

# ISM Schematron Rules

The following documentation is informative. The actual Schematron files are the normative record. This documentation is generated from the Schematron files via XSLT it may be missing some file or pieces of a file but whatever is here other than titles came from the original file.

It is envisioned that this will be a useful resource to search and read but for questions and debates the source files should be consulted.

Rules are all of the format ISM-ID-XXXXX any other heading is a supporting file that may strongly influence a rule but is not actually a numbered rule.

## FileName:./ISM\_XML.sch

### Code Description:

This is the root file for the ISM Schematron rule set. It loads all of the required CVEs declares some variables and includes all of the Rule .sch files.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- UNCLASSIFIED --><!-- WARNING:
Once compiled into an XSLT the result will
be the aggregate classification of all the CVEs
and included .sch files
-->
<sch:schema xmlns:sch="http://purl.oclc.org/dsdl/schematron"
xmlns:cve="urn:us:gov:ic:cve:v1"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
queryBinding="xslt2">
<sch:ns uri="urn:us:gov:ic:ism" prefix="ism"/>
<sch:ns uri="urn:us:gov:ic:cve:v1" prefix="cve"/>
<sch:ns uri="deprecated:value:function" prefix="dvf"/>

<!-- (U) Resources -->
<sch:let name="countriesList"
value="document('.../CVE/ISM/CVEnumISMOwnerProducer.xml')//cve:CVE/cve:Enumeration/
cve:Term/cve:Value"/>
<sch:let name="classificationAllList"
value="document('.../CVE/ISM/CVEnumISMClassificationAll.xml')//cve:CVE/cve:Enumeration/
cve:Term/cve:Value"/>
<sch:let name="classificationUSList"
value="document('.../CVE/ISM/CVEnumISMClassificationUS.xml')//cve:CVE/cve:Enumeration/
cve:Term/cve:Value"/>
<sch:let name="ownerProducerList"
value="document('.../CVE/ISM/CVEnumISMOwnerProducer.xml')//cve:CVE/cve:Enumeration/
cve:Term/cve:Value"/>
<sch:let name="declassExceptionList"
value="document('.../CVE/ISM/CVEnumISM25X.xml')//cve:CVE/cve:Enumeration/cve:Term/
cve:Value"/>
<sch:let name="FGISourceOpenList"
value="document('.../CVE/ISM/CVEnumISMFGIOpen.xml')//cve:CVE/cve:Enumeration/cve:Term/
cve:Value"/>
<sch:let name="FGISourceProtectedList"
value="document('.../CVE/ISM/CVEnumISMFGIProtected.xml')//cve:CVE/cve:Enumeration/
cve:Term/cve:Value"/>
<sch:let name="nonICmarkingsList"
value="document('.../CVE/ISM/CVEnumISMNonIC.xml')//cve:CVE/cve:Enumeration/cve:Term/
cve:Value"/>
<sch:let name="releasableToList"
```

```

value="document('.../CVE/ISM/CVEnumISMRelTo.xml')//cve:CVE/cve:Enumeration/cve:Term/
cve:Value"/>
<sch:let name="SCIControlsList"
value="document('.../CVE/ISM/CVEnumISMSCIControls.xml')//cve:CVE/cve:Enumeration/
cve:Term/cve:Value"/>
<sch:let name="SARIdentifierList"
value="document('.../CVE/ISM/CVEnumISM SAR.xml')//cve:CVE/cve:Enumeration/cve:Term/
cve:Value"/>
<sch:let name="validAttributeList"
value="document('.../CVE/ISM/CVEnumISMAttributes.xml')//cve:CVE/cve:Enumeration/
cve:Term/cve:Value"/>
<sch:let name="validElementList"
value="document('.../CVE/ISM/CVEnumISMElements.xml')//cve:CVE/cve:Enumeration/cve:Term/
cve:Value"/>
<sch:let name="noticeList"
value="document('.../CVE/ISM/CVEnumISMNotice.xml')//cve:CVE/cve:Enumeration/cve:Term/
cve:Value"/>
<sch:let name="nonUSControlsList"
value="document('.../CVE/ISM/CVEnumISMNonUSControls.xml')//cve:CVE/cve:Enumeration/
cve:Term/cve:Value"/>
<sch:let name="compliesWithList"
value="document('.../CVE/ISM/CVEnumISMCompliesWith.xml')//cve:CVE/cve:Enumeration/
cve:Term/cve:Value"/>
<sch:let name="atomicEnergyMarkingsList"
value="document('.../CVE/ISM/CVEnumISMAtomicEnergyMarkings.xml')//cve:CVE/
cve:Enumeration/cve:Term/cve:Value"/>
<sch:let name="displayOnlyToList"
value="document('.../CVE/ISM/CVEnumISMRelTo.xml')//cve:CVE/cve:Enumeration/cve:Term/
cve:Value"/>

<!-- (U) Resources that may include FOUO values -->
<sch:let name="disseminationControlsList"
value="document('.../CVE/ISM/CVEnumISMDissem.xml')//cve:CVE/cve:Enumeration/cve:Term/
cve:Value"/>

<!--=====>
<!-- (U) Universal Lets -->
<!--=====>
<!-- ISM_RESOURCE_ELEMENT (node): The first element with attribute
ism:resourceElement='true' -->
<sch:let name="ISM_RESOURCE_ELEMENT"
value="(for $each in (/*) return if($each/@ism:resourceElement=true()) then $each else
null)[1]"/>

<!-- (U) ISM_RESOURCE_CREATE_DATE (date): The date a Resource was created, or the
ism:createDate attribute on the
ISM_RESOURCE_ELEMENT node. -->
<sch:let name="ISM_RESOURCE_CREATE_DATE" value="$ISM_RESOURCE_ELEMENT/@ism:createDate"/>

```

```

<!-- (U) ISM_CAPCO_RESOURCE (boolean): True if the resource is a CAPCO-applicable
resource, or the ism:ownerProducer attribute on the
ISM_RESOURCE_ELEMENT node contains [USA]. False otherwise. -->
<sch:let name="ISM_CAPCO_RESOURCE"
value="index-of(tokenize(normalize-space(string($ISM_RESOURCE_ELEMENT/
@ism:ownerProducer))), ' '), 'USA') &gt; 0"/>

<!-- (U) ISM_ICDOCUMENT_APPLIES (boolean): True if the document claims compliance rules
for an IC Document, or if the
ism:compliesWith attribute of the ISM_RESOURCE_ELEMENT node contains [ICDocument]. False
otherwise. -->
<sch:let name="ISM_ICDOCUMENT_APPLIES"
value="index-of(tokenize(normalize-space(string($ISM_RESOURCE_ELEMENT/@ism:compliesWith))),
' '), 'ICDocument') &gt; 0"/>

<!-- (U) ISM_DOD5230_24_APPLIES (boolean): True if the document claims compliance with
DoD5230.24, or if the
ism:compliesWith attribute of the ISM_RESOURCE_ELEMENT node contains [DoD5230.24]. False
otherwise. -->
<sch:let name="ISM_DOD5230_24_APPLIES"
value="index-of(tokenize(normalize-space(string($ISM_RESOURCE_ELEMENT/@ism:compliesWith))),
' '), 'DoD5230.24') &gt; 0"/>

<!-- (U) ISM_ORCON_POC_DATE (date): The date after which a point-of-contact is required
for all documents containing ORCON data -->
<sch:let name="ISM_ORCON_POC_DATE" value="xs:date('2011-03-11')"/>

<!-- (U) Get Banner Attributes -->
<sch:let name="bannerClassification"
value="normalize-space(string($ISM_RESOURCE_ELEMENT/@ism:classification))"/>
<sch:let name="bannerDisseminationControls"
value="normalize-space(string($ISM_RESOURCE_ELEMENT/@ism:disseminationControls))"/>
<sch:let name="bannerDisplayOnlyTo"
value="normalize-space(string($ISM_RESOURCE_ELEMENT/@ism:displayOnlyTo))"/>
<sch:let name="bannerNonICmarkings"
value="normalize-space(string($ISM_RESOURCE_ELEMENT/@ism:nonICmarkings))"/>
<sch:let name="bannerFGISourceOpen"
value="normalize-space(string($ISM_RESOURCE_ELEMENT/@ism:FGISourceOpen))"/>
<sch:let name="bannerFGISourceProtected"
value="normalize-space(string($ISM_RESOURCE_ELEMENT/@ism:FGISourceProtected))"/>
<sch:let name="bannerReleasableTo"
value="normalize-space(string($ISM_RESOURCE_ELEMENT/@ism:releasableTo))"/>
<sch:let name="bannerSCIconcontrols"
value="normalize-space(string($ISM_RESOURCE_ELEMENT/@ism:SCIconcontrols))"/>
<sch:let name="bannerNotice"
value="normalize-space(string($ISM_RESOURCE_ELEMENT/@ism:noticeType))"/>
<sch:let name="bannerAtomicEnergyMarkings"

```

```

value="normalize-space(string($ISM_RESOURCE_ELEMENT/@ism:atomicEnergyMarkings))"/>

<!-- (U) Tokenize Banner Attributes -->
<sch:let name="bannerDisseminationControls_tok"
value="tokenize(normalize-space(string($bannerDisseminationControls)), ' ')" />
<sch:let name="bannerDisplayOnlyTo_tok"
value="tokenize(normalize-space(string($bannerDisplayOnlyTo)), ' ')" />
<sch:let name="bannerNonICmarkings_tok"
value="tokenize(normalize-space(string($bannerNonICmarkings)), ' ')" />
<sch:let name="bannerFGISourceOpen_tok"
value="tokenize(normalize-space(string($bannerFGISourceOpen)), ' ')" />
<sch:let name="bannerFGISourceProtected_tok"
value="tokenize(normalize-space(string($bannerFGISourceProtected)), ' ')" />
<sch:let name="bannerReleasableTo_tok"
value="tokenize(normalize-space(string($bannerReleasableTo)), ' ')" />
<sch:let name="bannerSCIcontrols_tok"
value="tokenize(normalize-space(string($bannerSCIcontrols)), ' ')" />
<sch:let name="bannerNotice_tok"
value="tokenize(normalize-space(string($bannerNotice)), ' ')" />
<sch:let name="bannerAtomicEnergyMarkings_tok"
value="tokenize(normalize-space(string($bannerAtomicEnergyMarkings)), ' ')" />

<!-- (U) Get all portions that meet ISM_CONTRIBUTES, or all non-resource nodes that do not
specify ism:excludeFromRollup='true' -->
<!-- (U) Similar portion classifications include ISM_CONTRIBUTES_CLASSIFIED, or all
portions meeting ISM_CONTRIBUTES that
also have an ism:classification attribute not equal to [U], and ISM_CONTRIBUTES_USA, or
all portions meeting ISM_CONTRIBUTES
that also have an ism:ownerProducer containing [USA]. -->
<sch:let name="partTags"
value="//*[@ism:* and not(@ism:excludeFromRollup=true()) and not(generate-id(.) =
generate-id($ISM_RESOURCE_ELEMENT))]" />

<!-- (U) Get Part Attributes -->
<sch:let name="partClassification"
value="for $token in $partTags/@ism:classification return tokenize(normalize-
space(string($token)), ' ')" />
<sch:let name="partDisseminationControls"
value="for $token in $partTags/@ism:disseminationControls return tokenize(normalize-
space(string($token)), ' ')" />
<sch:let name="partDisplayOnlyTo"
value="for $token in $partTags/@ism:displayOnlyTo return tokenize(normalize-
space(string($token)), ' ')" />
<sch:let name="partAtomicEnergyMarkings"
value="for $token in $partTags/@ism:atomicEnergyMarkings return tokenize(normalize-
space(string($token)), ' ')" />
<sch:let name="partNonICmarkings"

```

```

value="for $token in $partTags/@ism:nonICmarkings return tokenize(normalize-
space(string($token)), ' ')" />
<sch:let name="partFGISourceOpen"
value="for $token in $partTags/@ism:FGISourceOpen return tokenize(normalize-
space(string($token)), ' ')" />
<sch:let name="partFGISourceProtected"
value="for $token in $partTags/@ism:FGISourceProtected return tokenize(normalize-
space(string($token)), ' ')" />
<sch:let name="partSCIconcontrols"
value="for $token in $partTags/@ism:SCIconcontrols return tokenize(normalize-
space(string($token)), ' ')" />
<sch:let name="partNotice"
value="for $token in $partTags/@ism:noticeType return tokenize(normalize-
space(string($token)), ' ')" />
<sch:let name="partPocType"
value="for $token in $partTags/@ism:pocType return tokenize(normalize-
space(string($token)), ' ')" />

<!-- (U) Tokenize portion Attributes -->
<sch:let name="partClassification_tok"
value="for $token in $partClassification return tokenize(normalize-space(string($token)),
' ')" />
<sch:let name="partDisseminationControls_tok"
value="for $token in $partDisseminationControls return tokenize(normalize-
space(string($token)), ' ')" />
<sch:let name="partDisplayOnlyTo_tok"
value="for $token in $partDisplayOnlyTo return tokenize(normalize-space(string($token)), '
')" />
<sch:let name="partAtomicEnergyMarkings_tok"
value="for $token in $partAtomicEnergyMarkings return tokenize(normalize-
space(string($token)), ' ')" />
<sch:let name="partNonICmarkings_tok"
value="for $token in $partNonICmarkings return tokenize(normalize-space(string($token)), '
')" />
<sch:let name="partSCIconcontrols_tok"
value="for $token in $partSCIconcontrols return tokenize(normalize-space(string($token)), '
')" />
<sch:let name="partNotice_tok"
value="for $token in $partNotice return tokenize(normalize-space(string($token)), ' ')" />
<sch:let name="partPocType_tok"
value="for $token in $partPocType return tokenize(normalize-space(string($token)), ' ')" />

<!-- (U) ISM_NSI_EO_APPLIES (boolean): True if the current Classified National Security
Information Executive
Order applies to this resource. This will be false if any of the following conditions are
true:
* The document is not a ISM_CAPCO_RESOURCE
OR

```

```

* The ISM_RESOURCE_ELEMENT node has attribute ism:classification with a value of [U]
OR
* The ISM_RESOURCE_CREATE_DATE is before 11 April 1996
OR
* Every element meeting ISM_CONTRIBUTES_CLASSIFIED also has attribute
ism:disseminationControls containing [FRD] or [RD]
-->
<sch:let name="ISM_NSI_EO_APPLIES"
value="if(not($ISM_CAPCO_RESOURCE) or ($ISM_RESOURCE_ELEMENT/@ism:classification='U')
or index-of($partAtomicEnergyMarkings_tok,'FRD')>0 or index-
of($partAtomicEnergyMarkings_tok,'RD')>0 or ($ISM_RESOURCE_CREATE_DATE and
$ISM_RESOURCE_CREATE_DATE < xs:date('1996-04-14')) then false() else true()"/>

<!-- (U) Define countries that are included in the THREE-, FOUR- and FIVE-EYES
designations -->
<sch:let name="TEYE_tok" value="tokenize(string('USA CAN GBR'), ' ' )"/>
<sch:let name="ACGU_tok" value="tokenize(string('USA AUS CAN GBR'), ' ' )"/>
<sch:let name="FVEY_tok" value="tokenize(string('USA AUS CAN GBR NZL'), ' ' )"/>

<!-- (U) Check roll-up of attributes in portions to the banner -->
<sch:let name="dcTags"
value="for $piece in $disseminationControlsList return $piece/text()"/>
<sch:let name="dcTagsFound"
value="for $token in $dcTags return if ( index-of($partDisseminationControls_tok,$token)
&gt; 0 and (not(index-of($bannerDisseminationControls_tok,$token) &gt; 0))) then $token
else null"/>
<sch:let name="aeaTags"
value="for $piece in $atomicEnergyMarkingsList return $piece/text()"/>
<sch:let name="aeaTagsFound"
value="for $token in $aeaTags return if ( index-of($partAtomicEnergyMarkings_tok,$token)
&gt; 0 and (not(index-of($bannerAtomicEnergyMarkings_tok,$token) &gt; 0))) then $token
else null"/>

<!-- XSLT Functions for routine tasks -->

<!-- Evaluate the attribute value tokens to determine whether any values
have been deprecated by comparing deprecation dates against $curDate.
If the value is deprecated, return either an ERROR or WARNING message,
depending on the isError flag. -->
<xsl:function name="dvf:deprecated" as="xs:string*">
<xsl:param name="attribute" as="xs:string"/>
<xsl:param name="depTerms" as="element()*"/>
<xsl:param name="curDate" as="xs:date?"/>
<xsl:param name="isError" as="xs:boolean"/>
<!--$curDate param is optional in order to prevent compiled XSLT from breaking
when otherwise invalid documents are executed against the rules
(e.g. missing @ism:createDate).

```



```

Only execute if we have a date to compare against. -->
<xsl:if test="count($curDate)=1">
<xsl:for-each select="$depTerms[cve:Value=tokenize($attribute,' ')]">
<xsl:if test="($isError and $curDate gt xs:date(@deprecated)) or (not($isError) and
$curDate le xs:date(@deprecated))">
<xsl:sequence select="concat('[',string(current()/cve:Value),'] has been deprecated and is
not authorized for use after ', current()/@deprecated)"/>
</xsl:if>
</xsl:for-each>
</xsl:if>
</xsl:function>

<!--*****-->
<!-- (U) ISM ID Rules -->
<!--*****-->

<!--(U) atomicEnergyMarkings-->
<sch:include href="./Rules/atomicEnergyMarkings/ISM_ID_00173.sch"/>
<sch:include href="./Rules/atomicEnergyMarkings/ISM_ID_00174.sch"/>
<sch:include href="./Rules/atomicEnergyMarkings/ISM_ID_00175.sch"/>
<sch:include href="./Rules/atomicEnergyMarkings/ISM_ID_00176.sch"/>
<sch:include href="./Rules/atomicEnergyMarkings/ISM_ID_00178.sch"/>
<sch:include href="./Rules/atomicEnergyMarkings/ISM_ID_00181.sch"/>
<sch:include href="./Rules/atomicEnergyMarkings/ISM_ID_00182.sch"/>
<sch:include href="./Rules/atomicEnergyMarkings/ISM_ID_00183.sch"/>
<sch:include href="./Rules/atomicEnergyMarkings/ISM_ID_00184.sch"/>
<sch:include href="./Rules/atomicEnergyMarkings/ISM_ID_00185.sch"/>

<!--(U) classification-->
<sch:include href="./Rules/classification/ISM_ID_00015.sch"/>
<sch:include href="./Rules/classification/ISM_ID_00016.sch"/>
<sch:include href="./Rules/classification/ISM_ID_00040.sch"/>
<sch:include href="./Rules/classification/ISM_ID_00142.sch"/>

<!--(U) classifiedBy-->
<sch:include href="./Rules/classifiedBy/ISM_ID_00017.sch"/>

<!--(U) compliesWith-->
<sch:include href="./Rules/compliesWith/ISM_ID_00222.sch"/>

<!--(U) declassException-->
<sch:include href="./Rules/declassException/ISM_ID_00133.sch"/>

<!--(U) derivativelyClassifiedBy-->
<sch:include href="./Rules/derivativelyClassifiedBy/ISM_ID_00143.sch"/>
<sch:include href="./Rules/derivativelyClassifiedBy/ISM_ID_00221.sch"/>

<!--(U) displayOnlyTo-->

```

```
<sch:include href="./Rules/displayOnlyTo/ISM_ID_00167.sch"/>
<sch:include href="./Rules/displayOnlyTo/ISM_ID_00168.sch"/>

<!--(U) disseminationControls-->
<sch:include href="./Rules/disseminationControls/ISM_ID_00026.sch"/>
<sch:include href="./Rules/disseminationControls/ISM_ID_00028.sch"/>
<sch:include href="./Rules/disseminationControls/ISM_ID_00030.sch"/>
<sch:include href="./Rules/disseminationControls/ISM_ID_00031.sch"/>
<sch:include href="./Rules/disseminationControls/ISM_ID_00033.sch"/>
<sch:include href="./Rules/disseminationControls/ISM_ID_00034.sch"/>
<sch:include href="./Rules/disseminationControls/ISM_ID_00094.sch"/>
<sch:include href="./Rules/disseminationControls/ISM_ID_00107.sch"/>
<sch:include href="./Rules/disseminationControls/ISM_ID_00124.sch"/>
<sch:include href="./Rules/disseminationControls/ISM_ID_00140.sch"/>
<sch:include href="./Rules/disseminationControls/ISM_ID_00164.sch"/>
<sch:include href="./Rules/disseminationControls/ISM_ID_00169.sch"/>
<sch:include href="./Rules/disseminationControls/ISM_ID_00213.sch"/>
<sch:include href="./Rules/disseminationControls/ISM_ID_00215.sch"/>

<!--(U) FGISourceOpen-->
<sch:include href="./Rules/FGISourceOpen/ISM_ID_00095.sch"/>

<!--(U) FGISourceProtected-->
<sch:include href="./Rules/FGISourceProtected/ISM_ID_00096.sch"/>
<sch:include href="./Rules/FGISourceProtected/ISM_ID_00097.sch"/>
<sch:include href="./Rules/FGISourceProtected/ISM_ID_00217.sch"/>

<!--(U) generalConstraints-->
<sch:include href="./Rules/generalConstraints/ISM_ID_00002.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00012.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00102.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00103.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00119.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00125.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00126.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00166.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00170.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00179.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00180.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00188.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00189.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00190.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00191.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00192.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00193.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00194.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00195.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00196.sch"/>
```

```
<sch:include href="./Rules/generalConstraints/ISM_ID_00197.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00198.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00199.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00200.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00201.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00202.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00203.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00204.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00205.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00206.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00207.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00208.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00209.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00210.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00211.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00223.sch"/>
<sch:include href="./Rules/generalConstraints/ISM_ID_00226.sch"/>

<!--(U) nonICmarkings-->
<sch:include href="./Rules/nonICmarkings/ISM_ID_00035.sch"/>
<sch:include href="./Rules/nonICmarkings/ISM_ID_00036.sch"/>
<sch:include href="./Rules/nonICmarkings/ISM_ID_00037.sch"/>
<sch:include href="./Rules/nonICmarkings/ISM_ID_00038.sch"/>
<sch:include href="./Rules/nonICmarkings/ISM_ID_00148.sch"/>
<sch:include href="./Rules/nonICmarkings/ISM_ID_00225.sch"/>

<!--(U) nonUSControls-->
<sch:include href="./Rules/nonUSControls/ISM_ID_00163.sch"/>

<!--(U) notice-->
<sch:include href="./Rules/notice/ISM_ID_00127.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00128.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00129.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00130.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00134.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00135.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00136.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00137.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00138.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00139.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00150.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00151.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00152.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00153.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00156.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00157.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00158.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00159.sch"/>
```

```
<sch:include href="./Rules/notice/ISM_ID_00160.sch"/>
<sch:include href="./Rules/notice/ISM_ID_00161.sch"/>

<!--(U) ownerProducer-->
<sch:include href="./Rules/ownerProducer/ISM_ID_00001.sch"/>
<sch:include href="./Rules/ownerProducer/ISM_ID_00099.sch"/>
<sch:include href="./Rules/ownerProducer/ISM_ID_00100.sch"/>
<sch:include href="./Rules/ownerProducer/ISM_ID_00219.sch"/>

<!--(U) pocType-->
<sch:include href="./Rules/pocType/ISM_ID_00224.sch"/>

<!--(U) releasableTo-->
<sch:include href="./Rules/releasableTo/ISM_ID_00032.sch"/>
<sch:include href="./Rules/releasableTo/ISM_ID_00041.sch"/>
<sch:include href="./Rules/releasableTo/ISM_ID_00214.sch"/>

<!--(U) resourceElement-->
<sch:include href="./Rules/resourceElement/ISM_ID_00013.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00014.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00056.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00057.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00058.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00059.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00060.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00061.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00062.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00063.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00064.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00065.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00066.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00067.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00068.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00070.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00071.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00072.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00073.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00074.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00075.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00077.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00078.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00079.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00080.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00081.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00082.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00083.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00084.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00085.sch"/>
```

```
<sch:include href="./Rules/resourceElement/ISM_ID_00086.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00087.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00088.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00090.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00104.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00105.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00108.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00109.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00110.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00111.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00112.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00113.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00116.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00118.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00132.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00141.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00145.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00146.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00147.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00149.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00154.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00155.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00162.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00165.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00171.sch"/>
<sch:include href="./Rules/resourceElement/ISM_ID_00227.sch"/>

<!--(U) SARIdentifier-->
<sch:include href="./Rules/SARIdentifier/ISM_ID_00121.sch"/>

<!--(U) SCIcontrols-->
<sch:include href="./Rules/SCIcontrols/ISM_ID_00042.sch"/>
<sch:include href="./Rules/SCIcontrols/ISM_ID_00043.sch"/>
<sch:include href="./Rules/SCIcontrols/ISM_ID_00044.sch"/>
<sch:include href="./Rules/SCIcontrols/ISM_ID_00045.sch"/>
<sch:include href="./Rules/SCIcontrols/ISM_ID_00046.sch"/>
<sch:include href="./Rules/SCIcontrols/ISM_ID_00047.sch"/>
<sch:include href="./Rules/SCIcontrols/ISM_ID_00048.sch"/>
<sch:include href="./Rules/SCIcontrols/ISM_ID_00049.sch"/>
<sch:include href="./Rules/SCIcontrols/ISM_ID_00122.sch"/>
<sch:include href="./Rules/SCIcontrols/ISM_ID_00123.sch"/>
<sch:include href="./Rules/SCIcontrols/ISM_ID_00177.sch"/>
<sch:include href="./Rules/SCIcontrols/ISM_ID_00186.sch"/>
<sch:include href="./Rules/SCIcontrols/ISM_ID_00187.sch"/>
</sch:schema>
<!-- UNCLASSIFIED -->
```

## Rule: ISM-ID-00001

FileName:../Rules/ownerProducer/ISM\_ID\_00001.sch

### Rule Description:

[ISM-ID-00001][Error] The attribute ownerProducer, when it exists, must have a non-null value.

### Code Description:

This code makes sure that if ownerProducer is specified that it contains content that is a non-whitespace value.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00001">

<sch:rule context="*[@ism:ownerProducer]">
<sch:assert id="ISM-00001" test="normalize-space(string(..@ism:ownerProducer))!='' "
flag="error">
[ISM-ID-00001][Error] The attribute ownerProducer, when it exists, must have
a non-null value.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00002

FileName:../Rules/generalConstraints/ISM\_ID\_00002.sch

### Rule Description:

[ISM-ID-00002][Error] For every optional attribute that is used in a document a non-null value must be present.

### Code Description:

This code checks that if an attribute is present and has an empty or whitespace string as a value, then we return false since all attributes must have a value specified.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00002">

<sch:rule context="*[@ism:*]">
<sch:assert id="ISM-00002"
test=" every $attribute in ./@ism:* satisfies not(normalize-space(string($attribute)) =
'' ) "
flag="error">
[ISM-ID-00002][Error] For every optional attribute that is used in a document a non-null
value must be present.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00012

FileName:./Rules/generalConstraints/ISM\_ID\_00012.sch

### Rule Description:

[ISM-ID-00012][Error] If any of the attributes defined in this DES other than DESVersion, unregisteredNoticeType, or pocType are specified for an element, then attributes classification and ownerProducer must be specified for the element.

### Code Description:

This code triggers on elements that have an ISM attribute whose name is not 'DESVersion' and it ensures that both the ownerProducer and classification attributes are present on the element.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00012">

<sch:rule context="*[@ism:* except (@ism:pocType | @ism:DESVersion |
@ism:unregisteredNoticeType)]">
<sch:assert id="ISM-00012"
test=" (./@ism:ownerProducer and ./@ism:classification) "
flag="error">
[ISM-ID-00012][Error] If any of the attributes defined in
this DES other than DESVersion, unregisteredNoticeType, or pocType are specified for an
element,
then attributes classification and ownerProducer must be specified for the element.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00013

FileName:../Rules/resourceElement/ISM\_ID\_00013.sch

### Rule Description:

[ISM-ID-00013][Error] If ISM\_NSI\_EO\_APPLIES then either attribute classifiedBy or derivedFrom must be specified on the ISM\_RESOURCE\_ELEMENT. Human Readable: Documents under E.O. 13526 must have classification authority block information.

### Code Description:

If the current Classified National Security Information Executive Order does not apply to the document then the rule does not apply and we return true. Otherwise, we make sure that the resourceElement has attribute classifiedBy or derivedFrom specified.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00013">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00013"
test=" if(not($ISM_NSI_EO_APPLIES)) then true() else if(./@ism:classifiedBy or ./
@ism:derivedFrom) then true() else false() "
flag="error">
[ISM-ID-00013][Error] Documents under E.O. 13526 must have classification authority block
information.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00014

FileName:./Rules/resourceElement/ISM\_ID\_00014.sch

### Rule Description:

[ISM-ID-00014][Error] If ISM\_NSI\_EO\_APPLIES then one or more of the following attributes: declassDate, declassEvent, or declassException must be specified on the ISM\_RESOURCE\_ELEMENT. Human Readable: Documents under E.O. 13526 must have declassification instructions included in the classification authority block information.

### Code Description:

If the current Classified National Security Information Executive Order does not apply to the document then the rule does not apply and we return true. Otherwise, we make sure that the resourceElement has attribute declassDate, declassEvent, or declassException specified.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00014">

<sch:rule context="//*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00014"
test=" if(not($ISM_NSI_EO_APPLIES)) then true() else if(./@ism:declassDate or ./
@ism:declassEvent or ./@ism:declassException) then true() else false() "
flag="error">
[ISM-ID-00014][Error] If ISM_NSI_EO_APPLIES then one or more of the following
attributes: declassDate, declassEvent, or declassException must be specified on the
ISM_RESOURCE_ELEMENT.

Human Readable: Documents under E.O. 13526 must have declassification instructions
included in the
classification authority block information.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00015

FileName:../Rules/classification/ISM\_ID\_00015.sch

### Rule Description:

[ISM-ID-00015][Error] If ISM\_CAPCO\_RESOURCE and attribute classification has a value of [U], then attributes releasableTo, SARIdentifier, and SCIcontrols must not be specified.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute classification with a value of [U] then we do not have attributes releasableTo, SARIdentifier, or SCIcontrols on the same element.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00015">

<sch:rule context="*[@ism:classification and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00015"
test=" if(./@ism:classification='U' and (./@ism:releasableTo or ./@ism:SARIdentifier or ./
@ism:SCIcontrols)) then false() else true() "
flag="error">
[ISM-ID-00015][Error] If ISM_CAPCO_RESOURCE and attribute
classification has a value of [U], then attributes releasableTo, SARIdentifier, and
SCIcontrols
must not be specified.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00016

FileName:../Rules/classification/ISM\_ID\_00016.sch

### Rule Description:

[ISM-ID-00016][Error] If ISM\_CAPCO\_RESOURCE and attribute classification has a value of [U], then attributes classificationReason, classifiedBy, derivativelyClassifiedBy, declassDate, declassEvent, declassException and derivedFrom must not be specified.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute classification with a value of [U] then we do not have attributes classifiedBy, declassDate, declassEvent, declassException, derivativelyClassifiedBy, or derivedFrom on the same element.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00016">

<sch:rule context="*[@ism:classification and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00016"
test=" if(matches(./@ism:classification,'^U$') and (./@ism:classificationReason or ./
@ism:classifiedBy or ./@ism:declassDate or ./@ism:declassEvent or ./@ism:declassException
or ./@ism:derivativelyClassifiedBy or ./@ism:derivedFrom)) then false() else true() "
flag="error">
[ISM-ID-00016][Error] If ISM_CAPCO_RESOURCE and attribute
classification has a value of [U], then attributes classificationReason, classifiedBy,
derivativelyClassifiedBy, declassDate, declassEvent, declassException,
and derivedFrom must not be specified.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00017

FileName:./Rules/classifiedBy/ISM\_ID\_00017.sch

### Rule Description:

[ISM-ID-00017][Error] If ISM\_NSI\_EO\_APPLIES and attribute classifiedBy is specified, then attribute classificationReason must be specified. Human Readable: Documents under E.O. 13526 containing Originally Classified data require a classification reason be identified.

### Code Description:

If current Classified National Security Information Executive Order does not apply to this resource then the rule does not apply and we return true. Otherwise we ensure that any element with the attribute classifiedBy also has the attribute classificationReason.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00017">

<sch:rule context="*[@ism:classifiedBy and $ISM_NSI_EO_APPLIES]">
<sch:assert id="ISM-00017" test="./@ism:classificationReason" flag="error">
[ISM-ID-00017][Error] If ISM_NSI_EO_APPLIES and attribute
classifiedBy is specified, then attribute classificationReason must be specified.

Human Readable: Documents under E.O. 13526 containing Originally Classified data
require a classification reason be identified.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00026

FileName:./Rules/disseminationControls/ISM\_ID\_00026.sch

### Rule Description:

[ISM-ID-00026][Error] If ISM\_CAPCO\_RESOURCE and attribute disseminationControls is specified, then each of its values must be ordered in accordance with CVEnumISMDissem.xml.

### Code Description:

To perform sorting, this rule first retrieves the CVE values for the attribute to be sorted, which in this case is disseminationControls. Then, each attribute token is converted into a numerical value based on its characters. Next, each attribute token is given an order number, which compares its position to that of its value in the CVE file. Next, each order number is compared to that of its previous sibling to determine if the tokens are in order. If a token is found whose order number is less than that of its previous sibling, 0 is returned for its sorted order number. If a token's order number is greater than that of its previous sibling, 1 is returned. If two tokens have the same order number, their original attribute values are compared. If the original attribute value contains numbers then the comparison is made on its converted numerical value; otherwise, the comparison is made on its string value. If an attribute value is found whose value is less than that of its previous sibling, 0 is returned for its sorted order number; otherwise 2 is returned. Finally, if any tokens are found with 0 as its sorted order number, then the rule fails as those tokens are out of order.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00026">

  <sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
    <!-- Define variables -->
    <sch:let name="capcoRestriction" value="true()"/>
    <sch:let name="errMsg_AlphabeticalOrder"
value=" '[ISM-ID-00026][Error] If ISM_CAPCO_RESOURCE and attribute disseminationControls
is specified, then each of its values must be ordered in accordance with
CVEnumISMDissem.xml.' "/>

    <sch:let name="dataFileElems" value="$disseminationControlsList"/>
    <sch:let name="attrValues" value="./@ism:disseminationControls"/>
    <sch:let name="attrValueTokens" value="tokenize(string($attrValues),' ')/>

    <!-- Convert each character to a numerical value, then concatenate the results to form a
number-string -->
    <sch:let name="convertStrToNum"
value=" for $token in $attrValueTokens return number(string-join( for $index in 1
to string-length($token) return for $char in substring($token, $index, 1) return if
(contains(string('0123456789'), $char)) then $char else if (contains(string('ABCDEFGHI'),
$char)) then translate(string($char), 'ABCDEFGHI', '123456789') else if
(contains(string('JKLMNOPQRS'), $char)) then concat('1', translate(string($char),
'JKLMNOPQRS', '0123456789')) else if (contains(string('TUVWXYZ'), $char)) then concat('2',
translate(string($char), 'TUVWXYZ', '0123456')) else '0' , '')) "/>
```

```

<!-- Get the position of each client node relative to its position in the master list. If
the node is not found, return a -1 -->
<sch:let name="orderNums"
value=" for $token in $attrValueTokens return if
($dataFileElems[matches($token,concat('^',text(),'$'))]) then
count(($dataFileElems[matches($token,concat('^',text(),'$'))])/preceding::* ) + 1 else -1"/
>

<!-- Create a sequence that returns a 0 if the previous sibling has a higher order number,
else return a 1 -->
<sch:let name="sortedOrderNums"
value=" for $index in distinct-values(for $token in $orderNums return index-of($orderNums,
$token)) return if($index = 1 or $orderNums[$index] > $orderNums[$index - 1])
then 1 else if ($orderNums[$index] < $orderNums[$index - 1]) then 0 else if
(matches($attrValueTokens[$index], '[0-9]') or matches($attrValueTokens[$index - 1],
'[0-9]')) then if ($convertStrToNum[$index - 1] > $convertStrToNum[$index]) then 0 else
2 else if (compare($attrValueTokens[$index - 1],$attrValueTokens[$index])=1) then 0 else 2
"/>
<sch:let name="hasUnsorted" value="count(index-of($sortedOrderNums,0)) > 0"/>
<sch:let name="unsortedValues"
value=" if ($hasUnsorted) then distinct-values( for $token in index-of($sortedOrderNums,0)
return $attrValueTokens[$token] ) else null "/>

<sch:assert id="ISM-00026" test="not($hasUnsorted)" flag="error">
<sch:value-of select="$errMsg_AlphabeticalOrder"/>
The following values are out of order [<sch:value-of select="$unsortedValues"/>] for
[<sch:value-of select="$attrValueTokens"/>] </sch:assert>
</sch:rule>
</sch:pattern>

```

## Rule: ISM-ID-00028

FileName:./Rules/disseminationControls/ISM\_ID\_00028.sch

### Rule Description:

[ISM-ID-00028][Error] If ISM\_CAPCO\_RESOURCE and attribute disseminationControls contains the name token [OC], [EYES], or [RELIDO], then attribute classification must have a value of [TS], [S], or [C]. Human Readable: Portions marked for ORCON, EYES ONLY, or RELIDO dissemination in a USA document must be CLASSIFIED, SECRET, or TOP SECRET.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute disseminationControls with a value of [OC], [EYES], or [RELIDO] then we also have the attribute classification specified with a value of [C], [S], or [TS] on the same element.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00028">

<sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00028"
test=" count(for $token in tokenize(string(./@ism:disseminationControls), ' ') return
if(matches($token, '^(OC|EYES|RELIDO)$') and not( matches(./@ism:classification, '^(TS|S|
C)$')) then $token else null )=0 "
flag="error">
[ISM-ID-00028][Error] If ISM_CAPCO_RESOURCE and attribute
disseminationControls contains the name token [OC], [EYES], or [RELIDO],
then attribute classification must have a value of [TS], [S], or [C].

Human Readable: Portions marked for ORCON, EYES ONLY, or RELIDO dissemination
in a USA document must be CLASSIFIED, SECRET, or TOP SECRET.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00030

FileName:./Rules/disseminationControls/ISM\_ID\_00030.sch

### Rule Description:

[ISM-ID-00030][Error] If ISM\_CAPCO\_RESOURCE and attribute disseminationControls contains the name token [FOUO], then attribute classification must have a value of [U]. Human Readable: Portions marked for FOUO dissemination in a USA document must be classified UNCLASSIFIED.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute disseminationControls without a value of [FOUO] then we return true because the rule does not apply. Otherwise we make sure the attribute classification is specified with a value of [U] on the same element.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00030">

<sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00030"
test=" if(index-of(tokenize(string(@ism:disseminationControls),' '), 'FOUO')>0) then
@ism:classification='U' else true() "
flag="error">
[ISM-ID-00030][Error] If ISM_CAPCO_RESOURCE and attribute
disseminationControls contains the name token [FOUO], then attribute classification must
have
a value of [U].

Human Readable: Portions marked for FOUO dissemination in a USA document must be
classified UNCLASSIFIED.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00031

FileName:./Rules/disseminationControls/ISM\_ID\_00031.sch

### Rule Description:

[ISM-ID-00031][Error] If ISM\_CAPCO\_RESOURCE and attribute disseminationControls contains the name token [REL] or [EYES], then attribute releasableTo must be specified. Human Readable: USA documents containing REL TO or EYES ONLY dissemination must specify to which countries the document is releasable.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute disseminationControls with a value of [REL] or [EYES] then the attribute releasableTo is specified and does not have an empty value set.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00031">

<sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00031"
test=" if(matches(concat(' ',string-join(string(@ism:disseminationControls),' '), ' '),
' (REL|EYES) ')) then @ism:releasableTo else true() "
flag="error">
[ISM-ID-00031][Error] If ISM_CAPCO_RESOURCE and attribute
disseminationControls contains the name token [REL] or [EYES], then attribute releasableTo
must be specified.

Human Readable: USA documents containing REL TO or EYES ONLY dissemination must
specify to which countries the document is releasable.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00032

FileName:./Rules/releasableTo/ISM\_ID\_00032.sch

### Rule Description:

[ISM-ID-00032][Error] If ISM\_CAPCO\_RESOURCE and attribute disseminationControls is not specified, or is specified and does not contain the name token [REL] or [EYES], then attribute releasableTo must not be specified. Human Readable: USA documents must only specify to which countries it is authorized for release if dissemination information contains REL TO or EYES ONLY data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the attribute releasableTo is specified, then we make sure that the attribute disseminationControls is specified with a value containing [EYES] or [REL].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00032">

<sch:rule context="*[@ism:releasableTo and $ISM_CAPCO_RESOURCE]">

<sch:let name="dissemTok" value="tokenize(string(./@ism:disseminationControls),' ')" />
<sch:assert id="ISM-00032"
test=" index-of($dissemTok,'EYES')>0 or index-of($dissemTok,'REL')>0 "
flag="error">
[ISM-ID-00032][Error] If ISM_CAPCO_RESOURCE and attribute
disseminationControls is not specified, or is specified and does not contain the name
token
[REL] or [EYES], then attribute releasableTo must not be specified.

Human Readable: USA documents must only specify to which countries it is
authorized for release if dissemination information contains REL TO or EYES ONLY data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00033

FileName:../Rules/disseminationControls/ISM\_ID\_00033.sch

### Rule Description:

[ISM-ID-00033][Error] If ISM\_CAPCO\_RESOURCE, then tokens [REL], [EYES] and [NF] are mutually exclusive for attribute disseminationControls.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute disseminationControls and counting 1 for each value of [REL], [NF] or [EYES] found. If the count is greater than one then the values are not being used exclusively with respect to each other.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00033">

<sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
<sch:let name="dissemTok" value="tokenize(string(./@ism:disseminationControls),' ')/>
<sch:assert id="ISM-00033"
test=" not( count(( if(index-of($dissemTok,'REL')>0) then 1 else null, if(index-
of($dissemTok,'NF')>0) then 1 else null, if(index-of($dissemTok, 'EYES')>0) then 1
else null) ) > 1 ) "
flag="error">
[ISM-ID-00033][Error] If ISM_CAPCO_RESOURCE, then
tokens [REL], [EYES] and [NF] are mutually exclusive for attribute disseminationControls.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00034

FileName:../Rules/disseminationControls/ISM\_ID\_00034.sch

### Rule Description:

[ISM-ID-00034][Error] If ISM\_CAPCO\_RESOURCE, then tokens "RELIDO" and "NF" are mutually exclusive for attribute disseminationControls.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute disseminationControls that does not have values [RELIDO] and [NF] at the same time.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00034">

<sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
<sch:let name="dissemTok" value="tokenize(string(./@ism:disseminationControls),' ')" />
<sch:assert id="ISM-00034"
test="not(index-of($dissemTok,'RELIDO')>0 and index-of($dissemTok,'NF')>0)"
flag="error">
[ISM-ID-00034][Error] If ISM_CAPCO_RESOURCE, then
tokens "RELIDO" and "NF" are mutually exclusive for attribute disseminationControls.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00035

FileName:./Rules/nonICmarkings/ISM\_ID\_00035.sch

### Rule Description:

[ISM-ID-00035][Error] If ISM\_CAPCO\_RESOURCE and attribute nonICmarkings is specified, then each of its values must be ordered in accordance with CVEnumISMNonIC.xml.

### Code Description:

To perform sorting, this rule first retrieves the CVE values for the attribute to be sorted, which in this case is nonICmarkings. Then, each attribute token is converted into a numerical value based on its characters. Next, each attribute token is given an order number, which compares its position to that of its value in the CVE file. Next, each order number is compared to that of its previous sibling to determine if the tokens are in order. If a token is found whose order number is less than that of its previous sibling, 0 is returned for its sorted order number. If a token's order number is greater than that of its previous sibling, 1 is returned. If two tokens have the same order number, their original attribute values are compared. If the original attribute value contains numbers then the comparison is made on its converted numerical value; otherwise, the comparison is made on its string value. If an attribute value is found whose value is less than that of its previous sibling, 0 is returned for its sorted order number; otherwise 2 is returned. Finally, if any tokens are found with 0 as its sorted order number, then the rule fails as those tokens are out of order.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00035">

  <sch:rule context="*[@ism:nonICmarkings]">
    <!-- Define variables -->
    <sch:let name="errMsg_AlphabeticalOrder"
value="' [ISM-ID-00035][Error] If ISM_CAPCO_RESOURCE and attribute nonICmarkings is
specified, then each of its values must be ordered in accordance with CVEnumISMNonIC.xml.
'"/>

    <sch:let name="dataFileElems" value="$nonICmarkingsList"/>
    <sch:let name="attrValues" value="./@ism:nonICmarkings"/>
    <sch:let name="attrValueTokens" value="tokenize(string($attrValues),' ')/>

    <!-- Convert each character to a numerical value, then concatenate the results to form a
number-string -->
    <sch:let name="convertStrToNum"
value=" for $token in $attrValueTokens return number(string-join( for $index in 1
to string-length($token) return for $char in substring($token, $index, 1) return if
(contains(string('0123456789'), $char)) then $char else if (contains(string('ABCDEFGHI'),
$char)) then translate(string($char), 'ABCDEFGHI', '123456789') else if
(contains(string('JKLMNOPQRS'), $char)) then concat('1', translate(string($char),
'JKLMNOPQRS', '0123456789')) else if (contains(string('TUVWXYZ'), $char)) then concat('2',
translate(string($char), 'TUVWXYZ', '0123456')) else '0' , '')) "/>
```

```

<!-- Get the position of each client node relative to its position in the master list. If
the node is not found, return a -1 -->
<sch:let name="orderNums"
value=" for $token in $attrValueTokens return if
($dataFileElems[matches($token,concat('^',text(),'$'))]) then
count(($dataFileElems[matches($token,concat('^',text(),'$'))])/preceding::*) + 1 else -1"/
>

<!-- Create a sequence that returns a 0 if the previous sibling has a higher order number,
else return a 1 -->
<sch:let name="sortedOrderNums"
value=" for $index in distinct-values(for $token in $orderNums return index-of($orderNums,
$token)) return if($index = 1 or $orderNums[$index] > $orderNums[$index - 1])
then 1 else if ($orderNums[$index] < $orderNums[$index - 1]) then 0 else if
(matches($attrValueTokens[$index], '[0-9]') or matches($attrValueTokens[$index - 1],
'[0-9]')) then if ($convertStrToNum[$index - 1] > $convertStrToNum[$index]) then 0 else
2 else if (compare($attrValueTokens[$index - 1],$attrValueTokens[$index])=1) then 0 else 2
"/>
<sch:let name="hasUnsorted" value="count(index-of($sortedOrderNums,0)) > 0"/>
<sch:let name="unsortedValues"
value=" if ($hasUnsorted) then distinct-values( for $token in index-of($sortedOrderNums,0)
return $attrValueTokens[$token] ) else null "/>

<sch:assert id="ISM-00035" test="not($hasUnsorted)" flag="error">
<sch:value-of select="$errMsg_AlphabeticalOrder"/>
The following values are out of order [<sch:value-of select="$unsortedValues"/>] for
[<sch:value-of select="$attrValueTokens"/>] </sch:assert>
</sch:rule>

</sch:pattern>

```

## Rule: ISM-ID-00036

FileName:./Rules/nonICmarkings/ISM\_ID\_00036.sch

### Rule Description:

[ISM-ID-00036][Error] If ISM\_CAPCO\_RESOURCE and attribute nonICmarkings contains the name token [SC], then attribute classification must have a value of [TS], [S], or [C]. Human Readable: SC data must be marked CONFIDENTIAL, SECRET or TOP SECRET in USA documents.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check if the attribute disseminationControls contains the value [SC] and if it does we check that the classification attribute has a value of [C], [S], or [TS].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00036">

<sch:rule context="*[@ism:nonICmarkings]">
<sch:assert id="ISM-00036"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of(tokenize(string(./
@ism:nonICmarkings), ' '), 'SC')>0) then matches(./@ism:classification, '^(TS|S|C)$')
else true() "
flag="error">
[ISM-ID-00036][Error] If ISM_CAPCO_RESOURCE and attribute nonICmarkings
contains the name token [SC], then attribute classification must have a
value of [TS], [S], or [C].

Human Readable: SC data must be marked CONFIDENTIAL, SECRET or TOP SECRET in USA
documents.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00037

FileName:./Rules/nonICmarkings/ISM\_ID\_00037.sch

### Rule Description:

[ISM-ID-00037][Error] If ISM\_CAPCO\_RESOURCE and attribute nonICmarkings contains the name token [SINFO], [SBU], or [SBU-NF], then attribute classification must have a value of [U]. Human Readable: SINFO, SBU, and SBU-NF data must be marked UNCLASSIFIED in USA documents.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check if the attribute nonICmarkings contains a value of [SINFO], [SBU], or [SBU-NF] then the classification attribute must have a value of [U].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00037">

<sch:rule context="*[@ism:nonICmarkings]">
<sch:let name="nonICTok" value="tokenize(string(./@ism:nonICmarkings),' ')" />
<sch:assert id="ISM-00037"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($nonICTok, 'SINFO')>0
or index-of($nonICTok, 'SBU')>0 or index-of($nonICTok, 'SBU-NF')>0) then ./
@ism:classification='U' else true() "
flag="error">
[ISM-ID-00037][Error] If ISM_CAPCO_RESOURCE and attribute nonICmarkings contains
the name token [SINFO], [SBU], or [SBU-NF], then attribute classification must have a
value of [U].

Human Readable: SINFO, SBU, and SBU-NF data must be marked UNCLASSIFIED in USA documents.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00038

FileName:./Rules/nonICmarkings/ISM\_ID\_00038.sch

### Rule Description:

[ISM-ID-00038][Error] If ISM\_CAPCO\_RESOURCE, then Name tokens [XD] and [ND] are mutually exclusive for attribute nonICmarkings. Human Readable: USA documents must not specify both XD and ND on a single element.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that the attribute nonICmarkings does not contain both a value of [XD] and a value of [ND].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00038">

<sch:rule context="*[@ism:nonICmarkings]">
<!-- get list of tokens in nonICmarkings attribute -->
<sch:let name="nicmTok" value="tokenize(string(./@ism:nonICmarkings),' ')" />

<sch:assert id="ISM-00038"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not(index-of($nicmTok,'XD')>0 and
index-of($nicmTok,'ND')>0) "
flag="error">
[ISM-ID-00038][Error] If ISM_CAPCO_RESOURCE, then Name tokens [XD] and [ND] are mutually
exclusive for attribute nonICmarkings.

Human Readable: USA documents must not specify both XD and ND on a single element.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00040

FileName:./Rules/classification/ISM\_ID\_00040.sch

### Rule Description:

[ISM-ID-00040][Error] If ISM\_CAPCO\_RESOURCE and attribute ownerProducer contains [USA] then attribute classification must have a value in CVEnumISMClassificationUS.xml.

### Code Description:

To determine the valid values, this rule first retrieves the CVE values for the attribute, which in this case is classification. Then, each attribute token is converted into a numerical value based on its characters. Next, each attribute token is given an order number, which compares its position to that of its value in the CVE file. If the token is not found, its order number will be -1. If the document is a CAPCO resource and the ownerProducer of this element is 'USA', then the rule will fail if tokens are found with order numbers of -1. The rule will also fail if duplicate values are found for the element, or when its count is greater than 1.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00040">

<sch:rule context="*[@ism:classification and $ISM_CAPCO_RESOURCE]">
<!-- Define variables -->
<sch:let name="errMsg_ValueNotFound"
value="' [ISM-ID-00040][Error] If ISM_CAPCO_RESOURCE and attribute ownerProducer contains
[USA] then attribute classification must have a value in CVEnumISMClassificationUS.xml.'
"/>

<sch:let name="dataFileElems" value="$classificationUSList"/>
<sch:let name="attrValues" value="./@ism:classification"/>
<sch:let name="attrValueTokens" value="tokenize(string($attrValues),' ')/>
<sch:let name="capco" value="contains(string(./@ism:ownerProducer), 'USA')"/>

<!-- Get the position of each client node relative to its position in the master list. If
the node is not found, return a -1 -->
<sch:let name="orderNums"
value=" for $token in $attrValueTokens return if
($dataFileElems[matches($token,concat('^',text(),'$'))]) then
count(($dataFileElems[matches($token,concat('^',text(),'$'))])/preceding::*) + 1 else -1"/>

<!-- Determine if the list has invalid values. If and only if it does, figure out which
ones are invalids -->
<sch:let name="hasInvalids" value="count(index-of($orderNums,-1)) > 0"/>
<sch:let name="invalidValues"
value=" if ($hasInvalids) then distinct-values( for $token in index-of($orderNums,-1)
return $attrValueTokens[$token] ) else null "/>
```

```

<!-- Determine if the list has duplicate values. If and only if it does, figure out which
ones are duplicates -->
<sch:let name="hasDups"
value="count(distinct-values($attrValueTokens)) != count($attrValueTokens)"/>
<sch:let name="dupValues"
value=" if ($hasDups) then distinct-values( for $token in $attrValueTokens return
if (count(index-of($attrValueTokens,$token)) > 1) then $attrValueTokens[index-
of($attrValueTokens,$token)[1]] else null ) else null "/>

<!-- Execute tests -->
<sch:assert id="ISM-00040" flag="error"
test="if(not($capco)) then true() else not($hasInvalids)">
<sch:value-of select="$errMsg_ValueNotFound"/>
Invalid value of [<sch:value-of select="$invalidValues"/>]</sch:assert>
<sch:assert test="not($hasDups)" flag="undefined">Duplicate values found [<sch:value-of
select="$dupValues"/>] for [<sch:value-of select="$attrValueTokens"/>] </sch:assert>
</sch:rule>
</sch:pattern>

```

## Rule: ISM-ID-00041

FileName:./Rules/releasableTo/ISM\_ID\_00041.sch

### Rule Description:

[ISM-ID-00041][Error] If ISM\_CAPCO\_RESOURCE and attribute releasableTo is specified, then each of its values must be ordered in accordance with CVEnumISMRelTo.xml.

### Code Description:

To perform sorting, this rule first retrieves the CVE values for the attribute to be sorted, which in this case is releasableTo. Then, each attribute token is converted into a numerical value based on its characters. Next, each attribute token is given an order number, which compares its position to that of its value in the CVE file. Next, each order number is compared to that of its previous sibling to determine if the tokens are in order. If a token is found whose order number is less than that of its previous sibling, 0 is returned for its sorted order number. If a token's order number is greater than that of its previous sibling, 1 is returned. If two tokens have the same order number, their original attribute values are compared. If the original attribute value contains numbers then the comparison is made on its converted numerical value; otherwise, the comparison is made on its string value. If an attribute value is found whose value is less than that of its previous sibling, 0 is returned for its sorted order number; otherwise 2 is returned. Finally, if any tokens are found with 0 as its sorted order number, then the rule fails as those tokens are out of order.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00041">

<sch:rule context="*[@ism:releasableTo]">
<!-- Define variables -->
<sch:let name="errMsg_AlphabeticalOrder"
value="' [ISM-ID-00041][Error] If ISM_CAPCO_RESOURCE and attribute releasableTo is
specified, then each of its values must be ordered in accordance with CVEnumISMRelTo.xml.
'"/>

<sch:let name="dataFileElems" value="$releasableToList"/>
<sch:let name="attrValues" value="./@ism:releasableTo"/>
<sch:let name="attrValueTokens" value="tokenize(string($attrValues),' ')/>

<!-- Convert each character to a numerical value, then concatenate the results to form a
number-string -->
<sch:let name="convertStrToNum"
value=" for $token in $attrValueTokens return number(string-join( for $index in 1
to string-length($token) return for $char in substring($token, $index, 1) return if
(contains(string('0123456789'), $char)) then $char else if (contains(string('ABCDEFGHI'),
$char)) then translate(string($char), 'ABCDEFGHI', '123456789') else if
(contains(string('JKLMNOPQRS'), $char)) then concat('1', translate(string($char),
'JKLMNOPQRS', '0123456789')) else if (contains(string('TUVWXYZ'), $char)) then concat('2',
translate(string($char), 'TUVWXYZ', '0123456')) else '0' , '')) "/>
```

```

<!-- Get the position of each client node relative to its position in the master list. If
the node is not found, return a -1 -->
<sch:let name="orderNums"
value=" for $token in $attrValueTokens return if
($dataFileElems[matches($token,concat('^',text(),'$'))]) then
count(($dataFileElems[matches($token,concat('^',text(),'$'))]/preceding::*) + 1 else -1"/
>

<!-- Create a sequence that returns a 0 if the previous sibling has a higher order number,
else return a 1 -->
<sch:let name="sortedOrderNums"
value=" for $index in distinct-values(for $token in $orderNums return index-of($orderNums,
$token)) return if($index = 1 or $orderNums[$index] > $orderNums[$index - 1])
then 1 else if ($orderNums[$index] < $orderNums[$index - 1]) then 0 else if
(matches($attrValueTokens[$index], '[0-9]') or matches($attrValueTokens[$index - 1],
'[0-9]')) then if ($convertStrToNum[$index - 1] > $convertStrToNum[$index]) then 0 else
2 else if (compare($attrValueTokens[$index - 1],$attrValueTokens[$index])=1) then 0 else 2
"/>
<sch:let name="hasUnsorted" value="count(index-of($sortedOrderNums,0)) > 0"/>
<sch:let name="unsortedValues"
value=" if ($hasUnsorted) then distinct-values( for $token in index-of($sortedOrderNums,0)
return $attrValueTokens[$token] ) else null "/>

<sch:assert id="ISM-00041" test="not($hasUnsorted)" flag="error">
<sch:value-of select="$errMsg_AlphabeticalOrder"/>
The following values are out of order [<sch:value-of select="$unsortedValues"/>] for
[<sch:value-of select="$attrValueTokens"/>] </sch:assert>
</sch:rule>
</sch:pattern>

```

## Rule: ISM-ID-00042

FileName:./Rules/SCIcontrols/ISM\_ID\_00042.sch

### Rule Description:

[ISM-ID-00042][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols is specified, each of its values must be ordered in accordance with CVEnumISMSCIControls.xml.

### Code Description:

To perform sorting, this rule first retrieves the CVE values for the attribute to be sorted, which in this case is SCIcontrols. Then, each attribute token is converted into a numerical value based on its characters. Next, each attribute token is given an order number, which compares its position to that of its value in the CVE file. Next, each order number is compared to that of its previous sibling to determine if the tokens are in order. If a token is found whose order number is less than that of its previous sibling, 0 is returned for its sorted order number. If a token's order number is greater than that of its previous sibling, 1 is returned. If two tokens have the same order number, their original attribute values are compared. If the original attribute value contains numbers then the comparison is made on its converted numerical value; otherwise, the comparison is made on its string value. If an attribute value is found whose value is less than that of its previous sibling, 0 is returned for its sorted order number; otherwise 2 is returned. Finally, if any tokens are found with 0 as its sorted order number, then the rule fails as those tokens are out of order.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00042">

<sch:rule context="*[@ism:SCIcontrols]">
<!-- Define variables -->
<sch:let name="errMsg_AlphabeticalOrder"
value="' [ISM-ID-00042][Error] If ISM_CAPCO_RESOURCE and attribute SCIcontrols is
specified, each of its values must be ordered in accordance with CVEnumISMSCIControls.xml.
'"/>

<sch:let name="dataFileElems" value="$SCIcontrolsList"/>
<sch:let name="attrValues" value="./@ism:SCIcontrols"/>
<sch:let name="attrValueTokens" value="tokenize(string($attrValues),' ')/>

<!-- Convert each character to a numerical value, then concatenate the results to form a
number-string -->
<sch:let name="convertStrToNum"
value=" for $token in $attrValueTokens return number(string-join( for $index in 1
to string-length($token) return for $char in substring($token, $index, 1) return if
(contains(string('0123456789'), $char)) then $char else if (contains(string('ABCDEFGHI'),
$char)) then translate(string($char), 'ABCDEFGHI', '123456789') else if
(contains(string('JKLMNOPQRS'), $char)) then concat('1', translate(string($char),
'JKLMNOPQRS', '0123456789')) else if (contains(string('TUVWXYZ'), $char)) then concat('2',
translate(string($char), 'TUVWXYZ', '0123456')) else '0' , '')) "/>
```

```

<!-- Get the position of each client node relative to its position in the master list. If
the node is not found, return a -1 -->
<sch:let name="orderNums"
value=" for $token in $attrValueTokens return if
($dataFileElems[matches($token,concat('^',text(),'$'))]) then
count(($dataFileElems[matches($token,concat('^',text(),'$'))]/preceding::*) + 1 else -1"/
>

<!-- Create a sequence that returns a 0 if the previous sibling has a higher order number,
else return a 1 -->
<sch:let name="sortedOrderNums"
value=" for $index in distinct-values(for $token in $orderNums return index-of($orderNums,
$token)) return if($index = 1 or $orderNums[$index] > $orderNums[$index - 1])
then 1 else if ($orderNums[$index] < $orderNums[$index - 1]) then 0 else if
(matches($attrValueTokens[$index], '[0-9]') or matches($attrValueTokens[$index - 1],
'[0-9]')) then if ($convertStrToNum[$index - 1] > $convertStrToNum[$index]) then 0 else
2 else if (compare($attrValueTokens[$index - 1],$attrValueTokens[$index])=1) then 0 else 2
"/>
<sch:let name="hasUnsorted" value="count(index-of($sortedOrderNums,0)) > 0"/>
<sch:let name="unsortedValues"
value=" if ($hasUnsorted) then distinct-values( for $token in index-of($sortedOrderNums,0)
return $attrValueTokens[$token] ) else null "/>

<sch:assert id="ISM-00042" test="not($hasUnsorted)" flag="error">
<sch:value-of select="$errMsg_AlphabeticalOrder"/>
The following values are out of order [<sch:value-of select="$unsortedValues"/>] for
[<sch:value-of select="$attrValueTokens"/>] </sch:assert>
</sch:rule>
</sch:pattern>

```



## Rule: ISM-ID-00043

FileName:./Rules/SCIcontrols/ISM\_ID\_00043.sch

### Rule Description:

[ISM-ID-00043][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols contains the name token [SI], then attribute classification must have a value of [TS], [S], or [C]. Human Readable: A USA document containing Special Intelligence (SI) data must be classified CONFIDENTIAL, SECRET, or TOP SECRET.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIcontrols specified with a value containing [SI], then we make sure that attribute classification has a value of [TS], [S], or [C].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00043">

<sch:rule context="*[@ism:SCIcontrols]">
<sch:assert id="ISM-00043"
test="if(not($ISM_CAPCO_RESOURCE)) then true() else if ( index-of(tokenize(string(./
@ism:SCIcontrols), ' '), 'SI')>0 and not( matches(./@ism:classification, '^([TS|S|C])$')) )
then false() else true() "
flag="error">
[ISM-ID-00043][Error] If ISM_CAPCO_RESOURCE and attribute SCIcontrols contains the name
token [SI], then attribute
classification must have a value of [TS], [S], or [C].

Human Readable: A USA document containing Special Intelligence (SI) data must be
classified CONFIDENTIAL, SECRET, or TOP SECRET.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00044

FileName:./Rules/SCIcontrols/ISM\_ID\_00044.sch

### Rule Description:

[ISM-ID-00044][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols contains the name token [SI-G], then attribute classification must have a value of [TS]. Human Readable: A USA document containing Special Intelligence (SI) GAMMA compartment data must be classified TOP SECRET.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIcontrols specified with a value containing [SI-G], then we make sure that attribute classification has a value of [TS].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00044">

<sch:rule context="*[@ism:SCIcontrols]">
<sch:assert id="ISM-00045"
test="if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of(tokenize(string(./
@ism:SCIcontrols),' '), 'SI-G')>0 and not(matches(./@ism:classification, '^TS$')) ) then
false() else true() "
flag="error">
[ISM-ID-00044][Error] If ISM_CAPCO_RESOURCE and attribute SCIcontrols contains the name
token [SI-G], then attribute
classification must have a value of [TS].

Human Readable: A USA document containing Special Intelligence (SI) GAMMA compartment data
must be classified TOP SECRET.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00045

FileName:./Rules/SCIcontrols/ISM\_ID\_00045.sch

### Rule Description:

[ISM-ID-00045][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols contains the name token [SI-G], then attribute disseminationControls must contain the name token [OC]. Human Readable: A USA document containing Special Intelligence (SI) GAMMA compartment data must be marked for ORIGINATOR CONTROLLED dissemination.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIcontrols specified with a value containing [SI-G], then we make sure that attribute disseminationControls contains the value [OC].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00045">

<sch:rule context="*[@ism:SCIcontrols]">
<sch:assert id="ISM-00045"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of(tokenize(string(./
@ism:SCIcontrols),' '), 'SI-G')>0) then index-of(tokenize(string(./
@ism:disseminationControls),' '), 'OC')>0 else true() "
flag="error">
[ISM-ID-00045][Error] If ISM_CAPCO_RESOURCE and attribute SCIcontrols contains the name
token [SI-G], then attribute
disseminationControls must contain the name token [OC].

Human Readable: A USA document containing Special Intelligence (SI) GAMMA compartment data
must
be marked for ORIGINATOR CONTROLLED dissemination.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00046

FileName:./Rules/SCIcontrols/ISM\_ID\_00046.sch

### Rule Description:

[ISM-ID-00046][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols contains a name token starting with [SI-ECI], then attribute classification must have a value of [TS]. Human Readable: A USA document containing Special Intelligence (SI) ECI compartment data must be classified TOP SECRET.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute classification specified with a value of [TS] then the rule does not apply and we return true. Otherwise, we make sure that attribute SCIcontrols does not contain the value [SI-ECI].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00046">

<sch:rule context="*[@ism:SCIcontrols]">
<sch:assert id="ISM-00046"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(matches(./@ism:classification,'^TS$')) then true() else count( for $token in tokenize(string(./@ism:SCIcontrols),' ') return if(matches($token,'^SI-ECI')) then 1 else null )=0 "
flag="error">
[ISM-ID-00046][Error] If ISM_CAPCO_RESOURCE and attribute SCIcontrols contains
a name token starting with [SI-ECI], then attribute classification must have a
value of [TS].

Human Readable: A USA document containing Special Intelligence (SI) ECI compartment
data must be classified TOP SECRET.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00047

FileName:./Rules/SCIcontrols/ISM\_ID\_00047.sch

### Rule Description:

[ISM-ID-00047][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols contains the name token [TK], then attribute classification must have a value of [TS] or [S]. Human Readable: A USA document containing TALENT KEYHOLE data must be classified SECRET or TOP SECRET.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIcontrols specified with a value containing [TK], then we make sure that attribute classification has a value of [TS] or [S].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00047">

<sch:rule context="*[@ism:SCIcontrols]">
<sch:assert id="ISM-00047"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of(tokenize(string(./
@ism:SCIcontrols), ' '), 'TK')>0 and not( matches(./@ism:classification, '^(TS|S)$')) )
then false() else true() "
flag="error">
[ISM-ID-00047][Error] If ISM_CAPCO_RESOURCE and attribute SCIcontrols contains
the name token [TK], then attribute classification must have a value of [TS] or [S].

Human Readable: A USA document containing TALENT KEYHOLE data must be classified
SECRET or TOP SECRET.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00048

FileName:./Rules/SCIcontrols/ISM\_ID\_00048.sch

### Rule Description:

[ISM-ID-00048][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols contains the name token [HCS], then attribute classification must have a value of [TS], [S], or [C]. Human Readable: A USA document containing HCS data must be classified CONFIDENTIAL, SECRET, or TOP SECRET.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIcontrols specified with a value containing [HCS], then we make sure that attribute classification has a value of [TS], [S], or [C].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00048">

<sch:rule context="*[@ism:SCIcontrols]">
<sch:assert id="ISM-00048"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if( index-of(tokenize(string(./
@ism:SCIcontrols), ' '), 'HCS')>0 and not(matches(./@ism:classification, '^(TS|S|C)$')) )
then false() else true() "
flag="error">
[ISM-ID-00048][Error] If ISM_CAPCO_RESOURCE and attribute SCIcontrols contains the name
token [HCS], then attribute
classification must have a value of [TS], [S], or [C].

Human Readable: A USA document containing HCS data must be classified CONFIDENTIAL,
SECRET, or TOP SECRET.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00049

FileName:./Rules/SCIcontrols/ISM\_ID\_00049.sch

### Rule Description:

[ISM-ID-00049][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols contains the name token [HCS], then attribute disseminationControls must contain the name token [NF]. Human Readable: A USA document containing HCS data must be marked for NO FOREIGN dissemination.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIcontrols specified with a value containing [HCS], then we make sure that attribute disseminationControls contains the value [NF].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00049">

<sch:rule context="*[@ism:SCIcontrols]">
<sch:assert id="ISM-00048"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not( index-of(tokenize(string(./
@ism:SCIcontrols), ' '), 'HCS')>0 and not(index-of(tokenize(string(./
@ism:disseminationControls), ' '), 'NF')) ) "
flag="error">
[ISM-ID-00049][Error] If ISM_CAPCO_RESOURCE and attribute SCIcontrols contains the name
token [HCS], then attribute
disseminationControls must contain the name token [NF].

Human Readable: A USA document containing HCS data must be marked for NO FOREIGN
dissemination.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00056

FileName:./Rules/resourceElement/ISM\_ID\_00056.sch

### Rule Description:

[ISM-ID-00056][Error] If ISM\_CAPCO\_RESOURCE and attribute classification of ISM\_RESOURCE\_ELEMENT has a value of [U] then no element meeting ISM\_CONTRIBUTES\_USA in the document may have a classification attribute of [C], [S] or [TS]. Human Readable: USA UNCLASSIFIED documents can't have TOP SECRET, SECRET, or CONFIDENTIAL data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute classification specified with a value of [U], then we make sure that no portion with ownerProducer containing USA has attribute classification specified with a value of [C], [S], or [TS].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00056">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00056"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(not(./@ism:classification='U'))
then true() else count( for $each in $partTags return if(contains(string($each/
@ism:ownerProducer), 'USA') and not($each/@ism:classification='U')) then $each else
null )=0 "
flag="error">
[ISM-ID-00056][Error] USA UNCLASSIFIED documents can't have TOP SECRET, SECRET, or
CONFIDENTIAL data.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00057

FileName:./Rules/resourceElement/ISM\_ID\_00057.sch

### Rule Description:

[ISM-ID-00057][Error] If ISM\_CAPCO\_RESOURCE and attribute classification of ISM\_RESOURCE\_ELEMENT has a value of [U] then no element meeting ISM\_CONTRIBUTES in the document may have a classification attribute of [R].

Human Readable: USA UNCLASSIFIED documents cannot have RESTRICTED data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute classification specified with a value of [U], then we make sure that no portion has attribute classification specified with a value of [R].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00057">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00057"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(not(./@ism:classification='U'))
then true() else not(index-of($partClassification_tok, 'R')>0) "
flag="error">
[ISM-ID-00057][Error] USA UNCLASSIFIED documents cannot have RESTRICTED data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00058

FileName:./Rules/resourceElement/ISM\_ID\_00058.sch

### Rule Description:

[ISM-ID-00058][Error] If ISM\_CAPCO\_RESOURCE and attribute classification of ISM\_RESOURCE\_ELEMENT has a value of [C] then no element meeting ISM\_CONTRIBUTES\_USA in the document may have a classification attribute of [S] or [TS]. Human Readable: USA CONFIDENTIAL documents can't have TOP SECRET or SECRET data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute classification specified with a value of [C], then we make sure that no portion with ownerProducer containing USA has attribute classification specified with a value of [TS] or [S].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00058">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00058"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(not(./@ism:classification='C'))
then true() else count( for $each in $partTags return if(contains(string($each/
@ism:ownerProducer), 'USA') and not($each/@ism:classification='U' or $each/
@ism:classification='C') ) then $each else null )=0 "
flag="error">
[ISM-ID-00058][Error] USA CONFIDENTIAL documents can't have TOP SECRET or SECRET data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00059

FileName:./Rules/resourceElement/ISM\_ID\_00059.sch

### Rule Description:

[ISM-ID-00059][Error] If ISM\_CAPCO\_RESOURCE and attribute classification of ISM\_RESOURCE\_ELEMENT has a value of [S] then no element meeting ISM\_CONTRIBUTES\_USA in the document may have a classification attribute of [TS]. Human Readable: USA SECRET documents can't have TOP SECRET data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute classification specified with a value of [S], then we make sure that no portion with ownerProducer containing USA has attribute classification specified with a value of [TS].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00059">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00059"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(not(./@ism:classification='S'))
then true() else count( for $each in $partTags return if(contains(string($each/
@ism:ownerProducer), 'USA') and not($each/@ism:classification='U' or $each/
@ism:classification='C' or $each/@ism:classification='S')) then $each else null )=0 "
flag="error">
[ISM-ID-00059][Error] USA SECRET documents can't have TOP SECRET data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00060

FileName:./Rules/resourceElement/ISM\_ID\_00060.sch

### Rule Description:

[ISM-ID-00060][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute SClcontrols containing a value of [SI] then the ISM\_RESOURCE\_ELEMENT element's SClcontrols must contain [SI]. Human Readable: USA documents having SI data must have SI at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SClcontrols specified with a value containing [SI] and does not have attribute excludeFromRollup set to true, then we make sure that the resourceElement has attribute SClcontrols specified with a value containing [SI].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00060">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00060"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partSClcontrols_tok,
'SI') &gt; 0) then index-of($bannerSClcontrols_tok, 'SI') &gt; 0 else true() "
flag="error">
[ISM-ID-00060][Error] USA documents having SI data must have SI at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00061

FileName:./Rules/resourceElement/ISM\_ID\_00061.sch

### Rule Description:

[ISM-ID-00061][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute SCIcontrols containing a value of [SI-G] then the ISM\_RESOURCE\_ELEMENT element's SCIcontrols must contain [SI-G]. Human Readable: USA documents having SI-G data must have SI-G at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIcontrols specified with a value containing [SI-G] and does not have attribute excludeFromRollup set to true, then we make sure that the resourceElement has attribute SCIcontrols specified with a value containing [SI-G].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00061">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00061"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partSCIcontrols_tok,
'SI-G') &gt; 0) then index-of($bannerSCIcontrols_tok, 'SI-G') &gt; 0 else true() "
flag="error">
[ISM-ID-00061][Error] USA documents having SI-G data must have SI-G at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00062

FileName:./Rules/resourceElement/ISM\_ID\_00062.sch

### Rule Description:

[ISM-ID-00062][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute SCIconcontrols containing a value of [TK] then the ISM\_RESOURCE\_ELEMENT node's SCIconcontrols must contain [TK]. Human Readable: USA documents having TK data must have TK at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIconcontrols specified with a value containing [TK] and does not have attribute excludeFromRollup set to true, then we make sure that the resourceElement has attribute SCIconcontrols specified with a value containing [TK].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00062">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00062"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partSCIconcontrols_tok,
'TK') &gt; 0) then index-of($bannerSCIconcontrols_tok, 'TK') &gt; 0 else true() "
flag="error">
[ISM-ID-00062][Error] USA documents having TK data must have TK at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00063

FileName:./Rules/resourceElement/ISM\_ID\_00063.sch

### Rule Description:

[ISM-ID-00063][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute SCIconcontrols containing a value of [HCS] then the ISM\_RESOURCE\_ELEMENT node's SCIconcontrols must contain [HCS]. Human Readable: USA documents having HCS data must have HCS at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIconcontrols specified with a value containing [HCS] and does not have attribute excludeFromRollup set to true, then we make sure that the resourceElement has attribute SCIconcontrols specified with a value containing [HCS].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00063">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00063"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partSCIconcontrols_tok,
'HCS') > 0) then index-of($bannerSCIconcontrols_tok, 'HCS') > 0 else true() "
flag="error">
[ISM-ID-00063][Error] USA documents having HCS data must have HCS at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00064

FileName:../Rules/resourceElement/ISM\_ID\_00064.sch

### Rule Description:

[ISM-ID-00064][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute FGIsorceOpen containing any value then the ISM\_RESOURCE\_ELEMENT must have either FGIsorceOpen or FGIsorceProtected with a value. Human Readable: USA documents having FGI Open data must have FGI Open or FGI Protected at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute FGIsorceOpen specified and does not have attribute excludeFromRollup set to true, then we make sure that the resourceElement has one of the following attributes specified: FGIsorceOpen or FGIsorceProtected.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00064">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00064"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(not(empty($partFGIsorceOpen)))
then ($bannerFGIsorceOpen or $bannerFGIsorceProtected) else true() "
flag="error">
[ISM-ID-00064][Error] USA documents having FGI Open data must have FGI Open or FGI
Protected at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00065

FileName:../Rules/resourceElement/ISM\_ID\_00065.sch

### Rule Description:

[ISM-ID-00065][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute FGIsorceProtected containing any value then the ISM\_RESOURCE\_ELEMENT must have FGIsorceProtected with a value. Human Readable: USA documents having FGI Protected data must have FGI Protected at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute FGIsorceProtected specified with a non-empty value and does not have attribute excludeFromRollup set to true, then we make sure that the banner has attribute FGIsorceProtected specified with a non-empty value.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00065">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00065"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else
if(not(empty($partFGIsorceProtected))) then $bannerFGIsorceProtected else true() "
flag="error">
[ISM-ID-00065][Error] USA documents having FGI Protected data must have FGI Protected at
the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00066

FileName:./Rules/resourceElement/ISM\_ID\_00066.sch

### Rule Description:

[ISM-ID-00066][Error] If ISM\_CAPCO\_RESOURCE and: 1. Any element meeting ISM\_CONTRIBUTES in the document has the attribute disseminationControls containing [FOUO] AND 2. ISM\_RESOURCE\_ELEMENT has the attribute classification [U] AND 3. No element meeting ISM\_CONTRIBUTES in the document has nonICmarkings containing [SBU], [SBU-NF], [LES], [LES-NF] Then the ISM\_RESOURCE\_ELEMENT must have disseminationControls containing [FOUO]. Human Readable: USA Unclassified documents having FOUO data and not having SBU, SBU-NF, LES, or LES-NF must have FOUO at the resource level.

### Code Description:

If CAPCO rules do not apply to the document, then the rule does not apply and we return true. Verify that this is actually the ISM\_RESOURCE\_ELEMENT If the resourceElement has attribute classification specified with a value other than [U], then the rule does not apply and we return true. If the banner has attribute disseminationControls specified with a value containing [FOUO], or no element has attribute disseminationControls specified with value containing [FOUO], then the rule does not apply and we return true. Otherwise, we make sure that an element has attribute nonICmarkings specified with a value of [SBU], [SBU-NF], [LES], or [LES-NF].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00066">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00066"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(not($bannerClassification='U'))
then true() else if(not(index-of($dcTagsFound,'FOUO')>0)) then true() else index-
of($partNonICmarkings_tok,'SBU')>0 or index-of($partNonICmarkings_tok,'SBU-NF')>0
or index-of($partNonICmarkings_tok,'LES')>0 or index-of($partNonICmarkings_tok,'LES-
NF')>0 "
flag="error">
[ISM-ID-00066][Error] USA Unclassified documents having FOUO data and not having SBU, SBU-
NF, LES, or LES-NF must have
FOUO at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00067

FileName:./Rules/resourceElement/ISM\_ID\_00067.sch

### Rule Description:

[ISM-ID-00067][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document has the attribute disseminationControls containing [OC] then the ISM\_RESOURCE\_ELEMENT must have disseminationControls containing [OC]. Human Readable: USA documents having ORCON data must have ORCON at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. We make sure that no element does not have attribute excludeFromRollup set to true has attribute disseminationControls specified with a value containing [OC] unless the resourceElement also has attribute disseminationControls specified with a value containing [OC].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00067">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00067"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not(index-of($dcTagsFound,'OC') >
0) "
flag="error">
[ISM-ID-00067][Error] USA documents having ORCON data must have ORCON at the resource
level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00068

FileName:../Rules/resourceElement/ISM\_ID\_00068.sch

### Rule Description:

[ISM-ID-00068][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute disseminationControls containing [IMC] then the ISM\_RESOURCE\_ELEMENT must have disseminationControls containing [IMC]. Human Readable: USA documents having IMCON data must have IMCON at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. We make sure that no element that does not have attribute excludeFromRollup set to true has attribute disseminationControls specified with a value containing [IMC] unless the resourceElement also has attribute disseminationControls specified with a value containing [IMC].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00068">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00068"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not(index-of($dcTagsFound,'IMC') >
0) "
flag="error">
[ISM-ID-00068][Error] USA documents having IMCON data must have IMCON at the resource
level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00070

FileName:./Rules/resourceElement/ISM\_ID\_00070.sch

### Rule Description:

[ISM-ID-00070][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute disseminationControls containing [NF] then the ISM\_RESOURCE\_ELEMENT must have disseminationControls containing [NF]. Human Readable: USA documents having NF data must have NF at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. We make sure that no element that does not have attribute excludeFromRollup set to true has attribute disseminationControls specified with a value containing [NF] unless the resourceElement also has attribute disseminationControls specified with a value containing [NF].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00070">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00070"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not(index-of($dcTagsFound,'NF') >
0) "
flag="error">
[ISM-ID-00070][Error] USA documents having NF data must have NF at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00071

FileName:./Rules/resourceElement/ISM\_ID\_00071.sch

### Rule Description:

[ISM-ID-00071][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute disseminationControls containing [PR] then the ISM\_RESOURCE\_ELEMENT must have disseminationControls containing [PR]. Human Readable: USA documents having PROPIN data must have PROPIN at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. We make sure that no element that does not have attribute excludeFromRollup set to true has attribute disseminationControls specified with a value containing [PR] unless the resourceElement also has attribute disseminationControls specified with a value containing [PR].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00071">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00071"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not(index-of($dcTagsFound,'PR') >
0) "
flag="error">
[ISM-ID-00071][Error] USA documents having PROPIN data must have PROPIN at the resource
level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00072

FileName:./Rules/resourceElement/ISM\_ID\_00072.sch

### Rule Description:

[ISM-ID-00072][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute atomicEnergyMarkings containing [RD] then the ISM\_RESOURCE\_ELEMENT must have atomicEnergyMarkings containing [RD]. Human Readable: USA documents having Restricted Data (RD) must have RD at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. We make sure that no element that does not have attribute excludeFromRollup set to true has attribute atomicEnergyMarkings specified with a value containing [RD] unless the resourceElement also has attribute atomicEnergyMarkings specified with a value containing [RD].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00072">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00072"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not(index-of($aeaTagsFound,'RD') >
0) "
flag="error">
[ISM-ID-00072][Error] USA documents having Restricted Data (RD) must have RD at the
resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00073

FileName:./Rules/resourceElement/ISM\_ID\_00073.sch

### Rule Description:

[ISM-ID-00073][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute atomicEnergyMarkings containing [RD-CNWDI] then the ISM\_RESOURCE\_ELEMENT must have atomicEnergyMarkings containing [RD-CNWDI]. Human Readable: USA documents having Restricted CNWDI Data must have Restricted CNWDI Data at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. We make sure that no element that does not have attribute excludeFromRollup set to true has attribute atomicEnergyMarkings specified with a value containing [RD-CNWDI] unless the resourceElement also has attribute atomicEnergyMarkings specified with a value containing [RD-CNWDI].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00073">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00073"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not(index-of($aeaTagsFound,'RD-
CNWDI') &gt; 0) "
flag="error">
[ISM-ID-00073][Error] USA documents having Restricted CNWDI Data must have Restricted
CNWDI Data at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00074

FileName:./Rules/resourceElement/ISM\_ID\_00074.sch

### Rule Description:

[ISM-ID-00074][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document has the attribute atomicEnergyMarkings containing [RD-SG-##] then the ISM\_RESOURCE\_ELEMENT must have atomicEnergyMarkings containing [RD-SG-##]. ## represent digits 1 through 99 the ## must match. Human Readable: USA documents having Restricted SIGMA-## Data must have the same Restricted SIGMA-## Data at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. We make sure that no element that does not have attribute excludeFromRollup set to true has attribute atomicEnergyMarkings specified with a value containing [RD-SG-##], where ## is represented by a regular expression matching numbers 1 through 99, unless the resourceElement also has attribute atomicEnergyMarkings specified with a value containing [RD-SG-##].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00074">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00074"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else sum( for $item in for $token in
$partAtomicEnergyMarkings_tok return if(matches($token,'^RD-SG-[1-9][0-9]?$')) then $token
else null return if(index-of($bannerAtomicEnergyMarkings_tok, $item)>0) then 0 else
1 )=0 "
flag="error">
[ISM-ID-00074][Error] USA documents having Restricted SIGMA-## Data must have the same
Restricted SIGMA-## Data at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00075

FileName:./Rules/resourceElement/ISM\_ID\_00075.sch

### Rule Description:

[ISM-ID-00075][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute atomicEnergyMarkings containing [FRD] then the ISM\_RESOURCE\_ELEMENT must have atomicEnergyMarkings containing [FRD]. Human Readable: USA documents having Formerly Restricted Data (FRD) must have FRD at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. We make sure that no element that does not have attribute excludeFromRollup set to true has attribute atomicEnergyMarkings specified with a value containing [FRD], unless the resourceElement also has attribute atomicEnergyMarkings specified with a value containing [FRD].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00075">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00075"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not(index-of($aeaTagsFound,'FRD')
    > 0) "
flag="error">
[ISM-ID-00075][Error] USA documents having Formerly Restricted Data (FRD) must have FRD at
the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00077

FileName:./Rules/resourceElement/ISM\_ID\_00077.sch

### Rule Description:

[ISM-ID-00077][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document has the attribute atomicEnergyMarkings containing [FRD-SG-##] then the ISM\_RESOURCE\_ELEMENT must have atomicEnergyMarkings containing [FRD-SG-##]. ## represent digits 1 through 99 the ## must match. Human Readable: USA documents having Formerly Restricted SIGMA-## Data must have the same Formerly Restricted SIGMA-## Data at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. We make sure that no element that does not have attribute excludeFromRollup set to true has attribute atomicEnergyMarkings specified with a value containing [FRD-SG-##], where ## is represented by a regular expression matching numbers 1 through 99, unless the resourceElement also has attribute atomicEnergyMarkings specified with a value containing [FRD-SG-##].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00077">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00077"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else sum( for $item in for $token in
$partAtomicEnergyMarkings_tok return if(matches($token,'^FRD-SG-[1-9][0-9]?$')) then
$token else null return if(index-of($bannerAtomicEnergyMarkings_tok, $item)>0) then 0
else 1 )=0 "
flag="error">
[ISM-ID-00077][Error] USA documents having Formerly Restricted SIGMA-## Data must have the
same Formerly Restricted SIGMA-## Data at
the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00078

FileName:./Rules/resourceElement/ISM\_ID\_00078.sch

### Rule Description:

[ISM-ID-00078][Error] If ISM\_CAPCO\_RESOURCE and the ISM\_RESOURCE\_ELEMENT node's classification has the value of [U] and any element meeting ISM\_CONTRIBUTES in the document has the attribute atomicEnergyMarkings containing [DCNI] then the ISM\_RESOURCE\_ELEMENT must have atomicEnergyMarkings containing [DCNI]. Human Readable: Unclassified USA documents having DCNI Data must have DCNI at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute classification specified with a value other than [U] then the rule does not apply and we return true. Otherwise, we make sure that no element has attribute atomicEnergyMarkings specified with a value containing [DCNI] and does not have attribute excludeFromRollup set to true, unless the resourceElement also has attribute atomicEnergyMarkings specified with a value containing [DCNI].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00078">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00078"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(not($bannerClassification='U'))
then true() else not(index-of($aeaTagsFound,'DCNI') > 0) "
flag="error">
[ISM-ID-00078][Error] Unclassified USA documents having DCNI Data must have DCNI at the
resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00079

FileName:./Rules/resourceElement/ISM\_ID\_00079.sch

### Rule Description:

[ISM-ID-00079][Error] If ISM\_CAPCO\_RESOURCE and ISM\_RESOURCE\_ELEMENT element's classification has the value of [U] and any element meeting ISM\_CONTRIBUTES in the document has the attribute atomicEnergyMarkings containing [UCNI] then the ISM\_RESOURCE\_ELEMENT must have atomicEnergyMarkings containing [UCNI]. Human Readable: Unclassified USA documents having UCNI Data must have UCNI at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute classification specified with a value other than [U] then the rule does not apply and we return true. Otherwise, we make sure that no element has attribute atomicEnergyMarkings specified with a value containing [UCNI] and does not have attribute excludeFromRollup set to true, unless the resourceElement also has attribute atomicEnergyMarkings specified with a value containing [UCNI].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00079">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00079"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(not($bannerClassification='U'))
then true() else not(index-of($aeaTagsFound,'UCNI') > 0) "
flag="error">
[ISM-ID-00079][Error] Unclassified USA documents having UCNI Data must have UCNI at the
resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00080

FileName:./Rules/resourceElement/ISM\_ID\_00080.sch

### Rule Description:

[ISM-ID-00080][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute disseminationControls containing [DSEN] then the ISM\_RESOURCE\_ELEMENT must have disseminationControls containing [DSEN]. Human Readable: USA documents having DSEN Data must have DSEN at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. We make sure that no element that does not have attribute excludeFromRollup set to true has attribute disseminationControls specified with a value containing [DSEN], unless the resourceElement also has attribute disseminationControls specified with a value containing [DSEN].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00080">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00080"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not(index-of($dcTagsFound,'DSEN')
> 0) "
flag="error">
[ISM-ID-00080][Error] USA documents having DSEN Data must have DSEN at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00081

FileName:./Rules/resourceElement/ISM\_ID\_00081.sch

### Rule Description:

[ISM-ID-00081][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute disseminationControls containing [FISA] then the ISM\_RESOURCE\_ELEMENT must have disseminationControls containing [FISA]. Human Readable: USA documents having FISA Data must have FISA at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. We make sure that no element that does not have attribute excludeFromRollup set to true has attribute disseminationControls specified with a value containing [FISA] and does not have attribute excludeFromRollup set to true, unless the resourceElement also has attribute disseminationControls specified with a value containing [FISA].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00081">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00081"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not(index-of($dcTagsFound,'FISA')
> 0) "
flag="error">
[ISM-ID-00081][Error] USA documents having FISA Data must have FISA at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00082

FileName:./Rules/resourceElement/ISM\_ID\_00082.sch

### Rule Description:

[ISM-ID-00082][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute nonICmarkings containing [SC] then the ISM\_RESOURCE\_ELEMENT must have nonICmarkings containing [SC]. Human Readable: USA documents having SC Data must have SC at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute nonICmarkings specified with a value containing [SC] and does not have attribute excludeFromRollup set to true, then we make sure that the resourceElement also has attribute nonICmarkings specified with a value containing [SC].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00082">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00082"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partNonICmarkings_tok,
'SC') &gt; 0) then (index-of($bannerNonICmarkings_tok, 'SC') &gt; 0) else true() "
flag="error">
[ISM-ID-00082][Error] USA documents having SC Data must have SC at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00083

FileName:./Rules/resourceElement/ISM\_ID\_00083.sch

### Rule Description:

[ISM-ID-00083][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute nonICmarkings containing [SINFO] then the ISM\_RESOURCE\_ELEMENT must have nonICmarkings containing [SINFO]. Human Readable: USA documents having SINFO Data must have SINFO at the resource level.

### Code Description:

If CAPCO rules do not apply to the document or the banner is classified then the rule does not apply and we return true. If any element has attribute nonICmarkings specified with a value containing [SINFO] and does not have attribute excludeFromRollup set to true, then we make sure that the resourceElement also has attribute nonICmarkings specified with a value containing [SINFO].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00083">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00083"
test=" if(not($ISM_CAPCO_RESOURCE) or not(matches(/@ism:classification, '^U$'))
then true() else if(index-of($partNonICmarkings_tok, 'SINFO') > 0) then (index-
of($bannerNonICmarkings_tok, 'SINFO') > 0) else true() "
flag="error">
[ISM-ID-00083][Error] USA documents having SINFO Data must have SINFO at the resource
level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00084

FileName:../Rules/resourceElement/ISM\_ID\_00084.sch

### Rule Description:

[ISM-ID-00084][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute nonICmarkings containing [DS] then the ISM\_RESOURCE\_ELEMENT must have nonICmarkings containing [DS]. Human Readable: USA documents having DS Data must have DS at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute nonICmarkings specified with a value containing [DS] and does not have attribute excludeFromRollup set to true, then we make sure that the resourceElement also has attribute nonICmarkings specified with a value containing [DS].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00084">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00084"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partNonICmarkings_tok,
'DS') > 0) then (index-of($bannerNonICmarkings_tok, 'DS') > 0) else true() "
flag="error">
[ISM-ID-00084][Error] USA documents having DS Data must have DS at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00085

FileName:./Rules/resourceElement/ISM\_ID\_00085.sch

### Rule Description:

[ISM-ID-00085][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document has the attribute nonICmarkings containing [XD] and does not have any element meeting ISM\_CONTRIBUTES in the document having the attribute nonICmarkings containing [ND] then the ISM\_RESOURCE\_ELEMENT must have nonICmarkings containing [XD]. Human Readable: USA documents having XD Data and not having ND must have XD at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute nonICmarkings specified with a value containing [XD] and no element has attribute nonICmarkings specified with a value containing [ND] and does not have attribute excludeFromRollup set to true, then we make sure that the resourceElement has attribute nonICmarkings specified with a value containing [XD].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00085">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00085"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partNonICmarkings_tok,
'XD') > 0 and not(index-of($partNonICmarkings_tok, 'ND')>0)) then index-
of($bannerNonICmarkings_tok, 'XD') > 0 else true() "
flag="error">
[ISM-ID-00085][Error] USA documents having XD Data and not having ND must have XD at the
resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00086

FileName:./Rules/resourceElement/ISM\_ID\_00086.sch

### Rule Description:

[ISM-ID-00086][Error] If ISM\_CAPCO\_RESOURCE and any element in the document: 1. Meets ISM\_CONTRIBUTES AND 2. Has the attribute nonICmarkings containing [ND] Then the ISM\_RESOURCE\_ELEMENT must have nonICmarkings containing [ND]. Human Readable: USA documents having ND Data must have ND at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute nonICmarkings specified with a value containing [ND] and does not have attribute excludeFromRollup set to true, then we make sure that the resourceElement has attribute nonICmarkings specified with a value containing [ND].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00086">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00086"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partNonICmarkings_tok,
'ND')>0) then index-of($bannerNonICmarkings_tok, 'ND') > 0 else true() "
flag="error">
[ISM-ID-00086][Error] USA documents having ND Data must have ND at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00087

FileName:./Rules/resourceElement/ISM\_ID\_00087.sch

### Rule Description:

[ISM-ID-00087][Error] If ISM\_CAPCO\_RESOURCE and there exist at least 2 elements in the document: 1. Each element: Meets ISM\_CONTRIBUTES AND 2. One of the elements: Has the attribute nonICmarkings containing [SBU-NF] AND 3. One of the elements: Has the attribute classification NOT having a value of [U] Then the ISM\_RESOURCE\_ELEMENT must have disseminationControls containing [NF]. Human Readable: Classified USA documents having SBU-NF Data must have NF at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute nonICmarkings specified with a value containing [SBU-NF], does not have attribute excludeFromRollup set to true, and the resourceElement has attribute classification specified with a value other than [U], then we make sure that the resourceElement has attribute disseminationControls specified with a value containing [NF].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00087">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00087"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partNonICmarkings_tok,
'SBU-NF') > 0 and not($bannerClassification='U')) then (index-
of($bannerDisseminationControls_tok, 'NF') > 0) else true() "
flag="error">
[ISM-ID-00087][Error] Classified USA documents having SBU-NF Data must have NF at the
resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00088

FileName:./Rules/resourceElement/ISM\_ID\_00088.sch

### Rule Description:

[ISM-ID-00088][Error] If ISM\_CAPCO\_RESOURCE and releasableTo is specified on the resource element then all classified portions must specify releasableTo. Human Readable: USA documents having any classified portion that is not RELEASABLE (REL) cannot be REL at the resource level.

### Code Description:

If CAPCO rules apply to the document, we verify that all portions either have the attribute classification specified with a value of [U] or classified portions of the document have the attribute releasableTo. Attribute releasableTo is only valid on an element if attribute disseminationControls is specified with a value containing [REL] or [EYES], as [REL] supersedes [EYES] in the banner. If any elements do not meet either of the two requirements stated above, then the assertion fails since attribute releasableTo appears on the banner but is not present on all classified portions.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00088">

<sch:rule context="*[$ISM_CAPCO_RESOURCE and generate-id(.) = generate-
id($ISM_RESOURCE_ELEMENT) and @ism:releasableTo]">
<sch:assert id="ISM-00088"
test="every $portion in $partTags satisfies ($portion/@ism:classification='U' or $portion/
@ism:releasableTo[normalize-space()]) "
flag="error">
[ISM-ID-00088][Error] USA documents having any classified portion that is not
Releasable cannot be REL at the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00090

FileName:./Rules/resourceElement/ISM\_ID\_00090.sch

### Rule Description:

[ISM-ID-00090][Error] If ISM\_CAPCO\_RESOURCE and any element: 1. Meets ISM\_CONTRIBUTES AND 2. Has the attribute disseminationControls containing [REL] Then the ISM\_RESOURCE\_ELEMENT must not have attribute disseminationControls containing [EYES]. Human Readable: USA documents with any portion that is REL must not be EYES at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute disseminationControls specified with a value containing [REL], then we make sure that the resourceElement has attribute disseminationControls specified with a value other than [EYES].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00090">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00090"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(sum(for $token in $partTags return
if(contains(string($token/@ism:disseminationControls), 'REL')) then 1 else 0) ) then
not(index-of($bannerDisseminationControls_tok, 'EYES')>0) else true() "
flag="error">
[ISM-ID-00090][Error] USA documents with any portion that is REL must not be EYES at the
resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00094

FileName:./Rules/disseminationControls/ISM\_ID\_00094.sch

### Rule Description:

[ISM-ID-00094][Error] ISM\_CAPCO\_RESOURCE and attribute disseminationControls contains the name token [REL], then attribute classification must not have a value of [U]. Human Readable: REL may not be used on UNCLASSIFIED portions.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check the classification attribute of the current element. If it does not have value of [U] we return true since this rule only applies to unclassified elements. If it is not [U] then we check that the attribute disseminationControls does not have a value of [REL].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00094">

<sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00094"
test=" if(not(./@ism:classification='U')) then true() else not(index-of(tokenize(string(./
@ism:disseminationControls), ' '), 'REL')>0) "
flag="error">
[ISM-ID-00094][Error] REL may not be used on UNCLASSIFIED portions.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00095

FileName:./Rules/FGIsourceOpen/ISM\_ID\_00095.sch

### Rule Description:

[ISM-ID-00095][Error] If ISM\_CAPCO\_RESOURCE and attribute FGIsourceOpen is specified then each of its values must be ordered in accordance with CVEnumISMFGIOpen.xml.

### Code Description:

To perform sorting, this rule first retrieves the CVE values for the attribute to be sorted, which in this case is FGIsourceOpen. Then, each attribute token is converted into a numerical value based on its characters. Next, each attribute token is given an order number, which compares its position to that of its value in the CVE file. Next, each order number is compared to that of its previous sibling to determine if the tokens are in order. If a token is found whose order number is less than that of its previous sibling, 0 is returned for its sorted order number. If a token's order number is greater than that of its previous sibling, 1 is returned. If two tokens have the same order number, their original attribute values are compared. If the original attribute value contains numbers then the comparison is made on its converted numerical value; otherwise, the comparison is made on its string value. If an attribute value is found whose value is less than that of its previous sibling, 0 is returned for its sorted order number; otherwise 2 is returned. Finally, if any tokens are found with 0 as its sorted order number, then the rule fails as those tokens are out of order.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00095">

  <sch:rule context="*[@ism:FGIsourceOpen and $ISM_CAPCO_RESOURCE]">
    <!-- Define variables -->
    <sch:let name="errMsg_AlphabeticalOrder"
value="' [ISM-ID-00095][Error] If ISM_CAPCO_RESOURCE and attribute FGIsourceOpen is
specified then each of its values must be ordered in accordance with CVEnumISMFGIOpen.xml.
'"/>

    <sch:let name="dataFileElems" value="$FGIsourceOpenList"/>
    <sch:let name="attrValues" value="string(./@ism:FGIsourceOpen)"/>
    <sch:let name="attrValueTokens" value="tokenize(string($attrValues),' ')/>

    <!-- Convert each character to a numerical value, then concatenate the results to form a
number-string -->
    <sch:let name="convertStrToNum"
value=" for $token in $attrValueTokens return number(string-join( for $index in 1
to string-length($token) return for $char in substring($token, $index, 1) return if
(contains(string('0123456789'), $char)) then $char else if (contains(string('ABCDEFGHI'),
$char)) then translate(string($char), 'ABCDEFGHI', '123456789') else if
(contains(string('JKLMNOPQRS'), $char)) then concat('1', translate(string($char),
'JKLMNOPQRS', '0123456789')) else if (contains(string('TUVWXYZ'), $char)) then concat('2',
translate(string($char), 'TUVWXYZ', '0123456')) else '0' , '')) "/>
```

```

<!-- Get the position of each client node relative to its position in the master list. If
the node is not found, return a -1 -->
<sch:let name="orderNums"
value=" for $token in $attrValueTokens return if
($dataFileElems[matches($token,concat('^',text(),'$'))]) then
count(($dataFileElems[matches($token,concat('^',text(),'$'))]/preceding::*) + 1 else -1"/
>

<!-- Create a sequence that returns a 0 if the previous sibling has a higher order number,
else return a 1 -->
<sch:let name="sortedOrderNums"
value=" for $index in distinct-values(for $token in $orderNums return index-of($orderNums,
$token)) return if($index = 1 or $orderNums[$index] > $orderNums[$index - 1])
then 1 else if ($orderNums[$index] < $orderNums[$index - 1]) then 0 else if
(matches($attrValueTokens[$index], '[0-9]') or matches($attrValueTokens[$index - 1],
'[0-9]')) then if ($convertStrToNum[$index - 1] > $convertStrToNum[$index]) then 0 else
2 else if (compare($attrValueTokens[$index - 1],$attrValueTokens[$index])=1) then 0 else 2
"/>
<sch:let name="hasUnsorted" value="count(index-of($sortedOrderNums,0)) > 0"/>
<sch:let name="unsortedValues"
value=" if ($hasUnsorted) then distinct-values( for $token in index-of($sortedOrderNums,0)
return $attrValueTokens[$token] ) else null "/>

<sch:assert id="ISM-00095" test="not($hasUnsorted)" flag="error">
<sch:value-of select="$errMsg_AlphabeticalOrder"/>
The following values are out of order [<sch:value-of select="$unsortedValues"/>] for
[<sch:value-of select="$attrValueTokens"/>] </sch:assert>
</sch:rule>
</sch:pattern>

```

## Rule: ISM-ID-00096

FileName:../Rules/FGIsourceProtected/ISM\_ID\_00096.sch

### Rule Description:

[ISM-ID-00096][Error] If ISM\_CAPCO\_RESOURCE and attribute FGIsourceProtected is specified then each of its values must be ordered in accordance with CVEnumISMFGIProtected.xml.

### Code Description:

To perform sorting, this rule first retrieves the CVE values for the attribute to be sorted, which in this case is FGIsourceProtected. Then, each attribute token is converted into a numerical value based on its characters. Next, each attribute token is given an order number, which compares its position to that of its value in the CVE file. Next, each order number is compared to that of its previous sibling to determine if the tokens are in order. If a token is found whose order number is less than that of its previous sibling, 0 is returned for its sorted order number. If a token's order number is greater than that of its previous sibling, 1 is returned. If two tokens have the same order number, their original attribute values are compared. If the original attribute value contains numbers then the comparison is made on its converted numerical value; otherwise, the comparison is made on its string value. If an attribute value is found whose value is less than that of its previous sibling, 0 is returned for its sorted order number; otherwise 2 is returned. Finally, if any tokens are found with 0 as its sorted order number, then the rule fails as those tokens are out of order.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00096">

<sch:rule context="*[@ism:FGIsourceProtected and $ISM_CAPCO_RESOURCE]">
<!-- Define variables -->
<sch:let name="errMsg_AlphabeticalOrder"
value="' [ISM-ID-00096][Error] If ISM_CAPCO_RESOURCE and attribute FGIsourceProtected
is specified then each of its values must be ordered in accordance with
CVEnumISMFGIProtected.xml. '"/>

<sch:let name="dataFileElems" value="$FGIsourceProtectedList"/>
<sch:let name="attrValues" value="string(./@ism:FGIsourceProtected)"/>
<sch:let name="attrValueTokens" value="tokenize(string($attrValues),' ')/>

<!-- Convert each character to a numerical value, then concatenate the results to form a
number-string -->
<sch:let name="convertStrToNum"
value=" for $token in $attrValueTokens return number(string-join( for $index in 1
to string-length($token) return for $char in substring($token, $index, 1) return if
(contains(string('0123456789'), $char)) then $char else if (contains(string('ABCDEFGHI'),
$char)) then translate(string($char), 'ABCDEFGHI', '123456789') else if
(contains(string('JKLMNOPQRS'), $char)) then concat('1', translate(string($char),
'JKLMNOPQRS', '0123456789')) else if (contains(string('TUVWXYZ'), $char)) then concat('2',
translate(string($char), 'TUVWXYZ', '0123456')) else '0' , '')) "/>
```

```

<!-- Get the position of each client node relative to its position in the master list. If
the node is not found, return a -1 -->
<sch:let name="orderNums"
value=" for $token in $attrValueTokens return if
($dataFileElems[matches($token,concat('^',text(),'$'))]) then
count(($dataFileElems[matches($token,concat('^',text(),'$'))])/preceding::*) + 1 else -1"/
>

<!-- Create a sequence that returns a 0 if the previous sibling has a higher order number,
else return a 1 -->
<sch:let name="sortedOrderNums"
value=" for $index in distinct-values(for $token in $orderNums return index-of($orderNums,
$token)) return if($index = 1 or $orderNums[$index] > $orderNums[$index - 1])
then 1 else if ($orderNums[$index] < $orderNums[$index - 1]) then 0 else if
(matches($attrValueTokens[$index], '[0-9]') or matches($attrValueTokens[$index - 1],
'[0-9]')) then if ($convertStrToNum[$index - 1] > $convertStrToNum[$index]) then 0 else
2 else if (compare($attrValueTokens[$index - 1],$attrValueTokens[$index])=1) then 0 else 2
"/>
<sch:let name="hasUnsorted" value="count(index-of($sortedOrderNums,0)) > 0"/>
<sch:let name="unsortedValues"
value=" if ($hasUnsorted) then distinct-values( for $token in index-of($sortedOrderNums,0)
return $attrValueTokens[$token] ) else null "/>

<sch:assert id="ISM-00096" test="not($hasUnsorted)" flag="error">
<sch:value-of select="$errMsg_AlphabeticalOrder"/>
The following values are out of order [<sch:value-of select="$unsortedValues"/>] for
[<sch:value-of select="$attrValueTokens"/>] </sch:assert>
</sch:rule>
</sch:pattern>

```

## Rule: ISM-ID-00097

FileName:../Rules/FGIsourceProtected/ISM\_ID\_00097.sch

### Rule Description:

[ISM-ID-00097][Warning] If ISM\_CAPCO\_RESOURCE and attribute FGIsourceProtected is specified with a value other than [FGI] then the value(s) must not be discoverable in IC shared spaces. Human Readable: FGI Protected should rarely if ever be seen outside of an agency's internal systems.

### Code Description:

Checks that the resource is a CAPCO resource and that the FGIsourceProtected attribute contains only the value [FGI].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00097">

<sch:rule context="*[@ism:FGIsourceProtected and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00097"
test="normalize-space(string(..@ism:FGIsourceProtected))='FGI' "
flag="warning">
[ISM-ID-00097][Warning] FGI Protected should rarely if ever be seen outside of an agency's
internal systems.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00099

FileName:../Rules/ownerProducer/ISM\_ID\_00099.sch

### Rule Description:

[ISM-ID-00099][Error] If ISM\_CAPCO\_RESOURCE attribute ownerProducer contains [FGI], it must be the only value.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. Otherwise, we make sure that the current element has attribute ownerProducer specified with [FGI] as its only value.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00099">

<sch:rule context="*[@ism:ownerProducer]">
<sch:let name="opTok" value="tokenize(string(..@ism:ownerProducer),' ')" />
<sch:assert id="ISM-00099"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not(index-of($opTok,'FGI')>0 and
count($opTok)>1) "
flag="error">
[ISM-ID-00099][Error] If ISM_CAPCO_RESOURCE attribute ownerProducer contains [FGI],
it must be the only value.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00100

FileName:./Rules/ownerProducer/ISM\_ID\_00100.sch

### Rule Description:

[ISM-ID-00100][Error] If ISM\_CAPCO\_RESOURCE and attribute ownerProducer is specified, then each of its values must be ordered in accordance with CVEnumISMOwnerProducer.xml.

### Code Description:

To perform sorting, this rule first retrieves the CVE values for the attribute to be sorted, which in this case is ownerProducer. Then, each attribute token is converted into a numerical value based on its characters. Next, each attribute token is given an order number, which compares its position to that of its value in the CVE file. Next, each order number is compared to that of its previous sibling to determine if the tokens are in order. If a token is found whose order number is less than that of its previous sibling, 0 is returned for its sorted order number. If a token's order number is greater than that of its previous sibling, 1 is returned. If two tokens have the same order number, their original attribute values are compared. If the original attribute value contains numbers then the comparison is made on its converted numerical value; otherwise, the comparison is made on its string value. If an attribute value is found whose value is less than that of its previous sibling, 0 is returned for its sorted order number; otherwise 2 is returned. Finally, if any tokens are found with 0 as its sorted order number, then the rule fails as those tokens are out of order.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00100">

  <sch:rule context="*[@ism:ownerProducer]">
    <!-- Define variables -->
    <sch:let name="errMsg_AlphabeticalOrder"
value="' [ISM-ID-00100][Error] If ISM_CAPCO_RESOURCE and attribute ownerProducer
is specified, then each of its values must be ordered in accordance with
CVEnumISMOwnerProducer.xml. '"/>

    <sch:let name="dataFileElems" value="$ownerProducerList"/>
    <sch:let name="attrValues" value="./@ism:ownerProducer"/>
    <sch:let name="attrValueTokens" value="tokenize(string($attrValues),' ')/>

    <!-- Convert each character to a numerical value, then concatenate the results to form a
number-string -->
    <sch:let name="convertStrToNum"
value=" for $token in $attrValueTokens return number(string-join( for $index in 1
to string-length($token) return for $char in substring($token, $index, 1) return if
(contains(string('0123456789'), $char)) then $char else if (contains(string('ABCDEFGHI'),
$char)) then translate(string($char), 'ABCDEFGHI', '123456789') else if
(contains(string('JKLMNOPQRS'), $char)) then concat('1', translate(string($char),
'JKLMNOPQRS', '0123456789')) else if (contains(string('TUVWXYZ'), $char)) then concat('2',
translate(string($char), 'TUVWXYZ', '0123456')) else '0' , '')) "/>
```

```

<!-- Get the position of each client node relative to its position in the master list. If
the node is not found, return a -1 -->
<sch:let name="orderNums"
value=" for $token in $attrValueTokens return if
($dataFileElems[matches($token,concat('^',text(),'$'))]) then
count(($dataFileElems[matches($token,concat('^',text(),'$'))]/preceding::*) + 1 else -1"/
>

<!-- Create a sequence that returns a 0 if the previous sibling has a higher order number,
else return a 1 -->
<sch:let name="sortedOrderNums"
value=" for $index in distinct-values(for $token in $orderNums return index-of($orderNums,
$token)) return if($index = 1 or $orderNums[$index] > $orderNums[$index - 1])
then 1 else if ($orderNums[$index] < $orderNums[$index - 1]) then 0 else if
(matches($attrValueTokens[$index], '[0-9]') or matches($attrValueTokens[$index - 1],
'[0-9]')) then if ($convertStrToNum[$index - 1] > $convertStrToNum[$index]) then 0 else
2 else if (compare($attrValueTokens[$index - 1],$attrValueTokens[$index])=1) then 0 else 2
"/>
<sch:let name="hasUnsorted" value="count(index-of($sortedOrderNums,0)) > 0"/>
<sch:let name="unsortedValues"
value=" if ($hasUnsorted) then distinct-values( for $token in index-of($sortedOrderNums,0)
return $attrValueTokens[$token] ) else null "/>

<sch:assert id="ISM-00100" test="not($hasUnsorted)" flag="error">
<sch:value-of select="$errMsg_AlphabeticalOrder"/>
The following values are out of order [<sch:value-of select="$unsortedValues"/>] for
[<sch:value-of select="$attrValueTokens"/>] </sch:assert>
</sch:rule>
</sch:pattern>

```



## Rule: ISM-ID-00102

FileName:../Rules/generalConstraints/ISM\_ID\_00102.sch

### Rule Description:

[ISM-ID-00102][Error] The root element must have the attribute DESVersion in the namespace urn:us:gov:ic:ism. Human Readable: The data encoding specification version number must be specified on the resource node.

### Code Description:

The code checks that the root node has attribute DESVersion specified.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00102">

<sch:rule context="/*">
<sch:assert id="ISM-00102" test=" ./@ism:DESVersion " flag="error">
[ISM-ID-00102][Error] The root element must have the attribute
DESVersion in the namespace urn:us:gov:ic:ism.

Human Readable: The data encoding specification version number must be specified on the
resource node.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00103

FileName:../Rules/generalConstraints/ISM\_ID\_00103.sch

### Rule Description:

[ISM-ID-00103][Error] At least one element must have resourceElement true.

### Code Description:

The code loops over all elements which have ISM attributes present and counts the elements which specify the attribute resourceElement. Then it makes sure that the total is greater than zero.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00103">

<sch:rule context="/*">
<sch:assert id="ISM-00103"
test=" count( for $token in /*[(@ism:*)] return if($token/@ism:resourceElement=true())
then 1 else null ) > 0 "
flag="error">
[ISM-ID-00103][Error] At least one element must have resourceElement true.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00104

FileName:./Rules/resourceElement/ISM\_ID\_00104.sch

### Rule Description:

[ISM-ID-00104][Error] If ISM\_CAPCO\_RESOURCE and any element in the document: 1. Meets ISM\_CONTRIBUTES AND 2. Has the attribute nonICmarkings containing [SBU-NF] AND 3. The ISM\_RESOURCE\_ELEMENT has attribute classification equal to [U] Then the ISM\_RESOURCE\_ELEMENT must have nonICmarkings containing [SBU-NF]. Human Readable: Unclassified USA documents having SBU-NF must have SBU-NF at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute nonICmarkings specified with a value containing [SBU-NF] and the resourceElement has the attribute classification specified with a value of [U], then we make sure that the resourceElement has attribute nonICmarkings specified with a value other than [SBU-NF].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00104">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00104"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partNonICmarkings_tok,
'SBU-NF') > 0 and $bannerClassification='U') then (index-of($bannerNonICmarkings_tok,
'SBU-NF') > 0) else true() "
flag="error">
[ISM-ID-00104][Error] Unclassified USA documents having SBU-NF must have SBU-NF at the
resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00105

FileName:./Rules/resourceElement/ISM\_ID\_00105.sch

### Rule Description:

[ISM-ID-00105][Error] If ISM\_CAPCO\_RESOURCE and any element in the document: 1. Meets ISM\_CONTRIBUTES AND 2. Has the attribute nonICmarkings containing [SBU] AND 3. The ISM\_RESOURCE\_ELEMENT has attribute classification equal to [U] Then the ISM\_RESOURCE\_ELEMENT must have nonICmarkings containing [SBU]. Human Readable: Unclassified USA documents having SBU must have SBU at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute nonICmarkings specified with a value containing [SBU] and the resourceElement has the attribute classification specified with a value of [U], then we make sure that the resourceElement has attribute nonICmarkings specified with a value of [SBU].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00105">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00105"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partNonICmarkings_tok,
'SBU') > 0 and $bannerClassification='U') then index-of($bannerNonICmarkings_tok,
'SBU') > 0 else true() "
flag="error">
[ISM-ID-00105][Error] Unclassified USA documents having SBU must have SBU at the resource
level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00107

FileName:./Rules/disseminationControls/ISM\_ID\_00107.sch

### Rule Description:

[ISM-ID-00107][Error] If ISM\_CAPCO\_RESOURCE and attribute disseminationControls contains the name token [IMC] then attribute classification must have a value of [TS] or [S]. Human Readable: IMCON data is SECRET (S), but may appear with S or TOP SECRET data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check the attribute classification of the element and return true if it has a value of [S] or [TS]. Otherwise we check that the attribute disseminationControls does not contain the value [IMC].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00107">

<sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00107"
test=" if(matches(./@ism:classification,'^(TS|S)$')) then true() else not(index-
of(tokenize(string(./@ism:disseminationControls), ' '), 'IMC')>0) "
flag="error">
[ISM-ID-00107][Error] IMCON data is SECRET (S), but may appear with S or TOP SECRET data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00108

FileName:./Rules/resourceElement/ISM\_ID\_00108.sch

### Rule Description:

[ISM-ID-00108][Error] If ISM\_CAPCO\_RESOURCE and attribute classification of ISM\_RESOURCE\_ELEMENT has a value of [TS] and attribute compilationReason does not have a value then at least one element meeting ISM\_CONTRIBUTES in the document must have a classification attribute of [TS]. Human Readable: USA TS documents not using compilation must have TS data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. Then make sure we are looking at the resourceElement and if the resourceElement has attribute classification specified with a value of [TS] and the resourceElement has attribute compilationReason specified with an empty value, then we make sure that at least one element in the document has attribute classification specified with a value of [TS].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00108">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00108"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if ($bannerClassification='TS' and
not(string-length(normalize-space(string(./@ism:compilationReason)))>0)) then index-
of($partClassification_tok,'TS')>0 else true() "
flag="error">
[ISM-ID-00108][Error] USA TS documents not using compilation must have TS data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00109

FileName:./Rules/resourceElement/ISM\_ID\_00109.sch

### Rule Description:

[ISM-ID-00109][Error] If ISM\_CAPCO\_RESOURCE and attribute classification of ISM\_RESOURCE\_ELEMENT has a value of [S] and attribute compilationReason does not have a value then at least one element meeting ISM\_CONTRIBUTES in the document must have a classification attribute of [S]. Human Readable: USA S documents not using compilation must have S data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute classification specified with a value of [S] and the resourceElement has attribute compilationReason specified with an empty value, then we make sure that at least one element in the document has attribute classification specified with a value of [S].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00109">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00109"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if ($bannerClassification='S' and
not(string-length(normalize-space(string(./@ism:compilationReason)))>0)) then index-
of($partClassification_tok,'S')>0 else true() "
flag="error">
[ISM-ID-00109][Error] USA S documents not using compilation must have S data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00110

FileName:./Rules/resourceElement/ISM\_ID\_00110.sch

### Rule Description:

[ISM-ID-00110][Error] If ISM\_CAPCO\_RESOURCE and attribute classification of ISM\_RESOURCE\_ELEMENT has a value of [C] and attribute compilationReason does not have a value then at least one element meeting ISM\_CONTRIBUTES in the document must have a classification attribute of [C]. Human Readable: USA C documents not using compilation must have C data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute classification specified with a value of [C] and the resourceElement has attribute compilationReason specified with an empty value, then we make sure that at least one element in the document has attribute classification specified with a value of [C].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00110">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00110"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if ($bannerClassification='C' and
not(string-length(normalize-space(string(..@ism:compilationReason)))>0)) then index-
of($partClassification_tok,'C')>0 else true() "
flag="error">
[ISM-ID-00110][Error] USA C documents not using compilation must have C data.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00111

FileName:./Rules/resourceElement/ISM\_ID\_00111.sch

### Rule Description:

[ISM-ID-00111][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols of ISM\_RESOURCE\_ELEMENT contains [SI] and attribute compilationReason does not have a value then at least one element meeting ISM\_CONTRIBUTES in the document must have a SCIcontrols attribute containing [SI]. Human Readable: USA SI documents not using compilation must have SI data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute SCIcontrols specified with a value containing [SI] and the resourceElement has attribute compilationReason specified with an empty value, then we make sure that at least one element in the document has attribute SCIcontrols specified with a value containing [SI].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00111">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00111"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if (index-of($bannerSCIcontrols_tok,
'SI')>0 and not(string-length(normalize-space(string(./@ism:compilationReason)))>0))
then index-of($partSCIcontrols_tok,'SI')>0 else true() "
flag="error">
[ISM-ID-00111][Error] USA SI documents not using compilation must have SI data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00112

FileName:./Rules/resourceElement/ISM\_ID\_00112.sch

### Rule Description:

[ISM-ID-00112][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols of ISM\_RESOURCE\_ELEMENT contains [SI-G] and attribute compilationReason does not have a value then at least one element meeting ISM\_CONTRIBUTES in the document must have a SCIcontrols attribute containing [SI-G]. Human Readable: USA SI-G documents not using compilation must have SI-G data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute SCIcontrols specified with a value containing [SI-G] and the resourceElement has attribute compilationReason specified with an empty value, then we make sure that at least one element in the document has attribute SCIcontrols specified with a value containing [SI-G].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00112">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00112"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if
(contains(string($bannerSCIcontrols), 'SI-G') and not(string-length(normalize-
space(string(./@ism:compilationReason)))>0)) then index-of($partSCIcontrols_tok,'SI-
G')>0 else true() "
flag="error">
[ISM-ID-00112][Error] USA SI-G documents not using compilation must have SI-G data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00113

FileName:./Rules/resourceElement/ISM\_ID\_00113.sch

### Rule Description:

[ISM-ID-00113][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols of ISM\_RESOURCE\_ELEMENT contains [TK] and attribute compilationReason does not have a value then at least one element meeting ISM\_CONTRIBUTES in the document must have a SCIcontrols attribute containing [TK]. Human Readable: USA TK documents not using compilation must have TK data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute SCIcontrols specified with a value containing [TK] and the resourceElement has attribute compilationReason specified with an empty value, then we make sure that at least one element in the document has attribute SCIcontrols specified with a value containing [TK].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00113">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00113"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if (index-of($bannerSCIcontrols_tok,
'TK')>0 and not(string-length(normalize-space(string(./@ism:compilationReason)))>0))
then index-of($partSCIcontrols_tok,'TK')>0 else true() "
flag="error">
[ISM-ID-00113][Error] USA TK documents not using compilation must have TK data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00116

FileName:./Rules/resourceElement/ISM\_ID\_00116.sch

### Rule Description:

[ISM-ID-00116][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols of ISM\_RESOURCE\_ELEMENT contains [HCS] and attribute compilationReason does not have a value then at least one element meeting ISM\_CONTRIBUTES in the document must have a SCIcontrols attribute containing [HCS]. Human Readable: USA HCS documents not using compilation must have HCS data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute SCIcontrols specified with a value containing [HCS] and the resourceElement has attribute compilationReason specified with an empty value, then we make sure that at least one element in the document has attribute SCIcontrols specified with a value containing [HCS].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00116">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00116"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if (index-
of($bannerSCIcontrols_tok, 'HCS')>0 and not(string-length(normalize-space(string(./
@ism:compilationReason)))>0)) then index-of($partSCIcontrols_tok, 'HCS')>0 else
true() "
flag="error">
[ISM-ID-00116][Error] USA HCS documents not using compilation must have HCS data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00118

FileName:../Rules/resourceElement/ISM\_ID\_00118.sch

### Rule Description:

[ISM-ID-00118][Error] The first element in document order having resourceElement true must have createDate specified.

### Code Description:

We make sure that the resourceElement has attribute createDate specified.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00118">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)][1]">
<sch:assert id="ISM-00118"
test=" if(./@ism:createDate) then true() else false() "
flag="error">
[ISM-ID-00118][Error] The first element in document order having
resourceElement true must have createDate specified.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00119

FileName:./Rules/generalConstraints/ISM\_ID\_00119.sch

### Rule Description:

[ISM-ID-00119][Error] If ISM\_CAPCO\_RESOURCE and 1. attribute classification is not [U] AND 2. ISM\_ICDOCUMENT\_APPLIES AND 3. Attribute disseminationControls must contain one or more of [DISPLAYONLY], [REL], [RELIDO], [EYES], or [NF] Human Readable: All classified NSI that claims compliance with ICD 710 must have an appropriate foreign disclosure or release marking.

### Code Description:

If CAPCO rules do not apply to the document, or ICDocument does not apply to the document, or the resource is unclassified, then the rule does not apply. Otherwise, we make sure that the attribute disseminationControls contains at least one of the values [DISPLAYONLY], [RELIDO], [REL], [EYES], or [NF].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00119">

<sch:rule context="*[@ism:* except (@ism:pocType | @ism:DESVersion |
@ism:unregisteredNoticeType) and $ISM_CAPCO_RESOURCE and $ISM_ICDOCUMENT_APPLIES and
not(@ism:classification='U')]">
<sch:assert id="ISM-00119"
test="some $token in tokenize(string(@ism:disseminationControls), ' ') satisfies
$token=('RELIDO','REL','EYES','DISPLAYONLY','NF') "
flag="error">
[ISM-ID-00119][Error] All classified NSI that claims compliance with ICD 710 must have an
appropriate
foreign disclosure or release marking.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00121

FileName:../Rules/SARIdentifier/ISM\_ID\_00121.sch

### Rule Description:

[ISM-ID-00121][Error] If ISM\_CAPCO\_RESOURCE and attribute SARIdentifier is specified, then each of its values must be ordered in accordance with CVEnumISMSAR.xml.

### Code Description:

To perform sorting, this rule first retrieves the CVE values for the attribute to be sorted, which in this case is SARIdentifier. Then, each attribute token is converted into a numerical value based on its characters. Next, each attribute token is given an order number, which compares its position to that of its value in the CVE file. Next, each order number is compared to that of its previous sibling to determine if the tokens are in order. If a token is found whose order number is less than that of its previous sibling, 0 is returned for its sorted order number. If a token's order number is greater than that of its previous sibling, 1 is returned. If two tokens have the same order number, their original attribute values are compared. If the original attribute value contains numbers then the comparison is made on its converted numerical value; otherwise, the comparison is made on its string value. If an attribute value is found whose value is less than that of its previous sibling, 0 is returned for its sorted order number; otherwise 2 is returned. Finally, if any tokens are found with 0 as its sorted order number, then the rule fails as those tokens are out of order.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00121">

<sch:rule context="*[@ism:SARIdentifier]">
<!-- Define variables -->
<sch:let name="errMsg_AlphabeticalOrder"
value="' [ISM-ID-00121][Error] If ISM_CAPCO_RESOURCE and attribute SARIdentifier is
specified, then each of its values must be ordered in accordance with CVEnumISMSAR.xml.'" />
>

<sch:let name="dataFileElems" value="$SARIdentifierList"/>
<sch:let name="attrValues" value="./@ism:SARIdentifier"/>
<sch:let name="attrValueTokens" value="tokenize(string($attrValues), ' ')" />

<!-- Convert each character to a numerical value, then concatenate the results to form a
number-string -->
<sch:let name="convertStrToNum"
value=" for $token in $attrValueTokens return number(string-join( for $index in 1
to string-length($token) return for $char in substring($token, $index, 1) return if
(contains(string('0123456789'), $char)) then $char else if (contains(string('ABCDEFGHI'),
$char)) then translate(string($char), 'ABCDEFGHI', '123456789') else if
(contains(string('JKLMNOPQRS'), $char)) then concat('1', translate(string($char),
'JKLMNOPQRS', '0123456789')) else if (contains(string('TUVWXYZ'), $char)) then concat('2',
translate(string($char), 'TUVWXYZ', '0123456')) else '0' , '')) " />
```

```

<!-- Get the position of each client node relative to its position in the master list. If
the node is not found, return a -1 -->
<sch:let name="orderNums"
value=" for $token in $attrValueTokens return if
($dataFileElems[matches($token,concat('^',text(),'$'))]) then
count(($dataFileElems[matches($token,concat('^',text(),'$'))])/preceding::* ) + 1 else -1"/
>

<!-- Create a sequence that returns a 0 if the previous sibling has a higher order number,
else return a 1 -->
<sch:let name="sortedOrderNums"
value=" for $index in distinct-values(for $token in $orderNums return index-of($orderNums,
$token)) return if($index = 1 or $orderNums[$index] > $orderNums[$index - 1])
then 1 else if ($orderNums[$index] < $orderNums[$index - 1]) then 0 else if
(matches($attrValueTokens[$index], '[0-9]') or matches($attrValueTokens[$index - 1],
'[0-9]')) then if ($convertStrToNum[$index - 1] > $convertStrToNum[$index]) then 0 else
2 else if (compare($attrValueTokens[$index - 1],$attrValueTokens[$index])=1) then 0 else 2
"/>
<sch:let name="hasUnsorted" value="count(index-of($sortedOrderNums,0)) > 0"/>
<sch:let name="unsortedValues"
value=" if ($hasUnsorted) then distinct-values( for $token in index-of($sortedOrderNums,0)
return $attrValueTokens[$token] ) else null "/>

<sch:assert id="ISM-00121" test="not($hasUnsorted)" flag="error">
<sch:value-of select="$errMsg_AlphabeticalOrder"/>
The following values are out of order [<sch:value-of select="$unsortedValues"/>] for
[<sch:value-of select="$attrValueTokens"/>] </sch:assert>
</sch:rule>

</sch:pattern>

```



## Rule: ISM-ID-00122

FileName:./Rules/SCIcontrols/ISM\_ID\_00122.sch

### Rule Description:

[ISM-ID-00122][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols contains the name token [KDK], then attribute classification must have a value of [TS] or [S]. Human Readable: A USA document with KLONDIKE data must be classified SECRET or TOP SECRET.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIcontrols specified with a value containing [KDK], then we make sure that attribute classification has a value [TS] or [S].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00122">

<sch:rule context="*[@ism:SCIcontrols]">
<sch:assert id="ISM-00122"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not(index-of(tokenize(string(./
@ism:SCIcontrols), ' '), 'KDK')>0 and not(matches(./@ism:classification, '^(TS|S)$')) ) "
flag="error">
[ISM-ID-00122][Error] If ISM_CAPCO_RESOURCE and attribute SCIcontrols contains the name
token [KDK], then attribute
classification must have a value of [TS] or [S].

Human Readable: A USA document with KLONDIKE data must be classified SECRET or TOP SECRET.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00123

FileName:./Rules/SCIcontrols/ISM\_ID\_00123.sch

### Rule Description:

[ISM-ID-00123][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols contains the name token [KDK], then attribute disseminationControls must contain the name token [NF]. Human Readable: A USA document containing KLONDIKE data must also be marked for NO FOREIGN dissemination.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIcontrols specified with a value containing [KDK], then we make sure that attribute disseminationControls contains the value [NF].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00123">

<sch:rule context="*[@ism:SCIcontrols]">
<sch:assert id="ISM-00123"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of(tokenize(string(./
@ism:SCIcontrols), ' '), 'KDK')>0 and not( index-of(tokenize(string(./
@ism:disseminationControls), ' '), 'NF')) ) then false() else true() "
flag="error">
[ISM-ID-00123][Error] If ISM_CAPCO_RESOURCE and attribute SCIcontrols contains the name
token [KDK], then attribute
disseminationControls must contain the name token [NF].

Human Readable: A USA document containing KLONDIKE data must also be marked for NO FOREIGN
dissemination.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00124

FileName:./Rules/disseminationControls/ISM\_ID\_00124.sch

### Rule Description:

[ISM-ID-00124][Warning] If ISM\_CAPCO\_RESOURCE and 1. Attribute ownerProducer does not contain [USA]. AND 2. Attribute disseminationControls contains [RELIDO] Human Readable: RELIDO is not authorized for non-US portions.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check the attribute ownerProducer for not having a value of [USA] and that the attribute disseminationControls contains a value of [RELIDO] then we return false because the resource is not in compliance with the rule.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00124">

<sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00124"
test=" if(not(index-of(tokenize(string(..@ism:ownerProducer),' '), 'USA')>0) and index-
of(tokenize(string(..@ism:disseminationControls),' '), 'RELIDO')>0) then false() else
true() "
flag="warning">
[ISM-ID-00124][Warning] RELIDO is not authorized for non-US portions.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00125

FileName:./Rules/generalConstraints/ISM\_ID\_00125.sch

### Rule Description:

[ISM-ID-00125][Error] If any attributes in namespace urn:us:gov:ic:ism exist, the local name must exist in CVEnumISMAttributes.xml. Human Readable: Ensure that attributes in the ISM namespace are defined by ISM.XML.

### Code Description:

To determine the valid values, this rule first retrieves the list of valid attribute names as defined in CVEnumISMAttributes.xml. Next, each attribute token is given an order number, which compares its position to that of its value in the CVE file. If the token is not found, its order number will be -1 and it is considered invalid. If the document is a CAPCO resource, then the rule will fail if invalid tokens are found. The rule will also fail if duplicate values are found for an attribute name.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00125">

  <sch:rule context="*[@ism:* and $ISM_CAPCO_RESOURCE]">
    <!-- Define variables -->
    <sch:let name="errMsg_ValueNotFound"
value="' [ISM-ID-00125][Error] If any attributes in namespace urn:us:gov:ic:ism exist, the
local name must exist in CVEnumISMAttributes.xml. '"/>

    <sch:let name="dataFileElems" value="$validAttributeList"/>
    <sch:let name="attrValues" value="string-join(./@ism:*/local-name(),' ')/>
    <sch:let name="attrValueTokens" value="tokenize(string($attrValues),' ')/>

    <!-- Get the position of each client node relative to its position in the master list. If
the node is not found, return a -1 -->
    <sch:let name="orderNums"
value=" for $token in $attrValueTokens return if
($dataFileElems[matches($token,concat('^',text(),'$'))]) then
count(($dataFileElems[matches($token,concat('^',text(),'$'))])/preceding::*) + 1 else -1"/>

    <!-- Determine if the list has invalid values. If and only if it does, figure out which
ones are invalids -->
    <sch:let name="hasInvalids" value="count(index-of($orderNums,-1)) > 0"/>
    <sch:let name="invalidValues"
value=" if ($hasInvalids) then distinct-values( for $token in index-of($orderNums,-1)
return $attrValueTokens[$token] ) else null "/>

  </sch:rule>
</sch:pattern>
```

```

<!-- Determine if the list has duplicate values. If and only if it does, figure out which
ones are duplicates -->
<sch:let name="hasDups"
value="count(distinct-values($attrValueTokens)) != count($attrValueTokens)"/>
<sch:let name="dupValues"
value=" if ($hasDups) then distinct-values( for $token in $attrValueTokens return
if (count(index-of($attrValueTokens,$token)) > 1) then $attrValueTokens[index-
of($attrValueTokens,$token)[1]] else null ) else null "/>

<!-- Execute tests -->
<sch:assert id="ISM-00125" flag="error" test="not($hasInvalids)">
<sch:value-of select="$errMsg_ValueNotFound"/>
Invalid value of [<sch:value-of select="$invalidValues"/>]</sch:assert>
<sch:assert test="not($hasDups)" flag="undefined">Duplicate values found [<sch:value-of
select="$dupValues"/>] for [<sch:value-of select="$attrValueTokens"/>] </sch:assert>
</sch:rule>
</sch:pattern>

```

## Rule: ISM-ID-00126

FileName:./Rules/generalConstraints/ISM\_ID\_00126.sch

### Rule Description:

[ISM-ID-00126][Error] Attributes with namespace urn:us:gov:ic:ism must not appear with attribute @xs:any. Human Readable: Ensure that no attributes that appear to be in the ISM namespace, but are not defined by ISM.XML, are used in a schema that might have an xs:any.

### Code Description:

The code checks that any element having ISM attributes does not have the attribute xs:any specified.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00126">

<sch:rule context="*[@ism:*]">
<sch:assert id="ISM-00126" test=" not(./@xs:any) " flag="error">
[ISM-ID-00126][Error] Attributes with namespace urn:us:gov:ic:ism must
not appear with attribute @xs:any.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00127

FileName:../Rules/notice/ISM\_ID\_00127.sch

### Rule Description:

[ISM-ID-00127][Error] If ISM\_CAPCO\_RESOURCE and: 1. Any element meeting ISM\_CONTRIBUTES in the document has the attribute atomicEnergyMarkings containing [RD] AND 2. No element meeting ISM\_CONTRIBUTES in the document has notice containing [RD]. Human Readable: USA documents containing RD data must also have an RD notice.

### Code Description:

If CAPCO rules do not apply to the document and the element is not excluded from the rollup then the rule does not apply and we return true. If the current element has attribute atomicEnergyMarkings specified with a value containing [RD], then we make sure that attribute notice is specified with a value containing [RD] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00127">

<sch:rule context="*[@ism:atomicEnergyMarkings]">
<sch:assert id="ISM-00127"
test=" if(not($ISM_CAPCO_RESOURCE)or ./@ism:excludeFromRollup=true()) then true() else
if(index-of(tokenize(string(./@ism:atomicEnergyMarkings), ' '), 'RD')>0) then index-
of($partNotice_tok, 'RD')>0 else true() "
flag="error">
[ISM-ID-00127][Error] If ISM_CAPCO_RESOURCE and:
1. Any element meeting ISM_CONTRIBUTES in the document has the attribute
atomicEnergyMarkings containing [RD]
AND
2. No element meeting ISM_CONTRIBUTES in the document has notice containing [RD].

Human Readable: USA documents containing RD data must also have an RD notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00128

FileName:./Rules/notice/ISM\_ID\_00128.sch

### Rule Description:

[ISM-ID-00128][Error] If ISM\_CAPCO\_RESOURCE and: 1. Any element meeting ISM\_CONTRIBUTES in the document has the attribute atomicEnergyMarkings containing [FRD] AND 2. No element meeting ISM\_CONTRIBUTES in the document has notice containing [FRD] Human Readable: USA documents containing FRD data must also have an FRD notice.

### Code Description:

If CAPCO rules do not apply to the document and the element is not excluded from the rollup then the rule does not apply and we return true. If the current element has attribute atomicEnergyMarkings specified with a value containing [FRD], then we make sure that attribute notice is specified with a value containing [FRD] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00128">

<sch:rule context="*[@ism:atomicEnergyMarkings]">
<sch:assert id="ISM-00128"
test=" if(not($ISM_CAPCO_RESOURCE) or ./@ism:excludeFromRollup=true()) then true() else
if(index-of(tokenize(string(./@ism:atomicEnergyMarkings), ' '), 'FRD')>0) then index-
of($partNotice_tok, 'FRD')>0 else true() "
flag="error">
[ISM-ID-00128][Error] If ISM_CAPCO_RESOURCE and:
1. Any element meeting ISM_CONTRIBUTES in the document has the attribute
atomicEnergyMarkings containing [FRD]
AND
2. No element meeting ISM_CONTRIBUTES in the document has notice containing [FRD]

Human Readable: USA documents containing FRD data must also have an FRD notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00129

FileName:./Rules/notice/ISM\_ID\_00129.sch

### Rule Description:

[ISM-ID-00129][Error] If ISM\_CAPCO\_RESOURCE and: 1. Any element meeting ISM\_CONTRIBUTES in the document has the attribute disseminationControls containing [IMC] AND 2. No element meeting ISM\_CONTRIBUTES in the document has notice containing [IMC] Human Readable: USA documents containing IMC data must also have an IMC notice.

### Code Description:

If CAPCO rules do not apply to the document and the element is not excluded from the rollup then the rule does not apply and we return true. If the current element has attribute disseminationControls specified with a value containing [IMC], then we make sure that attribute notice is specified with a value containing [IMC] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00129">

<sch:rule context="*[@ism:disseminationControls]">
<sch:assert id="ISM-00129"
test=" if(not($ISM_CAPCO_RESOURCE) or ./@ism:excludeFromRollup=true()) then true() else
if(index-of(tokenize(string(./@ism:disseminationControls), ' '), 'IMC')>0) then index-
of($partNotice_tok, 'IMC')>0 else true() "
flag="error">
[ISM-ID-00129][Error] If ISM_CAPCO_RESOURCE and:
1. Any element meeting ISM_CONTRIBUTES in the document has the attribute
disseminationControls containing [IMC]
AND
2. No element meeting ISM_CONTRIBUTES in the document has notice containing [IMC]
Human Readable: USA documents containing IMC data must also have an IMC notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00130

FileName:./Rules/notice/ISM\_ID\_00130.sch

### Rule Description:

[ISM-ID-00130][Error] If ISM\_CAPCO\_RESOURCE and: 1. Any element meeting ISM\_CONTRIBUTES in the document has the attribute disseminationControls containing [FISA] AND 2. No element meeting ISM\_CONTRIBUTES in the document has notice containing [FISA] Human Readable: USA documents containing FISA data must also have an FISA notice.

### Code Description:

If CAPCO rules do not apply to the document and the element is not excluded from the rollup then the rule does not apply and we return true. If the current element has attribute disseminationControls specified with a value containing [FISA], then we make sure that attribute notice is specified with a value containing [FISA] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00130">

<sch:rule context="*[@ism:disseminationControls]">
<sch:assert id="ISM-00130"
test=" if(not($ISM_CAPCO_RESOURCE) or ./@ism:excludeFromRollup=true()) then true() else
if(index-of(tokenize(string(./@ism:disseminationControls), ' '), 'FISA')>0) then index-
of($partNotice_tok, 'FISA')>0 else true() "
flag="error">
[ISM-ID-00130][Error] If ISM_CAPCO_RESOURCE and:
1. Any element meeting ISM_CONTRIBUTES in the document has the attribute
disseminationControls containing [FISA]
AND
2. No element meeting ISM_CONTRIBUTES in the document has notice containing [FISA]

Human Readable: USA documents containing FISA data must also have an FISA notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00132

FileName:./Rules/resourceElement/ISM\_ID\_00132.sch

### Rule Description:

[ISM-ID-00132][Error] If ISM\_CAPCO\_RESOURCE and the ISM\_RESOURCE\_ELEMENT has the attribute disseminationControls containing [RELIDO] then every element meeting ISM\_CONTRIBUTES\_CLASSIFIED in the document must have the attribute disseminationControls containing [RELIDO]. Human Readable: USA documents having RELIDO at the resource level must have every classified portion having RELIDO.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute disseminationControls specified with a value containing [RELIDO], then we make sure that the number of elements in the document that have attribute classification specified with a value other than [U] and attribute disseminationControls specified with a value containing [RELIDO] is the same as the number of elements in the document that have attribute classification specified with a value other than [U].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00132">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00132"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-
of($bannerDisseminationControls_tok, 'RELIDO')>0) then sum(for $token in $partTags
return if(not($token/@ism:classification='U') and index-of(tokenize(string($token/
@ism:disseminationControls),' '), 'RELIDO')>0) then 1 else 0 ) = count(for $tag in
$partTags return if($tag/@ism:classification='U') then null else 1) else true() "
flag="error">
[ISM-ID-00132][Error] USA documents having RELIDO at the resource level must have every
classified portion having RELIDO.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00133

FileName:./Rules/declassException/ISM\_ID\_00133.sch

### Rule Description:

[ISM-ID-00133][Error] If ISM\_NSI\_EO\_APPLIES and attribute declassException is specified and contains the tokens [25X1-human], [50X1-HUM], or [50X2-WMD], then attribute declassDate or declassEvent must NOT be specified. Human Readable: Documents under E.O. 13526 must not specify declassDate or declassEvent if a declassException of 25X1-human, 50X1-HUM, or 50X2-WMD is specified.

### Code Description:

If current Classified National Security Information Executive Order does not apply to this resource then the rule does not apply and we return true. Otherwise we ensure that any element with the attribute declassException specified with a value of [25X1-human], [50X1-HUM], or [50X1-WMD], then we ensure that the attribute declassDate or attribute declassEvent are not specified.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00133">

<sch:rule context="*[@ism:declassException and $ISM_NSI_EO_APPLIES]">
<sch:let name="dd" value="exists(./@ism:declassDate)"/>
<sch:let name="de" value="exists(./@ism:declassEvent)"/>
<sch:let name="paddedDeclassEx"
value="concat(' ', normalize-space(string(./@ism:declassException)), ' ')">

<sch:assert id="ISM-00133"
test=" if(not(matches($paddedDeclassEx, ' (25X1-human|50X1-HUM|50X2-WMD) '))) then true()
else not($dd or $de) "
flag="error">
[ISM-ID-00133][Error] If ISM_NSI_EO_APPLIES and attribute
declassException is specified and contains the tokens [25X1-human],
[50X1-HUM], or [50X2-WMD], then attribute declassDate or
declassEvent must NOT be specified. Invalid presence of
<sch:value-of select="if($dd) then 'declassDate' else null"/>
<sch:value-of select="if($dd and $de) then ' and ' else null"/>
<sch:value-of select="if($de) then 'declassEvent' else null"/>.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00134

FileName:./Rules/notice/ISM\_ID\_00134.sch

### Rule Description:

[ISM-ID-00134][Error] If ISM\_CAPCO\_RESOURCE and: 1. Any element meeting ISM\_CONTRIBUTES in the document has the attribute nonICmarkings containing [DS] AND 2. No element meeting ISM\_CONTRIBUTES in the document has the attribute notice containing [DS] Human Readable: USA documents containing DS data must also have a DS notice.

### Code Description:

If CAPCO rules do not apply to the document and the element is not excluded from the rollup then the rule does not apply and we return true. If the current element has attribute nonICmarkings specified with a value containing [DS], then we make sure that attribute notice is specified with a value containing [DS] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00134">

<sch:rule context="*[@ism:nonICmarkings]">
<sch:assert id="ISM-00134"
test=" if(not($ISM_CAPCO_RESOURCE) or ./@ism:excludeFromRollup=true()) then true()
else if(index-of(tokenize(string(./@ism:nonICmarkings), ' '), 'DS')>0) then index-
of($partNotice_tok, 'DS')>0 else true() "
flag="error">
[ISM-ID-00134][Error] If ISM_CAPCO_RESOURCE and:
1. Any element meeting ISM_CONTRIBUTES in the document has the attribute nonICmarkings
containing [DS]
AND
2. No element meeting ISM_CONTRIBUTES in the document has the attribute notice containing
[DS]

Human Readable: USA documents containing DS data must also have a DS notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00135

FileName:./Rules/notice/ISM\_ID\_00135.sch

### Rule Description:

[ISM-ID-00135][Warning] If ISM\_CAPCO\_RESOURCE and: 1. No element meeting ISM\_CONTRIBUTES in the document has the attribute atomicEnergyMarkings containing [RD] AND 2. Any element meeting ISM\_CONTRIBUTES in the document has the attribute notice containing [RD] Human Readable: USA documents containing an RD notice must also have RD data.

### Code Description:

If CAPCO rules do not apply to the document and the element is not excluded from the rollup then the rule does not apply and we return true. If the current element is the resourceElement then the rule does not apply and we return true. If the current element has attribute notice specified with a value containing [RD], then we make sure that attribute atomicEnergyMarkings is specified with a value containing [RD] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00135">

<sch:rule context="*[@ism:noticeType]">
<sch:assert id="ISM-00135"
test=" if(not($ISM_CAPCO_RESOURCE) or ./@ism:excludeFromRollup=true()) then
true() else if(generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)) then true()
else if(index-of(tokenize(string(./@ism:noticeType), ' '), 'RD')>0 and not(./
@ism:excludeFromRollup=true())) then index-of($partAtomicEnergyMarkings_tok, 'RD')>0
else true() "
flag="warning">
[ISM-ID-00135][Warning] If ISM_CAPCO_RESOURCE and:
1. No element meeting ISM_CONTRIBUTES in the document has the attribute
atomicEnergyMarkings containing [RD]
AND
2. Any element meeting ISM_CONTRIBUTES in the document has the attribute notice containing
[RD]

Human Readable: USA documents containing RD data must also have an RD notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00136

FileName:./Rules/notice/ISM\_ID\_00136.sch

### Rule Description:

[ISM-ID-00136][Warning] If ISM\_CAPCO\_RESOURCE and: 1. No element without ism:excludeFromRollup=true() in the document has the attribute atomicEnergyMarkings containing [FRD] AND 2. Any element without ism:excludeFromRollup=true() in the document has the attribute notice containing [FRD] Human Readable: USA documents containing an FRD notice must also have FRD data.

### Code Description:

If CAPCO rules do not apply to the document and the element is not excluded from the rollup then the rule does not apply and we return true. If the current element has attribute notice specified with a value containing [FRD], then we make sure that attribute atomicEnergyMarkings is specified with a value containing [FRD] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00136">

<sch:rule context="*[@ism:noticeType]">
<sch:assert id="ISM-00136"
test=" if(not($ISM_CAPCO_RESOURCE) or ./@ism:excludeFromRollup=true()) then true()
else if(index-of(tokenize(string(./@ism:noticeType), ' '), 'FRD')>0 and not(./
@ism:excludeFromRollup=true())) then index-of($partAtomicEnergyMarkings_tok, 'FRD')>0
else true() "
flag="warning">
[ISM-ID-00136][Warning] If ISM_CAPCO_RESOURCE and:
1. No element without ism:excludeFromRollup=true() in the document has the attribute
atomicEnergyMarkings containing [FRD]
AND
2. Any element without ism:excludeFromRollup=true() in the document has the attribute
notice containing [FRD]

Human Readable: USA documents containing FRD data must also have an FRD notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00137

FileName:./Rules/notice/ISM\_ID\_00137.sch

### Rule Description:

[ISM-ID-00137][Warning] If ISM\_CAPCO\_RESOURCE and: 1. No element without ism:excludeFromRollup=true() in the document has the attribute disseminationControls containing [IMC] AND 2. Any element without ism:excludeFromRollup=true() in the document has the attribute notice containing [IMC] Human Readable: USA documents containing an IMC notice must also have IMC data.

### Code Description:

If CAPCO rules do not apply to the document and the element is not excluded from the rollup then the rule does not apply and we return true. If the current element has attribute notice specified with a value containing [IMC] and is not excluded from the rollup, then we make sure that attribute disseminationControls is specified with a value containing [IMC] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00137">

<sch:rule context="*[@ism:noticeType]">
<sch:assert id="ISM-00137"
test=" if(not($ISM_CAPCO_RESOURCE) or ./@ism:excludeFromRollup=true()) then true()
else if(index-of(tokenize(string(./@ism:noticeType), ' '), 'IMC')>0 and not(./
@ism:excludeFromRollup=true())) then index-of($partDisseminationControls_tok, 'IMC')>0
else true() "
flag="warning">
[ISM-ID-00137][Warning] If ISM_CAPCO_RESOURCE and:
1. No element without ism:excludeFromRollup=true() in the document has the attribute
disseminationControls containing [IMC]
AND
2. Any element without ism:excludeFromRollup=true() in the document has the attribute
notice containing [IMC]

Human Readable: USA documents containing IMC data must also have an IMC notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00138

FileName:./Rules/notice/ISM\_ID\_00138.sch

### Rule Description:

[ISM-ID-00138][Warning] If ISM\_CAPCO\_RESOURCE and: 1. No element without ism:excludeFromRollup=true() in the document has the attribute nonICmarkings containing [DS] AND 2. Any element without ism:excludeFromRollup=true() in the document has the attribute notice containing [DS] Human Readable: USA documents containing a DS notice must also have DS data.

### Code Description:

If CAPCO rules do not apply to the document and the element is not excluded from the rollup then the rule does not apply and we return true. If the current element has attribute notice specified with a value containing [DS] and is not excluded from the rollup, then we make sure that attribute nonICmarkings is specified with a value containing [DS] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00138">

<sch:rule context="*[@ism:noticeType]">
<sch:assert id="ISM-00138"
test=" if(not($ISM_CAPCO_RESOURCE) or ./@ism:excludeFromRollup=true()) then true()
else if(index-of(tokenize(string(./@ism:noticeType), ' '), 'DS')>0 and not(./
@ism:excludeFromRollup=true())) then index-of($partNonICmarkings_tok, 'DS')>0 else
true() "
flag="warning">
[ISM-ID-00138][Warning] If ISM_CAPCO_RESOURCE and:
1. No element without ism:excludeFromRollup=true() in the document has the attribute
nonICmarkings containing [DS]
AND
2. Any element without ism:excludeFromRollup=true() in the document has the attribute
notice containing [DS]

Human Readable: USA documents containing DS data must also have a DS notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00139

FileName:./Rules/notice/ISM\_ID\_00139.sch

### Rule Description:

[ISM-ID-00139][Warning] If ISM\_CAPCO\_RESOURCE and: 1. No element without ism:excludeFromRollup=true() in the document has the attribute disseminationControls containing [FISA] AND 2. Any element without ism:excludeFromRollup=true() in the document has the attribute notice containing [FISA] Human Readable: USA documents containing an FISA notice must also have FISA data.

### Code Description:

If CAPCO rules do not apply to the document and the element is not excluded from the rollup then the rule does not apply and we return true. If the current element has attribute notice specified with a value containing [FISA] and is not excluded from the rollup, then we make sure that attribute disseminationControls is specified with a value containing [FISA] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00139">

<sch:rule context="*[@ism:noticeType]">
<sch:assert id="ISM-00139"
test=" if(not($ISM_CAPCO_RESOURCE) or ./@ism:excludeFromRollup=true()) then true()
else if(index-of(tokenize(string(./@ism:noticeType), ' '), 'FISA')>0 and not(./
@ism:excludeFromRollup=true())) then index-of($partDisseminationControls_tok, 'FISA')>0
else true() "
flag="warning">
[ISM-ID-00139][Warning] If ISM_CAPCO_RESOURCE and:
1. No element without ism:excludeFromRollup=true() in the document has the attribute
disseminationControls containing [FISA]
AND
2. Any element without ism:excludeFromRollup=true() in the document has the attribute
notice containing [FISA]

Human Readable: USA documents containing FISA data must also have an FISA notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00140

FileName:../Rules/disseminationControls/ISM\_ID\_00140.sch

### Rule Description:

[ISM-ID-00140][Error] If ISM\_CAPCO\_RESOURCE and attribute disseminationControls contains the name token [NF], then attribute classification must not have a value of [U] Human Readable: NF may not be used on UNCLASSIFIED portions.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check the attribute classification of the element and return true if it has a value of [U]. Otherwise we check that the attribute disseminationControls does not contain the value [NF].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00140">

<sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00140"
test=" if(not(./@ism:classification='U')) then true() else not(index-of(tokenize(string(./
@ism:disseminationControls),' '), 'NF')>0) "
flag="error">
[ISM-ID-00140][Error] NF may not be used on UNCLASSIFIED portions.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00141

FileName:./Rules/resourceElement/ISM\_ID\_00141.sch

### Rule Description:

[ISM-ID-00141][Error] If ISM\_NSI\_EO\_APPLIES and 1. ISM\_RESOURCE\_ELEMENT attribute declassException does not have a value of [25X1-human], [50X1-HUM], or [50X2-WMD] AND 2. ISM\_RESOURCE\_ELEMENT attribute declassDate is not specified AND 3. ISM\_RESOURCE\_ELEMENT attribute declassEvent is not specified AND 4. ISM\_RESOURCE\_ELEMENT attribute atomicEnergyMarkings is not specified with a value of [RD] or [FRD] Human Readable: Documents under E.O. 13526 require declassDate or declassEvent unless 25X1-human, 50X1-HUM, 50X2-WMD, RD, or FRD is specified.

### Code Description:

If the current Classified National Security Information Executive Order does not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute declassException specified with a value of [25X1-human], [50X1-HUM], or [50X2-WMD], then the rule does not apply and we return true. If the resourceElement has attribute atomicEnergyMarkings specified with a value containing [RD] or [FRD] then the rule does not apply and we return true. Otherwise, we make sure that the resourceElement has attribute declassDate specified or attribute declassEvent specified.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00141">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:let name="paddedDeclassEx"
value="concat(' ', normalize-space(string(./@ism:declassException)), ' ')" />

<sch:assert id="ISM-00141"
test=" if(not($ISM_NSI_EO_APPLIES)) then true() else if(matches($paddedDeclassEx, ' (25X1-
human|50X1-HUM|50X2-WMD) ')) then true() else if( sum( for $each in tokenize(string(./
@ism:atomicEnergyMarkings), ' ') return if(matches($each, '^F?RD-?.*$')) then 1 else
0 )>0 ) then true() else (./@ism:declassDate or ./@ism:declassEvent) "
flag="error">
[ISM-ID-00141][Error] Documents under E.O. 13526 require declassDate or declassEvent
unless 25X1-human,
50X1-HUM, 50X2-WMD, RD, or FRD is specified.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00142

FileName:./Rules/classification/ISM\_ID\_00142.sch

### Rule Description:

[ISM-ID-00142][Error] If ISM\_NSI\_EO\_APPLIES and attribute classification has a value other than [U] then attribute classifiedBy or derivativelyClassifiedBy must be specified on the ISM\_RESOURCE\_ELEMENT. Human Readable: Classified data including DOE data requires either an original classifier or a derivative classifier be identified.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute classification with a value of [U] then we return true because the rule does not apply. Otherwise we make sure that the resourceElement has the attribute classifiedBy or derivativelyClassifiedBy.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00142">

<sch:rule context="*[@ism:classification and $ISM_NSI_EO_APPLIES]">
<sch:assert id="ISM-00142"
test=" if(./@ism:classification='U') then true() else ($ISM_RESOURCE_ELEMENT/
@ism:classifiedBy or $ISM_RESOURCE_ELEMENT/@ism:derivativelyClassifiedBy) "
flag="error">
[ISM-ID-00142][Error] Classified data including DOE data requires either an original
classifier or a derivative classifier be identified.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00143

FileName:./Rules/derivativelyClassifiedBy/ISM\_ID\_00143.sch

### Rule Description:

[ISM-ID-00143][Error] If ISM\_CAPCO\_RESOURCE and attribute derivativelyClassifiedBy is specified, then attribute derivedFrom must be specified. Human Readable: Derivatively Classified data including DOE data requires a derived from value to be identified.

### Code Description:

If CAPCO rules do not apply to the resource then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute derivativelyClassifiedBy then we also have the attribute derivedFrom on the same element.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00143">

<sch:rule context="*[@ism:derivativelyClassifiedBy and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00143" test="./@ism:derivedFrom" flag="error">
[ISM-ID-00143][Error] Derivatively Classified data including DOE data requires a derived
from value to be identified.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00145

FileName:./Rules/resourceElement/ISM\_ID\_00145.sch

### Rule Description:

[ISM-ID-00145][Error] If ISM\_CAPCO\_RESOURCE and any element in the document: 1. Meets ISM\_CONTRIBUTES AND 2. Has the attribute nonICmarkings containing [LES] AND 3. No element meeting ISM\_CONTRIBUTES in the document has nonICmarkings containing any of [LES-NF] Then the ISM\_RESOURCE\_ELEMENT must have nonICmarkings containing [LES]. Human Readable: USA documents having LES and not having LES-NF must have LES at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute nonICmarkings specified with a value containing [LES] and no element has attribute nonICmarkings specified with a value containing [LES-NF], then we make sure that the resourceElement has attribute nonICmarkings specified with a value containing [LES].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00145">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00145"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partNonICmarkings_tok,
'LES') > 0 and not(index-of($partNonICmarkings_tok, 'LES-NF') > 0)) then (index-
of($bannerNonICmarkings_tok, 'LES') > 0) else true() "
flag="error">
[ISM-ID-00145][Error] USA documents having LES and not having LES-NF must have LES at the
resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00146

FileName:./Rules/resourceElement/ISM\_ID\_00146.sch

### Rule Description:

[ISM-ID-00146][Error] If ISM\_CAPCO\_RESOURCE and there exist at least 2 elements in the document: 1. Each element: Meets ISM\_CONTRIBUTES AND 2. One of the elements: Has the attribute nonICmarkings containing [LES-NF] AND 3. One of the elements: meets ISM\_CONTRIBUTES\_CLASSIFIED Then the ISM\_RESOURCE\_ELEMENT must have disseminationControls containing [NF]. Human Readable: Classified USA documents having LES-NF Data must have NF at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute nonICmarkings specified with a value containing [LES-NF] and the resourceElement has attribute classification specified with a value other than [U], then we make sure that the resourceElement has attribute disseminationControls specified with a value containing [NF].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00146">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00146"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partNonICmarkings_tok,
'LES-NF') > 0 and not($bannerClassification='U')) then (index-
of($bannerDisseminationControls_tok, 'NF') > 0) else true() "
flag="error">
[ISM-ID-00146][Error] Classified USA documents having LES-NF Data must have NF at the
resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00147

FileName:./Rules/resourceElement/ISM\_ID\_00147.sch

### Rule Description:

[ISM-ID-00147][Error] If ISM\_CAPCO\_RESOURCE and there exist at least 2 elements in the document: 1. Each element: Meets ISM\_CONTRIBUTES AND 2. One of the elements: Has the attribute nonICmarkings containing [LES-NF] AND 3. One of the elements: meets ISM\_CONTRIBUTES\_CLASSIFIED Then the ISM\_RESOURCE\_ELEMENT must have nonICmarkings containing [LES]. Human Readable: Classified USA documents having LES-NF Data must have LES at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute nonICmarkings specified with a value containing [LES-NF] and the resourceElement has attribute classification specified with a value other than [U], then we make sure that the resourceElement has attribute nonICmarkings specified with a value containing [LES].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00147">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00147"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partNonICmarkings_tok,
'LES-NF') > 0 and not($bannerClassification='U')) then (index-
of($bannerNonICmarkings_tok, 'LES') > 0) else true() "
flag="error">
[ISM-ID-00147][Error] Classified USA documents having LES-NF Data must have LES at the
resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00148

FileName:./Rules/nonICmarkings/ISM\_ID\_00148.sch

### Rule Description:

[ISM-ID-00148][Error] If ISM\_CAPCO\_RESOURCE, then Name tokens [LES] and [LES-NF] are mutually exclusive for attribute nonICmarkings. Human Readable: USA documents must not specify both LES and LES-NF on a single element.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that the attribute nonICmarkings does not contain both a value of [LES] and a value of [LES-NF].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00148">

<sch:rule context="*[@ism:nonICmarkings]">
<!-- get list of tokens in nonICmarkings attribute -->
<sch:let name="nicmTok" value="tokenize(string(./@ism:nonICmarkings), ' ')" />

<sch:assert id="ISM-00148"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else not(index-of($nicmTok,'LES')>0 and
index-of($nicmTok,'LES-NF')>0) "
flag="error">
[ISM-ID-00148][Error] If ISM_CAPCO_RESOURCE, then tokens [LES] and [LES-NF] are mutually
exclusive for attribute nonICmarkings.

Human Readable: USA documents must not specify both LES and LES-NF on a single element.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00149

FileName:./Rules/resourceElement/ISM\_ID\_00149.sch

### Rule Description:

[ISM-ID-00149][Error] If ISM\_CAPCO\_RESOURCE and 1. Any element in the document meets ISM\_CONTRIBUTES in the document has the attribute nonICmarkings contain [LES-NF] AND 2. ISM\_RESOURCE\_ELEMENT has the attribute classification [U] THEN the ISM\_RESOURCE\_ELEMENT must have nonICmarkings containing [LES-NF] Human Readable: Unclassified USA documents having LES-NF must have LES-NF at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute nonICmarkings specified with a value containing [LES-NF] and the resourceElement has attribute classification specified with a value of [U], then we make sure that the resourceElement has attribute nonICmarkings specified with a value containing [LES-NF].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00149">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00149"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($partNonICmarkings_tok,
'LES-NF') > 0 and $bannerClassification='U') then (index-of($bannerNonICmarkings_tok,
'LES-NF') > 0) else true() "
flag="error">
[ISM-ID-00149][Error] Unclassified USA documents having LES-NF data must have LES-NF at
the resource level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00150

FileName:./Rules/notice/ISM\_ID\_00150.sch

### Rule Description:

[ISM-ID-00150][Error] If ISM\_CAPCO\_RESOURCE and: 1. Any element meeting ISM\_CONTRIBUTES in the document has the attribute nonICmarkings containing [LES] AND 2. No element meeting ISM\_CONTRIBUTES in the document has notice containing [LES] Human Readable: USA documents containing LES data must also have an LES notice.

### Code Description:

If CAPCO rules do not apply to the document or the element is excluded from the rollup then the rule does not apply and we return true. If the current element is the resourceElement then the rule does not apply and we return true. If the current element has attribute nonICmarkings specified with a value containing [LES], then we make sure that attribute notice is specified with a value containing [LES] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00150">

<sch:rule context="*[@ism:nonICmarkings]">
<sch:assert id="ISM-00150"
test=" if(not($ISM_CAPCO_RESOURCE) or ./@ism:excludeFromRollup=true()) then true()
else if(generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)) then true() else
if(index-of(tokenize(string(./@ism:nonICmarkings), ' '), 'LES')>0) then index-
of($partNotice_tok, 'LES')>0 else true() "
flag="error">
[ISM-ID-00150][Error] If ISM_CAPCO_RESOURCE and:
1. Any element meeting ISM_CONTRIBUTES in the document has the attribute nonICmarkings
containing [LES]
AND
2. No element meeting ISM_CONTRIBUTES in the document has notice containing [LES]

Human Readable: USA documents containing LES data must also have an LES notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00151

FileName:./Rules/notice/ISM\_ID\_00151.sch

### Rule Description:

[ISM-ID-00151][Warning] If ISM\_CAPCO\_RESOURCE and: 1. No element without ism:excludeFromRollup=true() in the document has the attribute nonICmarkings containing [LES] AND 2. Any element without ism:excludeFromRollup=true() in the document has the attribute notice containing [LES] Human Readable: USA documents containing an LES notice must also have LES data.

### Code Description:

If CAPCO rules do not apply to the document and the element is not excluded from the rollup then the rule does not apply and we return true. If the current element has attribute notice specified with a value containing [LES] and it is included in the rollup, then we make sure that attribute nonICmarkings is specified with a value containing [LES] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00151">

<sch:rule context="*[@ism:noticeType]">
<sch:assert id="ISM-00151"
test=" if(not($ISM_CAPCO_RESOURCE) or ./@ism:excludeFromRollup=true()) then true()
else if(index-of(tokenize(string(./@ism:noticeType), ' '), 'LES')>0 and not(./
@ism:excludeFromRollup=true())) then index-of($partNonICmarkings_tok, 'LES')>0 else
true() "
flag="warning">
[ISM-ID-00151][Warning] If ISM_CAPCO_RESOURCE and:
1. No element without ism:excludeFromRollup=true() in the document has the attribute
nonICmarkings containing [LES]
AND
2. Any element without ism:excludeFromRollup=true() in the document has the attribute
notice containing [LES]

Human Readable: USA documents containing LES data must also have an LES notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00152

FileName:./Rules/notice/ISM\_ID\_00152.sch

### Rule Description:

[ISM-ID-00152][Error] If ISM\_CAPCO\_RESOURCE and: 1. Any element meeting ISM\_CONTRIBUTES in the document has the attribute nonICmarkings containing [LES-NF] AND 2. No element meeting ISM\_CONTRIBUTES in the document has notice containing [LES-NF] Human Readable: USA documents containing LES-NF data must also have an LES-NF notice.

### Code Description:

If CAPCO rules do not apply to the document and the element is not excluded from the rollup then the rule does not apply and we return true. If the current element specifies attribute nonICmarkings with a value containing [LES-NF], then we make sure that attribute notice is specified with a value containing [LES-NF] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00152">

<sch:rule context="*[@ism:nonICmarkings]">
<sch:assert id="ISM-00152"
test=" if(not($ISM_CAPCO_RESOURCE) or ./@ism:excludeFromRollup=true()) then true()
else if(generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)) then true() else
if(index-of(tokenize(string(./@ism:nonICmarkings), ' '), 'LES-NF')>0) then index-
of($partNotice_tok, 'LES-NF')>0 else true() "
flag="error">
[ISM-ID-00152][Error] If ISM_CAPCO_RESOURCE and:
1. Any element meeting ISM_CONTRIBUTES in the document has the attribute nonICmarkings
containing [LES-NF]
AND
2. No element meeting ISM_CONTRIBUTES in the document has notice containing [LES-NF]

Human Readable: USA documents containing LES-NF data must also have an LES-NF notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00153

FileName:./Rules/notice/ISM\_ID\_00153.sch

### Rule Description:

[ISM-ID-00153][Error] If ISM\_CAPCO\_RESOURCE and: 1. No element without ism:excludeFromRollup=true() in the document has the attribute nonICmarkings containing [LES-NF] AND 2. Any element without ism:excludeFromRollup=true() in the document has the attribute notice containing [LES-NF]. Human Readable: USA documents containing an LES-NF notice must also have LES-NF data.

### Code Description:

If CAPCO rules do not apply to the document and the element is not excluded from the rollup then the rule does not apply and we return true. If the current element has attribute notice specified with a value containing [LES-NF] and it is included in the rollup, then we make sure that attribute nonICmarkings is specified with a value containing [LES-NF] in one of the portions of the document.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00153">

<sch:rule context="*[@ism:noticeType]">
<sch:assert id="ISM-00153"
test=" if(not($ISM_CAPCO_RESOURCE) or ./@ism:excludeFromRollup=true()) then true()
else if(index-of(tokenize(string(./@ism:noticeType), ' '), 'LES-NF')>0 and not(./
@ism:excludeFromRollup=true())) then index-of($partNonICmarkings_tok, 'LES-NF')>0 else
true() "
flag="error">
[ISM-ID-00153][Error] If ISM_CAPCO_RESOURCE and:
1. No element without ism:excludeFromRollup=true() in the document has the attribute
nonICmarkings containing [LES-NF]
AND
2. Any element without ism:excludeFromRollup=true() in the document has the attribute
notice containing [LES-NF]

Human Readable: USA documents containing LES-NF data must also have an LES-NF notice.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00154

FileName:./Rules/resourceElement/ISM\_ID\_00154.sch

### Rule Description:

[ISM-ID-00154][Error] If ISM\_CAPCO\_RESOURCE and 1. Attribute disseminationControls of ISM\_RESOURCE\_ELEMENT contains [FOUO] AND 2. Attribute compilationReason does not have a value then at least one element meeting ISM\_CONTRIBUTES in the document must have a disseminationControls attribute contain [FOUO].  
Human Readable: USA FOUO documents not using compilation must have FOUO data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the resourceElement has attribute disseminationControls specified with a value containing [FOUO] and the resourceElement has attribute compilationReason specified with an empty value, then we make sure that at least one element in the document has attribute disseminationControls specified with a value containing [FOUO].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00154">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00154"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if (index-
of($bannerDisseminationControls_tok, 'FOUO')>0 and not(string-
length(normalize-space(string(./@ism:compilationReason)))>0)) then index-
of($partDisseminationControls_tok,'FOUO')>0 else true() "
flag="error">
[ISM-ID-00154][Error] USA FOUO documents not using compilation must have FOUO data.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00155

FileName:./Rules/resourceElement/ISM\_ID\_00155.sch

### Rule Description:

[ISM-ID-00155][Error] If ISM\_CAPCO\_RESOURCE and 1. ISM\_DoD5230\_24\_Applies AND 2. Attribute notice of ISM\_RESOURCE\_ELEMENT does not contain one of [DoD-Dist-A], [DoD-Dist-B], [DoD-Dist-C], [DoD-Dist-D], [DoD-Dist-E], [DoD-Dist-F], or [DoD-Dist-X] Human Readable: All USA documents that claim compliance with DoD5230.24 must have a distribution statement for the entire document.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If ISM\_DoD5230\_24\_Applies does not apply to the document then the rule does not apply and we return true. Otherwise, we make sure that the resourceElement has attribute notice specified with a value containing [DoD-Dist-A], [DoD-Dist-B], [DoD-Dist-C], [DoD-Dist-D], [DoD-Dist-E], [DoD-Dist-F], or [DoD-Dist-X].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00155">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00155"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(not($ISM_DOD5230_24_APPLIES))
then true() else count( (if(index-of($bannerNotice_tok, 'DoD-Dist-A')>0) then 1 else
null, if(index-of($bannerNotice_tok, 'DoD-Dist-B')>0) then 1 else null, if(index-
of($bannerNotice_tok, 'DoD-Dist-C')>0) then 1 else null, if(index-of($bannerNotice_tok,
'DoD-Dist-D')>0) then 1 else null, if(index-of($bannerNotice_tok, 'DoD-Dist-E')>0)
then 1 else null, if(index-of($bannerNotice_tok, 'DoD-Dist-F')>0) then 1 else null,
if(index-of($bannerNotice_tok, 'DoD-Dist-X')>0) then 1 else null) )>0 "
flag="error">
[ISM-ID-00155][Error] All USA documents that claim compliance with DoD5230.24 must have a
distribution statement
for the entire document.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00156

FileName:./Rules/notice/ISM\_ID\_00156.sch

### Rule Description:

[ISM-ID-00156][Error] If ISM-CAPCO-RESOURCE and: 1. The attribute notice contains on of the [DoD-Dist-B], [DoD-Dist-C], [DoD-Dist-D], [DoD-Dist-E], [DoD-Dist-F], or [DoD-Dist-X] AND 2. Attribute noticeDate is not specified AND 3. Attribute pocType is not specified on some element in the document with the same value as that of notice Human Readable: DoD distribution statements B, C, D ,E ,F, and X all require a Date and a POC.

### Code Description:

For every element that has a @noticeType attribute in a CAPCO document, if the value of @noticeType is specified with a value containing [DoD-Dist-B], [DoD-Dist-C], [DoD-Dist-D], [DoD-Dist-E], [DoD-Dist-F], or [DoD-Dist-X], then we make sure that attribute @noticeDate is specified on the current element, and somewhere in the document, the @pocType attribute is specified with the given value of @noticeType.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00156">

<sch:rule context="*[@ism:noticeType and $ISM_CAPCO_RESOURCE]">
<!-- tokenize the element's notice attribute -->
<sch:let name="noticeTok" value="normalize-space(string(./@ism:noticeType))"/>

<sch:assert id="ISM-00156"
test=" if(index-of($noticeTok, 'DoD-Dist-B')>0 or index-of($noticeTok, 'DoD-Dist-
C')>0 or index-of($noticeTok, 'DoD-Dist-D')>0 or index-of($noticeTok, 'DoD-Dist-
E')>0 or index-of($noticeTok, 'DoD-Dist-F')>0 or index-of($noticeTok, 'DoD-Dist-
X')>0) then (./@ism:noticeDate and index-of($partPocType_tok,$noticeTok)>0) else
true() "
flag="error">
[ISM-ID-00156][Error] DoD distribution statements B, C, D ,E ,F, and X all require a Date
and a POC.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00157

FileName:./Rules/notice/ISM\_ID\_00157.sch

### Rule Description:

[ISM-ID-00157][Error] If ISM\_CAPCO\_RESOURCE and: 1. The attribute notice contains one of the [DoD-Dist-B], [DoD-Dist-C], [DoD-Dist-D], or [DoD-Dist-E] AND 2. The attribute noticeReason is not specified. Human Readable: DoD distribution statements B, C, D , or E all require a reason.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute notice specified with a value containing [DoD-Dist-B], [DoD-Dist-C], [DoD-Dist-D], [DoD-Dist-E], or [DoD-Dist-F], then we make sure that attribute noticeReason is also specified on the resourceElement.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00157">

<sch:rule context="*[@ism:noticeType]">
<!-- tokenize the notice attribute -->
<sch:let name="noticeTok" value="tokenize(string(./@ism:noticeType), ' ')" />

<sch:assert id="ISM-00157"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if( count( (if(index-of($noticeTok,
'DoD-Dist-B')>0) then 1 else null, if(index-of($noticeTok, 'DoD-Dist-C')>0)
then 1 else null, if(index-of($noticeTok, 'DoD-Dist-D')>0) then 1 else null,
if(index-of($noticeTok, 'DoD-Dist-E')>0) then 1 else null) )=0 ) then true() else ./
@ism:noticeReason "
flag="error">
[ISM-ID-00157][Error] DoD distribution statements B, C, D , or E all require a reason.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00158

FileName:./Rules/notice/ISM\_ID\_00158.sch

### Rule Description:

[ISM-ID-00158][Error] If ISM\_CAPCO\_RESOURCE and: 1. ISM\_DoD5230\_24\_Applies AND 2. attribute classification of ISM\_RESOURCE\_ELEMENT is not [U] AND 3. The attribute notice does not contain one of [DoD-Dist-B], [DoD-Dist-C], [DoD-Dist-D], [DoD-Dist-E], or [DoD-Dist-F]. Human Readable: All classified documents that claim compliance with DoD5230.24 must use one of DoD distribution statements B, C, D, E, or F.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If DoD-5230-24 does not apply then the rule does not apply and we return true. If the resource is Unclassified then the rule does not apply and we return true. Otherwise, we make sure that the resourceElement attribute notice does not contain a value of [DoD-Dist-B], [DoD-Dist-C], [DoD-Dist-D], [DoD-Dist-E], or [DoD-Dist-F].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00158">

<sch:rule context="*[@ism:resourceElement=true()]">
<sch:assert id="ISM-00158"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(not($ISM_DOD5230_24_APPLIES))
then true() else if(./@ism:classification='U') then true() else if( count( (if(index-
of($bannerNotice_tok, 'DoD-Dist-B')>0) then 1 else null, if(index-of($bannerNotice_tok,
'DoD-Dist-C')>0) then 1 else null, if(index-of($bannerNotice_tok, 'DoD-Dist-D')>0)
then 1 else null, if(index-of($bannerNotice_tok, 'DoD-Dist-E')>0) then 1 else null,
if(index-of($bannerNotice_tok, 'DoD-Dist-F')>0) then 1 else null) )=0 ) then false()
else true() "
flag="error">
[ISM-ID-00158][Error] All classified documents that claim compliance with DoD5230.24 must
use one of DoD
distribution statements B, C, D, E, or F.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00159

FileName:./Rules/notice/ISM\_ID\_00159.sch

### Rule Description:

[ISM-ID-00159][Error] If ISM\_CAPCO\_RESOURCE and: 1. attribute classification of ISM\_RESOURCE\_ELEMENT is not [U] AND 2. The attribute notice does contains [DoD-Dist-A]. Human Readable: Distribution statement A (Public Release) is forbidden on classified documents.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the document is Unclassified then the rule does not apply and we return true. Otherwise, we check that the current element does not have attribute notice specified with a value containing [DoD-Dist-A].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00159">

<sch:rule context="*[@ism:noticeType]">
<!-- tokenize the notice attribute -->
<sch:let name="noticeTok" value="tokenize(string(./@ism:noticeType), ' ')" />

<sch:assert id="ISM-00159"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if($bannerClassification='U') then
true() else not(index-of($noticeTok, 'DoD-Dist-A')>0) "
flag="error">
[ISM-ID-00159][Error] Distribution statement A (Public Release) is forbidden on classified
documents.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00160

FileName:./Rules/notice/ISM\_ID\_00160.sch

### Rule Description:

[ISM-ID-00160][Error] If ISM\_CAPCO\_RESOURCE and: 1. The attribute notice of ISM\_RESOURCE\_ELEMENT does contain [DoD-Dist-A] AND 2. attribute disseminationControls contains any of [FOUO], [PR], [DSEN], OR [FISA] AND 3. attribute atomicEnergyMarkings contains any of [DCNI] or [UCNI]. Human Readable: Distribution statement A (Public Release) is incompatible with [FOUO], [PR], [DCNI], [UCNI], [DSEN], OR [FISA].

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if the resourceElement has attribute notice containing a value of [DoD-Dist-A] that the resourceElement's attribute disseminationControls does not contain values [FOUO], [PR], [DSEN], or [FISA] and attribute atomicEnergyMarkings does not contain values [UCNI] or [DCNI].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00160">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00160"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($bannerNotice_tok, 'DoD-
Dist-A')>0) then count( (if(index-of($bannerDisseminationControls_tok, 'FOUO')>0)
then 1 else null, if(index-of($bannerDisseminationControls_tok, 'PR')>0) then 1
else null, if(index-of($bannerAtomicEnergyMarkings_tok, 'DCNI')>0) then 1 else
null, if(index-of($bannerAtomicEnergyMarkings_tok, 'UCNI')>0) then 1 else null,
if(index-of($bannerDisseminationControls_tok, 'DSEN')>0) then 1 else null, if(index-
of($bannerDisseminationControls_tok, 'FISA')>0) then 1 else null) )=0 else true() "
flag="error">
[ISM-ID-00160][Error] Distribution statement A (Public Release) is
incompatible with [FOUO], [PR], [DCNI], [UCNI], [DSEN], OR [FISA].
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00161

FileName:./Rules/notice/ISM\_ID\_00161.sch

### Rule Description:

[ISM-ID-00161][Error] If ISM\_CAPCO\_RESOURCE and: 1. The attribute notice of ISM\_RESOURCE\_ELEMENT does contains [DoD-Dist-A] AND 2. attribute nonICmarkings contains any of [SINFO], [XD], [ND], [SBU], [SBU-NF], [LES], OR [LES-NF]. Human Readable: Distribution statement A (Public Release) is incompatible with [SINFO], [XD], [ND], [SBU], [SBU-NF], [LES], OR [LES-NF].

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource and the resourceElement has attribute notice specified with a value containing [DoD-Dist-A], then we make sure that the resourceElement's attribute nonICmarkings does not contain values [SINFO], [XD], [ND], [SBU], [SBU-NF], [LES], or [LES-NF].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00161">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00161"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($bannerNotice_tok,
'DoD-Dist-A')>0) then count( (if(index-of($bannerNonICmarkings_tok,
'SINFO')>0) then 1 else null, if(index-of($bannerNonICmarkings_tok, 'XD')>0)
then 1 else null, if(index-of($bannerNonICmarkings_tok, 'ND')>0) then 1
else null, if(index-of($bannerNonICmarkings_tok, 'SBU')>0) then 1 else
null, if(index-of($bannerNonICmarkings_tok, 'SBU-NF')>0) then 1 else null,
if(index-of($bannerNonICmarkings_tok, 'LES')>0) then 1 else null, if(index-
of($bannerNonICmarkings_tok, 'LES-NF')>0) then 1 else null) )=0 else true() "
flag="error">
[ISM-ID-00161][Error] Distribution statement A (Public Release) is incompatible with
[SINFO], [XD], [ND], [SBU], [SBU-NF], [LES], OR [LES-NF].
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00162

FileName:./Rules/resourceElement/ISM\_ID\_00162.sch

### Rule Description:

[ISM-ID-00162][Error] If ISM\_CAPCO\_RESOURCE and 1. ISM\_DoD5230\_24\_Applies AND 2. Attribute notice of ISM\_RESOURCE\_ELEMENT contains more than one of [DoD-Dist-A], [DoD-Dist-B], [DoD-Dist-C], [DoD-Dist-D], [DoD-Dist-E], [DoD-Dist-F], or [DoD-Dist-X] Human Readable: All USA documents that claim compliance with DoD5230.24 must have only 1 distribution statement for the entire document.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If ISM\_DoD5230\_24\_Applies does not apply to the document then the rule does not apply and we return true. Otherwise, we make sure that the resourceElement has attribute notice specified with a value containing only one of [DoD-Dist-A], [DoD-Dist-B], [DoD-Dist-C], [DoD-Dist-D], [DoD-Dist-E], [DoD-Dist-F], or [DoD-Dist-X].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00162">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00155"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(not($ISM_DOD5230_24_APPLIES))
then true() else not(count( ( if(index-of($bannerNotice_tok, 'DoD-Dist-A')>0) then 1
else null, if(index-of($bannerNotice_tok, 'DoD-Dist-B')>0) then 1 else null, if(index-
of($bannerNotice_tok, 'DoD-Dist-C')>0) then 1 else null, if(index-of($bannerNotice_tok,
'DoD-Dist-D')>0) then 1 else null, if(index-of($bannerNotice_tok, 'DoD-Dist-E')>0)
then 1 else null, if(index-of($bannerNotice_tok, 'DoD-Dist-F')>0) then 1 else null,
if(index-of($bannerNotice_tok, 'DoD-Dist-X')>0) then 1 else null) )>1) "
flag="error">
[ISM-ID-00162][Error] All USA documents that claim compliance with DoD5230.24 must have
only 1 distribution statement
for the entire document.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00163

FileName:../Rules/nonUSControls/ISM\_ID\_00163.sch

### Rule Description:

[ISM-ID-00163][Error] If attribute nonUSControls exists the attribute ownerProducer must equal [NATO]. Human Readable: NATO is the only owner of classification markings for which nonUSControls are currently authorized.

### Code Description:

The code ensures that any element containing the attribute nonUSControls also has attribute ownerProducer specified with a value of [NATO].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00163">

<sch:rule context="*[@ism:nonUSControls]">
<sch:assert id="ISM-00163" test="./@ism:ownerProducer='NATO'" flag="error">
[ISM-ID-00163][Error] NATO is the only owner of classification markings for which
nonUSControls are currently authorized.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00164

FileName:./Rules/disseminationControls/ISM\_ID\_00164.sch

### Rule Description:

[ISM-ID-00164][Error] If ISM\_CAPCO\_RESOURCE and attribute disseminationControls contains the name token [RS], then attribute classification must have a value of [TS] or [S]. Human Readable: USA documents with RISK SENSITIVE dissemination must be classified SECRET or TOP SECRET.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check the attribute classification of the element and return true if it has a value of [S] or [TS]. Otherwise we check that the attribute disseminationControls does not contain the value [RS].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00164">

<sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
<sch:let name="dissemTok" value="tokenize(string(./@ism:disseminationControls), ' ')" />
<sch:assert id="ISM-00164"
test=" if(matches(./@ism:classification, '^(TS|S)$')) then true() else not(index-
of($dissemTok, 'RS')>0) "
flag="error">
[ISM-ID-00164][Error] If ISM_CAPCO_RESOURCE and attribute
disseminationControls contains the name token [RS],
then attribute classification must have a value of [TS] or [S].

Human Readable: USA documents with RISK SENSITIVE dissemination must
be classified SECRET or TOP SECRET.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00165

FileName:./Rules/resourceElement/ISM\_ID\_00165.sch

### Rule Description:

[ISM-ID-00165][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document have the attribute disseminationControls containing [RS] then the ISM\_RESOURCE\_ELEMENT must have disseminationControls containing [RS]. Human Readable: USA documents having RISK SENSITIVE (RS) data must have RS at the resource level.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If any element has attribute disseminationControls specified with a value containing [RS], then we make sure that the resourceElement has attribute disseminationControls specified with a value containing [RS].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00165">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT)]">
<sch:assert id="ISM-00165"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-
of($partDisseminationControls_tok, 'RS')>0) then index-
of($bannerDisseminationControls_tok, 'RS') > 0 else true() "
flag="error">
[ISM-ID-00165][Error] USA documents having RS Data must have RS at the resource level.
Human Readable: USA documents having RISK SENSITIVE (RS) data must have RS at the resource
level.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00166

FileName:../Rules/generalConstraints/ISM\_ID\_00166.sch

### Rule Description:

[ISM-ID-00166][Warning] Attribute classification contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate, determine if the values are being used but it is still prior to the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00166">

  <sch:rule context="*[@ism:classification]">

    <sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMClassificationAll.xml')//cve:CVE/cve:Enumeration/
cve:Term[./@deprecated]" />
    <sch:let name="isError" value="false()" />

    <sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:classification), $depTerms,
$ISM_RESOURCE_CREATE_DATE, $isError) " />

    <sch:assert id="ISM-00166" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00166][Warning] For attribute classification, value(s) <sch:value-of
select="$reportWarn" />
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00167

FileName:./Rules/displayOnlyTo/ISM\_ID\_00167.sch

### Rule Description:

[ISM-ID-00167][Error] If ISM\_CAPCO\_RESOURCE and attribute displayOnlyTo is specified, then each of its values must be ordered in accordance with CVEnumISMRelTo.xml.

### Code Description:

To perform sorting, this rule first retrieves the CVE values for the attribute to be sorted, which in this case is displayOnlyTo. Then, each attribute token is converted into a numerical value based on its characters. Next, each attribute token is given an order number, which compares its position to that of its value in the CVE file. Next, each order number is compared to that of its previous sibling to determine if the tokens are in order. If a token is found whose order number is less than that of its previous sibling, 0 is returned for its sorted order number. If a token's order number is greater than that of its previous sibling, 1 is returned. If two tokens have the same order number, their original attribute values are compared. If the original attribute value contains numbers then the comparison is made on its converted numerical value; otherwise, the comparison is made on its string value. If an attribute value is found whose value is less than that of its previous sibling, 0 is returned for its sorted order number; otherwise 2 is returned. Finally, if any tokens are found with 0 as its sorted order number, then the rule fails as those tokens are out of order.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00167">

  <sch:rule context="*[@ism:displayOnlyTo]">
    <!-- Define variables -->
    <sch:let name="errMsg_AlphabeticalOrder"
value="' [ISM-ID-00167][Error] If ISM_CAPCO_RESOURCE and attribute displayOnlyTo is
specified, then each of its values must be ordered in accordance with CVEnumISMRelTo.xml.
'"/>

    <sch:let name="dataFileElems" value="$displayOnlyToList"/>
    <sch:let name="attrValues" value="./@ism:displayOnlyTo"/>
    <sch:let name="attrValueTokens" value="tokenize(string($attrValues),' ')/>

    <!-- Convert each character to a numerical value, then concatenate the results to form a
number-string -->
    <sch:let name="convertStrToNum"
value=" for $token in $attrValueTokens return number(string-join( for $index in 1
to string-length($token) return for $char in substring($token, $index, 1) return if
(contains(string('0123456789'), $char)) then $char else if (contains(string('ABCDEFGHI'),
$char)) then translate(string($char), 'ABCDEFGHI', '123456789') else if
(contains(string('JKLMNOPQRS'), $char)) then concat('1', translate(string($char),
'JKLMNOPQRS', '0123456789')) else if (contains(string('TUVWXYZ'), $char)) then concat('2',
translate(string($char), 'TUVWXYZ', '0123456')) else '0' , '')) "/>
```

```

<!-- Get the position of each client node relative to its position in the master list. If
the node is not found, return a -1 -->
<sch:let name="orderNums"
value=" for $token in $attrValueTokens return if
($dataFileElems[matches($token,concat('^',text(),'$'))]) then
count(($dataFileElems[matches($token,concat('^',text(),'$'))]/preceding::*) + 1 else -1"/
>

<!-- Create a sequence that returns a 0 if the previous sibling has a higher order number,
else return a 1 -->
<sch:let name="sortedOrderNums"
value=" for $index in distinct-values(for $token in $orderNums return index-of($orderNums,
$token)) return if($index = 1 or $orderNums[$index] > $orderNums[$index - 1])
then 1 else if ($orderNums[$index] < $orderNums[$index - 1]) then 0 else if
(matches($attrValueTokens[$index], '[0-9]') or matches($attrValueTokens[$index - 1],
'[0-9]')) then if ($convertStrToNum[$index - 1] > $convertStrToNum[$index]) then 0 else
2 else if (compare($attrValueTokens[$index - 1],$attrValueTokens[$index])=1) then 0 else 2
"/>
<sch:let name="hasUnsorted" value="count(index-of($sortedOrderNums,0)) > 0"/>
<sch:let name="unsortedValues"
value=" if ($hasUnsorted) then distinct-values( for $token in index-of($sortedOrderNums,0)
return $attrValueTokens[$token] ) else null "/>

<sch:assert id="ISM-00167" test="not($hasUnsorted)" flag="error">
<sch:value-of select="$errMsg_AlphabeticalOrder"/>
The following values are out of order [<sch:value-of select="$unsortedValues"/>] for
[<sch:value-of select="$attrValueTokens"/>] </sch:assert>
</sch:rule>
</sch:pattern>

```

## Rule: ISM-ID-00168

FileName:./Rules/displayOnlyTo/ISM\_ID\_00168.sch

### Rule Description:

[ISM-ID-00168][Error] If ISM\_CAPCO\_RESOURCE and attribute disseminationControls is not specified or is specified and does not contain the name token [DISPLAYONLY], then attribute displayOnlyTo must not be specified. Human Readable: If a portion in a USA document is not marked for DISPLAY ONLY dissemination, it must not list countries to which it can be disseminated.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if the attribute displayOnlyTo is specified then the attribute disseminationControls is also specified that it contains a value of [DISPLAYONLY]

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00168">

<sch:rule context="*[@ism:displayOnlyTo and $ISM_CAPCO_RESOURCE]">

<sch:let name="dissemTok" value="tokenize(string(./@ism:disseminationControls),' ')" />
<sch:assert id="ISM-00168" test="index-of($dissemTok,'DISPLAYONLY')>0" flag="error">
[ISM-ID-00168][Error] If ISM_CAPCO_RESOURCE and attribute
disseminationControls is not specified or is specified and does not contain the name token
[DISPLAYONLY], then attribute displayOnlyTo must not be specified.

Human Readable: If a portion in a USA document is not marked for DISPLAY ONLY
dissemination,
it must not list countries to which it can be disseminated.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00169

FileName:../Rules/disseminationControls/ISM\_ID\_00169.sch

### Rule Description:

[ISM-ID-00169][Error] If ISM\_CAPCO\_RESOURCE, and attribute disseminationControls contains name token [DISPLAYONLY] then tokens [RELIDO] and [NF] may not also be used. Human Readable: In a USA document, DISPLAY ONLY, RELIDO and NO FOREIGN dissemination are mutually exclusive for a single element.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute disseminationControls with a value of [DISPLAYONLY] then it does not have a value of [RELIDO] or [NF] also.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00169">

<sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
<sch:let name="dissemTok" value="tokenize(string(./@ism:disseminationControls),' ')" />
<sch:assert id="ISM-00169"
test=" if(index-of($dissemTok,'DISPLAYONLY')>0) then not(index-
of($dissemTok,'RELIDO')>0 or index-of($dissemTok,'NF')>0) else true() "
flag="error">
[ISM-ID-00169][Error] If ISM_CAPCO_RESOURCE, and attribute disseminationControls
contains name token [DISPLAYONLY] then tokens [RELIDO] and [NF] may not also be used.

Human Readable: In a USA document, DISPLAY ONLY, RELIDO and NO FOREIGN dissemination are
mutually exclusive for a single element.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00170

FileName:../Rules/generalConstraints/ISM\_ID\_00170.sch

### Rule Description:

[ISM-ID-00170][Error] Attribute classification must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date..

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00170">

<sch:rule context="*[@ism:classification]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMClassificationAll.xml')//cve:CVE/cve:Enumeration/
cve:Term[./@deprecated]" />
<sch:let name="isError" value="true()" />

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:classification), $depTerms,
$ISM_RESOURCE_CREATE_DATE, $isError) " />

<sch:assert id="ISM-00170" test="count($reportErr)=0" flag="error">
[ISM-ID-00170][Error] For attribute classification, value(s) <sch:value-of
select="$reportErr" />
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00171

FileName:./Rules/resourceElement/ISM\_ID\_00171.sch

### Rule Description:

[ISM-ID-00171][Warning] If ISM\_CAPCO\_RESOURCE and displayOnlyTo is specified on the resource element then all classified portions must specify displayOnlyTo. Human Readable: USA documents having DISPLAYONLY data at the resource level must have all classified portions authorized for DISPLAYONLY.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. Otherwise, we loop over all portions of the document and count the number of elements which have attribute classification specified with a value other than [U] and do not have attribute displayOnlyTo. The loop checks, if the current element has attribute classification specified with a value other than [U], or has attribute displayOnlyTo, then the rule does not apply to this element and we return 0. Otherwise, we return 1 to indicate the element is classified but does not specify attribute displayOnlyTo. If the count of elements not meeting either of the two requirements stated above is greater than zero, then the assertion fails since attribute displayOnly appears on the banner but is not present on all classified portions.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00171">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT) and
@ism:displayOnlyTo]">
<sch:assert id="ISM-00171"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else sum( for $token in $partTags return
if(matches($token/@ism:classification, '^U$') or exists($token/@ism:displayOnlyTo)) then 0
else 1 ) = 0 "
flag="warning">
[ISM-ID-00171][Warning] If ISM_CAPCO_RESOURCE and displayOnlyTo is specified on
the resource element then all classified portions must specify displayOnlyTo.

Human Readable: USA documents having DISPLAYONLY data at the resource level
must have all classified portions authorized for DISPLAYONLY.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00173

FileName:./Rules/atomicEnergyMarkings/ISM\_ID\_00173.sch

### Rule Description:

[ISM-ID-00173][Error] If ISM\_CAPCO\_RESOURCE and attribute atomicEnergyMarkings contains a name token starting with [RD-SG] or [FRD-SG], then attribute classification must have a value of [TS], [S], or [C]. Human Readable: Portions in a USA document that contain RD or FRD SIGMA data should be CONFIDENTIAL, SECRET, or TOP SECRET.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. Then we check that the attribute atomicEnergyMarkings has a value of [RD-SG] or [FRD-SG] with a single digit [1-9] or double digit [10-99]. If this element does contain one of those values and its classification is not equal to TS, S or C, then the token is returned. If any token is returned, the count will be greater than 0 and the rule will fail.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00173">

<sch:rule context="*[@ism:atomicEnergyMarkings]">
<sch:assert id="ISM-00173"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else count( for $token in
tokenize(string(./@ism:atomicEnergyMarkings), ' ') return if(matches($token,'^F?RD-SG')
and not( matches(./@ism:classification,'^(TS|S|C)$')) then $token else null )=0 "
flag="error">
[ISM-ID-00173][Error] If ISM_CAPCO_RESOURCE and attribute
atomicEnergyMarkings contains a name token starting with [RD-SG] or [FRD-SG], then
attribute
classification must have a value of [TS], [S], or [C].

Human Readable: Portions in a USA document that contain RD or FRD SIGMA
data should be CONFIDENTIAL, SECRET, or TOP SECRET.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00174

FileName:../Rules/atomicEnergyMarkings/ISM\_ID\_00174.sch

### Rule Description:

[ISM-ID-00174][Error] If ISM\_CAPCO\_RESOURCE and attribute atomicEnergyMarkings contains the name token [RD] or [FRD], then attribute classification must have a value of [TS], [S], or [C]. Human Readable: USA documents with RD or FRD data must be marked CLASSIFIED, SECRET, or TOP SECRET.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute atomicEnergyMarkings with a value of [RD] or [FRD] then we also have the attribute classification specified with a value of [C], [S], or [TS] on the same element.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00174">

<sch:rule context="*[@ism:atomicEnergyMarkings]">
<sch:assert id="ISM-00174"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else count( for $token in
tokenize(string(./@ism:atomicEnergyMarkings), ' ') return if(matches($token,'^(F?RD)$')
and not( matches(./@ism:classification,'^(TS|S|C)$')) then $token else null )=0 "
flag="error">
[ISM-ID-00174][Error] If ISM_CAPCO_RESOURCE and attribute
atomicEnergyMarkings contains the name token [RD] or [FRD],
then attribute classification must have a value of [TS], [S], or [C].

Human Readable: USA documents with RD or FRD data must be marked CLASSIFIED,
SECRET, or TOP SECRET.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00175

FileName:../Rules/atomicEnergyMarkings/ISM\_ID\_00175.sch

### Rule Description:

[ISM-ID-00175][Error] If ISM\_CAPCO\_RESOURCE and attribute atomicEnergyMarkings contains the name token [RD-CNWDI], then attribute classification must have a value of [TS] or [S].

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute atomicEnergyMarkings without a value of [RD-CNWDI] then we return true because the rule does not apply. Otherwise we make sure the attribute classification is specified with a value of [S] or [TS] on the same element.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00175">

<sch:rule context="*[@ism:atomicEnergyMarkings]">
<sch:assert id="ISM-00175"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(not(index-of(tokenize(string(./
@ism:atomicEnergyMarkings),' '), 'RD-CNWDI')>0)) then true() else matches(./
@ism:classification, '^(TS|S)$') "
flag="error">
[ISM-ID-00175][Error] If ISM_CAPCO_RESOURCE and attribute
atomicEnergyMarkings contains the name token [RD-CNWDI], then attribute
classification must have a value of [TS] or [S].
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00176

FileName:../Rules/atomicEnergyMarkings/ISM\_ID\_00176.sch

### Rule Description:

[ISM-ID-00176][Error] If ISM\_CAPCO\_RESOURCE and attribute atomicEnergyMarkings has a name token containing [RD] or [FRD], then attributes declassDate and declassEvent cannot be specified on the resourceElement. Human Readable: Automatic declassification of documents containing RD or FRD information is prohibited. Attributes declassDate and declassEvent cannot be used in the classification authority block when RD or FRD is present.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute atomicEnergyMarkings with a value containing [RD] or [FRD] then we make sure that the resourceElement does not have attributes declassDate or declassEvent specified.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00176">

<sch:rule context="*[@ism:atomicEnergyMarkings]">
<sch:assert id="ISM-00176"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else count( for $each in
tokenize(string(./@ism:atomicEnergyMarkings),' ') return if(matches($each, '^(F?RD)$'))
then if($ISM_RESOURCE_ELEMENT/@ism:declassDate or $ISM_RESOURCE_ELEMENT/@ism:declassEvent)
then 1 else null else null )=0 "
flag="error">
[ISM-ID-00176][Error] Automatic declassification of documents containing RD or FRD
information is prohibited.
Attributes declassDate and declassEvent cannot be used in the classification authority
block when RD or FRD is present.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00177

FileName:./Rules/SCIcontrols/ISM\_ID\_00177.sch

### Rule Description:

[ISM-ID-00177][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols contains the name token [SI-ECI], then it must also contain the name token [SI]. Human Readable: A USA document containing Special Intelligence (SI) ECI compartment data must also specify that it contains SI data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIcontrols specified with a value containing [SI-ECI], then we make sure that attribute SCIcontrols also contains the value [SI].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00177">

<sch:rule context="*[@ism:SCIcontrols]">
<sch:assert id="ISM-00177"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if( count( for $each in
tokenize(string(./@ism:SCIcontrols),' ') return if(matches($each,'^SI-ECI')) then 1 else
null )>0 ) then index-of(tokenize(string(./@ism:SCIcontrols),' '), 'SI')>0 else
true() "
flag="error">
[ISM-ID-00177][Error] If ISM_CAPCO_RESOURCE and attribute SCIcontrols contains the name
token [SI-ECI],
then it must also contain the name token [SI].

Human Readable: A USA document containing Special Intelligence (SI) ECI compartment data
must also
specify that it contains SI data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00178

FileName:./Rules/atomicEnergyMarkings/ISM\_ID\_00178.sch

### Rule Description:

[ISM-ID-00178][Error] If ISM\_CAPCO\_RESOURCE and attribute atomicEnergyMarkings is specified, then each of its values must be ordered in accordance with CVEnumISMAAtomicEnergyMarkings.xml.

### Code Description:

To perform sorting, this rule first retrieves the CVE values for the attribute to be sorted, which in this case is atomicEnergyMarkings. Then, each attribute token is converted into a numerical value based on its characters. Next, each attribute token is given an order number, which compares its position to that of its value in the CVE file. Next, each order number is compared to that of its previous sibling to determine if the tokens are in order. If a token is found whose order number is less than that of its previous sibling, 0 is returned for its sorted order number. If a token's order number is greater than that of its previous sibling, 1 is returned. If two tokens have the same order number, their original attribute values are compared. If the original attribute value contains numbers then the comparison is made on its converted numerical value; otherwise, the comparison is made on its string value. If an attribute value is found whose value is less than that of its previous sibling, 0 is returned for its sorted order number; otherwise 2 is returned. Finally, if any tokens are found with 0 as its sorted order number, then the rule fails as those tokens are out of order.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00178">

  <sch:rule context="*[@ism:atomicEnergyMarkings]">
    <!-- Define variables -->
    <sch:let name="capcoRestriction" value="true()"/>
    <sch:let name="errMsg_AlphabeticalOrder"
value=" '[ISM-ID-00178][Error] If ISM_CAPCO_RESOURCE and attribute atomicEnergyMarkings
is specified, then each of its values must be ordered in accordance with
CVEnumISMAAtomicEnergyMarkings.xml.' "/>

    <sch:let name="dataFileElems" value="$atomicEnergyMarkingsList"/>
    <sch:let name="attrValues" value="./@ism:atomicEnergyMarkings"/>
    <sch:let name="attrValueTokens" value="tokenize(string($attrValues),' ')/>

    <!-- Convert each character to a numerical value, then concatenate the results to form a
number-string -->
    <sch:let name="convertStrToNum"
value=" for $token in $attrValueTokens return number(string-join( for $index in 1
to string-length($token) return for $char in substring($token, $index, 1) return if
(contains(string('0123456789'), $char)) then $char else if (contains(string('ABCDEFGHI'),
$char)) then translate(string($char), 'ABCDEFGHI', '123456789') else if
(contains(string('JKLMNOPQRS'), $char)) then concat('1', translate(string($char),
'JKLMNOPQRS', '0123456789')) else if (contains(string('TUVWXYZ'), $char)) then concat('2',
translate(string($char), 'TUVWXYZ', '0123456')) else '0' , '')) "/>
```



```

<!-- Get the position of each client node relative to its position in the master list. If
the node is not found, return a -1 -->
<sch:let name="orderNums"
value=" for $token in $attrValueTokens return if
($dataFileElems[matches($token,concat('^',text(),'$'))]) then
count(($dataFileElems[matches($token,concat('^',text(),'$'))])/preceding::*) + 1 else -1"/
>

<!-- Create a sequence that returns a 0 if the previous sibling has a higher order number,
else return a 1 -->
<sch:let name="sortedOrderNums"
value=" for $index in distinct-values(for $token in $orderNums return index-of($orderNums,
$token)) return if($index = 1 or $orderNums[$index] > $orderNums[$index - 1])
then 1 else if ($orderNums[$index] < $orderNums[$index - 1]) then 0 else if
(matches($attrValueTokens[$index], '[0-9]') or matches($attrValueTokens[$index - 1],
'[0-9]')) then if ($convertStrToNum[$index - 1] > $convertStrToNum[$index]) then 0 else
2 else if (compare($attrValueTokens[$index - 1],$attrValueTokens[$index])=1) then 0 else 2
"/>
<sch:let name="hasUnsorted" value="count(index-of($sortedOrderNums,0)) > 0"/>
<sch:let name="unsortedValues"
value=" if ($hasUnsorted) then distinct-values( for $token in index-of($sortedOrderNums,0)
return $attrValueTokens[$token] ) else null "/>

<sch:assert id="ISM-00178" test="not($hasUnsorted)" flag="error">
<sch:value-of select="$errMsg_AlphabeticalOrder"/>
The following values are out of order [<sch:value-of select="$unsortedValues"/>] for
[<sch:value-of select="$attrValueTokens"/>] </sch:assert>
</sch:rule>
</sch:pattern>

```

## Rule: ISM-ID-00179

FileName:./Rules/generalConstraints/ISM\_ID\_00179.sch

### Rule Description:

[ISM-ID-00179][Warning] Attribute disseminationControls contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate, determine if the values are being used but it is still prior to the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00179">

<sch:rule context="*[@ism:disseminationControls]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMDissem.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="false()"/>

<sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:disseminationControls), $depTerms,
$ISM_RESOURCE_CREATE_DATE, $isError) "/>

<sch:assert id="ISM-00179" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00179][Warning] For attribute disseminationControls, value(s) <sch:value-of
select="$reportWarn"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00180

FileName:./Rules/generalConstraints/ISM\_ID\_00180.sch

### Rule Description:

[ISM-ID-00180][Error] Attribute disseminationControls must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date..

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00180">

<sch:rule context="*[@ism:disseminationControls]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMDissem.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="true()"/>

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:disseminationControls), $depTerms,
$ISM_RESOURCE_CREATE_DATE, $isError) "/>

<sch:assert id="ISM-00180" test="count($reportErr)=0" flag="error">
[ISM-ID-00180][Error] For attribute disseminationControls, value(s) <sch:value-of
select="$reportErr"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00181

FileName:../Rules/atomicEnergyMarkings/ISM\_ID\_00181.sch

### Rule Description:

[ISM-ID-00181][Error] If ISM\_CAPCO\_RESOURCE and element's classification does not have a value of "U" then attribute atomicEnergyMarkings must not contain the name token [UCNI]. Human Readable: UCNI may only be used on UNCLASSIFIED portions.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check the classification attribute of the current element. If it has a value of [U] we return true since this rule only applies to classified elements. If it is not [U] then we check that the attribute atomicEnergyMarkings does not have a value of [UCNI].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00181">

<sch:rule context="*[@ism:atomicEnergyMarkings]">
<sch:assert id="ISM-00181"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(./@ism:classification='U') then
true() else not(index-of(tokenize(string(./@ism:atomicEnergyMarkings), ' '), 'UCNI')>0)
"
flag="error">
[ISM-ID-00181][Error] UCNI may only be used on UNCLASSIFIED portions.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00182

FileName:../Rules/atomicEnergyMarkings/ISM\_ID\_00182.sch

### Rule Description:

[ISM-ID-00182][Error] If ISM\_CAPCO\_RESOURCE and element's classification does not have a value of "U" then attribute atomicEnergyMarkings must not contain the name token [DCNI]. Human Readable: DCNI may only be used on Unclassified portions.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check the classification attribute of the current element. If it has a value of [U] we return true since this rule only applies to classified elements. If it is not [U] then we check that the attribute atomicEnergyMarkings does not have a value of [DCNI].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00182">

<sch:rule context="*[@ism:atomicEnergyMarkings]">
<sch:assert id="ISM-00182"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(./@ism:classification='U') then
true() else not(index-of(tokenize(string(./@ism:atomicEnergyMarkings), ' '), 'DCNI')>0)
"
flag="error">
[ISM-ID-00182][Error] DCNI may only be used on Unclassified portions.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00183

FileName:./Rules/atomicEnergyMarkings/ISM\_ID\_00183.sch

### Rule Description:

[ISM-ID-00183][Error] If ISM\_CAPCO\_RESOURCE and attribute atomicEnergyMarkings contains the name token [RD-SG], then it must also contain the name token [RD].

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute atomicEnergyMarkings with a value of [RD-SG] then we also have a value of [RD].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00183">

<sch:rule context="*[@ism:atomicEnergyMarkings]">
<sch:let name="atc_tok" value="tokenize(string(./@ism:atomicEnergyMarkings), ' ')" />
<sch:assert id="ISM-00183"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(count(for $each in $atc_tok
return if(matches($each, '^RD-SG')) then $each else null)>0) then index-of($atc_tok,
'RD')>0 else true() "
flag="error">
[ISM-ID-00183][Error] If ISM_CAPCO_RESOURCE and attribute atomicEnergyMarkings contains
the name token [RD-SG],
then it must also contain the name token [RD].
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00184

FileName:./Rules/atomicEnergyMarkings/ISM\_ID\_00184.sch

### Rule Description:

[ISM-ID-00184][Error] If ISM\_CAPCO\_RESOURCE and attribute atomicEnergyMarkings contains the name token [FRD-SG], then it must also contain the name token [FRD].

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute atomicEnergyMarkings with a value of [FRD-SG] then we also have a value of [FRD].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00184">

<sch:rule context="*[@ism:atomicEnergyMarkings]">
<sch:let name="atc_tok" value="tokenize(string(./@ism:atomicEnergyMarkings), ' ')" />
<sch:assert id="ISM-00184"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(count(for $each in $atc_tok
return if(matches($each, '^FRD-SG')) then $each else null)>0) then index-of($atc_tok,
'FRD')>0 else true() "
flag="error">
[ISM-ID-00184][Error] If ISM_CAPCO_RESOURCE and attribute atomicEnergyMarkings contains
the name token [FRD-SG],
then it must also contain the name token [FRD].
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00185

FileName:../Rules/atomicEnergyMarkings/ISM\_ID\_00185.sch

### Rule Description:

[ISM-ID-00185][Error] If ISM\_CAPCO\_RESOURCE and attribute atomicEnergyMarkings contains the name token [RD-CNWDI], then it must also contain the name token [RD].

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute atomicEnergyMarkings with a value of [RD-CNWDI] then we also have a value of [RD].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00185">

<sch:rule context="*[@ism:atomicEnergyMarkings]">
<sch:let name="atc_tok" value="tokenize(string(./@ism:atomicEnergyMarkings), ' ')" />
<sch:assert id="ISM-00185"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if(index-of($atc_tok, 'RD-
CNWDI')>0) then index-of($atc_tok, 'RD')>0 else true() "
flag="error">
[ISM-ID-00185][Error] If ISM_CAPCO_RESOURCE and attribute atomicEnergyMarkings contains
the name token [RD-CNWDI],
then it must also contain the name token [RD].
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00186

FileName:./Rules/SCIcontrols/ISM\_ID\_00186.sch

### Rule Description:

[ISM-ID-00186][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols contains the name token [SI-G-XXXX], then it must also contain the name token [SI-G]. Human Readable: A USA document that contains Special Intelligence (SI) GAMMA sub-compartments must also specify that it contains SI-GAMMA compartment data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIcontrols specified with a value containing [SI-G-XXXX], where X is represented by the regular expression character class [A-Z], then we make sure that attribute SCIcontrols also contains the value [SI-G].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00186">

<sch:rule context="*[@ism:SCIcontrols]">
<sch:assert id="ISM-00186"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if( count( for $each in
tokenize(string(./@ism:SCIcontrols),' ') return if(matches($each,'^SI-G-[A-Z][A-Z][A-Z][A-
Z]')) then 1 else null )>0 ) then index-of(tokenize(string(./@ism:SCIcontrols),' '),
'SI-G')>0 else true() "
flag="error">
[ISM-ID-00186][Error] If ISM_CAPCO_RESOURCE and attribute SCIcontrols contains the name
token [SI-G-XXXX],
then it must also contain the name token [SI-G].

Human Readable: A USA document that contains Special Intelligence (SI) GAMMA sub-
compartments must
also specify that it contains SI-GAMMA compartment data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00187

FileName:./Rules/SCIcontrols/ISM\_ID\_00187.sch

### Rule Description:

[ISM-ID-00187][Error] If ISM\_CAPCO\_RESOURCE and attribute SCIcontrols contains the name token [SI-G], then it must also contain the name token [SI]. Human Readable: A USA document that contains Special Intelligence (SI) -GAMMA compartment data must also specify that it contains SI data.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the current element has attribute SCIcontrols specified with a value containing [SI-G], then we make sure that attribute SCIcontrols also contains the value [SI].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00187">

<sch:rule context="*[@ism:SCIcontrols]">
<sch:assert id="ISM-00187"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else if( count( for $each in
tokenize(string(./@ism:SCIcontrols),' ') return if(matches($each,'^SI-G')) then 1 else
null )>0 ) then index-of(tokenize(string(./@ism:SCIcontrols),' '), 'SI')>0 else
true() "
flag="error">
[ISM-ID-00187][Error] If ISM_CAPCO_RESOURCE and attribute SCIcontrols contains the name
token [SI-G],
then it must also contain the name token [SI].

Human Readable: A USA document that contains Special Intelligence (SI) -GAMMA compartment
data must also specify that
it contains SI data.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00188

FileName:../Rules/generalConstraints/ISM\_ID\_00188.sch

### Rule Description:

[ISM-ID-00188][Warning] Attribute FGIsorceOpen contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate, determine if the values are being used but it is still prior to the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00188">

<sch:rule context="*[@ism:FGIsorceOpen]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMFGIOpen.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="false()"/>

<sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:FGIsorceOpen), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) "/>

<sch:assert id="ISM-00188" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00188][Warning] For attribute FGIsorceOpen, value(s) <sch:value-of
select="$reportWarn"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00189

FileName:./Rules/generalConstraints/ISM\_ID\_00189.sch

### Rule Description:

[ISM-ID-00189][Error] Attribute FGIsSourceOpen must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00189">

<sch:rule context="*[@ism:FGIsSourceOpen]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMFGIOpen.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="true()"/>

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:FGIsSourceOpen), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) "/>

<sch:assert id="ISM-00189" test="count($reportErr)=0" flag="error">
[ISM-ID-00189][Error] For attribute FGIsSourceOpen, value(s) <sch:value-of
select="$reportErr"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00190

FileName:./Rules/generalConstraints/ISM\_ID\_00190.sch

### Rule Description:

[ISM-ID-00190][Warning] Attribute FGIsSourceProtected contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate, determine if the values are being used but it is still prior to the deprecated date

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00190">

  <sch:rule context="*[@ism:FGIsSourceProtected]">

    <sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMFGIProtected.xml')//cve:CVE/cve:Enumeration/
cve:Term[./@deprecated]" />
    <sch:let name="isError" value="false()" />

    <sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:FGIsSourceProtected), $depTerms,
$ISM_RESOURCE_CREATE_DATE, $isError) " />

    <sch:assert id="ISM-00190" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00190][Warning] For attribute FGIsSourceProtected, value(s) <sch:value-of
select="$reportWarn" />
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00191

FileName:../Rules/generalConstraints/ISM\_ID\_00191.sch

### Rule Description:

[ISM-ID-00191][Error] Attribute FGIsSourceProtected must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00191">

<sch:rule context="*[@ism:FGIsSourceProtected]">

<sch:let name="depTerms"
value="document(' ../../CVE/ISM/CVEnumISMFGIProtected.xml')//cve:CVE/cve:Enumeration/
cve:Term[./@deprecated]" />
<sch:let name="isError" value="true()" />

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:FGIsSourceProtected), $depTerms,
$ISM_RESOURCE_CREATE_DATE, $isError) " />

<sch:assert id="ISM-00191" test="count($reportErr)=0" flag="error">
[ISM-ID-00191][Error] For attribute FGIsSourceProtected, value(s) <sch:value-of
select="$reportErr" />
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00192

FileName:../Rules/generalConstraints/ISM\_ID\_00192.sch

### Rule Description:

[ISM-ID-00192][Warning] Attribute nonICmarkings contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate, determine if the values are being used but it is still prior to the deprecated date

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00192">

  <sch:rule context="*[@ism:nonICmarkings]">

    <sch:let name="depTerms"
value="document(' ../../CVE/ISM/CVEnumISMNonIC.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
    <sch:let name="isError" value="false()"/>

    <sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:nonICmarkings), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) "/>

    <sch:assert id="ISM-00192" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00192][Warning] For attribute nonICmarkings, value(s) <sch:value-of
select="$reportWarn"/>
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00193

FileName:./Rules/generalConstraints/ISM\_ID\_00193.sch

### Rule Description:

[ISM-ID-00193][Error] Attribute nonICmarkings must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00193">

<sch:rule context="*[@ism:nonICmarkings]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMNonIC.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="true()"/>

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:nonICmarkings), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) "/>

<sch:assert id="ISM-00193" test="count($reportErr)=0" flag="error">
[ISM-ID-00193][Error] For attribute nonICmarkings, value(s) <sch:value-of
select="$reportErr"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00194

FileName:../Rules/generalConstraints/ISM\_ID\_00194.sch

### Rule Description:

[ISM-ID-00194][Warning] Attribute notice contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate, determine if the values are being used but it is still prior to the deprecated date

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00194">

<sch:rule context="*[@ism:noticeType]">

<sch:let name="depTerms"
value="document(' ../../CVE/ISM/CVEnumISMNotice.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="false()"/>

<sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:noticeType), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) "/>

<sch:assert id="ISM-00194" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00194][Warning] For attribute notice, value(s) <sch:value-of select="$reportWarn"/
>
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00195

FileName:../Rules/generalConstraints/ISM\_ID\_00195.sch

### Rule Description:

[ISM-ID-00195][Error] Attribute notice must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00195">

<sch:rule context="*[@ism:noticeType]">

<sch:let name="depTerms"
value="document(' ../../CVE/ISM/CVEnumISMNotice.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="true()"/>

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:noticeType), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) "/>

<sch:assert id="ISM-00195" test="count($reportErr)=0" flag="error">
[ISM-ID-00195][Error] For attribute notice, value(s) <sch:value-of select="$reportErr"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00196

FileName:../Rules/generalConstraints/ISM\_ID\_00196.sch

### Rule Description:

[ISM-ID-00196][Warning] Attribute ownerProducer contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate, determine if the values are being used but it is still prior to the deprecated date

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00196">

<sch:rule context="*[@ism:ownerProducer]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMOwnerProducer.xml')//cve:CVE/cve:Enumeration/
cve:Term[./@deprecated]" />
<sch:let name="isError" value="false()" />

<sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:ownerProducer), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) " />

<sch:assert id="ISM-00196" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00196][Warning] For attribute ownerProducer, value(s) <sch:value-of
select="$reportWarn" />
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00197

FileName:../Rules/generalConstraints/ISM\_ID\_00197.sch

### Rule Description:

[ISM-ID-00197][Error] Attribute ownerProducer must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00197">

<sch:rule context="*[@ism:ownerProducer]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMOwnerProducer.xml')//cve:CVE/cve:Enumeration/
cve:Term[./@deprecated]" />
<sch:let name="isError" value="true()" />

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:ownerProducer), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) " />

<sch:assert id="ISM-00197" test="count($reportErr)=0" flag="error">
[ISM-ID-00197][Error] For attribute ownerProducer, value(s) <sch:value-of
select="$reportErr" />
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00198

FileName:../Rules/generalConstraints/ISM\_ID\_00198.sch

### Rule Description:

[ISM-ID-00198][Warning] Attribute releasableTo contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate, determine if the values are being used but it is still prior to the deprecated date

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00198">

<sch:rule context="*[@ism:releasableTo]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMRelTo.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="false()"/>

<sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:releasableTo), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) "/>

<sch:assert id="ISM-00198" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00198][Warning] For attribute releasableTo, value(s) <sch:value-of
select="$reportWarn"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00199

FileName:./Rules/generalConstraints/ISM\_ID\_00199.sch

### Rule Description:

[ISM-ID-00199][Error] Attribute releasableTo must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00199">

<sch:rule context="*[@ism:releasableTo]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMRelTo.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="true()"/>

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:releasableTo), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) "/>

<sch:assert id="ISM-00199" test="count($reportErr)=0" flag="error">
[ISM-ID-00199][Error] For attribute releasableTo, value(s) <sch:value-of
select="$reportErr"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00200

FileName:../Rules/generalConstraints/ISM\_ID\_00200.sch

### Rule Description:

[ISM-ID-00200][Warning] Attribute displayOnlyTo contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate date, determine if the values are being used but it is still prior to the deprecated date

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00200">

<sch:rule context="*[@ism:displayOnlyTo]">

<sch:let name="depTerms"
value="document(' ../../CVE/ISM/CVEnumISMRelTo.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="false()"/>

<sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:displayOnlyTo), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) "/>

<sch:assert id="ISM-00200" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00200][Warning] For attribute displayOnlyTo, value(s) <sch:value-of
select="$reportWarn"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00201

FileName:../Rules/generalConstraints/ISM\_ID\_00201.sch

### Rule Description:

[ISM-ID-00201][Error] Attribute displayOnlyTo must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00201">

<sch:rule context="*[@ism:displayOnlyTo]">

<sch:let name="depTerms"
value="document(' ../../CVE/ISM/CVEnumISMRelTo.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="true()"/>

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:displayOnlyTo), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) "/>

<sch:assert id="ISM-00201" test="count($reportErr)=0" flag="error">
[ISM-ID-00201][Error] For attribute displayOnlyTo, value(s) <sch:value-of
select="$reportErr"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00202

FileName:../Rules/generalConstraints/ISM\_ID\_00202.sch

### Rule Description:

[ISM-ID-00202][Warning] Attribute SARIdentifier contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate, determine if the values are being used but it is still prior to the deprecated date

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00202">

<sch:rule context="*[@ism:SARIdentifier]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMSAR.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="false()"/>

<sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:SARIdentifier), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) "/>

<sch:assert id="ISM-00202" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00202][Warning] For attribute SARIdentifier, value(s) <sch:value-of
select="$reportWarn"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00203

FileName:./Rules/generalConstraints/ISM\_ID\_00203.sch

### Rule Description:

[ISM-ID-00203][Error] Attribute SARIdentifier must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00203">

<sch:rule context="*[@ism:SARIdentifier]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMSAR.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="true()"/>

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:SARIdentifier), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) "/>

<sch:assert id="ISM-00203" test="count($reportErr)=0" flag="error">
[ISM-ID-00203][Error] For attribute SARIdentifier, value(s) <sch:value-of
select="$reportErr"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00204

FileName:../Rules/generalConstraints/ISM\_ID\_00204.sch

### Rule Description:

[ISM-ID-00204][Warning] Attribute SCIControls contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate, determine if the values are being used but it is still prior to the deprecated date

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00204">

<sch:rule context="*[@ism:SCIControls]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMSCIControls.xml')//cve:CVE/cve:Enumeration/
cve:Term[./@deprecated]" />
<sch:let name="isError" value="false()" />

<sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:SCIControls), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) " />

<sch:assert id="ISM-00204" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00204][Warning] For attribute SCIControls, value(s) <sch:value-of
select="$reportWarn" />
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00205

FileName:./Rules/generalConstraints/ISM\_ID\_00205.sch

### Rule Description:

[ISM-ID-00205][Error] Attribute SCIcontrols must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00205">

<sch:rule context="*[@ism:SCIcontrols]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMSCIcontrols.xml')//cve:CVE/cve:Enumeration/
cve:Term[./@deprecated]" />
<sch:let name="isError" value="true()" />

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:SCIcontrols), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) " />

<sch:assert id="ISM-00205" test="count($reportErr)=0" flag="error">
[ISM-ID-00205][Error] For attribute SCIcontrols, value(s) <sch:value-of
select="$reportErr" />
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00206

FileName:../Rules/generalConstraints/ISM\_ID\_00206.sch

### Rule Description:

[ISM-ID-00206][Warning] Attribute declassException contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate, determine if the values are being used but it is still prior to the deprecated date

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00206">

  <sch:rule context="*[@ism:declassException]">

    <sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISM25X.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
    <sch:let name="isError" value="false()"/>

    <sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:declassException), $depTerms,
$ISM_RESOURCE_CREATE_DATE, $isError) "/>

    <sch:assert id="ISM-00206" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00206][Warning] For attribute declassException, value(s) <sch:value-of
select="$reportWarn"/>
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00207

FileName:../Rules/generalConstraints/ISM\_ID\_00207.sch

### Rule Description:

[ISM-ID-00207][Error] Attribute declassException must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00207">

<sch:rule context="*[@ism:declassException]">

<sch:let name="depTerms"
value="document(' ../../CVE/ISM/CVEnumISM25X.xml')//cve:CVE/cve:Enumeration/cve:Term[./
@deprecated]"/>
<sch:let name="isError" value="true()"/>

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:declassException), $depTerms,
$ISM_RESOURCE_CREATE_DATE, $isError) "/>

<sch:assert id="ISM-00207" test="count($reportErr)=0" flag="error">
[ISM-ID-00207][Error] For attribute declassException, value(s) <sch:value-of
select="$reportErr"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00208

FileName:../Rules/generalConstraints/ISM\_ID\_00208.sch

### Rule Description:

[ISM-ID-00208][Warning] Attribute atomicEnergyMarkings contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used prior to the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00208">

<sch:rule context="*[@ism:atomicEnergyMarkings]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMAtomicEnergyMarkings.xml')//cve:CVE/
cve:Enumeration/cve:Term[./@deprecated]"/>
<sch:let name="isError" value="false()"/>

<sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:atomicEnergyMarkings), $depTerms,
$ISM_RESOURCE_CREATE_DATE, $isError) "/>

<sch:assert id="ISM-00208" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00208][Warning] For attribute atomicEnergyMarkings, value(s) <sch:value-of
select="$reportWarn"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00209

FileName:./Rules/generalConstraints/ISM\_ID\_00209.sch

### Rule Description:

[ISM-ID-00209][Error] Attribute atomicEnergyMarkings must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00209">

<sch:rule context="*[@ism:atomicEnergyMarkings]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMAtomicEnergyMarkings.xml')//cve:CVE/
cve:Enumeration/cve:Term[./@deprecated]"/>
<sch:let name="isError" value="true()"/>

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:atomicEnergyMarkings), $depTerms,
$ISM_RESOURCE_CREATE_DATE, $isError) "/>

<sch:assert id="ISM-00209" test="count($reportErr)=0" flag="error">
[ISM-ID-00209][Error] For attribute atomicEnergyMarkings, value(s) <sch:value-of
select="$reportErr"/>
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00210

FileName:../Rules/generalConstraints/ISM\_ID\_00210.sch

### Rule Description:

[ISM-ID-00210][Warning] Attribute nonUSControls contains a value that will be deprecated.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate, determine if the values are being used but it is still prior to the deprecated date

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00210">

<sch:rule context="*[@ism:nonUSControls]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMNonUSControls.xml')//cve:CVE/cve:Enumeration/
cve:Term[./@deprecated]" />
<sch:let name="isError" value="false()" />

<sch:let name="reportWarn"
value=" dvf:deprecated(string(./@ism:nonUSControls), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) " />

<sch:assert id="ISM-00210" test="count($reportWarn)=0" flag="warning">
[ISM-ID-00210][Warning] For attribute nonUSControls, value(s) <sch:value-of
select="$reportWarn" />
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00211

FileName:../Rules/generalConstraints/ISM\_ID\_00211.sch

### Rule Description:

[ISM-ID-00211][Error] Attribute nonUSControls must not contain values that have passed their deprecation date.

### Code Description:

Traverse the CVE file pulling out deprecated values and their dates. Using the ism:createDate determine if the values are being used passed the deprecated date.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00211">

<sch:rule context="*[@ism:nonUSControls]">

<sch:let name="depTerms"
value="document('.../CVE/ISM/CVEnumISMNonUSControls.xml')//cve:CVE/cve:Enumeration/
cve:Term[./@deprecated]" />
<sch:let name="isError" value="true()" />

<sch:let name="reportErr"
value=" dvf:deprecated(string(./@ism:nonUSControls), $depTerms, $ISM_RESOURCE_CREATE_DATE,
$isError) " />

<sch:assert id="ISM-00211" test="count($reportErr)=0" flag="error">
[ISM-ID-00211][Error] For attribute nonUSControls, value(s) <sch:value-of
select="$reportErr" />
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00213

FileName:./Rules/disseminationControls/ISM\_ID\_00213.sch

### Rule Description:

[ISM-ID-00213][Error] If ISM\_CAPCO\_RESOURCE and attribute disseminationControls contains the name token [DISPLAYONLY], then attribute displayOnlyTo must be specified. Human Readable: A USA document with DISPLAY ONLY dissemination must indicate the countries to which it can be disseminated.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute disseminationControls with a value of [DISPLAYONLY] then the attribute displayOnlyTo is specified and does not have an empty value set.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00213">

<sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00031"
test=" if(index-of(tokenize(string(..@ism:disseminationControls),' '), 'DISPLAYONLY')>0)
then ../@ism:displayOnlyTo and not(..@ism:displayOnlyTo = '') else true() "
flag="error">
[ISM-ID-00213][Error] If ISM_CAPCO_RESOURCE and attribute
disseminationControls contains the name token [DISPLAYONLY], then attribute displayOnlyTo
must be specified.

Human Readable: A USA document with DISPLAY ONLY dissemination must indicate the countries
to which it can be disseminated.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00214

FileName:../Rules/releasableTo/ISM\_ID\_00214.sch

### Rule Description:

[ISM-ID-00214][Error] If ISM\_CAPCO\_RESOURCE then attribute releasableTo must start with [USA].

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If the attribute releasableTo is specified, then we make sure that it starts with [USA].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00214">

<sch:rule context="*[@ism:releasableTo]">

<sch:let name="dissemTok" value="tokenize(string(./@ism:disseminationControls),' ')" />
<sch:assert id="ISM-00032"
test=" if(not($ISM_CAPCO_RESOURCE)) then true() else index-of(tokenize(string(./
@ism:releasableTo),' '), 'USA')=1 "
flag="error">
[ISM-ID-00214][Error] If ISM_CAPCO_RESOURCE then attribute
releasableTo must start with [USA].
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00215

FileName:./Rules/disseminationControls/ISM\_ID\_00215.sch

### Rule Description:

[ISM-ID-00215][Error] If ISM\_CAPCO\_RESOURCE and attribute disseminationControls contains the name token [DISPLAYONLY], then attribute classification must have a value of [TS], [S], or [C]. Human Readable: USA documents with DISPLAYONLY dissemination must be classified CLASSIFIED, SECRET, or TOP SECRET.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. If it is a CAPCO resource then we check that if we have an element having attribute disseminationControls with a value of [DISPLAYONLY] then we also have the attribute classification specified with a value of [C], [S], or [TS] on the same element.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00215">

<sch:rule context="*[@ism:disseminationControls and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00215"
test=" if(index-of(tokenize(string(./@ism:disseminationControls), '
'), 'DISPLAYONLY')>0) then matches(./@ism:classification, '^(TS|S|C)$') else true()"
flag="error">
[ISM-ID-00215][Error] If ISM_CAPCO_RESOURCE and attribute
disseminationControls contains the name token [DISPLAYONLY],
then attribute classification must have a value of [TS], [S], or [C].

Human Readable: USA documents with DISPLAYONLY dissemination must be classified
CLASSIFIED, SECRET, or TOP SECRET.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00217

FileName:./Rules/FGIsourceProtected/ISM\_ID\_00217.sch

### Rule Description:

[ISM-ID-00217][Error] If ISM\_CAPCO\_RESOURCE attribute FGIsourceProtected contains [FGI], it must be the only value.

### Code Description:

If CAPCO rules do not apply to the document then the rule does not apply and we return true. Otherwise, we make sure that the current element has attribute FGIsourceProtected specified with [FGI] as its only value.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00217">

<sch:rule context="*[@ism:FGIsourceProtected and $ISM_CAPCO_RESOURCE]">
<sch:let name="opTok" value="tokenize(string(@ism:FGIsourceProtected), ' ')" />
<sch:assert id="ISM-00217"
test=" not(index-of($opTok, 'FGI')>0 and count($opTok)>1) "
flag="error">
[ISM-ID-00217][Error] If ISM_CAPCO_RESOURCE attribute FGIsourceProtected contains [FGI],
it must be the only value.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00219

FileName:./Rules/ownerProducer/ISM\_ID\_00219.sch

### Rule Description:

[ISM-ID-00219][Error] If element meets ISM\_CONTRIBUTES and attribute ownerProducer contains [FGI], then FGIsorceProtected must have a value of [FGI]. Human Readable: Any non-resource element that contributes to the document's banner roll-up and has FOREIGN GOVERNMENT INFORMATION (FGI) must also specify attribute FGIsorceProtected with token FGI.

### Code Description:

If not ISM\_CONTRIBUTES then return true, otherwise, we make sure that if the current element has attribute ownerProducer specified with [FGI] then FGIsorceProtected also has a value of [FGI].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00219">

<sch:rule context="*[@ism:ownerProducer and not(generate-id(.) = generate-
id($ISM_RESOURCE_ELEMENT)) and not(@ism:excludeFromRollup=true())]">
<sch:assert id="ISM-00219"
test=" if(index-of(./@ism:ownerProducer,'FGI')>0) then index-of(./
@ism:FGIsorceProtected,'FGI')>0 else true() "
flag="error">
[ISM-ID-00219][Error] If element meets ISM_CONTRIBUTES and attribute ownerProducer
contains [FGI],
then FGIsorceProtected must have a value of [FGI].

Human Readable: Any non-resource element that contributes to the document's banner roll-up
and has
FOREIGN GOVERNMENT INFORMATION (FGI) must also specify attribute FGIsorceProtected with
token FGI.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00221

FileName:./Rules/derivativelyClassifiedBy/ISM\_ID\_00221.sch

### Rule Description:

[ISM-ID-00221][Error] If ISM\_CAPCO\_RESOURCE and attribute derivativelyClassifiedBy is specified, then attribute classificationReason must not be specified. Human Readable: USA documents that are derivatively classified must not specify a classification reason.

### Code Description:

For each element with the attribute derivativelyClassifiedBy specified, we make sure that the attribute classificationReason is not specified.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00221">

<sch:rule context="*[@ism:derivativelyClassifiedBy and $ISM_CAPCO_RESOURCE]">
<sch:assert id="ISM-00221" test="not(@ism:classificationReason)" flag="error">
[ISM-ID-00221][Error] If ISM_CAPCO_RESOURCE and attribute
derivativelyClassifiedBy is specified, then attribute classificationReason
must not be specified.

Human Readable: USA documents that are derivatively classified must not
specify a classification reason.
</sch:assert>
</sch:rule>
</sch:pattern>
```



## Rule: ISM-ID-00222

FileName:./Rules/compliesWith/ISM\_ID\_00222.sch

### Rule Description:

[ISM-ID-00222][Error] If ISM\_ICDOCUMENT\_APPLIES, then the pocType attribute with value [ICD-710] must also be specified on some element in the document. Human Readable: A document claiming compliance with ICD-710 must specify a point-of-contact to whom questions about the document can be directed.

### Code Description:

If ISM\_ICDOCUMENT\_APPLIES, then ensure that some element specifies attribute @pocType with value 'ICD-710'.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00222">

  <sch:rule context="/*[$ISM_ICDOCUMENT_APPLIES]">
    <sch:assert id="ISM-00222" test="index-of($partPocType_tok, 'ICD-710') > 0"
      flag="error">
      [ISM-ID-00222][Error] If ISM_ICDOCUMENT_APPLIES, then the pocType attribute with value
      [ICD-710]
      must also be specified on some element in the document.

      Human Readable: A document claiming compliance with ICD-710 must specify a point-of-
      contact
      to whom questions about the document can be directed.
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00223

FileName:./Rules/generalConstraints/ISM\_ID\_00223.sch

### Rule Description:

[ISM-ID-00223][Error] If any elements in namespace urn:us:gov:ic:ism exist, the local name must exist in CVEnumISMElements.xml. Human Readable: Ensure that elements in the ISM namespace are defined by ISM.XML.

### Code Description:

To determine the valid values, this rule first retrieves the list of valid element names as defined in CVEnumISMElements.xml. The test will pass if there exists in the list an element name that matches the name of the current element.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00223">

    <sch:rule context="ism:*[$ISM_CAPCO_RESOURCE]">
    <!-- Define variables -->
    <sch:let name="errMsg_ValueNotFound"
value="" [ISM-ID-00223][Error] If any elements in namespace urn:us:gov:ic:ism exist, the
local name must exist in CVEnumISMElements.xml. ""/>

    <sch:let name="localName" value="local-name()"/>

    <!-- Execute tests -->
    <sch:assert id="ISM-00223" flag="error"
test="exists($validElementList[text() = $localName])">
    <sch:value-of select="normalize-space(string($errMsg_ValueNotFound))"/>
Invalid value of [<sch:value-of select="name()"/>]</sch:assert>

    </sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00224

FileName:./Rules/pocType/ISM\_ID\_00224.sch

### Rule Description:

[ISM-ID-00224][Error] If ISM\_CAPCO\_RESOURCE and any element meeting ISM\_CONTRIBUTES in the document has the attribute disseminationControls containing [OC], then the attribute @ism:pocType with value [ORCON] must be specified on some element in the document. Human Readable: In accordance with the ORCON Memo dated March 11, 2011, USA documents containing ORIGINATOR CONTROLLED data must specify a point-of-contact to whom adjudication decisions about those data can be directed.

### Code Description:

The rule will apply to the resource element of a CAPCO document that was created after the date after which ORCON points-of-contact became required. For this element, if ORCON data is found, then the code checks if the attribute @ism:pocType is specified with the value 'ORCON' on some element. Otherwise, the rule does not apply and true is returned.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00224">

<sch:rule context="*[generate-id(.)=generate-id($ISM_RESOURCE_ELEMENT) and
$ISM_CAPCO_RESOURCE and $ISM_RESOURCE_CREATE_DATE and $ISM_RESOURCE_CREATE_DATE >
$ISM_ORCON_POC_DATE and $bannerDisseminationControls_tok='OC' ]">
<sch:assert id="ISM-00224" test=" $partPocType_tok='ORCON' "
flag="error">
[ISM-ID-00224][Error] If ISM_CAPCO_RESOURCE and any element meeting ISM_CONTRIBUTES
in the document has the attribute disseminationControls containing [OC], then the
attribute @ism:pocType with value [ORCON] must be specified on some element in the
document.

Human Readable: After March 11, 2011, USA documents containing ORIGINATOR CONTROLLED
data must specify a point-of-contact to whom adjudication decisions about
those data can be directed.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00225

FileName:../Rules/nonICmarkings/ISM\_ID\_00225.sch

### Rule Description:

[ISM-ID-00225][Error] If ISM-ICDOCUMENT-APPLIES, then attribute ism:nonICmarkings must not be specified with a value containing any name token starting with [ACCM]. Human Readable: ACCM tokens are not valid for documents that claim compliance with IC rules.

### Code Description:

For each element which specifies attribute ism:nonICmarkings, if \$ISM-ICDOCUMENT-APPLIES, then we make sure that attribute ism:nonICmarkings is not specified with a value containing a token which starts with [ACCM].

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00225">

  <sch:rule context="*[@ism:nonICmarkings]">
    <sch:assert id="ISM-00225"
      test=" if($ISM_ICDOCUMENT_APPLIES) then not(some $token in tokenize(normalize-
        space(string(@ism:nonICmarkings)), ' ') satisfies starts-with($token, 'ACCM')) else true()
      "
      flag="error">
      [ISM-ID-00225][Error] If ISM-ICDOCUMENT-APPLIES, then attribute ism:nonICmarkings must not
      be specified with a value containing any name token starting with [ACCM].

      Human Readable: ACCM tokens are not valid for documents that claim compliance with IC
      rules.
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00226

FileName:../Rules/generalConstraints/ISM\_ID\_00226.sch

### Rule Description:

[ISM-ID-00226][Error] @ism:noticeType and @ism:unregisteredNoticeType may not both be used on the same element.

Human Readable: Ensure that the ISM attributes noticeType and unregisteredNoticeType are not used on the same element.

### Code Description:

Any element that has either @ism:noticeType or @ism:unregisteredNoticeType.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00226">

<sch:rule context="*[@ism:noticeType|@ism:unregisteredNoticeType]">
<!-- Execute tests -->
<sch:assert id="ISM-00226" flag="error"
test="count(@ism:noticeType|@ism:unregisteredNoticeType)=1">
[ISM-ID-00226][Error]
@ism:noticeType and @ism:unregisteredNoticeType may not both be applied to the same
element.

Human Readable: The ISM attributes noticeType and unregisteredNoticeType
are mutually exclusive and cannot both be applied to the same element.
</sch:assert>
</sch:rule>
</sch:pattern>
```

## Rule: ISM-ID-00227

FileName:./Rules/resourceElement/ISM\_ID\_00227.sch

### Rule Description:

[ISM-ID-00227][Error] Attribute @noticeType may only appear on the resource node when it contains the values [DoD-Dist-A], [DoD-Dist-B], [DoD-Dist-C], [DoD-Dist-D], [DoD-Dist-E], [DoD-Dist-F], or [DoD-Dist-X]. Human Readable: Documents may only specify a document-level notice if it pertains to DoD Distribution.

### Code Description:

For every resource element with the @ism:noticeType attribute specified, this rule ensures that the attribute's value starts with the string 'DoD-Dist-'. Otherwise, the rule returns false.

### Schematron Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:pattern xmlns:sch="http://purl.oclc.org/dsdl/schematron" id="ISM-ID-00227">

<sch:rule context="*[generate-id(.) = generate-id($ISM_RESOURCE_ELEMENT) and
@ism:noticeType]">
<sch:assert id="ISM-00227"
test=" starts-with(normalize-space(string(@ism:noticeType)), 'DoD-Dist-') "
flag="error">
[ISM-ID-00227][Error] Attribute @noticeType may only appear on the
resource node when it contains the values [DoD-Dist-A], [DoD-Dist-B],
[DoD-Dist-C], [DoD-Dist-D], [DoD-Dist-E], [DoD-Dist-F], or [DoD-Dist-X].

Human Readable: Documents may only specify a document-level notice if
it pertains to DoD Distribution.
</sch:assert>
</sch:rule>
</sch:pattern>
```

For any questions or comments regarding the ISM rules, contact us by email at [datastandardssupport@ugov.gov](mailto:datastandardssupport@ugov.gov).