Monitoring Overlay Syntax Proposal

May 13, 2020

This document contains a proposed syntax (but not a serialization) for a new monitoring overlay SCAP file. The purpose of the monitoring overlay file is to travel with other SCAP content, specifically checking instructions, and indicate how frequently specific checks should be re-run to determine if there have been changes.

# Monitoring Overlay Syntax

The following describes the proposed syntax of a monitoring overlay file:

The **monitoring overlay file** contains:

* A unique identifier for that file
* An identifier for the version of the monitoring overlay file schema
* A version number for the file itself
* A list of one or more **file bindings**

The **file binding** contains:

* The identifier for a specific file containing check content
* A list of one or more **periodicity blocks**

A **periodicity block** contains:

* A **periodicity statement**
* A list of one or more identifiers for checks contained within the indicated check file

A **periodicity statement** is one of the following:

* "on-connect"
* "on-change"
* A time interval following one of the following forms and sub-fields:
	+ "monthly"
		- number of months between checks
		- day of the month to check
		- an optional time to run the check
		- whether the check should also be re-run on connection, always or only if a check was missed
	+ "weekly"
		- number of weeks between checks
		- day of the week to check
		- an optional time to run the check
		- whether the check should also be re-run on connection, always or only if a check was missed
	+ "daily"
		- number of days between checks
		- an optional time to run the check
		- whether the check should also be re-run on connection, always or only if a check was missed
	+ "hourly"
		- number of hours between checks
		- an optional number of minutes past the hour to run the check
		- whether the check should also be re-run on connection, always or only if a check was missed
	+ "minutely"
		- number of minutes between checks
		- whether the check should also be re-run on connection, always or only if a check was missed

Graphically, this production looks like the following:

File Start

File Unique Identifier

File Schema Version

File Version

Check File Identifier

"On
Change"

"On
Connect"

"Monthly"

"Weekly"

"Hourly"

"Daily"

"Minutely"

Day of
month

Day of
week

Months
between
checks

Weeks
between
checks

Days
between
checks

Hours
between
checks

Minutes
between
checks

Time

Minutes

Check Identifier

File End

Time

Time

"On
Connect"

"After
Miss"

**File Binding Part**

**Periodicity Block**

**Periodicity Statement**

# Syntax Interpretation

This section gets into some of the nuances of interpretation of syntax elements.

## Not all checks in an SCAP content file are necessarily subject to re-evaluation

When a monitoring overlay is bound to a file, not all check in the file will necessarily be listed for periodic rechecking. Checks not listed in the monitoring overlay file will be run during the initial baseline collection but will never be rechecked thereafter.

## Handling on-change

The on-change statement indicates that, ideally, all of the system elements that are examined when the check is run would be monitored on each targeted asset in real time. If the relevant PCEs are not capable for monitoring some of these system elements in real time, the Collector will create a timer for the smallest practical interval and re-assess those elements using that timer. The "smallest practical interval" is left to the discretion of the Collector and its knowledge of its PCEs' capabilities. In particular, the interval should not be so small that a new re-assessment is issued before the previous re-assessment has completed. Given that some PCEs and check types will take longer than others, no universal assertion can be made about what interval to use, so this decision is left to Collectors.

## Handling on-connect

The on-connect statement, either alone or as part of an interval statement, indicates that the check should be re-run whenever an asset re-binds to a Collector after a period in which it was un-bound. (For example, when it is connected to the enterprise network after a period when it was disconnected.)

The after-miss statement, when it appears as part of an interval statement, has the same meaning as "on-connect", but with the added condition the re-assessment only occurs if the asset was unavailable when the preceding re-assessment should have occurred. If no re-assessments have been missed while the asset was disconnected, then no additional re-assessment is performed and the asset will simply be re-assessed at the next scheduled interval.

If an asset targeted by the assessment becomes bound to a Collector but has never undergone the baseline assessment (either because this is a new asset or because a known asset was not connected when the initial baseline assessment was performed) then this is not a situation that triggers an "on-connect" action. Instead the Collector must perform the complete baseline assessment, including checks that are not listed in the monitoring overlay and thus not subject to periodic reassessment, after which the Collector will engage in periodic reassessments using the schedule indicated in the monitoring overlay file. In other words, if a targeted asset was not available when the original baseline was collected, when that asset becomes available for the first time it will be subject to the same baseline assessment as when the assessment request first arrived.

## Unification of timers

A Collector may be implemented such that, for checks that will be re-assessed on a common interval, that all assets within a Collector's purview be reassessed using the same timer. For example, if a particular check is to be re-run every three hours, a single timer might be used to kick-off all reassessments of targeted assets subject to that interval. This may be the case even if some assets joined late and had their initial baseline assessment performed less than three hours before the timer triggers a re-assessment. This allows all assets to be reported together, even if they joined the network at different times. It also simplifies timer management on the Collector.

## Last day of the month

For a monthly recheck schedule, if the month in which the re-assessment should occur has fewer days than the number given in the interval statement, this as treated as indicating that the reassessment should occur on the last day of the month.

## Granularity of referenced checks

The checks referenced in a monitoring overlay can refer to XCCDF Groups, XCCDF Rules, OVAL Definitions, or elements of any other checking system. (Note, however, that in the latter case the Collector must be familiar enough with the structure of that checking system to find the relevant identifiers and correctly extract the appropriate sub-portion of the file.) If a particular referenced check has multiple parts, all parts are re-assessed based on the periodicity conditions attacked to the reference check in the monitoring overlay file.

# Examples

This section contains examples of what the syntax might look like using some example serializations:

## XML

<?xml version="1.0" ?>

<MonitoringOverlay id="example.org-TestMonitoringOverlayFile"

 schemaVersion="http://scapstandards.com/monitoring\_overlay/0.1"

 fileVersion="1"

 xmlns="http://scapstandards.com/monitoring\_overlay/0.1">

 <CheckingFile id="xccdf\_org.example\_benchmark\_example1">

 <Period>

 <OnConnect/>

 <Check id="Rule\_A"/>

 <Check id="Rule\_B"/>

 <!-- Check rules A and B whenever a device binds to a collector -->

 </Period>

 <Period>

 <Monthly every\_x\_months="3" day\_of\_month="31" time="01:00"

on\_connect="AFTER\_MISS"/>

 <Check id="Rule\_C"/>

 <!-- Check rule C on the last day of every quarter at 1AM. Check immediately on

connect if assessment is missed. -->

 </Period>

 <Period>

 <Monthly every\_x\_months="1" day\_of\_month="15" time="01:10"

on\_connect="AFTER\_MISS"/>

 <Check id="Rule\_D"/>

 <Check id="Rule\_E"/>

 <!-- Check rules D and E monthly on the 15th day of the month at 1:10 AM. Check

immediately on connect if assessment is missed -->

 </Period>

 <!-- Rule F is not listed despite being in the file. It would be checked during

the baseline assessment, but never reassessed -->

 <Period>

 <Weekly every\_x\_weeks="2" day\_of\_week="2" time="12:15" on\_connect="AFTER\_MISS"/>

 <Check id="Rule\_G"/>

 <!-- Check rule G bi-weekly on Mondays (day 2) at 12:15 PM. Check immediately on

connect if assessment is missed -->

 </Period>

 <Period>

 <Daily every\_x\_days="3" time="01:30" on\_connect="TRUE"/>

 <Check id="Rule\_H"/>

 <Check id="Rule\_I"/>

 <!-- Check rules H and I every 3 days at 1:30 AM. Also check immediately on

connect (regardless of whether a check was missed). -->

 </Period>

 </CheckingFile>

 <CheckingFile id="oval.org.example\_definitions">

 <Period>

 <Daily every\_x\_days="1" time="1:30" on\_connect="TRUE"/>

 <Check id="Def\_Z"/>

 <Check id="Def\_Y"/>

 <Check id="Def\_X"/>

 <!-- Check OVAL definitions X, Y, and Z daily at 1:30. Also check immediately on

connect (regardless of whether a check was missed). -->

 </Period>

 <Period>

 <Hourly every\_x\_hours="2" on\_connect="FALSE"/>

 <Check id="Def\_W"/>

 <!-- Check definition W every 2 hours (at the start of the hour). Do not check

on connection, even if checks were missed. -->

 </Period>

 <Period>

 <Minutely every\_x\_minutes="20" on\_connect="FALSE"/>

 <Check id="Def\_V"/>

 <!-- Check definition V every 20 minutes. Do not check on connection, even if

checks were missed -->

 </Period>

 <Period>

 <OnChange/>

 <Check id="Def\_U"/>

 <Check id="Def\_T"/>

 <Check id="Def\_S"/>

 <!-- Monitor definitions S, T, and U in real time (or as close thereto as

practical) -->

 </Period>

 </CheckingFile>

</MonitoringOverlay>

## JSON

This is intended to be equivalent to the preceding XML.

{

 "monitoring\_overlay\_id" : "example.org-TestMonitoringOverlayFile",

 "schema\_version" : "MonitoringOverlayJSONSchema0.1.schema.json",

 "file\_version" : "1",

 "checking\_files" : [

 {

 "file\_id" : "xccdf\_org.example\_benchmark\_example1",

 "periods" : [

 {

 "on\_connect" : null,

 "check\_ids" : [ "Rule\_A", "Rule\_B" ]

 },

 {

 "monthly" : {

 "every\_x\_months" : 3,

 "day\_of\_month" : 31,

 "time" : "01:00",

 "on\_connect" : "AFTER\_MISS"

 },

 "check\_ids" : ["Rule\_C"]

 },

 {

 "monthly" : {

 "every\_x\_months" : 1,

 "day\_of\_month" : 15,

 "time" : "01:20",

 "on\_connect" : "AFTER\_MISS"

 },

 "check\_ids" : ["Rule\_D", "Rule\_E"]

 },

 {

 "weekly" : {

 "every\_x\_weeks" : 2,

 "day\_of\_week" : 2,

 "time" : "12:15",

 "on\_connect" : "AFTER\_MISS"

 },

 "check\_ids" : ["Rule\_G"]

 },

 {

 "daily" : {

 "every\_x\_days" : 3,

 "time" : "01:30",

 "on\_connect" : "TRUE"

 },

 "check\_ids" : ["Rule\_H","Rule\_I"]

 }

 ]

 },

 {

 "file\_id" : "oval.org.example\_definitions",

 "periods" : [

 {

 "daily" : {

 "every\_x\_days" : 1,

 "time" : "1:30",

 "on\_connect" : "TRUE"

 },

 "check\_ids" : ["Def\_Z", "Def\_Y", "Def\_X"]

 },

 {

 "hourly" : {

 "every\_x\_hours" : 2,

 "on\_connect" : "FALSE"

 },

 "check\_ids" : ["Def\_W"]

 },

 {

 "minutely" : {

 "every\_x\_minutes" : 20,

 "on\_connect" : "FALSE"

 },

 "check\_ids" : ["Def\_V"]

 },

 {

 "on\_change" : null,

 "check\_ids" : ["Def\_U", "Def\_T", "Def\_S"]

 }

 ]

 }

 ]

}