Defending Against Rules and Forms Injection

By kexugit

Archived: 2026-04-06 03:15:49 UTC

Over the last year, Office 365 security has been tracking an emergent attacker persistence mechanism in the Exchange Online ecosystem. The release of a security research tool called Ruler enables an attacker to install a persistence mechanism once an account has been breached to maintain access even through a password roll. While we haven't seen widespread use of the tool in Office 365, we wanted to equip IT Security Administrators with a robust understanding of the threat, a model for how to prevent the exploit, and some lightweight tooling to detect, monitor, and remediate the threat. The good news here is that if you keep your clients patched to the latest version, you are not vulnerable to the threat as Outlook client defaults block both mechanisms. You should still perform the actions in the 'How to Detect It' section, however, in case an account was breached prior to the patch releases last year.

The Threat

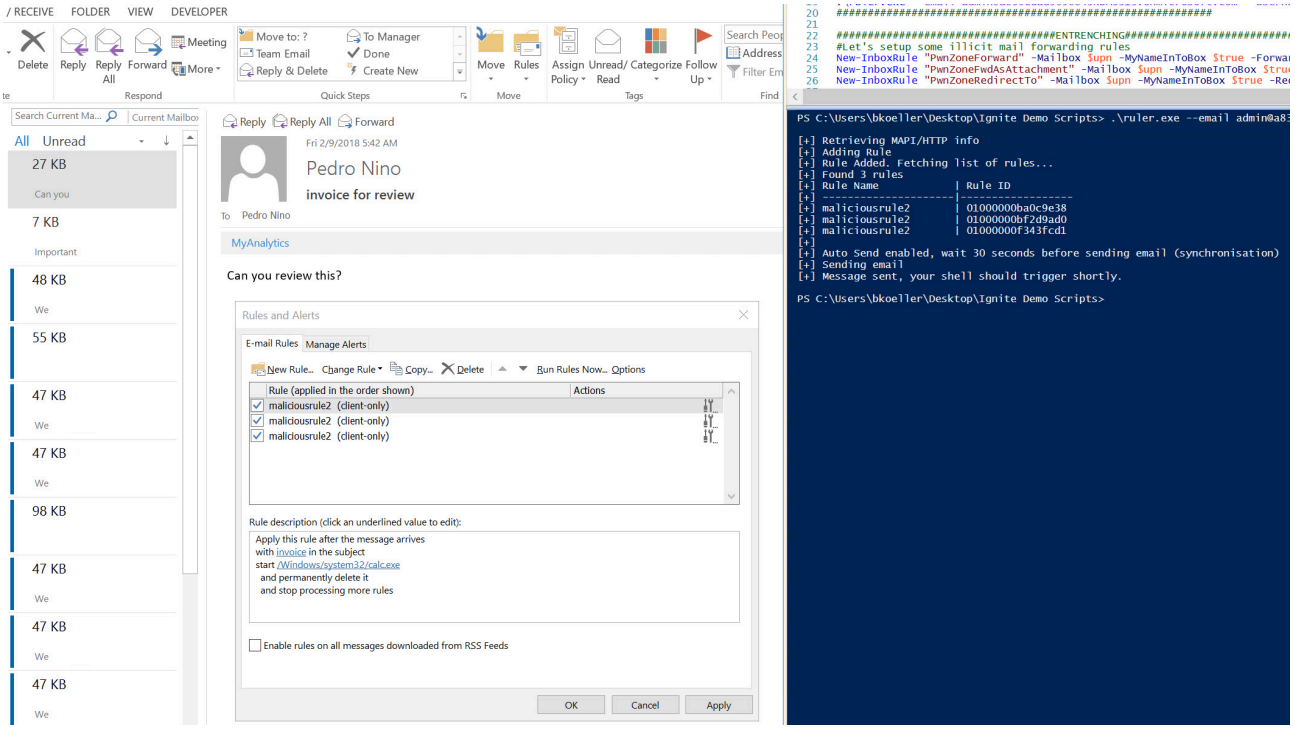
There are two variants of the persistence mechanism: mailbox rules and custom mail forms. In both cases, the rule/form is synced from the cloud service down to the desktop client, so a full format and re-install of the client software doesn't eliminate the injection mechanism as resyncing the mailbox will re-download the rules/forms from the cloud. The rules/forms, once placed, are used to execute remote or custom code, usually to install malware on the local machine, which is in turn leveraged to re-steal credentials or perform other illicit activity. You can read an in-depth description of the exploits here:

- SilentBreak Security Post about Rules Vector: https://silentbreaksecurity.com/malicious-outlook-rules/
- Sensepost Blog about Mailrule Pwnage: https://sensepost.com/blog/2016/mapi-over-http-and-mailrule-pwnage/
- Sensepost Blog about Forms Threat Vector: https://sensepost.com/blog/2017/outlook-forms-and-shells/
- Ruler Codebase: https://github.com/sensepost/ruler

The exploits exhibit the following characteristics:

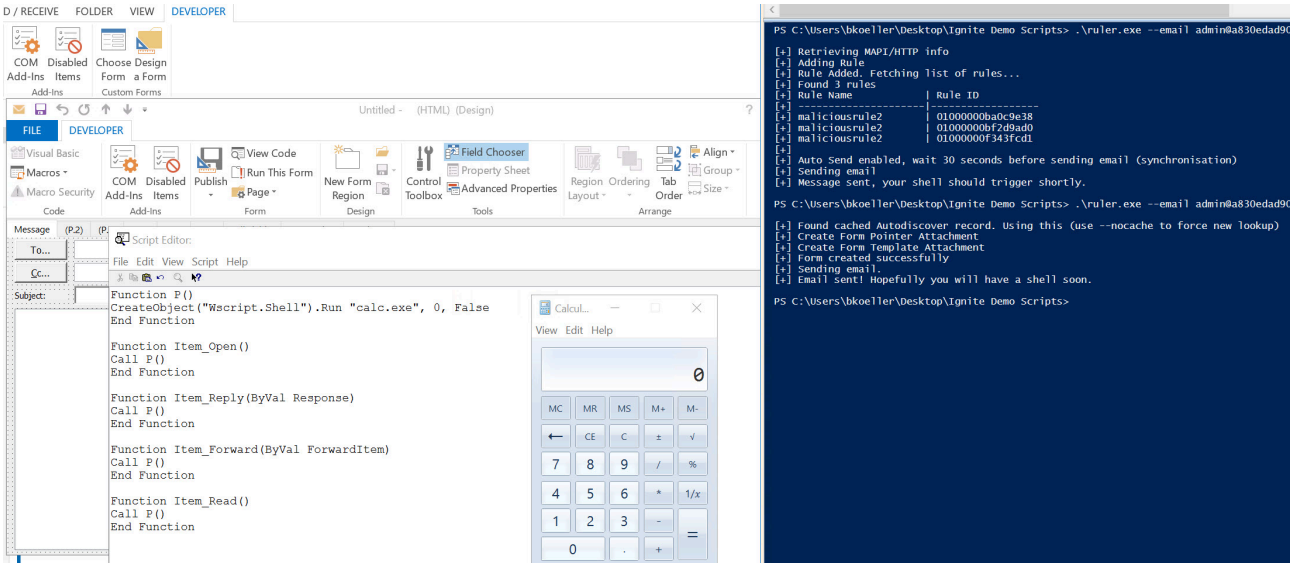
1. A user's credential is compromised and an attacker logs on to their Exchange account (Exchange Online, or On-Premise)
2. The attacker creates a forwarding rule in the target's mailbox...
3. That when triggered with an email that matches the rule criteria...
4. Will execute an application...
5. Usually hosted on a remote WebDav server...
6. That installs malware locally such as Powershell Empire
7. Which allows attacker to dump creds from local machine and perform other malicious activities

This is what the exploit looks like:



1. A user's credential is compromised and an attacker logs on to their Exchange account (Exchange Online, or On-Premise)
2. The attacker creates a custom mail form template and injects it into the target's mailbox...
3. That when triggered with an email that matches the custom form template type...
4. Will execute custom code in the template...
5. That usually installs malware locally such as [Powershell Empire]
6. Which allows attacker to dump creds from local machine and perform other malicious activities

This is what the exploit looks like:



How to Prevent It

The Ruler exploit described here is considered a post-exploit technique, which means the account has already been breached by the attacker when this exploit is executed. Taking aggressive steps to prevent the initial breach in the first place is advised. There are many account breach vectors, including phishing and password spraying, which are relatively easy to execute. Using protection mechanisms like multi-factor authentication on user accounts, whether conditional or always-on, is an effective mitigation. This should be your first line of defense. Some effective tactics you can leverage include:

1. Enable Multi-factor Authentication for all of your users, especially those with administrative privileges.
2. Monitor your accounts for illicit activity. You can do this manually, or with a security monitoring toolset. This may not prevent the initial breach, but will shorten the duration, and the impact of the breach.
3. Leverage a tool like the Office 365 Secure Score (https://securescore.office.com) to manage account security configurations and behaviors.

The second key prevention mechanism is to use the fully-updated latest versions of Outlook (2013/2016) and ensuring that rule actions that can start applications/macros are disabled per the security patches. This will ensure that, even if an attacker breaches the account, the rule and form execution mechanisms will be blocked.

Patched versions are 15.0.4937.1000 or greater (Outlook 2013) and 16.0.4534.1001 or greater (Outlook 2016). Details about the patches are here:

- Outlook 2013 Security Patch - https://support.microsoft.com/en-us/help/3191938
- Outlook 2016 Security Patch - https://support.microsoft.com/en-us/help/3191883

Customers with on-premises Exchange installations should consider blocking older versions of Outlook that do not have patches available. Details on this process can be found in the article “Configure Outlook client blocking”.

Please note that the rule mitigation is to disable the "Start Application" rule action by default. Even with the patch, it is possible for an attacker to change the local machine configuration to re-enable this. Make sure you monitor and enforce local machine policies on your clients. If you do happen to have an important business process that relies on this functionality, you can enable it for your users, but you will need to monitor your deployed user roles as per below with extra vigilance.

To determine if “Start application” is enabled via override in the registry, check for these subkeys:

- Outlook 2016: HKEY_CURRENT_USER\Software\Microsoft\Office\16.0\Outlook\Security\
- Outlook 2013: HKEY_CURRENT_USER\Software\Microsoft\Office\15.0\Outlook\Security\

If the key EnableUnsafeClientMailRules exists with a value of 1, the patch is overridden and allows the action above. If the value is 0, the rule actions are disabled. If a current (fixed) version of Outlook is installed and this registry key is NOT present or present and set to 0, then a system is not susceptible to the mail rule injection exploit.

The custom mail form injection exploit relies on custom code being injected into a very rarely used custom mail form feature and then executed outside the normal office macro sandbox. New versions of Outlook client prevent this code execution.

How to Detect It

There are several methods you can leverage to detect and monitor for this exploit in your organization.

Using Powershell

The simplest method is to leverage a Powershell script we have created which allows you to dump all of the mail forwarding rules and custom forms for all the users in your tenancy: https://github.com/OfficeDev/O365-InvestigationTooling/blob/master/Get-AllTenantRulesAndForms.ps1 . You will need to have a global admin role elevation to run the script (because it connects to every mailbox in the tenancy to read the rules and forms). It takes no parameters. The script is written for an Office 365 organization, so if you are using Exchange on-prem, you'll need to modify it. The script will produce two time-stamped CSV files: one for rules and one for forms. The rules one should be examined for action conditions that include applications or executables. Two columns in the MailboxRulesExport_DATE.csv file will help focus your search.

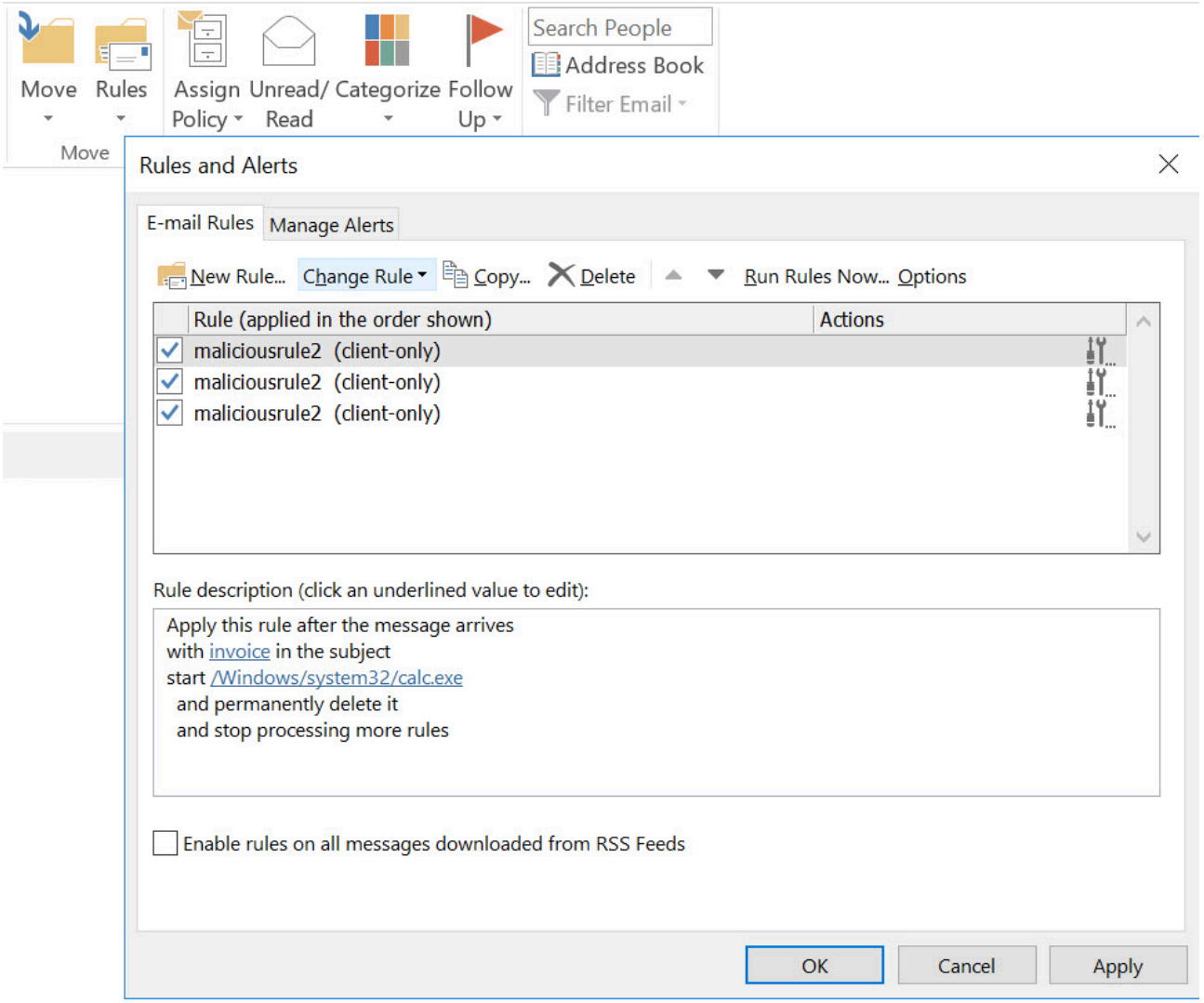
- This assesses the rule to see if there is a custom action, if the action is to execute an application, or if the action is run a macro. Any of these conditions will flag the rule as being potentially malicious.
- This field contains the actual execution string if the rule executes an action. If it calls a remote server, or a suspiciously named application, it is likely malicious.

Using Security Researcher Toolkit

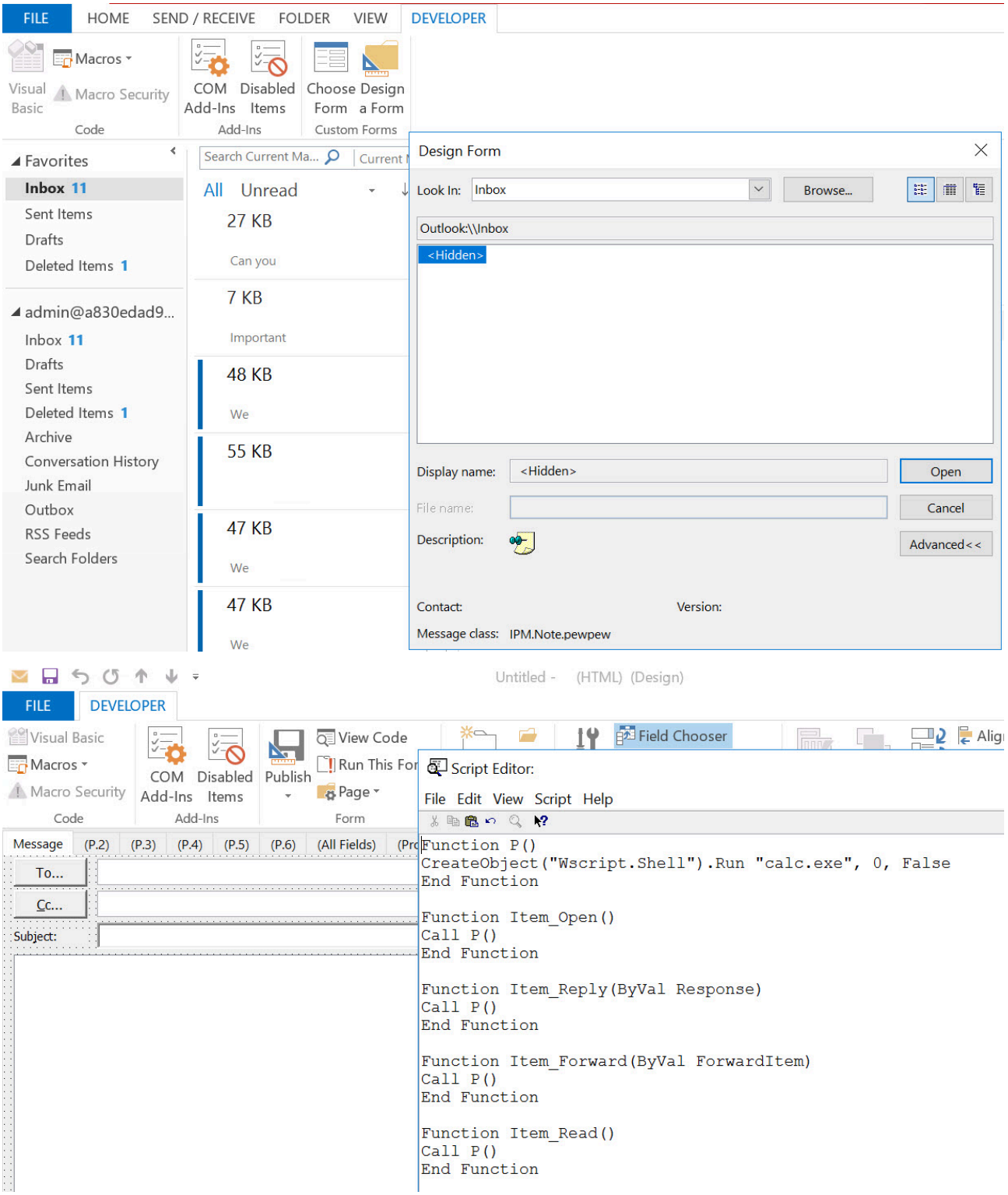
Another method you can leverage is to use the defensive version of the Ruler toolkit called NotRuler, authored by the same researcher: https://github.com/sensepost/notruler. The author has helpfully included a list of IOCs for the Ruler toolkit: https://github.com/sensepost/notruler/blob/master/iocs.md.

Using Outlook or MFCMAPI

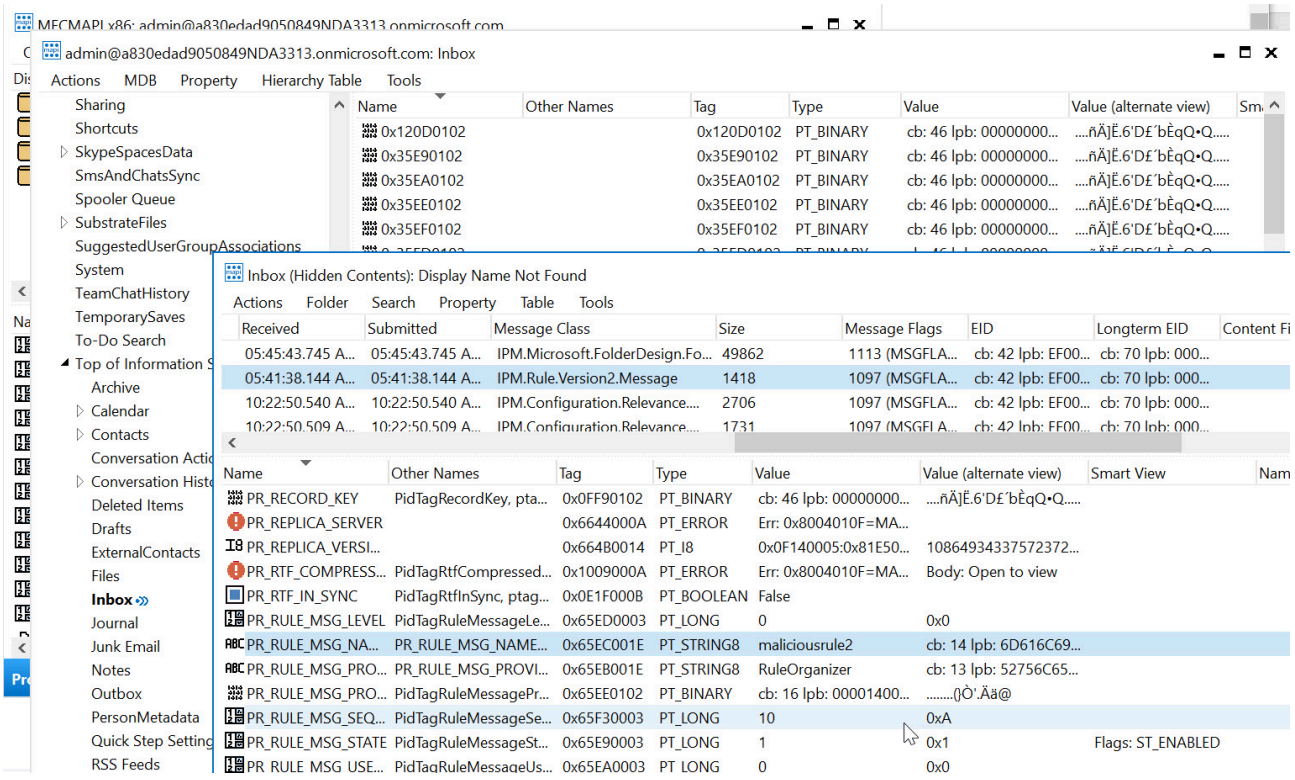
If you are willing to do substantial spelunking one mailbox at a time, you can also leverage the Outlook client or access the mailbox database directly by using the MFCMAPI tool. While this is time consuming and can result in a breach of the machine you are using to investigate the suspected breached mailbox, it is the most straightforward and complete enumeration of the exploit in context. Use the rule management interface form outlook and examine each rule description:



Or, enable the developer tab in the Outlook client, click *design a form*, and look in the user's Inbox for custom forms. The Ruler tool will, by default, mark the custom form as hidden. Once you open the form, click "View Code" to see what the custom form actually runs:



You can also MFCMAPI to connect to the suspected breached mailbox, find the Inbox Folder, right click and open associated contents table, the look for Message Class IPM.Rule.Version2.Message (for malicious rules) and IPM.Microsoft.FolderDesign.FormsDescription (for malicious forms). Custom forms are rare enough that if you have *any* custom forms, it is worth a deeper look.



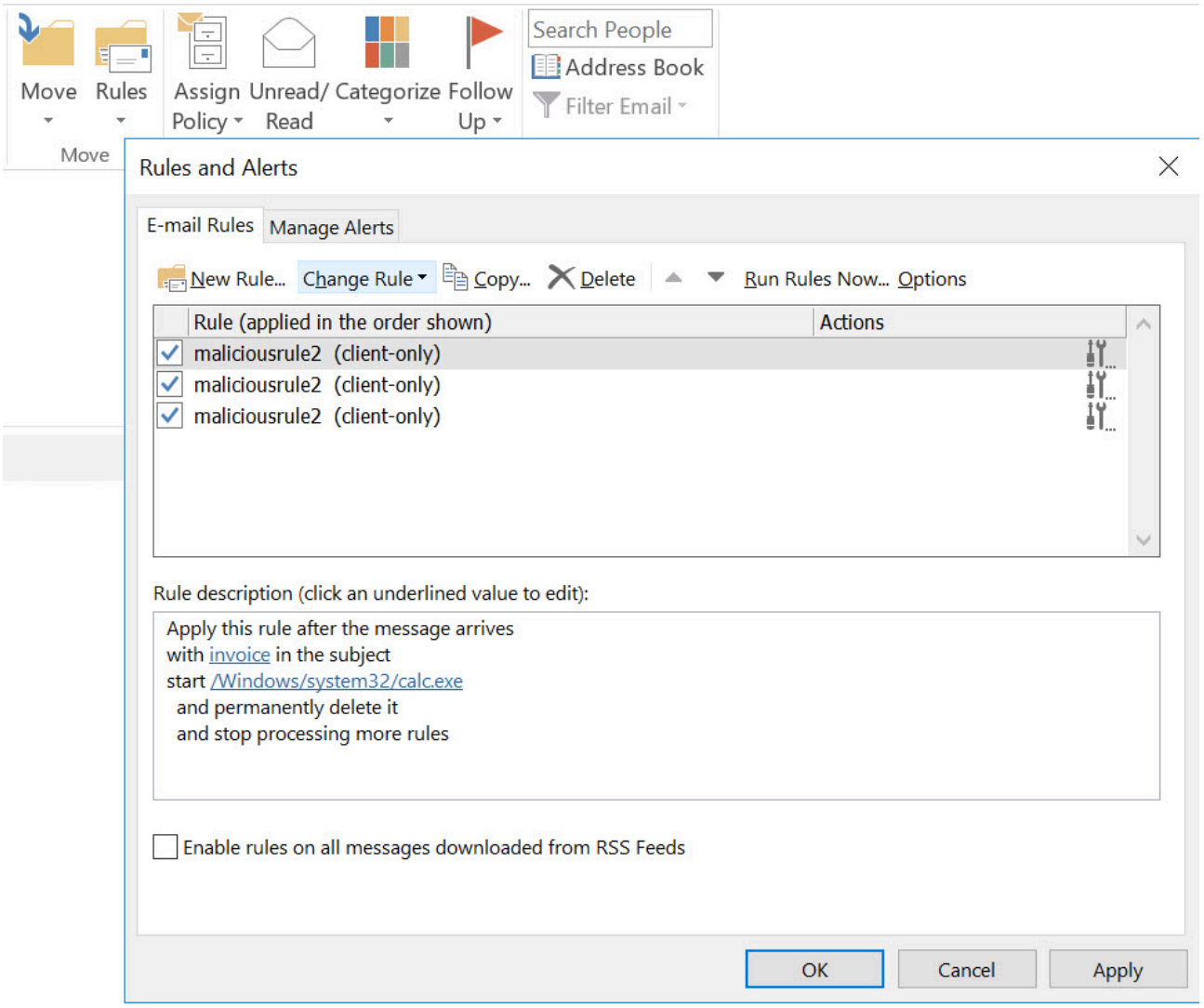
Broadly speaking, you are looking for the following indicators of compromise:

- Indicators of compromise for rules
  - Rule Action is to start an application
  - Rule References an EXE, ZIP, or URL
  - On the local machine, look for new process starts that originate from the Outlook PID
- Indicators of compromise for forms
  - Custom form present saved as their own message class
  - Message class contains executable code
  - Usually stored in Personal Forms Library or Inbox folders
  - Form is named IPM.Note.[custom name]

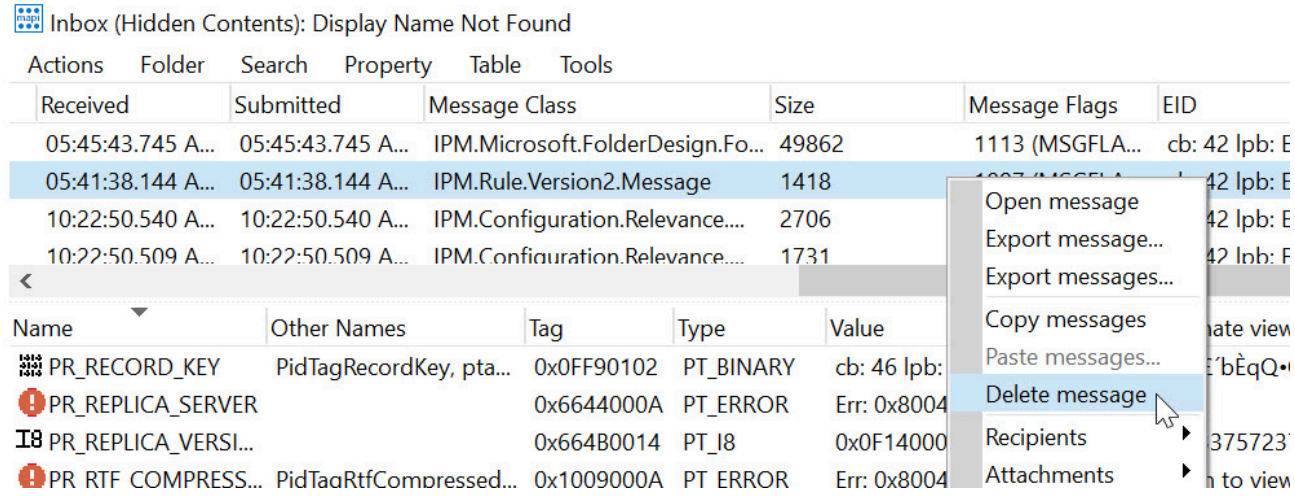
If you discover that you have been compromised with this vector, remediation of the actual exploit is straightforward: delete the rule or form from the mailbox. You can do this with the Outlook client, MFCMapi, or using remote PowerShell to remove rules. A full remediation, however, is more in depth and looks like:

1. Remove the rule or form from the user's mailbox using Outlook, remote Powershell, or MFCMapi.
2. Do not allow the user to logon and use email until the full remediation process is complete.
3. Every machine that the user has used with Outlook will need to be cleaned of potential malware. The simplest way to accomplish this is to format and re-install everything on the machine, but if you can ensure that all malware has been removed with high confidence that might work.
4. Ensure that you install the most up-to-date versions of Outlook so that the vector is closed.
5. Once all offline copies of the mailbox have been removed, reset the user's password (ensuring it has a high-quality one set) and enable multi-factor authentication on the account to ensure the user's credentials are not exposed via other means (such as phishing or password re-use).

Deleting with Outlook



Deleting with MFCMapi



There are two remote powershell cmdlets you can use to remove or disable dangerous rules:

- Remove-InboxRule: https://technet.microsoft.com/en-us/library/dd351272(v=exchg.160).aspx

- Disable-InboxRule: https://technet.microsoft.com/en-us/library/dd298120(v=exchg.160).aspx (use if you need to retain the rule contents for further investigation)

Ruler Codebase: https://github.com/sensepost/ruler

SilentBreak Security Post about Rules Vector: https://silentbreaksecurity.com/malicious-outlook-rules/

Sensepost Blog about Mailrule Pwnage: https://sensepost.com/blog/2016/mapi-over-http-and-mailrule-pwnage/

Sensepost Blog about Forms Threat Vector: https://sensepost.com/blog/2017/outlook-forms-and-shells/

Get-AllTenantRulesAndForms.ps1: https://github.com/OfficeDev/O365-InvestigationTooling/blob/master/Get-AllTenantRulesAndForms.ps1

MFCMAPI: https://github.com/stephenegriffin/mfcmapi

Outlook 2013 Security Patch - https://support.microsoft.com/en-us/help/3191938

Outlook 2016 Security Patch - https://support.microsoft.com/en-us/help/3191883
