Exchange Outlook Online - Give Users Send As Permissions

Install and Configure Windows PowerShell

Applies to: *Office 365 for professionals and small businesses, Office 365 for enterprises, Live@edu*

Topic Last Modified: *2010-05-18*

Before you can use Windows PowerShell, make sure you have the correct versions of Windows PowerShell and Windows Remote Management (WinRM) installed and configured on your computer. You have to use the Windows Management Framework, which contains the correct versions of Windows PowerShell v2 and WinRM 2.0.

If your computer is running Windows 7 or Windows Server 2008 R2, you don't have to install anything. The Windows Management Framework is already installed.

You can download and install the Windows Management Framework if your computer is running one of the following operating systems:

- Windows Vista Service Pack 1 (SP1) or SP2
- Windows Server 2008 SP1 or SP2
- Windows Server 2003 SP2
- Windows XP SP3

Let's get started:

Before you can install the Windows Management Framework, you have to uninstall any existing versions of Windows PowerShell.

Note This step isn't required for Windows 7 or Windows Server 2008 R2.

Uninstall Windows PowerShell from Windows Vista

---

1. In Control Panel, in Programs, open Programs and Features, and uninstall any instances of Windows PowerShell that appear in the installed programs list. For example, the Community Technology Preview (CTP) version of Windows PowerShell v2 may appear as Windows PowerShell (TM) V2.
2. Under Tasks, select View installed updates and uninstall any instances of Windows PowerShell that appear in the installed updates list. For example, Windows PowerShell V1 may appear as a Windows update with one of the following Microsoft Knowledge Base article numbers:
    - KB928439
    - KB923569

Uninstall Windows PowerShell from Windows Server 2008

1. Start Server Manager and go to Features.
    1. Click Uninstall Features.
    2. Select Windows PowerShell and follow the directions to uninstall.
2. In Control Panel, in Programs, open Programs and Features, and uninstall any instances of Windows PowerShell that appear in the installed programs list.
3. Under Tasks, select View installed updates. Uninstall any instances of Windows PowerShell that appear in the installed updates list.

Top of page

Uninstall Windows PowerShell from Windows Server 2003 and Windows XP

1. In Control Panel, open Add or Remove Programs, and uninstall any instances of Windows PowerShell that appear in the installed programs list.
2. In Add or Remove Programs, select Show updates. Uninstall any instances of Windows PowerShell that appear in the installed updates list. For example, Windows PowerShell V1 may appear as a Windows update with the Knowledge Base article number KB926139.

Top of page

Before you can install the Windows Management Framework, you must uninstall any existing versions of WinRM.

Note This step isn't required for Windows 7 or Windows Server 2008 R2.

Uninstall WinRM from Windows Vista or Windows Server 2008

1. In Control Panel, in Programs, open Programs and Features, and uninstall any instances of Windows Remote Management that appear in the installed programs list.
2. Under Tasks, select View installed updates. Uninstall any instances of Windows Remote Management that appear in the installed updates list. For example, the Community Technology Preview (CTP) of WinRM 2.0 may appear as WindowsRemoteManagement with one of the following Knowledge Base article numbers:
    - KB936059
    - KB950099

Uninstall Windows PowerShell from Windows Server 2003 and Windows XP

1. In Control Panel, open Add or Remove Programs, and uninstall any instances of Windows Remote Management that appear in the installed programs list.
2. In Add or Remove Programs, select Show updates. Uninstall any instances of Windows Remote Management that appear in the installed updates list. For example, WinRM might appear as a Windows update with the Knowledge Base article number KB936059.

Top of page

- Download and install the Windows Management Framework. Choose the package that includes Windows PowerShell v2 and WinRM 2.0, and that applies to your operating system, system architecture, and language.
  After you install WinRM and Windows PowerShell, configure the software to work correctly as described in the next steps.
  Note If your local computer is protected by a Microsoft Internet Security and Acceleration (ISA) server, you may have to install the Windows Firewall Client or configure a proxy server on your local computer to connect Windows PowerShell to the cloud-based service. For more information, see Windows PowerShell: FAQs for Administrators.

Top of page

1. Click Start > All Programs > Accessories > Windows PowerShell.
2. Do one of the following to open Windows PowerShell:
    - If you're running Windows Vista, Windows 7, or Windows Server 2008 R2, right-click Windows PowerShell and select Run as administrator. If you get a user account control prompt that asks if you would like to continue, respond Continue.
    - If you're running Windows XP or Windows Server 2003, click Windows PowerShell.
3. Run the following command:

```
Get-ExecutionPolicy
```

4. If the value returned is anything other than RemoteSigned, you need to change the value to RemoteSigned.
   Note When you set the script execution policy to RemoteSigned, you can only run scripts that you create on your computer or scripts that are signed by a trusted source.

Enable scripts to run in Windows PowerShell

In Windows PowerShell session you just opened as an administrator, run the following command:

```
Set-ExecutionPolicy RemoteSigned
```

Top of page

---

1. Click Start > All Programs > Accessories.
2. Do one of the following to open a command prompt:
    - If you're running Windows Vista, Windows 7, or Windows Server 2008 R2, right-click Command Prompt and select Run as administrator. If you get a user account control prompt that asks if you would like to continue, respond Continue.
    - If you're running Windows XP or Windows Server 2003, click Command Prompt.
3. At the command prompt, run the following commands:

```
net start winrm  winrm get winrm/config/client/auth
```

Note If the WinRM service is already running, you don't have to start it. You can check the status of the WinRM service by running the command sc query winrm.

4. In the results, look for the value Basic = . If the value is Basic = false, you must change the value to Basic = true.
   Note If you started the WinRM service, and you don't need to change the Basic value, run the command net stop winrm to stop the WinRM service.

Configure WinRM to support basic authentication

---

1. At the command prompt you just opened as an administrator, run the following commands. The value between the braces { } is case-sensitive:

```
winrm set winrm/config/client/auth @{Basic="true"}
```

2. In the command output, verify the value Basic = true.
   Note If you started the WinRM service, run the command net stop winrm to stop the WinRM service.

STEP 2

Connect Windows PowerShell to the Service

Applies to: *Office 365 for professionals and small businesses, Office 365 for enterprises,*

Live@edu

Topic Last Modified: *2013-01-24*

After you have installed and configured Windows PowerShell and Windows Remote Management (WinRM) on your computer, you have to connect the Windows PowerShell on your local computer to the cloud-based service to perform tasks in your cloud-based organization.

When you start Windows PowerShell, you're in the Windows PowerShell *session* of your local computer. A session is an instance of Windows PowerShell that contains all the commands that are available to you.

The Windows PowerShell session of your local computer, called the *client-side session*, has only the basic Windows PowerShell commands available to it. By connecting to the cloud-based service, you connect to the Microsoft datacenter's server environment, called the *server-side session*. This contains the commands used in the cloud-based service.

Before you begin

Before you connect, make sure you have the correct version of Windows PowerShell and WinRM installed and configured on your computer. For more information, see Install and Configure Windows PowerShell.

Verify that the account you will use to make the connection is authorized to connect by using Windows PowerShell. For more information, see Control Users' Access to Windows Remote Management.

Connect Windows PowerShell on your local computer to the cloud-based service

---

1. Click **Start**, point to **All Programs**, click **Accessories**, click **Windows PowerShell**, and then click **Windows PowerShell**.
2. Run the following command:

```
$LiveCred = Get-Credential
```

3. In the **Windows PowerShell Credential Request** window, type the credentials of an account in your cloud-based organization. Then, click **OK**.
4. Run the following command:

```
$Session = New-PSSession -ConfigurationName Microsoft.Exchange -C
onnectionUri https://ps.outlook.com/powershell/ -Credential $Live
Cred -Authentication Basic -AllowRedirection
```

Note:
The *AllowRedirection* parameter enables cloud-based organizations in datacenters all over the world to connect Windows PowerShell to the cloud-based service by using the same URL.

5. Run the following command:

```
Import-PSSession $Session
```

Commands that are used in the cloud-based service will now be imported into the client-side session of your local computer, as tracked by a progress bar. When this process is complete, you can run these commands.

Disconnect Windows PowerShell from the cloud-based service

When you're finished using the server-side session, always disconnect Windows PowerShell by running the following command:

```
Remove-PSSession
```

For example, to disconnect from the server-side session that is defined by the *$Session* variable, run the following command:

```
Remove-PSSession $Session
```

Important If you close the Windows PowerShell window without disconnecting from the server-side session, your connection will remain open for 15 minutes. Your account can have only three connections to the server-side session at one time.

STEP 3

Give Users Send As Permission

Applies to: *Office 365 for professionals and small businesses, Office 365 for enterprises, Live@edu*

Topic Last Modified: *2011-03-19*

Send As permission, also known as SendAs permission, gives a user permission to use another recipient's e-mail address in the From address. For example, when you give the user Chris Send As permission on the mailbox of a user named Michelle, Chris can send e-mail messages that appear to be sent by Michelle, with no indication to the recipient that anyone other than Michelle sent the message. Or, if your organization uses a Help Desk distribution group, you can give Help Desk staff Send As permission on the Help Desk distribution group. That way, replies to messages sent to the Help Desk group appear to come from the group instead of an individual Help Desk technician.

To give a user Send As permission, you use Windows PowerShell.

Before you begin

- To learn how to install and configure Windows PowerShell and connect to the service, see Use Windows PowerShell in Exchange Online.
- The Send As permission is different than the Send on Behalf permission. If the user Chris has Send on Behalf permission on Michelle's mailbox, when Chris sends an e-mail as Michelle, the From address shows Chris on behalf of Michelle. Microsoft Outlook users can configure Send on Behalf permissions on their own mailbox using delegates. Administrators can configure Send on Behalf permissions on any recipient type using the *GrantSendOnBehalfTo* parameter.
- Want more information about parameters? See An explanation of parameters.

Give a user Send As permission

Run the following command:

```
Add-RecipientPermission  -AccessRights SendAs -Trustee
```

For example, to give the user named Ayla Kol Send As permission for the Help Desk mailbox , run the following command:

```
Add-RecipientPermission "Help Desk" -AccessRights SendAs -Trustee "Ayl
a Kol"
```

Ayla can now send messages that appear to come directly from the Help Desk mailbox.

Note By default, you are asked to confirm the addition of the Send As permission. To skip the confirmation prompt, use -Confirm:$false.

View Send As permissions

Use the **Get-RecipientPermission** cmdlet to display all the Send As permissions configured in your organization. You can filter the list to show Send As permissions granted to a specific user and to see the Send As permission on a specific recipient.

View Send As permission for a specific user

Run the following command:

```
Get-RecipientPermission - Trustee
```

For example, to list the recipients for whom the user named Kim Akers has Send As permission, run the following command:

```
Get-RecipientPermission -Trustee "Kim Akers"
```

Kim can send messages that appear to come directly from the recipients.

View Send As permission on a specific recipient

Run the following command:

```
Get-RecipientPermission
```

For example, to list the users who have Send As permission on the Help Desk mailbox, run the following command:

```
Get-RecipientPermission "Help Desk"
```

The users listed can send messages that appear to come directly from the Help Desk mailbox.

View all Send As permissions you've configured in your organization

---

Run the following command:

```
Get-RecipientPermission | where {($_.Trustee -ne 'nt authority\self')
-and ($_.Trustee -ne 'null sid')}
```

Note The filter hides the automatic Send As permission that allows a user to send messages from their own mailbox, and also any results from system objects like mailbox plans.

Revoke Send As permission

Run the following command:

```
Remove-RecipientPermission  -AccessRights SendAs -Trustee
```

For example, to revoke Ayla Kol's Send As permission for the Help Desk mailbox, run the following command:

```
Remove-RecipientPermission "Help Desk" -AccessRights SendAs -Trustee "
Ayla Kol"
```

Now Ayla can't send messages that appear to come directly from the Help Desk mailbox.

To skip the confirmation prompt, use -Confirm:$false.

How people use the Send As permission

Individual users or members of security groups with Send As permission can open their own mailboxes and send messages using the From address of the recipient.

Send As permission doesn't give a user access to another user's mailbox. To give an individual or members of a security group access to a mailbox, use the following command:

```
Add-MailboxPermission  -User  -AccessRights FullAccess
```

When you give someone access to a mailbox and Send As permission on the mailbox, that person can open the mailbox using their own credentials, compose new messages, and reply to messages in the mailbox.

An explanation of parameters

You use the **Add-RecipientPermission**, **Remove-RecipientPermission**, and **Get-RecipientPermission** cmdlets to add, remove and view Send As permissions. These cmdlets use the same basic parameters:

- **Identity** This parameter specifies the target recipient. The user or group specified by the *Trustee* parameter can operate on this recipient.
  You can specify any type of recipient. For example:
    - Mailboxes
    - Mail users
    - External contacts
    - Distribution groups
    - Dynamic distribution groups

  The *Identity* parameter is a positional parameter. The first argument on a cmdlet is assumed to be the *Identity* parameter when no parameter label is specified. This lets you specify the parameter's value without specifying the parameter's name. For example, instead of typing, Get-RecipientPermission -Identity "Kim Akers" you can type, Get-RecipientPermission "Kim Akers".
- **Trustee** This parameter specifies the user or group to whom you're granting the permission. This allows the user or group to operate on the recipient specified by the *Identity* parameter.
  You can specify the following types of users or groups:
    - Mailbox users
    - Mail users with a user account
    - Security groups

For the *Identity* and *Trustee* parameters, you can use any value that uniquely identifies the recipient.

For example:

- Alias
- Distinguished name (DN)
- GUID
- Name
- Display name
- LegacyExchangeDN
- E-mail address