SDC 20
SNIA INDIA

Storage Developer Conference
December 4-5, 2020
BY Developers FOR Developers

AI/ML Based Customer IO Workload Simulation

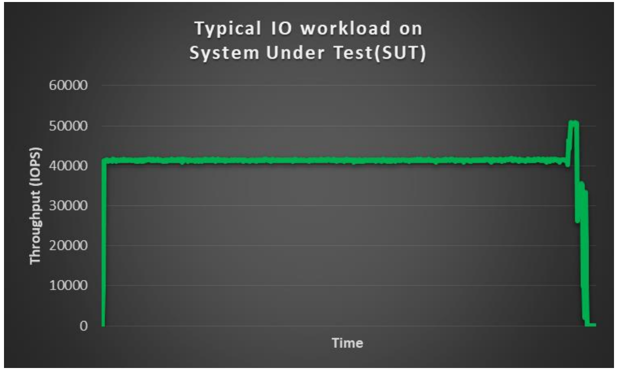
Susanta Dutta
Hewlett Packard Enterprise

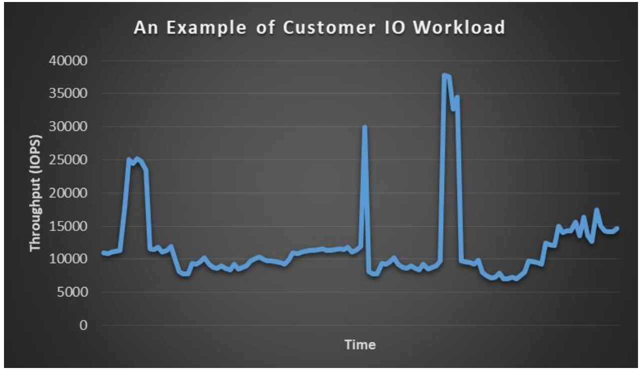
Agenda

- Objective
- Challenges
- ML Model
- Simulation
- Summary

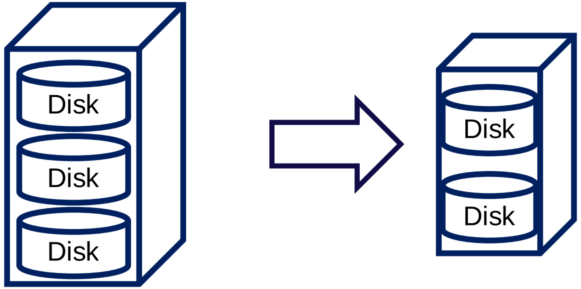
Objective

- Test environment plays an important role in effective testing, thus reducing Customer Found Issues (CFIs)
- Simulation of customer like environment, especially the IO workload pattern for non-functional requirements, is an important aspect to consider.
- Typical IO Workload in test environment vs. customer environment

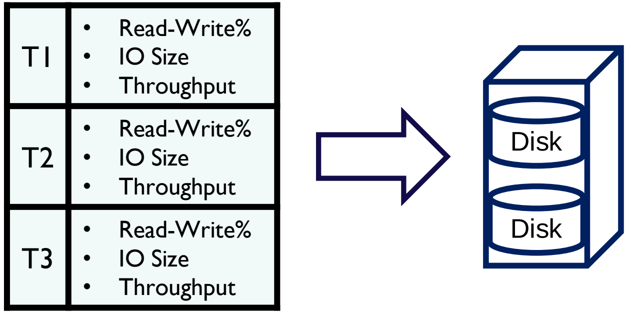




Challenges



Configuration Incompatibility : IO Performance depends on multiple factors – Storage Array configuration is one of them. Source array configuration most likely not the same as array under test.



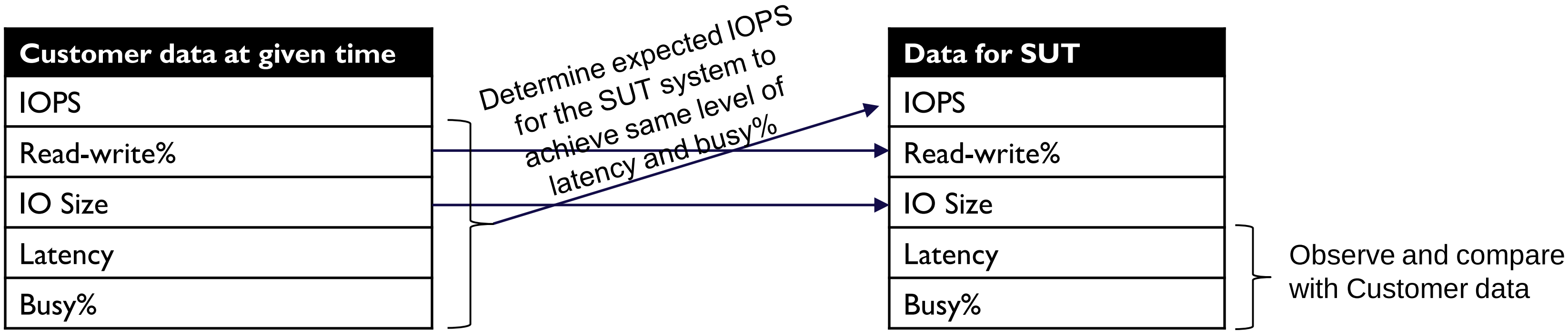
T1	Read-Write% IO Size Throughput
T2	Read-Write% IO Size Throughput
T3	Read-Write% IO Size Throughput

IO Workload Replay : Customer storage system performance varies along with time and tool must be capable of replaying the workload along with time.

Workload Completeness : There are many factors than just one or more parameters of the workload characteristics does not make simulation complete.

Approach

- Get System Under Test(SUT) storage array in similar busy state as that of customer array for each data sample.
- Instead of using IOPS directly from customer performance data, determine expected IOPS for the SUT system for given Read-Write%, IO Size, Latency and Busy% from Customer data.



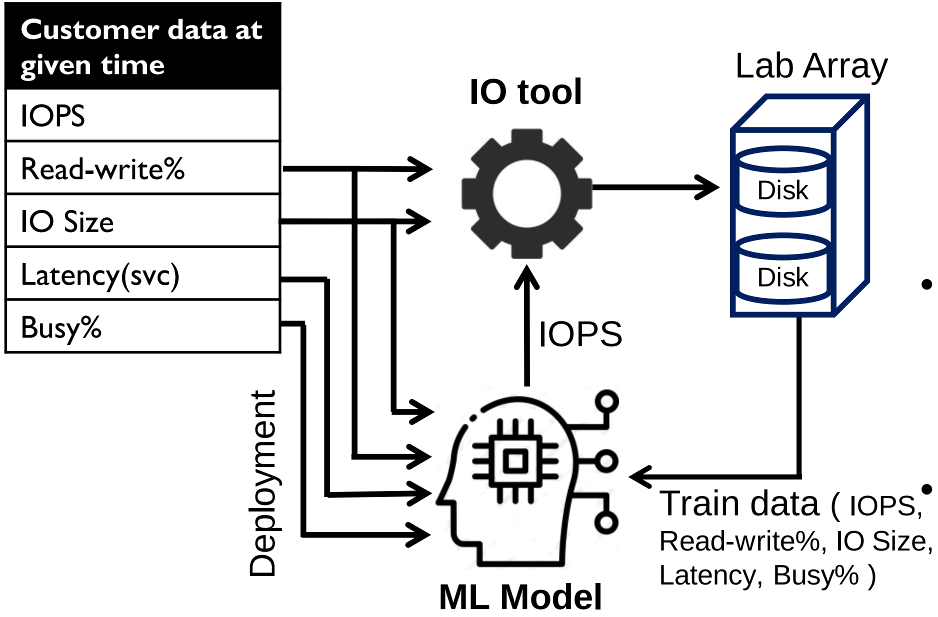
Customer data at given time
IOPS
Read-write%
IO Size
Latency
Busy%

Determine expected IOPS for the SUT system to achieve same level of latency and busy%

Data for SUT
IOPS
Read-write%
IO Size
Latency
Busy%

Observe and compare with Customer data

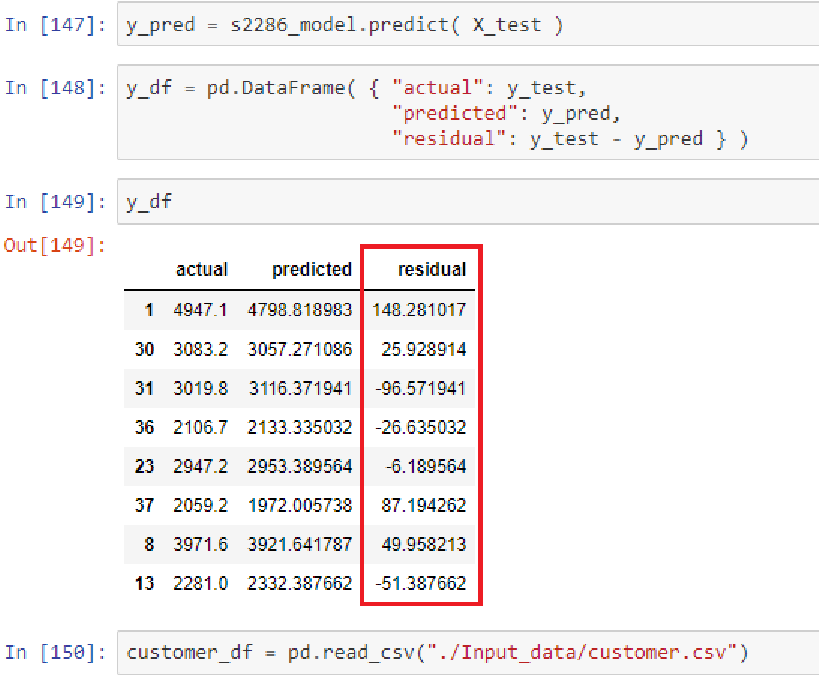
Machine Learning (ML) model



IOPS = f (Read-write%, IO Size, Latency, Busy%)

- **Step1**: To match load similar to source array on a different configuration, first ran different workloads on lab array to build ML model using standard python modules.
- **Step2**:Once the model is built, used customer performance data to determine required IOPS for SUT system.
- **Step3**: Generated workload with all identified parameters on to SUT - Read-write%, IO Size, along with ML estimated IOPS.
- **Step4**: Repeat same with each data sample for all time slots.

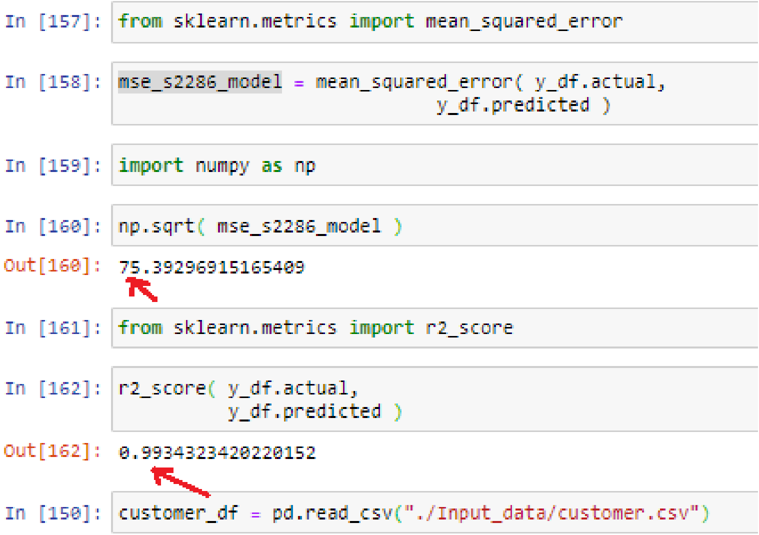
Accuracy of the ML Model



```
In [147]: y_pred = s2286_model.predict( X_test )

In [148]: y_df = pd.DataFrame( { "actual": y_test,
                                 "predicted": y_pred,
                                 "residual": y_test - y_pred } )

In [149]: y_df
Out[149]:
```

	actual	predicted	residual
1	4947.1	4798.818983	148.281017
30	3083.2	3057.271086	25.928914
31	3019.8	3116.371941	-96.571941
36	2106.7	2133.335032	-26.635032
23	2947.2	2953.389564	-6.189564
37	2059.2	1972.005738	87.194262
8	3971.6	3921.641787	49.958213
13	2281.0	2332.387662	-51.387662

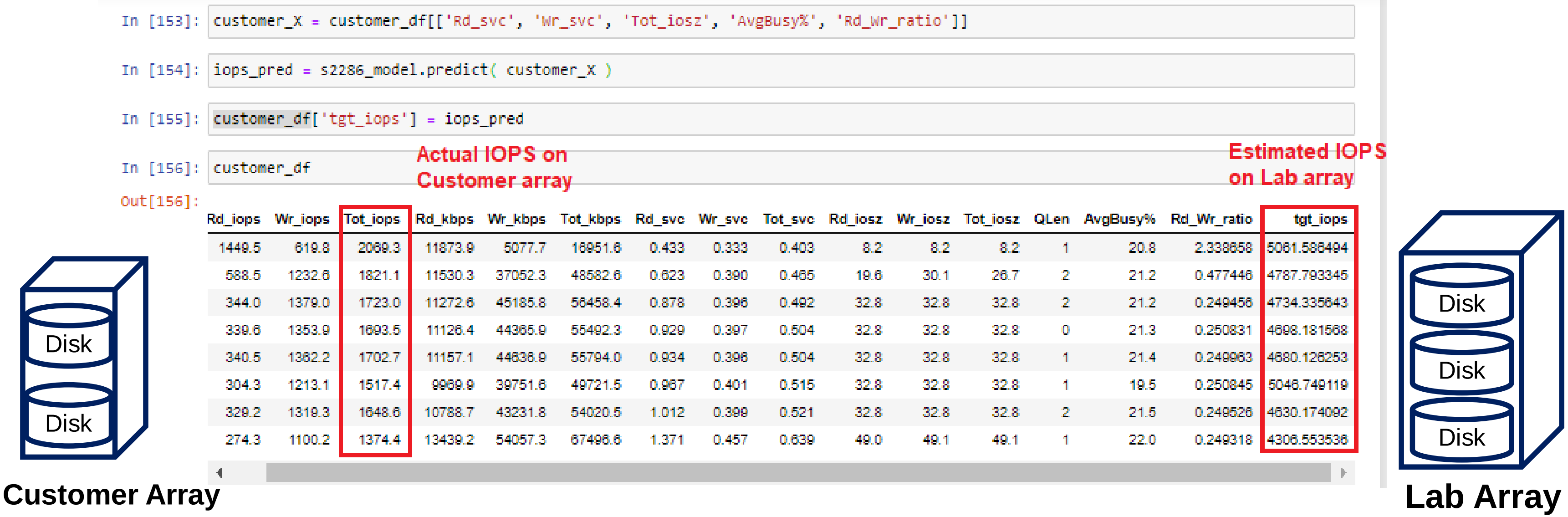
```
In [150]: customer_df = pd.read_csv("./Input_data/customer.csv")
```



```
In [157]: from sklearn.metrics import mean_squared_error

In [158]: mse_s2286_model = mean_squared_error( y_df.actual,
                                                y_df.predicted )

In [159]: import numpy as np

In [160]: np.sqrt( mse_s2286_model )

Out[160]: 75.39296915165409

In [161]: from sklearn.metrics import r2_score

In [162]: r2_score( y_df.actual,
                    y_df.predicted )

Out[162]: 0.9934323420220152

In [150]: customer_df = pd.read_csv("./Input_data/customer.csv")
```

- This set of data generated on SUT system only used for testing the model. Model was not built with these data
- Mean error between actual test data and predicted data is just 75 IOPS.
- R2-score value is between 0.0 and 1.0. Where 0.0 is being highly incorrect model and 1.0 indicates perfect model – In this case, it shows .99 with test data set

Prediction Vs. Simulation



```
In [153]: customer_X = customer_df[['Rd_svc', 'Wr_svc', 'Tot_iosz', 'AvgBusy%', 'Rd_Wr_ratio']]

In [154]: iops_pred = s2286_model.predict( customer_X )

In [155]: customer_df['tgt_iops'] = iops_pred

In [156]: customer_df
```

Out[156]:

Rd_iops	Wr_iops	Tot_iops	Rd_kbps	Wr_kbps	Tot_kbps	Rd_svc	Wr_svc	Tot_svc	Rd_iosz	Wr_iosz	Tot_iosz	QLen	AvgBusy%	Rd_Wr_ratio	tgt_iops
1449.5	619.8	2069.3	11873.9	5077.7	16951.6	0.433	0.333	0.403	8.2	8.2	8.2	1	20.8	2.338658	5061.586494
588.5	1232.6	1821.1	11530.3	37052.3	48582.6	0.623	0.390	0.465	19.6	30.1	26.7	2	21.2	0.477446	4787.793345
344.0	1379.0	1723.0	11272.6	45185.8	56458.4	0.878	0.396	0.492	32.8	32.8	32.8	2	21.2	0.249456	4734.335643
339.6	1353.9	1693.5	11126.4	44365.9	55492.3	0.929	0.397	0.504	32.8	32.8	32.8	0	21.3	0.250831	4698.181568
340.5	1362.2	1702.7	11157.1	44636.9	55794.0	0.934	0.396	0.504	32.8	32.8	32.8	1	21.4	0.249963	4680.126253
304.3	1213.1	1517.4	9969.9	39751.6	49721.5	0.967	0.401	0.515	32.8	32.8	32.8	1	19.5	0.250845	5046.749119
329.2	1319.3	1648.6	10788.7	43231.8	54020.5	1.012	0.399	0.521	32.8	32.8	32.8	2	21.5	0.249526	4630.174092
274.3	1100.2	1374.4	13439.2	54057.3	67496.6	1.371	0.457	0.639	49.0	49.1	49.1	1	22.0	0.249318	4306.553536

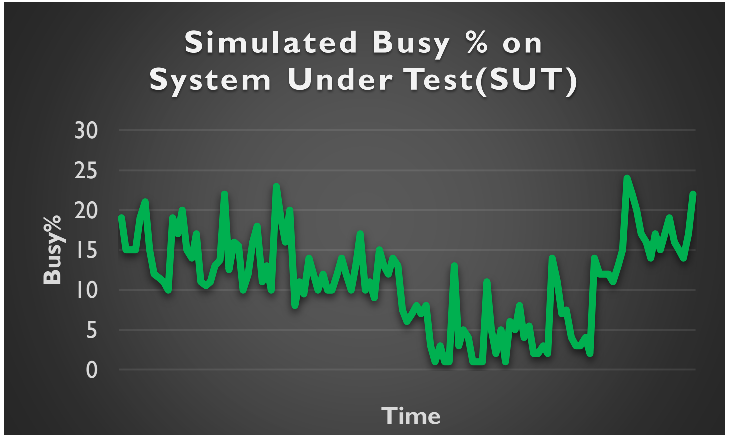
Actual IOPS on Customer array

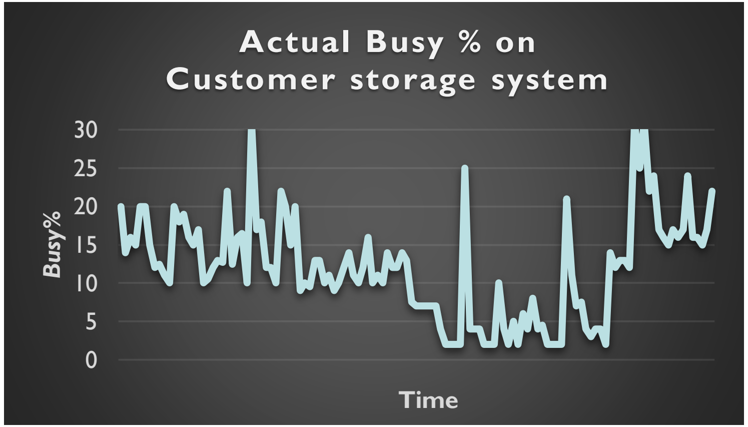
Estimated IOPS on Lab array

```
root@mnodee117c885:~# srstatvlun -hires -btsecs "2020-09-02 09:35:00" -etsecs "2020-09-02 09:50:00" -vv vv1
                                   --------IO/s-------- ---------KBytes/s--------- -----Svct ms----- -IOSz KBytes--
                    Time        Secs     Rd     Wr    Tot       Rd       Wr      Tot    Rd    Wr   Tot   Rd   Wr  Tot QLen AvgBusy%
2020-09-02 09:35:00 PDT 1591115700 986.6 3949.9 4936.5 32360.4 129556.7 161917.2 0.631 0.374 0.405 32.8 32.8 32.8    0     19.7
2020-09-02 09:40:00 PDT 1591116000 928.5 3819.8 4748.3 30454.8 125289.4 155744.2 0.866 0.372 0.511 32.8 32.8 32.8    1     20.2
2020-09-02 09:45:00 PDT 1591116300 898.3 3609.9 4508.2 29464.2 118404.7 147868.9 0.582 0.364 0.416 32.8 32.8 32.8    0     20.8
2020-09-02 09:50:00 PDT 1591116600 951.6 3994.1 4945.7 31212.4 131006.4 162218.9 0.699 0.562 0.419 32.8 32.8 32.8    2     20.1
-----------------------------------------------------------------------------------------------------------------------------
                                 4 932.7 3718.4 4651.1 30592.5 121963.5 152556.1 0.668 0.374 0.463 32.8 32.8 32.8  0.8     20.5
root@mnodee117c885:~#
```

Busy% of Lab vs Customer Array

- Existing internal IO tool has been enhanced to supply IOPS as an input along with other parameters and iterate every data samples to replay the workload.
- Following graphs shows that simulated load on System under test is very close to the one in Customer storage system:
- Able to reach close to Busy% and IO processing(Latency) condition on Lab array with same as customer workload characteristics





Thank You for Attending

Please take a moment
to rate this session.

Your feedback matters to us.