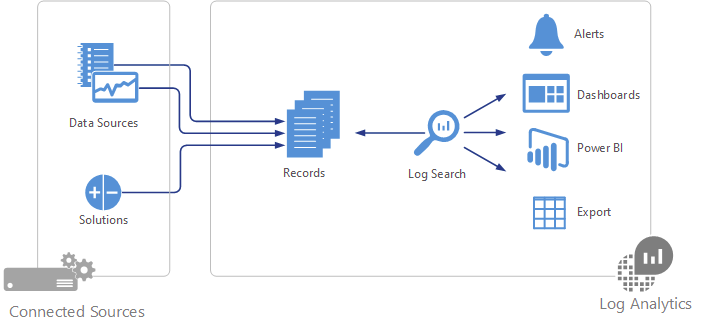
**Production Log Analytics**

* **Log Analytics :**

Log Analytics is the service provided by the Microsoft Azure for the analysis of the log files.

****

* **Connected Sources:**

Connected sources are the computers and other resources that generate data collected by Log Analytics. This can include agents installed on [Windows](https://docs.microsoft.com/en-us/azure/log-analytics/log-analytics-windows-agent) and [Linux](https://docs.microsoft.com/en-us/azure/log-analytics/log-analytics-linux-agents) computers that connect directly or agents in a connected System Center Operations Manager management group.

* **Data Sources:**

Data sources are the different kinds of data collected from each connected source.

For example: Custom log files from the /var/log/audit/ folder.

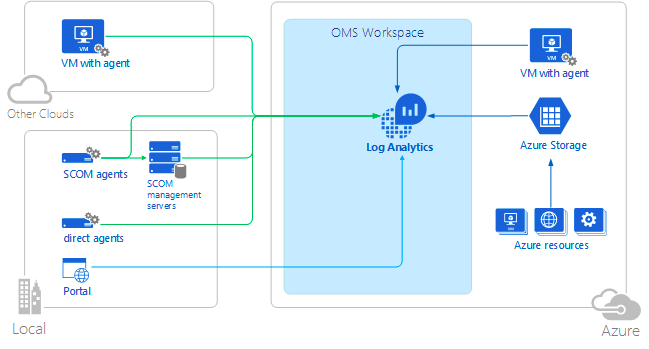
* **Service Provided :**
* Analysis of the contents of the log file
* Customized log file and customize fields
* Execute Queries and save them for future use
* Chart and Design view of particular log analysis
* Dashboards

* **Requirements :**
* Log Files has to be specifically in /var/log/audit/
* Log file has to be in .txt file

i.e /var/log/audit/log\*.txt

* Oms Agent has to be downloaded in your linux machine

Oms Agent is the middle man between the log files generated in the machine and the azure log analytics workspace.



* **Steps for setting up the log analytics:**
* Login to [Azure Portal](https://portal.azure.com/)
* Create Workspace (viz already created as **”*devloganalyticspoc*”** )
* **Steps for the Installation of Omsagent in the ubuntu machine:-**
* $> wget https://raw.githubusercontent.com/Microsoft/OMS-Agent-for-Linux/master/installer/scripts/onboard\_agent.sh && sh onboard\_agent.sh -w <YOUR OMS WORKSPACE ID> -s <YOUR OMS WORKSPACE PRIMARY KEY>

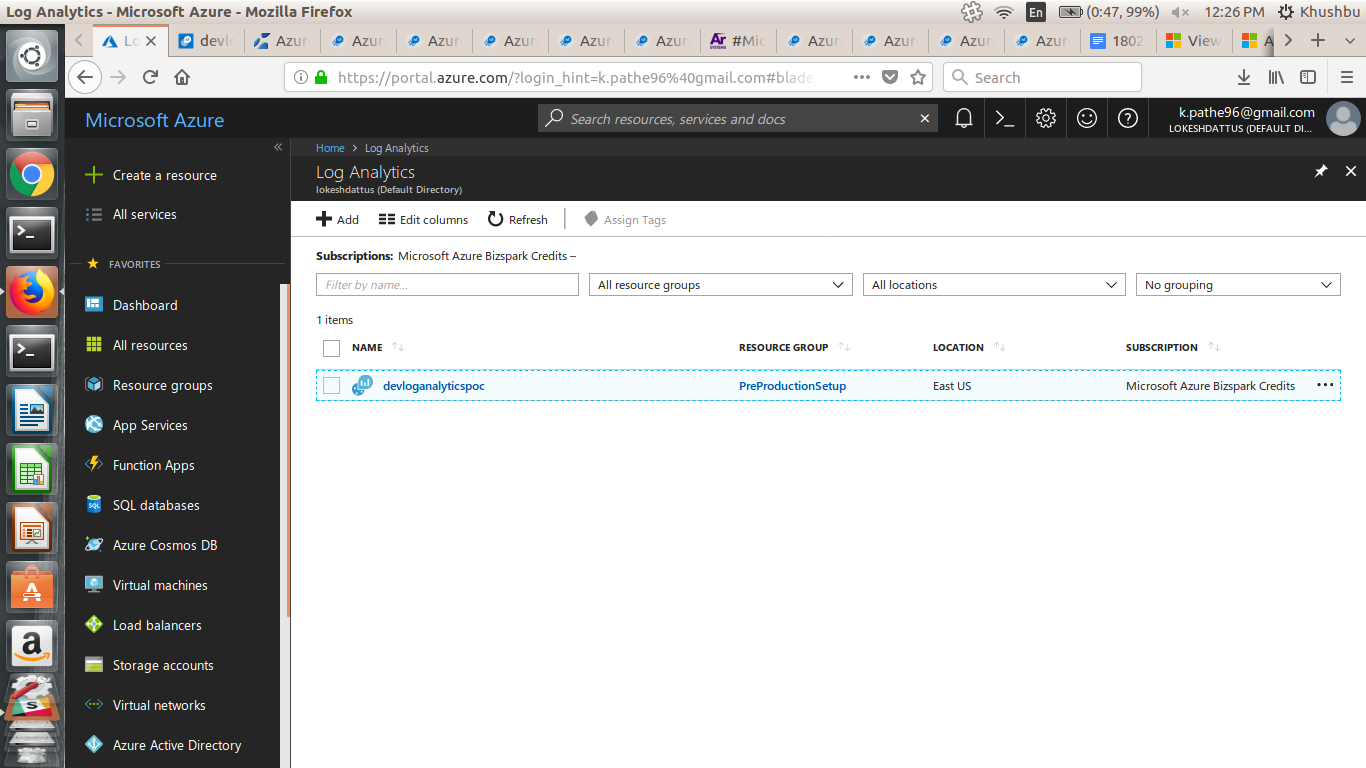
Here:

Our Oms Workspace ID: 59f2d5e7-6346-46e4-93ee-74549869ccaa

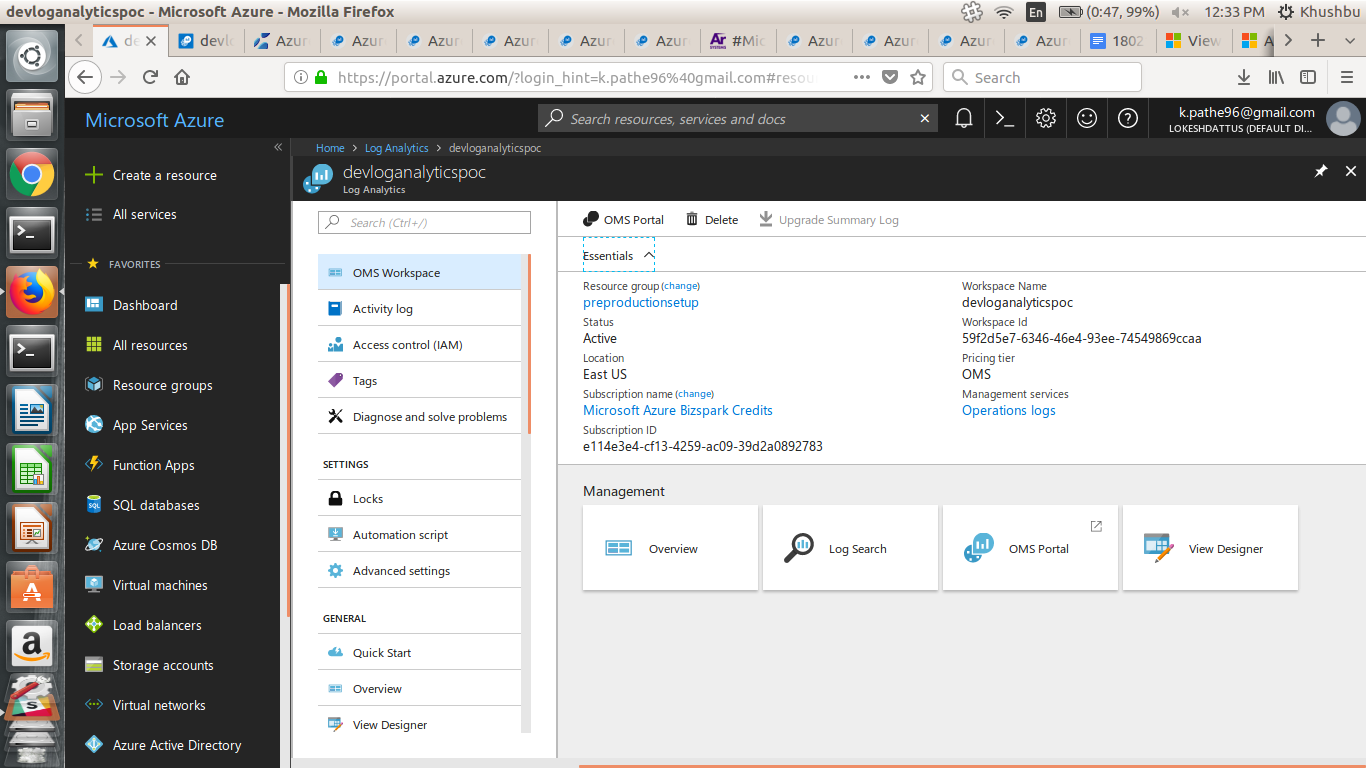
Our Oms Workspace Primary Key: -s UogJphlNtSpTeTzBllE761ns1l7RNOL3SEwgQYo8k/sOj9z+UBpofAnNz1KkNU9cLeoEUk+pM9RgYGDgBWL87A==

So the command goes as,

* wget https://raw.githubusercontent.com/Microsoft/OMS-Agent-for-Linux/master/installer/scripts/onboard\_agent.sh && sh onboard\_agent.sh -w 59f2d5e7-6346-46e4-93ee-74549869ccaa -s UogJphlNtSpTeTzBllE761ns1l7RNOL3SEwgQYo8k/sOj9z+UBpofAnNz1KkNU9cLeoEUk+pM9RgYGDgBWL87A==
* **Connecting our VM with the log analytics:**
* On Selecting the log analytics workspace (illustrated in next point), in the left bar of the workspace you will find several options, scroll down and select ***“virtual machines”***
* Select the VM which you want to connect and select connect. (If your VM doesnot have any configuration issue then it will get connected within seconds but if this isn’t the case then you might have to wait forever to get it connected and see an error)
* If the VM is not connected then you have to install the oms agent by your own for the particular VM
* **Steps involved in using the service :**
* Login to the azure portal
* Select ***“more services”*** in the left bottom side.
* Search in search bar as ***“Log Analytics”***
* Select **”*devloganalyticspoc*”** from the listed workspaces (we will have only one listed i.e **”*devloganalyticspoc*”**)



* Once you are done with it, you will get the below window



* **Query Language for Azure Log Analytics:-**
* All queries can be written in Log search
* Log search is a space provided to execute queries
* We use directly the oms portal log search
* Azure Log analytics query language is case sensitive
* The query always starts with the *Table\_name* or *search \**
* Table\_name is the column name of the particular field
* Here, **|** works as comma

For example: search \* | where (Type==”Log\_CL”)

* Aggregation Functions work as functions defined in normal Sql language
* **Description of all the saved queries in the oms portal**

***Description goes as follows : Name of the query saved as, then description and then the query***

* **all\_data\_from\_Log\_production**

This query shows up all the data from the dataproducer.txt file which is stored as Log\_CL in the azure log analytics. “**\_CL**” is automatically attached to the file name. Type is the field used to specify the name of the file.

search \*

| where(Type == "Log\_CL")

* **data\_before\_24\_hrs**

This query shows up all the data which is received before 24 hrs from the query is hit. TimeGenerated is the new element used here. TimeGenerated will always show the time when the data was received by the azure log analytics. If you wish to see the data for for any different time, just change the time: ***h*** for hr, ***m*** for minutes, ***s*** for seconds and ***d*** for days.

search \*

| where(Type == "Log\_CL")

| where TimeGenerated > ago(24h)

* **data\_with\_heartbeat\_before\_24\_hrs**

This query shows up all the data where heartbeat was generated before 24 hrs from query is hit. Here onwards, custom fields comes in picture.

***Custom fields :***

The data from the log file is a raw data, we need to customize the raw data as per our requirements. We need to pick all the required “words” such as HeartbeatImpl, Mac address, time generated, CPU usage, Ram Usage, Disk usage etc. Here HeartbeatImpl is customized as heartbeat, \_CF is automatically attached to the custom fields selected, here it goes as heartbeat\_CF.

search \*

| where(Type == "Log\_CL")

| where TimeGenerated < ago(24h)

| where (heartbeat\_CF == "HeartbeatImpl")

* **all\_data\_which\_didnot \_send\_heartbeat\_for\_last\_24\_hrs**

This query will return all the data that does not received heartbeat for last 24 hrs. If you want to see data for no heartbeat for overall log file just erase TimeGenerated ro from the following query.

search \*

| where ( Type == "Log\_CL" )

| where TimeGenerated > ago(1d)

| where (heartbeat\_CF != "HeartbeatImpl")

* **summary\_of\_data\_not\_sent\_for\_24\_hrs**

This query will return the list of all the mac id which did not send heartbeat for the past 1 day i.e 24 hrs**.** Now summarize function comes in picture, with the “**makelist**” function. **Summarize** is used to get all the mac address together and makelist is used to make a list of all the mac address which are brought together by the summarize function.

search \*

| where ( Type == "Log\_CL" )

| where TimeGenerated > ago(1d)

| where (heartbeat\_CF != "HeartbeatImpl")

| summarize makelist(mac\_CF) by heartbeat\_CF

* **last\_updated\_record\_by\_each\_hub**

This query will return the list of mac address with their latest updated record. The arg\_max function returns the max time generated.

Log\_CL

| summarize arg\_max(TimeGenerated, \*) by mac\_CF

* **hubs\_which\_didnot\_send\_heartbeat\_latest**

This query returns the mac address of hubs which did not send heartbeat for the latest TimeGenerated.

Log\_CL

| summarize arg\_max(TimeGenerated, \*) by mac\_CF

| where (heartbeat\_CF != "HeartbeatImpl")

* **active\_but\_didnt\_send\_heartbeat**

This query returns the hubs which were active before 24 hrs but didn’t send heartbeat in last 1 hr

Log\_CL

| where TimeGenerated > ago(1d)

| summarize LastHeartbeat = max(TimeGenerated) by mac\_CF,heartbeat\_CF

| where isnotempty(mac\_CF)

| where LastHeartbeat < ago(1h)

* **count\_of\_user\_login**

This query returns the total count of users logged in into the system.

search \*

| where Type == "Log\_CL"

| where isnotempty(user\_CF)

| summarize count(user\_CF) by user\_CF, TimeGenerated,login\_status\_CF

* **Availability Rate of all the computers connected to the azure log analytics via oms agent**

This query returns the availability of the computer connected to the azure log analytics .

let midnight=startofday(now());

Heartbeat

| where TimeGenerated>midnight

| summarize heartbeat\_per\_hour=count() by bin\_at(TimeGenerated, 1h, midnight), Computer

| extend available\_per\_hour=if(heartbeat\_per\_hour>0, true, false)

| summarize total\_available\_hours=countif(available\_per\_hour==true) by Computer

| extend number\_of\_buckets=hourofday(now())+1

| extend availability\_rate=total\_available\_hours\*100/number\_of\_buckets

* **cpu\_utilization\_per\_minute**

This query returns the list of the Computer whose Process time has exceeded 80%.

search \*

| where (Type == "Perf")

| where CounterName == "% Processor Time"

| summarize AggregatedValue = avg(CounterValue) by bin(TimeGenerated, 1m), Computer

* **available\_mbytes**

This query returns the available memory in Megabytes.

search \*

| where (Type == "Perf")

| where CounterName == "Available MBytes Memory"

| summarize AggregatedValue = avg(CounterValue) by bin(TimeGenerated, 1m), Computer

* **cpu\_usage\_timechart**

This query visualizes the timechart of CPU utilization.

search \*

| where (Type == "Perf")

| where TimeGenerated > ago(5h)

| where CounterName == "% Processor Time"

| summarize CPU\_Utilization = avg(CounterValue) by Computer, bin(TimeGenerated, 15m)

| render timechart

* **free\_disk**

This query returns the free space on logical disk.

search \*

| where ( Type == "Perf" )

| where ( ObjectName == "Logical Disk" )

| where ( CounterName == "Free Megabytes" )

| summarize FreeMegabytes = min(CounterValue) by Computer

| sort by FreeMegabytes asc

| render piechart

* **nilam\_login\_count\_past\_24\_hrs**

This query returns the total number of logins done by Nilam Nikam in last 24 hrs.

search \*

| where ( Type == "Log\_CL" )

| where (Computer == "TestManagement")

| where TimeGenerated > ago(24h)

| summarize LoginCount = countif(user\_CF == "nnikam" or user\_CF == "nnikam@dattus.com") by TimeGenerated, Timestamp\_CF

| render piechart

* **nilam\_login\_count\_past\_24\_hrs**

This query returns the total number of logins done by Nilam Nikam in last 24 hrs nilam2 ID

search \*

| where ( Type == "Log\_CL" )

| where (Computer == "TestManagement")

| where TimeGenerated > ago(24h)

| summarize LoginCount = countif(user\_CF == "nilam2" or user\_CF == "nilam2@dattus.com") by TimeGenerated, Timestamp\_CF

| render piechart

* **nilam\_login\_count\_past\_24\_hrs**

This query returns the total number of logins done by Nilam Nikam in last 24 hrs with nilam1 ID

search \*

| where ( Type == "Log\_CL" )

| where (Computer == "TestManagement")

| where TimeGenerated > ago(24h)

| summarize LoginCount = countif(user\_CF == "nilam1" or user\_CF == "nilam1@dattus.com") by TimeGenerated, Timestamp\_CF

| render piechart

* **admin\_login\_count\_past\_24\_hrs**

This query returns the total number of logins done by Nilam Nikam in last 24 hrs with admin ID

search \*

| where ( Type == "Log\_CL" )

| where (Computer == "TestManagement")

| where TimeGenerated > ago(24h)

| summarize LoginCount = countif(user\_CF == "admin" or user\_CF == "admin@dattus.com") by TimeGenerated, Timestamp\_CF

| render piechart

* **sushil\_login\_count\_past\_24\_hrs**

This query returns the total number of logins done by Sushil Taskar in last 24 hrs.

search \*

| where ( Type == "Log\_CL" )

| where (Computer == "TestManagement")

| where TimeGenerated > ago(24h)

| summarize LoginCount = countif(user\_CF == "sushil.taskar" or user\_CF == "sushil.taskar@dattus.com") by TimeGenerated,Timestamp\_CF

* **pradeep\_login\_count\_past\_24\_hrs**

This query returns the total number of logins done by Pradeep Surale in last 24 hrs.

search \*

| where ( Type == "Log\_CL" )

| where (Computer == "TestManagement")

| where TimeGenerated > ago(24h)

| summarize LoginCount = countif(user\_CF == "pradeep.surale" or user\_CF == "pradeep.surale@dattus.com") by TimeGenerated,Timestamp\_CF

* **mayur\_login\_count\_past\_24\_hrs**

This query returns the total number of logins done by Pradeep Surale in last 24 hrs.

search \*

| where ( Type == "Log\_CL" )

| where (Computer == "TestManagement")

| where TimeGenerated > ago(24h)

| summarize LoginCount = countif(user\_CF == "pradeep.surale" or user\_CF == "pradeep.surale@dattus.com") by TimeGenerated,Timestamp\_CF

* **swapneel\_login\_count\_past\_24\_hrs**

This query returns the total number of logins done by Swapneel Datta in last 24 hrs.

search \*

| where ( Type == "Log\_CL" )

| where (Computer == "TestManagement")

| where TimeGenerated > ago(24h)

| summarize LoginCount = countif(user\_CF == "swapneel.datta" or user\_CF == "swapneel.datta@dattus.com") by TimeGenerated,Timestamp\_CF

* **total\_count\_of\_login\_in\_last\_24hrs**

This query returns the total count of logins done by the Authorised users in last 24 hrs.

search \*

| where ( Type == "Log\_CL" )

| where (Computer == "TestManagement")

| where TimeGenerated > ago(24h)

| summarize LoginCount = count(user\_CF) by user\_CF, Timestamp\_CF, Computer, Authorised\_CF