XCCDF Developer Day Workshop

February 23, 2010

Introduction

This document contains a summary of the discussions and consensus decisions from the February 23 XCCDF Developer Day Workshop held at the National Institute of Standards and Technology. This workshop covered more than a dozen open issues in XCCDF on a variety of topics. The slides and source material for the XCCDF workshop, as well as material from other workshops from the week's events, can be found at http://measurablesecurity.mitre.org/participation/devdays.html.

The rest of this document contains a summary of each of the topics discussed in the workshop. The document concludes with a summary of points of consensus as well as an outline of follow-on activities. Topic titles correspond to the titles from the *Discussion Topics and Fix Proposals* document sent to the mailing list on Monday, February 15. An archived copy of this document can be retrieved at http://n2.nabble.com/attachment/4576272/0/Discussion%20Topics%20and%20Fix%20Proposals.pdf or from the aforementioned workshop web page. Readers may find the *Discussion Topics and Fix Proposals* document useful for its presentation of background information since the current document will only present a minimal background of discussion topics.

Contents

Using CVSS/CCSS in Scoring

Background: It was proposed that XCCDF adopt new scoring models that utilized CVSS and CCSS metrics in score computation.

Discussion Highlights:
It was noted that a new scoring model is not strictly needed – one could always compute a score for a CVSS vector ahead of time and use that in the weight of a Rule. This proposal is more about supporting the automated generation of "weights" from a provided scoring vector. It was also noted that, if only a weight is provided then a reader cannot see how the author arrived at this weight. However, if a weight is derived from a CVSS vector, the reader can see the reasons for a particular weight. As such, the CVSS vector does provide useful documentation for explaining weights.

It was noted that, unlike a CVSS vector, CCSS vectors cannot be computed ahead of time. This is because the selections in a CCSS vector may vary depending on the actual settings on the system rather than the nature of the control itself. As such, including a CCSS vector in a Rule's impact-metric field is not realistic. It was noted that it would be useful if XCCDF had the ability to record target system information that might be used to create a CCSS vector. Discussions of this particular feature fell into the intended scope of the check-import discussion and so were deferred until then.

A poll indicated that there was no significant interest in creating a new XCCDF scoring model.

Conclusions:

- There was not significant interest in a new scoring metric based on CVSS or CCSS.

Using CVSS temporal and environmental vectors and CCSS vectors in the impact-metric field

Background: It was also proposed that the impact-metric field be expanded from only allowing CVSS base vectors to including temporal and environmental vectors for both CVSS and CCSS.

Discussion Highlights:
Continuing from the prior discussion, it was noted that CCSS vectors could not usefully be provided by authors in the impact-metric field. Discussion turned to whether CVSS temporal and environmental vectors would be worthwhile to add to impact-metric. It was noted that both scores are, by nature, dependent on temporal and environmental factors and thus less broadly applicable than the base vector. It was also noted that temporal CVSS scores might be provided by a third party other than the document author and, as such, any inclusion of this information would also need to identify the source of the information so that provenance could be traced. Finally, it was noted that within the CVSS community the temporal and environmental vectors are considered experimental and that only the base vector is used in general practice. Given these factors, it was concluded that there was little call to include temporal or environmental vectors in impact-metric.

The question was raised as to at what stage of processing a CVSS (or CCSS) vector might be useful. It was broadly agreed that this information was more relevant when processing results, possibly during post-processing actions such as result aggregation, rather than within the original body of the Benchmark. It was further noted that the impact-metric field is not generally used. Based on these observations, a poll was taken and the decision made to deprecate the impact-metric field within Rules.

Given that CVSS/CCSS metrics were felt to provide some useful information for result processing, the question was raised as whether CVSS should be included in XCCDF results. The general consensus was that XCCDF interpreters shouldn't be burdened with filling CVSS or CCSS vectors and that this should be left to post-processing tools.

Conclusions:

- Deprecate the impact-metric field.

Update CPE Version

Background: The current reference to the CPE schema from the XCCDF specification and schema is a couple versions out of date. Multiple ways to remedy this were put forward.

Discussion Highlights:
It was noted that the proposal to support all versions of CPE within a major release (past and future) is complicated by the challenges of forward compatibility. Specifically, the evolution of CPE within a given minor release could create new implicit support requirements among XCCDF implementers without a corresponding explicit change in XCCDF version. (For example, consider an interpreter written for XCCDF 1.2 when CPE 2.3 was the latest version where XCCDF 1.2 supports any CPE 2.x. If CPE is revised to 2.4, documents that utilize CPE 2.4 would still be considered to use XCCDF 1.2. However, the aforementioned XCCDF 1.2 interpreter might not be able to correctly process this new content.) For this reason, it was felt that an upper limit on the CPE version should be explicitly stated in the specification requirements.

Regarding the proposal to support annotation of platform information with explicit namespace and version identifiers, thus allowing authors to select arbitrary platform identifiers, no one felt there was any need for a platform identification structure of this complexity. This left the third from the initial proposals – to explicitly update the XCCDF specification to support the latest version of CPE as of the date of publication of the revised XCCDF specification as well as previous CPE versions within that major release. (This should be the version of CPE included in SCAP 1.2.)

In response to the observation that the CPE schema namespace only changed on major versions, it was suggested that CPE (and other SCAP namespaces) change on minor versions to make it clear which version of CPE was intended by a namespace reference. Others argued this would break backwards compatibility within a major version but proponents disagreed that this was true. MITRE had recently held some internal discussions on this topic and agreed to publish a summary of these conversations. The topic was tabled to give participants some time to think about the issue.

Conclusions:

- Update XCCDF to use the latest (as of the release of SCAP 1.2) version of CPE.
- There is an SCAP-wide need for standardized namespace conventions. There should be follow-on discussions on this topic.

Explicit mapping of check results to XCCDF results

Background: Currently, mappings of check results to XCCDF results exist by convention or through higher-level documents, such as NIST IR 800-126. Community members suggested that XCCDF checks be able to explicitly perform these mappings to override convention (or handle any lack of convention).

Discussion Highlights:
It was noted that there are some cases where the default result mappings are not desired. For example, OVAL inventory checks always map to pass if the software is found however sometimes a policy statement seeks to preclude the presence of a piece of software.

It was noted that, if there was consistency in result values across the SCAP standards, this problem would be much simpler. It was suggested that SCAP produce a standard list of results for all its component specifications. Others were concerned that different check languages had slightly different perspectives which needed to be reflected in their results and that standardization could be counterproductive. It was agreed that proposal was out of scope for the current discussion but that it should be raised in the context of a cross-standard SCAP discussion.

It was suggested that the correct XCCDF result could be ensured simply by making the check language return the appropriate value. For example, if one wanted to ensure that detection of a particular piece of software result cause a Rule to fail, instead of calling the OVAL definition an inventory definition, call it a vulnerability definition so that detection maps to XCCDF Rule failure. However, it was noted that this requires the OVAL definition to encapsulate the purpose to which it is being put. As such, OVAL definitions would no longer be concerned with a simple system state discovery, but would reflect a particular system policy based on that discovery. While detection of malware might easily be viewed as always bad, sometimes software switches from good to bad when it becomes obsolete, and the point in time at which that switch occurs will vary between organizations. For example, a particular version of a web browser might be an acceptable member of a system inventory until it becomes obsolete at which point its presence (and the same checks that determine that presence) would then need to indicate a policy violation.

The discussion made it clear that virtually all use cases involved a need to switch a pass to a fail or vice versa. It was noted that a complex-check element already has the ability to "negate" a result to toggle pass to fail or fail to pass. It was suggested, and the audience agreed, that adding a "negate" field to a check element would solve most existing problems relating to result mapping.

Conclusions:

- Add an optional "negate" attribute to checks that inverts the standard pass-fail mapping.
- There in interest in standardizing the result values across the SCAP standards. There should be follow-on discussions on this topic.

Segregated or Mixed Extensions to Value

Background: Concerns were raised that a prior proposal to support more complicated structures in XCCDF Values would be burdensome if the old (singleton) value types could be intermingled with the new complex types arbitrarily.

Discussion Highlights:
The community was queried whether mixing values of differing complexity within a Value presented any additional burdens. It was noted that implementers need to handle both kinds of value anyway so there was no additional burden in needing to handle both at the same time. As a result, a mixed format was agreed upon as having a simpler schema and providing more power.

Conclusions:

- The community expressed no concerns about the complexity of mingling data types.

Content Categorization

Background: A request was made to support the encapsulating of many-to-many relationships within XCCDF Items.

Discussion Highlights:
It was noted that the ways of categorizing information would probably vary between communities of interest. As such, any given categorization of content might not be broadly applicable. It was noted content categorization had been attempted in the initial releases of FDCC content in the mapping of Rules to NIST IR 800-53 controls, but that lacking a field to support this type of categorization, the document had kludged this information in the "requires" field. However, NIST indicated it was acting to remove these control references so this use case was no longer as compelling.

It was suggested that categorization could simply be included as metadata within a Rule. There was a general agreement that doing this was probably a better idea than creating a new dedicated field for category information. Given that there was a later discussion planned on the subject of metadata, the issue of including metadata in Rules was deferred until the later session.

Conclusions:

- Do not create a new field for content categorization. Instead, authors can use metadata fields for this. Metadata field content will be discussed later.

Clarify check-import behavior

Background: Use of the check-import field in XCCDF Rule checks is currently under-defined. It was proposed to establish and document the proper use of this field.

Discussion Highlights:
Two possible intents for this field were identifier: archiving discovered state information and pulling in values for the purpose of exporting them back out as part of a check. Multiple audience members noted that, whatever changes were accepted, this field cannot force multiple interactions with a target system. In other words, it must remain possible to send a single set of instructions to the target and get a single set of results back.

For the use case of archiving discovered information, it was noted that, while we could hold all collected OVAL data in the TestResults, this would result in massive bloating of the result file. People expressed an interest in getting targeted data elements rather than a flood of mostly unwanted information. It was noted that OVAL Reporting might also be a solution to this. However, use of OVAL Reporting would only be a solution for OVAL checks.

For archiving, a concern was raised that, without context, individual values would be meaningless. For example, "is a given integer a timeout measured in minutes or seconds" or "does 1 mean enabled or disabled?" Having a translation method was suggested. It was noted that OVAL Reporting is intended to do this. A concern was raised about integrating evaluation capabilities into XCCDF and it was suggested that we continue to keep such translation functionality separate from XCCDF.

An informal poll indicated that there was also interest in being able to gather data for use in populating exports, but only if this did not require multiple interactions with the target. It was noted that by scoping imports to only be applicable within a Rule, that multiple interactions would become unnecessary. It was noted that OVAL can already use queries to populate variables used in subsequent queries. As such, allowing XCCDF to map imports to exports would not create an entirely new capability, but it would add the ability to control the source of these population queries from XCCDF.

It was suggested that a query control language could be developed to do this sort of population. For example, the control language could initiate checks under certain conditions and with certain discovered values. It was noted that a previous proposal to do exactly this, called the automation schema, got little interest. It was also noted that, for the current discussion, we aren't looking for a complex management of checks because we want to avoid multiple interactions.

The question was raised as to how to specifically identify the information we wanted. Suggestions included:

- only use variables as targets since variables are already targeted values
- have an artifact stylesheet for XML-based languages that filtered XML content to get the desired value
- define an expression language to tell XCCDF how to get result

Concern was also raised as to how to handle multiply instantiated targets. We discussed returning full OVAL Items (from the system-characteristics schema) for archiving results but multiple parties felt that this was not sufficiently precise and could lead to bloated result files.

The discussion was getting bogged down in the technical details. We concluded there is some interest in both the archiving and variable population capabilities but that some specific proposals as to how to get the relevant information in a useful fashion should be drafted and sent to the community for review.

Conclusions:

- There is interest in using check-import for both discovery reporting and for feeding in to subsequent export statements. However, there are technical complexities with how to identify specific pieces of data from the check system. Additional proposals for this detail will be developed and submitted for discussion.

Open discussion

Background: This part of the workshop gave participants a chance to raise issues that were not otherwise on the agenda.

Discussion Highlights:

The discussion started with a question raised earlier in the day as to whether there were aspects of XCCDF that no one was using. No such issues were explicitly noted. One participant commented that there seemed to be some duplication across standards. The "type" field in XCCDF Values was given as an example of this since variables are also typed within OVAL and it is the OVAL data type that is ultimately used. It was explained that the Value type field actually exists to filter values given to an XCCDF interpreter during tailoring and, as such, are not actually duplicative of OVAL capabilities. However, it was agreed that the broader point is something the community should be watchful of.

It was noted that many OVAL definitions reference platform tests as part of their checks. Since these platform tests are then performed with every check, this causes redundant checking. It was suggested that such platform checks not be included in OVAL checks. It was noted that this behavior is not required by XCCDF or OVAL and that it is done at the discretion of the author. As such, concerns on this topic should be raised with the content authors.

It was asked if there was interest in being able to indicate periodicity with which assessments should be run. For example, noting that certain recommendations should be verified weekly while others could be verified monthly. It was suggested that this fell under the broad topic of a security automation control language, which is something that NIST has been working on. NIST indicated that this use case could be supported by their control language work and noted that development of this language is ongoing.

Conclusions:

- Although no specific features of XCCDF were noted in the workshop as being broadly unused, community members are encouraged to post features that are seldom used, are redundant with other standards, or which have an inconsistent use in practice.
- OVAL Reporting addresses multiple concerns about the volume of OVAL output files. Specific proposals should be available for community review shortly.
- There is significant interest in functionality to control XCCDF use and result handling. This overlaps with the proposed Control Language suite of protocols currently under development by NIST. NIST will continue to work to advance these capabilities.

Open the metadata field to additional types of metadata

Background: A proposal was made to allow additional types of content in the Benchmark metadata field.

Discussion Highlights:
The audience was asked if there were any objections to opening up the metadata field to allow any type of XML content instead of just Dublin Core or SCCF (still currently in draft). One participant noted that, as the author of an editing tool, they currently know exactly what to expect in the metadata field. He commented that whenever a field is completely unconstrained it makes things difficult for editors.

Other community members advocated for allowing unconstrained content. It was noted that without such fields, authors end up attempt to place data in inappropriate locations. By providing locations that allow arbitrary content, vendors can innovate without disrupting the normal processing of other fields. Towards that end, it was suggested that, in addition to adding an unconstrained metadata field in Rules, Groups, etc., that the sequences of these elements end with another any tag to allow additional fields to be appended. However, others felt that it might be better to limit the unconstrained content to a specific subfield of these items.

A concern was raised that, by completely opening the contents of the field, it could be abused in a way that broke standardization. For example, vendors could encode information in the metadata that modified how Benchmarks were processed, thus producing different results between tools. It was suggested that the standard include specific language noting that metadata was not allowed to take the place of existing XCCDF fields or affect the core processing procedures, although it would still be allowed to contain instructions that could provide additional value to those procedures. It was noted that various communities of interest could use schema restriction to eliminate unwanted or unrecognized metadata tags and thus normalize behavior in that way.

A concern was raised that a vendor might add a field to the metadata element but that a later version of the standard might add a specific separate field for that purpose. An example of this happening in OVAL was cited. The OVAL team, however, argued that this was a positive thing in that it allowed the schema to be flexible until the specification was able to catch up with practice. As long as the vendor switched to using the new canonical field when it became available, there would be no conflict.

The audience was polled and voted to open metadata the field in Benchmark as well as add a new open metadata field to Items.

A suggestion was made that the metadata support annotation to indicate information such as the name, version, and location for the utilized metadata schema. Such information would assist in processing of the metadata. For example, if a tool is able to utilize a specific metadata standard, the annotations would allow the tool to quickly identify information that used that standard. MITRE agreed to draft a proposal for metadata annotation based on example information to be provided by Lt. Col. Joseph Wolfkiel.

Conclusions:

- Allow arbitrary metadata in Benchmark metadata fields.
- Allow the inclusion of arbitrary metadata in XCCDF Items in a new metadata field. Conventions would be established to prohibit changes or control of the XCCDF processing models via information encapsulated in the metadata field. Vendors could use this field to support added functionality beyond that prescribed by the XCCDF specification.
- A proposal will be created for labeling metadata to allow for better automated processing.

Adding Dublin Core to status entries

Background: A proposal was made to allow status fields to contain Dublin Core information to support more expressive status statements.

Discussion Highlights:
After a short discussion it was agreed that there was interest in expanding the expressiveness of the status field. After it was noted that since the old status field was a simple type it could not cleanly be updated to hold complex content, a new DCStatus field was proposed that would be of the complex type needed to hold the Dublin Core elements.

Conclusions:

- A new DCstatus field for Dublin Core information will be added wherever a status field exists.

Clarify use of selected vs. role="unchecked" vs. UNCHECKED rule result

Background: There is some ambiguity surrounding the use of the role field. This discussion sought to clarify its use.

Discussion Highlights:
It was explained that the purpose of the role field was to disable Rules through a mechanism other than tailoring or explicit selection. This was done primarily to support draft content. It was noted that the field is virtually unused and that the same functionality can be achieved in other ways. The only change to the specification needed for identical behavior would be to modify the flat-unweighted scoring model to keep a 0-weighted Rule from contributing to a final score. The group voted to deprecate this field and tweak the flat-unweighted scoring model as described.

Conclusions:

- The role field will be deprecated.
- The "flat unweighted" scoring model will be modified to preserve 0-weights for Rules.

Clarify the processing model for group selection and requires capabilities

Background: Some unintuitive results were noted in the current processing model for the processing of require and conflict fields. This discussion sought to address this.

Discussion Highlights:
It was noted that, under the current requires/conflicts processing model, some edge cases will result in unintuitive procedures. Multiple members of the audience suggested that, if these behaviors were not intended by the author, testing of the benchmark should catch them. The audience concluded that, as long as the processing model always resulted in the same result, that there was little harm from these edge cases.

It was observed that the processing model of requires/conflicts is set out in the specification, but it was requested that this be elaborated, possibly with examples, to ensure uniform understanding.

Conclusions:

- The audience expressed no concern with the unintuitive results as long as all tools behaved consistently in processing.
- The documentation will be enhanced to provide clearer processing instructions to ensure consistency of tool processing.

Check-refs without names and the "multiple" property

Background: It was noted that there is no documentation of expected behavior of checks-content-refs that lack a name attribute. It was also noted that there is confusion regarding the use of the "multiple" property of Rules. This discussion sought to clarify both these functions.

Discussion Highlights:
It was noted that existing content exists that uses the assumption that if the name is absent from a check-content-ref then the interpreter should use all checks in the targeted file. There was no objection to continuing this practice.

It was noted that current practice is "all-or-nothing" if a file with multiple checks is referenced. I.e. if one check in the referenced file fails, the entire Rule fails. Multiple parties noted that this is not helpful because it doesn't provide any information as to what needs to be done to remediate a failure.

It was suggested that the "multiple" property of checks be used to allow splitting of multiple target checks, such as happens when a check-content-ref references a file without naming a specific check in that file. This was countered by the observation that the intent of the multiple property was to split result reporting for each instance of a target (e.g. for each record in a database) and this should be maintained. It was noted that OVAL cannot support such splitting because OVAL definitions find all instances and compute a result over the whole set. However, others noted that the ability to split up results by instances would be useful. The group agreed that it would be better to add this sort of new functionality to OVAL than remove it from XCCDF.

Given the above suggestion, a proposal was made to create a new attribute, tentatively named "multi-check", to split out results when multiple checks are referenced. This attribute would only apply in the case of check-content-refs that did not name a specific check in a given file and where the file contained multiple checks. If this was the scenario and multi-check was true, then individual results would be recorded for each of the component checks in the referenced file. Otherwise, XCCDF would continue to report a single result as it does in the current version of XCCDF.

It was suggested that rule-result elements be marked with the same Rule ID to denote results arising from a single Rule with multi-check or multiple set to true and that other fields of a rule-result element would be used to distinguish component results. It was noted that there is already an "instance" field in rule-results to differentiate results in the case of multiple being set to true.

Conclusions:

- Update documentation to indicate that a nameless check-content-ref should execute all checks in the referenced file and AND their results together.
- Add a new field, tentatively named multi-check, that, if true and a nameless check-content-ref is used, each check in the targeted file should be reported separately in the XCCDF results.

- Update documentation of the "multiple" property to explicitly note that that setting it to true should cause the check system to assess and report each target instance separately and have these each reported separately in the XCCDF results.

Allow XCCDF check-content-ref statements to refer to other XCCDF documents

Background: A proposal was made to allow XCCDF check-content-ref statements to refer to XCCDF Rules or documents so that one Benchmark could be called from another.

Discussion Highlights:
A question was raised as to the order in which check-content-refs are executed within a single check. It was noted that the specification dictates that the check-content-ref statements be processed in the order in which they appear within the XML document, stopping the first time the target can be resolved. By request, this will be further clarified in the specification. It was suggested that, if there is a fall-through (that is, if one check-content-ref fails to resolve and the interpreter must go on to the next reference) that a warning should be issued. It was countered that, if the author didn't feel the fall-through was correct, it shouldn't be present. It was suggested that the message field in rule-results could be used to record such warnings in a non-disruptive manner. It was noted that failing to resolve all check-content-refs is undefined and that a behavior for such an eventuality should be dictated.

It was noted that allowing XCCDF-XCCDF calls could allow FDCC to create a patch XCCDF that allowed individual patches to be scored separately. The group discussed creating a control structure for passing instructions to checking engines (such as those needed to tailor XCCDF). It was noted that a proposed control structure for XCCDF was basically an external profile and that this might argue for re-opening the proposal to support external profiles. As an alternative to defining a new XML element to hold control instructions, it was suggested the control structure could be passed using existing mechanisms by exporting a special variable. It was noted that checking engines needed to create a stub variable to support this. It was also observed that the proposed control structure was extremely similar to a full XCCDF Profile and that, if this was going to be the case, it probably made sense just to reuse the current Profile structure instead of inventing something new.

Members of the community then asked how necessary calling external XCCDF documents would be. A few parties noted that it would be useful in some cases, but no one seemed to view this capability as being necessary. In light of this observation, it was agreed that the issue should be raised to the broader mailing list and, if significant support was not found there, the proposal would simply be closed.

Conclusions:

- Update the documentation to make it clear the order in which check contents should be evaluated.
- Support for the capability in general was only marginal. We will seek input from the mailing list. Barring the appearance of significant support, the proposal will be rejected.

Clarify the order of operation of Profile selectors

Background: It was discovered that a previous decision to revise processing of Profiles selectors under inheritance could lead to unexpected behaviors with regard to allowing an extending Profile to change the behaviors of the extended Profile.

Discussion Highlights:
It was noted that the recently agreed-upon change in selector processing under extension could lead to problems in some cases. As such, the issue was being reopened. It was observed that eliminating the prohibition against multiple instances of the same type of selector naming the same idref would allow an extending Profile to override the behavior of its parent without resulting in the problems observed in the original proposal. There was no objection to removing this restriction so the prior proposal was retracted and selectors will continue to be appended under extension, but now duplication is permitted.

It was asked that examples of an extending Profile overriding a parent Profile be provided to help clarify expectations.

Conclusions:

- The previous revision of Profile selector processing will be retracted. Selectors will continue to be extended under the "append" processing model.
- All restrictions against duplication of selectors in profiles will be removed. Items may now be modified by selectors any number of times.
- Update the documentation to clarify processing of selectors, especially in cases of overlap.

Clarify the concept of "default values" in Values

Background: There is no description of default behavior in Values (i.e. the state of a Value absent tailoring actions). This discussion sought to create a default behavior for Values.

Discussion Highlights:
An initial suggestion was made to treat the first instance of a tailorable element in the order in which these elements appeared in the XML as the default. It was noted that there is already a prededent for selecting a default value in Rule processing. Specifically, in the case of multiple check elements within a Rule, checks with empty or absent selectors are used in the absence of any tailoring actions. It was suggested that, for consistency, this precedent be followed within Values. The community agreed.

It was noted that the value field of Values is slightly more complicated because at least one value field must be present. This would not be the case if none of the value fields had an empty selector. It was agreed that, in the absence of a value with an empty selector, the default value field would be the first field in the order in which the fields appeared in the Value's XML.

Conclusions:

- The default value field in a Value will be the one with an empty or absent selector. If there is no value field with an empty or absent selector, the first value field in top-down processing of the XML will be the default field.
- For all other selectable Value fields, the default activity will be to ignore all fields without an empty or absent selector.

Local vs. remote imports

Background: XCCDF imports other schemas using local, relative file paths. It was suggested that these schemas should be imported from canonical remote sources.

Discussion Highlights:
Some participants noted that they operate in a closed environment without network access and that pulling schemas from a remote location was not an option. Others noted that this situation was not unique to the Department of Defense and that some commercial enclaves also frequently operate without network access, especially those that are more security sensitive.

It was suggested that a common response is to use a catalog resolver when processing the XCCDF schema. Such a resolver would look at the namespace to resolve and then uses its own logic to determine where to find the actual schema file without using the schemaLocation attribute. The resolver logic could access the schema remotely or from an internal archive based on network availability and user preferences. In this scenario, the schemaLocation becomes a canonical reference rather than direct input into processing behavior. It was noted that many tools already use resolvers and have internal archives of commonly used files. It was also suggested that the use of local files in security automation standards was resulting in branching of the standards since each standard carried its own "suite" of utilized standards along with it.

The suggestion of using resolvers was countered by observations that some tools do not have catalog resolvers and would therefore be unable to work with only a remote URI while a local schemaLocation would always work regardless of the capabilities of the tool interpreting the schema. A suggestion was made to use an HTTP proxy that could redirect a reference a remote schema to an internal archive, but it was noted that at this required the user to make infrastructure changes to support the new remote reference and that this was overly burdensome.

As it was clear that the group was not heading towards consensus, it was agreed that the issue would be raised to the list.

Conclusions:

- The community was evenly split between those who wished to switch to remote importing and those who wished to continue to use local imports. No consensus was achieved. The issue will be raised on the mailing list to see if the deadlock can be broken.

Add an enumeration to classify types of notice elements

Background: It was suggested that a field be added to notice elements so authors could indicate the specific use of a particular notice field for display purposes.

Discussion Highlights:
There was a suggestion that this sort of content annotation might be useful in other fields, possibly as inline annotations in descriptions, warnings, etc. It was agreed to investigate this as a follow-on activity but to limit the current conversation to the notice element.

Multiple parties indicated this type of annotation would be useful and, since there were no objections, it was agreed that this should be added into the specification.

Conclusions:

- A field will be added to notice elements to indicate the general type of the content.
- A further proposal will be developed for adding similar capabilities to other fields in XCCDF, such as Item warning fields.

Summary

The following changes will be made to XCCDF:

- Deprecate the impact-metric field.
- Update XCCDF to use the latest (as of the release of SCAP 1.2) version of CPE.
- An optional "negate" attribute will be added to the check element.
- The Benchmark metadata field will be opened to any content.
- A new metadata field will be created in Items.
- A new DCstatus field for Dublin Core information will be added wherever a status field exists.
- The role field will be deprecated.
- The "flat unweighted" scoring model will be modified to preserve 0-weights for Rules.
- Updated documentation to provide clearer processing instructions for requires & conflicts fields.
- Update documentation to indicate expected behavior of a nameless check-content-ref.
- Add a new multi-check field to allow separate reporting of individual checks when a check-content-ref refers to a file of multiple checks without naming a check.
- Update the documentation of the "multiple" property.
- Update the documentation to make it clear the order in which check contents should be evaluated.
- The previous revision of Profile selector processing will be retracted.
- All restrictions against duplication of selectors in profiles will be removed.
- Update the documentation to clarify processing of selectors, especially in cases of overlap.
- Update documentation to provide default behavior of Value fields.
- Add a type field to notice elements.

The XCCDF development team will perform the following follow-on activities:

- Develop a proposal for how the check-import field could identify specific data elements in a check language for import.
- Develop a proposal for labeling metadata in XCCDF metadata fields.
- Query the broader XCCDF community regarding interest in allowing XCCDF check-content-ref statements to refer to other XCCDF documents.
- Query the broader XCCDF community to see if consensus can be achieved regarding local vs. remote schema import statements.
- Develop a proposal for adding capabilities for categorizing content in various XCCDF text fields.
- Hold a developer meeting regarding stand-alone TestResults. This was the only discussion topic that we could not touch on because of time.

In addition, the following activities were deemed important for the broader SCAP community and will be pursued by the following parties:

- Hold a discussion to standardize namespace conventions within SCAP – MITRE
- Look into an SCAP-wide convention on result values – NIST
- Present a summary of the proposed OVAL Reporting (formerly OCRL) language – MITRE OVAL team

- Continue development of a control language protocol compatible with SCAP – NIST

Thank you to everyone who provided comments ahead of the workshop and to everyone who was able to participate in the workshop.