|  |
| --- |
|  |

|  |  |
| --- | --- |
| CHANGE REQUEST | |
| Meeting ID:\* | SDS 42 |
| Source:\* | Bob Flynn, Convida Wireless , Bob.Flynn@convidawireless.com |
| Date:\* | 2020-01-29 |
| Reason for Change/s:\* | Non-Originator context ACPS |
| CR against: Release\* | Rel-4 |
| CR against: WI\* | Active < WI-0077>  MNT maintenance / < Work Item number(optional)>  Is this a mirror CR? Yes  No  mirror CR number: (Note to Rapporteur - use latest agreed revision)  STE Small Technical Enhancements / < Work Item number (optional)>  Only ONE of the above shall be ticked |
| CR against: TS/TR\* | TS-0001 v4.1.0 |
| Clauses \* | 5.4.3 |
| Type of change: \* | Editorial change  Bug Fix or Correction  Change to existing feature or functionality  New feature or functionality  Only ONE of the above shall be ticked |
| Other TS/TR(s) impacted | None |
| Post Freeze checking:\* | This CR contains only essential changes and corrections? YES  NO  This CR may break backwards compatibility with the last approved version of the TS? YES  NO |
| Template Version: January 2019 (do not modify) | |

**oneM2M Notice**

The document to which this cover statement is attached is submitted to oneM2M. Participation in, or attendance at, any activity of oneM2M, constitutes acceptance of and agreement to be bound by terms of the Working Procedures and the Partnership Agreement, including the Intellectual Property Rights (IPR) Principles Governing oneM2M Work found in Annex 1 of the Partnership Agreement.

GUIDELINES for Change Requests:

Provide an informative introduction containing the problem(s) being solved, and a summary list of proposals.

Each CR should contain changes related to only one particular issue/problem.

In case of a correction, and the change apply to previous releases, a separate “mirror CR” should be posted at the same time of this CR

Mirror CR: applies only when the text, including clause numbering are exactly the same.

Companion CR: applies when the change means the same but the baselines differ in some way (e.g. clause number).

Follow the principle of completeness, where all changes related to the issue or problem within a deliverable are simultaneously proposed to be made E.g. A change impacting 5 tables should not only include a proposal to change only 3 tables. Includes any changes to references, definitions, and acronyms in the same deliverable.

Follow the drafting rules.

All pictures must be editable.

Check spelling and grammar to the extent practicable.

Use Change bars for modifications.

The change should include the current and surrounding clauses to clearly show where a change is located and to provide technical context of the proposed change. Additions of complete clauses need not show surrounding clauses as long as the proposed clause number clearly shows where the new clause is proposed to be located.

Multiple changes in a single CR shall be clearly separated by horizontal lines with embedded text such as, start of change 1, end of change 1, start of new clause, end of new clause.

When subsequent changes are made to content of a CR, then the accepted version should not show changes over changes. The accepted version of the CR should only show changes relative to the baseline approved text.

## Introduction

TR-0050 section 6.3.4 presents a solution for Non-Originator Context.

This contribution incorporates that solution into TS-0001.

Note: there is a tangential change that I discovered while preparing this contribution that I fix in change 1.

*eventCriteria* was used in the description of <action> in early drafts that were changed to *evalCriteria*. Related typo is fixed.

Change 2: Add the *evalCriteria* parameter to the *accessControlContexts* attribute of an <accessControlPolicy>. In a manner similar to other “contexts”, when present the *evalCriteria* SHALL evaluate to **true** for the requested operation to be allowed.

TR-0050 section 6.3.5 presents a solution for ACPs with limited usage configurations.

Change 2: add *accessControlLimit* parameter to the *accessControlContexts* attribute of an <accessControlPolicy>. In a manner similar to other “contexts”, when present the *accessControlLimit* value SHALL be greater than zero for the requested operation to be allowed.

TR-0050 section 6.3.6 presents a solution for improved default behavior for ACPs when resources are created.

Change 3: Adds new parameters to *accessControlObjectDetails* attribute of an <accessControlPolicy>

TR-0050 section 6.3.7 presents a solution for identifying the permissions that an originator has on a target resource.

Change 4: defines a new **Result Content** type **Permission** and it defines a new ***filter Criteria*** type *operations.*

The rcn= Permission returns the permissions that the originator has for the target resource.

The new *filter Criteria operations* supports specifying the specific operations that the originator wants to be present for the resources returned by the request. For example a discovery request where the filter criteria specifies labels=parkingLot and operations = UPDATE

R03 –

Change 3 requires further discussion to address the opinion that it results in “leaking” privilege information ad therefore represents a security threat and bad practice.

I will remove change 3 from this contribution.

That leaves change 1 and change 2 which are completely independent and have been agreed in discussions.

Change 3 and 4 will be addressed in new separate contributions

R02 –

During review we were not able to achieve consensus on the manner that ACP propagation is defined. We agreed that a solution should be explored. We agreed to remove that content from this contribution and explore it further in a separate contribution.

Therefore change 3 is removed

R01 –

For the non-originator contexts, clarify that the evaluation may not be related to the originator of the request.

For ACP Propagation, ensure that this does not create a ACP vulnerability by stating that the level of propagation applies to levels relative to the location of the parent resource of the ACP resource and that duplicated ACPs do not have these propagation parameters set.

For example:

<AE1>

<ACP1> - propagation level set to 1 means that this will apply to children of <AE1>

<ACP2> - propagation level set to 2 means that this will apply to grand-children of <AE1>

<container1> - <ACP1> and <ACP2> are applied to *acpids* (only if created with no *acpids*)

<container2> - <ACP2> is applied to *acpids* (only if created with no *acpids*)

<container3> - nothing added to *acpids*

-------------------------------------------------- Start of Change 1--------------------------------------------------

### Resource Type *action*

The <*action*> resource enables the system to specify which actions or operations to be performed on system resources (e.g. devices or services) based on monitored events. The events may include state changes of subject resources, requests from applications, etc. Once the event occurs, and if specified <dependency> conditions are met, the Hosting CSE sends a primitive defined by <action> resource attributes.

These child resources and attributes provide information about:

* The subject resource, which is the resource monitored to determine if a primary event occurs. Action is to be triggered if the primary event is the change of the state of the subject resource .
* The object resource, which is the resource which is the target of the triggered action
* The input resource, which is a resource which may provide input parameters for the action. Alternatively, an input value may be provided.
* Conditions to be monitored in order to determine if the primary event occurs
* Dependencies to be evaluated in order to determine if the action is to be conditionally performed.
* Parameters providing priorities of the action and priorities of the dependencies.

The *<action>* resource shall contain the child resources specified in table 9.6.61-1.

Table 9.6.61-1: Child resources of *<action>* resource

| Child Resources of *<action>* | Child Resource Type | Multiplicity | Description | *<actionAnnc>* Child Resource Types |
| --- | --- | --- | --- | --- |
| *[variable]* | *<dependency>* | 0..n | See clause 9.6.62 | *<dependencyAnnc>* |
| *[variable]* | *<subscription>* | 0..n | See clause 9.6.8 | *<subscription>* |

The *<action>* resource shall contain the attributes specified in table 9.6.61-2.

Table 9.6.61-2: Attributes of *<action>* resource

| Attributes of  *<action>* | Multiplicity | RW/  RO/  WO | Description | *<actionAnnc>* Attributes |
| --- | --- | --- | --- | --- |
| *resourceType* | 1 | RO | See clause 9.6.1.3. | NA |
| *resourceID* | 1 | RO | See clause 9.6.1.3. | NA |
| *resourceName* | 1 | WO | See clause 9.6.1.3. | NA |
| *parentID* | 1 | RO | See clause 9.6.1.3. | NA |
| *expirationTime* | 1 | RW | See clause 9.6.1.3. | MA |
| *accessControlPolicyIDs* | 0..1 (L) | RW | See clause 9.6.1.3. | MA |
| *Labels* | 0..1 (L) | RW | See clause 9.6.1.3. | MA |
| *creationTime* | 1 | RO | See clause 9.6.1.3. | NA |
| *lastModifiedTime* | 1 | RO | See clause 9.6.1.3. | NA |
| *stateTag* | 1 | RO | See clause 9.6.1.3. | OA |
| *announceTo* | 0..1 (L) | RW | See clause 9.6.1.3. | NA |
| *announcedAttribute* | 0..1 (L) | RW | See clause 9.6.1.3. | NA |
| *dynamicAuthorizationConsultationIDs* | 0..1 (L) | RW | See clause 9.6.1.3. | OA |
| *Creator* | 1 | RO | See clause 9.6.1.3. | NA |
| *actionSubjectResource* | 0..1 | RW | The resource that is the subject to be monitored for triggering the evaluation of the *evalCriteria* attribute. The action indicated by the *operation* attribute Action is to be triggered based on the change of the state of the resource indicated the *actionSubject Resource* attribute.The subject can be any resource in the system. If the attribute is missing the subject is the parent resource. | OA |
| *actionPriority* | 0..1 | RW | Attribute indicating the priority of the action when compared with the other actions with the same event expression (*subjectResourceID* and *evalCriteria*), with highest priority indicated by the lowest value. Local policies are applied for equal or unspecified priorities. | OA |
| *subjectResourceID* | 0..1 | RW | The *resourceID* of the resource that is the subject of monitoring for evaluation of the *evalCriteria* attribute. The subject can be any resource in the system. If the attribute is missing the subject is the parent resource of this <action> resource. | OA |
| *evalCriteria* | 0..1 | RW | This attribute provides the conditions determining if the action is to be conditionally triggered to the object resource.  The *evalCriteria* attribute, in conjunction with the *subjectResourceID,* forms the event expression that is being monitored. See further description below and in table 9.6.61-3. | OA |
| *evalMode* | 1 | RW | Attribute provides the macro control mode of the evaluation of the <action> resource. Some values are:  0 = off  1 = once  2 = periodic  3 = continuous | OA |
| *evalControlParam* | 0..1 | RW | When *evalMode*= periodic, *evalControlParam* represents the periodicity.  When *evalMode*= continuous the evaluation is restarted as soon as an action has been triggered. In this case, if *evalControlParam* is specified, it determines the number of times the Hosting CSE shall trigger the event , otherwise it is repeated indefinitely.  When *evalMode* is off or once this attribute can be ignored. | OA |
| *evalPriority* | 0..1 (L) | RW | List of priorities to be applied for the evaluation of the dependencies . This is an ordered list of the *resourceIDs* of the child <dependency> resources. | OA |
| *objectResourceID* | 0..1 | RW | The *resourceID* of the target of the primitive to be sent when the event specified by *evalCriteria* occurs at the subject resource. The target may be specified also as the URI of a resource attribute. | MA |
| *actionPrimitive* | 1 | RW | This attribute stores the entire content of the primitive associated with this action with its parameters (e.g. the op parameter indicating the operation to be performed on the resource identified by objectResourceID), with some parameters being overwritten by the Hosting CSE when the action is triggered as follows:   * the *to* and *resource type* parameters will be derived from *objectResourceID* attribute. * the *content* parameter is derived from the *input* attribute.   When *objectResourceID* or *content* the *input* attributes are empty, the respective parameters are maintained as indicated by *actionPrimitive.* | MA |
| *input* | 0..1 (L) | RW | Input parameters for the operation, used by the Hosting CSE to overwrite the *content* parameter of the *actionPrimitive* before sending the primitive, at the time that the action is triggered. The attribute could be one of the following:   * a string * a resourceID * the URI of a resource attribute.   The Hosting CSE overwrites the c*ontent* parameter of *actionPrimitive* with the following (respectively):   * the given string * the resource representation of the resource with the given *resourceID.* * the value of the given resource attribute | OA |
| *outputResourceID* | 0..1 | RW | The *resourceID* of a resource where the results of the action are stored. | OA |
| *actionResult* | 1 | RW | The result of the action. If *outputResourceID* is set, the Hosting CSE performs a RETRIEVE based on it when the response primitive of the action primitive is received and stores its content. Otherwise, the response primitive of the action primitive is stored in this attribute. | OA |

The conditions represented in the *evalCriteria* attribute determine if the action is to be conditionally triggered.

The *evalCriteria* attribute is comprised of triples (*subject*, *operator*, *threshold*) with the parameters shown and described in table 9.6.61-3.

Table 9.6.61-3: Parameters in *evalCriteria* triple

| Name | Description |
| --- | --- |
| *subject* | This parameter shall indicate an attribute of the *subjectResourceID* resource e.g. “tempContainer.content” |
| *operator* | This parameter is a keyword used to construct the evaluation logic, e.g. ‘equals’, ‘not equals’, ‘greater than’, ‘less than’, ‘greater or equal’, ‘less or equal’, ‘string match’, etc. |
| *threshold* | This parameter provides a value used to evaluate the criteria. The value type shall match the type of the attribute used as *subject*. |

-------------------------------------------------- End of Change 1---------------------------------------------------

-------------------------------------------------- Start of Change 2--------------------------------------------------

#### 9.6.2.2 *accessControlContexts*

The *accessControlContexts* is an optional parameter in an access-control-rule-tuple that contains a list, where each element of the list, when present, represents a context that is permitted to use this access control rule. Each request context is described by a set of parameters, where the types of the parameters can vary within the set. Table 9.6.2.2-1 describes the supported types of parameters in *accessControlContexts*.

The following Originator *accessControlContexts* shall be considered for access control policy check by the CSE.

Table 9.6.2.2-1: Types of Parameters in *accessControlContexts*

| Name | Description |
| --- | --- |
| *accessControlTimeWindow* | Represents a time window constraint which is compared against the time that the request is received at the Hosting CSE. |
| *accessControlLocationRegion* | Represents a location region constraint which is compared against the location of the Originator of the request. |
| *accessControlIpAddress* | Represents an IP address constraint or IP address block constraint which is compared against the IP address of the Originator of the request. |
| *accessControlEvalCriteria* | This attribute provides the conditions determining if the request *operation* is to be allowed. It allows conditional access to the resource based on conditions not contained in the received request. The *accessControlEvalCriteria* parameter consists of a mandatory *subjectResourceID* attribute as defined in table 9.6.61-2 and the *evalCriteria* attribute described in table 9.6.61-3. NOTE: this uses the same definitions that are present in the <action> resource, but does not use the <action> resource. |
| *accessControlLimit* | Represents the number of times that the policy defined in this accessControlRule can allow authorization to the requested resource. This attribute maintains of the number of authorizations granted based on this policy. This value is decremented each time the evaluation grants access to the requested resource. If this value is greater than zero (0) then the request operation is allowed. If the *accessControlLimit* parameter is not present then the request operation is allowed (unlimited access). |

-------------------------------------------------- End of Change 2---------------------------------------------------