ORCAPreview PowerShell module installed. At the time we may not have checked to see if there was a newer version and we do not have the page bookmarked to look at on the web. Luckily we can easily check to see if there is a newer version with two options:

### Using What if:

```
Get-InstalledModule OrcaPreview | Update-Module -WhatIf
```

Which shows us this:

```
PS C:\> Get-InstalledModule OrcaPreview | Update-Module -WhatIf
What if: Performing the operation "Update-Module" on target "Version '1.6.3' of module 'ORCAPreview', updating to version '1.6.5'".
```

So we see that there is indeed a newer version. Which we did not upgrade to. Or, just updating without the WhatIf:

```
Get-InstalledModule OrcaPreview | Update-Module
```

Which just updates the module:

```
PS C:\> Get-InstalledModule OrcaPreview | Update-Module
PS C:\>
```

Looks like it installed with no feedback? Let's check:

```
PS C:\> Get-InstalledModule OrcaPreview

Version          Name              Repository
-----          -
1.6.5            ORCAPreview      PSGallery
```

Now what if we already had 1.6.5 installed? Would we get any feedback from either cmdlet?

```
PS C:\> Get-InstalledModule OrcaPreview | Update-Module -WhatIf
PS C:\> Get-InstalledModule OrcaPreview | Update-Module
PS C:\>
```

Nope. So if you need to automate this and need a check in your script, use the WhatIf as it will at least provide

some sort of feedback if checking for a new version. OR. Just update it every time the script runs and then load the module to make sure the latest version is available to the script.

Finally we could use Find-Module to check what the latest version in a PowerShell repository is, like so:

Find-Module ORCA

```
PS C:\> Find-Module ORCA
```

Version	Name	Repository	Description
1.6.3	ORCA	PSGallery	Office 365 Advance...

## How to Use ORCA

Between the Stable version and the Preview version there is only one difference and that is the cmdlet to run to execute the test. Keep in mind that each module only has ONE cmdlet. There are no other cmdlets included.

Stable and Preview Versions:

Get-ORCAReport

If you execute this cmdlet AND you are not logged into any tenant, you will be asked for credentials and then the script will run through all of its checks.

As we can see from the sample run on a test tenant, there are a lot of checks that are performed and a lot of those are targeted to Advanced Threat Protection (ATP) settings:

```
04/26/2020 10:40:57 Performing ORCA Version check...
Creating a new Remote PowerShell session using MFA for implicit remoting of "Get-Mailbox" command ...
04/26/2020 10:41:04 Getting Anti-Spam Settings
04/26/2020 10:41:05 Getting Tenant Settings
04/26/2020 10:41:05 Getting Anti Phish Settings
04/26/2020 10:41:05 Getting Anti-Malware Settings
04/26/2020 10:41:06 Getting Transport Rules
04/26/2020 10:41:06 Getting ATP Policies
04/26/2020 10:41:08 Getting Accepted Domains
04/26/2020 10:41:08 Getting DKIM Configuration
04/26/2020 10:41:08 Analysis - Advanced Threat Protection Policies - Mailbox Intelligence Protection
04/26/2020 10:41:08 Analysis - Advanced Threat Protection Policies - Mailbox Intelligence Protection Action
04/26/2020 10:41:08 Analysis - Advanced Threat Protection Policies - Action for unknown potentially malicious URLs in messages
04/26/2020 10:41:08 Analysis - Advanced Threat Protection Policies - Anti-spoofing protection action
04/26/2020 10:41:08 Analysis - Advanced Threat Protection Policies - Do not let users click through safe links
04/26/2020 10:41:08 Analysis - Advanced Threat Protection Policies - Anti-spoofing protection
04/26/2020 10:41:08 Analysis - Advanced Threat Protection Policies - Intra-organization Safe Links
04/26/2020 10:41:08 Analysis - Advanced Threat Protection Policies - Safe Links Tracking
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Unusual Characters Safety Tips
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Safe attachments unknown malware response
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Office Enablement
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Similar Domains Safety Tips
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Safe Attachments SharePoint and Teams
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Safe Links Policy Rules
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Similar Users Safety Tips
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Safe Links Synchronous URL detonation
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Anti-phishing trusted domains
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Anti-phishing trusted senders
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Safe Attachments Policy Rules
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - User Impersonation Action
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Safe Links Whitelisting
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Unauthenticated Sender (tagging)
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Safe Attachment Whitelisting
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Mailbox Intelligence Enabled
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Domain Impersonation Action
04/26/2020 10:41:09 Analysis - Advanced Threat Protection Policies - Advanced Phishing Threshold Level
04/26/2020 10:41:09 Analysis - Content Filter Policies - High Confidence Spam Action
04/26/2020 10:41:09 Analysis - Content Filter Policies - Safe Links
```

```

04/26/2020 10:41:09 Analysis - Content Filter Policies - Safety Tips
04/26/2020 10:41:09 Analysis - Content Filter Policies - Phish Action
04/26/2020 10:41:09 Analysis - Content Filter Policies - Bulk Action
04/26/2020 10:41:09 Analysis - Content Filter Policies - Spam Action
04/26/2020 10:41:09 Analysis - Content Filter Policies - Quarantine retention period
04/26/2020 10:41:09 Analysis - Content Filter Policies - High Confidence Phish Action
04/26/2020 10:41:09 Analysis - Content Filter Policies - Domain Whitelisting
04/26/2020 10:41:09 Analysis - Content Filter Policies - End-user Spam notifications
04/26/2020 10:41:10 Analysis - Content Filter Policies - IP Allow Lists
04/26/2020 10:41:10 Analysis - Content Filter Policies - Mark Bulk as Spam
04/26/2020 10:41:10 Analysis - Content Filter Policies - Allowed Senders
04/26/2020 10:41:10 Analysis - Content Filter Policies - Bulk Complaint Level
04/26/2020 10:41:10 Analysis - Content Filter Policies - Outbound spam filter policy settings
04/26/2020 10:41:10 Analysis - Content Filter Policies - Advanced Spam Filter (ASF)
04/26/2020 10:41:10 Analysis - DKIM - DNS Records
04/26/2020 10:41:10 Analysis - DKIM - Signing Configuration
04/26/2020 10:41:10 Analysis - Malware Filter Policy - Internal Sender Notifications
04/26/2020 10:41:10 Analysis - Malware Filter Policy - External Sender Notifications
04/26/2020 10:41:10 Analysis - Malware Filter Policy - Common Attachment Type Filter
04/26/2020 10:41:10 Analysis - Tenant Settings - Unified Audit Log
04/26/2020 10:41:10 Analysis - Transport Rules - Domain Whitelisting
04/26/2020 10:41:10 Analysis - Zero Hour Autopurge - Supported filter policy action
04/26/2020 10:41:10 Analysis - Zero Hour Autopurge - Zero Hour Autopurge Enabled for Spam
04/26/2020 10:41:10 Analysis - Zero Hour Autopurge - Zero Hour Autopurge Enabled for Malware
04/26/2020 10:41:10 Analysis - Zero Hour Autopurge - Zero Hour Autopurge Enabled for Phish
04/26/2020 10:41:10 Generating Output
04/26/2020 10:41:12 Complete! Output is in C:\Users\... \AppData\Local\Microsoft\ORCA\ORCA-scoles-202004261041.html
PS C:\>
    
```

Once the script is complete, a new HTML file will open up in your chosen browser. This report is quite detailed with Recommendations and a list of items that checked out OK. It is color coded as well with green for OK and yellow for Recommended actions. Each is listed in a subcategory with numerical counters to indicate how many items need to be reviewed after the report is run. Sample report and explanation:

**Tenant Name** (blue arrow pointing to `scoles.onmicrosoft.com`)

**ORCA Version** (red arrow pointing to `Version 1.6.3`)

**Total recommended actions to be taken.** (red arrow pointing to `25` Recommendations)

**Total items configured per current best practices.** (red arrow pointing to `27` OK)

**summary of configuration areas as well as recommendations / correctly configured items.** (red arrow pointing to the Summary table)

Area	Recommendations	OK
Content Filter Policies	7	8
Advanced Threat Protection Policies	14	12
DKIM	1	1
Malware Filter Policy	3	0



As we can see from the top of the report, we have a total of recommendations as well as a total of best practices that are being followed. Take this list with a grain of salt. These are Microsoft's general and broad recommendations. You do not have to do anything if you do not want to. While some are common sense, some may not be what your users would expect or want. Also displayed is the version of the ORCA module in use, the Office 365 tenant name and other summary information. If you are running an old version, this should be displayed at the top:

ORCA is out of date. You're running version 1.6.3 but version 1.6.5 is available! Run Update-Module ORCA to get the latest definitions!

And if you run the preview version, an additional box will be shown as well:

You are running a preview version of ORCA! Preview versions may contain errors which could result in an incorrect report. Verify the results and any configuration before deploying changes.

Scrolling down on the report, we will see each recommendation or best practice followed. All items are color coded for easy reading:

Main category

Subcategory

Color coded. Yellow is for one of the Recommended actions.

Content Filter Policies

Quarantine retention period

Configure the Quarantine retention period to 30 days **Improvement**

You can view, release, download, delete and report false positive quarantined email messages or files captured by Advance Threat Protection (ATP) for SharePoint Online, OneDrive for Business, and Microsoft Teams in Office 365. Keep messages in the quarantine for 30 days to allow enough time for further investigation. This is the default value and also the maximum.

Effected objects

Content Filter Policy	Quarantine Retention Period	
Spam Filter - Custom 1	30	Standard ✓
Default	20	Not Recommended ✗

Recommended setting. In this case a change from the default values

Manage quarantined messages and files as an administrator in Office 365

Recommended settings for EOP and Office 365 ATP security

Below is a sampling of the items that were checked:

Change Phish action to Quarantine message
Change High Confidence Spam action to Quarantine message
Enable End-user Spam notification and set the frequency to 3 days
Set the Bulk Complaint Level threshold to be 6
Turn off Advanced Spam filter (ASF) options in Content filter policies
Set RecipientLimitExternalPerHour to 500, RecipientLimitInternalPerHour to 1000, and ActionWhenThresholdReached to block.
Spam action set to move message to junk mail folder or quarantine
Safety Tips are enabled
Bulk action set to Move message to Junk Email Folder
High Confidence Phish action set to Quarantine message
Bulk is marked as spam
Senders are not being whitelisted in an unsafe manner
Domains are not being whitelisted in an unsafe manner
No IP Allow Lists have been configured
Enable Similar Users Safety Tips so that users can receive visible indication on incoming messages
Enable Safe Attachments for SharePoint and Teams
Enable Unusual Characters Safety Tips so that users can receive visible indication on incoming

Reading through the list we see there are a lot of good recommendations and settings for a tenant. There are also settings that are being deprecated, where this tool provides notifications of that change. In the case of my report, the tenant is un-configured and solely for testing. Your own report will look vastly different and may have items that are flagged for change when in fact they are what you need for your own tenant.

### Parameters and Switches for Get-ORCAReport

Now, since this is a custom module and custom cmdlet by Microsoft, we need to see if there are any parameters that are available, or does it kick off some automated process. First we can review available parameters for the cmdlet with a Ctrl+Space as shown below:

```
PS C:\> Get-ORCAReport -NoConnect
NoConnect NoUpdate NoVersionCheck AlternateDNS Collection
```

OK. So we have NoConnect, NoUpdate, NoVersionCheck, AlternateDNS and Collection. Let's review each option:

**NoConnect:** If we already have an open connection to Exchange Online, we can use this option to prevent the cmdlet from opening that connection again.

**NoUpdate:** Prevents the script from exiting if its not up to date.

**NoVersionCheck:** Do not check for old versions.

**AlternateDNS:** Can specify different DNS servers than what the current computer has configured.

**Collection:** For processing a pre-created object collection.

**NOTE:** In the examples section of the Get-Help for the cmdlet, we see that there is also a -Report parameter for specifying a report name. We can instead use -Output and specify a destination HTML file for the report like so:

```
Get-ORCA Report -Order c:\scripts\ORCA-Output.html
```

## Cleanup

What if you have a computer where you were using ORCA / ORCAPreview for quite some time and wanted to remove older versions to prevent any confusion for you or your scripts? First, we would need to see what versions are available on the computer:

```
Get-Module -ListAvailable |where {$_.Name -like 'ORCA*'}
```

```
Directory: C:\Users\james\Documents\WindowsPowerShell\Modules

ModuleType Version      Name                               ExportedCommands
-----
Manifest    1.6.3         ORCA                               Get-ORCAReport
Manifest    1.6.5         ORCAPreview                        Get-ORCAReport
Manifest    1.6.3         ORCAPreview                        Get-ORCAReport

Directory: C:\Program Files\WindowsPowerShell\Modules

ModuleType Version      Name                               ExportedCommands
-----
Manifest    0.3.2         ORCA                               Get-ORCAReport
```

Notice I have ORCA versions 0.3.2 and 1.6.3 as well as ORCAPreview versions 1.6.3 and 1.6.5. Also see the one cmdlet on the right for all of the modules. Now, how do we remove a specific module? We would need to specify the versions somehow with the Uninstall-Module cmdlet. But how? Let's see what is available to us:

```
PS C:\> Uninstall-Module -Name
Name                AllVersions      Verbose          ErrorVariable    PipelineVariable
-----
InputObject         Force           Debug           WarningVariable
MinimumVersion      AllowPrerelease  ErrorAction     InformationVariable
RequiredVersion     WhatIf          WarningAction   OutVariable
MaximumVersion      Confirm         InformationAction OutBuffer
```

So we have Maximum and Minimum Version options. Minimum would work, but if we specify a very low version number, would we be initiating an uninstall for all module with a version number greater than that? So for ORCA if we specify 0.3.2, would it uninstall 0.3.2 AND 1.6.3? Better not take a chance. Our other option is MaximumVersion. We could then cap off where the uninstall process stops. So in our case we would need these two cmdlets:

```
Uninstall-Module ORCAPreview -MaximumVersion 1.6.3
Uninstall-Module ORCA -MaximumVersion 0.3.2
```

As always, be aware of your session and if you are running it as an Administrator as you may need elevated rights for some operations:

```
PS C:\> Uninstall-Module ORCAPreview -MaximumVersion 1.6.3
PS C:\> Uninstall-Module ORCA -MaximumVersion 0.3.2
PackageManagement\Uninstall-Package : You cannot uninstall the module 'ORCA' from 'C:\Program Files\WindowsPowerShell\Modules\ORCA\0.3.2' because Administrator rights are required.
```

And once the cmdlets run properly, we see that there are only two versions left.

**NOTE:** For your operation, you may need one or more versions of a module installed. The above is just an exercise in cleaning up older versions if that is something that you would need to do.

```
PS C:\> Get-Module -ListAvailable | Where {$_.Name -Like 'ORCA*'}

Directory: C:\Users\██████████\Documents\WindowsPowerShell\Modules

ModuleType Version      Name                ExportedCommands
-----
Manifest    1.6.3         ORCA                Get-ORCAReport
Manifest    1.6.5         ORCAPreview         Get-ORCAReport
```

Further reading on ORCA:

<https://github.com/cammurray/orca>



<https://docs.microsoft.com/en-us/microsoft-365/security/office-365-security/recommended-settings-for-eop-and-office365-atp>

## Configuration Analyzer

We just covered the ORCA script and explored how to download, run and use the script to create useful reports on an Office 365 tenant's configuration. Recently Microsoft added a button to the Security Center which does an evaluation similar to that of the ORCA script. We can find that button under **Email & Collaboration > Policies and Rules > Threat Policies > Configuration Analyzer**:

### Threat policies

**Templated policies**

	Preset Security Policies	Easily configure protection by applying all policies at once using our recommended protection templates
	Configuration analyzer	Identify issues in your current policy configuration to improve your security






### How to run the analyzer?


Simple. Click on Configuration Analyzer and an analysis of your tenant will be performed, like so:

### Configuration analyzer

The configuration analyzer can help identify issues in your current configuration, and help improve your policies for better security. [Learn more.](#)

Standard recommendations
Strict recommendations
Configuration drift analysis and history

 <b>3</b>	 <b>10</b>	 <b>1</b>	 <b>3</b>	 <b>2</b>
----------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------

 Refresh

Recommendations	Policy	Policy group/setting name	Policy type
Change 2 Entries to 0 Entries	Default	Allowed Senders	Anti-spam
Change 2 Entries to 0 Entries	AntiSpam - Practical PowerShell Doma...	Allowed Senders	Anti-spam
Enable end-user spam notifications	AntiSpam - Practical PowerShell Doma...	Enable end-user spam notifications	Anti-spam

Couple notes that can be made on this screenshot: first, we see tabs for ‘Standard recommendations’, the default, ‘Strict recommendations’ and ‘Configuration drift analysis and history’. This new layout provides a better list like view, which is similar to what we see in the Secure Score as well as the new Compliance Score.

If we review the recommendations, we see that there are columns for the current configuration and what it applies to, whether or not the change has been made and so on. We have two options that we can choose for each line item, we can ‘Apply recommendation’ to make the proposed change, or we can simply view the policy for a better explanation:

**View:** Takes us to the policy at hand and allows an analysis of the configuration before applying the change.

**Apply:** Whereas Apply will allow us to accept the recommended change:



What else do we get? Well if we look at The ‘Configuration drift analysis and history’ tab at the top, we see how our configuration analysis has progressed over time:

Home > Policy > Configuration analyzer

### Configuration analyzer

The configuration analyzer can help identify issues in your current configuration, and help improve your policies for better security. [Learn more.](#)

Setting and recommendations Configuration drift analysis and history

↓ Export ↻ Refresh 108 items 🔍 Search ⏏ Filter

Applied filters: Date: 9/4/2020-12/3/2020 Type: Standard protection

Last modified	Modified by	Setting name	Policy	Type	Configuration c
Oct 21, 2020 11:49 PM	damian...	Turn on ATP for SharePoint, OneDrive, and Microsoft Teams	Default	AtpPolicyForO365	N/A -> False
Oct 21, 2020 11:49 PM	damian...	Turn on Safe Documents for Office clients	Default	AtpPolicyForO365	N/A -> False
Oct 21, 2020 11:49 PM	damian...	Allow people to click through Protected View even if Safe Docume...	Default	AtpPolicyForO365	N/A -> False
Oct 18, 2020 10:06 PM	damian...	Turn on ATP for SharePoint, OneDrive, and Microsoft Teams	Default	AtpPolicyForO365	False -> True
Oct 18, 2020 10:06 PM	damian...	Turn on Safe Documents for Office clients	Default	AtpPolicyForO365	False -> True
Oct 18, 2020 10:06 PM	damian...	Allow people to click through Protected View even if Safe Docume...	Default	AtpPolicyForO365	False -> True
Oct 18, 2020 5:24 PM	damian...	Bulk email threshold	AntiSpam - Practical PowerS...	Anti-spam	N/A -> 6
Oct 5, 2020 11:09 AM	damian...	Turn on ATP for SharePoint, OneDrive, and Microsoft Teams	Default	AtpPolicyForO365	True -> False
Oct 5, 2020 11:09 AM	damian...	Turn on Safe Documents for Office clients	Default	AtpPolicyForO365	True -> False

Also, we can deep dive as far back as three months if we wish to explore that far. This page provides a good audit trail for your configuration and may even aid in troubleshooting an issue if a change was found on a particular date.

What is notably different between this and ORCA is that ORCA provides more background information on the issues that are found. For example, on the ‘Enable end user spam notifications, we see these block of articles to read

as well as the link to the configuration page for Anti-Spam settings:

- [Configure end-user spam notifications in Exchange Online](#)
- [Security & Compliance Center - Anti-spam settings](#)
- [Recommended settings for EOP and Office 365 ATP security](#)



The Configuration Analyzer does not provide the knowledge links. Now, ORCA also has a flaw in that it does not show the changes that need to be made as does the Config Analyzer. Overall they complement each other and are geared towards different groups of people as one is GUI based and the other is driven via PowerShell.

## Microsoft Defender for Office 365 Evaluation Mode

A new addition to the Security and Compliance Center (as well as the Security Admin Center) is the evaluation mode for Microsoft Defender for Office 365. If you remember, the Advanced Threat Protection product was renamed to Defender for Office 365. As a way to help administrators test out this feature in Office 365, Microsoft came up with an evaluation mode that will let an organization run Defender for Office 365 in evaluation mode to see how its features and functions stack up.

We can find this in the Security Center under **Email & Collaboration > Policies and Rules > Threat Policies > Evaluate Defender for Office 365:**

### Others

- |                                                                                     |                                |                                                                                           |
|-------------------------------------------------------------------------------------|--------------------------------|-------------------------------------------------------------------------------------------|
|  | User reported message settings | Enable end users to report spam and malicious email for review and analysis               |
|  | Evaluation mode                | Configure Microsoft Defender for Office 365 without impacting your production environment |

Clicking on this button brings us to a simple scenario configuration:

## Microsoft Defender for Office 365

- Routing settings
- Review your settings

### Routing settings

You can protect your email environment and control mail routing when you have a mix of on-premises and cloud mailboxes. If you're using Exchange Online, see [mail routing](#) for details.

I'm using a third party or on-premises service provider ⓘ

I'm using Microsoft Exchange ⓘ

Note that there is no explanation or link to a Microsoft Doc page. It can be found [ [here](#) ].

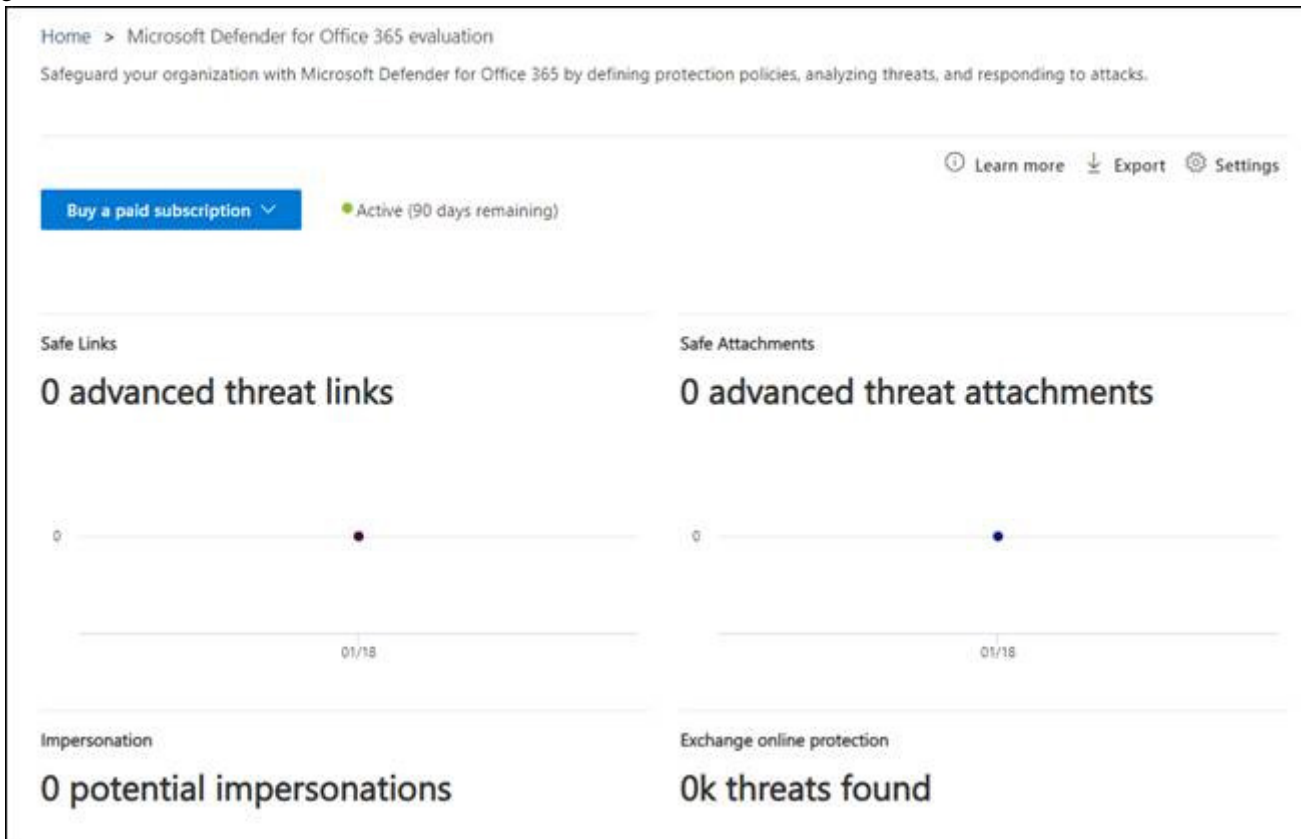
This feature is also in Public Preview at this time and when enabled, it will run for approximately 90 days. Microsoft also states the following about the evaluation:

- The configuration of Microsoft Defender for Office 365 will be in the background and not visible to the administrator.

- Enhanced Filtering is configured if using a third-party filtering solution.
- Daily reports will be created and sent for up to 90 days about what protection would have been provided.
- Qualified licensing is required Plan 1 or 2, Microsoft 365 E5 or Office 365 E5. Otherwise, a Trial License is required to use this feature. A trial license will only provide 30 days of evaluation for this feature.
- URLs will be detonated and exclusions may need to be made.
- Scoping can be provided with an Inbound connector.

## Reporting

Once an evaluation is created, we are provided a Dashboard to review current findings which replaces the configuration section we used to start the evaluation:



Once real live data gets process, we will start seeing results in the dashboard like so:



Additionally we can request a report on these findings, which will send an email with a link to download and evaluate:



Once the CSV is downloaded, open it up in Excel and see these results:

A	B	C
Date	Sender	Subject
1/29/2021 21:46	v-bhenmli_icojgblab_hbfmiejo_hbfmiejo_a@bounce.em.sierratradingpost.com	Beat the cold with epic SAVINGS
1/29/2021 21:45	bounce-mc.us20_114270466.112030-c3da7ef928@mail87.atl51.rsgsv.net	Nicholas: 2021 Regalia Recommended For You P+
1/29/2021 20:00	<>	Tartan Testing Progress Update
1/29/2021 13:14	bounce-21974_HTML-272395485-2187066-6240568-636@bounce.email.nationstarmail.com	You may qualify for a lower mortgage payment.
1/28/2021 20:48	heartland@mail198.connectingdonors.org	You make an impact each time you donate blood
1/28/2021 18:06	<>	\$10 Off Again!
1/28/2021 14:09	1axb4fdhm4ach3mnjxbgxfrc9nib5vt7duy2jv-nicholas=m365securitybook.com@pgg3jc.cs.hubs	More Testosterone - For Half the Price.
1/27/2021 22:33	bounce-69181_HTML-217078340-1537560-520000649-5233@bounce.inboxdollarsemail.com	New day + new question = Easy cash
1/26/2021 21:56	16116858029374773-125541-1-practicalpowershell.com@delivery.mail.sheerse.com	SEO Analysis for practicalpowershell.com
1/26/2021 16:08	346ae7fc-e450-11e6-afd8-a45e60e86e11@bounce.r.livingsocial.com	20 Units of Jeuveau + Up to 20% OFF
1/26/2021 14:20	v-bheimmo_icojgblab_hbfijdlg_hbfijdlg_a@bounce.em.sierratradingpost.com	Shoes on shoes on SHOES
1/26/2021 13:05	bounce_hibfobl_o-sarah=m365securitybook.com@cp20.com	Best of the Best from Bell's Best Cookbook
1/25/2021 12:26	bounces+6630771-d74f-sarah=m365securitybook.com@sg-email.robinhood.com	"<∅ß Walmart, Amazon, and the vaccine rollout "
1/23/2021 15:51	robin.appexpert@aol.com	## Mobile Apps ##

E	F
NetworkMessageId	Technology
7e04e953-3fbe-4659-3387-08d8c49f4a85	Spam
5fa69275-8d2e-4500-ee40-08d8c49f2cfa	Unauthenticated
ebbc95b6-c68f-46d1-4459-08d8c4909339	Phish
7446914f-a27f-4441-0512-08d8c457c971	Spam
43f3abe3-f222-4898-ea9a-08d8c3ce0dab	Spam
60310770-c673-4db6-f27e-08d8c3b773d8	Unauthenticated
7e5c1282-db96-4a3a-df2d-08d8c3964113	Spam
21fcdbea-b835-489f-4692-08d8c313a177	Spam
f04d85c7-7fa3-4039-6081-08d8c2453eb6	Spam
9cc75347-ecb7-43d1-6cea-08d8c2148c77	Spam
f765cdf-9f9a-4da3-3e83-08d8c2058d43	Spam
7a953008-5dd3-4b62-a2f3-08d8c1fb02eb	Unauthenticated
21f3e6b1-52a3-4998-1e0f-08d8c12c7cca	Spam
79919b4c-9b43-4edf-4261-08d8bfb6c96e	Spam

From this evaluation CSV file, we can review what technology was applied (Column F) as well as the NetworkMessageId that can be used in Investigations in the Security and Compliance Centers.

## PowerShell

The new Evaluation Mode can also be managed somewhat from PowerShell. There are three cmdlets that are provided for this feature:

```
Add-ATPEvaluation
Get-ATPEvaluation
Remove-ATPEvaluation
```

By default a brand new Greenfield tenant will not have an existing ATP Evaluation to query. We can however set



one up by simply running the Add-ATPEvaluation cmdlet. When we run this cmdlet we get a 90 day evaluation that we can now use to test things out. What do we get with just using the defaults?

```
PS C:\> Add-ATPEvaluation

RunspaceId      : 36f5c7fd-c371-4bb1-aa36-cc7fb0627fde
Identity        : e547ab14-76b5-49fe-84d5-08d9772e2592
StartTime       : 9/14/2021 3:17:13 AM
ExpiryTime      : 12/13/2021 3:17:13 AM
TenantId        : 
Event           : Enabled
InboundConnectorNameHash : 
EvaluatedDomain : 
ShareWithMicrosoft : False
```

The provided start and end date are 90 days apart as expect for the evaluation. Additionally the 'SharedWithMicrosoft' setting is set to False. Since we have no Set cmdlet, we can only really kick off an evaluation, list an evaluation or remove it. When using the Remove-ATPEvaluation cmdlet, the existing evaluation is set to disabled:

```
PS C:\> Get-ATPEvaluation

RunspaceId      : 36f5c7fd-c371-4bb1-aa36-cc7fb0627fde
Identity        : 
StartTime       : 9/14/2021 3:23:48 AM
ExpiryTime      : 12/13/2021 3:23:48 AM
TenantId        : 
Event           : Disabled
InboundConnectorNameHash : 
EvaluatedDomain : 
ShareWithMicrosoft : False
```

When adding or setting a new evaluation, it may be worth it to set some other parameters for the test. Here are some of the available settings and what can be done with them:

**InboundConnectorName:** If there is a specific connector that you wish to scope the evaluation to, use this parameter.

**RecipientDomain:** The evaluation can also be scoped to one of the domains in the tenant for testing.

**ShareWithMicrosoft:** Set this to True (default is False) to allow Microsoft to collect data on this evaluation.

You can also create more than one evaluation if that is desired, perhaps to set different conditions based on connector or domain.