CcspTr069Pa

Introduction

Protocol Agents are CCSP components that directly interface to the Cloud. These components facilitate remote administration/management of GW device profiles. The Protocol Agents provides the necessary abstraction to the internal CCSP architecture and internal CCSP components for interacting with the Cloud. The internal CCSP components are not required to be aware of any protocol specific details on the cloud interfaces. As a Protocol Agent could be an SNMP PA or a TR-069 PA..etc.

TR69 ACS uses HTTP/SOAP to communicate with EP or GW. The TR69 protocol agent hides all the protocol specific details and communicates internally using internal namespaces and APIs over the CCSP message bus.

There may be multiple instances of Protocol Agents in a CCSP subsystem.

Protocol Agents perform the following functions:

- Authenticate and establish secure session with their corresponding cloud configuration server during initialization.
- Perform low level protocol handling for downstream traffic.
- Perform low level protocol handling for upstream traffic.
- Routes cloud messages to internal functional components registered for consumption/processing of those namespaces by looking up the Component Registrar.
- "Action" Normalization

    - Internal CCSP components use normalized action to process requests
- Define an XML based Mapping Schema that defines the mapping from external constructs to internal constructs. These constructs will include: Signals errors and routes responses to corresponding cloud adapters

    - External Objects and parameters to Internal Objects and parameters
    - Formatting Conversions
    - Internal Errors to External Errors
- Performs all transactions as an atomic operation.

    - For example, if a transaction involves a "SetParameterList" action of 10 key- value pairs, then the Protocol Agent applies the changes to all of the specified key-value pairs atomically. That is, either all of the value changes are applied together, or none of the changes are applied at all.
    - In cases where a single transaction involves multiple Components, the PA aggregates the response from all the components before sending the results back to PA.
- Manages the order of operations within a transaction and across transactions.

    - The PA serializes all transactions from its cloud interface and only allows one transaction at a time.
    - Generates and maintains Context for asynchronous notifications and transactions. This is explained in more details in the next section.

Architecture

Control Plane architecture abstracts the cloud interfaces for internal CCSP architecture. It makes internal common components like TR-069 PA to be agnostic to those cloud interfaces by using a set of normalized actions.
This component is responsible for all normalized actions listed below. This component does not provide interface for servicing normalized actions, rather it depends on functional components to expose APIs on CCSP message bus.

- GetParameterNames

- GetParameterValues

- SetParameterValues

- SetCommit

- GetParameterAttributes

- SetParameterAttributes

- AddTblRow

- DeleteTlbRow

Data Model and Structures

North bound interface to cloud, this component implements CWMP CPE. Through south bound interface, this component interacts with CR and FCs to perform normalized actions. And the related data structures and function prototypes will be defined by CCSP Architecture Specification, and please refer to it for details. This document only discusses specific items related to the TR-069 PA component.

Upon start up, the TR- 069 PA loads a mapping configuration file which includes information on how to map external namespaces specific to the PA to internal namespaces on supported data model parameters as well as external and internal data types of each parameter.

With respect to downloadable applications that define their own data model (defining external data model only), TR-069 PA must be able to route normalized calls to appropriate applications via Application Framework Adapter which should be covered by TR-069 Protocol Agent Software Design Document. The Application Framework Adapter is the default route for all data model namespace for which there is no explicit mapping to an internal namespace.

TR-069 PA mapping file schema

TR-069 PA needs a XML mapping file to achieve the following goals.

Error Mapping

CCSP Component Design Rules specification defines some CCSP generic error codes. As well, each functional component may define its own CCSP error codes. TR-069 PA has to maintain the mapping between CCSP error codes and TR-069 error codes. The goal is to make CCSP components agnostic to any protocol specific error codes and data models.

TR-069 data type and CCSP data type mapping

The internal CCSP data model is based on TR-181 and follows the data types as defined in TR-181. Therefore it is not necessary to map data types since the TR69 ACS also uses TR-181 for configuration and diagnostics. The optional data type mapping is only for simply mapping. In foreseeable future, TR-069 PA is not expected to implement any data type mapping due to the significant differences CCSP and cloud data types, value range, the way to interpret them. We list this item here for completeness of discussion.

Namespace mapping

It is required by the current architecture design that the TR-069 PA CCSP component needs to list all supported TR parameters. The mapping file contains the internal data model parameters supported by all CCSP components and their corresponding external cloud names. For TR-69, however this would be an easy mapping since the internal CCSP Data model is based on TR-181.
Also considering how complexity a CPE may be, TR-069 PA may have to know which sub-system a parameter is supposed to be retrieved from. Due to complexity of data type mapping, no data type/value mapping is needed.

To support use case of multiple sub-systems, each parameter listed includes an optional sub-system name to indicate that the owning Functional Component is on the specified sub-system. TR-069 PA therefore queries local CR to discover that owning FC.

Moreover, the mapping file also includes CCSP internal namespaces that are invisible to the cloud server through TR-069 PA. Usually those namespaces are extensions to TR data models, but this is not required, meaning even for TR data model namespaces, we can do exactly the same if it makes sense.

When PA receives a GetParameterNames request on "Device." for example, PA through the mapping file and knows the possible sub-systems, and queries local CR to discover all relating FCs that own namespaces under "Device.". Then PA makes a GetParameterNames normalized action call to each owning FC on local or remote sub-system. After the call returns, PA needs to check each returned namespace against the mapping file to filter out un-supported or invisible ones, and then merge results and sends to cloud server.

CCSP Message Bus APIs

Per CCSP Architecture specification and CCSP Common Component Design Rules document, we will only define message bus APIs for the following normalized actions.

- GetParameterNames
- GetParameterValues
- SetParameterValues
- GetParameterAttributes
- SetParameterAttributes
- SetCommit
- AddTblRow
- DeleteTblRow

CCSP namespaces defined for other normalized actions

For other normalized actions, they will be covered with data model approach. In details, CCSP namespaces are defined for each action. These actions will be requested by PA via setting their values respectively to the namespaces defined. These namespaces are defined in each Functional Component Software Architecture Specification.

CCSP Common Component Design Rules document does not define a specific action and let a FC to do it, the mapping must be included into TR-069 PA namespace mapping file. For action such as Download, depending on our methodology, we may require only single FC to handle all download, or we allow multiple FCs to handle different type of downloads. For latter case, the namespace mapping file must include a mapping entity per download file type. With current CCSP architecture, single functional component will take care of download of all file types. Please refer Secure Software Download Software

CCSP data model related events

The following data model related events must be handled by TR-069 PA.

- DUStateChangeComplete*
- AutonomousDUStateChangeComplete*
- TransferComplete**
- AutonomousTransferComplete**
- ValueChanged

> * must be supported if CPE supports normalized action 'ChangeDUState'.

> ** must be supported if CPE supports Download or Upload.

Event ValueChanged needs to be examined carefully by TR-069 PA. FC signals ValueChanged event on message bus on a parameter that the notification counter is larger than zero. Also FC indicates which component causes the value change by specifying 'writeID' which was used in SetParameterValues and SetCommit actions.

In order for TR-069 PA to capture event ValueChanged, TR-069 PA must subscribe to the signal through Component Registrar. And once TR-069 PA receives such event, it needs to check if the value change is caused by itself by checking 'writeID' included in the signal against TR-069 PA's own writeID. If they are same, this signal is simply ignored. Otherwise, PA needs to either notify ACS about a parameter value change in active or passive way as defined by TR-069 specification, or for some special namespaces, PA knows they are asynchronous event indication, could it be TransferComplete, DUStateChangeComplete, or diagnostics completion and so on.

Non-Data Model APIs

CCSP Message Bus APIs

Component Registration is the same for all common components. We will use whatever defined in CR software design specification to register TR-069 PA and subscribe signals this component wants to capture such as ParameterValueChangeSignal.

You may refer to Component Registrar SFS, SDS, and CCSP High Level Architecture Specification for function prototypes related to Message Bus APIs.

TR-069 PA follows transaction control mechanism defined by CCSP Architecture specification based on session ID.

In case normalized action call on external namespaces that do not have counterpart internal namespaces or say PA cannot get such mapping information from CR, TR-069 PA must be able to route calls to default namespace handler which could be Application Environment Manager through private interface per CCSP Component Registrar specification.

Code Flow

Figure 1 shows the overview of TR-069 Protocol Agent.
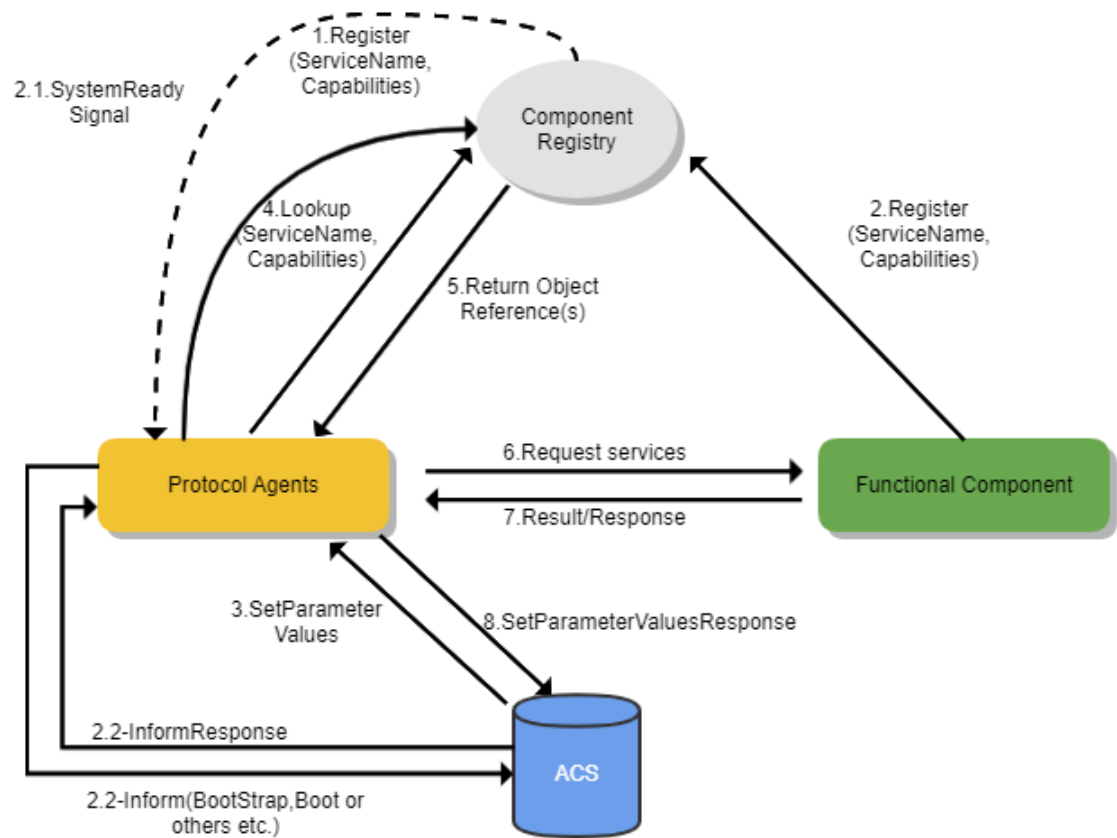
Figure 1 CCSP TR-069 PA System Overview

This component provides cloud interface to TR-069 ACS and get services from internal CCSP interface to communicate with functional components via normalized actions and internal namespaces.

As you can see from figure 1, when TR-069 PA receives a request from ACS, it resolves namespace with CR, and makes a normalized action invocation over message bus into target functional component. Once TR-069 PA receives response or error from functional component, it construct CWMP SOAP message back to ACS.
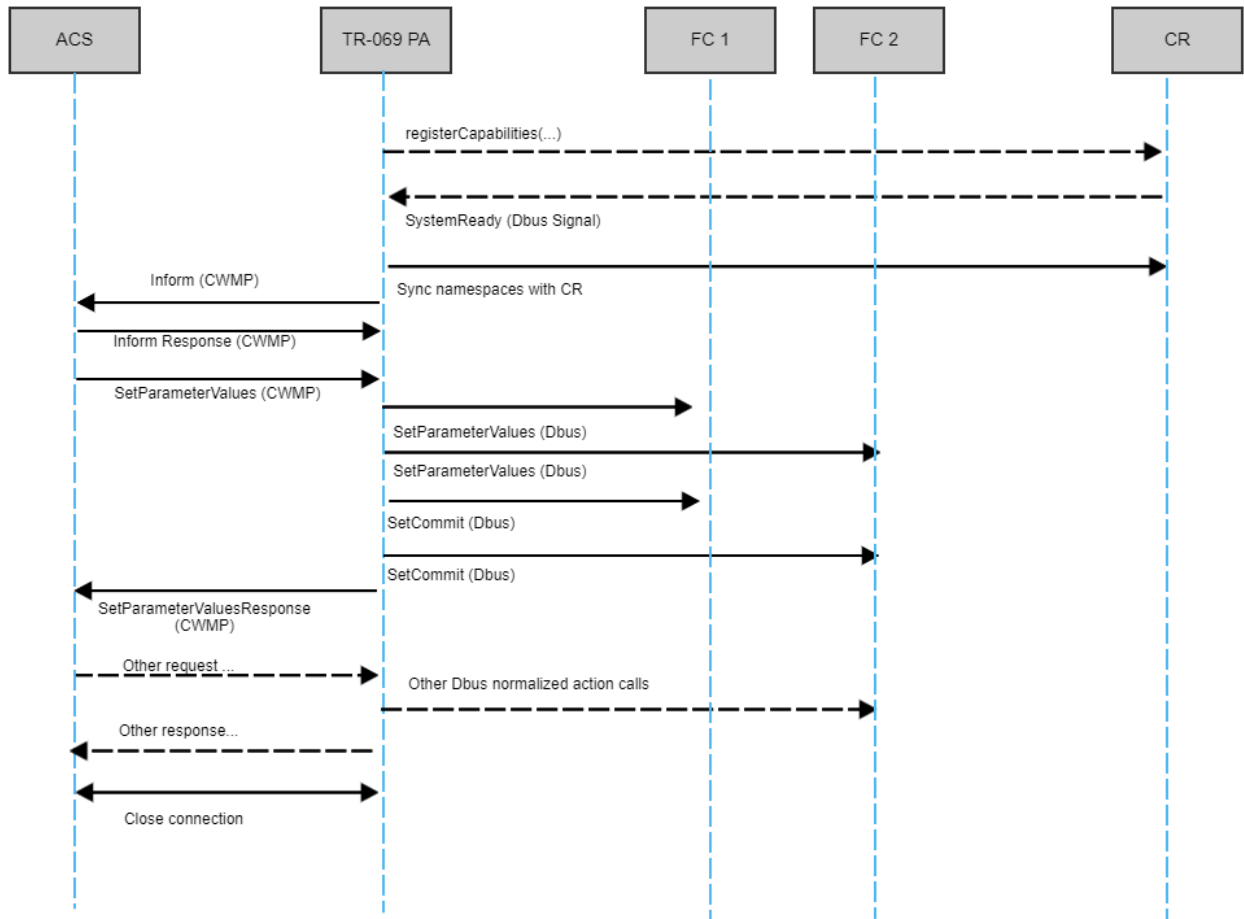
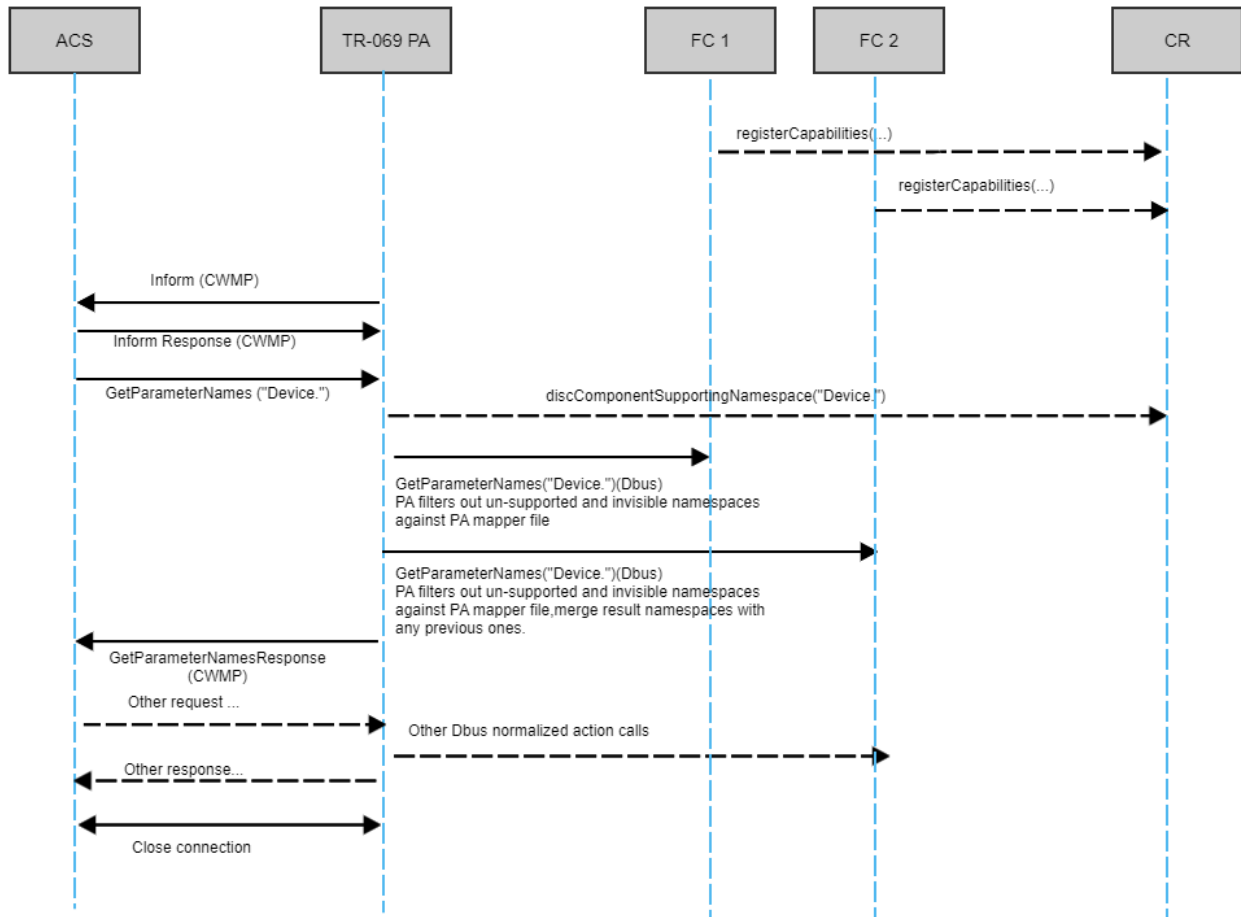**Figure 2 TR-069 PA SetParameterValues data flow**

Figure 3 TR-069 PA GetParameterNames data flow

GetParameterNames("Device."). PA needs to find out all owning FCs on local and remote sub-systems, and make GetParameterNames normalized action calls. Once it gets results, PA needs to filter out un-supported and invisible namespaces (no matter partial or full names) and send back to ACS.

TR-069 Notifications

Active and Passive notifications

Implemented as part of **notify_comp**, implements a notification function to all the Protocol Agents who have registered for notification of changes in the RDK-B stack.

- Notifies TR069 PA, in case of any value change on notify enabled parameters. Device.TR069Notify.X_RDKCENTRAL-COM_TR069_Notification
- No VALUE CHANGE notification may be sent to the ACS when the origin of the change is a SetParameterValues through TR-069. The TR-069 PA discard value change events based on the event's   origin.
- Persistence of "VALUE CHANGE" events across reboots or CcspTr069PaSsp restart , In current implementation, Value change event is getting lost in passive notification if reboot or restart happened.

Recent TR-069 Enhancements

- Support for TR-069 active and passive notifications.
  - Dependency: notify_comp and its recipe.
  - Dependency: when using an older release of RDK-B, be sure to apply the following locking bug fix, which prevents a race condition on a bulk parameter value set that results in intermittent loss of notifications for some changed parameters.
  - Tr069PA support in notify_comp.
  - No VALUE CHANGE notification may be sent to the ACS when the origin of the change is a SetParameterValues through TR-069. The TR-069 PA must discard value change events based on the event's origin.
  - Reliable delivery of events across TR-069 agent restart. If a VALUE CHANGE notification is not delivered before restart, it must be retried until successful, even across reboots.
- Support for TR-069 diagnostics complete notifications.

- ○ Send 8 DIAGNOSTICS COMPLETE to the ACS only when the diagnostics test was initiated through TR-069. Supported for Device.IP.Diagnostics.TraceRoute and Device.IP.Diagnostics.IPPing.
- *Custom Mapper* to translate the internal data model to a different one offered through TR-069. This can fix:
  - ○ TR-069 data model violations such as Device.SelfHeal
  - ○ CableLabs (CM-SP-eRouter) instance index standardisation, e.g. Device.WiFi.Radio.10000. and Device.WiFi.SSID.10001
  - ○ MSO-specific data model object and parameter names

Enhancement under Consideration

- *Device-Gateway Association*, TR-069 Annex F.
  - ○ This implies changes to the DHCPv4 and (optionally) DHCPv6 server; V-I Vendor Specific options sent and parsed, leases exposed in Device.ManagementServer.ManageableDevice., notification support for Device.ManagementServer.ManageableDeviceNumberOfEntries.

Identified Limitations

1. When several changes are made to a parameter that is configured for active notification, only a single VALUE CHANGE event is delivered, with the latest value. This only applies when the notifications are somehow prevented from being delivered immediately (e.g. WAN down or ACS down or unreachable). This is a design choice.
2. There is no support for lightweight notifications. Only active and passive notifications, SetParameterAttributes values 1 and 2, are supported.
3. There is no support for notifications on partial paths. This may require more fundamental changes to the notification system of the RDK bus. Because of this and lack of event notification support in some data model parameters, Set/GetParameterAttributes test cases in od_128.tcl of the TR-069 CDRouter add-on fail.
4. There is no support for HTTPS. This requires provisioning of a CA certificate list for server authentication, and a (secure handled) device private key and certificate pair for client authentication.
5. TR-069 strongly recommends random generation of the path of the ConnectionRequestURL. Currently, if Device.ManagementServer.X_CISCO_COM_ConnectionRequestURLPath is not set, a non-random value is generated.
6. If AccessListChange is true in a SetParameterAttributes method call, and no AccessList is provided, the Parameter provided as a Name is set to read only. The scope of this problem may be wider. In general, take care to set AccessListChange to false.