

Configuring and Managing Multiple Shasta Systems Best Practices Developed During the Perlmutter Deployment

2022/05/04

Ershaad Basheer - NERSC James Botts - NERSC David Fox - NERSC Aditi Gaur - NERSC Doug Jacobsen - NERSC Harold Longley - HPE Eric Roman - NERSC Chris Samuel - NERSC John Stile - NERSC

HPE Cray-EX Systems at NERSC

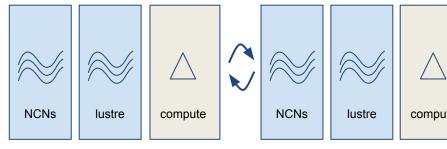
To meet demands of continuous operations:

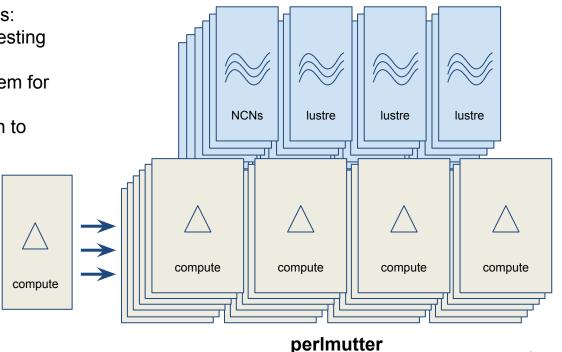
- Use pre-production test system for testing operational procedures
- Need longer-term development system for wider range experimentation

muller

testing for production

Need automation and API integration to • maintain consistency





production

BERKELEY LAB

U.S. DEPARTMENT OF

Office of

Science

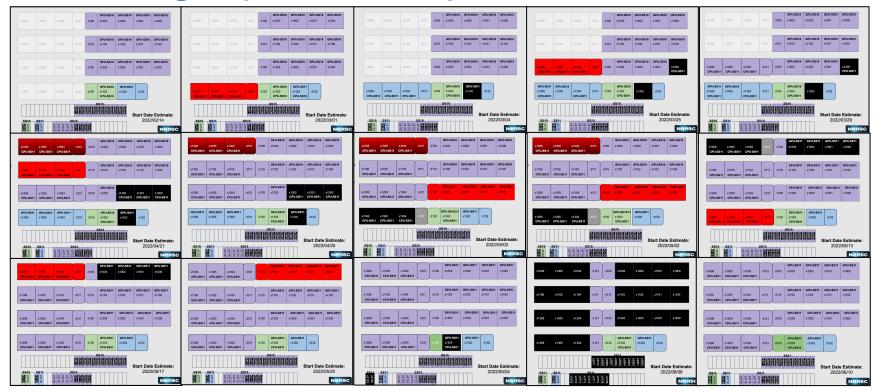


alvarez

experimentation



Don't forget perlmutter phase 2





BERKELEY LAB

Office of

Key Challenges for Deployment

- Hardware integration
 - SHCD (Shasta Hardware Configuration Document)
 - Validating physical cabling
 - Node testing
- Configuration Management (multiple systems!)
 - System configuration (gitea/CFS/IMS Recipes/BOS/BSS)
 - Secrets
 - HSN configuration
 - RPMs
 - Containers/Helm charts/Loftsman manifests for site-provided services
 - HPE-provided software and System Upgrades
- System Stabilization
 - DVS-Over-HSN
 - DVS/CPS Worker Node Separation







Office o

Hardware Integration: SHCD

- Must ensure that any all hardware changes are first in the SHCD
- Be sure the hardware configuration will support future modes of operation.
 - NERSC wants hsn0 interfaces on one set of switches and hsn1 on another in river racks to support system resiliency
- If the SHCD is wrong, nothing else on the system can be right

001	<u>ት 🖬 ዓ</u> ታ	ب گ					🖻 NE	RSC9 Perl	Imutter Re	v B7											
lome Ins		Page Lavou	it Formulas	a Data	Review Vie	w															2
		i ago Layor		Bata	1001007/ 110														-		_
a 🕺 💥 Cut	Centur	v Got + 10	- A- A-			📑 🔁 Wraj	n Text 🔻	Gene	ral				.	~ ~	4.00		_ I↔I		∑ Autos	Sum 🔹 🍟	۱
Co	v •											≠	🥖 İ I	💉 i 🛛					📀 Fill 🔻		ZT
ste	É R	ru.	🔹 🔦 🔹 🗛 -			👄 Mere	ge & Center		%)	00. 0.	Cond	itional I	Format	Cell	Insert	Delete	Form				Sort &
💞 For	mat -			- 10-0							Form	atting a	s Table	Styles					🥟 Clear		Filter
	$< \sqrt{f_x}$	4120																			
•									_	_											
A	В	C	D	E		G	<u> </u>	L	к	L	м	N	0	P	Q	R	S	T	U	v	w
	100g-15m-MPO		PO/MPO OM4 12115		Q1H66A		2							sw-ruguz	sw-10gu.	2					sw-10g
	100g-30m-MPO 100g-50m-MPO		PO/MPO OM4 12f 30 PO/MPO OM4 12f 50		Q1H67A 0K731A	4	-														
	1008-20m-Who	Inc remier Flex M	PO/MPO 0M4 121 50	m coi	QR/31A Totale	746	4														
		Label Inf	o (2 lines)		lotal	.40															
	Source	Source-L	Destination-L	Destination	Description 💌	Part Number		Source-N	Rack	v Location	Slot 💌		Port	Destination N	Rack	Location		Port		Changed T	Destinatio
phase 2	sw-100g02	x3112u46-j28	x3002u42-p53	25g-sw00	100g-30m-MPO	Q1H67A	E1000 uplink	sw-100g02	x3112	U46	1		28	25g-sw00	×3002	U42		p53			
phase 2	sw-100g02	x3112u46-j29	x3002u43-p53	25g-sw01	100g-30m-MPO	Q1H67A	E1000 uplink	sw-100g02	x3112	U46	-1		29	25g-sw01	x3002	U43	-	p53			
phase 2	sw-100g01	x3112u45-j28	x3002u42-p54	25g-sw00	100g-30m-MPO	Q1H67A	E1000 uplink	sw-100g01	x3112	U45	-1		28	25g-sw00	×3002	U42	-	p54	6		
phase 2	sw-100g01	x3112u45-j29	x3002u43-p54	25g-sw01	100g-30m-MPO	Q1H67A	E1000 uplink	sw-100g01	x3112	U45			29	25g-sw01	×3002	U43	-	p54	-	-	
CDS	sw-10g01	x3115u39-j51	x3115u40-j51	sw-10g02	100g-1m-DAC	R0Z25A	VSX	sw-10g01	x3115	u39	-		j51	sw-10g02	x3115			51	-		
CDS	sw-10g01 sw-10g01	x3115u39-j52 x3115u39-j48	x3115u40-j52 x3115u40-j48	sw-10g02 sw-10g02	100g-1m-DAC 6ft	R0Z25A	VSX	sw-10g01	x3115 x3115	U39	-		j52 i48	sw-10g02	x3115 x3115	U40 U40	-	52	-		
CDS	sw-10g01	x3115u39-j48 x3115u39-j49	x3115u40-j48 x3112u45-j25	sw-10g02 sw-100g01	611 100g-5m-DAC	Q1Q95A R0Z26A	keep alive cds uplink	sw-10g01 sw-10g01	x3115 x3115	U39 U39			j48 j49	sw-10g02 sw-100g01		U40 U45	1	48	CDS		
CDS	sw-10g01	x3115u39-j49	x3112u45-j25	sw-100g01 sw-100g02	100g-5m-DAC	R0Z26A	cds uplink	sw-10g01	x3115 x3115	U39 U39			149	sw-100g01 sw-100g02	x3112 x3112	U45 U46	1	25			
CDS	sw-10g02	x3115u40-j49	x3112u45-j26	sw-100g01	100g-5m-DAC	ROZZGA	cds uplink	sw-10g01 sw-10g02	x3115	U37			i49	sw-100g02	x3112	046	-1	26	1		
CDS	sw-10g02	x3115u40-j50	x3112u46-j26	sw-100g02	100g-5m-DAC	R0Z26A	cds uplink	sw-10g02	x3115	U40			i50	sw-100g02	x3112	U46	-1	26	1		
	sw-100g01	x3112u45-j30	x3112u46-j30	sw-100g02	100g-1m-DAC	R0Z25A	keep alive	sw-100g01	x3112	U45	-1		30	sw-100g02	x3112	U46	-	30		1	sw-100
161	sw-100g01	x3112u45-j31	x3112u46-j31	sw-100g02	100g-1m-DAC	R0Z25A	VSX	sw-100g01	x3112	U45	-		31	sw-100g02	×3112	U46	-	31	5		sw-100
162	sw-100g01	x3112u45-j32	x3112u46-j32	sw-100g02	100g-1m-DAC	R0Z25A	VSX	sw-100g01	x3112	U45	1		32	sw-100g02	x3112	U46	-1	32	T Š		sw-100
	sw-100g01	x3112u45-j23		NERSC CAN Conne	ction		NERSC CAN	sw-100g01	x3112	U45	1		23				<u> </u>		ild (
	sw-100g01	x3112u45-j24		NERSC CAN Conne	ction		NERSC CAN	sw-100g01	x3112	U45			24						8		
	sw-100g02	x3112u46-j23		NERSC CAN Conne			NERSC CAN	sw-100g02	x3112	U46	-1		23						-		
	sw-100g02	x3112u46-j24		NERSC CAN Conne			NERSC CAN	sw-100g02	x3112	U46	-1		24								
141	cdu0sw1	d100u01-j49	x3112u45-j01	sw-100g01	40g-30m-LC-LC	QK736A	cdu uplink	cdu0sw1	d100	u01	-		j49	sw-100g01			-1	01	0	-	sw-100
142	cdu0sw1	d100u01-j50	x3112u46-j01	sw-100g02	40g-30m-LC-LC	QK736A	cdu uplink	c du0sw1	d100	u01	-	-	j50	sw-100g02	x3112		+	01	-		sw-100
143	cdu0sw2 cdu0sw2	d100u02-j49 d100u02-j50	x3112u45-j02 x3112u46-j02	sw-100g01 sw-100g02	40g-30m-LC-LC 40g-30m-LC-LC	QK736A QK736A	cdu uplink	cdu0sw2	d100	U02	-		j49 i50	sw-100g01	x3112 x3112	U45 U46	-	02	-	-	sw-100
144	cdu0sw2	d110u01-i49	x3112u46-j02 x3112u45-j03	sw-100g02 sw-100g01	40g-30m-LC-LC 40g-30m-LC-LC	QK736A	cdu uplink cdu uplink	cdu0sw2 cdu1sw1	d100 d110	U02	-	-	150	sw-100g02 sw-100g01	x3112 x3112	U46 U45	-	02	+	1	sw-100 sw-100
145	cdu1sw1	d110u01-j50	x3112u45-j03	sw-100g01	40g-30m-LC-LC	QK736A	cdu uplink	cdulswl	d110	U01			150	sw-100g01	x3112	U45	1	03	+	1	sw-100
147	cdu1sw2	d110u02-j49	x3112u45-j04	sw-100g01	40g-30m-LC-LC	QK736A	cdu uplink	cdu1sw2	d110	U02			149	sw-100g02	x3112	U45	- 4	04			sw-100
148	cdu1sw2	d110u02-j50	x3112u46-j04	sw-100g02	40g-30m-LC-LC	QK736A	cdu uplink	cdu1sw2	d110	U02			j50	sw-100g02	x3112	U46	-	04			sw-100
149	cdu2sw1	d120u01-j49	x3112u45-j05	sw-100g01	40g-30m-LC-LC	QK736A	cdu uplink	cdu2sw1	d120	U01	-		j49	sw-100g01	×3112	U45	-j.	05			sw-100
150	cdu2sw1	d120u01-j50	x3112u46-j05	sw-100g02	40g-30m-LC-LC	QK736A	cdu uplink	cdu2sw1	d120	U01	-		j50	sw-100g02	x3112	U46	-1	05			sw-100
151	cdu2sw2	d120u02-j49	x3112u45-j06	sw-100g01	40g-30m-LC-LC	QK736A	cdu uplink	cdu2sv/2	d120	U02	-		j49	sw-100g01	x3112	U45	÷	06			sw-100
152	cdu2sw2	d120u02-j50	x3112u46-j06	sw-100g02	40g-30m-LC-LC	QK736A	cdu uplink	cdu2sw2	d120	U02	-		j50	sw-100g02	x3112	U46	÷	06		-	sw-100
phase 2	cdu3sw1	d101u01-j49	x3112u45-j15	sw-100g01	40g-30m-LC-LC	QK736A	cdu uplink	c du3sw1	d101	u01	-		j49	sw-100g01	x3112	U45	-1	15			sw-100g
phase 2	cdu3sw1	d101u01-j50	x3112u46-j15	sw-100g02	40g-30m-LC-LC	QK736A	cdu uplink	cdu3sw1	d101	U01			150	sw-100g02	x3112	U46	-	15	1	1	sw-100c

Best Practice: Get to know your SHCD, find ways to track and validate changes with your hardware support Missing functionality: Ability to validate SHCD against documented CSM requirements







Office of

Hardware Integration: Validate Physical Cabling

Confirm that wiring in the system matches the SHCD

 For HSN, cable_inventory and LLDP plus analysis scripts (NERSC custom script shown) can help confirm things are where they belong



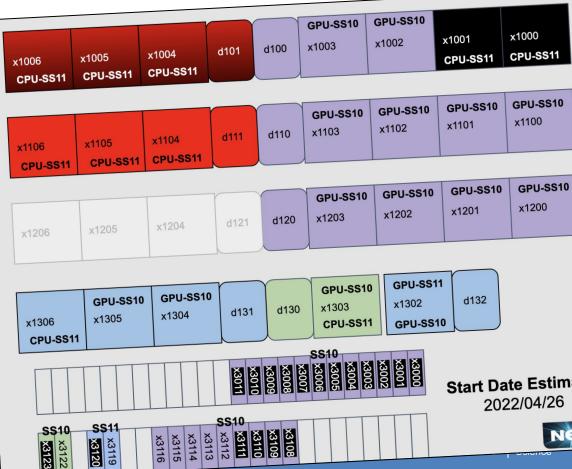
```
=== ANALYSIS ===
Incorrect island:
                   ['x1003c3r7j15', 'x3115c0r30j17', 'x1003c3r7j15', 'x3115c0r31j17']
Incorrect island:
                   ['x1201c1r1j15', 'x3000c0r44j19', 'x1201c1r1j15', 'x3000c0r45j19']
Incorrect island:
                   {'x3004c0r46j15', 'x3114c0r31j16', 'x3114c0r31j15', 'x3114c0r30j15',
'x3004c0r45j15'}
Incorrect island:
                   {'x3008c0r47j30', 'x3009c0r44j25', 'x3009c0r44j30', 'x3008c0r44j30'}
                   {'x3010c0r45j30', 'x3010c0r46j30', 'x3009c0r47j2', 'x3009c0r47j1'}
Incorrect island:
Incorrect island:
                   {'x3108c0r46j28', 'x3108c0r45j28', 'x3109c0r45j3', 'x3109c0r45j2'}
Incorrect island:
                   {'x3114c0r30j4', 'x3114c0r30j3', 'x3113c0r30j25', 'x3113c0r31j25'}
                  {'x3114c0r31j3', 'x3113c0r31j28', 'x3114c0r31j4', 'x3113c0r30j28'}
Incorrect island:
===========
Group of Incorrect connections: ['x1003c3r7j15', 'x3115c0r30j17', 'x1003c3r7j15', 'x3115c0
Incorrect link ('x1003c3r7j15', 'x3115c0r30j17') related to missing links:
     ('x1003c3r7j15', 'x3115c0r31j17')
   Check if x3115c0r30j17 should move to x3115c0r31j17
______
```

Check if x3000c0r44j19 should move to x3000c0r45j19

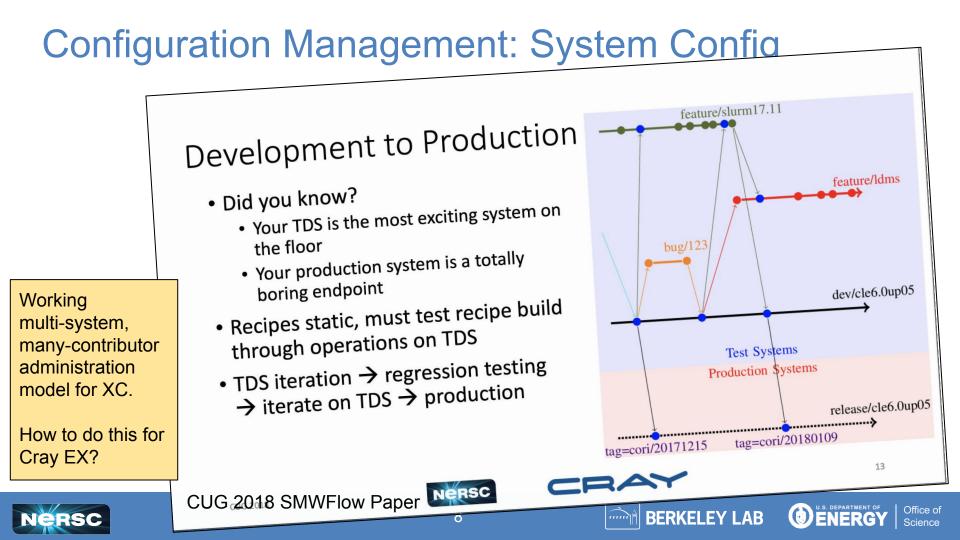
Hardware Integration: Node Testing

Stabilizing both the computes nodes and the network simultaneously is exceedingly difficult.

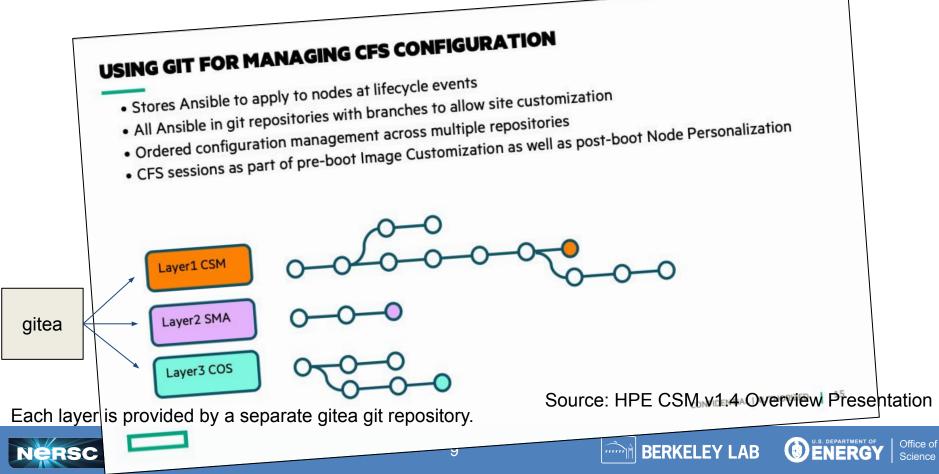
For the phase 2 system, NERSC is leveraging an HPCM test system to validate each row *before* it gets added to perlmutter.

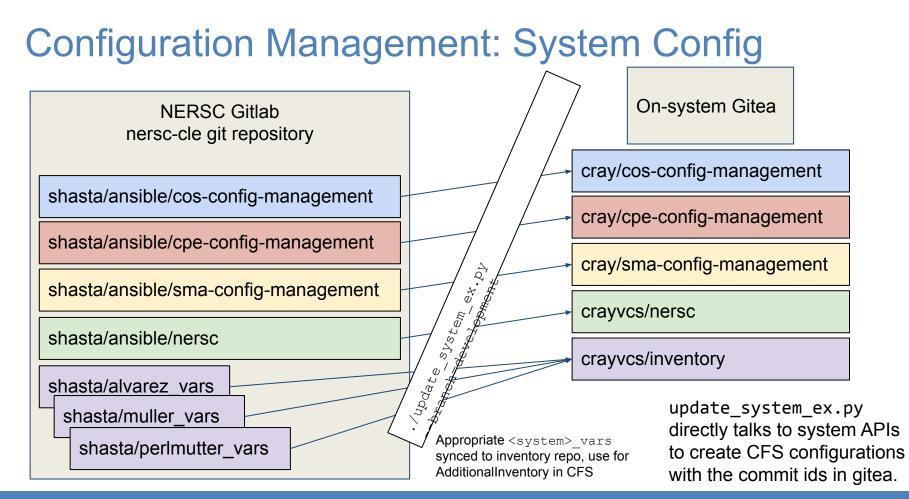






Configuration Management: System Config



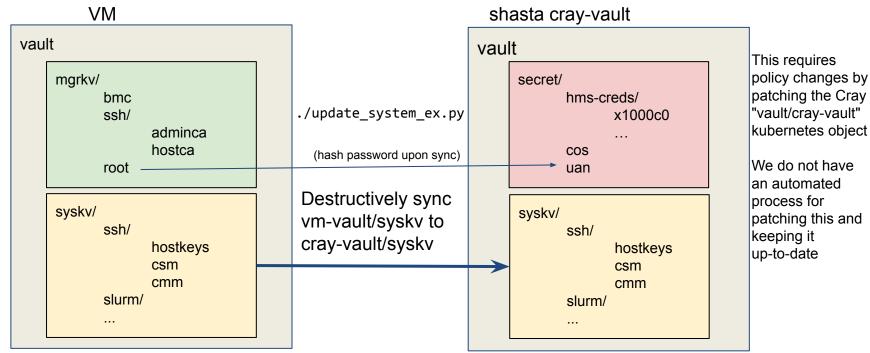






Office of

Configuration Management - Secrets



mgrkv has secrets that are only known on the manager VM, such as plaintext passwords, private keys for the host and admin CAs

syskv is sync'ed from the VM to cray-vault (using kubectl port forwarding) for use with CFS.

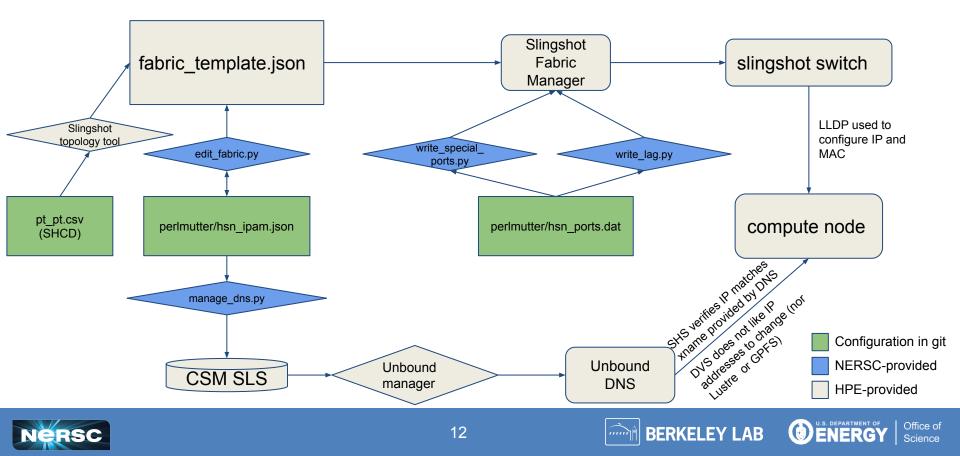
BERKELEY LAB

cray-vault also gets hashed passwords for deployment on the system

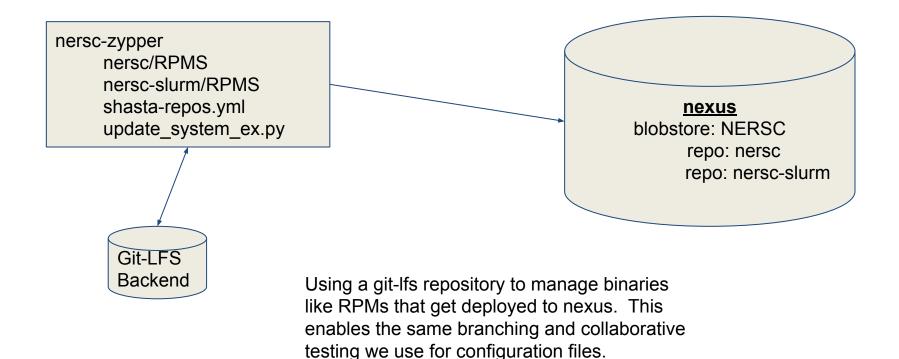
Office of



Configuration Management: HSN



Configuration Management: RPMs





BERKELEY LAB

Office of

Configuration Management

Helm Charts

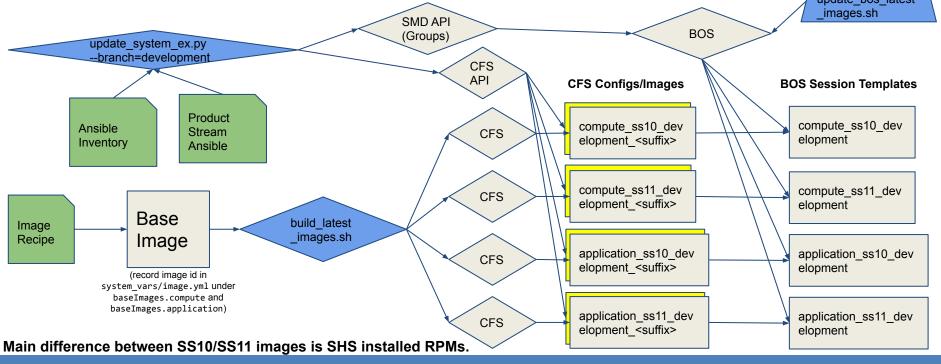
- NERSC-custom charts are stored in `nersc-cle` git repository
- Deployed with Cray's `loftsman` from manager VM (leveraging end-user's privileges with kubernetes)
- Containers
 - Based on parse of nersc-cle charts, use skopeo to sync containers from external source, registry.nersc.gov, or VM-constructed container to on-system nexus
- customizations.yaml
 - Site/system overrides to helm charts. Presently not well managed because we need a process to generate sealed secrets from git sources (don't want to keep secrets, even encrypted in git)

BERKELEY LAB



Managing SS10/SS11 Hybrid Systems

• This is explicitly *not* supported by HPE at this time. But is a necessary requirement for the integration of perlmutter phase 2







U.S. DEPARTMENT OF

Office of

Configuration Management: Workflow

- \$ cd nersc-zypper
- \$ git checkout development
 - ./update_system_ex.py
 - # uploads RPMs to nexus

- \$ cd nersc-cle
- \$ git checkout development
- \$./update_system_ex.py --branch=development
 - # record timestamp as `suffix
 - # sync HSM groups for SS10/SS11 differentiation`
 - # syncs secrets
 - # writes feature/a ansible directories to dmjtest branches in gitea
 - # generates CFS configuration objects, uploads using CFS API
 - # compute-development-<suffix>
 - # login-development-<suffix>
 - # gateway-development-<suffix>
 - # generates bos-sessiontemplates-<suffix> with unconfigured images
- \$./shasta/scripts/build_latest_images.sh development
- \$ cd bos-sessiontemplates-<suffix>
- \$../shasta/scripts/update_bos_latest_images.sh -i development -d development
 - # generates and uploads usable BOS sessiontemplates
 - # compute-development, login-development, gateway-development
- \$ cray bos session create --template-uuid compute-ss10-development --operation reboot
- \$ cray bos session create --template-uuid compute-ss11-development --operation reboot



Or use prep_boot_config.py to scalably rewrite BSS/CFS for all nodes and avoid bos completely. Enables dynamic rolling updates

BERKELEY LAB

Office of

Configuration: HPE Provided Software and Upgrades

- Installation of most CSM-compatible product streams have two phases:
 - Nexus artifact installation
 - Changes content in some common repositories non-destructively but content immediately is accessible and default for similar product lines (SLES repos vs SLES 22.01 repos)
 - Loftsman manifest deployment to configure the product
- NERSC does not have automation around the management of these products
 - the non-interactive `./install.sh` used *almost* across the board make automated data transfer/installation a clear future direction
- HPE Cray EX software recipe upgrade installation often start with CSM, then firmware (HFP), then COS and other products
 - This could result in three interruptions of each ncn-worker node
 - For major upgrades NERSC has usually found it's been fine to reorganize the update procedure to minimize the number of steps
 - The devil is in the details here, but there are many opportunities for automation and improving efficiency of the rolling update process







Office of

System Stabilization: DVS over HSN

- Switching DVS to HSN instead of NMN several benefits:
 - Nodes that have an incorrect HSN configuration fail to boot (making their problem obvious)
 - Use of HSN for DVS improves user experience because NMN bandwidth is limited
 - Increases options for backend LND (could use o2iblnd for SS10 or kkfilnd for SS11)
- In COS 2.0 and 2.1 this required patching the initrd to properly setup Inet and then dvs to add the needed Inds, and then modifying the cos/uan_config_management layers for configuration
- In COS 2.2 configuration in ansible inventory is all that is required

This was a key change during perlmutter deployment that led to major progress in the project







Office o

System Stabilization: CPS/DVS Worker Node Separation

- During perlmutter integration we had several instances of resource collisions and ncn-w* node crashes not tied directly to bugs in DVS, but contention with other critical workloads
- Perlmutter has 26 worker nodes, we chose to dedicate 8 of them to CPS/DVS by preventing management jobs to schedule on those nodes.
 - cray cps deployment update --nodes=<list of nodes>
 - kubectl drain --delete-local-data --ignore-daemonsets <node one at a time>
 - Leave them cordoned
- Estimate each worker node is capable of forwarding to 500 nodes
- NERSC's experience is that separating CPS/DVS from management pods has improved worker stability







Office o

What is the Impact?

- Focusing on *process-oriented* management that abstracts system details enables:
 - Scalable transfer of capability from small scale to test to large scale production
 - High fidelity transfer of features from test to production
 - Collaborative integration of difficult work on test systems with simple deployment to production
- Migrating HPE product stream code (ansible, recipes) into NERSC repository:
 - Enables rapid workarounds or improvements in HPE-provided code
 - Reinforces the separation of code vs data
 - Supports inclusion of HPE products in the development/deployment process







Conclusions

- Using the techniques in this presentation, NERSC was able to successfully deploy and maintain three Cray EX systems with minimal overlapped effort
- Insights gained using this work has enabled NERSC and HPE to develop a method of iteratively deploying perlmutter phase 2 with minimal disruption, despite all the hardware in the system being taken offline at some point during the process
- NERSC has demonstrated repeatedly the utility of having test systems to optimize deployment procedures and even start doing scalable test by moving production resources





