Towards a *declarative, reproducible, homogeneous, cross-platform command-line* environment across remote HPC machines

(with live demonstration pretending to be a masterhacker!)

The STEP-UP RSLondon Conference 2025
Dr Krishnakumar Gopalakrishnan, University College London
July 07, 2025

Who am I?

- Research Software Engineer at UCL ARC
- Rides bikes on weekends
- Uses Arch btw
- Test cricket fan

Who am I?

- Research Software Engineer at UCL ARC
- Rides bikes on weekends
- Uses Arch btw
- Test cricket fan

- Command-line enthusiast

```
# So in the ongoing Anderson-Tendulkar Trophy
```

———————————— [finished] ————————————

The Unix Philosophy

> This is the Unix philosophy:
> Write programs that do one thing and do it well.
> Write programs to work together.
> Write programs to handle ***text*** streams, because that is a universal interface.

Doug McIlroy, Bell Labs Computing Sciences Research Center

https://en.wikipedia.org/wiki/Unix_philosophy#Do_One_Thing_and_Do_It_Well

Command-line interfaces and HPC systems

- The default user interface on HPC systems
- Users are provided with Secure Shell (SSH) access
- Edit/Compile/Debug cycle on remote systems
- Job submission workflow and non-interactive compute node use that uses schedulers

Command-line interfaces and HPC systems

- The default user interface on HPC systems
- Users are provided with Secure Shell (SSH) access
- Edit/Compile/Debug cycle on remote systems
- Job submission workflow and non-interactive compute node use that uses schedulers

> ⓘ Note
>
> Often not emphasised: interactive use e.g. text editing, searching, command history ...

Towards a homogeneous environment

Goal

Need an environment identical to personal workstation on all remote hosts (HPCs, VMs, Containers ...)

Workstation platforms (OS/CPU Architectures)

Operating System	x86 (32-bit)	x86_64 (64-bit)	ARMv7 (32-bit)	ARM64 (AArch64)	RISC-V	PowerPC
Windows						
Linux						
*BSD						
macOS						
AIX						
Haiku						
ReactOS						
RedoxOS						

UCL Advanced Research Computing

Realistic support matrix for combined Desktop/HPC

Operating System	x86_64 (64-bit)	ARM64 (AArch64)
Linux	✅	✅
macOS	❓	✅
Windows	❓	❓

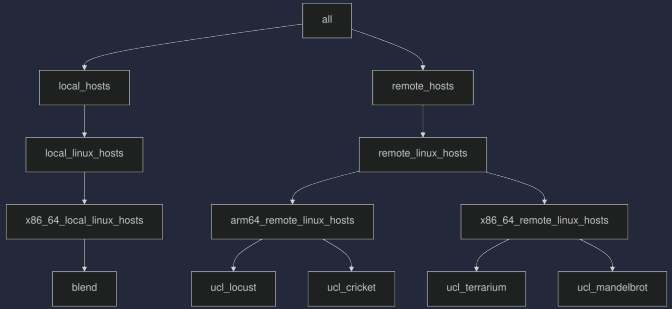
Common environment?

- Shell startup files (aliases, environment variables, cleanup ...)
- Cross-platform CLI/TUI tools
- Configuration files for tooling

Hands-on example

Scenario

- All hosts in the inventory



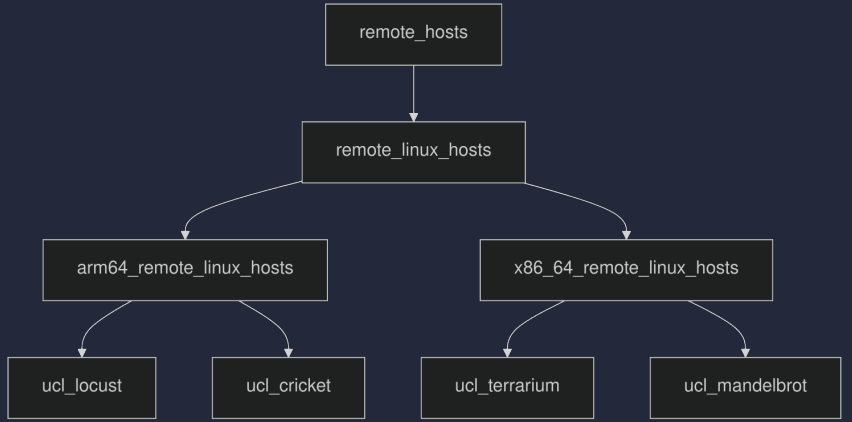
Consider only remote (HPC) hosts in the inventory

```
hostname
```

```
———————————————— [finished] ————————————————
```

```
blend
```

Remote hosts in the inventory



Config file synchronisation across hosts

- Clone/sync on remote host with a Version Control System (VCS)

- **A dotfiles manager** (*https://dotfiles.github.io/*)
  - GNU Stow
  - dotbot
  - rcm
  - and many many more!

Chezmoi

- I have settled on using Chezmoi (pronounced shay-mwa)
- Full power of go templates

```
# chezmoi data
```

```
──────────────────── [finished] ────────────────────
```

Chezmoi templating

```
$HOME/.config/git/config
[user]
    {{- if eq chezmoi.hostname "blend" }}
    email = "personal_email@personal_host.co.uk"
    {{- else }}
    email = "work_email@work.ac.uk"
    {{- end }}
```

```
user@blend $ chezmoi apply
```

```
[user]
    email = "personal_email@personal_host.co.uk"
```

Deploy on all systems

Use a configuration management tool

- Ansible (agentless, push model)
- Puppet
- Chef
- Salt stack
- Pyinfra

Tooling/Packages

Combinatorial explosion of packages

> Consider
> m: different processor architectures (target CPUs),
> n: different operating systems,
> p: different executable tools.

Total number of binaries to be managed:

$$\leq m \times n \times p$$

Even with 3 CPU architectures and 20 tools to be managed, we have about 60 executables to manage, from which the valid binaries for each platform must be identified and deployed on the p-th machine.

Managing package lifecycle

- Install (?), Update to latest, Pin versions of selected packages/tools, Remove packages

Managing package lifecycle

Install (?), Update to latest, Pin versions of selected packages/tools, Remove packages

```
act
actionlint
age
asdf:richin13/asdf
-neovim
awk-language-serve
r
bat
bat-extras
beautysh
bingrep
binsider
biome
bottom
broot
cargo:tlrc
ccache
checkmake
chezmoi
clang-tools
cmake
cosign
cppcheck
croc
curl
delta
difftastic
direnv
diskonaut
dive
fx
fzf
gdu
gh
git
git-cliff
gitui
glow
go
go:crictty
go:daylight
go:kubecolor
go:stew
goawk
graphviz
jinja-lsp
jless
jnv
jq
k9s
kubectl
lazydocker
lazygit
lazyjournal
less
ltex-ls
lua
lua-language-serve
r
lychee
```

Linters, Formatters, Language Runtimes, Compilation frameworks, Utilities ...

```
minikube
nasm
neocmakelsp
ninja
node
npm:ansible-langua
ge-server
npm:mermaid-cli
npm:compose-langua
ge-service
npm:alex
npm:bash-language-
server
npm:bibtex-tidy
npm:commitlint
```

```
pipx
pipx:ansible-d
ev-tools
pipx:ansible-i
nventory-graph
er
pipx:ansible-l
int
pipx:argcomple
te
pipx:bandit
pipx:basedpyri
ght
pipx:clang-for
mat
```

```
procs
pylyzer
python
ripgrep
ripgrep-all
ruff
rust
selene
shellcheck
shellharden
shfmt
starship
stylua
taplo
terraform
```

```
usage
uv
vault
vlang
viddy
xz
yamlfmt
yamllint
yj
zig
zk
```

```
# Let's count the
```

Advanced Research Computing

How to manage these software tools/packages everywhere?

System package manager?

- apt-get/apt
- yum/dnf
- zypper
- pacman

Let's try

```
which vim
```

[finished with error]

```
which: no vim in
(/home/chezmoi_trial1/.config/pixi/condabin:/home/chezmoi_trial1/.local/bin:/home/
chezmoi_trial1/.local/share/mise/installs/act/0.2.79:/home/chezmoi_trial1/.local/s
hare/mise/installs/actionlint/1.7.7:/home/chezmoi_trial1/.local/share/mise/install
s/age/1.2.1/age:/home/chezmoi_trial1/.local/share/mise/installs/bat/0.25.0:/home/c
hezmoi_trial1/.local/share/mise/installs/bat-extras/2024.08.24/bin:/home/chezmoi_t
rial1/.local/share/mise/installs/biome/2.0.6:/home/chezmoi_trial1/.local/share/mis
```

Let us install vim

```
# which OS are we on?
```

```
──────────────────── [finished] ────────────────────
```

```
NAME="blendOS"
PRETTY_NAME="blendOS"
ID=blendos
BUILD_ID=rolling
ANSI_COLOR="38;2;23;147;209"
HOME_URL="https://blendos.co/"
DOCUMENTATION_URL="https://docs.blendos.co/"
SUPPORT_URL="https://github.com/blend-os"
BUG_REPORT_URL="https://github.com/blend-os"
LOGO=blendos-logo
```

```
pacman -S vim
```

———————— [finished with error] ————————

```
error: you cannot perform this operation unless you are
root.
```

Cannot do this on remote HPC systems. But let us on our local machine

```
su krishnakumar
sudo pacman -S vim
```

Software Package Management on a HPC system

- Curated list of scientific software pre-installed (and upon request)
- Versions are user-switchable by a modules system (environment-modules/Lmod)
- Custom user-modules can be loaded
- Unpriveleged container execution with container runtimes like Apptainer/Singularity/Podman?

Software Package Management on a HPC system

- Curated list of scientific software pre-installed (and upon request)
- Versions are user-switchable by a modules system (environment-modules/Lmod)
- Custom user-modules can be loaded
- Unpriveleged container execution with container runtimes like Apptainer/Singularity/Podman?

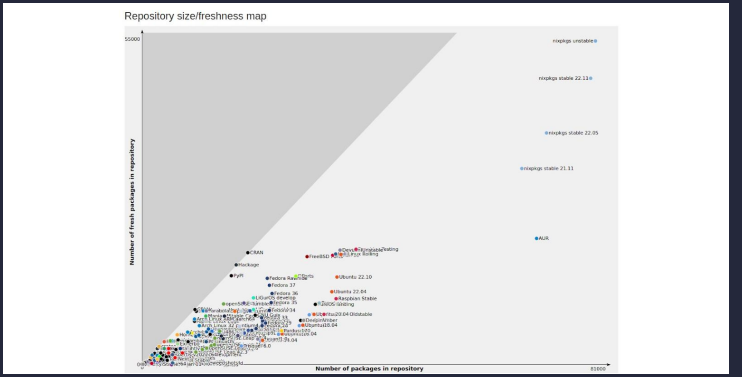
But we are talking about installing simple productivity tools and keeping them up-to-date
Not scientific libraries and programs

Popular HPC packaging ecosystem

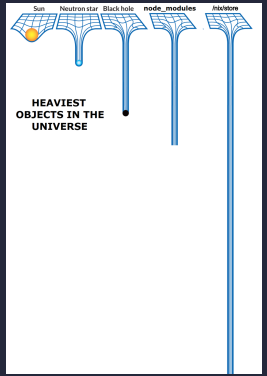
- Intended for scientific software and libraries (GROMACS, LAMMPS, MPI ...)
- Support complex build provenances (MPI libraries, dependency versions, compilation toolchain ...)
- Latest versions of utility tools often not available

So, what are the options available?

Nix was it for me!



However!


Sun
Neutron star
Black hole
node_modules
/nix/store
HEAVIEST
OBJECTS IN THE
UNIVERSE

So, is there a solution?

Desired characteristics

- Declarative/Idempotent
- Reproducible
- Installable/Runnable from user home directory
- Cross-platform

pixi global

- install
- edit
- sync
- manifest file `pixi-global.toml`

pixi global

- install
- edit
- sync
- manifest file `pixi-global.toml`

Mise

- `use -g`
- `prune`
- global `config.toml`

Keeping tools up to date

Further resources

- Worflow configuration codes, presentation in HTML & PDF formats

UCL Advanced Research Computing