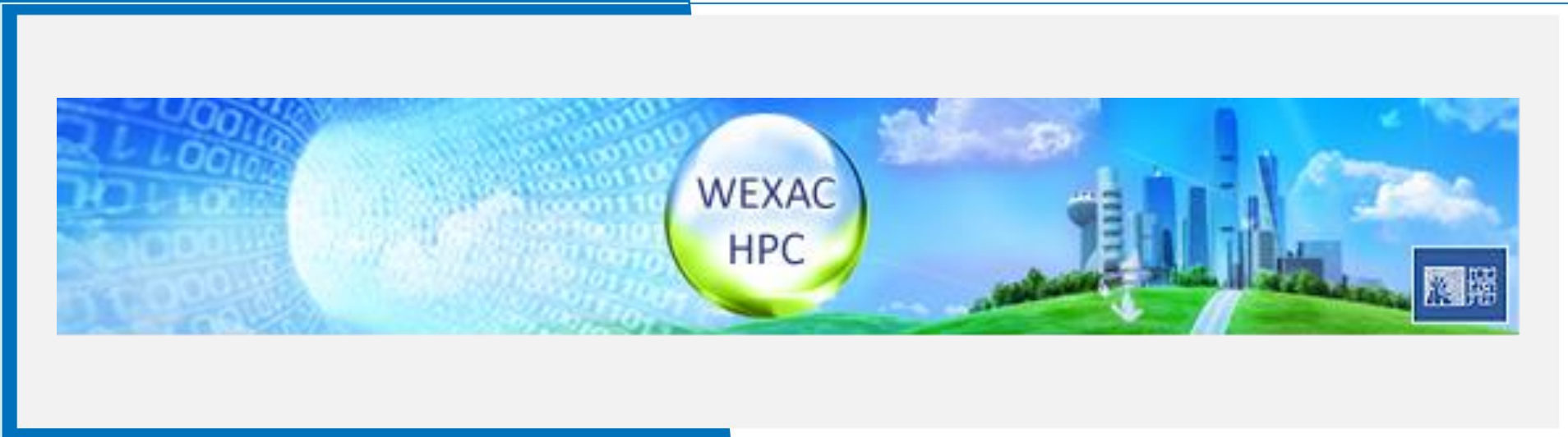


Information Technology Infrastructure Branch



WAIC workshop

Presented by: Vadim Malkin

June 25, 2019

Agenda

- Brief Introduction
- Monitoring
- Accessing the System
- Submitting Jobs
- Introduction to Docker
- Use Cases
- Question

Back to the Future: From WEIZAC to WEXAC

USERS TRAINING

hpc@weizmann.ac.il

Current WEXAC Status

- Computing power : more than 300 servers , above 7000 cores – the biggest Weizmann Institute of Science campus cluster
- Storage capacity : 2 PByte total capacity, about 1 PByte in use
 - 1.6 Pbyte of additional storage for StorWIS, about 0.9 PByte in use
- Modules : more than 600 scientific applications with different versions installed in WEXAC.
- Nvidia NGC repository integration.
- Containerized ready environment

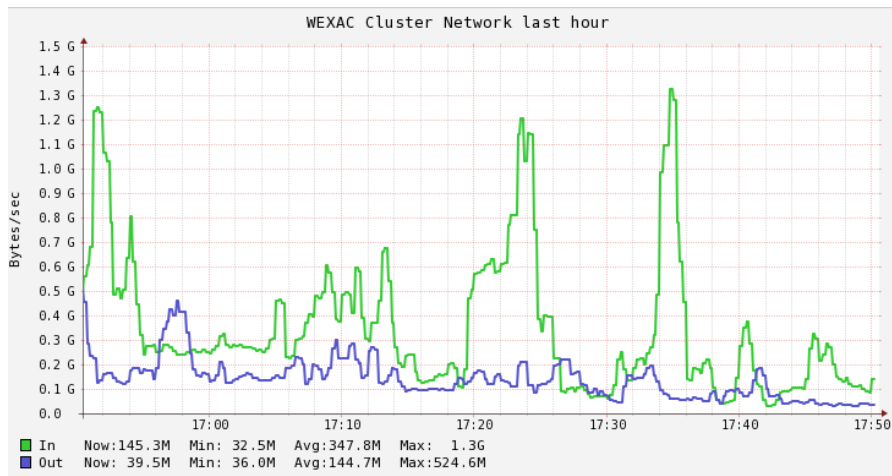
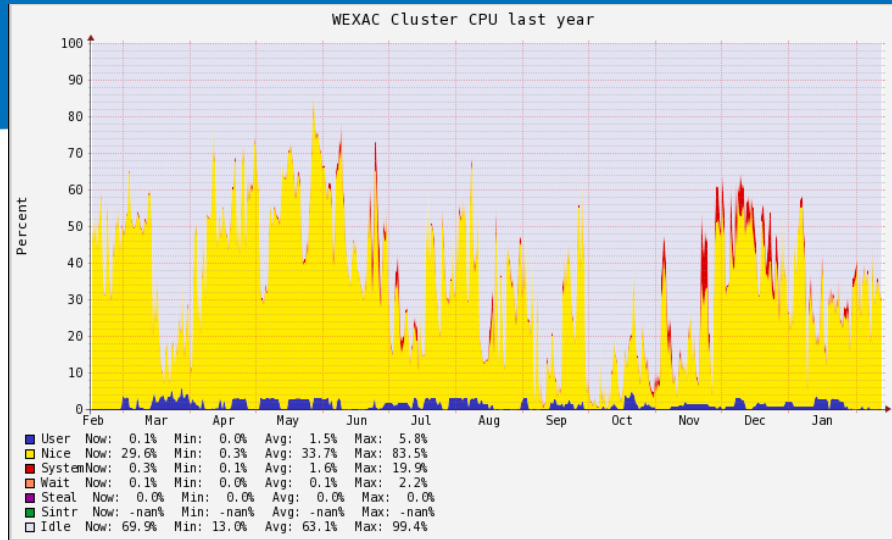
Best practices to compile serial and parallel applications.

- Use Latest Compilers and frameworks
- Use the relevant servers for compilation
- Use a fast interconnect (Infiniband)
- In parallel application cases use a MPI library with good collectiveness (MVAPICH2 or Intel MPI)
- Don't re-invent the wheel – use optimized libraries , like NetCDF , HDF5 , Intel MKL, Nvidia NGC
- Use a fast FFT library (FFTW or Intel FFTW wrappers)
- Use productivity tools like DDT , mpiP , OpenMPI , STAT, Nvidia profilers for debugging and application profiling

Ganglia Monitoring and Graphics Tool

- Ganglia monitoring tool is primary built for monitoring clusters of servers, and it does its job at the best.
- It represents the overall performance of a cluster of servers in an overview
- Suppose you have multiple clusters in a datacenter, then in that case you can arrange them and call it as a grid and have an overall performance overview of that grid.
- Performance monitoring to the base machine of a collection of virtual machines, can be done using ganglia monitoring tool.

WEXAC real-time monitor: Ganglia and RTM



Batch Job Filters [Updated 2 Minutes and 46 Seconds Ago]

Cluster: wexac User: All UGroup: All Status: RUI

Queue: All Host: All HGroup: All Records: 30

JobID: App: All JGroup: All

Search: ResReq:

Showing Rows 1 to 30 of 2561 [1,2,3,4,5,6,7,8]

JobID	Job Name	Queue	User	Project Name	Status	State	Changes	Mem Reserved	Max Memory
982567	/home/labs/fleishman...	fleishman	elazara	default	RUNNING	18		2.00G	-
965450	#/bin/bash.#BSUB ...	huge-queue	bndidi	default	RUNNING	2		59.57G	935.00M 923.00M 4.36h 0.65% 1 1 cn133
949858	/home/labs/fleishman...	fleishman	elazara	default	RUNNING	18		2.00G	- - - - 1 1 cn540
947763	/home/labs/fleishman...	fleishman	elazara	default	RUNNING	18		2.00G	- - - - 1 1 cn540
947762	/home/labs/fleishman...	fleishman	elazara	default	RUNNING	18		2.00G	- - - - 1 1 cn540
944849	/home/labs/fleishman...	fleishman	elazara	default	RUNNING	18		2.00G	- - - - 1 1 cn540
944848	/home/labs/fleishman...	fleishman	elazara	default	RUNNING	18		2.00G	- - - - 1 1 cn540
944846	/home/labs/fleishman...	fleishman	elazara	default	RUNNING	18		2.00G	- - - - 1 1 cn540
939479	math-noprompt-run ...	new-all.q	avimayo	default	RUNNING	18		2.00G	- - - - 1 1 cn540
939478	math-noprompt-run ...	new-all.q	avimayo	default	RUNNING	18		2.00G	- - - - 1 1 cn540
481228	gdc-client download ...	new-all.q	yoavw	default	RUNNING	4		19.53G	19.53G 1.19G 6.1h 0.72% 1 1 cn252
481203	gdc-client download ...	new-all.q	yoavw	default	RUNNING	2		19.53G	19.53G 19.53G 9.27h 1.08% 1 1 cn082
481156	gdc-client download ...	new-all.q	yoavw	default	RUNNING	2		19.53G	19.53G 8.70G 8.44h 0.98% 1 1 cn082
481155	gdc-client download ...	new-all.q	yoavw	default	RUNNING	2		19.53G	19.53G 887.00M 7.96h 0.92% 1 1 cn082

IBM® Spectrum LSF RTM 10.1.0.0

User Name: aawxc13

Password: *****

Realm: General Users

Log In

© Copyright International Business Machines Corp. 2006-2016. US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Ganglia Grid Overview

WEXAC Cluster Report at Thu, 09 Feb 2017 16:47:48 +0200

Get Fresh Data

Last or from to

Grid > **WEXAC** > --Choose a Node

Overview of WEXAC @ 2017-02-09 16:47

CPUs Total: **8928**
 Hosts up: **290**
 Hosts down: **0**

Current Load Avg (15, 5, 1m): **42%, 41%, 41%**
 Avg Utilization (last hour): **36%**

Server Load Distribution

WEXAC Cluster Load last hour

1-min	Now: 3.6k	Min: 749.3	Avg: 3.3k	Max: 6.1k
Nodes	Now: 289.0	Min: 289.0	Avg: 289.0	Max: 289.0
CPUs	Now: 8.9k	Min: 8.9k	Avg: 8.9k	Max: 8.9k
Procs	Now: 3.8k	Min: 755.8	Avg: 3.4k	Max: 6.1k

WEXAC Cluster Memory last hour

Use	Now: 1.6T	Min: 1.3T	Avg: 1.5T	Max: 1.9T
Share	Now: 0.0	Min: 0.0	Avg: 0.0	Max: 0.0
Cache	Now: 4.6T	Min: 4.6T	Avg: 4.6T	Max: 4.6T
Buffer	Now: 95.6G	Min: 95.4G	Avg: 95.5G	Max: 95.6G
Free	Now: 26.8T	Min: 26.6T	Avg: 26.9T	Max: 27.1T
Swap	Now: -nan	Min: -nan	Avg: -nan	Max: -nan
Total	Now: 33.2T	Min: 33.2T	Avg: 33.2T	Max: 33.2T

WEXAC Cluster CPU last hour

User	Now: 0.1%	Min: 0.1%	Avg: 0.1%	Max: 0.1%
Nice	Now: 37.2%	Min: 6.2%	Avg: 28.9%	Max: 40.6%
System	Now: 0.3%	Min: 0.2%	Avg: 0.4%	Max: 1.9%
wait	Now: 0.0%	Min: 0.0%	Avg: 0.0%	Max: 0.2%
Steal	Now: 0.0%	Min: 0.0%	Avg: 0.0%	Max: 0.0%
Sintr	Now: -nan%	Min: -nan%	Avg: -nan%	Max: -nan%
Idle	Now: 62.5%	Min: 57.5%	Avg: 70.6%	Max: 93.5%

WEXAC Cluster Network last hour

In	Now: 12.4M	Min: 2.2M	Avg: 41.7M	Max: 268.0M
Out	Now: 16.3M	Min: 11.0M	Avg: 14.4M	Max: 20.3M

Platform RTM

- Can be Used to monitor and manage LSF jobs in Real Time
- https://www.youtube.com/watch?v=if7R_Ild4Wg – learning Video
- <http://rtm.wexac.weizmann.ac.il> – WEXAC's RTM

RTM Graphs

The screenshot displays a web-based monitoring interface for IBM WEXAC. The browser window shows the URL `lsfutil.wexac.weizmann.ac.il/cacti/graph_view.php?action=tree&tree_id=3&leaf_id=3&select_first=true`. The interface includes a navigation menu on the left with options like Hostgroup Stats, Shared Resources, Queue Stats, and Reports. The main content area features a 'Graph Filters' section with a date range from 2017-01-30 00:00 to 2017-02-05 23:59 and a search filter for 'fleishman_hosts'. Two graphs are displayed: 'wexac_Summary - fleishman_hosts - Available Memory' and 'wexac_Summary - fleishman_hosts - CPU Capacity'. The available memory graph shows physical, swap, and total memory usage over time. The CPU capacity graph shows the CPU factor/UT usage.

Memory Type	Cur	Avg	Min	Max
Physical Memory	2.17 T	1.93 T	1.67 T	2.17 T
Swap Memory	1.23 T	1.21 T	1.21 T	1.23 T
Total Memory	2.29 T			

RTM Running Jobs

Grid -> View Job Listing | RTM is monitoring 160 hosts on 1 cluster.

Batch Job Filters [Updated 4 Minutes and 54 Seconds Ago]

Cluster: wexac User: All UGroup: All Status: RUNNING Effic: All

Queue: All Host: All HGroup: All Records: 30 Except: N/A

JobID: App: All JGroup: All

Search: ResReq: Dynamic Cluster TZ

Showing Rows 1 to 30 of 3118 [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21...]

JobID	Job Name	Queue	User	UGroup	Project Name	JGroup	Status	State Changes	Mem Reserved	Mem Usage	CPU Usage	CPU Effic	Num Nodes	Num CPUs	Max Procs	Execution Host	Start Time	End Time	
946540	/bin/bash,#BSUB ...	huge-queue	bndidi	-	default	-	RUNNING	2	59.57G	935.00M	923.00M	3.89%	0.65%	1	1	1	cn133	01-15 15:17:32	-
946592	/home/labs/fleishman...	fleishman	elazara	fleishman-wx-grp-lsf	default	-	RUNNING	2	2.00G	-	-	-	-	1	1	1	cn540	02-09 16:38:51	-
946591	/home/labs/fleishman...	fleishman	elazara	fleishman-wx-grp-lsf	default	-	RUNNING	2	2.00G	-	-	-	-	1	1	1	cn523	02-09 16:38:48	-
946590	/home/labs/fleishman...	fleishman	elazara	fleishman-wx-grp-lsf	default	-	RUNNING	2	2.00G	-	-	-	-	1	1	1	cn518	02-09 16:38:48	-
946589	/home/labs/fleishman...	fleishman	elazara	fleishman-wx-grp-lsf	default	-	RUNNING	2	2.00G	-	-	-	-	1	1	1	cn524	02-09 16:38:48	-
946588	/home/labs/fleishman...	fleishman	elazara	fleishman-wx-grp-lsf	default	-	RUNNING	2	2.00G	-	-	-	-	1	1	1	cn544	02-09 16:38:48	-
946587	/home/labs/fleishman...	fleishman	elazara	fleishman-wx-grp-lsf	default	-	RUNNING	2	2.00G	-	-	-	-	1	1	1	cn507	02-09 16:38:48	-
946586	/home/labs/fleishman...	fleishman	elazara	fleishman-wx-grp-lsf	default	-	RUNNING	2	2.00G	-	-	-	-	1	1	1	cn501	02-09 16:38:48	-
946585	/home/labs/fleishman...	fleishman	elazara	fleishman-wx-grp-lsf	default	-	RUNNING	2	2.00G	-	-	-	-	1	1	1	cn513	02-09 16:38:48	-
946584	/home/labs/fleishman...	fleishman	elazara	fleishman-wx-grp-lsf	default	-	RUNNING	2	2.00G	-	-	-	-	1	1	1	cn505	02-09 16:38:48	-
946583	/home/labs/fleishman...	fleishman	elazara	fleishman-wx-grp-lsf	default	-	RUNNING	2	2.00G	-	-	-	-	1	1	1	cn540	02-09 16:38:48	-

RTM Reports

The screenshot displays a web browser window with a URL: `Isfutit.wexac.weizmann.ac.il/cacti/plugins/heuristics/gnid_heuristics.php?heuristics_type=-1&clusterid=1&queue=new-all.q&rows_selector=30&go=Go&project=&reqcpus=&resreq=&users_page=1`. The interface includes a navigation menu on the left with options like 'Dashboards', 'Job Info', 'User/Group Info', 'Host Info', and 'Reports'. The main content area shows a 'Batch Grid Heuristics Filters' section with dropdowns for 'Rollup' (No Rollup), 'Cluster' (wexac), and 'Queue' (new-all.q). Below this is a table of job data.

Cluster	Queue	Project	resReq	CPU	Jobs	Cores	Run Time (Avg)	Run Time (Max)	Run Time (Min)	Run Time (Stddev)	Run Time (Median)	Run Time (25% throughput)	Run Time (75% throughput)	Run Time (90% throughput)	Job Throughput (Hourly Avg)	Job Throughput (Hourly Stddev)
wexac	new-all.q	default	select(((type = any) && (type == any))) order[r15s:pg] rusage[mem=1000.00] span [hosts=1]	4	1611	6444	2.67m	14.18m	35s	54.48s	N/A	N/A	N/A	N/A	22.437	11.115
wexac	new-all.q	default	select(((type = any) && (type == any))) order[r15s:pg] rusage[mem=3000.00]	1	955	955	12.04m	2.88h	3s	18.09m	N/A	N/A	N/A	N/A	4.985	0.905
wexac	new-all.q	default	select[type == local] order [r15s:pg] rusage [mem=2048.00]	1	868	868	34.84m	3.16h	22s	56.14m	N/A	N/A	N/A	N/A	1.722	0.295
wexac	new-all.q	default	select(((type = any) && (type == any))) order[r15s:pg] rusage[mem=8000.00] span [hosts=1]	1	855	855	11.61m	3.07h	1.57m	8.73m	N/A	N/A	N/A	N/A	5.168	1.587
wexac	new-all.q	default	select(((type = any) && (type == any))) order[r15s:pg] rusage[mem=2048.00]	1	841	841	15.62m	2.69h	1s	28.93m	N/A	N/A	N/A	N/A	3.841	0.586
wexac	new-all.q	default	select(((type = any) && (type == any))) order[r15s:pg] rusage[mem=2500.00] span [hosts=1]	1	731	731	29.67s	1.88m	22s	9.48s	N/A	N/A	N/A	N/A	121.35	61.962
wexac	new-all.q	default	select(((type = any) && (type == any))) order[r15s:pg] rusage[mem=4000.00] span [hosts=1]	2	727	1454	27.29m	2.1h	5.32m	17.23m	N/A	N/A	N/A	N/A	2.199	0.76
wexac	new-	default	select(((type = any) && (type == any))) order[r15s:pg]	1	560	560	2.1d	21.16d	3.42h	3.14d	N/A	N/A	N/A	N/A	0.02	0.004

RTM Batch Users

The screenshot displays the Grid Heuristics web interface for user 'aharonn'. The interface includes a navigation bar with 'Grid' and 'JobIQ' tabs, and a 'User Filter' section for 'RTM Batch Users'. The main content area is divided into several sections:

- User Filter:** Shows filters for User (aharonn), Cluster (All), Queue (All), and Severity (All). It includes checkboxes for Summary, Throughput, ExitAnalysis, and ClusterHealth.
- User aharonn Current Status by Cluster (all Queues):** A table showing 'No Records Found' for Pending, Running, Suspended, Finished/Hr, Exited/Hr, Hourly TPut, and 5 Min TPut.
- User aharonn Current Status by Queue/Project:** A table showing 'No Records Found' for Queue, Project, CPUS, Pending, Running, Suspended, 1 Hr TPut, 5 Min TPut, Estimate, Idle Jobs, Long Jobs, Pend Dpnd, and Mem Use.
- User aharonn Daily Throughput by Cluster:** A table showing throughput for the 'new-all.q' queue.

Queue	Ended Today**	Exited Today	Average TT Today	Ended Yesterday	Exited Yesterday
new-all.q	517	2	6 Minutes	153	5
- User aharonn Exit Analysis by Queue/Project:** A table showing exit analysis for the 'new-all.q' queue.

Queue	Project	Jobs**	Reason
new-all.q	default	2	App Exited, Code: 130, TERM_OWNER: job killed by owner
- Cluster Health:** A table showing 'No Records Found' for Cluster, Cluster Status, RTM Status, Master Status, Active Alerts, Down Hosts, and Throughput.
- Charts:**
 - User aharonn Job Statistics Today for All Queues in All Clusters:** A 3D pie chart showing 517 Started jobs (blue), 2 Ended jobs (yellow), and 517 Exited jobs (green).
 - User aharonn Memory Use Today for DONE Jobs in All Queues in All Clusters:** A bar chart showing memory usage for various ranges. The 4G to 8G range has the highest usage at 458.

Memory Range	Count
0G to 1G	7
1G to 2G	0
2G to 4G	33
4G to 8G	458
8G to 16G	16
16G to 24G	3
24G to 32G	0
32G to 64G	0
64G+	0

GPU monitoring tools

- <http://waicmgmt01.wexac.weizmann.ac.il:30200/>
- Grafana monitoring tool allow to visualize DGX machines utilization

GUI portal for GPU status



MobaXterm for Windows users

- Enhanced terminal for Windows with X11 server, tabbed SSH client, network tools and much more
- **Download:**
<http://mobaxterm.mobatek.net/download.html>

MobaXterm User Interface

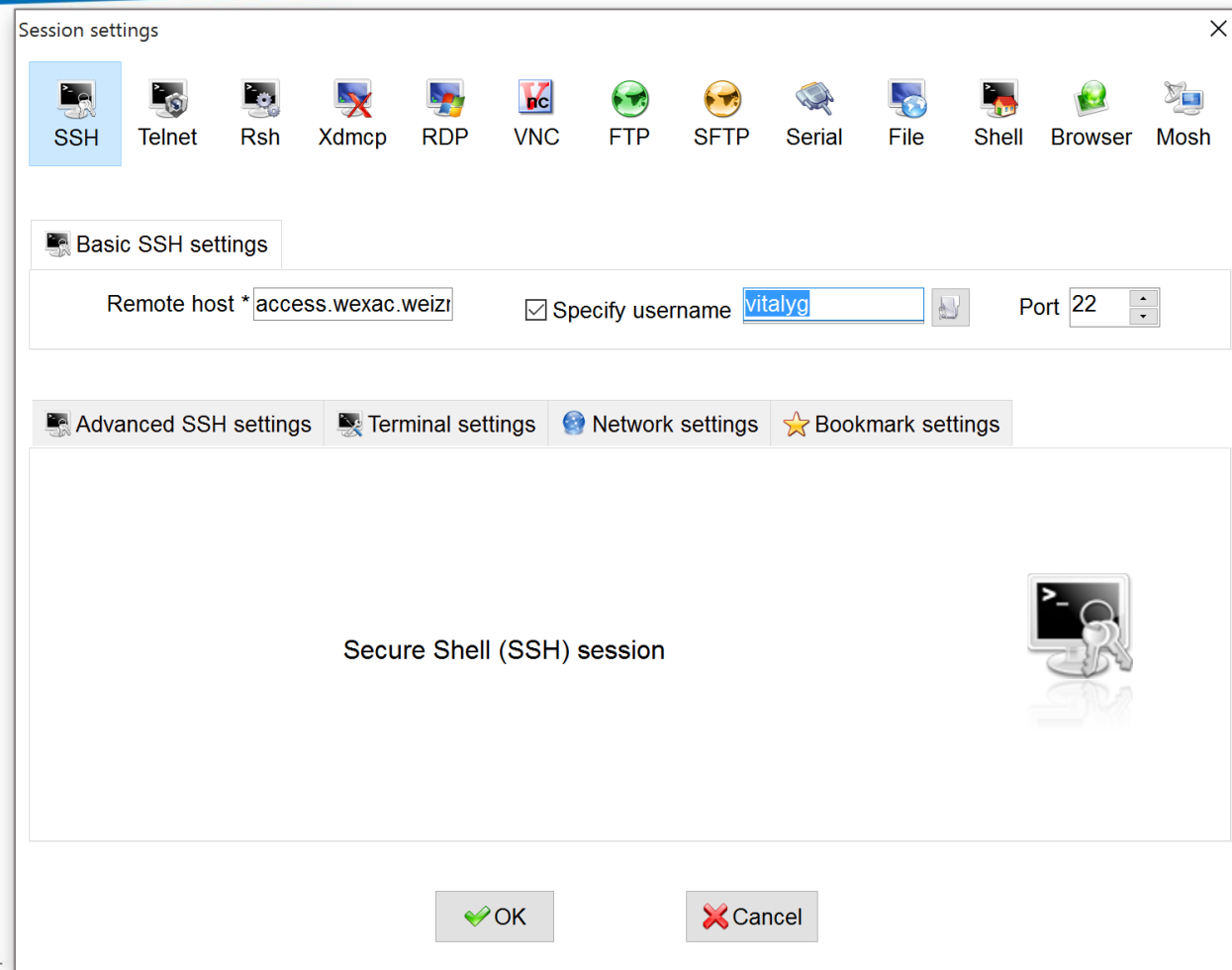
- MobaXterm Professional v8.6 •
(X server, SSH client and network tools)

- › Your computer drives are accessible through the /drives path
- › Your DISPLAY is set to 132.77.134.71:0.0
- › When using SSH, your remote DISPLAY is automatically forwarded
- › Your HOME folder is not persistent: it will be erased on restart
- › Each command status is specified by a special symbol (✓ or ✗)

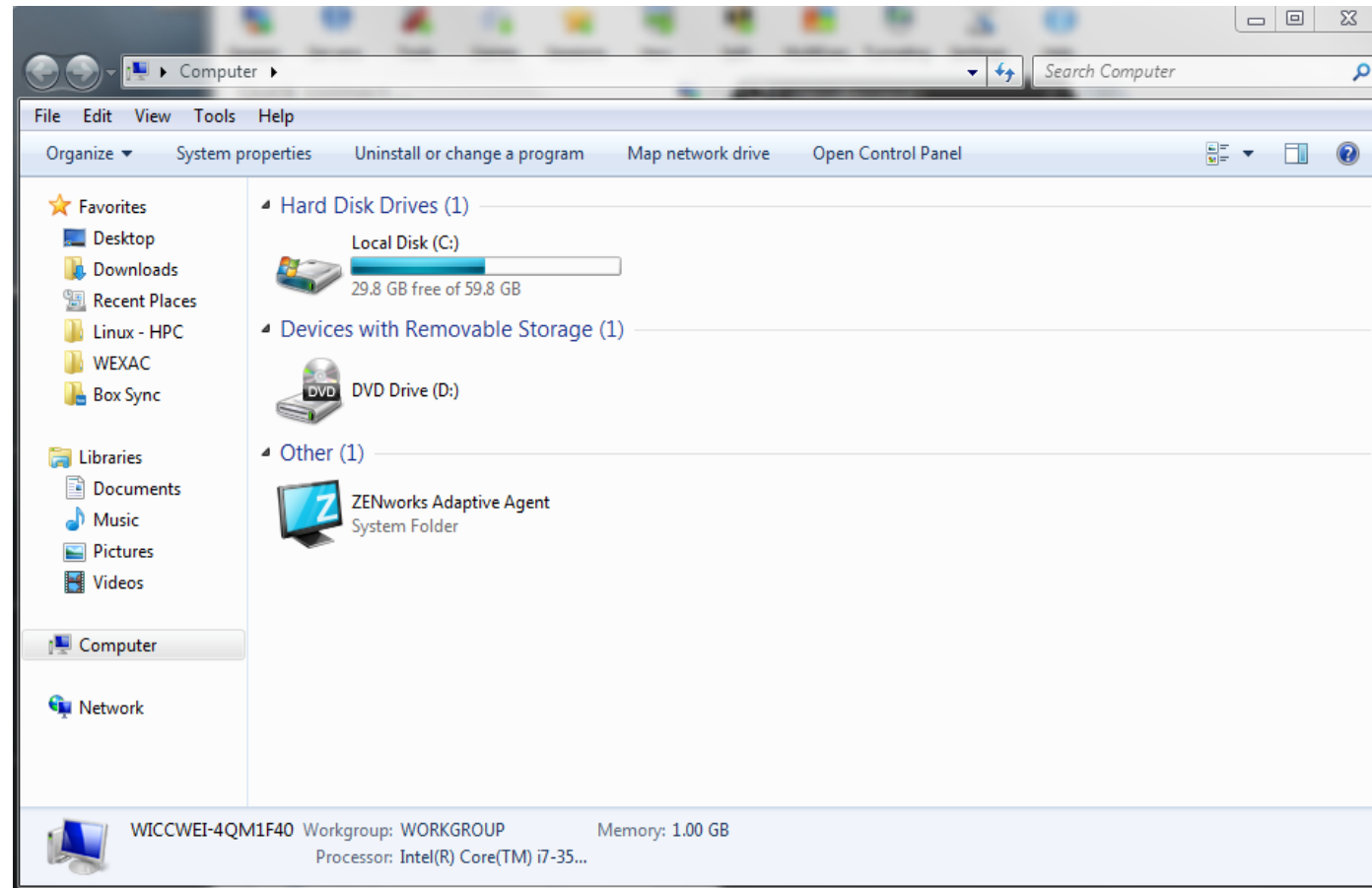
Registered to Weizmann Institute of Science (6 users)

```
[2017-02-12 17:48.47] ~  
[vadim.vadim-laptop] > █
```

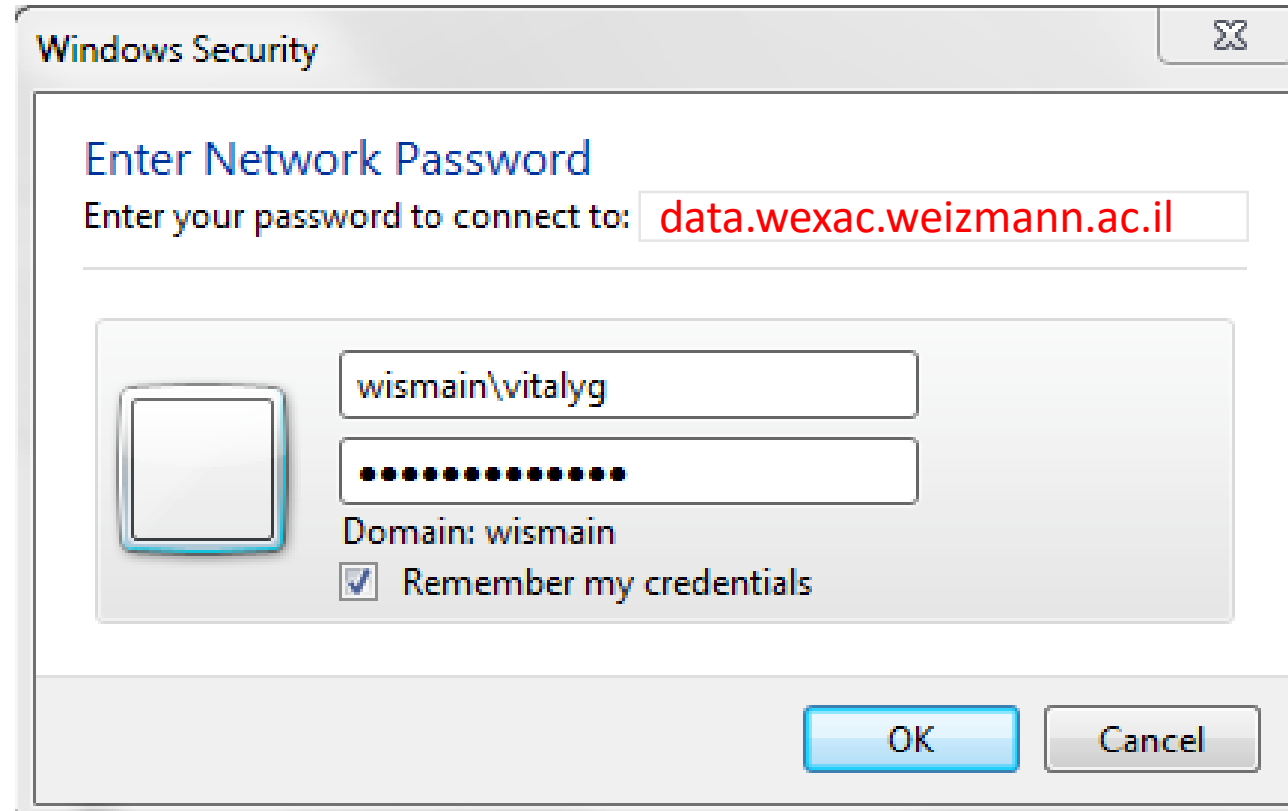
MobaXterm – Create SSH session



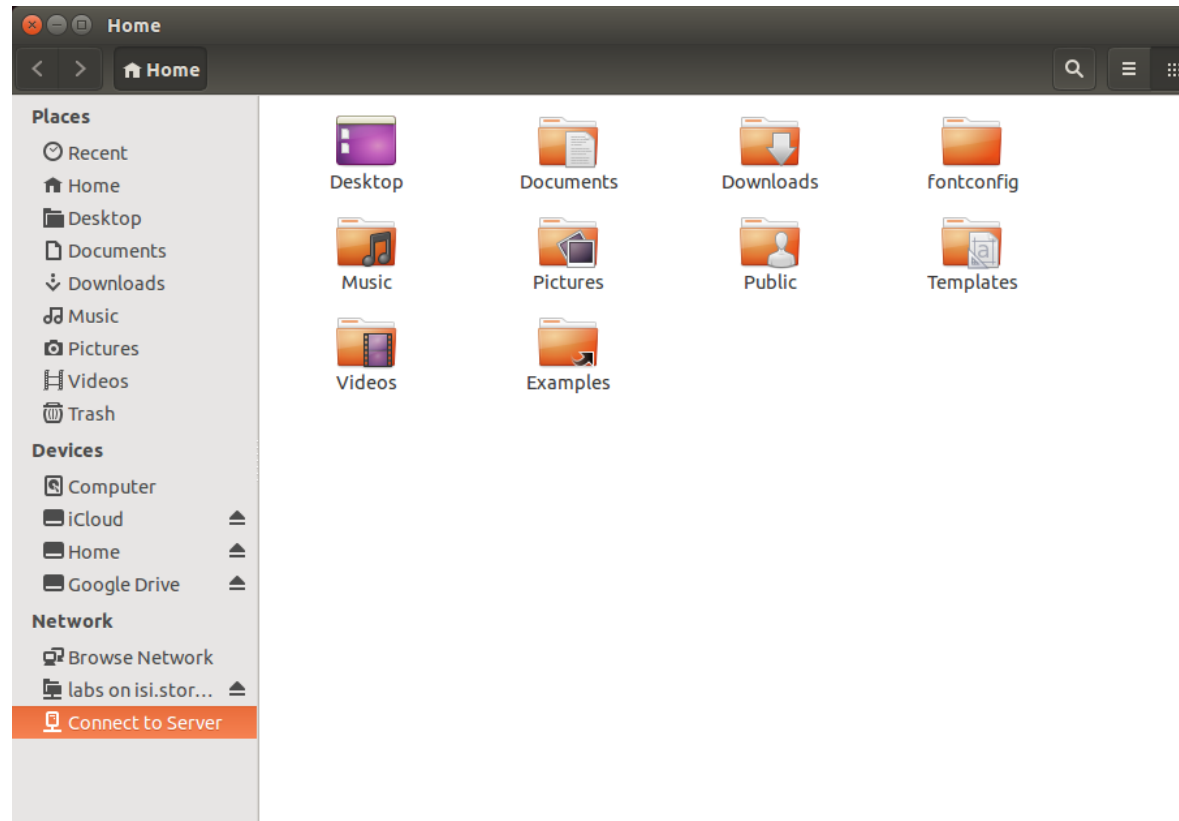
Map Network Drive – Windows workstation \\data.wexac.weizmann.ac.il



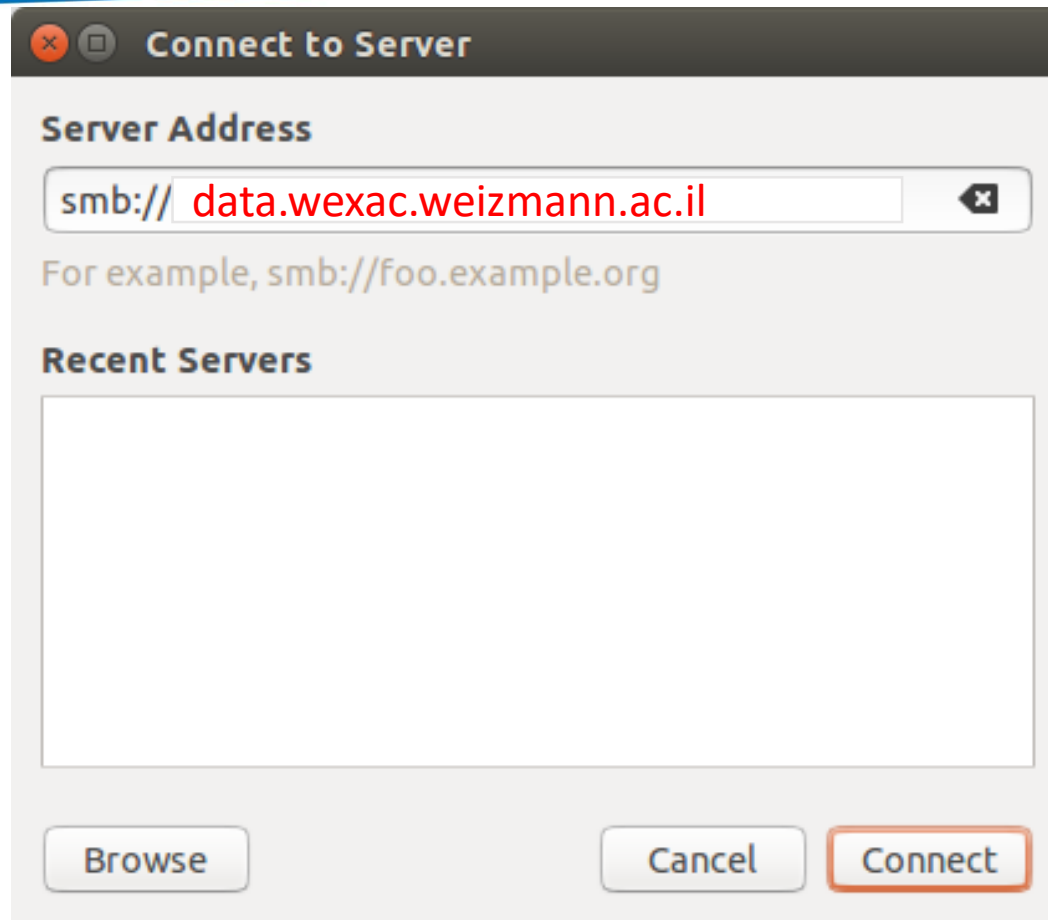
Universal WIS credentials



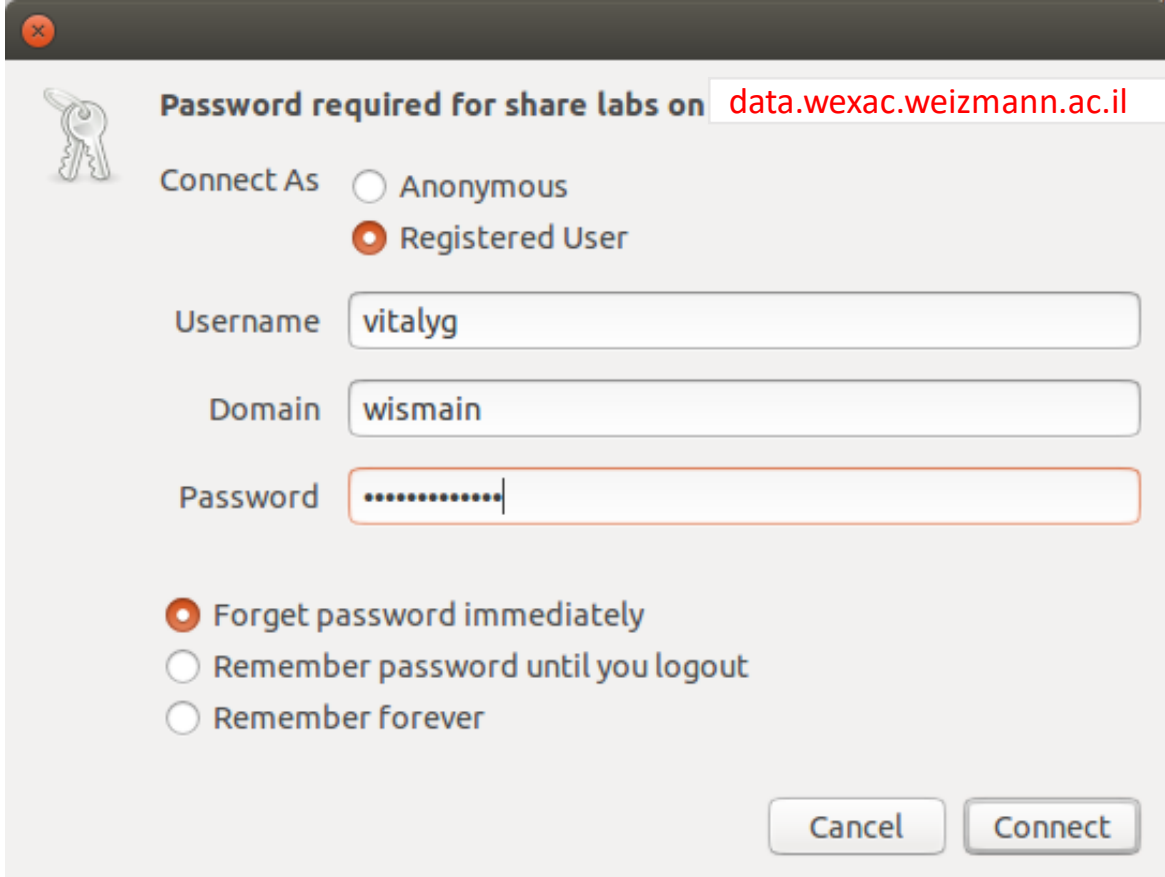
Access WEXAC from Ubuntu \\data.wexac.weizmann.ac.il



Access WEXAC from Ubuntu



Universal DIS credentials used to access WEXAC from Ubuntu



A screenshot of a Windows-style password dialog box. The title bar is dark gray with a close button (red 'x'). The main area is light gray. At the top left is a key icon. The text reads: "Password required for share labs on" followed by a text box containing "data.wexac.weizmann.ac.il". Below this are two radio buttons: "Anonymous" (unselected) and "Registered User" (selected). There are three text input fields: "Username" with "vitalyg", "Domain" with "wismain", and "Password" with a masked password of 12 dots. At the bottom are three radio buttons: "Forget password immediately" (selected), "Remember password until you logout" (unselected), and "Remember forever" (unselected). At the bottom right are two buttons: "Cancel" and "Connect".

Password required for share labs on

Connect As Anonymous
 Registered User

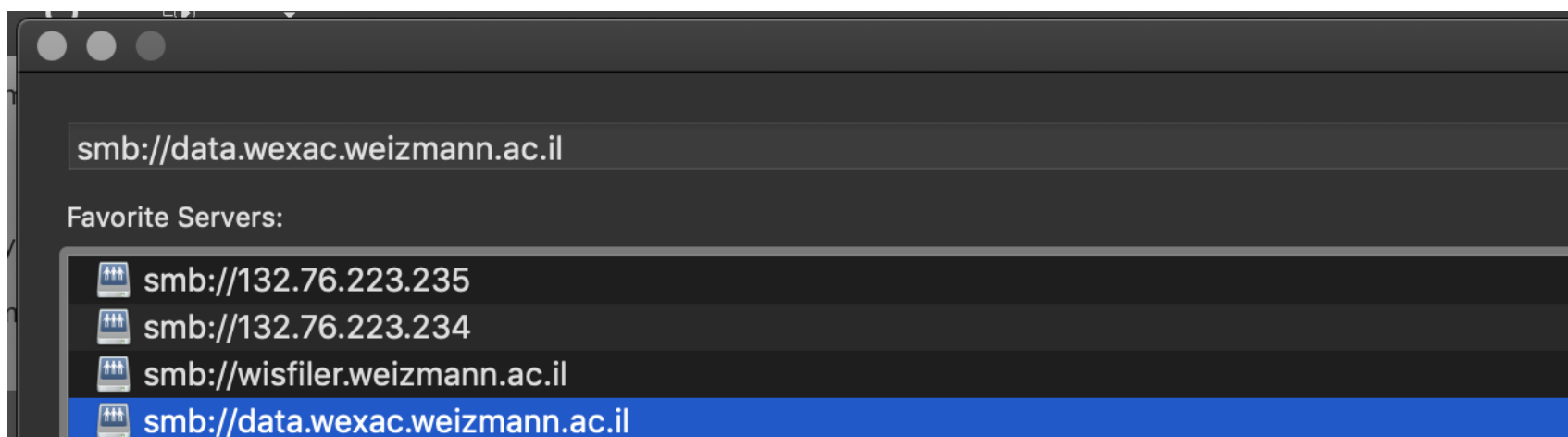
Username

Domain

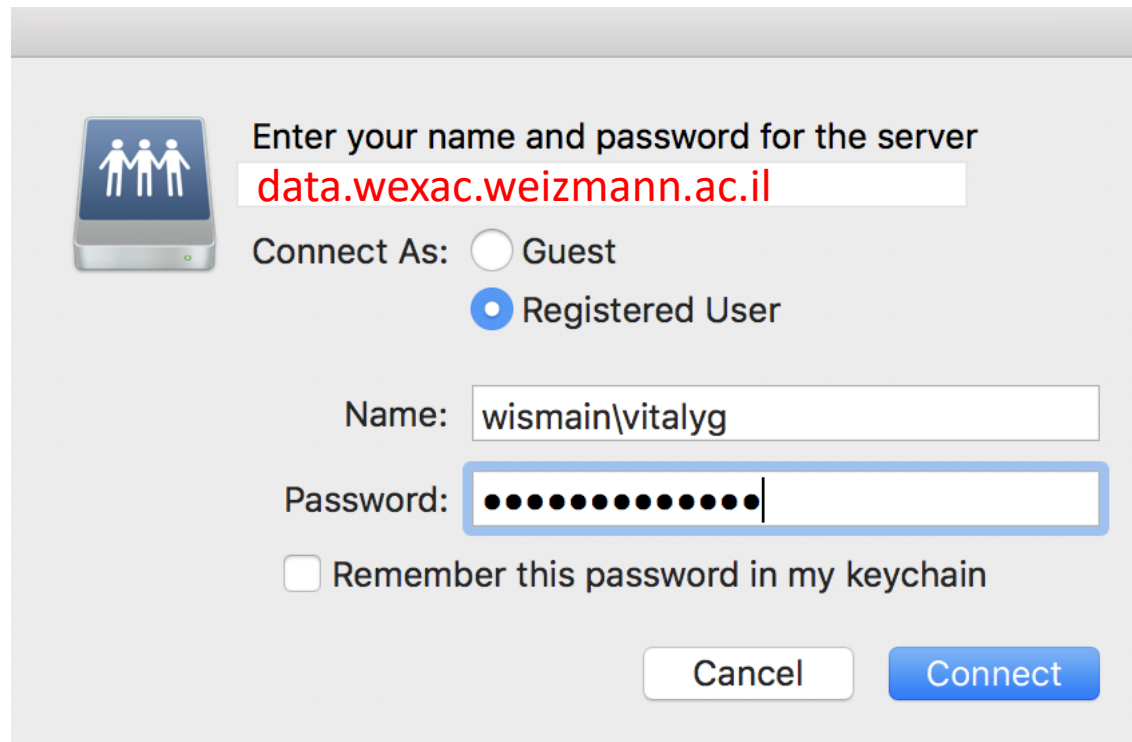
Password

Forget password immediately
 Remember password until you logout
 Remember forever

Access WEXAC from MAC



Access WEXAC from MAC using universal DIS credentials



Enter your name and password for the server
data.wexac.weizmann.ac.il

Connect As: Guest
 Registered User

Name:

Password:

Remember this password in my keychain

WEXAC Modules

```
[vitalyg@access ~]$ module avail
----- /apps/RH7U2/Modules/default/modulefiles/general -----
annovar/2016Feb01      homer/4.8              namd/2.12b1/multicore-CUDA
artemis/1.47           icompiler/2015         namd/2.9/ibverbs-smp-CUDA
aspera/3.5.4           icompiler/2016         namd/2.9/multicore-CUDA
assemblytics/2016Dec11 icompiler/2017         ngsplot/2.61
barcomb/1              IGV/2.3.88            pacbio/Jan-2017
blast/2.2.26           IMOD/4.7.15           prinseq/0.20.4
cap3/0.1               jdk/8.111             RepeatMasker/4.0.6
cellranger/1.2.1      jre/8.111             resmap/1.1.4
chimera/1.11.2         kallisto/0.43.0       SeqMonk/1.36.0
circos/0.69-3          mathematica/11.0(default) snpEff/3.0
cuda/7.5               mathematica/8.0        spider/23.03
cuda/8.0.44(default)   matlab/R2016b         svtyper/0.1.1
cytoscape/3.4.0        MotionCor2/10-19-2016 tablet/1.16.09.06
eigen/3.3.0            namd/2.12b1/ibverbs   trf/4.09
fastqc/0.11.5          namd/2.12b1/ibverbs-smp trimomatic/0.36
fastq_screen/0.9.5     namd/2.12b1/ibverbs-smp-CUDA unblur/1.02
GATK/3.7               namd/2.12b1/KNL-multicore weblogo/2.8.2

----- /apps/RH7U2/Modules/default/modulefiles/intel -----
bedtools/2.26.0        koren/intel-2017/hdf5/1.8.12 pigz/2.3.4
blat/35                koren/intel-2017/hdf5/1.8.16 python/2.7(default)
bowtie/1.1.2           koren/intel-2017/netcdf/4.1.3 python/3.5
bowtie2/2.2.9          kushnir/hdf5-2016/1.10.0   relion/1.4
bwa/0.7.15             ldhat/1.0               samtools/1.3.1
fftw/3.3.5             lumpypy/0.2.13          shrimp/2.2.3
fltk/1.3.4             moreutils/0.6           stringtie/1.3.0
hdf5/1.10.0            parallel-netcdf/1.7.0     vcftools/0.1.15
htslib/1.3.1           pear/0.9.10

----- /apps/RH7U2/Modules/default/modulefiles/gnu -----
abyss/2.0.2            cufflinks/2.2.1         hmer/3.1b1              ncbi-blast+/2.5.0      sqlite/3.15.2
bamtools/2.4.1         delly/0.7.6             jemalloc/4.4.0          openblas/0.2.19       sra-tools/2.8.0
bazel/0.4.2            delly_parallel/0.7.6   KaKs_Calculator/2.0    openmpi/2.0.1         stampy/1.0.31
bcftools/1.3.1         detonate/1.11           kentUtils               pandoc/1.18           star/2.5.2b
bc12fastq/2.17.1.14    dos2unix/7.3.4         lapack/3.7.0            pdsh/2.29             tabix/0.2.6
bedtools-gnu/2.26.0    ea-utils/1.1.2         libgd/2.2.2             perl/5.24.0           texlive/2016
bioawk/1.0             eig/6.1.3              libgs/2.3              perl/5.24.0_2         tiff/4.0.7
boost/1.55.0           emboss/6.6.0           libgttextutils/0.7      primer3/2.3.7         tophat/2.1.1
boost/1.62.0           expander/7.11          libxml2/2.9.4           python/2.7.12         trinity/2.3.2
breakdancer/1.4.5     falcon/0.3.0           libxslt/1.1.29         python/3.5.2          vcflib/1.0.0
breseq/0.29.0         fasta/36.3.8d          lynx/2.8.3              R/3.3.2               velvet/1.2.10
bwa-gnu/0.7.15        fastx/0.0.14           mafft/7.305            relion/2.0            ViennaRNA/2.3.1
bzip2/1.0.6           fftw/3.3.4             meme/4.11.2             rmbast/2.2.8          wxwidgets/3.1.0
cd-hit/4.6.6          file/5.29              mipgen/22-Dec-2016     rsem/1.3.0            zlib/1.2.8
clustalw/2.1          freebayes/1.1.0        mirDeep/2.0.0.8        samtools/0.1.19      zsh/5.2
cmake/3.7.1           gcc/5.4.0              motioncorr/2.1         samtools/1.3.1a
CORTEX/1.0.5.21       git/2.11.0             mummer/3.23            sga/0.10.15
cortex_con/0.05       graphviz/2.38.0        mvapich2/2.2           sickle/1.33
ctffind/4.1.5         hisat/2.0.5            mysql/5.7.16           sparsehash/2.0.3
```

Modules (contd.)

```
[vitalyg@access ~]$ module --help
```

```
Modules Release 3.2.10 2012-12-21 (Copyright GNU GPL v2 1991):
```

```
Usage: module [ switches ] [ subcommand ] [subcommand-args ]
```

Switches:

-H --help	this usage info
-V --version	modules version & configuration options
-f --force	force active dependency resolution
-t --terse	terse format avail and list format
-l --long	long format avail and list format
-h --human	readable format avail and list format
-v --verbose	enable verbose messages
-s --silent	disable verbose messages
-c --create	create caches for avail and apropos
-i --icase	case insensitive
-u --userlvl <lvl>	set user level to (nov[ice],exp[ert],adv[anced])

Available SubCommands and Args:

+ add load	modulefile [modulefile ...]
+ rm unload	modulefile [modulefile ...]
+ switch swap	[modulefile1] modulefile2
+ display show	modulefile [modulefile ...]
+ avail	[modulefile [modulefile ...]]
+ use [-a --append]	dir [dir ...]
+ unuse	dir [dir ...]
+ update	
+ refresh	
+ purge	
+ list	
+ clear	
+ help	[modulefile [modulefile ...]]
+ whatis	[modulefile [modulefile ...]]
+ apropos keyword	string
+ initadd	modulefile [modulefile ...]
+ initprepend	modulefile [modulefile ...]
+ initrm	modulefile [modulefile ...]
+ initswitch	modulefile1 modulefile2
+ initlist	
+ initclear	



Modules (contd.)

```
[vitalyg@access ~]$ module list
Currently Loaded Modulefiles:
  1) matlab/R2016b      2) mathematica/11.0   3) R/3.3.2             4) python/2.7
[vitalyg@access ~]$ module load bowtie2/2.2.9
[vitalyg@access ~]$ module list
Currently Loaded Modulefiles:
  1) matlab/R2016b      3) R/3.3.2            5) bowtie2/2.2.9
  2) mathematica/11.0  4) python/2.7
[vitalyg@access ~]$ module load samtools/1.3.1
[vitalyg@access ~]$ module list
Currently Loaded Modulefiles:
  1) matlab/R2016b      3) R/3.3.2            5) bowtie2/2.2.9      7) samtools/1.3.1
  2) mathematica/11.0  4) python/2.7        6) icompiler/2017
[vitalyg@access ~]$ █
```

WEXAC Job Submission & Control commands

- `bsub` - submits a batch job to LSF
- `bjobs` - displays information about LSF jobs
- `bhist` - displays historical information about jobs
- `bkill` - sends signals to kill, suspend, or resume unfinished jobs
- `bmod` - modifies job submission options of a job
- `bpeek` - displays the stdout and stderr output of an unfinished job
- `bstop` - suspends unfinished jobs
- `bresume` - resumes unfinished jobs
- `bswitch` - switches jobs from one queue to another

bsub – Commonly Used Options

- q *qname* submits the job to the specified queue
- o *file* redirect stdout, stderr and resource usage information of the job to the specified *output file*
- e *file* redirect stderr to the specified error file
- oo/-eo *file* same as -o/-e, but overwrite file if it exists
- i *filename* use the specified file as standard input for the job
- n *number* specify number of job slots
- g *jobgroup* submit job to specified group
- J *jobname* assigns the specified name to the job
- R *res_req* runs job on a host that meets the specified resource requirements
- L *shell* Initializes the execution environment using the specified login shell

bsub – Methods for Submitting Jobs

- By script or command

```
$ bsub -q new-new-all.q -J example -o example-%J.o -e  
example-%J.e date
```
- By job spooling

```
$ bsub < job_file
```

job_file example:

```
#BSUB -q new-all.q  
#BSUB -J example  
#BSUB -o example-%J.o  
#BSUB -e example-%J.e  
date
```

bsub – Methods for Submitting Jobs

Interactively

```
$ bsub
```

```
bsub> #BSUB -q new-all.q
```

```
bsub> #BSUB -J example
```

```
bsub> #BSUB -o example-%J.o
```

```
bsub> #BSUB -e example-%J.e
```

```
bsub> date
```

```
bsub> ^D
```

```
Job <2387> is submitted to queue <new-all.q>.
```

Resource Requirements (-R)

Resource requirement string is divided into following sections:

- Selection - **select**[*selection_string*]
- Usage - **rusage**[*rusage_string*]
- Ordering - **order**[*order_string*]
- Locality - **span**[*span_string*]
- Same - **same**[*same_string*]
- CU - **cu**[*cu_string*]

Span and same sections are used for parallel jobs

Resource Requirements Examples

Example 1. Select execution host candidates that have at least 2GB free RAM

```
$ bsub -R "select[mem>2GB]" myJob
```

Example 2. Select execution host candidates with Infiniband interconnect

```
$ bsub -R "defined(ib)" myJob
```

Example 3. Select candidate hosts that have 1000MB free swap and order by amount of available memory

```
$ bsub -R "select[swp>1000] order[mem]" myJob
```

Resource Requirements Examples – Parallel jobs example of usage

Example 4. Candidate hosts should have min 500MB free RAM, job will reserve 400MB RAM.

```
$ bsub -n 4 -R "select[mem>500] rusage[mem=400]" myJob
```

Example 5. All slots required for a parallel job should reside on the same host

```
$ bsub -n 4 -R "span[hosts=1]" parallelJob
```

Example 6. 12-CPU parallel job should run on up to 4 CPUs per host , all candidate hosts must have the same CPU model

```
$ bsub -n 12 -R "span[ptile=4] same[model]" parallelJob
```

Array Jobs – example of usage

- A job array is submitted using the syntax:
job_name_spec[index | start_index-end_index:step]

```
[vitalyg@access mathematica]$ cat array_math.job
#BSUB -q new-short
#BSUB -J mathematica[1-10]
#BSUB -o results.%J
#BSUB -e results.%J
math -script fullCluster.m ${LSB_JOBINDEX}
[vitalyg@access mathematica]$ bsub < array_math.job
Memory reservation is (MB): 2048
Memory Limit is (MB): 2048
Job <618114> is submitted to queue <new-short>.
[vitalyg@access mathematica]$ bjobs
```

JOBID	USER	STAT	QUEUE	FROM_HOST	EXEC_HOST	JOB_NAME	SUBMIT_TIME
618114	vitalyg	RUN	new-short	access.wexa	cn145.wexac	*matica[1]	Jan 22 13:22
618114	vitalyg	RUN	new-short	access.wexa	cn145.wexac	*matica[2]	Jan 22 13:22
618114	vitalyg	RUN	new-short	access.wexa	cn145.wexac	*matica[3]	Jan 22 13:22
618114	vitalyg	RUN	new-short	access.wexa	cn145.wexac	*matica[4]	Jan 22 13:22
618114	vitalyg	RUN	new-short	access.wexa	cn145.wexac	*matica[5]	Jan 22 13:22
618114	vitalyg	RUN	new-short	access.wexa	cn145.wexac	*matica[6]	Jan 22 13:22
618114	vitalyg	RUN	new-short	access.wexa	cn145.wexac	*matica[7]	Jan 22 13:22
618114	vitalyg	RUN	new-short	access.wexa	cn145.wexac	*matica[8]	Jan 22 13:22
618114	vitalyg	RUN	new-short	access.wexa	cn145.wexac	*matica[9]	Jan 22 13:22
618114	vitalyg	RUN	new-short	access.wexa	cn145.wexac	*matica[10]	Jan 22 13:22

```
[vitalyg@access mathematica]$
```

Array Jobs

```
[vitalyg@access mathematica]$ cat array_math.job
#BSUB -q new-short
#BSUB -J mathematica[1-10:2]
#BSUB -o results.%J
#BSUB -e results.%J
math -script fullCluster.m ${LSB_JOBINDEX}
[vitalyg@access mathematica]$ bsub < array_math.job
Memory reservation is (MB): 2048
Memory Limit is (MB): 2048
Job <618115> is submitted to queue <new-short>.
[vitalyg@access mathematica]$ bjobs
```

JOBID	USER	STAT	QUEUE	FROM_HOST	EXEC_HOST	JOB_NAME	SUBMIT_TIME
618115	vitalyg	PEND	new-short	access.wexa		*matica[1]	Jan 22 13:27
618115	vitalyg	PEND	new-short	access.wexa		*matica[3]	Jan 22 13:27
618115	vitalyg	PEND	new-short	access.wexa		*matica[5]	Jan 22 13:27
618115	vitalyg	PEND	new-short	access.wexa		*matica[7]	Jan 22 13:27
618115	vitalyg	PEND	new-short	access.wexa		*matica[9]	Jan 22 13:27

```
[vitalyg@access mathematica]$ █
```

bjobs – View Jobs

- a Display information about jobs in all states, including recently finished jobs
- A Displays summarized information about job arrays
- d Display information about jobs that finished recently
- l | -w Display information in long or wide format
- p Display information about pending jobs
- r Display information about running jobs
- g job_group Display information about jobs in specified group
- J job_name Display information about specified job or array
- q queue Display information about jobs in specified queue
- u user Display information about jobs for specified users/groups

bjobs – Example 1

```
[[vitalyg@access ~]$ bjobs
```

JOBID	USER	STAT	QUEUE	FROM_HOST	EXEC_HOST	JOB_NAME	SUBMIT_TIME
478504	vitalyg	RUN	short	access.wexa	cn165.wexac	example3	Jan 17 10:47
478502	vitalyg	RUN	new-all.q	access.wexa	cn165.wexac	example1	Jan 17 10:47
478503	vitalyg	USUSP	new-all.q	access.wexa	cn165.wexac	example2	Jan 17 10:47
478505	vitalyg	RUN	short	access.wexa	cn160.wexac	example4	Jan 17 10:48
478506	vitalyg	RUN	short	access.wexa	cn160.wexac	example5	Jan 17 10:48
478507	vitalyg	RUN	short	access.wexa	cn160.wexac	example6	Jan 17 10:48
478509	vitalyg	PEND	short	access.wexa		example7	Jan 17 10:50

```
[vitalyg@access ~]$ █
```

bjobs – Example 2 : detailed job view

```
[vitalyg@access ~]$ bjobs -l 478509

Job <478509>, Job Name <example7>, User <vitalyg>, Project <default>, Status <P
END>, Queue <short>, Command <sleep 999>
Tue Jan 17 10:50:42: Submitted from host <access.wexac.weizmann.ac.il>, CWD <$H
OME>, Output File </dev/null>, Re-runnable, Requested Reso
urces <rusage[mem=2048]>, Specified Hosts <cn149>;

MEMLIMIT
  2 G
PENDING REASONS:
There are no suitable hosts for the job;

SCHEDULING PARAMETERS:
loadSched  r15s  r1m  r15m  ut      pg      io      ls      it      tmp      swp      mem
loadStop   -      -      -      -      -      -      -      -      -      -      -

          ngpus ngpus_shared ngpus_excl_t ngpus_excl_p ngpus_prohibited
loadSched  -      -      -      -      -
loadStop   -      -      -      -      -

          gpu_temp0 gpu_ecc0 gpu_temp1 gpu_ecc1 gpu_temp2 gpu_ecc2  nmics
loadSched  -      -      -      -      -      -      -
loadStop   -      -      -      -      -      -      -

          mic_temp0 mic_temp1 mic_freq0 mic_freq1 mic_power0 mic_power1
loadSched  -      -      -      -      -      -
loadStop   -      -      -      -      -      -

          mic_freemem0 mic_freemem1 mic_util0 mic_util1 mic_ncores0 mic_ncores1
loadSched  -      -      -      -      -      -
loadStop   -      -      -      -      -      -

RESOURCE REQUIREMENT DETAILS:
Combined: select[(type = any ) && (type == any)] order[r15s:pg] rusage[mem=204
8.00]
Effective: -
```

bhist – Jobs History options

- a** Display information about all jobs
- b|-l|-w** Display information in brief, long, or wide format
- d** Display information about finished jobs
- p** Display information about pending jobs
- s** Display information about suspended jobs
- t** Display job events chronologically
- C|-D|-S|-T *start_time,end_time***
Display information about completed, dispatched, submitted, or all jobs in specified time window
- q *queue*** Display information about jobs submitted to specified queue
- u *username|all*** Display information about jobs submitted by user or all users

bhist – Example 1

```
[vitalyg@access ~]$ bhist
```

```
Summary of time in seconds spent in various states:
```

JOBID	USER	JOB_NAME	PEND	PSUSP	RUN	USUSP	SSUSP	UNKWN	TOTAL
478502	vitalyg	example1	0	0	539	0	0	0	539
478503	vitalyg	example2	1	0	46	486	0	0	533
478504	vitalyg	example3	1	0	520	0	0	0	521
478505	vitalyg	example4	0	0	471	0	0	0	471
478506	vitalyg	example5	1	0	467	0	0	0	468
478507	vitalyg	example6	0	0	464	0	0	0	464
478509	vitalyg	example7	342	0	0	0	0	0	342

```
[vitalyg@access ~]$ █
```

bhist – Example 2 : detailed job information

```
[vitalyg@access ~]$ bhist -l 478502
```

```
Job <478502>, Job Name <example1>, User <vitalyg>, Project <default>, Command <sleep 999>
```

```
Tue Jan 17 10:47:25: Submitted from host <access.wexac.weizmann.ac.il>, to Queue <new-all.q>, CWD <$HOME>, Output File </dev/null>, Re-runnable, Requested Resources <rusage[mem=2048]>;
```

```
MEMLIMIT  
2 G
```

```
Tue Jan 17 10:47:25: Dispatched 1 Task(s) on Host(s) <cn165.wexac.weizmann.ac.il>, Allocated 1 Slot(s) on Host(s) <cn165.wexac.weizmann.ac.il>, Effective RES_REQ <select[((type = any ) && (type = any))] order[r15s:pg] rusage[mem=2048.00] >;
```

```
Tue Jan 17 10:47:25: Starting (Pid 13646);
```

```
Tue Jan 17 10:47:25: Running with execution home </home/labs/training/vitalyg>, Execution CWD </home/labs/training/vitalyg>, Execution Pid <13646>;
```

```
Summary of time in seconds spent in various states by Tue Jan 17 10:58:15
```

PEND	PSUSP	RUN	USUSP	SSUSP	UNKWN	TOTAL
0	0	650	0	0	0	650



Manipulating Jobs

bkill - send signals to kill, suspend, or resume unfinished jobs

Tip: use JobID 0 to kill all your jobs

bmod - modify job submission options of a job

bpeek - display the stdout and stderr output of an unfinished job

bstop - suspend unfinished jobs

bresume - resume unfinished jobs

bswitch - switch jobs from one queue to another

Manipulating Jobs

```
[vitalyg@access ~]$ bsub -q new-all.q -oo output -eo error -J example sleep 999
Memory reservation is (MB): 2048
Memory Limit is (MB): 2048
Job <478532> is submitted to queue <new-all.q>.
[vitalyg@access ~]$ bjobs 478532
JOBID  USER  STAT  QUEUE          FROM_HOST  EXEC_HOST  JOB_NAME  SUBMIT_TIME
478532 vitalyg RUN   new-all.q     access.wexa cn169.wexac example   Jan 17 12:54
[vitalyg@access ~]$ bstop 478532
Job <478532> is being stopped
[vitalyg@access ~]$ bjobs 478532
JOBID  USER  STAT  QUEUE          FROM_HOST  EXEC_HOST  JOB_NAME  SUBMIT_TIME
478532 vitalyg USUSP new-all.q     access.wexa cn169.wexac example   Jan 17 12:54
[vitalyg@access ~]$ bresume 478532
Job <478532> is being resumed
[vitalyg@access ~]$ bjobs 478532
JOBID  USER  STAT  QUEUE          FROM_HOST  EXEC_HOST  JOB_NAME  SUBMIT_TIME
478532 vitalyg SSUSP new-all.q     access.wexa cn169.wexac example   Jan 17 12:54
[vitalyg@access ~]$ bjobs 478532
JOBID  USER  STAT  QUEUE          FROM_HOST  EXEC_HOST  JOB_NAME  SUBMIT_TIME
478532 vitalyg RUN   new-all.q     access.wexa cn169.wexac example   Jan 17 12:54
```



Manipulating Jobs

Example 2

```
[[vitalyg@access ~]$ bjobs
JOBID  USER  STAT  QUEUE          FROM_HOST  EXEC_HOST  JOB_NAME  SUBMIT_TIME
478529  vitalyg  RUN   new-all.q     access.wexa  cn146.wexac  example  Jan 17 12:41
478530  vitalyg  PEND  new-all.q     access.wexa           example  Jan 17 12:43
[[vitalyg@access ~]$ bmod -q new-short 478530
Parameters of job <478530> are being changed
[[vitalyg@access ~]$ bjobs
JOBID  USER  STAT  QUEUE          FROM_HOST  EXEC_HOST  JOB_NAME  SUBMIT_TIME
478530  vitalyg  RUN   new-short     access.wexa  cn146.wexac  example  Jan 17 12:43
478529  vitalyg  RUN   new-all.q     access.wexa  cn146.wexac  example  Jan 17 12:41
[[vitalyg@access ~]$ █
```


Cluster Query Commands

- lsinfo - displays load sharing configuration information
- lshosts - displays hosts and their static resource information
- lsload - displays load information for hosts
- bhosts - displays hosts and their static and dynamic resources
- bqueues - displays information about queues
- bmgroupp - displays information about host groups and compute units
- busers - displays information about users and user groups
- bugroupp - displays information about user groups

Query Commands

Example 1. Show resource information for compute nodes cn141 and cn142

```
[vitalyg@access ~]$ lshosts cn141 cn142
HOST_NAME      type    model  cpuf  ncpus  maxmem  maxswp  server  RESOURCES
cn141.wexac    X86_64 Intel_EM 60.0   16    63.8G  23.4G   Yes (mpichp4 intelmpi openmpi)
cn142.wexac    X86_64 Intel_EM 60.0   16    63.8G  23.4G   Yes (mpichp4 intelmpi openmpi)
```

Example 2 . Show load information for compute nodes cn141 and cn142

```
[vitalyg@access ~]$ lsload cn141 cn142
HOST_NAME      status  r15s   r1m   r15m   ut    pg   ls    it    tmp   swp   mem
cn142.wexac.wei  ok     1.0   1.0   1.0   3%   0.0  0  3834  18G  23.4G  54.3G
cn141.wexac.wei  ok     2.0  20.0  14.0  9%   0.0  0  3862  18G  23.4G  58.1G
```

Example 3. Show statistics for hosts cn141 and cn142

```
[vitalyg@access ~]$ bhosts cn141 cn142
HOST_NAME      STATUS      JL/U    MAX  NJOBS    RUN  SSUSP  USUSP  RSV
cn141.wexac.weizma ok          -      32   20      20    0      0      0
cn142.wexac.weizma ok          -      32   10      10    0      0      0
```

Query Commands

Example 4. Show queues

```
[vitalyg@access ~]$ bqueues
```

QUEUE_NAME	PRIO	STATUS	MAX	JL/U	JL/P	JL/H	NJOBS	PEND	RUN	SUSP
bio-mem	85	Open:Active	-	-	-	-	0	0	0	0
sorek-high-prio	80	Open:Active	-	6	-	-	0	0	0	0
bio	80	Open:Active	-	-	-	-	0	0	0	0
schwartz	80	Open:Active	-	-	-	-	0	0	0	0
feulf	80	Open:Active	-	-	-	-	1	1	0	0
bio-smrt	75	Open:Active	-	-	-	-	0	0	0	0
bio-guest	70	Open:Active	-	-	-	-	0	0	0	0
wicc-priority	65	Open:Active	-	-	-	-	0	0	0	0
fleishman-prior	60	Open:Active	-	40	-	-	0	0	0	0
hanna-priority	60	Open:Active	-	6	-	-	0	0	0	0
koren-priority	60	Open:Active	-	-	-	-	0	0	0	0
lancet-priority	60	Open:Active	-	6	-	-	0	0	0	0
schneidman-prio	60	Open:Active	-	60	-	-	0	0	0	0
all-high	60	Open:Active	-	1000	-	-	0	0	0	0
new-short	60	Open:Active	-	1024	-	-	0	0	0	0

Example 5. Display information about all hosts

```
[vitalyg@access ~]$ bhosts
```

HOST_NAME	STATUS	JL/U	MAX	NJOBS	RUN	SSUSP	USUSP	RSV
cn080.wexac.weizma	ok	-	24	10	10	0	0	0
cn081.wexac.weizma	ok	-	24	20	20	0	0	0
cn083.wexac.weizma	unavail	-	32	0	0	0	0	0
cn084.wexac.weizma	ok	-	32	30	30	0	0	0
cn103.wexac.weizma	ok	-	24	20	20	0	0	0
cn104.wexac.weizma	ok	-	24	20	20	0	0	0
cn105.wexac.weizma	ok	-	24	20	20	0	0	0
cn106.wexac.weizma	ok	-	24	20	20	0	0	0
cn107.wexac.weizma	ok	-	24	0	0	0	0	0
cn108.wexac.weizma	closed	-	24	20	20	0	0	0

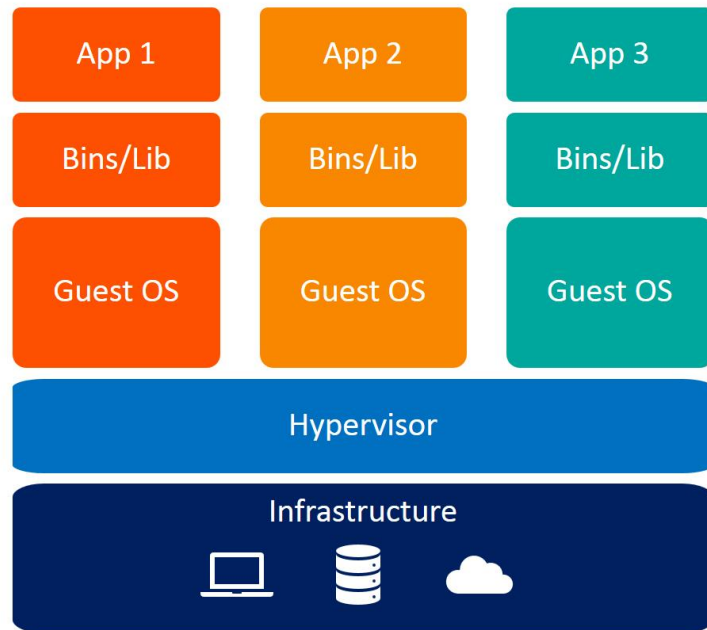
What Is A Docker Container?

- A Docker container is a mechanism for bundling a Linux application with all of its libraries, data files, and environment variables so that the execution environment is always the same, on whatever Linux system it runs and between instances on the same host.
- Unlike a VM which has its own isolated kernel, containers use the host system kernel. Therefore, all kernel calls from the container are handled by the host system kernel. DGX systems uses Docker containers as the mechanism for deploying deep learning frameworks.

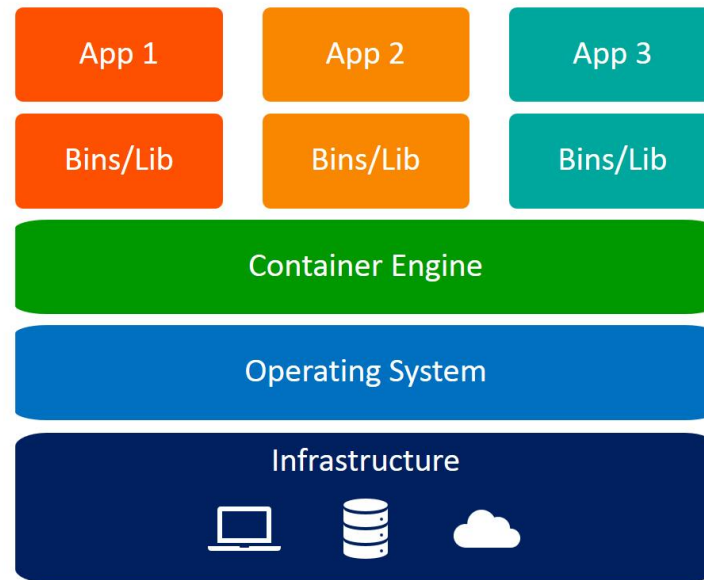
Why Use A Container?

- One of the many benefits to using containers is that you can install your application, dependencies and environment variables one time into the container image; rather than on each system you run on. In addition, the key benefits to using containers also include:
- Install your application, dependencies and environment variables one time into the container image; rather than on each system you run on.
- There is no risk of conflict with libraries that are installed by others.
- Containers allow use of multiple different deep learning frameworks, which may have conflicting software dependencies, on the same server.
- After you build your application into a container, you can run it on lots of other places, especially servers, without having to install any software.
- Legacy accelerated compute applications can be containerized and deployed on newer systems, on premise, or in the cloud.
- Specific GPU resources can be allocated to a container for isolation and better performance.
- You can easily share, collaborate, and test applications across different environments.
- Multiple instances of a given deep learning framework can be run concurrently with each having one or more specific GPUs assigned.
- Containers can be used to resolve network-port conflicts between applications by mapping container-ports to specific externally-visible ports when launching the container.

Virtual Machines vs. Containers



Virtual Machines



Containers

Docker Lifecycle

- DGXWS01 is used to build custom Docker images
- An image can be pulled and manipulated as required or built from a Dockerfile created by a user
- The image can be manipulated and then pushed to a private repository

Docker Cheat Sheet

ORCHESTRATE

Initialize swarm mode and listen on a specific interface
`docker swarm init --advertise-addr 10.1.0.2`

Join an existing swarm as a manager node
`docker swarm join --token <manager-token> 10.1.0.2:2377`

Join an existing swarm as a worker node
`docker swarm join --token <worker-token> 10.1.0.2:2377`

List the nodes participating in a swarm
`docker node ls`

Create a service from an image exposed on a specific port and deploy 3 instances
`docker service create --replicas 3 -p 80:80 --name web nginx`

List the services running in a swarm
`docker service ls`

Scale a service
`docker service scale web=5`

List the tasks of a service
`docker service ps web`

BUILD

Build an image from the Dockerfile in the current directory and tag the image
`docker build -t myapp:1.0 .`

List all images that are locally stored with the Docker engine
`docker images`

Delete an image from the local image store
`docker rmi alpine:3.4`

SHIP

Pull an image from a registry
`docker pull alpine:3.4`

Retag a local image with a new image name and tag
`docker tag alpine:3.4 myrepo/myalpine:3.4`

Log in to a registry (the Docker Hub by default)
`docker login my.registry.com:8000`

Push an image to a registry
`docker push myrepo/myalpine:3.4`



RUN

`docker run`
--rm remove container automatically after it exits
-it connect the container to terminal
--name web name the container
-p 5000:80 expose port 5000 externally and map to port 80
-v ~/dev:/code create a host mapped volume inside the container
alpine:3.4 the image from which the container is instantiated
/bin/sh the command to run inside the container

Stop a running container through SIGTERM
`docker stop web`

Stop a running container through SIGKILL
`docker kill web`

Create an overlay network and specify a subnet
`docker network create --subnet 10.1.0.0/24 --gateway 10.1.0.1 -d overlay mynet`

List the networks
`docker network ls`

List the running containers
`docker ps`

Delete all running and stopped containers
`docker rm -f $(docker ps -aq)`

Create a new bash process inside the container and connect it to the terminal
`docker exec -it web bash`

Print the last 100 lines of a container's logs
`docker logs --tail 100 web`

www.docker.com/getdocker

How to pull a prebuilt docker image from a repository

- <https://ngc.nvidia.com/>
- Pull the Docker image from NVIDIA NGC repository:

```
docker pull nvcr.io/nvidia/tensorflow:19.05-py3
```

```
root@dgxws01:~# docker pull nvcr.io/nvidia/tensorflow:19.05-py2
19.05-py2: Pulling from nvidia/tensorflow
7e6591854262: Already exists
089d60cb4e0a: Already exists
9c461696bc09: Already exists
45085432511a: Already exists
6ca460804a89: Already exists
2631f04ebf64: Already exists
86f56e03e071: Already exists
234646620160: Already exists
7f717cd17058: Already exists
e69a2ba99832: Already exists
bc9bca17b13c: Already exists
1870788e477f: Already exists
603e0d586945: Already exists
717dfedf079c: Already exists
1035ef613bc7: Already exists
c5bd7559c3ad: Already exists
d82c679b8708: Already exists
059d4f560014: Already exists
f3f14cff44df: Already exists
96502bde320c: Already exists
bc5bb9379810: Already exists
e4d8bb046bc2: Already exists
4e2187010a7c: Already exists
9d62684b94c3: Already exists
e70e61e48991: Already exists
67f66c5c2dbf: Downloading [=====] 26.99MB/76.22MB
a290186bda3b: Download complete
74711a0ac24c: Download complete
a2fff41d0a70: Downloading [=====] 1.182MB/5.352MB
30b0cf6a53e9: Downloading [=>] 1.617MB/52.75MB
787fc5f5c80a: Waiting
```

```
root@dgxws01:~# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
nvcr.io/nvidia/pytorch  19.05-py3          7e98758d4777       4 weeks ago        7.55GB
nvcr.io/nvidia/tensorflow  19.05-py2          5ce2c011e671       4 weeks ago        6.76GB
```

Build an image from a dockerfile

A dockerfile is a script that contains instructions to customize and configure a container from a base image

Here are some common commands:

- **FROM** is Mandatory as the first instruction. It denotes the base image to be built from. Use a tag to specify the image.
- **RUN** = runs commands after Docker image has been created.
- **CMD** = Run any command when Docker image is started. Use only one CMD instruction in a Dockerfile.
- **ENV** = Set container environment variables.
- **WORKDIR** = Sets the working directory for any RUN, CMD, ENTRYPOINT, COPY and ADD instructions.

Instructions can be found here: <https://docs.nvidia.com/ngc/ngc-user-guide/custcontfrm.html#custcontfrm>

Example using horovod and creating an image from a dockerfile (file is located in slide notes):

```
mkdir horovod-docker
cp /shareDB/wexac_workshop/Dockerfiles/Horovod/Dockerfile horovod-docker/Dockerfile
docker build -t horovod:latest horovod-docker
```

How to modify a docker image

There are two main ways to edit a docker image.

1. The best practice is to modify a dockerfile to include any changes you would require. This approach is preferred as one can easily document changes and better maintain the image.
2. Alternatively, one can use `docker commit` to create a new image from a currently open container.

```
root@ibdgx001:~# docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
6ac93b585313	horovod:latest	"/bin/bash"	8 days ago	Up 8 days	0.0.0.0:32768->22/tcp	horovod_master

```
root@ibdgx001:~# docker commit 6ac93b585313 ibdgx001:5000/horovod_master  
sha256:bf79c96f812e1c71c75dc7bb1cb4cf0ee8e9040238841e9b51a321e1332e55b9
```

```
root@ibdgx001:~# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
ibdgx001:5000/horovod_master	latest	bf79c96f812e	11 seconds ago	8.95GB
horovod	latest	ffe80b4ddef7	12 days ago	7.91GB

How to push an image to the local repository

- Tag the docker image:

```
docker tag pytorch:19.05-py3 ibdgx001:5000/tensorflow-test
```

- Push the tagged docker image to the local repository:

```
docker push ibdgx001:5000/tensorflow-test
```

```
root@dgxws01:~# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
nvr.io/nvidia/pytorch 19.05-py3          7e98758d4777       4 weeks ago        7.55GB
nvr.io/nvidia/tensorflow 19.05-py2          5ce2c011e671       4 weeks ago        6.76GB
root@dgxws01:~# docker tag nvr.io/nvidia/tensorflow:19.05-py2 ibdgx001:5000/tensorflow-test
root@dgxws01:~# docker push ibdgx001:5000/tensorflow-test
The push refers to repository [ibdgx001:5000/tensorflow-test]
84ac6b166875: Pushed
805620e17ac4: Pushed
```

- Check:

```
Docker images
```

- You should see both the original image as well as the tagged image

```
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
nvr.io/nvidia/pytorch 19.05-py3          7e98758d4777       4 weeks ago        7.55GB
ibdgx001:5000/tensorflow-test latest             5ce2c011e671       4 weeks ago        6.76GB
nvr.io/nvidia/tensorflow 19.05-py2          5ce2c011e671       4 weeks ago        6.76GB
root@dgxws01:~# █
```

How to run an interactive job

Run an interactive job by using the `-Is` option:

```
bsub -q waic-interactive -gpu num=1:j_exclusive=yes -Is /bin/bash
```

Opening an IDE:

```
bsub -q waic-interactive -gpu num=1:j_exclusive=yes -Is /bin/bash
source /etc/profile
module load anaconda
export DISPLAY=<your local computer ip here>:0
spyder
```

How to run a job using a pulled image

Create a file to run the job:

```
cat useCase1
#BSUB -J useCase1
#BSUB -env LSB_CONTAINER_IMAGE=ibdgx001:5000/tensorflow-test
#BSUB -app nvidia-gpu
#BSUB -gpu num=2:j_exclusive=yes
#BSUB -q waic-short
#BSUB -oo test_useCase_out.log
#BSUB -eo test_useCase_err.log
mpiexec -np 2 python /workspace/nvidia-examples/cnn/resnet.py --layers=50 --
precision=fp16 --log_dir=output/resnet50_useCase1
```

How to run a job using a pulled image

Run the job:

```
cat useCase1 | bsub
```

You can check the job using bjobs/bpeek:

```
bjobs -q waic-short -J useCase1
```

```
bpeek -f -J test_useCase
```

```
[adamma@access ~]$ cat useCase1 | bsub
Memory reservation is (MB): 1024
Memory Limit is (MB): 1024
Job <401028> is submitted to queue <waic-short>.
[adamma@access ~]$ bpeek -f -J useCase1
<< output from stdout >>
```

```
=====
NVIDIA Release 19.05 (build 6390159)
TensorFlow Version 1.13.1
```

```
Container image Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
Copyright 2017-2019 The TensorFlow Authors. All rights reserved.
```

```
Various files include modifications (c) NVIDIA CORPORATION. All rights reserved.
NVIDIA modifications are covered by the license terms that apply to the underlying project or file.
```

```
NOTE: Legacy NVIDIA Driver detected. Compatibility mode ENABLED.
```

```
NOTE: Detected MOFED driver 4.0-1.0.1; version automatically upgraded.
```

```
NOTE: MOFED driver was detected, but nv_peer_mem driver was not detected.
Multi-node communication performance may be reduced.
```

```
('PY', '2.7.12 (default, Nov 12 2018, 14:36:49) \n[GCC 5.4.0 20160609]')
('TF', '1.13.1')
('PY', '2.7.12 (default, Nov 12 2018, 14:36:49) \n[GCC 5.4.0 20160609]')
('TF', '1.13.1')
```

```
Script arguments:
```

```
--layers 50
--predict False
--precision fp16
--batch_size 256
--log_dir output/resnet50_useCase01
--iter_unit epoch
--num_iter 90
--use_xla False
--display_every 10
```

```
Training
```

```
Training
```

Step	Epoch	Img/sec	Loss	LR
1	1.0	47.5	7.589	8.559 2.00000
10	10.0	627.8	4.946	5.918 1.62000
20	20.0	1457.6	0.338	1.314 1.24469
30	30.0	1467.3	0.056	1.037 0.91877
40	40.0	1470.8	0.065	1.047 0.64222
50	50.0	1466.6	0.045	1.027 0.41506
60	60.0	1429.0	0.013	0.995 0.23728
70	70.0	1454.5	0.001	0.982 0.10889
80	80.0	1494.3	0.000	0.980 0.02988
90	90.0	1157.7	0.000	0.980 0.00025

```
[adamma@access ~]$
```

Submitting an array job

Array Script:

```
./seq_arr.sh -f c.sh -e 10
```

The `-e` must match the elements in the array. In this case (c.sh) there are ten, `"#BSUB -J test[1-10]"`.

bsub file(c.sh):

```
#BSUB -q new-short
#BSUB -J test[1-10]
#BSUB -o out/out.%J
#BSUB -e err/err.%J
#BSUB -H
sleep 5
```

The `-H` option is very important!

(Holds the job in the PSUSP state when the job is submitted).

Usage:
`seq_arr.sh -f bsub_file -e max_element -c bsub_command`

Examples:

Send bsub file with sequence array:
`seq_arr.sh -f my_bsub.sh -e 4`

Send the submit in the command line:
`seq_arr.sh -e 5 -c "bsub -H -J "test[1-5]" sleep 5"`

[adamma@access4 ~]\$./seq_arr.sh -f c.sh -e 10

NOTE: the bsub command must have the `-H` option!!!

```
c.sh
Job 130511
Parameters of job <130511[2]> are being changed
Parameters of job <130511[3]> are being changed
Parameters of job <130511[4]> are being changed
Parameters of job <130511[5]> are being changed
Parameters of job <130511[6]> are being changed
Parameters of job <130511[7]> are being changed
Parameters of job <130511[8]> are being changed
Parameters of job <130511[9]> are being changed
Parameters of job <130511[10]> are being changed
Job <130511>: Operation is in progress
[adamma@access4 ~]$ bjobs
JOBID  USER  STAT  QUEUE          FROM_HOST  EXEC_HOST  JOB_NAME  SUBMIT_TIME
130511  adamma PSUSP  new-short     access4.wex  test[1]    Jun 13 13:53
130511  adamma PSUSP  new-short     access4.wex  test[2]    Jun 13 13:53
130511  adamma PSUSP  new-short     access4.wex  test[3]    Jun 13 13:53
130511  adamma PSUSP  new-short     access4.wex  test[4]    Jun 13 13:53
130511  adamma PSUSP  new-short     access4.wex  test[5]    Jun 13 13:53
130511  adamma PSUSP  new-short     access4.wex  test[6]    Jun 13 13:53
130511  adamma PSUSP  new-short     access4.wex  test[7]    Jun 13 13:53
130511  adamma PSUSP  new-short     access4.wex  test[8]    Jun 13 13:53
130511  adamma PSUSP  new-short     access4.wex  test[9]    Jun 13 13:53
130511  adamma PSUSP  new-short     access4.wex  test[10]   Jun 13 13:53
[adamma@access4 ~]$ bjobs
JOBID  USER  STAT  QUEUE          FROM_HOST  EXEC_HOST  JOB_NAME  SUBMIT_TIME
130511  adamma RUN    new-short     access4.wex  cn293.wexac  test[2]    Jun 13 13:53
130511  adamma PEND  new-short     access4.wex  test[3]    Jun 13 13:53
130511  adamma PEND  new-short     access4.wex  test[4]    Jun 13 13:53
130511  adamma PEND  new-short     access4.wex  test[5]    Jun 13 13:53
130511  adamma PEND  new-short     access4.wex  test[6]    Jun 13 13:53
130511  adamma PEND  new-short     access4.wex  test[7]    Jun 13 13:53
130511  adamma PEND  new-short     access4.wex  test[8]    Jun 13 13:53
130511  adamma PEND  new-short     access4.wex  test[9]    Jun 13 13:53
130511  adamma PEND  new-short     access4.wex  test[10]   Jun 13 13:53
[adamma@access4 ~]$ bjobs
JOBID  USER  STAT  QUEUE          FROM_HOST  EXEC_HOST  JOB_NAME  SUBMIT_TIME
130511  adamma RUN    new-short     access4.wex  cn442.wexac  test[5]    Jun 13 13:53
130511  adamma PEND  new-short     access4.wex  test[6]    Jun 13 13:53
130511  adamma PEND  new-short     access4.wex  test[7]    Jun 13 13:53
130511  adamma PEND  new-short     access4.wex  test[8]    Jun 13 13:53
130511  adamma PEND  new-short     access4.wex  test[9]    Jun 13 13:53
130511  adamma PEND  new-short     access4.wex  test[10]   Jun 13 13:53
[adamma@access4 ~]$
```


UseCase1 + Grafana

```
cat << EOF_useCase1 > ~/useCase1

#BSUB -J useCase1

#BSUB -env LSB_CONTAINER_IMAGE=nvcr.io/nvidia/tensorflow:19.03-py3

#BSUB -app nvidia-gpu

#BSUB -gpu num=2:j_exclusive=yes

#BSUB -q waic-short

#BSUB -oo useCase1_out.log

#BSUB -eo useCase1_err.log

mpiexec -np 2 python \
/workspace/nvidia-examples/cnn/resnet.py \
--layers=50 \
--precision=fp16 \
--log_dir=output/resnet50_useCase1

EOF_useCase1
```

UseCase1 + Grafana

Instructions	Commands	Expected Results
Submit a job	<code>bsub < ~/useCase1</code>	Job is submitted to queue waic-short
Verify the job is enqueued and starts running	<code>bjobs -q waic-short -J useCase1</code>	Job useCase1 appears in the output, the job state is RUN
Verify the job is running correctly	<code>bpeek -f -J useCase1</code>	Job progresses as expected
Verify the job executed correctly	Inspect file <code>useCase1_out.log</code>	<p>File contains lines:</p> <pre> Step Epoch Img/sec Loss LR 1 1.0 156.8 7.726 8.697 2.00000 10 10.0 1289.8 5.758 6.729 1.62000 20 20.0 4417.6 2.035 3.010 1.24469 30 30.0 6297.6 0.010 0.991 0.91877 40 40.0 6303.1 0.000 0.976 0.64222 50 50.0 6339.3 0.000 0.968 0.41506 </pre> <p>Average Img/Sec value for steps 30+ should be above 6200</p>
Navigate to Grafana UI	Navigate to <code>http://waicmgmt01.wexac.weizmann.ac.il:30200</code>	Successful navigation to Grafana Home Dashboard
Navigate to GPU Nodes dashboard and select the node running . Check node utilization		Per-GPU utilization and Total GPU utilization is above 90%



UseCase2

```
cat << EOF_useCase2 > ~/useCase2
#BSUB -J useCase2
#BSUB -env "LSB_CONTAINER_IMAGE=ibdgx001:5000/usecase2_1"
#BSUB -app nvidia-gpu
#BSUB -gpu num=1:j_exclusive=yes
#BSUB -q gpu-short
#BSUB -oo useCase2_out.log
#BSUB -eo useCase2_err.log
export MLM_LICENSE_FILE=1700@license.weizmann.ac.il
cd /shareDB/Wexac_workshop/useCase2_code/usecase2/Matlab/exp_code
echo; echo; echo "Running MATLAB"
echo "Test 2.1.1:"
/usr/local/bin/matlab -nodisplay -nosplash -nodesktop -r
"eval_validation_set_exact('0', 0, 1); exit"
echo "Test 2.1.2:"
/usr/local/bin/matlab -nodisplay -nosplash -nodesktop -r
"eval_validation_set_exact('0', 16, 1); exit"
echo "Test 2.1.3:"
/usr/local/bin/matlab -nodisplay -nosplash -nodesktop -r
"eval_validation_set_exact('16', 0, 1); exit"
echo "Test 2.1.4:"
/usr/local/bin/matlab -nodisplay -nosplash -nodesktop -r
"eval_validation_set_exact('16', 16, 1); exit"
echo "Test 2.1.5:"
/usr/local/bin/matlab -nodisplay -nosplash -nodesktop -r
"eval_validation_set_exact('64_32_16_8_4_0', 0, 1); exit"
echo "Test 2.1.6:"
```

```
/usr/local/bin/matlab -nodisplay -nosplash -nodesktop -r
"eval_validation_set_exact('64_32_16_8_4_0', 16, 1); exit"
echo; echo; echo "Running PYTHON CONVERT"
echo "Test 2.2.1:"
cd /home/labs/testing/class5/Documents/MATLAB/usecase2/python
python3 convert_columbiagaze_imdb_mat_to_pkl.py
echo; echo; echo "Running PYTHON Imagenet ResNet50"
echo "Test 2.3.1:"
python3 imagenet_resnet50_readout.py --train-res 'hi' --test-blur 0 --cond
'eyes' --layer 3 --input-type 'face_wo_chinrest' --gpu 1 --batch-size 10
echo "Test 2.3.2:"
python3 imagenet_resnet50_readout.py --train-res 'hi' --test-blur 0 --cond
'head' --layer 3 --input-type 'face_wo_eyes' --gpu 1 --batch-size 10
echo "Test 2.3.3:"
python3 imagenet_resnet50_readout.py --train-res 'low' --test-blur 0 --cond
'eyes' --layer 3 --input-type 'face_wo_eyes' --gpu 1 --batch-size 10
echo "Test 2.3.4:"
python3 imagenet_resnet50_readout.py --train-res 'low' --test-blur 16 --cond
'head' --layer 3 --input-type 'eyes' --gpu 1 --batch-size 10
EOF_useCase2
```

UseCase2 - Matlab and PyTorch under a common environment

Instructions	Commands	Expected results
Verify custom image exists	<code>docker image ls testing:usecase2</code>	Image testing:usecase2 exists
Tag image for upload to repository	<code>docker tag testing:usecase2 ibdgx001:5000/usecase2_1</code>	Successful execution
Upload image to cloud repository	<code>docker push ibdgx001:5000/usecase2_1</code>	All layers are pushed successfully

UseCase2 - Matlab and PyTorch under a common environment

Instructions	Commands	Expected results	
Submit a job	<code>bsub < ~/useCase2</code>	Job is submitted to queue waic-short	<pre>[06-May-2019 10:45:08] Evaluating 168 trials (3 batches, 60 images per batch) [06-May-2019 10:45:08] Running batch #1 [06-May-2019 10:45:10] Running batch #2 [06-May-2019 10:45:11] Running batch #3 Mean classification (LEFT/RIGHT) eyes accuracy is 98.21% [06-May-2019 10:45:12] Evaluating 168 trials (3 batches, 60 images per batch) [06-May-2019 10:45:12] Running batch #1 [06-May-2019 10:45:13] Running batch #2 [06-May-2019 10:45:14] Running batch #3 Mean classification (LEFT/RIGHT) head accuracy is 100.00% Accuracy for sub-tests is: 2.1.1 - 100% 2.1.2 - above 70% 2.1.3 - above 80% 2.1.4 - above 95% 2.1.5 - 100% 2.1.6 - 100%</pre>
Verify the job is enqueued and starts running	<code>bjobs -q waic-short -J useCase2</code>	Job useCase2 appears in the output, the job state is RUN	
Verify the job is running correctly	<code>bpeek -f -J useCase2</code>	Job progresses as expected	
Verify Matlab part of the job executed correctly	Inspect <code>useCase2_out.log</code> file	Matlab runs with outputs similar to the following: →	
Verify "python convert" part of the job executed successfully	Inspect <code>useCase2_out.log</code> file	Log contains following output: Running PYTHON CONVERT Test 2.2.1: convert_columbiagaze_imdb_mat_to_pkl SUCCESSFUL!	
Verify "python imagenet resnet50" part of the job executed successfully			

UseCase3 - Intensive I/O – video dataset

```
cat << EOF_useCase3 > ~/useCase3
#BSUB -J useCase3
#BSUB -env LSB_CONTAINER_IMAGE=nvcr.io/weizmann1/waic:usecase3
#BSUB -app nvidia-gpu
#BSUB -gpu num=2:j_exclusive=yes
#BSUB -q gpu-short
#BSUB -oo useCase3_out.log
#BSUB -eo useCase3_err.log
export
PATH=/opt/conda/bin:/usr/local/mpi/bin:/usr/local/nvidia/bin:/usr/local/cuda/bin:/usr/local/sbin:/usr/local/bin:${PATH}
cd useCase3_code
/opt/conda/bin/python train.py --t_win 31 --epochs 2 --tag useCase3 -b 8
EOF_useCase3
```

UseCase3 - Intensive I/O – video dataset

Instructions	Commands	Expected results
First copy code to home directory: <code>cp -R /shareDB/wexac_workshop/useCase3_code/ ~/</code>		
Submit a job	<code>bsub < ~/useCase3</code>	Job is submitted to queue waic-short
Verify the job is enqueued and starts running	<code>bjobs -q waic-short -J useCase3</code>	Job useCase3 appears in the output, the job state is RUN
Verify the job is running correctly	<code>bpeek -f -J useCase3</code>	Job progresses as expected. After running for about 5 minutes useCase3_err.log file's last line looks like: train 0: 1% 8/1000 [05:21<7:24:43, 26.90s/it]
Verify the job executed correctly	Wait till the job finishes running (should take about 6-7 hours). Inspect useCase3_*.log files	useCase3_err.log file doesn't contain any significant errors. Any errors related to qsub can be ignored. useCase2_out.log file contains expected job execution logs

Troubleshooting for usecases

For the use cases we need to copy the code to our home directory (run from your specific home directory):

```
cp -R /shareDB/wexac_workshop/useCase3_code/ ~/
```

If output/ exists after running useCase1 remove it and try running useCase1 again

```
rm -R output/
```


WEXAC Web Resources

<http://www.weizmann.ac.il/hpc>

http://www.weizmann.ac.il/DIS/sites/DIS/files/uploads/it/wexac_training_session.pdf

<https://insightiq.weizmann.ac.il>

<http://master-ops.wexac.weizmann.ac.il/ganglia/>

<http://lsfutil.wexac.weizmann.ac.il>

<https://www.facebook.com/HpcAtWeizmann/>

http://wiki.weizmann.ac.il/ai_wiki/index.php/WAIC_cluster

Join the WhatsApp group!

<https://chat.whatsapp.com/05qVVCCcR8v9234vit28gc>

Questions ?





Thank you !