Tracking Object Type Statistics in EVO (Intern Project 2024)

Name: Jiro Ryzard Noor
Mentor: Jithin Girish (jithing@juniper.net)
Manager: Sudhir Cheruathur (scheruathur@juniper.net)
Email: jnoor@juniper.net, jiro.noor@berkeley.edu
School: University of California, Berkeley
Project Description: Tracking Application Typeguid Subscription Request Statistics in JUNOS EVO Publisher Subscriber Model

Background

In the current JUNOS CLI, when running the command “show platform distributor statistics summary“, we get the total object stats as an output

```
Distributor total object stats
  Node: fpc0
    Object adds          : 14183
    Object changes       : 129391
    Object deletes       : 1
    Object cleanup pending : 0
    Object add errors    : 0
    Object change errors: 0
    Object delete errors: 0
    Object delete done notification requests
      received                    : 0
      sent                        : 0
```

However, this summary often lacks specificity and fails to indicate the particular types of objects that are being added, changed, deleted, or updated. Consequently, this ambiguity makes it challenging to decipher the exact nature of the objects flowing throughout the system. To address this issue, my project aims to extend this functionality by implementing a more detailed tracking system. This enhanced

system will categorize and monitor each type of object separately, providing a clear and comprehensive overview of all operations performed on these objects. By doing so, it will allow for more precise monitoring and analysis, improving the system's transparency and facilitating better decision-making based on the specific data being processed.

CLI Command

Purpose: Getting the command “show platform distributor type-statistics” to appear on the CLI

```
in file /cevo/evoapp/distributord/ddl/src/distributord.cmd.dd
```

```
root@EVOvSCAPA1-RE0-re0> show platform distributor statistics
type-statistics ?
Possible completions:
  <[Enter]>            Execute this command
  app                  App Name
  client               Client Number
  node                 Node name
  priority             Priority level (high, normal, bulk)
  |                    Pipe through a command
```

Options:

`app`

- Specify the application name.
- When you choose `app`, you can also specify a priority level.
- **Priority Levels:**
  - `high`: Show statistics for high priority.
  - `normal`: Show statistics for normal priority.
  - `bulk`: Show statistics for bulk priority.
  - If no priority is specified, the default is to show statistics for all priority levels associated with the specified app name.

`client`

- Specify the client number.
- When you choose `client`, the priority level will not be taken into account.
- The command will display statistics for all clients matching the specified client number.

`node`

- Specify the node name.
- If no node is specified, the default is to show statistics for all nodes.

`priority`

- Specify the priority level.
- Only applicable when `app` is selected.
- **Priority Levels:**
  - `high`: Show statistics for high priority.
  - `normal`: Show statistics for normal priority.
  - `bulk`: Show statistics for bulk priority.
- If not specified, the default is to show statistics for all priority levels associated with the specified app name.

Data Structure (EVL)

dist_clnt_type_stats

In evo/evo_dds/distributord_module/src/main/evl/dist_stats.evl

```
type dist_type_stats {
string      obj_type;           /* Name of entity being tracke
d */
uint64      adds_rcvd;           /* Number of adds received */
uint64      changes_rcvd;         /* Number of changes receive
d */
uint64      deletes_rcvd;          /* Number of deletes receiv
ed */
uint64      updates_snt;          /* Number of updates sent*/
uint64      deletes_snt;          /* Number of deletes sent*/
}
```

```
type dist_clnt_type_stats {
string node;  /* Name of Node */
string client; /* Name of Client */
string priority; /* Name of Priority */
array(dist_type_stats) dist_type_stats_arr;
}

object dist_clnt_type_stats;
```

This is the EVL data structure used to represent the type-statistics of a single client/app. Each client/app belongs to a node, contains an id (and name) and priority channel (high, normal, bulk), as well as a list of objects. Each of these objects will have a typeguid name (in either human readable or hexadecimal), as well as request statistics (Adds Received, Change Received, Deletes Received, Updates Sent, and Deletes Sent).

Data Structures (C++)

The system involves two main components: the `cdist_type_guid_stat` structure for storing statistics and the `EvoTypeGuidKey` class for managing and accessing these statistics using unique type GUIDs.

struct cdist_type_guid_stat_

```
typedef struct cdist_type_guid_stat_ {
string      obj_type;
u_int64_t   adds_received = 0;  /**< # of obj adds received fr
om clnt */
u_int64_t   changes_received = 0;  /**< # of obj changes recei
ved from clnt */
u_int64_t   deletes_received = 0;  /**< # of obj deletes recei
ved from clnt */
```

```
u_int64_t   updates_sent = 0;  /**< # of obj deletes received
from clnt */
u_int64_t   deletes_sent = 0;  /**< errs in receiving objs fro
m clnt */
} cdist_type_guid_stat;
```

The `cdist_type_guid_stat` structure holds statistics related to object operations received from a client and sent by the server.

Members

- `obj_type`: A string representing the type of the object.
- `adds_received`: The number of add operations received from the client.
- `changes_received`: The number of change operations received from the client.
- `deletes_received`: The number of delete operations received from the client.
- `updates_sent`: The number of update operations sent to the client.
- `deletes_sent`: The number of delete operations sent to the client.

EvoTypeGuidKey class

Members and Methods

- Constructors:
  - `EvoTypeGuidKey(const std::vector<uint8_t> tg)`: Initializes the `tguid` from a vector of bytes.
  - `EvoTypeGuidKey(const evo_type_guid_t tg)`: Initializes the `tguid` from an existing `evo_type_guid_t`.
- Hash Function:
  - `struct EvoTypeGuidKeyHash`: Provides a hash function for `EvoTypeGuidKey`.
- Comparison Operators:
  - `bool operator==(const EvoTypeGuidKey &other) const`: Compares two `EvoTypeGuidKey` objects for equality.
  - `bool operator!=(const EvoTypeGuidKey &other) const`: Compares two `EvoTypeGuidKey` objects for inequality.
  - `struct LessThan`: Provides a less-than comparison operator for sorting.
- String Conversion:
  - `std::string ToString()`: Converts the `tguid` to a string representation.

Type Definitions

- **`type_guid_t`**: A type alias for `std::vector<uint8_t>`, representing a type GUID.
- **`clnt_typeguid_stats_map_t`**: A type alias for an unordered map that uses `EvoTypeGuidKey` as the key and a pointer to `cdist_type_guid_stat` as the value. The map uses the `EvoTypeGuidKeyHash` for hashing.

We then add the HashMap **cl_typeguid_stats_map** is of type clnt_typeguid_stats_map_t to **class cdist_clnt_t**

Calculation and Retrieval of Statistic (C++)

Statistic Calculation

1. The adds_received and changes_received stat per object per client is updated in DistMainChannel::rcvObjMsg in file /cevo/evoapp/distributord/src/cdist/DistMainThread.cpp
2. The deletes_received stat per object per client is updated in DistMainChannel::deleteObject in /cevo/evoapp/distributord/src/cdist/DistMainThread.cpp
3. The updates_sent and deletes_sent stat per object per client is updated in cdist_evostate_update in file cevo/evoapp/distributord/src/cdist/cdist_evostate_clnt.cpp

TypeStatsSnapShot Class

Overview

The `TypeStatsSnapShot` class is designed to capture and store statistics for clients and their associated type GUIDs based on the sessionID of the request. It interacts with other components such as `cdist_clnt_t` and `EvoTypeGuidKey` to achieve this.

Class Definition

`TypeStatsSnapShot`

Members

- `size_t index`: Index for iterating or accessing elements within the vector.
- `std::vector<dist_clnt_type_stats_h> dist_clnt_type_stats_vector`: A vector to store client type statistics handles.

Constructor

- **Parameterized Constructor**

```
TypeStatsSnapShot(std::string client_arg, std::string app_arg,
std::string prio_arg)
```

- Initializes the `TypeStatsSnapShot` object with specific parameters for client, application, and priority, and processes the relevant statistics.
  - Initializes `index` to -1.
  - Determines whether to process all clients based on the provided arguments (`client_arg`, `app_arg`, `prio_arg`).
  - If a specific client number is provided and found, processes its statistics.
  - If no specific client is provided or the arguments specify processing all, iterates over all clients in `cdist_clnt_map` and processes them as necessary.
  - Constructor uses fill_dist_type_stats_arr function to fill out the dist_type_stats array for each client/app and uses the handle_client_stats to create the dist_clnt_type_stats object

Private Methods

```
dist_type_stats_h fill_dist_type_stats_arr(EvoTypeGuidKey key,
cdist_type_guid_stat* type_stats)
```

- Fills the distributor object stats based on the provided `key` and `type_stats`.
  - Creates a new `dist_type_stats_h` object for storing statistics.
  - Looks up the object type name using `TypeGuidMapper`.
  - Sets the object type and various statistics (`adds_received`, `changes_received`, `deletes_received`, `updates_sent`, `deletes_sent`) in the `dist_type_stats_h` object.
  - Returns the populated `dist_type_stats_h` object.

```
void handle_client_stats(cdist_clnt_t* clnt)
```

- Handles the statistics for a specific client, aggregating and storing them in the vector `dist_clnt_type_stats_vector`
  - Creates a new `dist_clnt_type_stats_h` object for the client.
  - Sets client-specific attributes such as display name, node, and priority.
  - Checks if the client's type GUID statistics map is empty. If so, adds the client stats object to the vector and returns.
  - Allocates an array for the type statistics in the client stats object.
  - Iterates over the client's type GUID statistics map, fills the statistics for each type using the fill_dist_type_stats_arr function

- Adds the populated client stats object to `dist_clnt_type_stats_vector`.

Displaying Object Type Name and GUID (C++)

The `TypeGuidMapper` class is designed to provide a mapping between type GUIDs (Globally Unique Identifiers) and their corresponding human-readable names. This class is implemented with static members and methods to ensure a single, shared mapping across the application. The mapping is initialized from a file upon the first lookup request. This class is located at cevo/evoapp/distributord/src/cdist/type_info_util.h

Class Definition

Members

- **`type_guid_to_name`**: A static map that stores the mapping from type GUIDs to their corresponding names.
- **`is_initialized`**: A static boolean flag indicating whether the mapping has been initialized.

Public Methods

- **`LookupTypeName`**: Looks up the human-readable name for a given type GUID. If the mapping is not initialized, it initializes the mapping from a file.

Private Methods

- **`initialize`**: Reads the mapping file and populates the `type_guid_to_name` map.

Detailed Description

Data Structures

`std::map<std::string, std::string> type_guid_to_name`

This map stores the relationship between type GUIDs and their human-readable names. The keys are type GUIDs represented as strings, and the values are the corresponding human-readable names.

Methods

```
static std::string LookupTypeName(std::string tg) {
```

- **Parameters**:
  - `tg`: The type GUID for which the human-readable name is to be looked up.
- **Return**:

    - The human-readable name corresponding to the type GUID if found in the map, otherwise the type GUID itself.
- **Functionality**:
    - Checks if the mapping is initialized. If not, it sets the `is_initialized` flag to `true` and calls the `initialize` method to populate the map.
    - Looks up the type GUID in the `type_guid_to_name` map and returns the corresponding human-readable name if found. If not found, it returns the type GUID itself.

```
static void initialize() {
```

- **Parameters**: None
- **Return**: None
- **Functionality**:
    - Opens the mapping file located at `/var/tmp/.__type_guid_to_name_mapping`.
    - Reads the file line by line. For each line, it splits the line into a key and value pair and inserts them into the `type_guid_to_name` map.
    - Closes the file after reading all lines.