Forcepoint DLP

10.3

Creating Remediation Scripts

Revision A

Contents

Creating Remediation Scripts for Forcepoint DLP

This document describes how discovery and data loss prevention (DLP) incidents are created, and explains how remediation scripts can be used to respond to incidents.

Short samples of remediation scripts are included. Contents:

- *What are discovery and DLP?*
- *Introducing remediation scripts*
- *Remediation script requirements*
- *Sample network discovery incident XML*
- *Sample Exchange discovery incident XML*
- *Sample DLP incident XML*
- *Using the Discovery Incident Processing module*
- *Remediation script code samples*

Related concepts

What are discovery and DLP? on page 2
Remediation script requirements on page 5
Sample network discovery incident XML on page 5
Sample Exchange discovery incident XML on page 10
Sample DLP incident XML on page 13
Using the DiscoveryIncidentProcessing module on page 16
Remediation script code samples on page 18
Tips for writing multi-platform Python code on page 22

Related reference

Introducing remediation scripts on page 3

What are discovery and DLP?

This section explains about the discovery, DLP, and their incidents.

What is discovery?

Discovery is the act of automatically scanning data at rest, classifying it, and potentially enforcing an action on the classification.

Discovery can be performed by two Forcepoint DLP system components:

- The crawler performs network discovery.
- Forcepoint DLP Endpoint performs endpoint discovery.

Both components perform essentially the same activity.

Once the documents are classified as having matched a policy rule, an action plan associated with this rule is executed. The action plan specifies what happens next, and this can include running a remediation script.

What is DLP?

Data Loss Prevention (DLP) is the activity of classifying real-time data that is communicated on various channels, by various means. Once classified, the data sent over the communication channel might trigger a policy and generate an incident.

Once an incident occurs, an action plan is executed. The action plan may specify which remediation scripts to run.

Classification is performed by the Policy Engine. This means it can be executed on Windows or Linux policy engines, or executed on servers or endpoints (including Linux endpoints, although this functionality is currently not available).

What are discovery and DLP incidents?

- A discovery incident is created when content that matches one or more rules in a discovery policy is found.
- A DLP incident is created when content that matches one or more rules in a DLP policy is found.

When remediation scripts are used, each incident results in an XML file that contains the incident details. Details include:

- Information about the rule or rules that were matched
- Other meta data, such as the file permissions, source and destination, the policy engine name, and so on

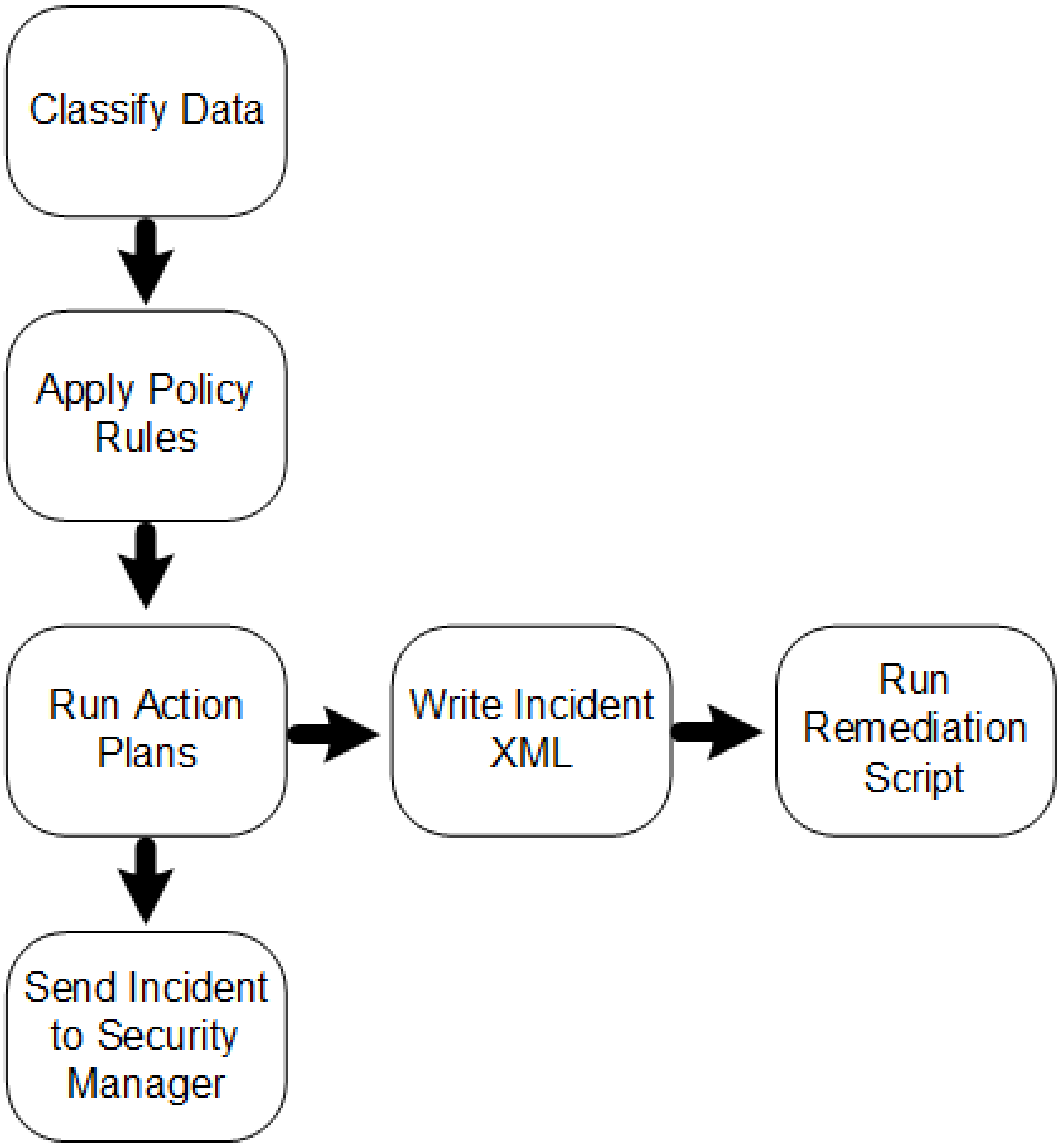
  The available meta data varies based on the type of incident.

The full path to the XML file is used as the first command-line parameter passed to the remediation script.

Introducing remediation scripts

Remediation scripts extend discovery and DLP functionality by allowing administrators configure how the system responds to specific types of incidents.

Each script can be run by a policy engine, endpoint agent, or management server when an incident is triggered.



Configure remediation scripts in the Data Security module of the Forcepoint Security Manager. Remediation scripts are considered resources, so they are managed from the **Main** > **Policy Management** > **Resources** > **Remediation Scripts** page (see Remediation scripts in the Forcepoint DLP Administrator Help).

Remediation scripts can be supplied with optional credentials, based on the operating system in which they run:

	Windows Server Policy Engine	**Linux Server Policy Engine**	**Windows Endpoint**	**Linux Endpoint**
With supplied credentials	Impersonate the supplied credentials	Not supported	Impersonate the supplied credentials	Not supported
Without supplied credentials	Impersonate the user running the policy engine	Effective UID of root	LocalSystem	Not supported

Remediation script limitations

- Remediation scripts are run after a response has been returned to the agent (Content Gateway, endpoint agent, protector, and so on). This means that remediation scripts cannot be used to alter data in motion.
- Remediation scripts do not have access to forensic information (the data the caused the incident).
- When there are several action plans configured for the same incident (in other words, when the incident matches multiple rules), all of the configured scripts are run in random order.
- On endpoint machines, scripts are run as the local system account. If impersonation is used, the endpoint installation folder is blocked for writing by anti-tampering protections.
- Remediation scripts cannot access the desktop. This means that:
  1) The script cannot be used to display messages to the user or open desktop applications.
  2) If scripting languages or executables generate popup windows (wscript echo, for example) the popups will be hidden and the script will hang.
- There is no built-in mechanism to stop scripts that are in a hung state.

Remediation script requirements

A remediation script can be an executable, or it can be a script written in an interpreted language.

The following scripting languages and language interpreters can be used to write remediation scripts without installing additional software:

- Windows - CMD Shell (.bat, .cmd)
- Windows & Linux - Python 2.5.4 (.py)
- Windows executable (.exe, .com)
- Linux ELF executable
- Linux BASH scripts

Other interpreted languages may be used, but they require installation of additional software. Make sure that the language interpreter is correctly installed on the server.

- For network discovery, the language interpreter must be installed on all crawler machines.
- For endpoint discovery, the language interpreter must be installed on all endpoint machines.

Because DLP and discovery incidents can occur on both Windows and Linux machines, it is highly desirable to write these scripts in Python—a language available on Windows and Linux servers as well as all endpoints as part of the Forcepoint DLP installation. In most cases, the same Python script can be used across operating systems without amendments.

The samples provided in this document are written in Python.

Sample network discovery incident XML

Discovery incident details take the form of an XML file with no DTD.

For example, the following sample is taken from network (file system) discovery:

```
<?xml version="1.0" encoding="UTF-8"?>
<ns1:pa-xml-rpc xmlns:ns1="http://www.portauthoritytech.com/ schmea/xml-rpc/1.0" xmlns:evt="http://
 www.portauthoritytech.com/schmea/incident/1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
<ns1:request>
<ns1:service-name>insertCrawlerService</ns1:service- name>
<ns1:params>
<evt:incident>
<evt:dataAtRest>
<evt:incidentInfo>
<evt:incidentId>5371106770671816417</ evt:incidentId>
<evt:serviceId isSecured="false">1800221564</ evt:serviceId>
<evt:analyzedBy>NLCTR.nolosscorp.com</ evt:analyzedBy>
 <evt:subject>\\10.4.228.150\DiscoveryTarget\TestFile.txt</ evt:subject>
<evt:localDetectedTime>2017-07- 18T14:54:11+10:00</evt:localDetectedTime>
<evt:installVersion>8.4</evt:installVersion>
<evt:resourceType>NETWORK</evt:resourceType>
```

Note

The <evt:resourceType> container has a value of either NETWORK or ENDPOINT, depending on whether the incident was triggered by network or endpoint discovery. This can be used to create scripts that work for both types of discovery.

```
<evt:totalSize>125</evt:totalSize>
</evt:incidentInfo>
<evt:rules>
```

Note

The <evt:rules> container holds a list of the violated rules.

```
<evt:rule id="170998" type="1" policyID="170893">
<evt:severity>2</evt:severity>
<evt:actionSettings id="172003"/>
<evt:numOfMatches>1</evt:numOfMatches>
<evt:classifierMatches>
```

Note

The <evt:classifierMatches> container includes a list of the classifiers matched within a rule.

```
<evt:classifierMatch id="171094">
<evt:numberOfMatches>1</ evt:numberOfMatches>
<evt:isTruncated>false</evt:isTruncated>
<evt:breachContent>
<evt:contentInfo>
<evt:pathPartInfo order="0">
<evt:path>\\10.4.228.150\DiscoveryTarget\TestFile.txt</ evt:path>
<evt:partType>3</evt:partType>
<evt:fileType>2</evt:fileType>
</evt:pathPartInfo>
</evt:contentInfo>
<evt:detectedValues>
<evt:detectedValue>
evt:unMasked>
<evt:unMasked>ForcepointTestKeyword</
</evt:detectedValue>
</evt:detectedValues>
<evt:numberOfMatches>1</
evt:numberOfMatches>
</evt:breachContent>
</evt:classifierMatch>
</evt:classifierMatches>
</evt:rule>
</evt:rules>
<evt:actionTaken type="2097152">
</evt:actionTaken>
<evt:properties>
```

Note

The <evt:properties> container includes a list of properties such as file owner, file ACL, and discovery task name.

```
<evt:property>
<evt:name>acl</evt:name>
 <evt:value>NLC\Administrator:wr,BUILTIN\Administrators:wr,NL C\websense:r,NT AUTHORITY\SYSTEM:wr</
evt:value>
</evt:property>
<evt:property>
<evt:name>checksum</evt:name>
<evt:value>7a0627c2efa25daedb56f19b79c22ab7</
evt:value>
</evt:property>
<evt:property>
<evt:name>fileOwner</evt:name>
<evt:value>BUILTIN\Administrators</evt:value>
</evt:property>
<evt:property>
<evt:name>folderOwner</evt:name>
<evt:value>BUILTIN\Administrators</evt:value>
</evt:property>
<evt:property>
<evt:name>jobID</evt:name>
<evt:value>172104</evt:value>
</evt:property>
<evt:property>
<evt:name>jobName</evt:name>
<evt:value>RemediationTest</evt:value>
</evt:property>
<evt:property>
<evt:name>resourceSubType</evt:name>
<evt:value>NETWORK</evt:value>
</evt:property>
</evt:properties>
<evt:file>
<evt:filepath>cifs://10.4.228.150/ DiscoveryTarget/TestFile.txt</evt:filepath>
```

Note

The <evt:filepath> container includes the path to the file in URI format. (To convert the path to UNC format, replace the slashes with backslashes.)

```
<evt:filesize>39</evt:filesize>
<evt:filetype>2</evt:filetype>
<evt:encodeType>N/A</evt:encodeType>
<evt:ip>10.4.228.150</evt:ip>
<evt:dateAccessed>2017-07-18T14:51:54</ evt:dateAccessed>
<evt:dateCreated>2017-07-18T14:51:54</ evt:dateCreated>
<evt:dateModified>2017-07-18T14:52:16</ evt:dateModified>
```

Note

The value of the <evt:dateModified> container is the date and time that the file was last modified in YYYY-MM- DDTHH:MM:SS format.

```
<evt:owner>
<evt:incidentUser>
```

Note

Network discovery incidents include an <evt:incidentUser> container with a value attribute of DOMAIN\Username (type 5). On the endpoint, the value attribute is the user's SID (type 8).

```
<evt:detail type="5" value="BUILTIN\Administrators" isLookedUp="false"/>
</evt:incidentUser>
</evt:owner>
<evt:folderOwner>
<evt:incidentUser>
<evt:detail type="5" value="BUILTIN\Administrators" isLookedUp="false"/>
</evt:incidentUser>
</evt:folderOwner>
</evt:file>
<evt:jobId>172104</evt:jobId>
<evt:jobName></evt:jobName>
<evt:scanStartTime>2017-07-18T14:54:06</ evt:scanStartTime>
<evt:discoveryEndpointInfo>
<evt:endpointType>Unknown</evt:endpointType>
</evt:discoveryEndpointInfo>
</evt:dataAtRest>
</evt:incident>
</ns1:params>
</ns1:request>
</ns1:pa-xml-rpc>
```

To compare this example with the XML file created for an Exchange incident, continue with *Sample Exchange discovery incident XML*.

Related concepts

Sample Exchange discovery incident XML on page 10

Sample Exchange discovery incident XML

Here is a sample incident XML file resulting from Exchange discovery:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ns1:pa-xml-rpc xmlns:ns1="http://www.portauthoritytech.com/ schmea/xml-rpc/1.0" xmlns:evt="http://
 www.portauthoritytech.com/schmea/incident/1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
<ns1:request>
<ns1:service-name>insertCrawlerService</ns1:service- name>
<ns1:params>
<evt:incident>
<evt:dataAtRest>
<evt:incidentInfo>
<evt:incidentId>4679778800686204169</ evt:incidentId>
<evt:serviceId isSecured="false">1800221564</ evt:serviceId>
<evt:analyzedBy>NLCTR.nolosscorp.com</ evt:analyzedBy>
<evt:subject>ismith/Deleted Items/DSS Incident [ID:12564].EML</evt:subject>
<evt:localDetectedTime>2017-07- 26T14:17:57+10:00</evt:localDetectedTime>
<evt:installVersion>8.4</evt:installVersion>
<evt:resourceType>EXCHANGE</evt:resourceType>
<evt:totalSize>36827</evt:totalSize>
</evt:incidentInfo><evt:rules>
<evt:rule id="170998" type="1" policyID="170893">
<evt:severity>2</evt:severity>
<evt:actionSettings id="172003"/>
<evt:numOfMatches>1</evt:numOfMatches>
<evt:classifierMatches>
<evt:classifierMatch id="171094">
<evt:numberOfMatches>1</ evt:numberOfMatches>
<evt:isTruncated>false</evt:isTruncated>
<evt:breachContent>
<evt:contentInfo>
<evt:pathPartInfo order="0">
<evt:path>ismith/Deleted Items/DSS Incident [ID:12564].EML</evt:path>
<evt:partType>1</evt:partType>
<evt:fileType>233</evt:fileType>
</evt:pathPartInfo>
<evt:pathPartInfo order="1">
<evt:path>Transaction Body.txt</evt:path>
<evt:partType>1</evt:partType>
<evt:fileType>236</evt:fileType>
</evt:pathPartInfo>
</evt:contentInfo>
<evt:detectedValues>
<evt:detectedValue>
evt:unMasked>
<evt:unMasked>WebsenseTestKeyword</
</evt:detectedValue>
</evt:detectedValues>
<evt:numberOfMatches>1</evt:numberOfMatches>
</evt:breachContent>
<evt:breachContent>
<evt:contentInfo>
<evt:pathPartInfo order="0">
<evt:path>ismith/Deleted Items/DSS Incident [ID:12564].EML</evt:path>
<evt:partType>1</evt:partType>
<evt:fileType>233</evt:fileType>
</evt:pathPartInfo>
<evt:pathPartInfo order="1">
 <evt:path>Original_Message_Incident_12564</evt:path>
<evt:partType>2</evt:partType>
<evt:fileType>233</evt:fileType>
</evt:pathPartInfo>
<evt:pathPartInfo order="2">
<evt:path>Transaction Body.txt</evt:path>
evt:unMasked>
```

Sample Exchange discovery incident XML (contd...)

```
<evt:partType>2</evt:partType>
<evt:fileType>2</evt:fileType>
</evt:pathPartInfo>
</evt:contentInfo>
<evt:detectedValues>
<evt:detectedValue>
<evt:unMasked>WebsenseTestKeyword</
</evt:detectedValue>
</evt:detectedValues>
<evt:numberOfMatches>1</
evt:numberOfMatches>
</evt:breachContent>
</evt:classifierMatch>
</evt:classifierMatches>
</evt:rule>
</evt:rules>
<evt:actionTaken type="2097152">
</evt:actionTaken>
<evt:properties>
<evt:property>
<evt:name>checksum</evt:name>
<evt:value>60104d41558c2d6aba1ad287813155ea</
evt:value>
</evt:property>
<evt:property>
<evt:name>exchange-from</evt:name>
<evt:value>&quot;DSS@nolosscorp.com&quot;
&lt;DSS@nolosscorp.com></evt:value>
</evt:property>
<evt:property>
<evt:name>exchange-subject</evt:name>
<evt:value>DSS Incident [ID:12564]</evt:value>
</evt:property>
<evt:property>
<evt:name>exchange-to</evt:name>
<evt:value>&quot;ismith@nolosscorp.com&quot; &lt;ismith@nolosscorp.com></evt:value>
</evt:property>
<evt:property>
<evt:name>fileOwner</evt:name>
<evt:value>ismith</evt:value>
</evt:property>
<evt:property>
<evt:name>folderOwner</evt:name>
<evt:value>N/A</evt:value>
</evt:property>
<evt:property>
<evt:name>jobID</evt:name>
<evt:value>172106</evt:value>
</evt:property>
<evt:property>
<evt:name>jobName</evt:name>
<evt:value>Test discovery</evt:value>
</evt:property>
<evt:property>
<evt:name>resourceSubType</evt:name>
<evt:value>PRIVATE FOLDER</evt:value>
</evt:property>
</evt:properties>
```

Sample Exchange discovery incident XML (contd..)

```
<evt:file>
<evt:filepath>cifs://ismith/Deleted Items/DSS Incident [ID:12564].EML</evt:filepath>
<evt:filesize>19672</evt:filesize>
<evt:filetype>233</evt:filetype>
<evt:encodeType>N/A</evt:encodeType>
<evt:hostname>ismith@nolosscorp.com</ evt:hostname>
<evt:dateAccessed>2010-10-21T03:10:51.505</ evt:dateAccessed>
<evt:dateCreated>2010-10-21T03:10:51.505</ evt:dateCreated>
<evt:dateModified>2010-10-21T03:10:51.505</ evt:dateModified>
<evt:owner>
<evt:incidentUser>
<evt:detail type="5" value="ismith" isLookedUp="false"/>
</evt:incidentUser>
</evt:owner>
<evt:folderOwner>
<evt:incidentUser>
<evt:detail type="5" value="N/A" isLookedUp="false"/>
</evt:incidentUser>
</evt:folderOwner>
</evt:file>
<evt:jobId>172106</evt:jobId>
<evt:jobName></evt:jobName>
<evt:scanStartTime>2017-07-26T14:16:49</ evt:scanStartTime>
<evt:discoveryEndpointInfo>
<evt:endpointType>Unknown</evt:endpointType>
</evt:discoveryEndpointInfo>
</evt:dataAtRest>
</evt:incident>
</ns1:params>
</ns1:request>
</ns1:pa-xml-rpc>
```

Please note the main differences between the network discovery incident and this Exchange incident:

- The <evt:parameters> containers hold more Exchange-specific information, such as email fields.
- The pathname in the <evt:file> section is invalid as a path name, but is valid as a URL suffix in OWA.
- The <evt:resourceType> value is EXCHANGE.

Include parsing code in custom scripts to get information from Exchange incidents. The sample script cannot extract any meaningful information from it.

Sample DLP incident XML

The following XML example is for a DLP incident:

```
<?xml version="1.0" encoding="UTF-8"?>
<ns1:pa-xml-rpc xmlns:ns1="http://www.portauthoritytech.com/ schmea/xml-rpc/1.0" xmlns:evt="http://
 www.portauthoritytech.com/schmea/incident/1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
<ns1:request>
 <ns1:service-name>insertEventService</ns1:service-name>
<ns1:params>
<evt:incident>
<evt:dataInMotion>
<evt:incidentInfo>
<evt:incidentId>5352285115603247792</ evt:incidentId>
<evt:serviceId isSecured="false">486169846</ evt:serviceId>
<evt:analyzedBy>nlcv10k-c-esg.nolosscorp.com</ evt:analyzedBy>
<evt:subject>test inbound 3</evt:subject>
```

Note

The value of the <evt:subject> container is channel dependent. For example, for email, it is the email subject. This same value appears in the subject field in incident reports.

```
<evt:localDetectedTime>2017-07- 21T12:33:35+10:00</evt:localDetectedTime>
<evt:installVersion>8.4</evt:installVersion>
<evt:resourceType>NETWORK</evt:resourceType>
```

Note

The <evt:resourceType> container has a value of either NETWORK or ENDPOINT, depending on the type of DLP incident.

```
<evt:totalSize>1740</evt:totalSize>
</evt:incidentInfo>
<evt:rules>
```

Note

The <evt:rules> container holds a list of the rules violated by the incident.

```
<evt:rule id="171601" type="1" policyID="170899">
<evt:severity>2</evt:severity>
<evt:actionSettings id="172004"/>
<evt:numOfMatches>1</evt:numOfMatches>
<evt:classifierMatches>
```

Note

The <evt:classifierMatches> container includes a list of matched classifiers within a specific rule.

```
<evt:classifierMatch id="171094">
<evt:numberOfMatches>1</ evt:numberOfMatches>
<evt:isTruncated>false</evt:isTruncated>
<evt:breachContent>
<evt:contentInfo>
<evt:pathPartInfo order="0">
<evt:path>/var/spool/postfix/tmp// 887C7850695.eml</evt:path>
<evt:partType>1</evt:partType>
<evt:fileType>233</evt:fileType>
</evt:pathPartInfo>
<evt:pathPartInfo order="1">
<evt:path>Transaction Body.txt</
evt:path>
<evt:partType>1</evt:partType>
<evt:fileType>2</evt:fileType>
evt:unMasked>
</evt:pathPartInfo>
</evt:contentInfo>
<evt:detectedValues>
<evt:detectedValue>
<evt:unMasked>WebsenseTestKeyword</
</evt:detectedValue>
</evt:detectedValues>
<evt:numberOfMatches>1</
evt:numberOfMatches>
</evt:breachContent>
</evt:classifierMatch>
</evt:classifierMatches>
</evt:rule>
</evt:rules>
<evt:actionTaken type="2097152">
</evt:actionTaken>
<evt:source>
```

Note

The <evt:source> container includes the source of the incident. The resource type depends on the source (-2 is email).

```
<evt:incidentUser>
<evt:detail type="2" value="test@arik.baratz.org" isLookedUp="false"/>
</evt:incidentUser>
</evt:source>
<evt:destinations>
```

Note

The <evt:destinations> container includes one or more incident destinations. In this sample file, the destinations are email addresses.

```
<evt:destination>
<evt:incidentUser>
<evt:detail type="2" value="administrator@nolosscorp.com" isLookedUp="false"/>
</evt:incidentUser>
<evt:destinationType>TO</evt:destinationType>
<evt:actionTaken type="2097152">
</evt:actionTaken>
<evt:direction>1</evt:direction>
</evt:destination>
<evt:destination>
<evt:incidentUser>
<evt:detail type="2" value="ragg@nolosscorp.com" isLookedUp="false"/>
</evt:incidentUser>
<evt:destinationType>TO</evt:destinationType>
<evt:actionTaken type="2097152">
</evt:actionTaken>
<evt:direction>1</evt:direction>
</evt:destination>
<evt:destination>
<evt:incidentUser>
<evt:detail type="2" value="ismith@nolosscorp.com" isLookedUp="false"/>
</evt:incidentUser>
<evt:destinationType>TO</evt:destinationType>
<evt:actionTaken type="2097152">
</evt:actionTaken>
<evt:direction>1</evt:direction>
</evt:destination>
</evt:destinations>
<evt:eventEndpointInfo>
<evt:endpointType>Unknown</evt:endpointType>
<evt:endpointSourceAppName>N/A</ evt:endpointSourceAppName>
<evt:endpointDestAppName>N/A</ evt:endpointDestAppName>
<evt:endpointDestDeviceName>N/A</ evt:endpointDestDeviceName>
<evt:endpointDestDeviceType>N/A</ evt:endpointDestDeviceType>
<evt:endpointOperationType>N/A</ evt:endpointOperationType>
<evt:endpointPolicyVersion>0</ evt:endpointPolicyVersion>
<evt:confirmationId>0</evt:confirmationId>
 <evt:confirmationString></evt:confirmationString>
<evt:endpointSourceAppID>N/A</ evt:endpointSourceAppID>
<evt:endpointDestAppID>N/A</ evt:endpointDestAppID>
</evt:eventEndpointInfo>
<evt:hasForensics>true</evt:hasForensics>
</evt:dataInMotion>
</evt:incident>
</ns1:params>
</ns1:request>
</ns1:pa-xml-rpc>
```

Continue with *Using the Discovery Incident Processing module*.

Related concepts

Using the DiscoveryIncidentProcessing module on page 16

Using the DiscoveryIncidentProcessing module

To make it easier to write remediation scripts for common use cases, Forcepoint DLP includes a helper Python module that performs some common tasks with the incident data XML file. The module:

- Is named Discovery Incident Processing
- Can be easily imported into your Python code
- Is not required

  Administrators can instead write their own XML parsing routines.

Note

The DiscoveryIncidentProcessing module cannot be used on endpoints with impersonation.

The DiscoveryIncidentProcessing module includes the routines described in the following sections.

GetFilePathFromXML(IncidentXml)

This routine reads and analyzes the incident details from the provided XML file.

Parameters

IncidentXml	Unicode	The path to the XML file containing the incident details.

Returns

0	Unicode	The discovery location: NETWORK or ENDPOINT
1	Unicode	The absolute file path. For endpoint discovery, uses UNC format `\\hostname\C\path\to\file`.
2	Unicode	True - File exists False - File not found

Example

```
>>> import DiscoveryIncidentProcessing
>>>
DiscoveryIncidentProcessing.GetFilePathFromXML(r'C:\Temp\ 5371106770671816417.xml')
(u'NETWORK',
u'\\\\10.4.228.150\\DiscoveryTarget\\TestFile.txt', True)
>>>
```

ProcessDicoveryIncident(IncidentXml, Command)

This routine runs a command, providing the incident file name as a parameter. This is quite useful to run commands that expect the original file as one of its parameters.

Note

The typo in the function name will be fixed in future versions.

Parameters

IncidentXML	Unicode	The path of the incident XML file.
Command	Unicode	A command to execute The string should contain the string "$filepath$", which is replaced with the actual filename in the incident XML.

Returns

None

Example

```
>>>
DiscoveryIncidentProcessing.ProcessDicoveryIncident(r'C:\ Temp\5371106770671816417.xml',
u'notepad.exe filepath ')
2017-07-19 18:32:45,312 root Debug Processing
C:\Temp\5371106770671816417.xml Encryption
2017-07-19 18:32:45,496 root Debug Processing
\\10.4.228.150\DiscoveryTarget\TestFile.txt
2017-07-19 18:32:45,500 root Debug
Command:notepad.exe
\\10.4.228.150\DiscoveryTarget\TestFile.txt
2017-07-19 18:32:50,898 root Debug
\\10.4.228.150\DiscoveryTarget\TestFile.txt RunCommand Successful
>>>
```

MoveDiscoveryIncident (IncidentXml, Location, RemoveFile, DaysKeepActiveFiles, QuarentineMsg)

This routine moves the file pointed to by the incident into a folder.

The file is moved by copying it to the destination folder, then overwriting the original file with a text message.

Alternatively, the file can be copied (rather than moved).

The file is checked for access before it is copied or moved, and it is not moved if it has been accessed recently.

Parameters

IncidentXML	Unicode	The path of the incident XML file

Location	Unicode	Destination folder to which to move or copy the file
RemoveFile	bool	If True, the original file is moved. If False, the original file is copied.
DaysKeepActiveFiles	Int	Don’t move the file if it was accessed within this number of days.
QuarantineMsg	str	A string which will replace the original file Make sure the file is formatted appropriately. For example, to use Unicode, encode it as UTF-8 or UTF-16 with BOM. The file will always have a “.txt” extension, so make sure it can be opened in Notepad.

Returns

None

Example

```
>>>
DiscoveryIncidentProcessing.MoveDiscoveryIncident(r'C:\Te mp\5371106770671816417.xml',r'C:
\Temp',False,0,'')
2017-07-21 16:03:16,365 root Debug Processing
C:\Temp\5371106770671816417.xml move file 0
2017-07-21 16:03:16,742 root Debug Moving
\\10.4.228.150\DiscoveryTarget\TestFile.txt to C:\Temp
2017-07-21 16:03:16,786 root Debug Creating
C:\Temp\10.4.228.150\DiscoveryTarget
>>>
```

Remediation script code samples

Some of the code samples in this section use DiscoveryIncidentProcessing, but others do not.

Example 1

This example is one of the 3 scripts that come with Forcepoint DLP and reside in the `%dss_home%/RunCommands` folder.

It provides an example of using the DiscoveryIncidentProcessing module, and is self- explanatory.

```
#Set the destination folder for sensitive files
Location = r'\\127.0.0.1\quarantine'
DaysKeepActiveFiles = 0
#
#DO NOT MODIFYSCRIPT PAST THIS LINE
import sys
from DiscoveryIncidentProcessing import
MoveDiscoveryIncident
MoveDiscoveryIncident(sys.argv[1],Location,False,DaysKeepAct iveFiles,'')
```

Example 2

The following example is a little more involved and does the XML parsing without using the helper module, which is useful only for discovery incidents.

```
#Send an email to the recipients of a DLP incident
# XML search path constants for easier XML handling
NS1=u".//{http://www.portauthoritytech.com/schmea/xml-rpc/ 1.0}"
EVT=u".//{http://www.portauthoritytech.com/schmea/incident/ 1.0}"
EVTSOURCE=EVT+u"source"
EVTDETAIL=EVT+u"detail"
EVTDESTINATIONS=EVT+u"destinations"
EVTDESTINATION=EVT+u"destination"
EVTSUBJECT=EVT+u"subject"
## Email message to send to the users
EMAILMESSAGE="""From: Forcepoint DLP <dlp@example.com>\r
To: %(deststring)s\r
Subject: Re: %(subject)s\r
\r
Dear Sir or Madam,\r
\r
A message sent to you from %(source)s with the subject "%(subject)s" has violated PCI regulations,
 and has been blocked by Forcepoint DLP. Please contact the sender and request that they redact all
 cardholder information (such as name, credit card number, expiration date, CVV) from the message
 and resend it.\r
\r
Regards,\r
\r
Forcepoint DLP\r
"""
# Email gateway
SMTPGATEWAY='10.4.228.240:25'
import sys
import xml.etree.ElementTree as ET
import smtplib
# Parse the XML file
oXMLTree=ET.parse(sys.argv[1])
## Search for a few key pieces of data
dIncidentDetails={}
# source
dIncidentDetails['source']=oXMLTree.find(EVTSOURCE).find(EVT DETAIL).get('value')
# destinations
lDests=[
elem.find(EVTDETAIL).get('value')
for elem
in oXMLTree.find(EVTDESTINATIONS)
]
dIncidentDetails['deststring']=', '.join(lDests)
# extract the subject
dIncidentDetails['subject']=oXMLTree.find(EVTSUBJECT).text
## send an email message
oSMTP=smtplib.SMTP(SMTPGATEWAY)
oSMTP.sendmail(dIncidentDetails['source'],lDests,EMAILMESSAG E % dIncidentDetails)
oSMTP=None
```

Please note several important aspects of this example:

- The search paths constructed on lines 4–10 do not contain the xmlns shortcuts for the namespaces but rather the full URL. This is a requirement of the particular XML library used (xml.etree). It is also more resilient to future changes in the XML format.
- Lines 13 to 24 contain an IMF message (RFC5322) ready to be sent by SMTP. The Pythonic variable substitutions inside the script makes for an easy ad-hock template language.
- Lines 30 and 34 are all that is required to parse the XML file. Then, extracting information from the XML file is done in a series of searches on lines 40, 44, 46, and 51. It does not get easier than this.
- Lines 43–47 are a Python construct called “list comprehension”—a powerful and useful Python constructs for transforming lists. Find more information about it at http://www.python.org/dev/peps/pep-0202/.
- Lines 55–57 send an SMTP message. The gateway must accept SMTP connections for this to work.
- There is a bug in this code: if an incoming mail has destinations outside of the organization as well as inside, then all of the destinations will receive the notification message. This is usually not the desired action. Fixing it is easy and left as an exercise to the reader.

Example 3

This example is a VB script that copies or moves files found in a discovery scan.

```
option explicit
const isMove = True
const quarantineFolder = "\\10.0.46.40\quarantine"
const quarantineText = "Content has been removed please contact administrator"
'
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ '
Dim xmlFile
Dim xmlDoc
Dim Node
Dim filePath
Dim objFSO
Dim objFile
Dim root
Dim destFilePath
Set objFSO = CreateObject("Scripting.FileSystemObject")
'Functions
'---------
Function GeneratePath(pFolderPath)
GeneratePath = False
wscript.echo "GeneratePath " & pFolderPath
If Not objFSO.FolderExists(pFolderPath) Then
If GeneratePath(objFSO.GetParentFolderName(pFolderPath))
Then
GeneratePath = True
Call objFSO.CreateFolder(pFolderPath)
End If
Else
GeneratePath = True
End If
End Function
'main '----
xmlFile = WScript.Arguments.Item(0)
set xmlDoc=CreateObject("Microsoft.XMLDOM")
if xmlDoc.load(xmlFile) then
wscript.echo "Load XML succeeded"
else
wscript.echo "Load XML failed"
wscript.exit -1
end if
Set Node = xmlDoc.documentElement.selectSingleNode("// ns1:pa-xml-rpc/ns1:request/ns1:params/
evt:incident/ evt:dataAtRest/evt:incidentInfo/evt:resourceType")
if Node.text <> "NETWORK" and Node.text <> "ENDPOINT" then
wscript.echo "Incident is not file system discovery incident"
wscript.exit 0
end if
Set Node = xmlDoc.documentElement.selectSingleNode("// ns1:pa-xml-rpc/ns1:request/ns1:params/
evt:incident/ evt:dataAtRest/evt:file/evt:filepath")
filePath = right(Node.text,len(Node.text)-5)
wscript.echo "file path is : " & filePath
destFilePath = quarantineFolder + "\" + right(filePath,len(filePath)-2)
wscript.echo "Destination: " & destFilePath
GeneratePath(objFSO.GetParentFolderName(destFilePath))
objFSO.CopyFile filePath, destFilePath
if isMove then
Set objFile = objFSO.CreateTextFile(filePath + ".txt")
objFile.WriteLine(quarantineText)
objFile.Close
objFSO.DeleteFile filePath
end if
wscript.echo "File, " & filePath & " was processed successfully"
```

To invoke the script, create a batch file with this command: `cscript "%~dp0DiscoveryIncidentProcessing.vbs" %1 %2`

Please note that this script requires **cscript.exe**; using wscript.exe will halt the script.

Example 4

The following example is short but very useful:

```
# Copy the context xml file into a backup folder
import sys,os
fileName=sys.argv[1]
# Check the current OS platform
if sys.platform[:5]=='linux':
tempFolder='/tmp'
elif sys.platform[:3]=='win':
tempFolder=r'c:\temp' # assumes the folder is there!!! else:
# different platform?
sys.exit(1)
newName=os.path.join(tempFolder,os.path.split(fileName)[1])
# copy!
newFile=open(newName,'wb')
oldFile=open(fileName,'rb')
newFile.write(oldFile.read())
newFile.close()
```

This short piece of code simply copies the provided XML file to a temporary location, usually for debugging or further examination.

Note

Lines 7–13 demonstrate how to write a remediation script that is valid on both Windows and Linux. It uses the "sys.platform" facility to determine the file system structure. The complete string in sys.platform may be linux2, win32, etc., but the script only looks at the first few characters to make a determination.

See *Tips for writing multi-platform Python code*.

Related concepts

Tips for writing multi-platform Python code on page 22

Tips for writing multi-platform Python code

Additional note: When writing multi-platform code in Python, it is best to avoid making assumptions about the environment, if possible. Python helps quite a bit in that regard by providing facilities that performs certain actions regardless of the platform. A very common example is concatenating filenames and folders.

The following code is not cross-platform because it will fail on a Linux machine.

```
if not folderName[-1:]=='\\':
folderName+=r'\\'
pathName=folderName+fileName
```

Python provides a comprehensive library of path handling routines (os.path) that works regardless of the operating system the script runs on: `pathName = os.path.join(folderName,fileName)`

The same applies to other operations such as splitting a path. For example, consider the following script:

```
pathParts = pathName.rsplit('/',1)
if len(pathParts)==2:
folderName,fileName=pathParts
```

It would work just as well using the following: `folderName,fileName = os.path.split(pathName)`

These scripts would work on both operating systems, and also cover corner cases you may not have considered (for example, the input is just a folder and not a filename).

Note

From Forcepoint DLP 10.3 release, the usage of Python 3.6.8 is initiated. This will impose an issue for customers running their own python 2.5 remediation scripts. For suggested solutions, see the Python Remediation Scripts Compatibility Matrix knowledge base article.