CHTC Student Handbook

Edition 0.1

Center for High Throughput Computing

July 7, 2017

Contents

1 Student Work 101

1.1 Tasks and Workflow

As a student worker for the CHTC you will be responsible for a large variety of tasks to help keep the CHTC Cluster up and running, but there a few basic things you will find yourself doing most often. Our primary job is to fix individual machines in the cluster that go down. Most often machines can be brought down by kernel panics, networking issues, or hardware issues. To fix a machine, we take a simple approach to diagnose the issue and come up with a solution.

1. Diagnose the problem. You might need to plug into the machine with a console, or SSH into it if it is still online. If a machine kernel panics it will often print a memory stack trace to the console and freeze up. You will need to physically reboot the machine to proceed.
2. Find a solution. Often times rebooting the machine will clear up minor problems such as kernel panics or networking issues.
    - Kernel Panic (crash) - reboot, check server for errors.
    - Networking issue - reboot machine or restart networking, then go to advanced networking troubleshooting if still needed.
    - Hardware issue - diagnose the hardware issue and fix it. (Replace a bad disk, make note of a bad RAID card, etc)
3. Fix the problem and confirm that condor is running jobs again.
    - his might mean rebooting the machine, rebuilding the machine, replacing a bad disk and rebuilding, or something else entirely. Fix it if possible. If not - make note of the issue in the ticket and await further instructions.

Some other tasks you might be assigned include, but are not limited to: archiving condor releases to DVD, building new execute machines, working with Dell/Cisco Tech Support to get replacement parts, moving servers, a nd other random tasks assigned by full time CHTC staff.

1.2 RT Ticket System

We use the Request Tracker (RT) Ticket System for managing work at the CHTC. crt.cs.wisc.edu is the web address to access our Request Tracker.This will be your home for managing work tickets. The basic flow is: You are assigned a ticket, click on the ticket name to open it. By default, your 10 highest priority, or newest, tickets will be listed on your home page. You can view all of your tickets by clicking the "10 Highest Priority Tickets I Own" link (counter-intuitive, I know).

Once you have selected a ticket, you can read all correspondence related to that particular issue. Under the "actions" button (top right) the most common things you will do are "reply" and "resolve." Whenever you make changes to a machine, make sure to log what you did in the ticket. When the machine is fixed or the task is completed, you can resolve the ticket.

1.3 Monitoring Nodes

You can monitor the cluster on monitor0.chtc.wisc.edu. This has links to Icinga, our host monitoring application, Grafana, and Ganglia. All of these are extremely powerful tools that can be used to monitor the status of machines in the cluster and diagnose problems with individual nodes.

1.4 How to Use This Guide

- Any line or section beginning with "$" is a command line operation. Enter it directly as it appears, without the preceding "$".
- Any bracketed item (eg. [System Name]) needs to be replaced with the corresponding entity. (eg. change [System Name] $\rightarrow$ e1000.chtc.wisc.edu).

- Following the step-by-step walkthroughs to complete most basic tasks and use this as a reference manual.
- The appendix includes a complete list of node locations, useful for location a machine for physical changes (hard reboot, HDD swap, etc).
- If you find an error in this guide, or you want to contribute, head on over to https://github.com/dcchambers/CHTC-Student-Handbook and file a bug report or submit a pull request with your own changes!

2 Rebuilding a Node

Rebuilding nodes is one of the primary responsibilities of student CHTC employees. Here's a step-by-step walkthrough for you.

1. Enable netboot in cobbler.
   (a) Via Command Line:
      i. SSH into wid-service-1.chtc.wisc.edu
      ii. `$ sudo cobbler system edit --name=[server name] --netboot-enabled=True`
      iii. Check that netboot is enabled → `$ sudo cobbler system report --name=[server name]`
      iv. Sync Cobbler → `$ sudo cobbler sync`
   (b) Via Web Interface
      i. Open Cobbler in your web browser
      ii. Click on "Systems" under the Configuration tab on the left side.
      iii. Find the node you want to rebuild and click on it's name in the list to open the node configuration tool.
      iv. Alternatively, navigate directly to wid-service-1.chtc.wisc.edu/cobbler_web/system/edit/[node name]
      v. Click on the "general" drop down button to reveal more options
      vi. Check the "enable netboot" option
      vii. Hit "save" on the bottom of the page
      viii. When you are redirected to the cobbler system list page, hit "Sync" Under the Actions tab on the left side.
      ix. Once you get a popup notification on the top-right of the screen, the sync has complete. This may take a few seconds.
   (c) Make sure you have the correct profile enabled in Cobbler as well. If you are doing a standard rebuild you probably won't have to change it, but make sure if it's a multi-disk execute node that it is set to the correct SL66 Exec profile.
2. Reboot the machine and netboot it.
   (a) Most machines are set to netboot by default, meaning if you reboot them they will search for a netboot entry and if they find it, they will netboot automatically. If a machine is not netbooting automatically, you may need to press a button on the keyboard (Often F12) when it POSTS in order to force it to netboot. If you have tried these and it still won't boot from the network, go into the BIOS and change the boot order to set netboot as boot priority #1.
   (b) If done correctly, it should launch the Scientific Linux installer. Once you see this is happening, move on to the next step.
   (c) To diagnose problems with the boot process, when the loading bar appears, you can press Alt+d to display a verbose boot screen.
3. Run Puppet.
   (a) We are going to need to run puppet once the machine is rebuilt in order to configure it.
   (b) While the machine is rebuilding, SSH into wid-service-1.chtc.wisc.edu
   (c) Run this command: `$ sudo puppetca -c [node name]` on wid-service-1.
   (d) If you do this before the machine finishes rebuilding, it may run puppet automatically when it rebuilds. To see if it is doing this: When the machine is booting Scientific Linux, hit an arrow key ← or →, or ALT+D on the console keyboard to view the verbose boot log. If the log is paused on "`Starting: anamon... [OK]` " for a while, that mean's it's running puppet. Good job! Once it finishes booting now, you should be able to log in with your username.
   (e) If the machine does not run puppet automatically, connect a console to the machine and log in as a root user (ask Admin for root login)

(f) ON THE TARGET NODE: run `$ sudo rm -rvf /var/lib/puppet/ssl` (After clearing the puppet files from wid-service-1 in the previous steps). DO NOT RUN THIS ON wid-service-1.

(g) After clearing the puppet files from wid-service-1 in the previous steps, run `$ sudo rm -rvf /var/lib/puppet/ssl` on the target node. DO NOT RUN THIS ON wid-service-1.

(h) Then run `$ sudo puppetd -tv --configtimeout=1000`

(i) Note: Puppet will not run if networking is broken (try to restart networking or reboot the machine if this is the case) or the system clock is broken. To set the clock run: `$ rdate -s ntp.doit.wisc.edu`

4. Log in and confirm condor is running.

(a) Once puppet finishes running, you should be able to log in with your own username

(b) Confirm condor is running: `$ condor_status $HOSTNAME`

(c) You should also see Condor running with `$ ps aux | grep condor`

3 Building a New Node

TODO - Elaborate.

Building a new node is very similar to rebuilding a current machine, with the added caveat of configuring the node in Cobbler and DHCP.

1. First, we need to create a new entry for the node in Cobbler. You should be provided a hostname and IP address to use for the new machine. You can create a new Cobbler entry via the Web interface or CLI.
    - Via Web Interface
        - Go to cobbler web.
        - Create a new system. The Easiest way to do this is to copy an existing execute node entry and change the relevant information.
        - Open up the new machine entry you just created and change the hostname, IP address, and MAC address. Make sure to do this under the correct network interface (generally eth0).
        - Make sure you selected the correct profile. Generally, you will use one of the SL6.6 profiles for execute nodes.
        - Enable netboot, save it, and sync cobbler once it's configured correctly.
    - Via CLI
        - TODO - CLI Intsructions.
2. Next, we need to create a DHCP entry for this machine.
    - DHCP is managed by different machines depending on the node location.
        - B240: cobbler-b240
        - 2360/CSL & WID Data Center: wid-service-1
        - 3370: host-6
    - SSH into the correct server and become root. Then open up /etc/dhcp/dhcpd.conf in VIM.
    - Copy an existing execute node entry and edit it to create a new entry with the correct IP and MAC address of the new machine. Save the file.
    - run `$ service dhcpd restart` - if an error comes up, you screwed up the .conf file.
3. You should now have the new node set up in Cobbler and DHCP. You can follow the standard steps to build a node at this point. Netboot the machine, run puppet, and make sure it's running jobs once it's finished.

4 Decommissioning a Node

1. Label the server for spare parts with ticket #, or SWAP it.
   - If you SWAP a server, remove the HDDs and any hardware that might be useful (eg, RAM).
   - If it has a CSL Inventory Sticker, email the CSL with the Inventory number and tell them you are sending the server to SWAP.
   - Drop the server off in the SWAP area of the CS basement (by the elevators).
2. Remove it from the DHCP config.
   - SSH into the relevant DHCP server.
     - B240: cobbler-b240
     - 2360/CSL & WID Data Center: wid-service-1
     - 3370: host-6
   - Become root with `$ sudo -i`
   - Open up the dhcpd.conf file under /etc/dhcp/dhcpd.conf on the correct server.
   - Delete the entry for the server you are decommissioning.
   - Restart DHCP - Run `$ sudo service dhcpd restart` , make sure that it restarts OK. If not, you deleted something you shouldn't have.
3. Remove it's cobbler system object.
   - From the cobbler web interface, view the list of systems by going to https://wid-service-1.chtc.wisc.edu/cobbler_web/system/list
   - Locate the server on the list, and then press the "Delete" button next to it.
   - Sync cobbler. (Press the "Sync" button under actions.)
4. Remove it's puppet entry.
   - SSH into wid-service-1, and become root.
   - Open up the correct file in this directory: /etc/puppet/environments/production/modules/site/manifests/chtc/nodes/
     - B240: chtcexecsl66_b240.pp
     - 2360/CSL: chtcexecsl66_2360.pp
     - WID Data Center: chtcexecsl66_wid.pp
     - 3370: chtcexecsl66_3370a.pp
   - Delete the puppet entry for the server you are decommissioning.
5. Remove it's Icinga & Nagios entry.
   - Icinga
     - SSH into monitor0.chtc.wisc.edu
     - Become root and open up /etc/icinga2/conf.d/hosts.conf
     - Delete the entry for the server you are decommissioning.
     - Restart Icinga - Run `$ sudo service icinga2 restart`
   - Nagios
     - SSH into monitor.chtc.wisc.edu
     - Become root and open up /etc/nagios/objects/hosts.cfg
     - Delete the entry for the server you are decommissioning.
     - Restart Nagios - Run `$ sudo service nagios restart`
6. Remove it's htcondor configuration host/*.local file (Ask Moate to do this).

5 Creating Tickets

If you don't have any tickets assigned to you in your RT Queue that you can work on, then you can go ahead and create your own tickets for work that needs to be done.

We'll use a simple example to outline how to create tickets.

Let's say your queue is empty - you can log into Icinga (Or another monitoring tool that we use) to look for any nodes that might be offline or experiencing other issues.

1. Head to monitor0.chtc.wisc.edu in your browser.
2. Click on the link to Icinga, and log in (ask Admin for login info). This will take you to the Icinga dashboard. From here, you can get a basic overview of server status.
3. Look for machines listed under the "Host Problems" section.
    - Let's say we see a node, e1000.chtc.wisc.edu, listed under that section with the status message "Critical - Host Unreachable".
    - Click on that host to bring up information about it. From here, you can get a better overview of what's wrong with the machine. In this case, the machine is "Down" - offline for whatever reason. It might have crashed, it might be powered off, we don't know yet. All we know is that Icinga can't connect to it.
4. Great, now that we see a server is down, we need to check RT to see if a ticket already exists for this issue.
    - Log in to RT (crt.cs.wisc.edu) in a new tab.
    - Start a new search - select "New Search" under the "Tickets" tab at the top of the screen.
    - Enter the server name (e1000) in the "Subject" box, and make sure the relational selector is set to "Matches".
    - Click on "Add these terms and search".
5. At this point, you will either see (a) ticket(s) matching the server, or none at all. We want only want to have 1 ticket for managing each server, so it is important to not open new tickets when one already exists.
6. Let's say that you see several tickets that come up. First, check the ticket status.
    - If all tickets are "Resolved", then you are fine to re-open one. You will always re-open the most recent, or most-related ticket.
    - If there is an OPEN ticket related to the issue, then someone is already working on this issue and you can not continue creating a ticket.
    - If there are NO TICKETS matching, then you can create a new ticket.
7. If you want to re-open a closed ticket, first you will have to assign the ticket to yourself. Under the "People" Tab, choose your name from the "Owner" drop-down, and hit "Save Changes".
8. Then to re-open the ticket, hit "Re-Open" under the "Actions" tab.
    - You may now work on the ticket like you would any other.
9. If there are no tickets matching, you will need to create a new ticket.
    - At the top-right of the page next to the search bar is a ticket creation button. From the drop-down, select "htcondor-inf" - This will open the page to create a new ticket in the htcondor-inf queue.
    - In the right-most box labeled "Basics", put yourself in the "Owner" field and "CHTC" in the inf_subgroup field.

- In the left-most box, enter an appropriate subject (”e1000 down”), and a short-description if desired.
- Click ”Create” on the bottom of the page.
- Assuming you made yourself the ticket owner, you will now see this ticket in your queue!

10. LASTLY, if you re-opened or created a new ticket, you need to acknowledge that you did so on Icinga so that other poeple won’t create a duplicate ticket.

    - Return to Icinga, and on the dashboard for that particular node, click on ”Acknowledge” under the ”Problem Handling” section.
    - enter a short message, (eg ”Created a ticket - [your name]”).
    - Hit ”Acknowledge Problem” - And you’re done!

Congratulations - you now know how to find problems on Icinga and create a ticket in order to fix it!

6 Checking for Errors

6.1 Monitoring with Icinga

6.1.1 Icinga Basics

6.1.2 Web Interface

6.1.3 Icinga CLI

6.2 Error Logs

6.3 Hardware Errors

7 Advanced Troubleshooting

TODO

8 Advanced Topics

8.1 Networking

8.1.1 Networking Basics

8.1.2 CHTC Network Structure

8.1.3 Troubleshooting Networking Issues

8.2 Making Your Life Easier

1. Rebooting stubborn nodes remotely

    - There's a handy "magic key" called sysrq that is managed through the /proc filesystem in Linux. Quite a few things can be done with this, but a faithful adminstrative trick is to echo a 'key' to the sysrq-trigger to force a reboot when performing a `$ sudo reboot` just doesn't cut it.
    - First, you'll need to su into root: `$ sudo su -`
    - Next, `$ echo "1" > /proc/sys/kernel/sysrq` to enable the sysrq 'key'.
    - Lastly, in order to force the node to reboot, `$ echo "b" > /proc/sysrq-trigger`
    - Not as good as full IPMI control but useful for pesky nodes. We'll go over IPMI next.

A Node Locations

A.1 B240

Rack	A2	A4	C1	C2	C4	E2	E5	E6
Node		mussel	e036	c031	atlas13	e061	atlas80	atlas64
		atlas18	e027	c032	atlas14	e060	atlas81	atlas65
		atlas19	e039	c033	atlas28	e059	atlas82	atlas66
		atlas20	atlas17	c035	atlas15	e058	atlas83	atlas67
		e047	atlas11	c036	starfish	e002	atlas84	atlas68
		e035	atlas12	c037	e013	e025	atlas85	atlas69
		e034	atlas16	c038	e012	e030	atlas86	atlas70
		[???]		c039	e011	e057	atlas87	atlas71
		e006		c040	spalding11	e022	atlas88	atlas72
		cobbler		c041	spalding10	e040	atlas89	atlas73
		e010		c020	spalding09	e023	atlas90	atlas74
		e045(missing)		c021	spalding08	e018	atlas91	atlas75
		e044		c022	spalding07	e016	atlas92	atlas76
		[???]		c023	spalding06	e026	atlas93	atlas77
		e003		c025	spalding05	e028	atlas95	atlas78
				c027	c081	e041	atlas96	atlas79
				c028	c082	e038	atlas97	atlas08
				c029	c084	e042	atlas98	atlas42
				c030	c085	e046	atlas99	atlas62
	e049			c046	c086	e056		
	e050			c047	c088	e052		
	e051			c058	c089	e043		
	e054			c059	c090	db1*		
	e055			c060	c077	e460		
	e062			c061	c078	e461		
	e063			c062		e462		
	e064			c064		e463		
	e065			c065		e464		
	e031			c069		e465		
				c002				
				c004				
				c007				
				c008				
				c012				
				c015				
				c016				
				c017				
				c070				
				c071				
				c074				

Rack	G1	G2	G5	G6
Node	satellite	e024	[???]	atlas44
	e029	e053	atlas29	atlas45
	stress1	e009	atlas30	atlas46
	stress2	e008	atlas31	atlas47
	stress3	e007	atlas32	atlas48
	stress4	e005	atlas33	atlas49
	host-5	e089	atlas34	atlas50
		e090	atlas35	atlas51
		e088*	atlas36	atlas52
		e087	atlas37	atlas53
		e086	atlas38	atlas54
		e085	atlas39	atlas55
		e084	atlas40	atlas56
		e083	atlas41	atlas57
		e082	atlas43	atlas58
		e081	atlas03	atlas59
		e080	atlas04	atlas60
		e079	atlas05	atlas61
		e078	atlas06	atlas63
		e077		
		e076		
		e075		
		e074		
		e073		
		e072		
		e071		
		e070		
		e069		
		e068		
		e067		

A.2 2360/CSL

Rack	GR 6	GR 0	GR 4	GR 3	10	GR 7
Node	atlas07	e299	e339	e379	e413	e418
	atlas02	e298	e338	e378	e412	e417
	mem3	e297	e337	e377	e411	e416
	mem2	e296	e336	e376	e410	e415
	mem1	e295	e335	e375	e409	e414
	gpu4	e294	e334	e374	e408	e454
	host-22	e293	e333	e373	e407	e453
	host-20	e292	e332	e372	e406	e452
	host-18	e291	e331	e371	e405	e451
	host-16	e290	e330	e370	e404	e450
	host-14	e289	e329	e369	e403	e449
	host-12	e288	e328	e368	e402	e448
	host-10	e287	e327	e367	e401	e447
		e286	e326	e366	e400	e446
		e285	e325	e365	e399	e445
		e284	e324	e364	e398	e444
		e283	e323	e363	e397	e443
		e282	e322	e362	e396	e442
		e281	e321	e361	e395	e441
		e280	e320	e360	e394	e440
		e279	e319	e359	e393	e439
		e278	e318	e358	e392	e438
		e277	e317	e357	e391	e437
		e276	e316	e356	e390	e436
		e275	e315	e355	e389	e435
		e274	e314	e354	e388	e434
		e273	e313	e353	e387	e433
		e272	e312	e352	e386	e432
		e271	e311	e351	e385	e431
		e270	e310	e350	e384	e430
		e269	e309	e349	e383	e429
		e268	e308	e348	e382	e428
		e267	e307	e347	e381	e427
		e266	e306	e346	e380	e426
		e265	e305	e345		e425
		e264	e304	e344		e424
		e263	e303	e343		e423
		e262	e302	e342		e422
		e261	e301	e341		e421
		e260	e300	e340		e420
						e419

A.3 3370

Rack	1	4	5	6	7	8	9	GR 5
Node	e198	swamp-nas	osghost	host-6	e222	submit-4	e116	spalding-4
	e197	swamp05	itb-data1	pagesubmit	e220	host-23	e117	spalding-1
	e196	swamp04	itb-data2	atlas21	e219	host-21	nmi-0067	spalding-2
	e195	swamp03	itb-data3	atlas22	e215	host-19	e115	spalding-3
	e194	swamp02	itb-data4	atlas27	e214	host-17	gpu-1	
	e193	swamp01	itb-data5	atlas26	e119	host-15	e114	
	e192		itb-data6	atlas25	e212	host-13	atlas10	
	e191		itb-host1	atlas24	e211	host-11	atlas09	
	e190		itb-host2	atlas23	e210	e238	e111	
	e189		itb-host3		e209	e237	matlab-build-5	
	e188		vdt-bastion		e208	e236	e113	
	e187		vdt-centos5-test		e207	e235	e112	
	e186		vdt-debian6-test		e206	e234	e246	
	e185		vdt-debian7-test		e205	e233	e245	
	e184				e204	e232	e244	
	e183				e203	e231	e243	
	e182				e202	e230	e242	
	e181				e201	e229	e241	
	e180				e200	e228	e240	
	e179				e199	e227	e239	
	e178				host-24	e226	e259	
	e177					e225	e258	
	e176					e224	e256	
	e175					e223	e255	
	e174						e254	
	e173						e253	
	e172						e252	
	e171						e251	
	e170						e250	
	e169						e249	
	e168						e248	
	e167						e247	
	e166*							
	e165							
	e164*							
	e163							
	e162							
	e161							
	e160							
	e159							
	e158							
	e157							
	e156							
	e155							
	e154							
	e153							
	e152							
	e151							

A.4 WID Data Center

Rack	A14	A12	B1	B5
Node		deepdivesubmit	wid-003	e110
	e095	host-3	[???]	e109
	e457	[???]	e019(swamp)	e108
	e121	[???]	e017	e107
	e122	[???]	e020	e106
	e123	host-7	e033	e105
	e124	submit-5	e021	e104
	e125	submit-3	e014	e103
	e126	quickstep	e015	e102
	e127	wid-service-1	matlab-build-1	e101
	e128	e092(missing)	e001	e100
	e129	osg-ss-se	e091(swamp)	e099
	e130	[batlab]	[???]	e098
	e131	host-1	[???]	e097
	e132	hypervisor0	[???]	e096
	e133		e458	
	e134		e459	
	e135		e466	
	e136		e467	
	e137		e468	
	e138			
	e139			
	e140			
	e141			
	e142			
	e143			
	e144			
	e145			
	e146			
	e147			
	e148			
	e149			
	e150*			

Rack	Z1-R11	Z1-R10	Z1-R8	Z1-R7	Z1-R5	Z2-R11	Z2-R10	Z2-R8	Z2-R7	Z2-R5
Node	aci-336	aci-288	aci-storage-7	aci-service-1	aci-144	aci-096	aci-048	aci-192	aci-240	aci-344
	aci-335	aci-287	aci-storage-8	aci-service-2	aci-143	aci-095	aci-047	aci-191	aci-239	aci-343
	aci-334	aci-286	aci-storage-9	aci-service-3	aci-142	aci-094	aci-046	aci-190	aci-238	aci-342
	aci-333	aci-285	aci-storage-10	aci-storage-1	aci-141	aci-093	aci-045	aci-189	aci-237	aci-341
	aci-332	aci-284	aci-storage-11	aci-storage-2	aci-140	aci-092	aci-044	aci-188	aci-236	aci-340
	aci-331	aci-283	aci-storage-12	aci-storage-3	aci-139	aci-091	aci-043	aci-187	aci-235	aci-339
	aci-330	aci-282	aci-storage-13	aci-storage-4	aci-138	aci-090	aci-042	aci-186	aci-234	aci-338
	aci-329	aci-281	aci-storage-14	aci-storage-5	aci-137	aci-089	aci-041	aci-185	aci-233	aci-337
	aci-328	aci-280	aci-storage-15	aci-storage-6	aci-136	aci-088	aci-040	aci-184	aci-232	
	aci-327	aci-279	aci-storage-16		aci-135	aci-087	aci-039	aci-183	aci-231	
	aci-326	aci-278			aci-134	aci-086	aci-038	aci-182	aci-230	
	aci-325	aci-277			aci-133	aci-085	aci-037	aci-181	aci-229	
	aci-324	aci-276			aci-132	aci-084	aci-036	aci-180	aci-228	
	aci-323	aci-275			aci-131	aci-083	aci-035	aci-179	aci-227	
	aci-322	aci-274			aci-130	aci-082	aci-034	aci-178	aci-226	
	aci-321	aci-273			aci-129	aci-081	aci-033	aci-177	aci-225	
	aci-320	aci-272			aci-128	aci-080	aci-032	aci-176	aci-224	
	aci-319	aci-271			aci-127	aci-079	aci-031	aci-175	aci-223	
	aci-318	aci-270			aci-126	aci-078	aci-030	aci-174	aci-222	
	aci-317	aci-269			aci-125	aci-077	aci-029	aci-173	aci-221	
	aci-316	aci-268			aci-124	aci-076	aci-028	aci-172	aci-220	
	aci-315	aci-267			aci-123	aci-075	aci-027	aci-171	aci-219	
	aci-314	aci-266			aci-122	aci-074	aci-026	aci-170	aci-218	
	aci-313	aci-265			aci-121	aci-073	aci-025	aci-169	aci-217	
	aci-312	aci-264			aci-120	aci-072	aci-024	aci-168	aci-216	
	aci-311	aci-263			aci-119	aci-071	aci-023	aci-167	aci-215	
	aci-310	aci-262			aci-118	aci-070	aci-022	aci-166	aci-214	
	aci-309	aci-261			aci-117	aci-069	aci-021	aci-165	aci-213	
	aci-308	aci-260			aci-116	aci-068	aci-020	aci-164	aci-212	
	aci-307	aci-259			aci-115	aci-067	aci-019	aci-163	aci-211	
	aci-306	aci-258			aci-114	aci-066	aci-018	aci-162	aci-210	
	aci-305	aci-257			aci-113	aci-065	aci-017	aci-161	aci-209	
	aci-304	aci-256			aci-112	aci-064	aci-016	aci-160	aci-208	
	aci-303	aci-255			aci-111	aci-063	aci-015	aci-159	aci-207	
	aci-302	aci-254			aci-110	aci-062	aci-014	aci-158	aci-206	
	aci-301	aci-253			aci-109	aci-061	aci-013	aci-157	aci-205	
	aci-300	aci-252			aci-108	aci-060	aci-012	aci-156	aci-204	
	aci-299	aci-251			aci-107	aci-059	aci-011	aci-155	aci-203	
	aci-298	aci-250			aci-106	aci-058	aci-010	aci-154	aci-202	
	aci-297	aci-249			aci-105	aci-057	aci-009	aci-153	aci-201	
	aci-296	aci-248			aci-104	aci-056	aci-008	aci-152	aci-200	
	aci-295	aci-247			aci-103	aci-055	aci-007	aci-151	aci-199	
	aci-294	aci-246			aci-102	aci-054	aci-006	aci-150	aci-198	
	aci-293	aci-245			aci-101	aci-053	aci-005	aci-149	aci-197	
	aci-292	aci-244			aci-100	aci-052	aci-004	aci-148	aci-196	
	aci-291	aci-243			aci-099	aci-051	aci-003	aci-147	aci-195	
	aci-290	aci-242			aci-098	aci-050	aci-002	aci-146	aci-194	
	aci-289	aci-241			aci-097	aci-049	aci-001	aci-145	aci-193	

B List of Resources

TODO - A useful list of external resources.

- http://pages.cs.wisc.edu/ van-lyse/ - Neil's CS Page contains tons of useful CHTC links.
- monitor0.chtc.wisc.edu - Our webpage for server monitoring. Contains links to Icinga, Grafana, Ganglia, and more.
- https://chtc-sa.cs.wisc.edu/trac/priv - CHTC Private Wiki. Login required, contains a massive amount of information and guides. Some things are dated, but still extremely useful.

C Glossary

TODO - Glossary of terms commonly used in this manual and in daily work.

- Cobbler - Cobbler is a frontend tool for configuring Linux network installs with PXE boot. Learn more: http://cobbler.github.io/
- Icinga - An open-source system monitoring tool with a web interface. Our Icinga server is at monitor0.chtc.wisc.edu. Learn more at: https://www.icinga.org/
- Puppet - Puppet is a tool for managing automatic configuration of systems. Our puppetmaster machine is wid-service-1. Learn more: https://puppet.com/
- RT (Request Tracker) - Ticket-tracking software. Access RT at: https://crt.cs.wisc.edu/rt/