Infrastructure Analytics - Understanding the need and scope

- **Manasee Godsay, Volunteer/Intern**

Need for log data analysis: What is log data and why is it important?

Data logging is a process that is crucial to understanding the health of your system, and in the age of cloud computing, these logs are your key resources, when you are tracking what went wrong or for that matter, what could be done in a better way. Understanding and troubleshooting network and deployment issues often starts with log data analysis.

Data logging enables the recording of activity performed on one or more data/file objects or sets. Typically data logging records events/actions, such as the data's size, most recent modification and username/name of the individual that modified the data.

Data logging also facilitates the storage and collection of computer or device information. For example, data logging can store processor temperature and memory utilization over time and network bandwidth usage. System/network administrators use this data to analyze system or network performance during a specific period.

Logs are a commonly used source of data to track, verify, and diagnose the state of a system. Regarding the three pillars of observability, logs typically offer the richest context and are the closest to the source of an event.

What are we looking for in the log data - anomalies, trends?

Log data, essentially could be logs from single threads or multiple threads, in case of concurrency. Vm logs or kubernetes logs basically describe the status of the clusters. Application and systems logs can help you understand what is happening inside your cluster. The logs are particularly useful for debugging problems and monitoring cluster activity. Most modern applications have some kind of logging mechanism; as such, most container engines are likewise designed to support some kind of logging. The easiest and most embraced logging method for containerized applications is to write to the standard output and standard error streams.

However, the native functionality provided by a container engine or runtime is usually not enough for a complete logging solution. For example, if a container crashes, a pod is evicted, or a node dies, you’ll usually still want to access your application’s logs. As such, logs should have a separate storage and lifecycle independent of nodes, pods, or containers. This concept is called cluster-level-logging. Cluster-level logging requires a separate backend to store, analyze, and query logs. Kubernetes provides no native storage solution for log data, but you can integrate many existing logging solutions into your Kubernetes cluster. (Kubernetes documentation)

Anomaly detection in logs is a crucial step in building a secure and trustworthy software. As systems are getting complex, the attacks are getting sophisticated. The anomalies in log data could be of two types, performance anomalies(ex: long arrival of log entries, depends on timestamp) and parameter anomalies(ex: a log entry with a much longer VM creation time than others).

Infrastructure analytics - existing approaches:

Existing infrastructure analytics techniques psan over supervised methods and unsupervised methods. Unsupervised methods such as PCA and clustering based approaches are used for finding trends in log data, as the errors cannot be explicitly classified due to their unpredictability or varied natures. systems that perform anomaly diagnosis using system logs include DISTALYZER that diagnoses system performance issues by comparing a problematic log against a normal one, LogCluster which clusters and organizes historical logs to help future problem identification, and Stitch that extracts different levels of identifiers from system logs and builds a web interface for users to visually monitor the progress of each session and locate performance problems. Note that they are for diagnosis purposes once an anomaly has been detected, and cannot be used for anomaly detection itself.

Who would benefit?

Software developers using kubernetes, VM logs, any systems generating log files, network troubleshooters would benefit from the log files analytics/infrastructure analytics. This would make them well prepared to deal with anomalies and face attacks such as DDOS(Distributed Denial of Service).

Data source for the sample analysis:

Open stack log data is used for sample analysis and exploratory data analysis to understand the scope of the data and where the insights could lead us to.

OpenStack is a free and open-source software platform for cloud computing, mostly deployed as infrastructure-as-a-service (IaaS), whereby virtual servers and other resources are made available to customers. The software platform consists of interrelated components that control diverse, multi-vendor hardware pools of processing, storage, and networking resources throughout a data center. Users either manage it through a web-based dashboard, through command-line tools, or through RESTful web services.

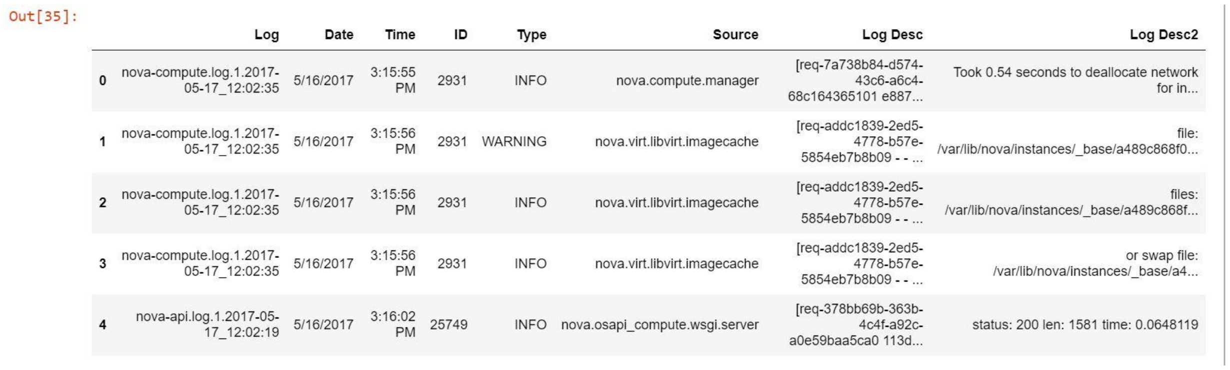
OpenStack (https://www.openstack.org) is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter. This dataset was generated on CloudLab, a flexible, scientific infrastructure for research on cloud computing. Both normal logs and abnormal cases with failure injection are provided, making the data amenable to anomaly detection research.

Analysis done: visualizations and trends

The features that we have in the sample data are: Log, Date, Time, Source, Log Desc, Log Desc2. The number of rows in sample dataset are 180047. The columns from the openstack data were merged after being converted to csv file to form log desc and log desc2 which are columns that portray the log information recorded for the VM logs.

The exploratory data analysis involves looking at graphs for log types, source of the logs and visualising the data by date and time. The log ID is the same for a session of VM logs.

This is how the data looks like:



Out[35]:

	Log	Date	Time	ID	Type	Source	Log Desc	Log Desc2
0	nova-compute.log.1.2017-05-17_12:02:35	5/16/2017	3:15:55 PM	2931	INFO	nova.compute.manager	[req-7a738b84-d574-43c6-a6c4-68c164365101 e887...	Took 0.54 seconds to deallocate network for in...
1	nova-compute.log.1.2017-05-17_12:02:35	5/16/2017	3:15:56 PM	2931	WARNING	nova.virt.libvirt.imagecache	[req-addc1839-2ed5-4778-b57e-5854eb7b8b09 - - ...	file: /var/lib/nova/instances/_base/a489c868f0...
2	nova-compute.log.1.2017-05-17_12:02:35	5/16/2017	3:15:56 PM	2931	INFO	nova.virt.libvirt.imagecache	[req-addc1839-2ed5-4778-b57e-5854eb7b8b09 - - ...	files: /var/lib/nova/instances/_base/a489c868f...
3	nova-compute.log.1.2017-05-17_12:02:35	5/16/2017	3:15:56 PM	2931	INFO	nova.virt.libvirt.imagecache	[req-addc1839-2ed5-4778-b57e-5854eb7b8b09 - - ...	or swap file: /var/lib/nova/instances/_base/a4...
4	nova-api.log.1.2017-05-17_12:02:19	5/16/2017	3:16:02 PM	25749	INFO	nova.osapi_compute.wsgi.server	[req-378bb69b-363b-4c4f-a92c-a0e59baa5ca0 113d...	status: 200 len: 1581 time: 0.0648119

Fig. 1: Data head

The logID counts shows domination of 2 count IDs - 2931,25749:

```
In [39]:
data['ID'].value_counts()
data['ID'].value_counts().plot('bar')
plt.show()
```

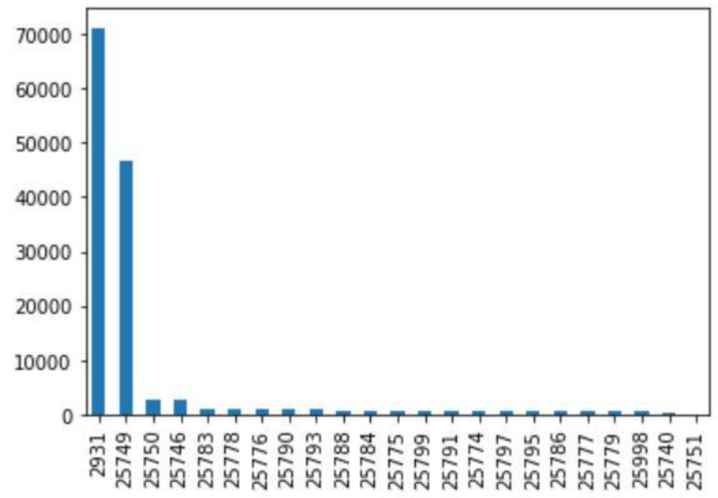


Fig. 2: ID counts bar graph

The source of the log files shows domination of server and image cache:

```
In [40]: data['Source'].value_counts()
         data['Source'].value_counts().plot('bar')
         plt.show()
```

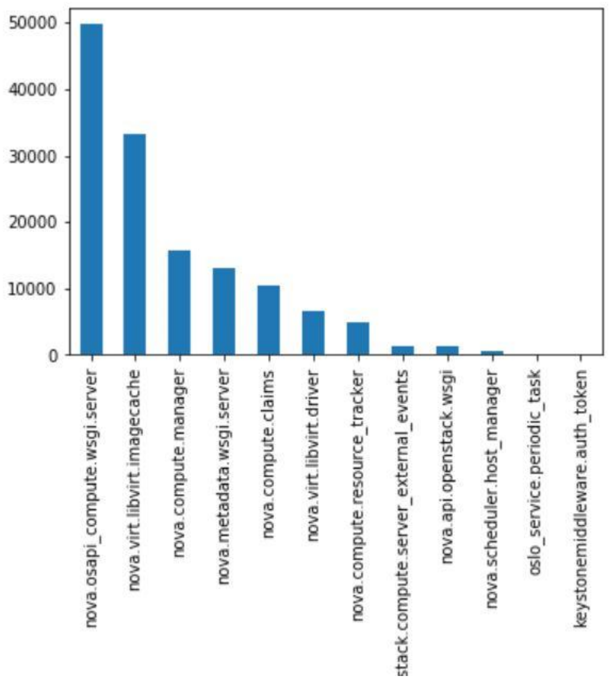
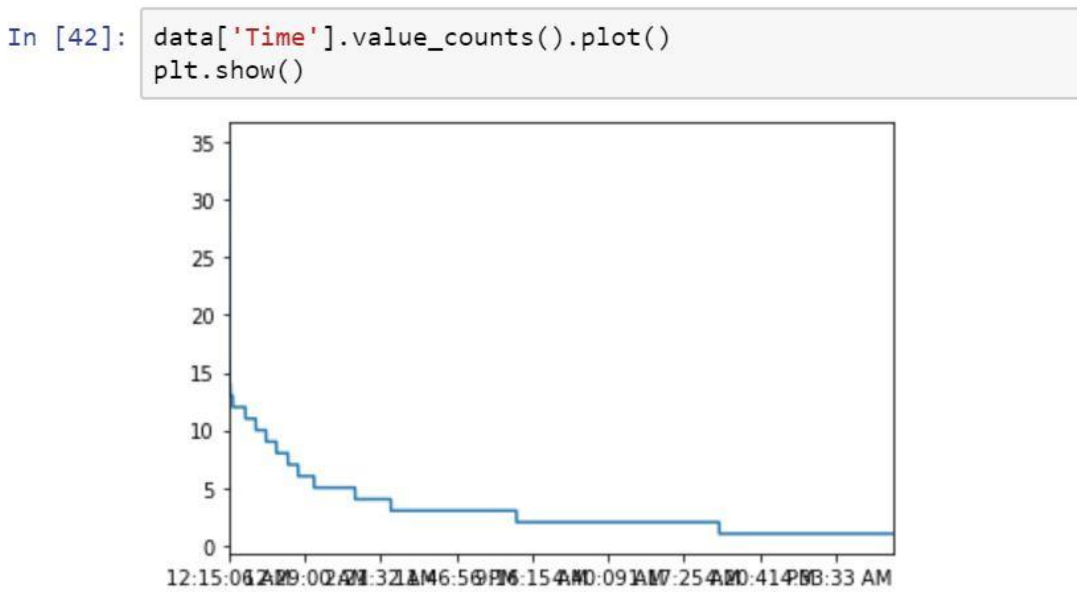


Fig. 3: Source counts graph

The following graph shows the distribution over time for the logs in the dataset, they are recorded for 2 dates - 5/16/2017: 71069 instances, 5/17/2017: 66005 instances:



```
In [42]: data['Time'].value_counts().plot()
         plt.show()
```

Fig. 4: Highest counts instances

The distribution of the data based on ID shows two peaks because of the dominance of the IDs 2931 and 25749:

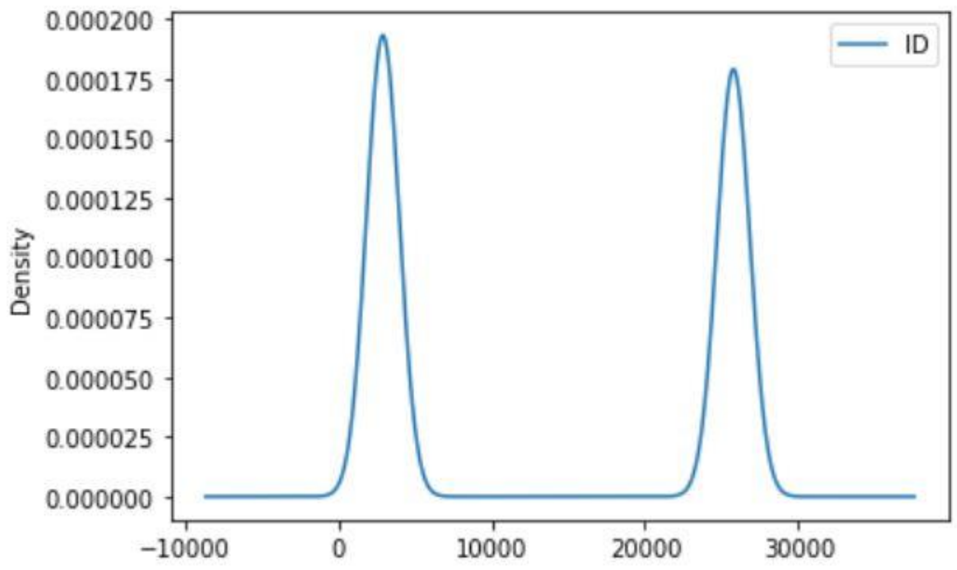


Fig. 5: Distribution of ID

Natural language processing for the log description:

A word cloud of the log desc2 column shows the following output. It is based on the count of words that occur in all the rows:

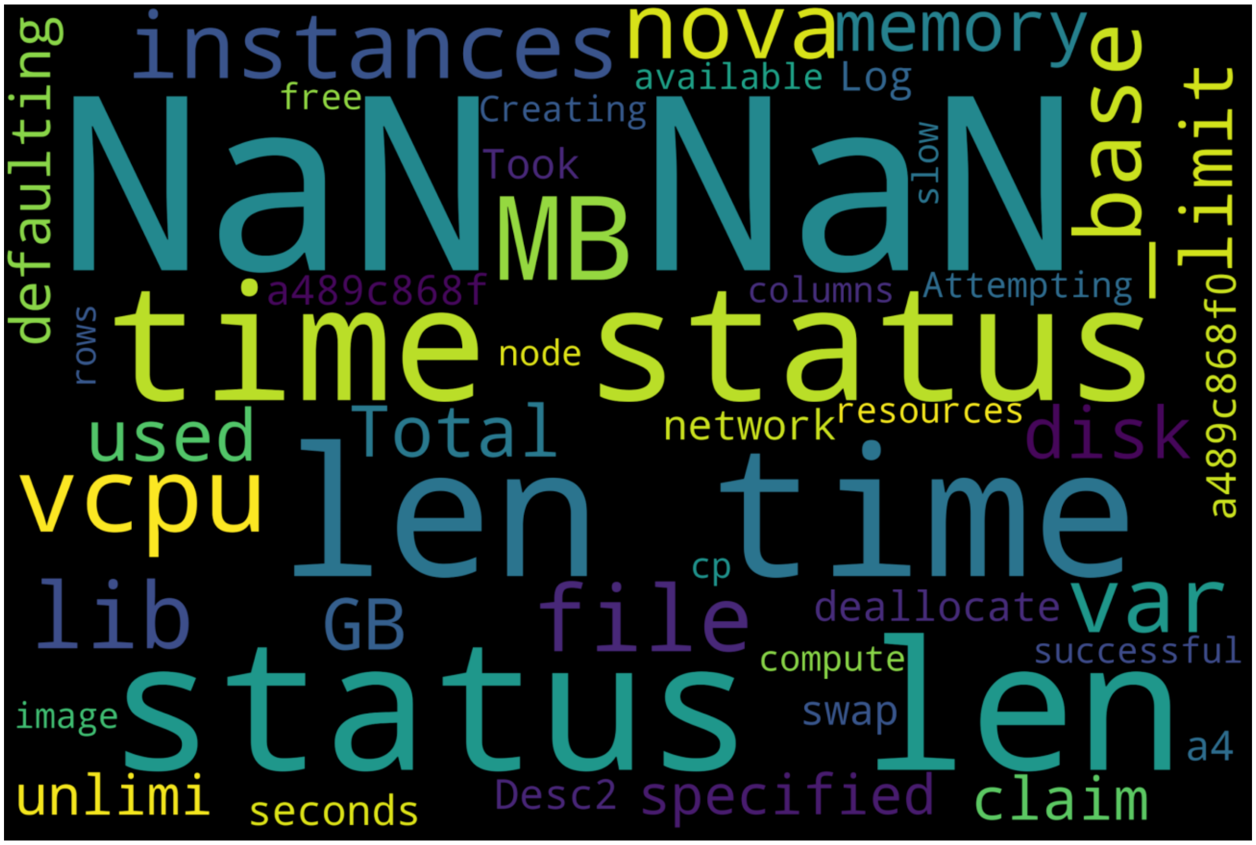


Fig. 6: Word cloud for log Desc2

Challenges with log data:

Log data is unstructured and the semantics vary from system to system. The existing few methods are rule-based. Certain actions need to be taken when certain situation arises, thus, requiring domain knowledge. Also, there are times, that decisions are to be made in streaming fashion, if the anomalies are to be detected and DOS like attacks are needed to be dealt with.

Another challenge comes from concurrency. The order of log messages can provide useful insights(ex: identifying execution path of a program), but this might be at different intervals for log messages produced by several different threads.

Future scope:

This study could be extended to applying the use of log data to predict anomalies and attacks when the logs are analysed in real time. This could be done with neural networks, specific type of RNN(Recurrent Neural Networks) as implemented in DeepLog. (https://acmccs.github.io/papers/p1285-duA.pdf)

References/Citations:

- Jieming Zhu, Shilin He, Jinyang Liu, Pinjia He, Qi Xie, Zibin Zheng, Michael R. Lyu. Tools and Benchmarks for Automated Log Parsing. *International Conference on Software Engineering (ICSE)*, 2019.
- Min Du, Feifei Li, Guineng Zheng, Vivek Srikumar. DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning, in Proc. of ACM Conference on Computer and Communications Security (CCS), 2017.
- Qingwei Lin, Hongyu Zhang, Jian-Guang Lou, Yu Zhang, and Xuewei Chen. 2016. Log clustering based problem identification for online service systems. In Proc. International Conference on Soware Engineering (ICSE). 102–111.
- Xu Zhao, Kirk Rodrigues, Yu Luo, Ding Yuan, and Michael Stumm. 2016. Nonintrusive performance proling for entire software stacks based on the flow reconstruction principle. In Proc. USENIX Symposium on Operating Systems Design and Implementation (OSDI). 603–618.
- Karthik Nagaraj, Charles Killian, and Jennifer Neville. 2012. Structured comparative analysis of systems logs to diagnose performance problems. In Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI). 26–26
- https://timber.io/blog/collecting-application-logs-on-kubernetes/
- https://www.techopedia.com/definition/596/data-logging
- https://en.wikipedia.org/wiki/OpenStack
- https://kubernetes.io/docs/concepts/cluster-administration/logging/