Monitoring and Metering using Open Source Technology in Enterprise Content Management Systems

Students:
Anisa Koleci
Abinaya Kumar
Jagan Shanmugam

Contents

1. Motivation:

To view and analyze the data from Enterprise Content Management systems, (hereafter referred as ECM systems) and to guarantee an excellent user experience, a monitoring tool which provides insights is needed. Through the use of open source tools Telegraf, Influxdb and Grafana, we want to monitor the ECM systems to observe the performance of the system and uncover the problems whenever there is a drop in performance.

We formulate our objectives as the following points.

1. To design a service provider dashboard with the below,
    a. Overall system load across all and by each tenant: CPU, I/O, Network, Memory
    b. Access patterns: Log-on, log-off : across and by each tenant
    c. Number of CRUD-S operations by time, Service Level Agreement (SLA) and related infringement
2. To design a tenant dashboard with the below objectives,
    a. Access patterns of the tenant and of its users
    b. Workload, Type and amount of documents: Number of CRUD-S (operations) by time, by type, by user

2. Setup and background:

Open source stack Telegraf, Influxdb, Chronograf and Kapacitor (TICK) along with Grafana is deployed in openstack infrastructure as docker containers.
Below are versions of each component in the environment.
Telegraf - 1.3.3
Influxdb - 1.2.4
Chronograf - 1.6.2
Kapacitor - 1.3.0
Grafana - 5.1.0

Telegraf plugins are installed to populate the system metrics into telegraf database. Along with that, new tables/measurements are created by defining a schema and data is populated using python scripts. Influxdb is configured as a data source in Grafana and Chronograf and thereby acts as a source for all the dashboards. Below, we explain the dashboards which are designed to monitor the ECM system from the data collected and explain common use cases.

3. Dashboards in Grafana:

3.1 Service provider dashboard:

The service provider offers its services to tenants. In order to maintain the credibility of the services offered, there needs to be a tool to monitor the performance of system resources and its client activities. This dashboard serves as a visualisation tool for ECM monitoring from a service provider's point of view. It gives stats like the health of system in terms of CPU usage percentage, disk space used and an overview of the tenants activities in terms of CRUDS operations.

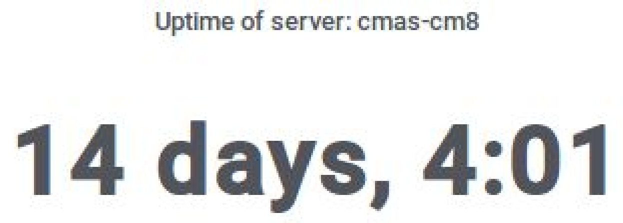
The main objective is to make sure that the system resources are well under the threshold and that the services comply to the agreements between the client and the provider.
The following are the sections under which panels are grouped:

3.1.1 ECM - System overview
3.1.2 ECM - System performance
3.1.3 ECM - Access pattern by tenants

3.1.1 ECM - System overview:

This section provides all the instant values related to the system usage, SLA and current users across tenants and of tenant filtered. In order to show the numeric values, Single stat panels are mostly used in this section. A quick glance of this section equips the user with the information about the current performance of the system and SLA infringement in the past 30 days.

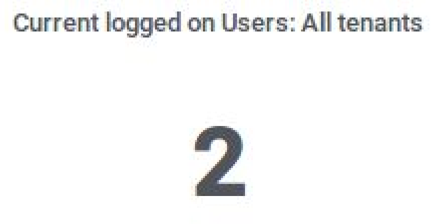
- **Uptime of server since last reboot: <server name>**


Uptime of server: cmas-cm8

14 days, 4:01

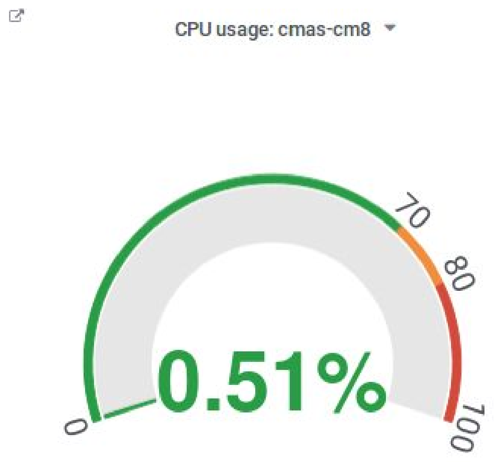
Description:
This panel shows the active time of the ECM server selected template variable since the last shut down. The server uptime is important to monitor the status of the server. A Single Stat panel is the best one to show the numeric value. By looking at this panel, one can quickly identify the last time when the server was down.
For the query click here.

- **Current logged on users: All tenants**



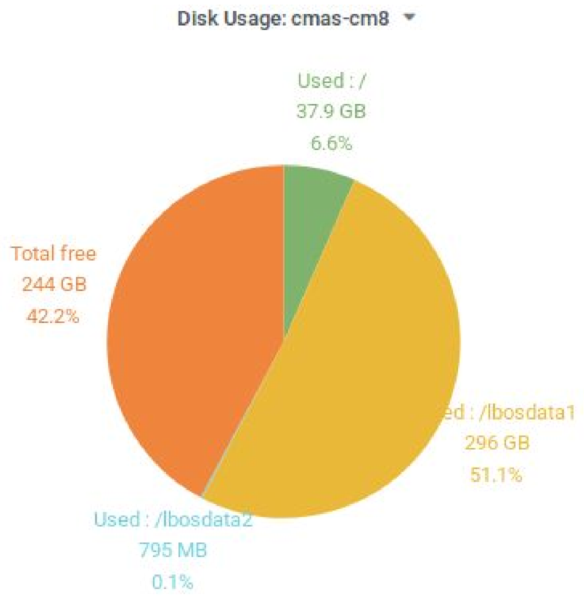
Description:
The total number of logged on users across all tenants in the ECM system at the moment is shown in this panel. This information helps to identify the current user load in the system across all tenants.
For the query click here.

- **CPU usage: <server name>**



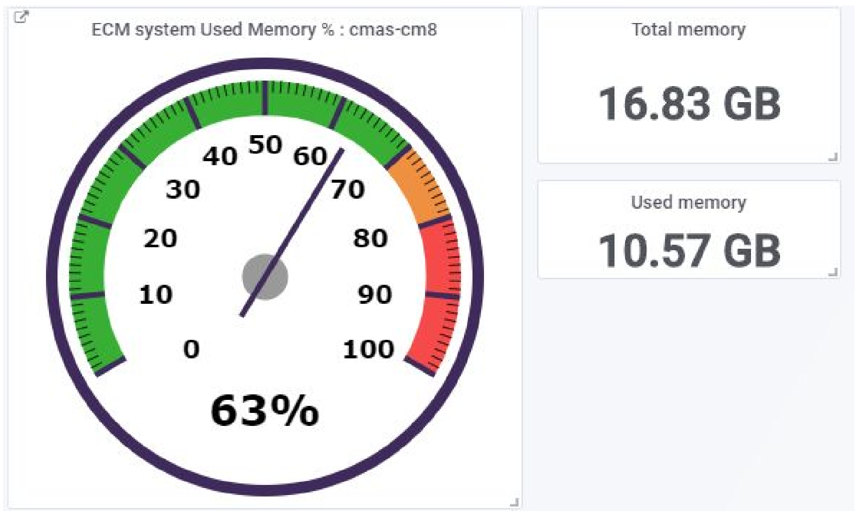
Description:
This panel provides the CPU usage percentage of the server filtered in the template variable. A Single Stat panel with Gauge display is used to visualize this metric as thresholds can be configured and shown with different colours. Custom thresholds have been set at points like 70%, 80%. On looking at this panel, one can instantly get the criticality of CPU usage.
For the query click here.

- **Disk usage: <server name>**



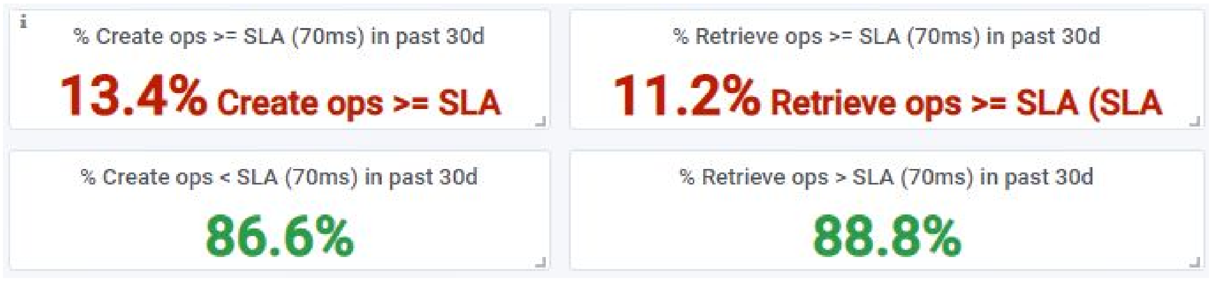
Description:
This provides an overview of the disk space occupied by root disk(CM8 catalog storage), object store disk(CM8 document storage) and total free space in the filtered host server. CM8 disk is subdivided into Catalog Storage disk and Content object document storage disk. The Catalog storage disk contains data indices and is a central storage of information. The document object storage stores the documents of users and is further divided into /lbosdata1 and /lbosdata2. A pie chart is used to quickly visualise the quantity occupied by each disk and the free space available.
For the query click here.

- **ECM System Used Memory %: <server name>**



Description:
Memory used percentage is a vital metric in monitoring the health of the system. The D3 Gauge panel plug-in is used which provides means to clearly differentiate the different thresholds configured. This panel gives a quick overview of CPU memory percentage used in the server selected above. The numeric values of the memory available and used are displayed for reference.
For the query click here.

- **SLA**


i

% Create ops >= SLA (70ms) in past 30d
13.4% Create ops >= SLA

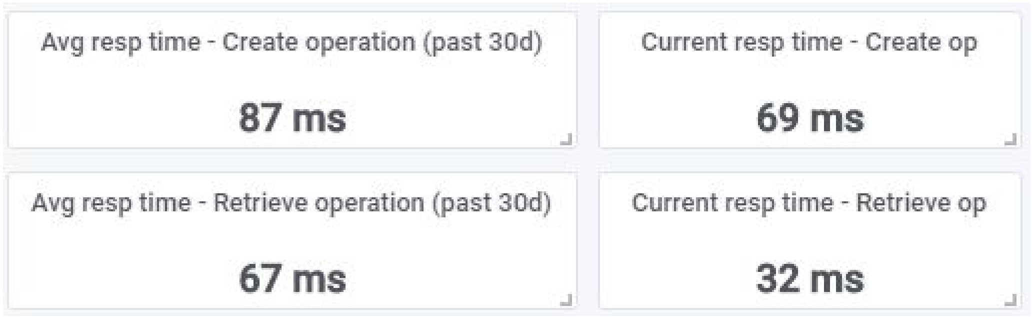
% Retrieve ops >= SLA (70ms) in past 30d
11.2% Retrieve ops >= SLA (SLA

% Create ops < SLA (70ms) in past 30d
86.6%

% Retrieve ops > SLA (70ms) in past 30d
88.8%

Description:
Service Level Agreements (SLA) defines the level of services that needs to be offered by the service provider to the tenant. For this panel, custom SLA’s have been defined for the duration of Create and retrieve operations. A custom threshold of 70 ms for create and 70ms for the retrieve operation has been defined. This panel provides information about the percentage of create and retrieve operations that have not complied with the defined threshold. This assists the service provider in taking immediate measures to bring back the system in order to bind to the SLA’s.
For the query click here.

- **Average response time of operations:**


Avg resp time - Create operation (past 30d)
87 ms

Current resp time - Create op
69 ms

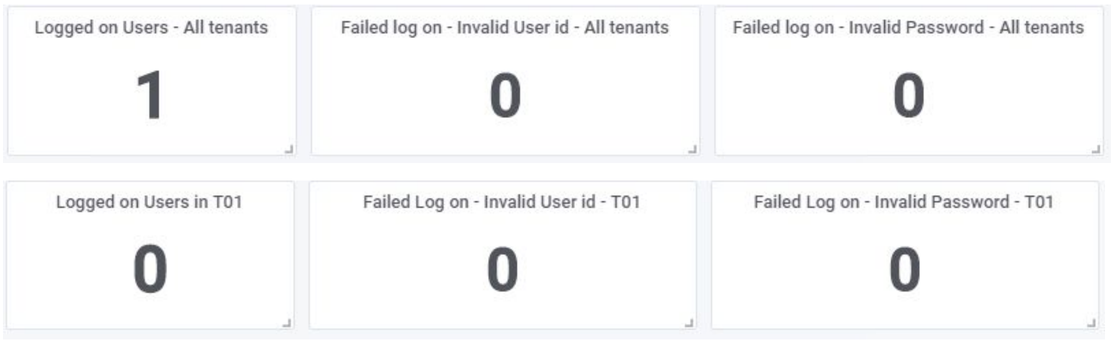
Avg resp time - Retrieve operation (past 30d)
67 ms

Current resp time - Retrieve op
32 ms

Description:
The response time of operations are usually affected by network bandwidth, number of users, number and type of requests submitted.

This panel shows both the average response time over the past 1 month and the current response time which helps to analyze system resources. For the query click here.

- **Logged on users:**



Logged on Users - All tenants	Failed log on - Invalid User id - All tenants	Failed log on - Invalid Password - All tenants
1	0	0

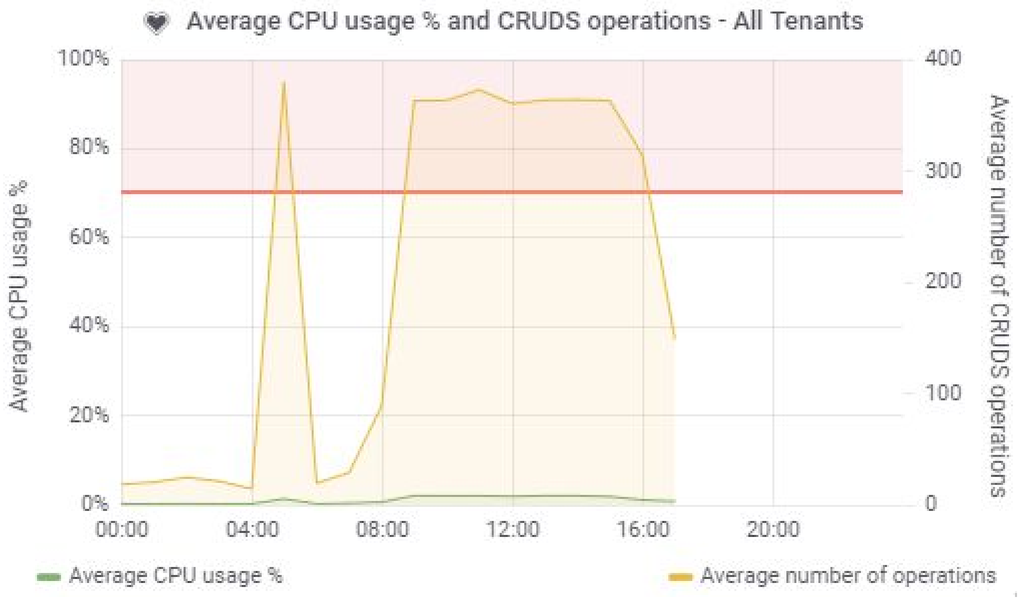
Logged on Users in T01	Failed Log on - Invalid User id - T01	Failed Log on - Invalid Password - T01
0	0	0

Description:
Current logged on users across all tenants and in each tenant are shown in the above panels. The current failed log on cases (invalid user id and password) are also displayed to analyze the tenant activities.
For the query click here.

3.1.2 ECM - System performance

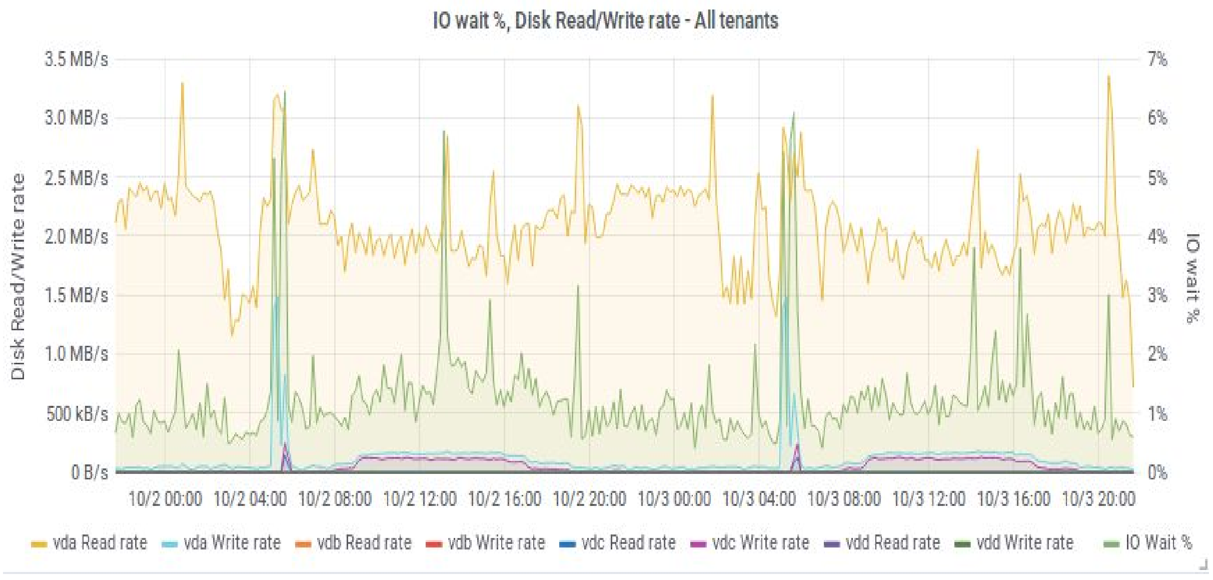
In this section, ECM system performance is tracked by comparing the system metrics with the ECM metrics over time with four different panels. Each panel is described below.

- **Average CPU usage % and CRUDS operations - All Tenants**



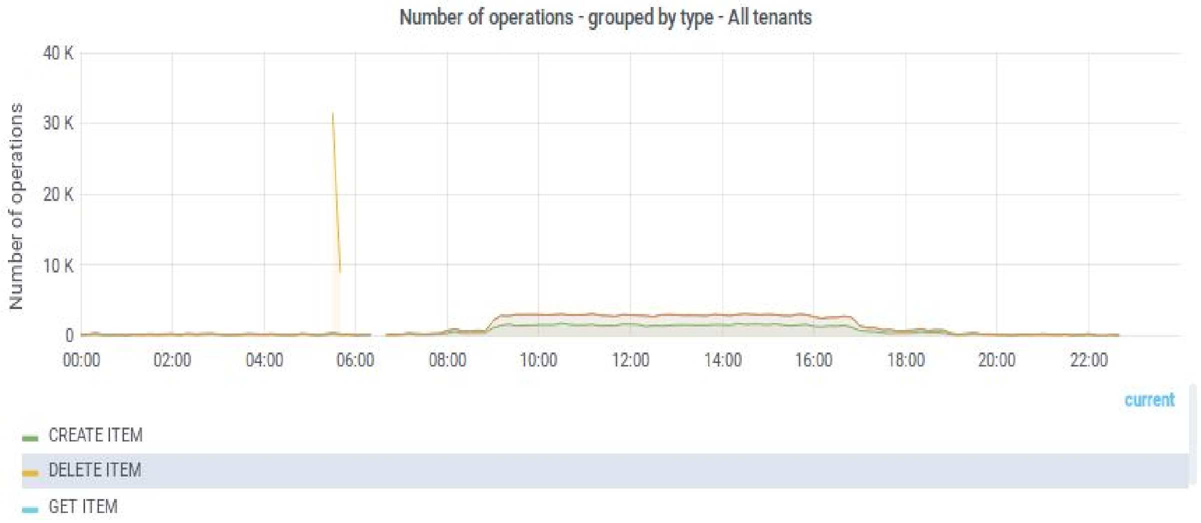
Description:
CPU usage percentage and number of CRUDS operations are metrics which are directly proportional and visualizing them in a time series graph provides intuitive sense to analyze the contribution of CRUDS operations to CPU usage. This panel compares the average CPU usage percentage with average number of CRUDS operations in the same graph. On the left y-axis, average CPU usage % is displayed and on the right y-axis, average number of CRUDS operations. This enables the service provider to evaluate the CPU usage at any particular point in time and the corresponding CRUDS operations performed in the ECM system.
For the query click here.

- **IO wait %, Disk Read/Write rate - All tenants**



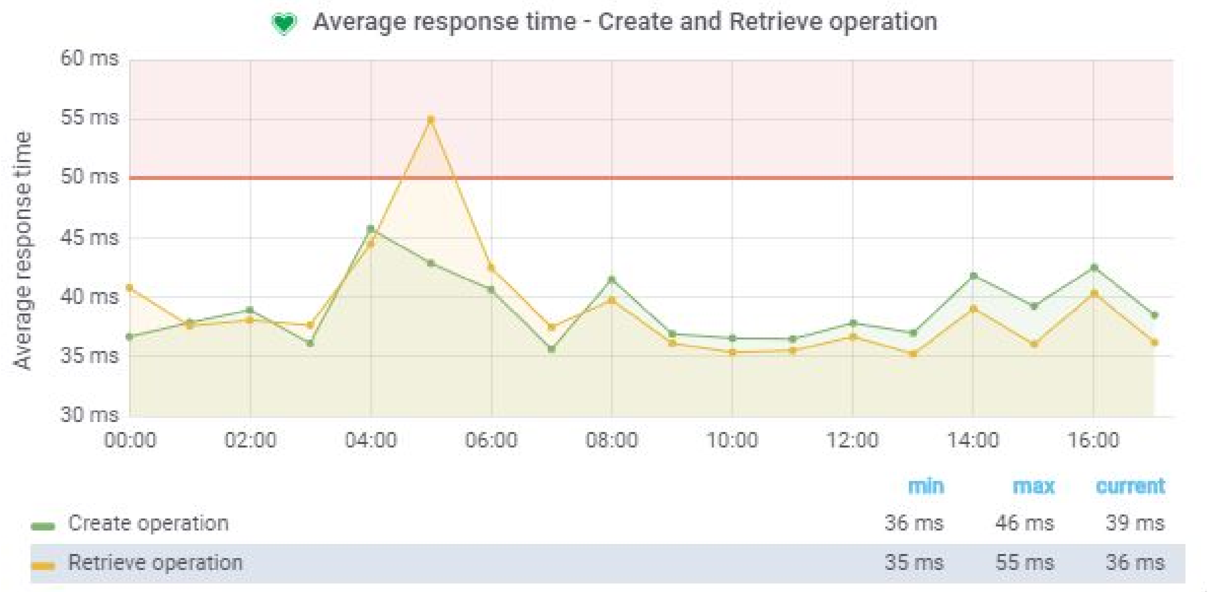
Description:
CRUDS operations from previous graph can also be directly correlated with Disk read and write rates. On the graph, read and write rates of different disks (devices:vda,vdb,vdc,vdd) in the system are displayed. We can notice that read rate of disk vda is around 2.5 MB/s and of disk vdd is around 500 kB/s and other disks are comparatively less. This indicates that disk vda is utilized more than other disks and the cause can be CRUDS operations or databases on the disk. This panel displays the read/write rate of all disks along with the IO wait percentage over time axis. Also, if IO wait percentage is high, we can verify if any of the disks contribute to the peaks.
For the query click here.

- **Number of operations - grouped by type - All tenants**



Description:
Tracking CRUDS operations on a time series individually lets the service provider to analyze the workload on the system. In this panel, CRUDS operations across all tenants and the count of each operation is displayed as a time series graph. In the screenshot above, peak for delete operation before 06:00 indicates that there is a bulk delete occuring during that time interval.
For the query click here.

- **Average response time - Create and Retrieve operation**



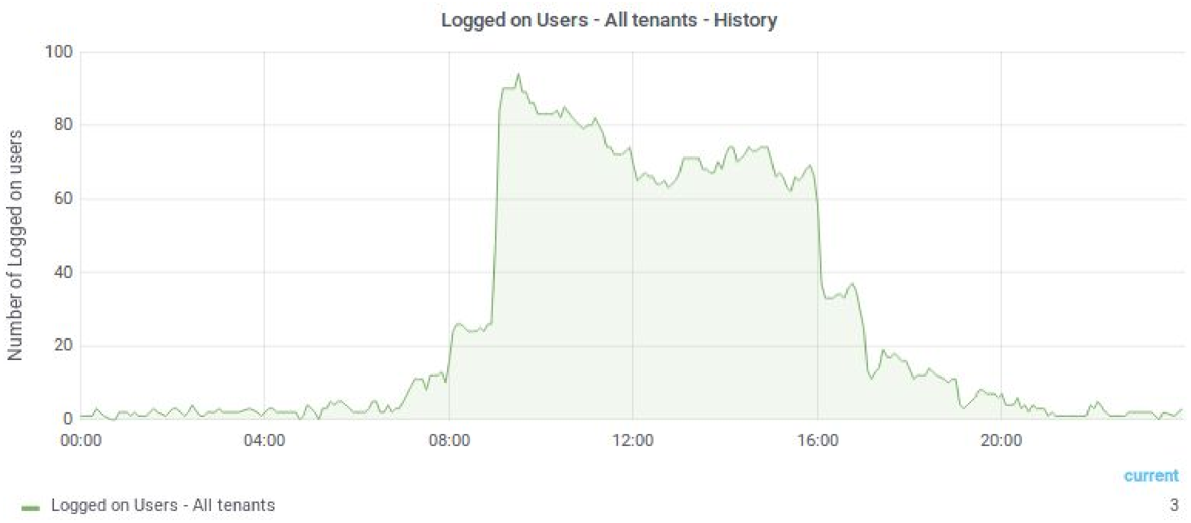
	min	max	current
Create operation	36 ms	46 ms	39 ms
Retrieve operation	35 ms	55 ms	36 ms

Description:
Average response time for create and retrieve operations are crucial for determining if the system responds normally at all times. Over time axis, average response time for create and retrieve is plotted. Peaks in this graph indicate that responsiveness of the ECM system is not normal.
For the query click here.

3.1.3 ECM - Access pattern by tenants

This section is used to monitor the tenant activities in terms of the log on events and workload across tenants and by each tenant. These informations allow us to correlate the system resources usage with the operations performed by the tenants. Since this section provides a history of tenant activities, a time-based Graph panel is used to visualise the data.

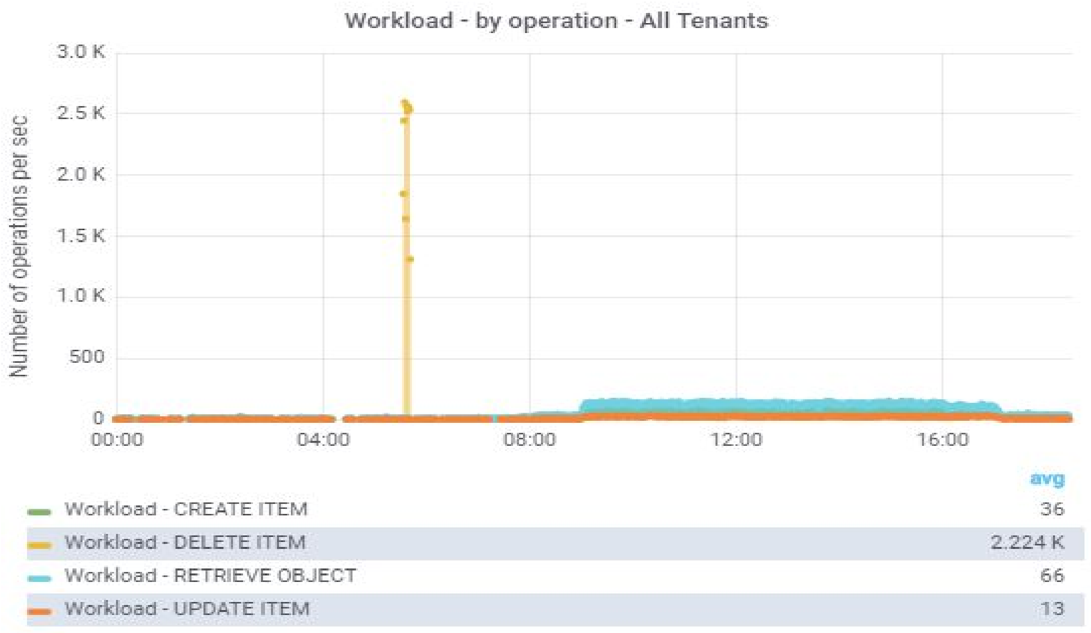
- **Logged on Users - History**



Description:

Each tenant can have many users. The panel shows the number of logged on users across all tenants at each point in time. It gives a history of the frequency of users of all tenants for the time range selected in the dashboard. For the query click here.

- **Workload - by operation**

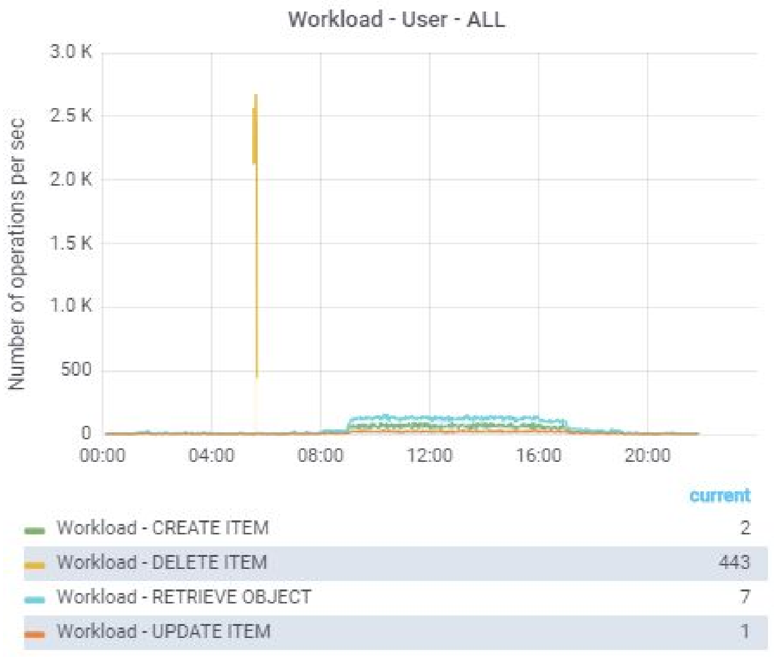


Description:
Workload denotes the number of CRUDS operations. The tenant activities are monitored by the workload as it affects the system resource usage. The higher the workload, the higher the chances of the system being overloaded. A time series graph of the workload of tenants helps to relate to the system usage.
For the query click here.

3.2 Tenant Dashboard

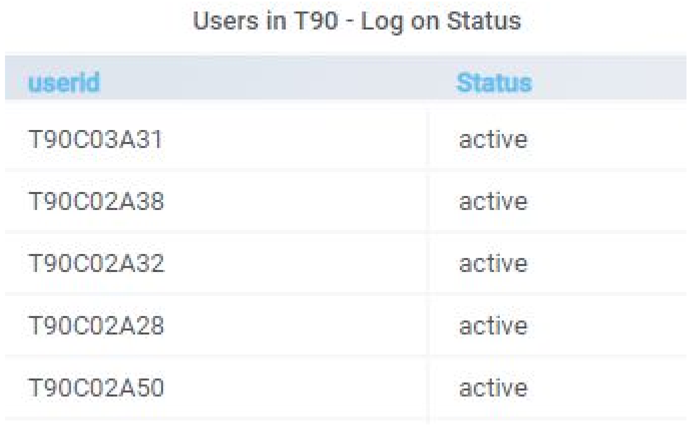
This dashboard serves as a visualisation tool for ECM monitoring from a tenant's point of view. It gives an overview of the tenants activities based on logged on users and CRUDS operations. Tenant dashboard is similar to the service provider dashboard on coarser level but there are extra panels by which we can drill down the metrics to the user level.

- **Workload**



Description:
Workload user is a time series graph that shows the number of operations for the selected user in the dashboard.
For the query click here.

- **Status of logged on users**



Users in T90 - Log on Status

userid	Status
T90C03A31	active
T90C02A38	active
T90C02A32	active
T90C02A28	active
T90C02A50	active

Description:
This table shows the status of all logged on users of the selected tenant in the dashboard. An active user is the one that has been logged on in the system for more than 15 minutes.
For the query click here.

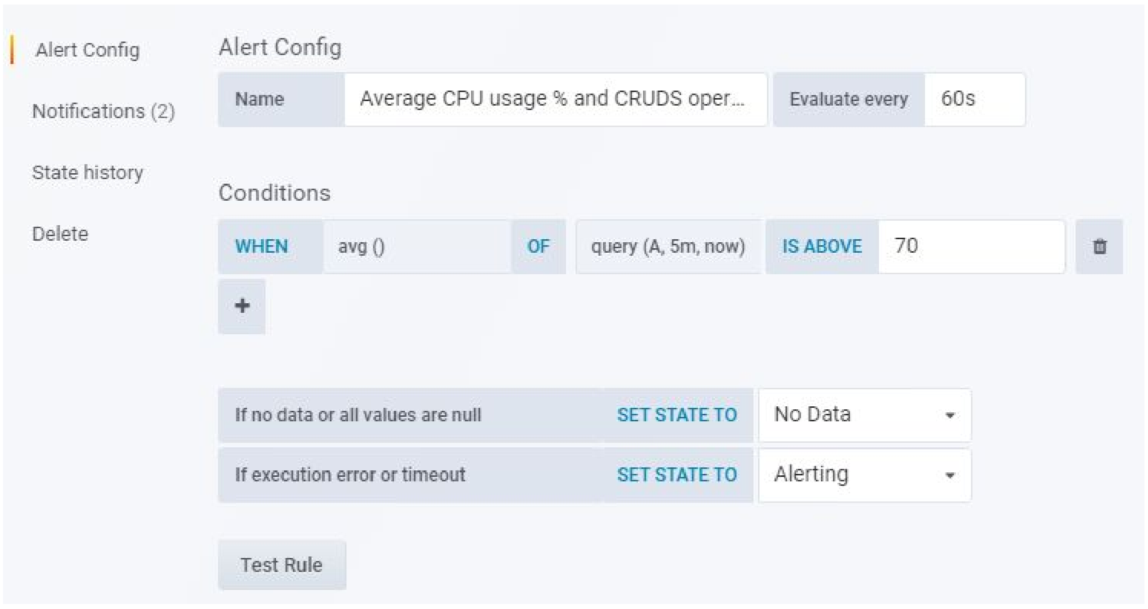
4. Features experimented

4.1 Features explored in Influxdb:

- Since Influxdb does not support joins over multiple measurements, continuous queries can be used to create new measurements and populate it with the relevant information from various measurements in InfluxDb.
  An example of continuous query can be found here.
- Sub queries in InfluxDb is a very useful feature which helps to make visualisation of our ECM data better. An example of a query can be found here. Subqueries helps to do aggregation of two queries in a single query. As an example, in the above query, the data needed is the percentage of retrieved documents exceeding a duration of 70 ms. In order to satisfy the requirements, the first query fetches the number of docs that exceed the 70 ms duration and second query fetches the total number of retrieved docs. In the outermost query, a % of retrieved documents exceeding 70 ms is obtained.

4.2 Features explored in Grafana
Dashboards in Grafana can be grouped into folders for easy access. Each dashboard is a collection of panels. There are many built-in panels provided by Grafana and also third party panel plug-ins that can be installed in Grafana.

- **Alerting:** Some panels also support alerts to be configured based on the queries used in the panels. These alerts can be used to trigger when certain conditions are not met and send message via notification channels. There are many mediums to send alerts. We experimented alerts via the Telegram notification channel. For this purpose,a a chat bot was created in Telegram and separate chat group was created to send alerts to. The chat id was used to configure Grafana to send alert notifications to the specifies chat id.



Alert Config

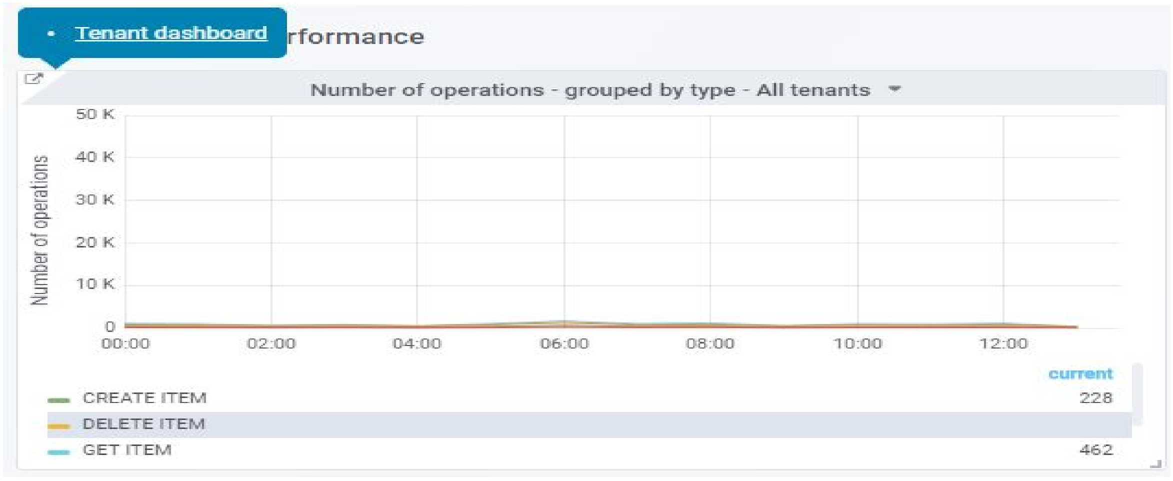
Name: Average CPU usage % and CRUDS oper...
Evaluate every: 60s

Conditions

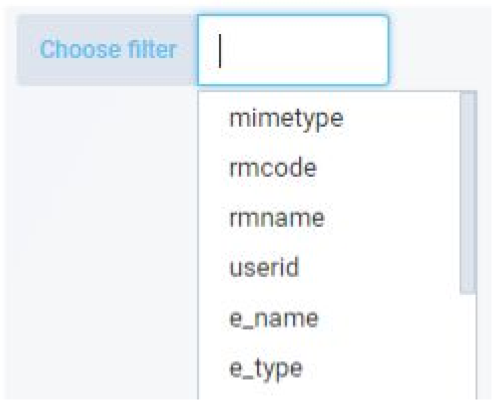
WHEN avg () OF query (A, 5m, now) IS ABOVE 70

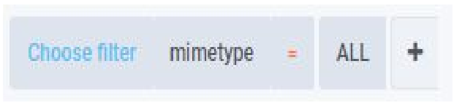
If no data or all values are null SET STATE TO No Data
If execution error or timeout SET STATE TO Alerting

- **Dashboard links:** Allows us to link any panel or external URL to the dashboards. In the example below, Tenant dashboard is linked to the current dashboard and clicking on the link leads us to the Tenant dashboard.

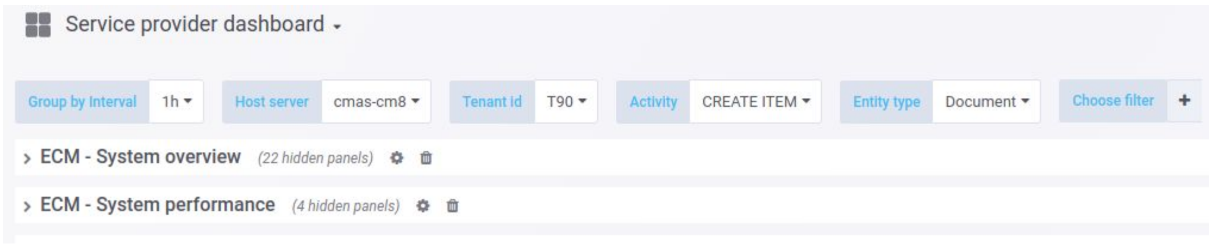


- **Ad-hoc filters**: Allows us to use any number of value/key filters that will be automatically applied to all queries.





- **Template variables:** In order to improve ease of use and quick filtering of data, template variables are provided in Grafana which helps in querying desired results. Value selected in the variable can be used inside the queries to filter the data. They are fixed by default at the top of the dashboard as shown in the figure below.



5. Limitations and Comparisons

5.1 Limitations of Influxdb

- Cannot use aggregators on the time column.
- DISTINCT function operates only on fields, not on tags, and ORDER BY only works against timestamps.
- SQL JOINs aren't available for InfluxDB measurements.
- Time is the constant primary key.

5.2 Limitations of Grafana

- Alerts in grafana cannot be added for queries having template variables.
- Template variables are fixed at the top of a dashboard and hence we cannot move the variables near the corresponding panels

5.3 Comparisons of Grafana and Chronograf

- Both Grafana and Chronograf provide ways to design queries interactively when setting up a panel. Also the visualization types like Gauge come built in as part of Chronograf (1.6) whereas in Grafana we need to manually install a plug in for Gauge panel.

- Grafana offers the ability to add new plug ins like pie chart and bubble chart, whereas in Chronograf there are external no plug ins available. In Chronograf we need to use the graphs/panels which come built in with Chronograf.
- In Grafana there are more options inside each of the categories (Axes,Legend,Display,Alert,Time range) for visualization compared to Chronograf.
- Grafana has the functionality to create dynamic panels based on template variables selected. When there are multiple values selected in a template variable, it will create a panel for each value of the template variable. In Chronograf this feature is not available.
- In Grafana alerts can be configured in graph panels in the Alert tab. Alerts in Chronograf corresponds to Kapacitor tasks. These tasks are stored as TICKscripts that can be edited through Chronograf or manually.
- In Grafana we can send notification alerts for all states in different Notification channels whereas in Chronograf we can customize the alerts by using kapacitor tasks which are TICK scripts. In TICK script, we can use the property method stateChangesOnly() that will issue an alert if the state of alert changes. An example where this feature comes in handy is described below. Say in Grafana an alert is configured to evaluate and send out every 5 minutes and in Chronograf, an alert is configured send out when there is a state change. When the observed metric is critical (exceeding threshold) for longer time period (1 hour), Grafana will alert the user every 5 minutes for the whole duration (1 hour) whereas Chronograf will alert once when the state changes to critical and alert again when the state changes to normal. This way the user will not be spammed with alerts if the system is down for a longer time window.

6 Conclusion

Based on the work conducted above, the components of the open source stack Telegraf, Influxdb and Grafana seems to offer the necessary tools for building a powerful and flexible monitoring tool for monitoring and metering Enterprise Content Management (ECM) Systems. In future, we can do predictive analytics on the time series data to forecast metrics such as workload to monitor the performance of the ECM system.

7 Appendix:

3.1 Service provider dashboard

3.1.1 ECM System overview

Uptime of server: <server name>
SELECT LAST("uptime_format") AS "value" FROM "system" WHERE "host" =~ /$server$/ AND time<=now()

Current logged on users: All tenants
SELECT LAST("loggedon") FROM "autogen"."cm8_users" WHERE ("tenantid" = 'ALL')

CPU usage: <server name>
*SELECT LAST("usage_idle") * -1 + 100 FROM "cpu" WHERE ("host" =~ /^$server$/ AND "cpu" = 'cpu-total') ORDER BY time DESC*

Disk usage: <server name>
SELECT LAST(used) as Used FROM disk WHERE host = 'cmas-cm8' AND path != '/mnt/swrepos' GROUP BY path
SELECT SUM(Free) as Total_Free FROM (SELECT LAST(free) as Free FROM disk WHERE host = 'cmas-cm8' AND path != '/mnt/swrepos' GROUP BY path)

ECM System Used Memory %: <server name>
SELECT LAST("used_percent") AS "Used Memory" FROM "mem" WHERE ("host" =~ /^$server$/)
SELECT LAST("total") FROM "mem" WHERE ("host" =~ /^$server$/) AND $timeFilter
SELECT "used" FROM "mem" WHERE ("host" =~ /^$server$/) AND $timeFilter

SLA
*SELECT SUM(c)/SUM(d) *100 FROM (select COUNT(create_doc) as c from cm8_ops where create_doc>0 and "duration">=70), (SELECT COUNT(create_doc) as d from cm8_ops where create_doc>0) WHERE time >= now() - 30d fill(null)*
*SELECT SUM(c)/SUM(d) *100 FROM (select COUNT(retrieve_doc) as c from cm8_ops where retrieve_doc>0 and "duration">=100), (SELECT COUNT(retrieve_doc) as d from cm8_ops where retrieve_doc>0) WHERE time >= now() - 30d fill(null)*

Average response time of operations
SELECT mean("duration") FROM "autogen"."cm8_ops2" WHERE ("operation" = 'create')
SELECT last("duration") FROM "autogen"."cm8_ops2" WHERE ("operation" = 'create') AND $timeFilter
SELECT mean("duration") FROM "autogen"."cm8_ops2" WHERE ("operation" = 'retrieve') AND time >= now() - 30d
SELECT last("duration") FROM "autogen"."cm8_ops2" WHERE ("operation" = 'retrieve') AND $timeFilter

Logged on users
SELECT LAST("loggedon") FROM "autogen"."cm8_users" WHERE ("tenantid" = 'ALL') and userid='ALL' and $timeFilter
SELECT count("eventdata1") FROM "autogen"."cm8_sysadm_events" WHERE ("eventcode" = '203') AND $timeFilter
SELECT count("eventdata1") FROM "autogen"."cm8_sysadm_events" WHERE ("eventcode" = '204') AND $timeFilter
SELECT last("loggedon") FROM "cm8_users" WHERE ("tenantid" =~ /^$tenantid$/ and "userid" = 'ALL') AND $timeFilter
SELECT COUNT("eventdata1") FROM "autogen"."cm8_sysadm_events" WHERE ("eventcode" = '203' AND "userid" =~ /$tenantid/) AND $timeFilter
SELECT count("eventdata1") FROM "autogen"."cm8_sysadm_events" WHERE ("eventcode" = '204' AND "userid" =~ /$tenantid/) AND $timeFilter

3.1.2 ECM - System performance

Average CPU usage % and CRUDS operations - All Tenants
SELECT mean("usage_user") AS "User Usage " FROM "cpu" WHERE ("host" =~ /^$server$/ AND "cpu" = 'cpu-total') AND $timeFilter GROUP BY time($timeint) fill(null)
SELECT mean("count") FROM "autogen"."cm8_item_events" WHERE ("eventname" = 'ALL' AND "userid" = 'ALL') AND $timeFilter GROUP BY time($timeint) fill(null)

IO wait %, Disk Read/Write rate - All tenants
SELECT mean("usage_iowait") AS "IO wait Usage" FROM "cpu" WHERE ("host" =~ /^$server$/ AND "cpu" = 'cpu-total') AND $timeFilter GROUP BY time($timeint) fill(null)
SELECT non_negative_derivative(mean("read_bytes"), 1s) as "Read rate", non_negative_derivative(mean("write_bytes"), 1s) as "Write rate" FROM "diskio" WHERE ("host" =~ /^$server$/ and "name" =~ /^vd.$/) AND $timeFilter GROUP BY time($timeint), "name" fill(null)

Number of operations - grouped by type - All tenants
SELECT sum("count") FROM "autogen"."cm8_item_events" WHERE ("eventname" != 'ALL') AND $timeFilter GROUP BY time($timeint), "eventname"

Average response time - Create and Retrieve operation
SELECT mean("duration") as "Create" FROM "autogen"."cm8_ops2" WHERE ("operation" = 'create') and $timeFilter GROUP BY time($timeint)
SELECT mean("duration") as "Retrieve" FROM "autogen"."cm8_ops2" WHERE ("operation" = 'retrieve') and $timeFilter GROUP BY time($timeint)

3.1.3 ECM - Access pattern by tenants

Logged on Users - History
SELECT ("loggedon") FROM "autogen"."cm8_users" WHERE ("tenantid" = 'ALL') AND userid = 'ALL' AND $timeFilter   fill(none)
SELECT ("loggedon") FROM "autogen"."cm8_users" WHERE ("tenantid" =~ /^$tenantid$/) AND userid = 'ALL' AND $timeFilter

Workload - by operation
SELECT SUM("count") as "Workload " FROM "autogen"."cm8_item_events" WHERE (("eventname" != 'ALL' AND "eventname" != 'GET ITEM') AND userid = 'ALL') AND $timeFilter GROUP BY time($timeint), "eventname" fill(0)
SELECT SUM("count") as "Workload " FROM "autogen"."cm8_item_events" WHERE (("eventname" != 'ALL' AND "eventname" != 'GET ITEM') AND userid =~ /^$tenantid/) AND $timeFilter GROUP BY time($timeint), "eventname" fill(0)

3.2 Tenant provider dashboard

Users - Log on Status
SELECT last("loggedon") AS "Status" FROM "telegraf"."autogen"."cm8_users" WHERE time > now() - 15m AND "tenantid"='$tenantid' AND "loggedon"=1 GROUP BY userid FILL(none)

Workload - User
SELECT SUM("count") FROM "autogen"."cm8_item_events" WHERE ("eventname" != 'ALL' AND "eventname" != 'GET ITEM' AND "userid" = '$userid') AND $timeFilter GROUP BY time(1s), "eventname" fill(none)

5. Features experimented

5.1 Features explored in Influxdb:

Continuous query

CREATE CONTINUOUS QUERY test_overview_t01 ON telegraf BEGIN SELECT mean(count) AS mean_count_t01 INTO telegraf.autogen.test_overview FROM telegraf.autogen.cm8_doc_type WHERE id_tenant =~ /T01/ AND e_type =~ /Document/ AND time > now() - 15m GROUP BY time(5m) END

Sub query

*SELECT SUM(c)/SUM(d) *100 FROM (select COUNT(retrieve_doc) as c from cm8_ops where retrieve_doc>0 and "duration">=70), (SELECT COUNT(retrieve_doc) as d from cm8_ops where retrieve_doc>0) WHERE time >= now() - 30d fill(null)*

References:

1. https://docs.influxdata.com/influxdb/v1.6/
2. http://docs.grafana.org/features/datasources/influxdb/
3. https://github.com/influxdata/docs.influxdata.com
4. https://docs.influxdata.com/chronograf/v1.6/introduction/getting-started/