CYBERARK-TENABLE INTEGRATION

Name of Company: Tenable

Website: www.tenable.com

Name of Product: Tenable.io

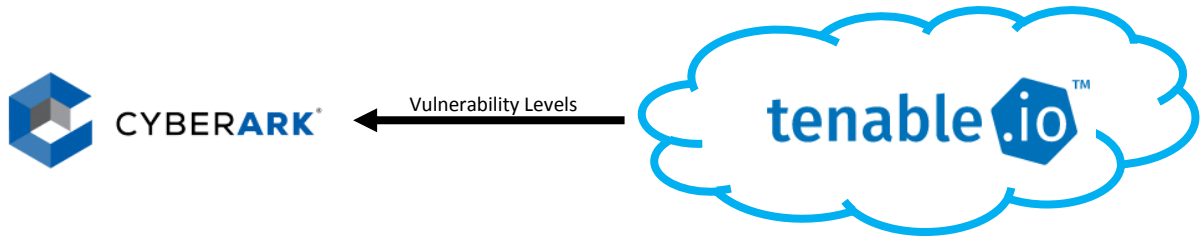
Date: June 2019

KEY BENEFITS

Armed with Tenable.io, a Cyber Defender can quarantine a server deemed vulnerable by Tenable. When a CyberArk user tries to access a privileged account in a vulnerable server, the CyberArk Privileged Account Security Solution can immediately implement higher security controls around that server.

This integration between Tenable.io and CyberArk helps mitigate the risk of a potential breach by preventing privileged accounts, on vulnerable servers, from being accessed.

PRODUCT DIAGRAM & DESCRIPTION OF PRODUCT INTEGRATION



The integration between Tenable.io and CyberArk was developed around the CyberArk's PVWA Ticketing Integration framework (which provides a powerful user-exit capability) used in this particular case, to validate whether access (Show, Copy, or Connect) to a server's privileged account should be granted or not, due to vulnerabilities exposed by the information queried against Tenable.io.

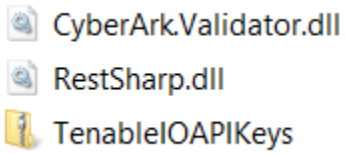
The distribution zip contains two DLL files (**CyberArk.Validator.dll** & **RestSharp.dll**) and the **TenableIOAPIKeys.zip** file. The interaction with Tenable.io is governed by a specific set of Ticketing Parameters described in the Configuration section of this document.

The integration implements a hierarchy: *Vulnerability Manager -> Vulnerability Subset -> Conditions*

These are defined as Ticketing System parameters defined in the PVWA Administration page. You can specify as many parameters as required. The parameters can be specified in different hierarchies and levels and are transferred to the integration module in an xml object when a user clicks on Show, Copy, or Connect. Based on the logic defined by these parameters, the integration module determines if the user will be denied access to a server with vulnerabilities. See Appendix 1 for an example of the XML object.

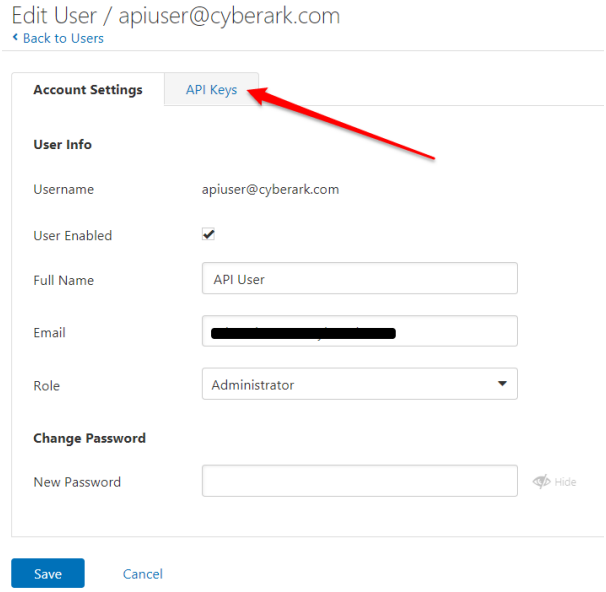
INSTALLATION

Extract the Tenable.IO-Dist distribution into a clean folder. It contains the following files:

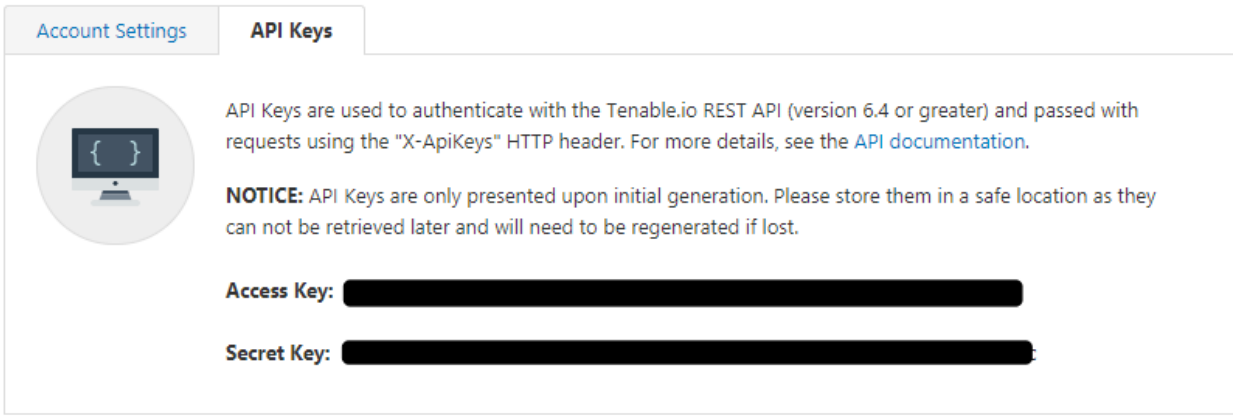


CREATE THE TENABLE API USER

Login to Tenable.io and create a user for the integration:

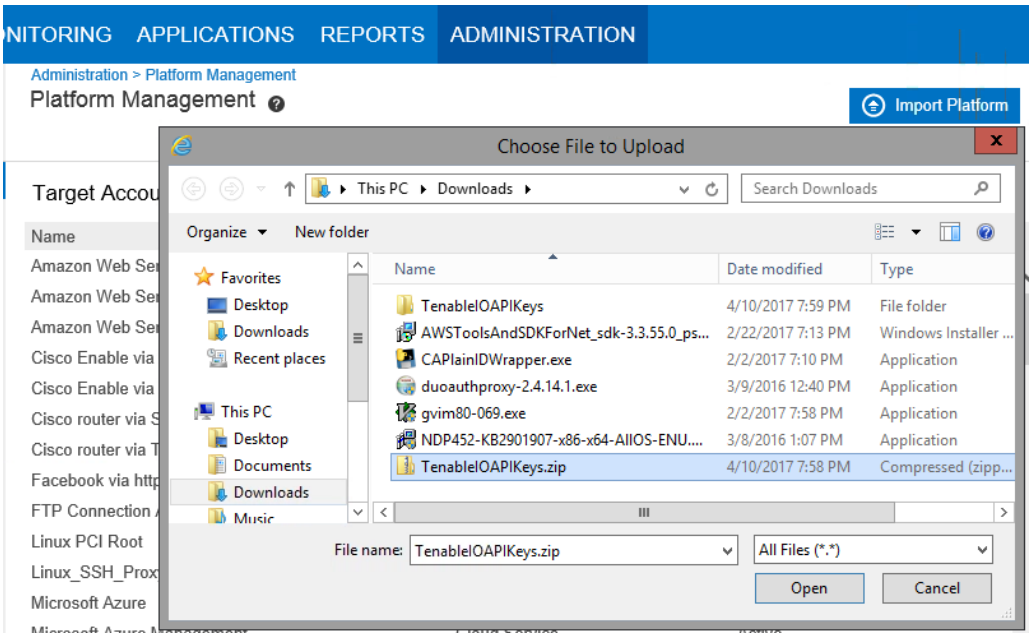


Click on API Keys to generate the keys that will allow the integration to login as this user through the Tenable REST API. You'll need them on the next step:

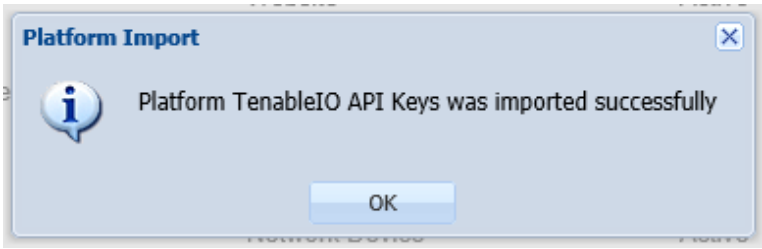


IMPORT THE TENABLE.IO API KEYS PLATFORM

The API Keys required to query the Tenable.io platform need to be stored inside CyberArk. For that you need to import the Tenable.IO API Keys platform that came in the distribution file. Login to the PVWA and click on **ADMINISTRATION** -> **Platform Management**. Click on **Import Platform**, select the **TenableIOAPIKeys.zip** file from the distribution, and click **Open**:



You should see an upload progress indicator followed by the following message:

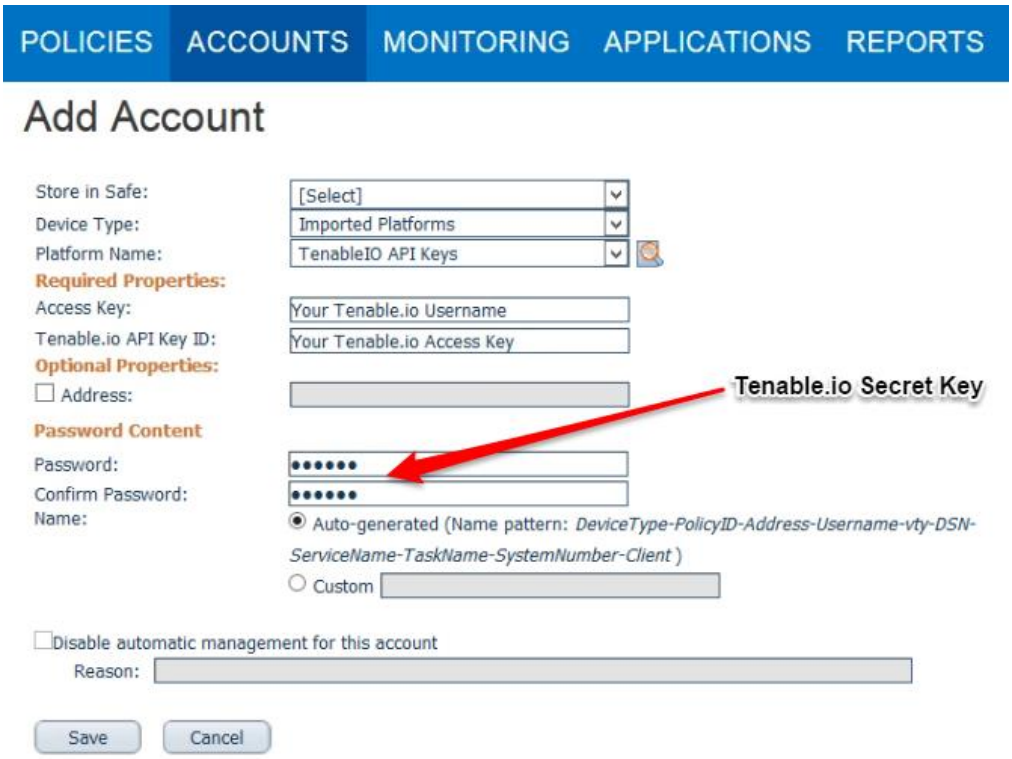


CREATE THE API ACCOUNT ON THE PVWA

Logon to the PVWA and go to **Accounts** -> **Add Account**.

Select a Safe, the **Device Type** must be **Imported Platforms**, **Platform Name** must be **TenableIO API Keys**.

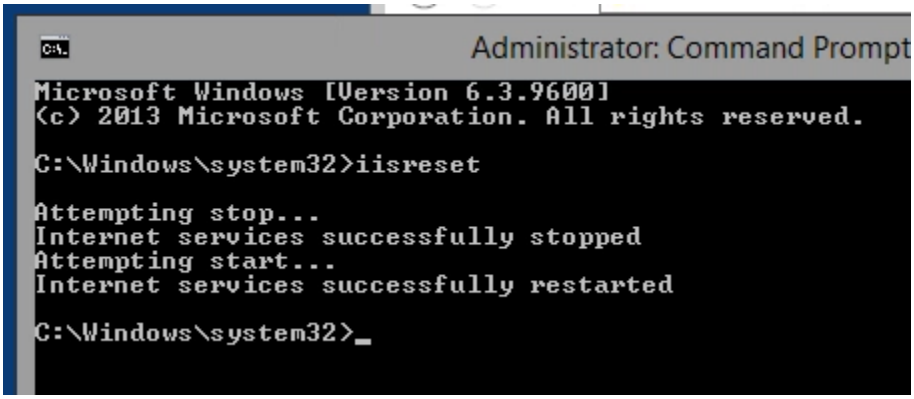
Enter your Tenable.io **Username** on the **Access Key** field, your **Access Key** in **Tenable.io API Key ID** field, and in the **Password** field enter you Tenable **Secret Key**:



Click on **Save** when done.

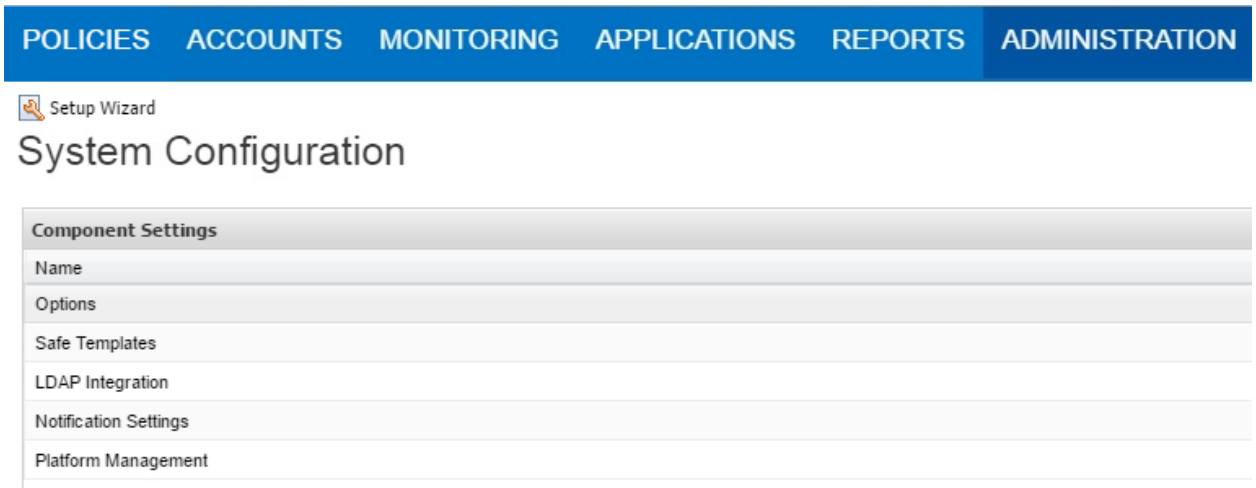
INSTALL THE CODE

Copy the two DLL's on the distribution to **%inetpub%\wwwroot\PasswordVault\Bin** on the PVWA server, then restart IIS:

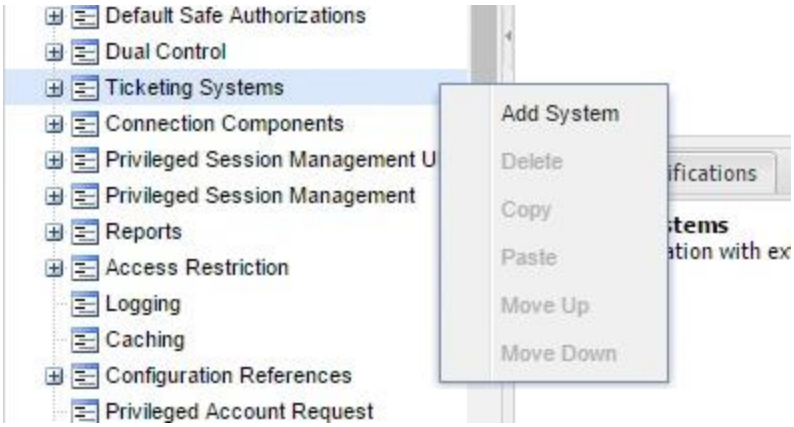


CONFIGURATION

Log-in to the PVWA as a vault administrator user, then go to **ADMINISTRATION** and click on **Options**:



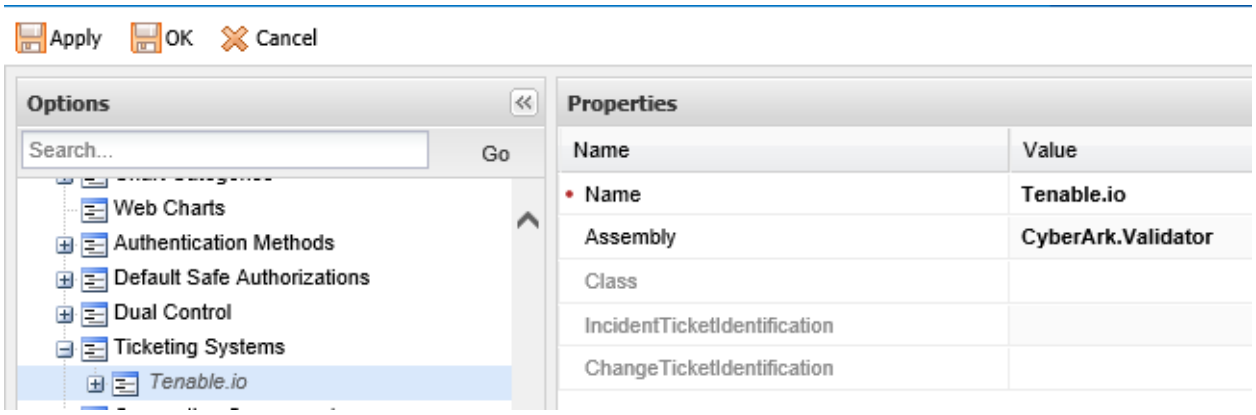
Right click on **Ticketing Systems** and select **Add System**:


Default Safe Authorizations
Dual Control
Ticketing Systems
Connection Components
Privileged Session Management U
Privileged Session Management
Reports
Access Restriction
Logging
Caching
Configuration References
Privileged Account Request
Add System
Delete
Copy
Paste
Move Up
Move Down

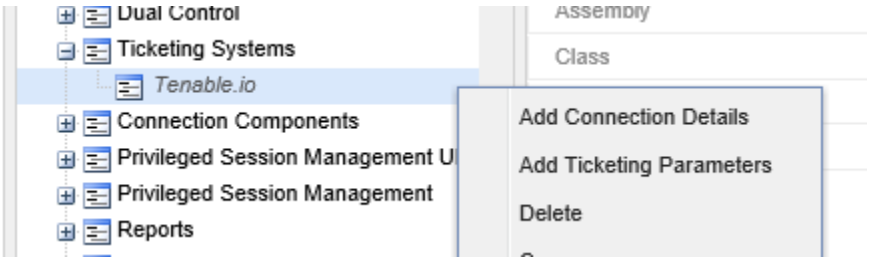
On the **Properties** section enter the following information:

Name: You can enter any value.

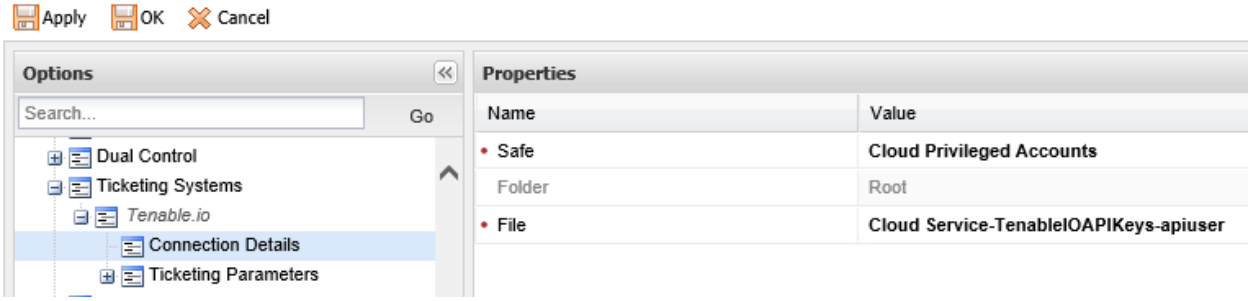
Assembly: Corresponds to the name of the dll file provided in the package. You *must* enter **CyberArk.Validator** as shown here:



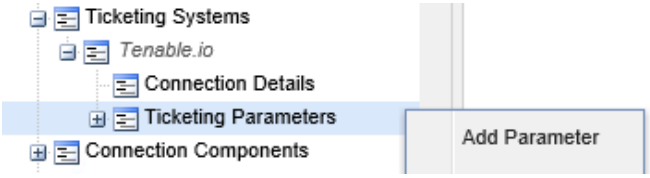
After clicking **Apply**, Tenable.io will appear under **Ticketing Systems**. Right click on it and select **Add Connection Details**:



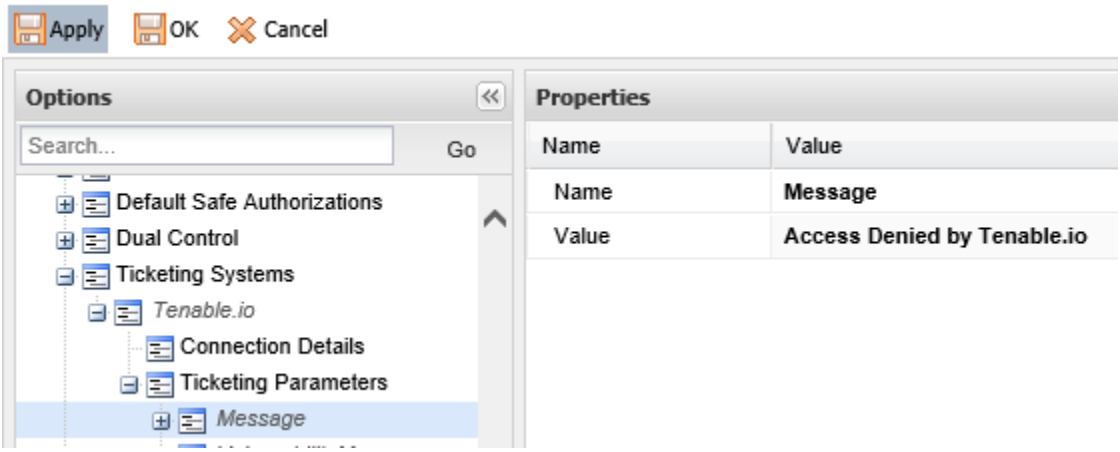
You will need to provide the safe, folder (root), and object name from the account you created on the previous step:



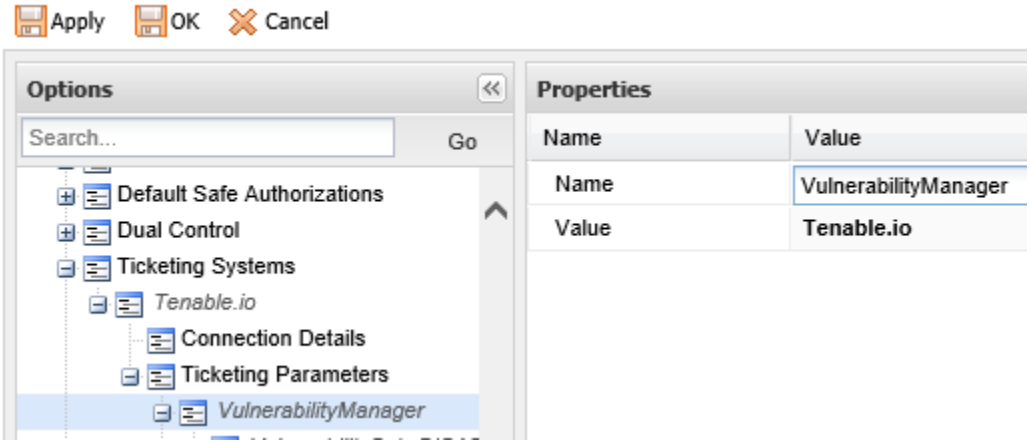
Click OK, then Right Click on Tenable.io and select **Add Ticketing Parameters**. A new **Ticketing Parameters** section will appear under Tenable.io, below Connection Details. Right click on it and select **Add Parameter**:



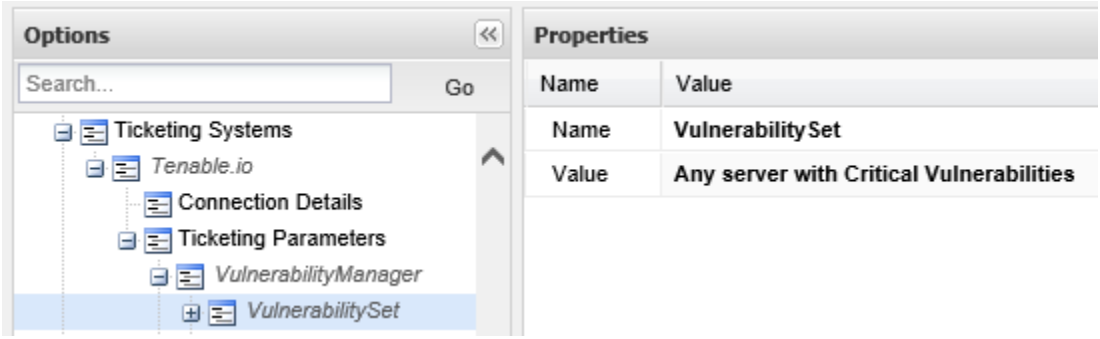
Name this Parameter **Message** and enter a generic message you want the PVWA to display when denying access to a vulnerable server. This message will apply to ALL Vulnerability Managers.



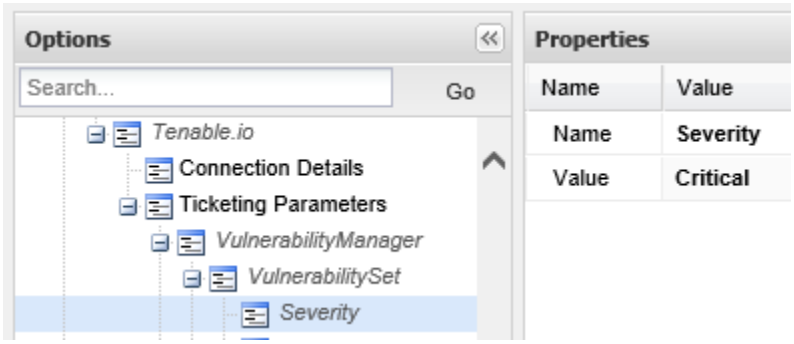
You will be able to define additional messages, which will follow the generic message, providing specific details, in the following subsections. Repeat the process to add a new parameter called **VulnerabilityManager**. The value of this parameter is not validated but you can use the value for reference purposes:



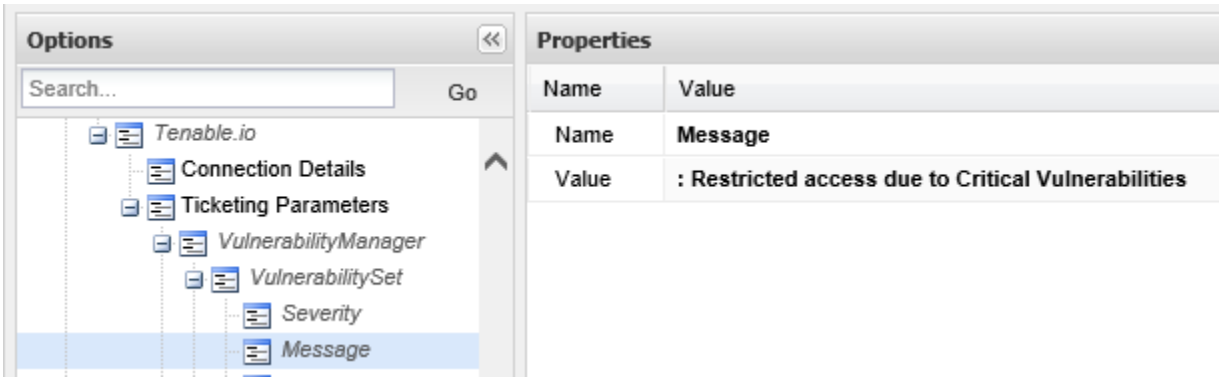
Next step is to add a **Vulnerability Set**. Right click on **VulnerabilityManager** and select **Add Parameter**. On Name enter **VulnerabilitySet**. Use Value for documentation purposes:



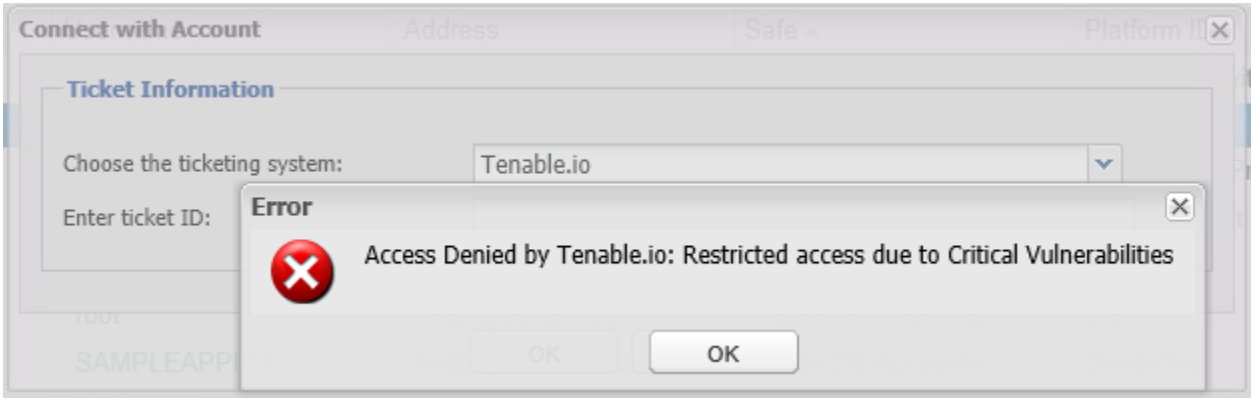
This integration is able to validate multiple properties from both Tenable and CyberArk. See Appendix 2 for the complete list. To restrict access to servers identified by Tenable to have a Severity Vulnerability 4, right click on **VulnerabilitySet** and select **Add Parameter**. Name it **Severity** and enter a Value of **Critical**.



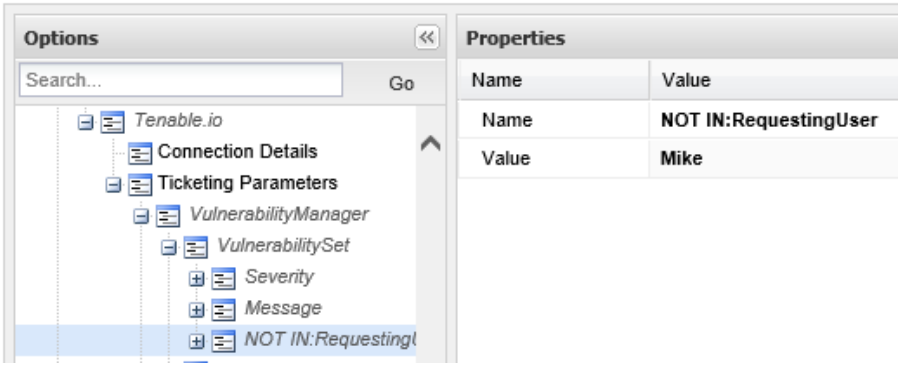
It's highly recommended that you also add a specific message, that will be appended to the generic message defined earlier, explaining the reason users are being denied access. Right click on **VulnerabilitySet** -> **Add Parameter** and named it **Message**. Enter **: Restricted access due to Critical Vulnerabilities** and save it:



When you try to access (show/copy/connect) to any account residing on a server that has a Tenable.io Severity 4 vulnerability you will get an error message:

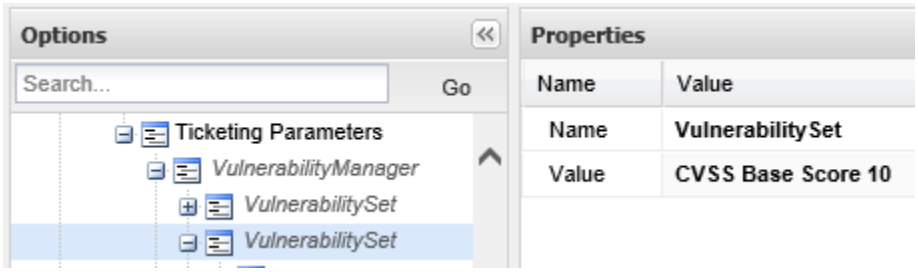


This completely locks all users from access to all servers with vulnerability 4. However this is not very useful because somebody will need to connect to fix the vulnerabilities. Let's amend the Vulnerability Set to allow a senior Sysadmin to connect and repair the vulnerabilities. Add another parameter to the Vulnerability Set called **NOT IN:RequestingUser** and assign a value of **Mike** and save it:

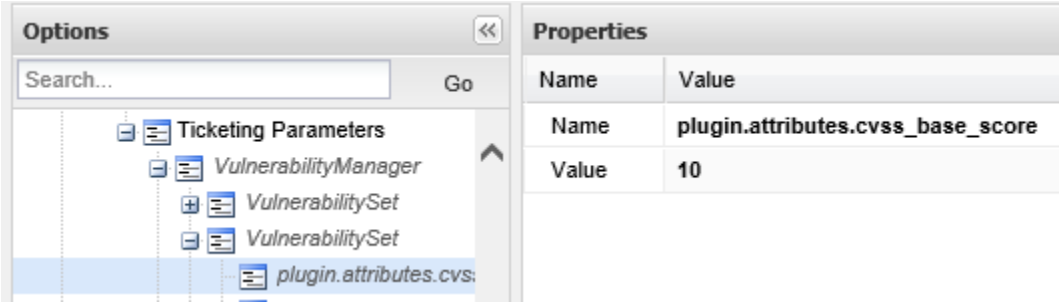


This will allow only user Mike to connect to these servers, so the vulnerabilities can be addressed. The NOT IN operator allows you to specify a list, in this case of users.

Let's restrict access to all OS accounts residing on servers classified by Tenable.io with a CVSS Base Score of 10, start by creating a new Vulnerability Set by right clicking on **Ticketing Parameters** and selecting **Add Parameter**. Under **Name** enter **VulnerabilitySet** and under **Value** a good label for documentation purposes:



Add a parameter to your newly created Vulnerability Set. Name it **plugin.attributes.cvss_base_score**. Enter 10 in **Value** and save it.

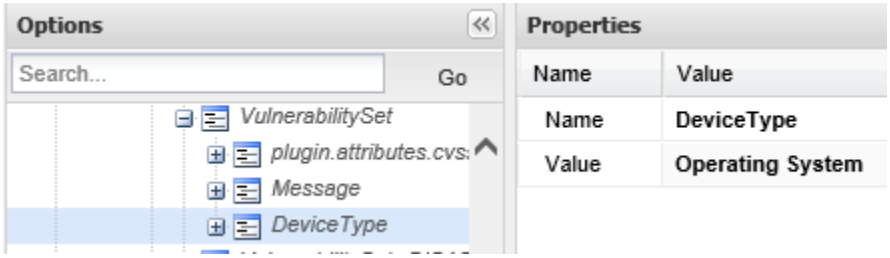


This will prevent access to servers that Tenable.io has identified to have a CVSS base score of 10. It's always a good idea to exclude a list of users who are capable or addressing the vulnerabilities. Add a new parameter called **NOT IN:RequestingUser** and enter a value of **Mike**. Finally let's add a customized error message that will identify both the Plugin ID under which access is been denied as well as the solution:

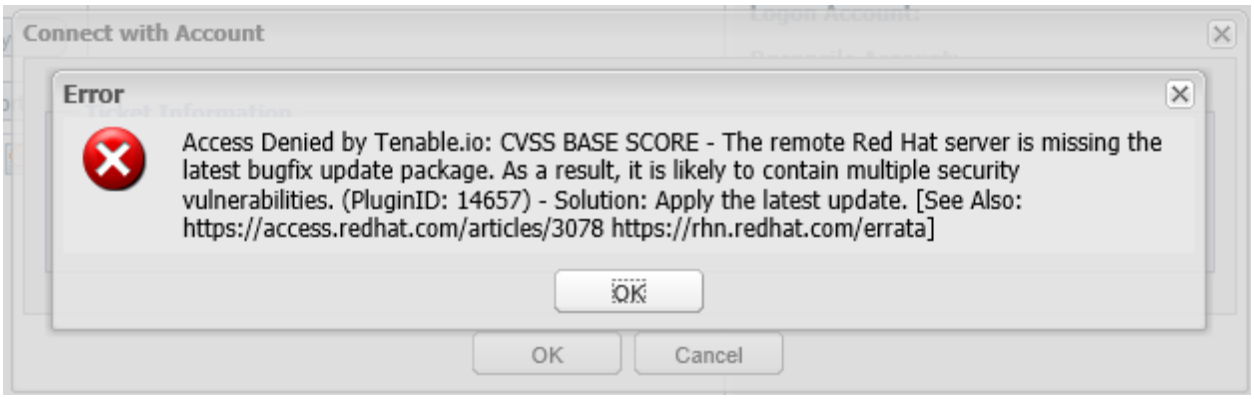
The value in the message field is:
: CVSS BASE SCORE - $vulnerability.description (PluginID: $vulnerability.plugin_id) - Solution: $vulnerability.solution [See Also: $vulnerability.see_also]

The full list of variables you can use is listed in Appendix 3.

Now add a new parameter called **DeviceType** and enter **Operating System** as its value

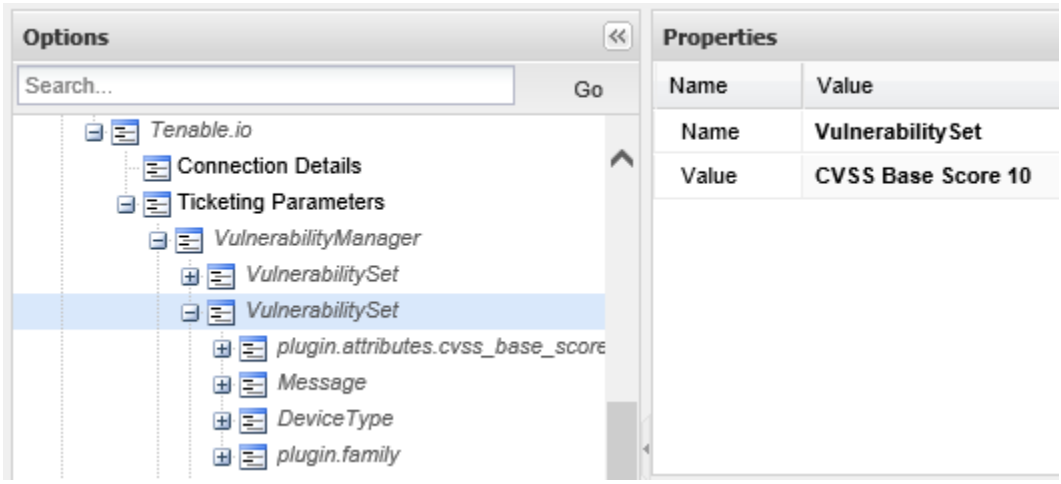


When a user, other than Mike, attempts to connect to a server on which Tenable.io has identified one of the plugins in the list, he/she will get the message:



Also notice that this will only apply to Operating Systems type of accounts, all other account types will be allowed access.

Finally keep in mind that Vulnerability Sets are evaluated as an OR, meaning when the first one is found to be true we stop evaluating the rest. The parameters and conditions inside a Vulnerability Set are evaluated as an AND, meaning that all of them must be true for the Vulnerability Set to deny access. Use this rule to properly segment conditions across Vulnerability Sets to manage access.



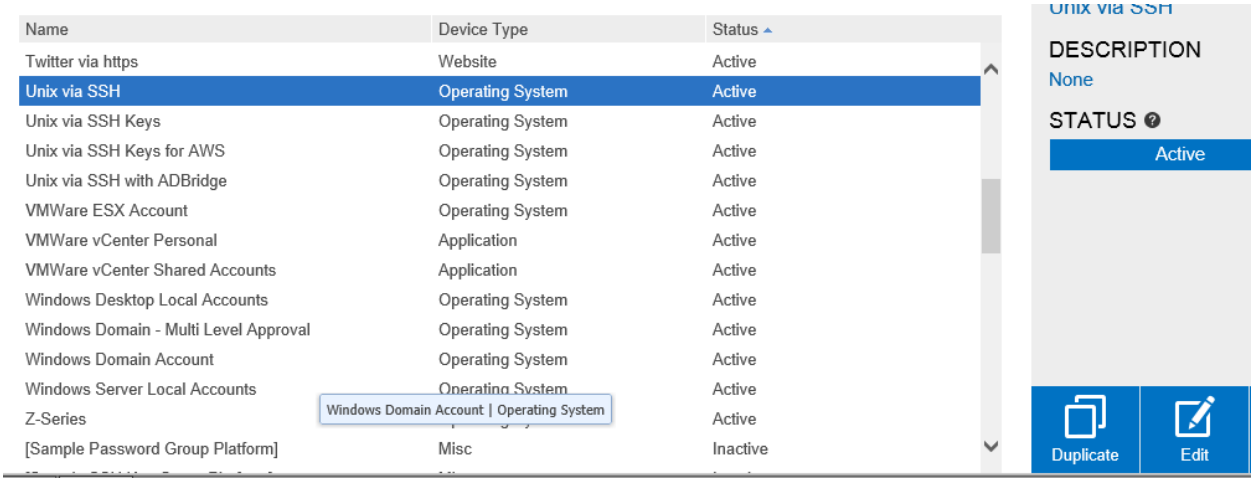
Properties	
Name	DeviceType
Value	Operating System

Properties	
Name	plugin.family
Value	Gain a shell remotely

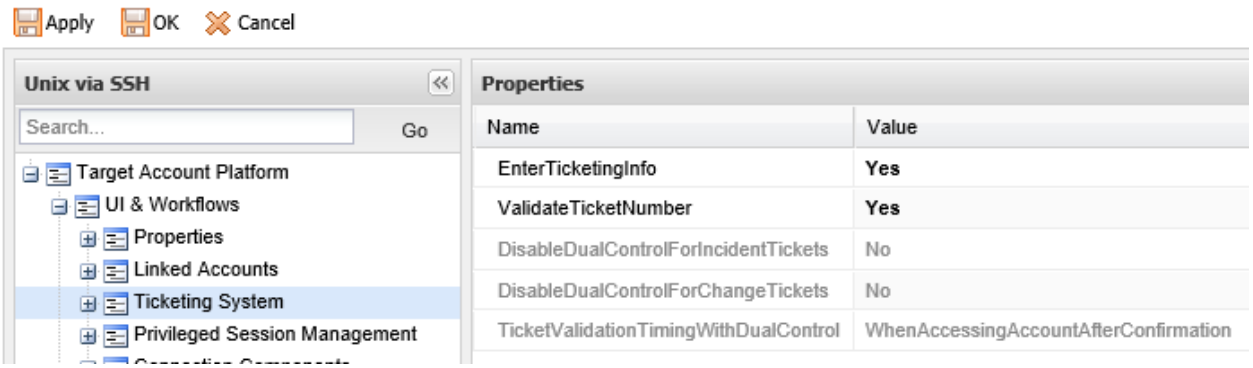
Properties	
Name	plugin.attributes.cvss_base_score
Value	10

ENABLING THE INTEGRATION FOR THE RELEVANT PLATFORMS

Enable the integration (Ticketing System) for platforms where you want to deny access to servers with specific vulnerabilities identified by Tenable.io. Log-in to the PVWA as a vault administrator user, go to **ADMINISTRATION** and click on **Platform Management**. Select your platform and click on **Edit**:



If you do not already have an active Ticketing System, then expand **UI & Workflows**, click on **Ticketing System**, and set both **EnterTicketinginfo** and **ValidateTicketNumber** to **Yes**. Then click **OK**:



Note: If you have multiple ticketing systems you can enable only some of them for a specific platform, please refer to section "*Integrating with Ticketing Systems*" of the "*Privileged Account Security Implementation Guide*" documentation.

APPENDIX 1: XML EXAMPLE

```
<TicketingParameters>
      <Parameter Name="VulnerabilityManager" Value="Tenable.io">
        <Parameter Name="VulnerabilitySet: Restrict OracleDB with High Risk" Value="Prevent access to Databases with high risk plugins">
          <Parameter Name="REGEX:PolicyID" Value="Oracle" />
          <Parameter Name="DeviceType" Value="Database" />
          <Parameter Name="IN:plugin.id" Value="55786" />
          <Parameter Name="Message" Value=": $vulnerability.description (PluginID: $vulnerability.plugin_id) - Solution: $vulnerability.solution [See Also: $vulnerability.see_also]" />
          <Parameter Name="AllowTransparentConnection" Value="True" />
        </Parameter>
        <Parameter Name="VulnerabilitySet: OS - Gain Shell Remote - Critical" Value="Restrict Show/Copy to Operating  System, where vulnerability family: &amp;quot;Gain a shell remotely&amp;quot; and Severity 4">
          <Parameter Name="DeviceType" Value="Operating System" />
          <Parameter Name="plugin.family" Value="Gain a shell remotely" />
          <Parameter Name="Severity" Value="Critical" />
          <Parameter Name="AllowTransparentConnection" Value="True" />
          <Parameter Name="Message" Value=": $vulnerability.synopsis (PluginID: $vulnerability.plugin_id) - Solution: $vulnerability.solution [See Also: $vulnerability.see_also]" />
        </Parameter>
        <Parameter Name="VulnerabilitySet:No Access for Critical" Value="Any server with Critical Vulnerabilities">
          <Parameter Name="Severity" Value="Critical" />
          <Parameter Name="Message" Value=": Restricted access due to Critical Vulnerabilities" />
          <Parameter Name="NOT IN:RequestingUser" Value="John" />
          <Parameter Name="AllowTransparentConnection" Value="True" />
        </Parameter>
      </Parameter>
      <Parameter Name="Message" Value="Access Denied by Tenable.io" />
    </TicketingParameters>
```

APPENDIX 2: SUPPORTED PROPERTIES AND CONDITIONS

You can add the following Tenable.io and CyberArk parameters to further refine what a CyberArk user will be allowed to do on a server with Vulnerabilities:

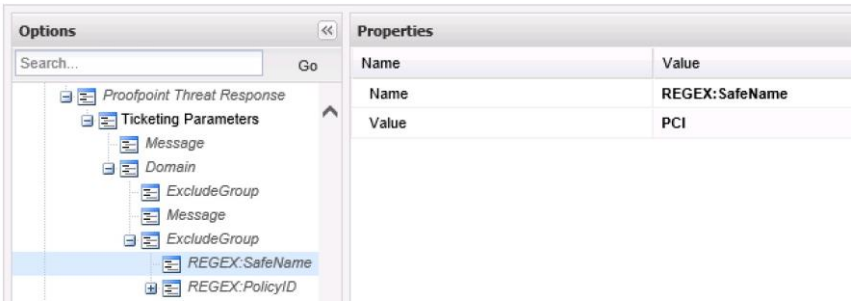
CyberArk Property	Description
SafeName	The Safe where the requested account is stored.
MachineAddress	The address of the machine specified on the account, if it exists.
TransparentMachine Address	The address of the machine used in a transparent connection, if it exists.
UserName	The name of the user specified on the requested account.
PolicyId	The name of the platform specified on the requested account.
RequestingUser	The name of the user requesting the account.
DeviceType	The DeviceType that's been accessed. It allows to restrict access to OS accounts while allowing access to Database accounts on a server with OS vulnerabilities.
AllowTransparentConnection	When True, this property grants login access to a vulnerable server while denying Show/Copy

In addition to the properties listed above, all target account properties are available, such as: DeviceType, Port, Database, LogonDomain, etc.

Supporting exact match comparisons or regular expressions you can add logic like:

- Do not allow if the DeviceType is "Operating System"
- Do not allow if SafeName or PolicyID has PCI in their name

If the property name is prefixed with **REGEX:** the value will be used as a regular expression comparison, otherwise it will be used as an exact match:



Tenable.io Property	Description
plugin.attributes.bid	Filter results based on if a Bugtraq ID is equal to, is not equal to, contains, or does not contain a given string (e.g., 51300).
plugin.attributes.exploit_framework_canvas	Filter results based on if the presence of an exploit in the CANVAS exploit framework is equal to or is not equal to true or false.
plugin.attributes.canvas_package	Filter results based on which CANVAS exploit framework package an exploit exists for. Options include CANVAS, D2ExploitPack, or White_Phosphorus.
plugin.attributes.xref:CERT-CC	Filter results based on if a CERT Advisory ID (now called Technical Cyber Security Alert) is equal to, is not equal to, contains, or does not contain a given string (e.g., TA12-010A).
plugin.attributes.xref:CERT	CERT Vulnerability ID. Equal, not equal, match, or not match a regular expression (ie: 10031)
plugin.attributes.exploit_framework_core	Filter results based on if the presence of an exploit in the CORE exploit framework is equal to or is not equal to true or false.
plugin.attributes.cpe	Filter results based on if the Common Platform Enumeration (CPE) is equal to, is not equal to, contains, or does not contain a given string (e.g., Solaris).
plugin.attributes.cve.raw	Filter results based on if a CVE reference is equal to, is not equal to, contains, or does not contain a given string (e.g., 2011-0123).
plugin.attributes.cvss_base_score	Filter results based on if a CVSS base score is less than, is more than, is equal to, is not equal to, contains, or does not contain a string (e.g., 5) This filter can be used to select by risk level. The severity ratings are derived from the associated CVSS score, where 0 is Info, less than 4 is Low, less than 7 is Medium, less than 10 is High, and a CVSS score of 10 will be flagged Critical.
plugin.attributes.cvss_temporal_score	Filter results based on if a CVSS temporal score is less than, is more than, is equal to, is not equal to, contains, or does not contain a string (e.g., 3.3).
plugin.attributes.cvss_temporal_vector.raw	Filter results based on if a CVSS temporal vector is equal to, is not equal to, contains, or does not contain a given string (e.g., E:F).
plugin.attributes.cvss_vector.raw	Filter results based on if a CVSS vector is equal to, is not equal to, contains, or does not contain a given string (e.g., AV:N).
plugin.attributes.cvss3_base_score	CVSS v3.0 Base Score. Equal, not equal, match, not match, greater than, or less than.
plugin.attributes.cvss3_temporal_score	CVSS v3.0 Temporal Score. Equal, not equal, match, not match, greater than, or less than.
plugin.attributes.cvss3_temporal_vector.raw	CVSS v3.0 Temporal Vector. Equal, not equal, match, or not match. Regex format: ^CVSS:3.0/E:(U|POC|F|H|ND)/RL:(OF|T|W|U|ND)/RC:(UC|UR|C|ND)$
plugin.attributes.cvss3_vector.raw	CVSS v3.0 Vector. Equal, not equal, match, or not match a regex: ^CVSS:3.0/E:(U|POC|F|H|ND)/RL:(OF|T|W|U|ND)/RC:(UC|UR|C|ND)$
plugin.attributes.xref:CWE	Filter results based on Common Weakness Enumeration (CWE®) if a CVSS vector is equal to, is not equal to, contains, or does not contain a CWE reference number (e.g., 200).
plugin.attributes.default_account	
plugin.attributes.exploit_framework_d2_elliot	Filter results based on if the presence of an exploit in the Elliot exploit framework is equal to or is not equal to true or false.
plugin.attributes.d2_elliot_name	Filter results based on if an Elliot exploit is equal to, is not equal to, contains, or does not contain a given string (e.g., Typo3 FD).
plugin.attributes.exploit_available	Filter results based on the vulnerability having a known public exploit.
plugin.attributes.xref:EDB-ID	Filter results based on if an Exploit Database ID (EBD-ID) reference is equal to, is not equal to, contains, or does not contain a given string (e.g., 18380).

Tenable.io Property	Description
plugin.attributes.exploit_framework_exploithub	Filter results based on if the presence of an exploit on the ExploitHub web site is equal to or is not equal to true or false.
plugin.attributes.exploitability_ease.raw	Filter results based on if the exploitability ease is equal to or is not equal to the following values: Exploits are available, No exploit is required, or No known exploits are available.
plugin.attributes.exploited_by_malware	Filter results based on if the presence of a vulnerability is exploitable by malware is equal to or is not equal to true or false.
plugin.attributes.exploited_by_nessus	
host.hostname	Filter results if the host is equal to, is not equal to, contains, or does not contain a given string (e.g., 192.168 or lab).
plugin.attributes.xref:IAVA	Filter results based on if an IAVA reference is equal to, is not equal to, contains, or does not contain a given string (e.g., 2012-A-0008).
plugin.attributes.xref:IAVB	Filter results based on if an IAVB reference is equal to, is not equal to, contains, or does not contain a given string (e.g., 2012-A-0008).
plugin.attributes.stig_severity	Filter results based on the IAVM severity level (e.g., IV).
plugin.attributes.xref:IAVT	Filter results based on if an IAVT reference is equal to, is not equal to, contains, or does not contain a given string (e.g., 2012-A-0008).
plugin.attributes.in_the_news	true or false
plugin.attributes.malware	true or false
plugin.attributes.exploit_framework_metasploit	Filter results based on if the presence of a vulnerability in the Metasploit Exploit Framework is equal to or is not equal to true or false.
plugin.attributes.metasploit_name	Filter results based on if a Metasploit name is equal to, is not equal to, contains, or does not contain a given string (e.g., xslt_password_reset).
plugin.attributes.xref:MSFT	Microsoft Bulletin. Readable_regex: MS0X-YZT. Regex format: ^MS[0-9]+-[0-9]+$
plugin.attributes.xref:OSVDB	Filter results based on if an Open Source Vulnerability Database (OSVDB) ID is equal to, is not equal to, contains, or does not contain a given string (e.g., 78300).
plugin.attributes.patch_publication_date	Filter results based on if a vulnerability patch publication date is less than, is more than, is equal to, is not equal to, contains, or does not contain a string (e.g., 12/01/2011).
plugin.attributes.description	Filter results if Plugin Description contains, or does not contain a given string (e.g., remote).
plugin.family	Filter results if Plugin Name is equal to or is not equal to one of the designated Nessus plugin families: AIX Local Security Checks, Amazon Linux Local Security Checks, Backdoors, CentOS Local Security Checks, CGI abuses, CGI abuses : XSS, CISCO, Databases, Debian Local Security Checks, Default Unix Accounts, Denial of Service, DNS, F5 Networks Local Security Checks, Fedora Local Security Checks, Firewalls, FreeBSD Local Security Checks, FTP, Gain a shell remotely, General, Gentoo Local Security Checks, HP-UX Local Security Checks, Huawei Local Security Checks, Incident Response, Junos Local Security Checks, MacOS X Local Security Checks, Mandriva Local Security Checks, Misc., Mobile Devices, Netware, Oracle Linux Local Security Checks, OracleVM Local Security Checks, Palo Alto Local Security Checks, Peer-To-Peer File Sharing, Policy Compliance, Red Hat Local Security Checks, RPC, SCADA, Scientific Linux Local Security Checks, Service detection, Settings, Slackware Local Security Checks, SMTP problems, SNMP, Solaris Local Security Checks, SuSE Local Security Checks, Ubuntu Local Security Checks, VMware ESX Local Security Checks, Web Servers, Windows, Windows : Microsoft Bulletins, Windows : User management

Tenable.io Property	Description
plugin.id	Filter results if Plugin ID is equal to, is not equal to, contains, or does not contain a given string (e.g., 42111).
plugin.attributes.plugin_modification_date	Filter results based on if a Nessus plugin modification date is less than, is more than, is equal to, is not equal to, contains, or does not contain a string (e.g., 02/14/2010).
plugin.name	Filter results if Plugin Name is equal to, is not equal to, contains, or does not contain a given string (e.g., windows).
output	Filter results if Plugin Description is equal to, is not equal to, contains, or does not contain a given string (e.g., PHP)
plugin.attributes.plugin_publication_date	Filter results based on if a Nessus plugin publication date is less than, is more than, is equal to, is not equal to, contains, or does not contain a string (e.g., 06/03/2011).
plugin.attributes.plugin_type	Filter results if Plugin Type is equal to or is not equal to one of the two types of plugins: local or remote.
port.port	Filter results based on if a port is equal to, is not equal to, contains, or does not contain a given string (e.g., 80).
port.protocol	Filter results if a protocol is equal to or is not equal to a given string (e.g., http).
plugin.attributes.xref:Secunia	Filter results based on if a Secunia ID is equal to, is not equal to, contains, or does not contain a given string (e.g., 47650).
plugin.attributes.see_also	Filter results based on if a Nessus plugin see also reference is equal to, is not equal to, contains, or does not contain a given string (e.g., seclists.org).
severity	Equal or not equal to: None, Low, Medium, High, Critical
plugin.attributes.solution	Filter results if the plugin Solution contains or does not contain a given string (e.g., upgrade).
plugin.attributes.synopsis	Filter results if the plugin Solution contains or does not contain a given string (e.g., PHP).
target_group	Equal or not equal to: default_group: 1, name: Default, shared: 1, owner: nessus_ms_agent, value: 4
plugin.attributes.unsupported_by_vendor	true or false
plugin.attributes.vuln_publication_date	Filter results based on if a vulnerability publication date earlier than, later than, on, not on, contains, or does not contain a string (e.g., 01/01/2012). Note: Pressing the button next to the date will bring up a calendar interface for easier date selection.
tracking.state	Vulnerability State. Equal or not equal to: New, Fixed, Active, Resurfaced

The following operators are available:

Operator Code	Operator Name
eq	Equal
neq	Not Equal
match	Match
nmatch	Not Match
lt	Less Than
gt	Greater Than
date-lt	Date Less Than
date-gt	Date Greater Than
date-eq	Date Equal
date-neq	Date Not Equal

APPENDIX 3: SUPPORTED VARIABLES

The following table contains a list of the variable allowed to be used on the Message parameter:

Variable	Description
$vulnerability.plugin_id	Plugin ID
$vulnerability.plugin_name	Plugin Name
$vulnerability.plugin_family	Plugin Family
$vulnerability.synopsis	Tenable.io provided vulnerability Synopsis
$vulnerability.solution	Tenable.io solution for vulnerability
$vulnerability.description	Vulnerability description
$vulnerability.see_also	References to see also