A Unified Lab Notes Framework for Experimental Reproducibility in HPC Systems

R. Marshall and P. Bangalore





Motivation

- HPC systems are increasingly diverse, with various
 - hardware configurations
 - firmware versions
 - operating systems
 - installed software versions
 - communication media ...
- By the time the results from an experiment can be published, some or all of the components of the environment could have changed.





Motivation (cont.)

- While a number of tools exist to aid in reproducibility, there is still a gap in experimental integrity that the researcher is often left to close manually:
 - input files and runtime parameters
 - output content and format
 - method of connecting dependencies with configuration management and program output







Trackable components

By breaking down an experiment into trackable components, we can ensure all aspects of a published experiment can be reproduced.

Components below this line are beyond scope to manage directly, though collection of metadata to record is OK.

System A



INPUT FILES AND METADATA	
CLI: PARAMETERS AND OUTPUT	
PROGRAM CODE	
COMPILERS AND DEPENDENCIES	
LOW LEVEL METADATA	
OPERATING SYSTEM	DRIVERS
FIRMWARE	
HARDWARE	



Experimental Integrity: Compilers and Dependencies

- Systems like Spack are useful for tracking and managing dependencies.

linux-centos8-skv	lake / gcc@8.4.1				
autoconf@2.69 autoconf-archive@201 automake@1.16.2 berkeley-db@18.1.40 cmake@3.18.4	diffutils@3.	<pre>7 isl@0.21 libiconv@1.16 libpciaccess@0.16 libsigsegv@2.12</pre>	libxml2@2.9.10 m4@1.4.18 mpc@1.1.0 mpfr@4.0.2 ncurses@6.2	numactl@2. openssl@1. perl@5.32. pkgconf@1. readline@8	1.1h xz@5.2.5 0 zlib@1.2.11 7.3 zstd@1.4.5
linux-centos8-sky	lake / gcc@10.2.0				
autoconf@2.69	gettext@0.21	libtool@2.4.6	pcre2@10.35		readline@8.0
automake@1.16.2	git@2.29.0	libunistring@0.9.10	perl@5.32.0		reframe@3.1
berkeley-db@18.1.40	hwloc@1.11.11	libuuid@1.0.3	pkgconf@1.7.3		sqlite@3.33.0
bzip2@1.0.8	libbsd@0.10.0	libxml2@2.9.10	py-attrs@20.3.	attrs@20.3.0 tar@1.32	
clamr@master	libedit@3.1-20191231	m4@1.4.18	py-jsonschema@	3.2.0	util-macros@1.19.1
cmake@3.18.4	libffi@3.3	ncurses@ <mark>6.2</mark>	py-pyrsistent@0.15.7 xz@5.2.5		xz@5.2.5
curl@7.72.0	libiconv@1.16	numactl@2.0.14	py-setuptools@50.3.2 zlib@1.2.11		zlib@1.2.11
diffutils@3.7	libidn2@2.3.0	openmpi@3.1.6	py-setuptools-	scm@4.1.2	
expat@2.2.10	libpciaccess@0.16	openssh@8.4pl	py-six@1.14.0		
gdbm@1.18.1	libsigsegv@2.12	openssl@1.1.1h	python@3.8.6		

Output of 'spack find'



Center for Understandable, Performant Exascale Communication Systems

• For full reproducibility of an experimental application that uses binary executables, the compiler and linked libraries should also be reproduced.



Experimental Integrity: User Input/Output

mlj

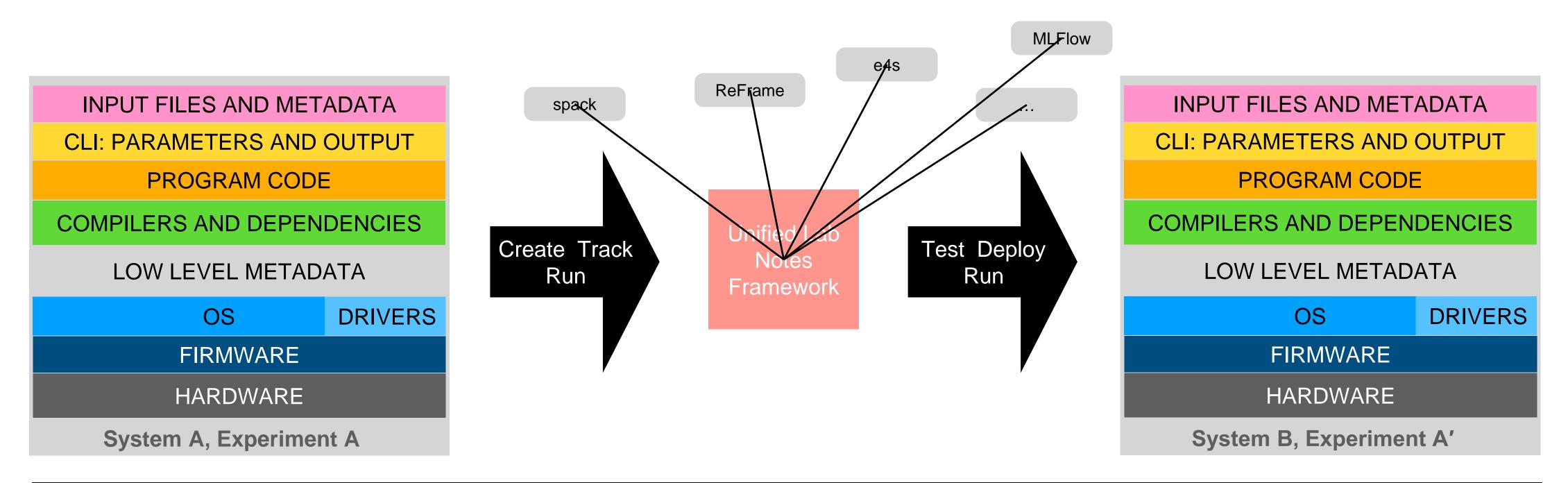
- The following should be tracked and stored:
 - user configuration files
 - program input files/metadata
 - command line parameters
 - program output
- Storing in a common format helps ensure experimental integrity
- Tools/systems of interest:
 - MLFlow
 - ReFrame



Immens	•							15 15:59	Jul						.nrome 🔻	🌀 Google
Cov Experiments Intents C I															× +	
Immens	* * 2														::5000/#/	 localhos
h Experiments At	GitHub Docs													Models	Experiments	low
it it<													t	Default	+ <	iments
Image: Image															S	h Experimen
Experiment ID: 0 Metrics Notes [] None Search Runs: metrics.rmse < 1 and params.model = "tree" and tags.mlflow.source.type = "LOCAL." © Filter Ournload CSV± Extra Time Ournload CSV± Extra Time Ournload CSV± Extra Time Ournload CSV± Extra Time Ournload CSV± Ournload CSV± Extra Time Ournload CSV± Ournload	×								re	arn mo	an experiment. <mark>Lea</mark>	ning runs	Track machine learning trair	()	<u> </u>	ult
None Search Runs: metricos.mse < 1 and parama.model = "tree" and tags.millow.source.type = "LOCAL."	lruns/0	w4kbtkxdrr6qky4cs/bir	ōmcybdasknisfw	mr-master-d55m	jcc-10.2.0/clar	os8-skylake/g	ack/linux-cent						nt ID: 0	Experime		
Search Runs:													S 🗹	 Notes 		
Showing 8 matching runs Compare Delete Download CSV± Showing 8 matching runs Compare Delete Download CSV± Image: Compare Metrics < Start Time User Source Ver mass mem_available mem_free mem_peak mem_sued mesh_timer_cal mesh_timer_fit state 0 2021-07-15 15:23:18 ryan parse_clamrout.py - 171748.5 7953820 4920320 14988 14816 1.092 0.788 7.833 0.933 0 2021-07-15 15:21:18 ryan parse_clamrout.py - 171748.5 7953820 4920412 14988 14816 1.092 0.788 7.833 0.933 0 2021-07-15 15:21:18 ryan parse_clamrout.py - 171748.5 7953820 4920412 14988 14816 1.065 0.772 7.593 0.900 0 2021-07-15 15:21:18 ryan parse_clamrout.py - 17174.5 7953820 492366 15036 14864 1.042 0.752 7.495 0.888														None		
Start Time User Source Ver mass mem_available mem_peak mem_used mesh_timer_ca mesh_timer_ha state_timer_fin state 2021-07-15 15:23:18 vyan parse_clamrout.py 10748.5 7953820 4920320 14988 14816 1.092 0.788 7.833 0.393 0.2021-07-15 15:21:18 vyan parse_clamrout.py - 171748.5 7953820 4920412 14988 14816 1.065 0.772 7.593 0.90 0	earch Clear	- Filter	0				AL"	ce.type = "LOCA	s.mlflow.sour	ind tag	ns.model = "tree" a	1 and par	uns: Q metrics.rmse <	Search R		
Start Time User Source Ver mass mem_available mem_preak mem_preak mem_negak mem_	Columns									csv	e Download C	e De	8 matching runs Compare	Showing 8		
 									Metrics <							
Image: Control of the state of the stat	te_timer_ref total_time	_ha state_timer_fin	cal mesh_timer.	mesh_timer_cal	mem_used	mem_peak	mem_free	mem_available	mass	Ver	ource	User	Start Time			
 2021-07-15 15:15:77 ryan _ parse_clamrout.py - 171748.5 7953820 4913064 17116 16944 1.178 0.852 8.714 1.06 2021-07-15 15:14:00 ryan _ parse_clamrout.py - 171748.5 7953820 4923688 15036 14864 1.042 0.752 7.495 7.495 0.88 7.49 0.88 7.49 0.88 7.49 0.749 0.749 0.749 0.749 0.74 7.49 0.749 0.74 7.49 0.88 7.49 7	39 11.02	7.833	0.788	1.092	14816	14988	4920320	7953820	171748.5	-] parse_clamrout.py	ryan	⊘ 2021-07-15 15:23:18			
Image: Construction of the parse clammout.py - 171748.5 7953820 4923868 15036 14864 1.042 0.752 7.495 0.88 Image: Construction of the parse clammout.py - 171748.5 7953820 4923692 15028 14860 1.031 0.748 7.495 0.88 Image: Construction of the parse clammout.py - 171748.5 7953820 4923692 15028 14860 1.031 0.748 7.449 0.88 Image: Construction of the parse clammout.py - 171748.5 7953820 4923528 14980 1.031 0.748 7.449 0.88 Image: Construction of the parse clammout.py - 171748.5 7953820 4923528 14988 14816 1.035 0.751 7.447 0.88 Image: Construction of the parse clammout.py - 171748.5 7953820 4922348 15132 14964 1.033 0.749 7.454 0.88 Image: Construction of the parse clammout.py - 171748.5 7953820 4922348 15132 14964 1.033 0.749 7.454 0.88 <td>01 10.68</td> <td>7.593</td> <td>0.772</td> <td>1.065</td> <td>14816</td> <td>14988</td> <td>4920412</td> <td>7953820</td> <td>171748.5</td> <td>-</td> <td>] parse_clamrout.py</td> <td>ryan</td> <td>⊘ 2021-07-15 15:21:18</td> <td></td> <td></td> <td></td>	01 10.68	7.593	0.772	1.065	14816	14988	4920412	7953820	171748.5	-] parse_clamrout.py	ryan	⊘ 2021-07-15 15:21:18			
¹ · · · · · · · · · · · · · · · · · · ·	5 12.24	8.714	0.852	1.178	16944	17116	4913064	7953820	171748.5	-] parse_clamrout.py	ryan	⊘ 2021-07-15 15:15:57			
○ 2021-07-15 15:01:09 ryan □ parse_clamrout.py - 171748.5 7953820 4923528 14988 14816 1.035 0.751 7.447 0.88 ○ 2021-07-15 14:57:44 ryan □ parse_clamrout.py - 171748.5 7953820 4922348 15132 14964 1.033 0.749 7.454 0.88	35 10.53	7.495	0.752	1.042	14864	15036	4923868	7953820	171748.5	-] parse_clamrout.py	ryan	⊘ 2021-07-15 15:14:00			
○ 2021-07-15 14:57:44 ryan □ parse_clamrout.py - 171748.5 7953820 4922348 15132 14964 1.033 0.749 7.454 0.88	31 10.45	7.449	0.748	1.031	14860	15028	4923692	7953820	171748.5	-] parse_clamrout.py	ryan	⊘ 2021-07-15 15:08:45			
	35 10.46	7.447	0.751	1.035	14816	14988	4923528	7953820	171748.5	-] parse_clamrout.py	ryan	⊘ 2021-07-15 15:01:09			
	3 10.46	7.454	0.749	1.033	14964	15132	4922348	7953820	171748.5	-] parse_clamrout.py	ryan	⊘ 2021-07-15 14:57:44			
O 2021-07-15 14:54:16 ryan □ parse_clamrout.py -	-	-	-	-	-	-	-	-	-	-] parse_clamrout.py	ryan	⊘ 2021-07-15 14:54:16			
Load more							Load more									



A Unified Lab Notes Framework



- 1. Create Experiment A with framework
- 2. Run Experiment A on System A



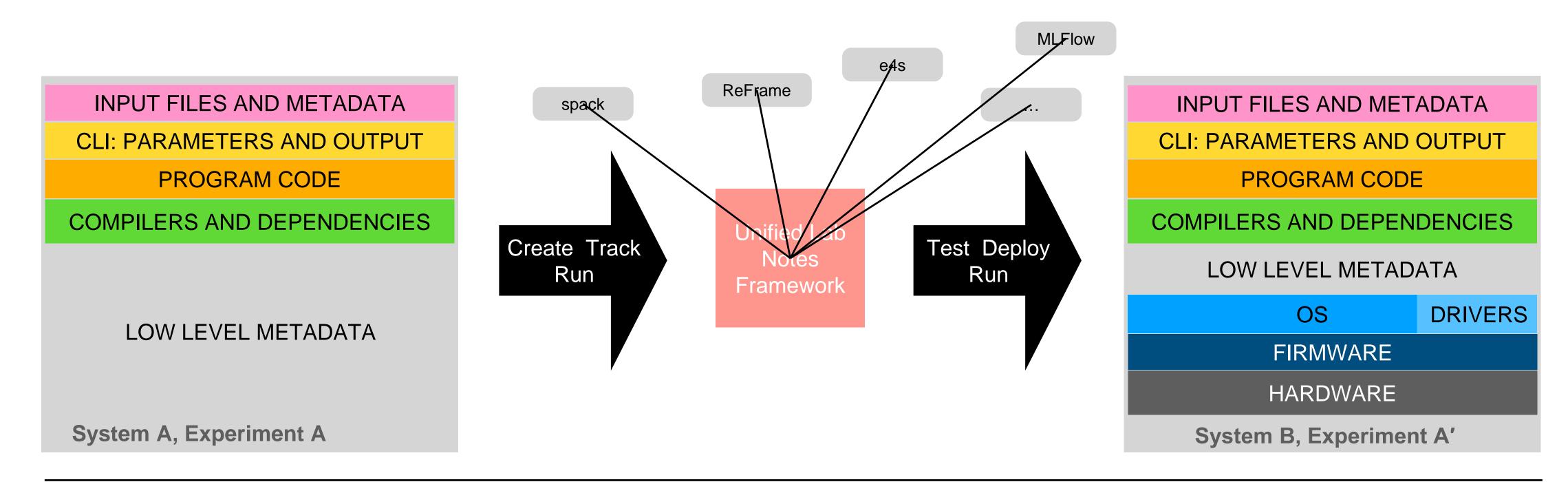
Center for Understandable, Performant Exascale Communication Systems

3. Generate Experiment A' with framework (based on Experiment A) 4. Deploy, test Experiment A' on System B

5. Run Experiment A' on System B

THE UNIVERSITY OF ALABAMA®

A Unified Lab Notes Framework



- 1. Create Experiment A with framework
- 2. Run Experiment A on System A



3. Generate Experiment A' with framework (based on Experiment A) 4. Deploy, test Experiment A' on System B

5. Run Experiment A' on System B

THE UNIVERSITY OF ALABAMA®

Contributions

- Productivity:

 - simpler and more accurate handoff
- Reproducibility:
 - development of standards
 - as a regression test
- Ongoing work:
 - software product/toolkit for release
 - technical documentation (reproducibility standards)
 - conference publication (1 or more)



new team members can easily get up to speed on existing experiments



Thanks for attending

Open for questions



