

# POD

PROOF on Demand

PROOF on Demand

Dynamic PROOF clusters with PoD.  
Status report and future plans

# What is PoD?

PROOF on Demand (PoD) is a tool-set, which sets up a PROOF cluster on any resource management system.

\*\*\*

PoD is NOT a substitution of PROOF!  
It is rather a helper tool for PROOF.

\*\*\*

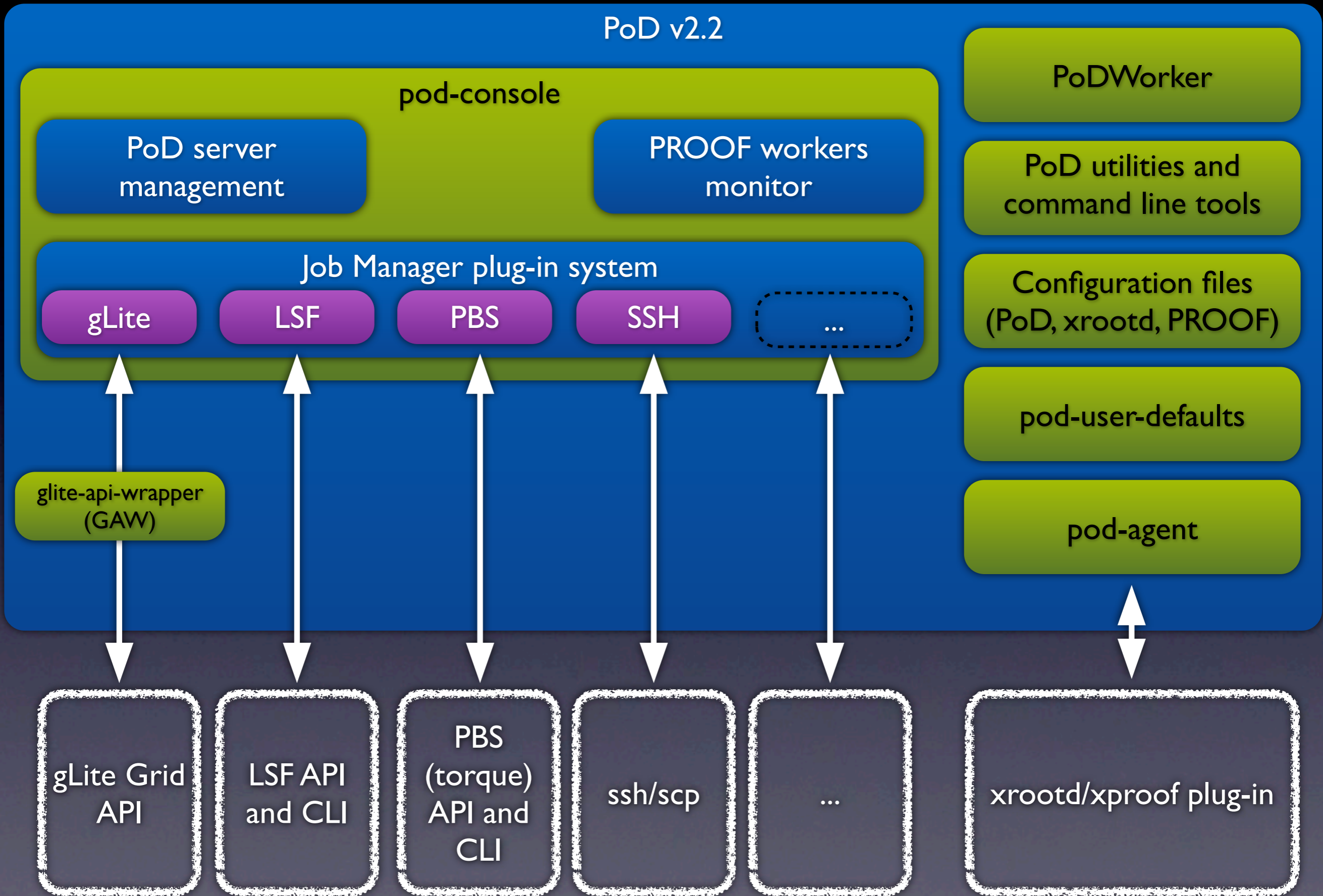
# “static”/pre-installed PROOF cluster

- One user can disturb other users.
- From time to time admin. interventions are needed.
- There is only one ROOT/xrootd version for PROOF services.
- There is a master node limitation.

# PROOF via PoD

## User

- can entirely control his/her dynamic cluster,
- can setup and use it on demand,
- can dynamically change an amount of workers,
- can select a preferable master host,
- doesn't need admins to take an action,
- doesn't disturb other users,
- is free to choose a ROOT version for services.



# Key features

- Easy to use
- GUI & Command-line
- Different job managers (gLite, LSF, PBS, SSH)
- Multiuser/-core environment
- Native PROOF connections
- Packet-forwarding
- User defaults - configuration

# Current status

v2.1.4 has been released (2010-05-26)

v2.2 - under development

As the product matures, the support of previous versions  
is extending...

# Improved build-system

- The project has been consolidated and cmake is now the only build system of it.
- A much easier/smarter installation procedure has been released.
- Implemented fully automated builds of releases, nightly and specific binary packages.
- Since years the buildbot serves as a continues integration system.



# New version numbers

Old:

maintained manually in VCS and in build config. files.

Old schema: “vX.Y.Z” (X - major, Y - minor, Z - patch).

New:

only set a tag (“v.X.Y”) in VCS (we use Git) and the rest is done automatically, using Git canonical ver. numbers.

“vX.Y” - for a release build

“vX.Y.NN.SSSS” (vX.Y - the latest tag, NN - a number of commits since the tag, and SSSS - object's hash) - for any patch and nightly.

# CLI

- pod-server
- pod-info
- pod-submit
- pod-user-defaults
- pod-prep-worker

# CLI

- pod-server
- pod-info
- pod-submit
- pod-user-defaults
- pod-prep-worker

## Example

```
$ pod-server start (status)
$ pod-submit -r lsf -q queue -n 150
$ pod-info -n (-l)
$ pod-server stop
```

# CLI

- pod-server
- pod-info
- pod-submit
- pod-user-defaults
- pod-prep-worker

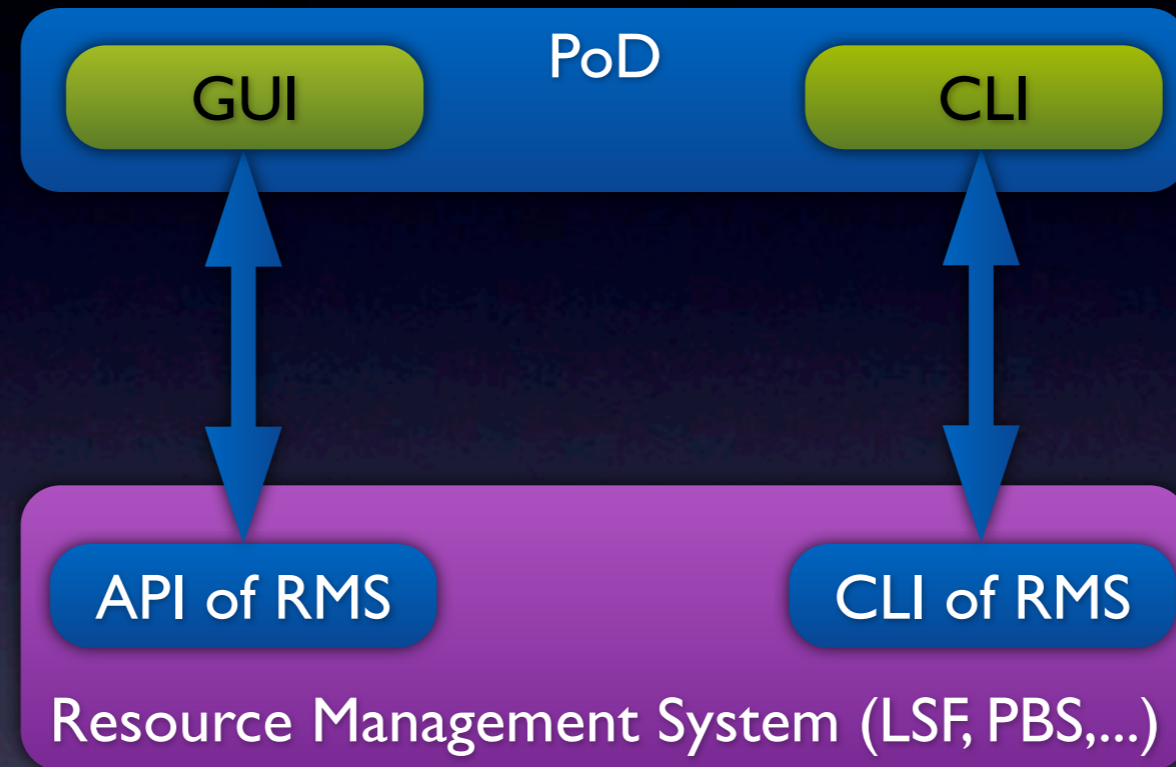
## Example

```
$ pod-server start (status)
$ pod-submit -r lsf -q queue -n 150
$ pod-info -n (-l)
$ pod-server stop
```

can be used in a PROOF script

```
TProof::Open( gSystem->GetFromPipe("pod-info -c") )
```

# GUI vs CLI



This strategy helps to be more flexible and better integrated into different setups and environments.

# gLite plug-in

Updated to support gLite UI 3.2

# PBS plug-in

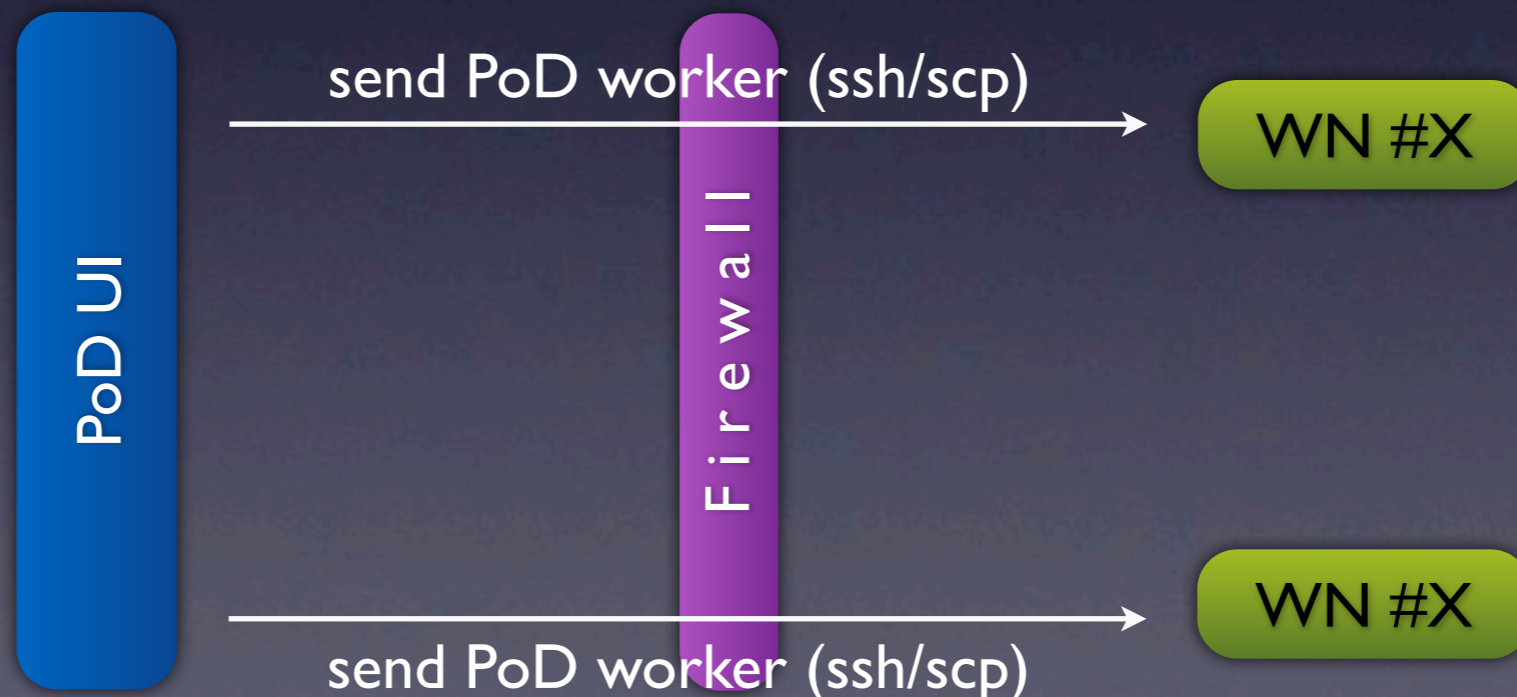
## Supports

- PBS Pro and Torque,
- shared and not shared file systems.

# SSH plug-in

a simple CSV file as an input to the plug-in:

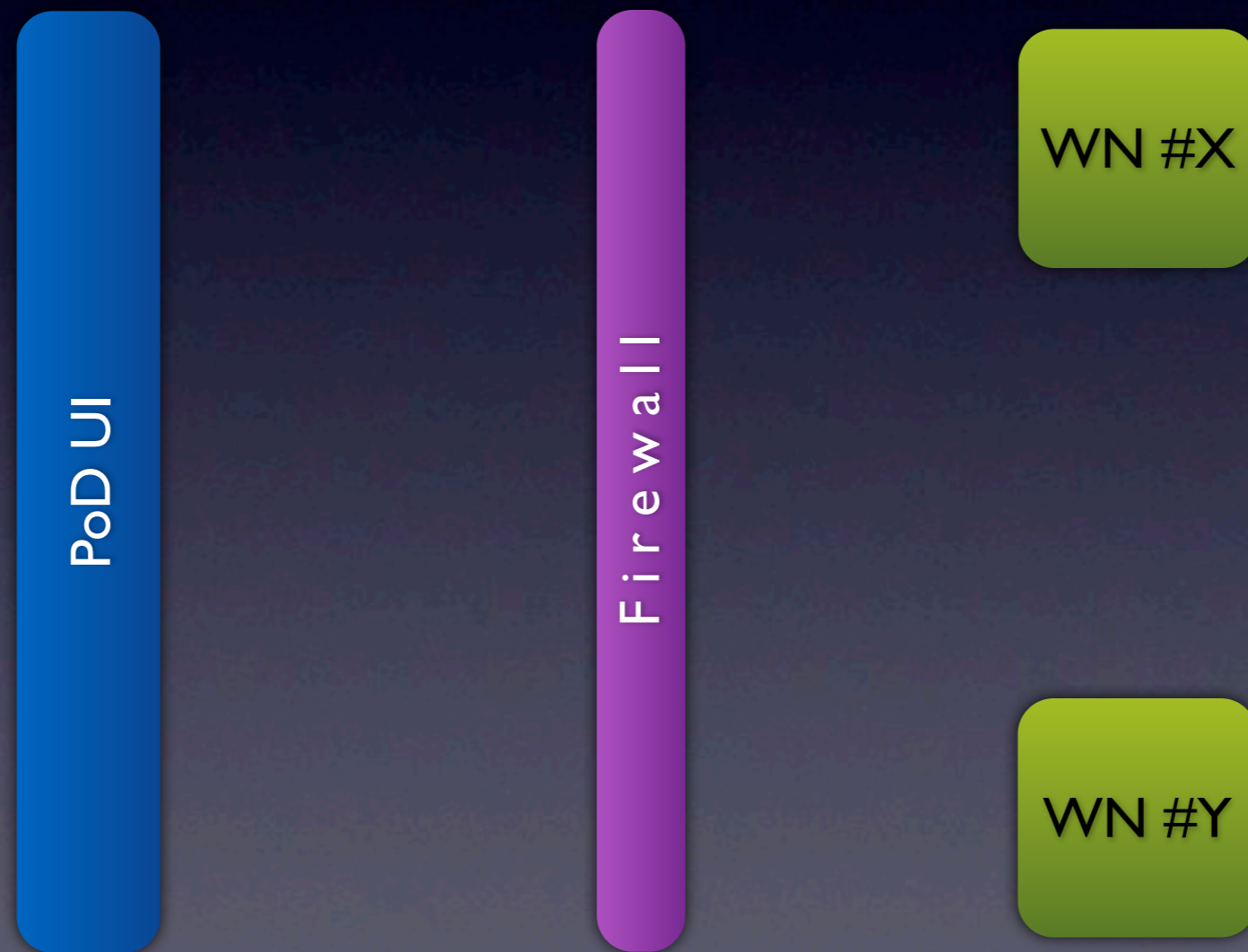
```
id , login@host , ssh_params , wn_dir , num_of_workers
r1 , anar@lxcg27.gsi.de , -p24 , /tmp/test , 10
a2 , user@lxi001 , , ~/pod_wn , 8
125 , doom@host.my , -p22 , /opt/pod , 16
```





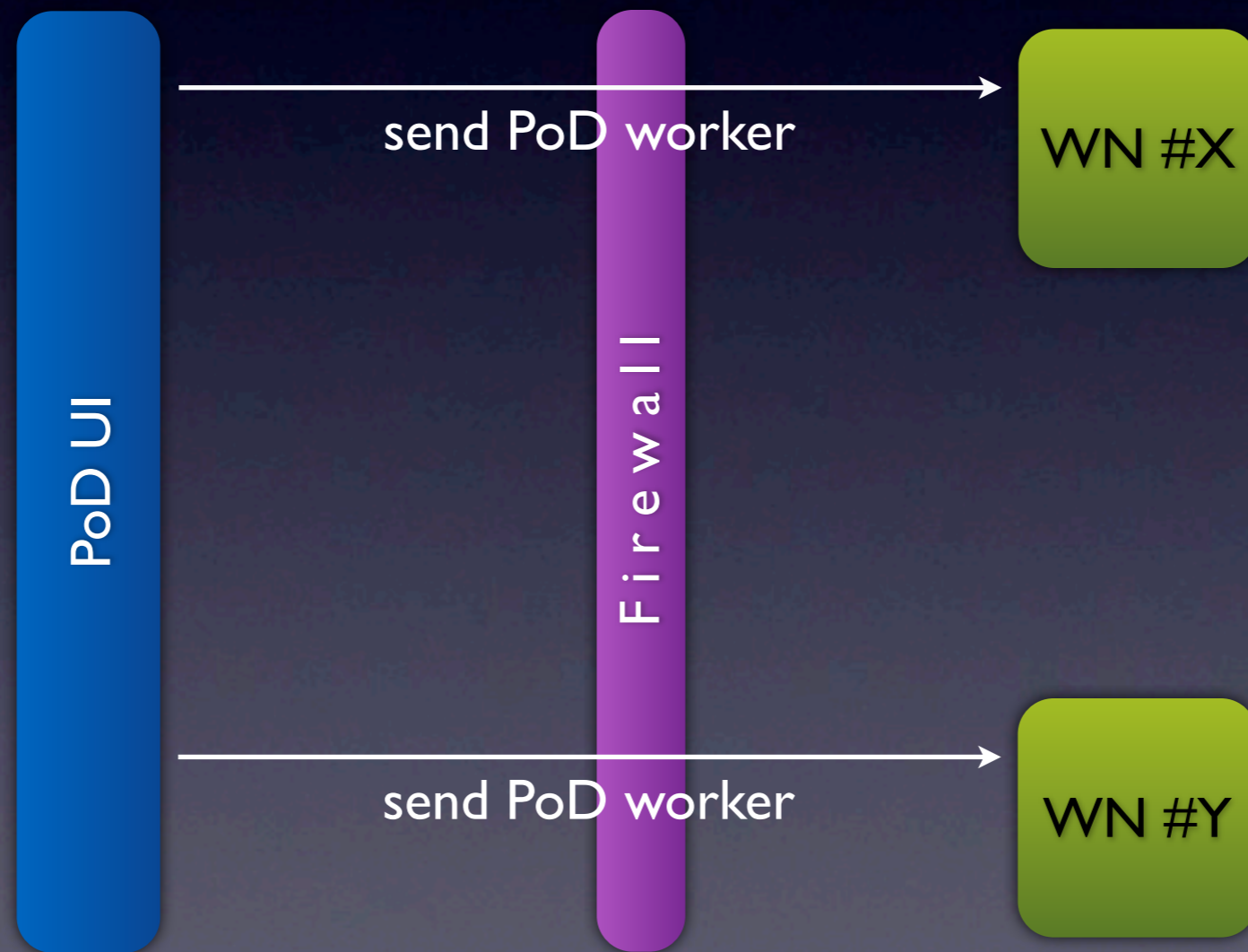
# SSH plug-in & tunnel

PoD will optionally create ssh tunnels for nodes behind a firewall.



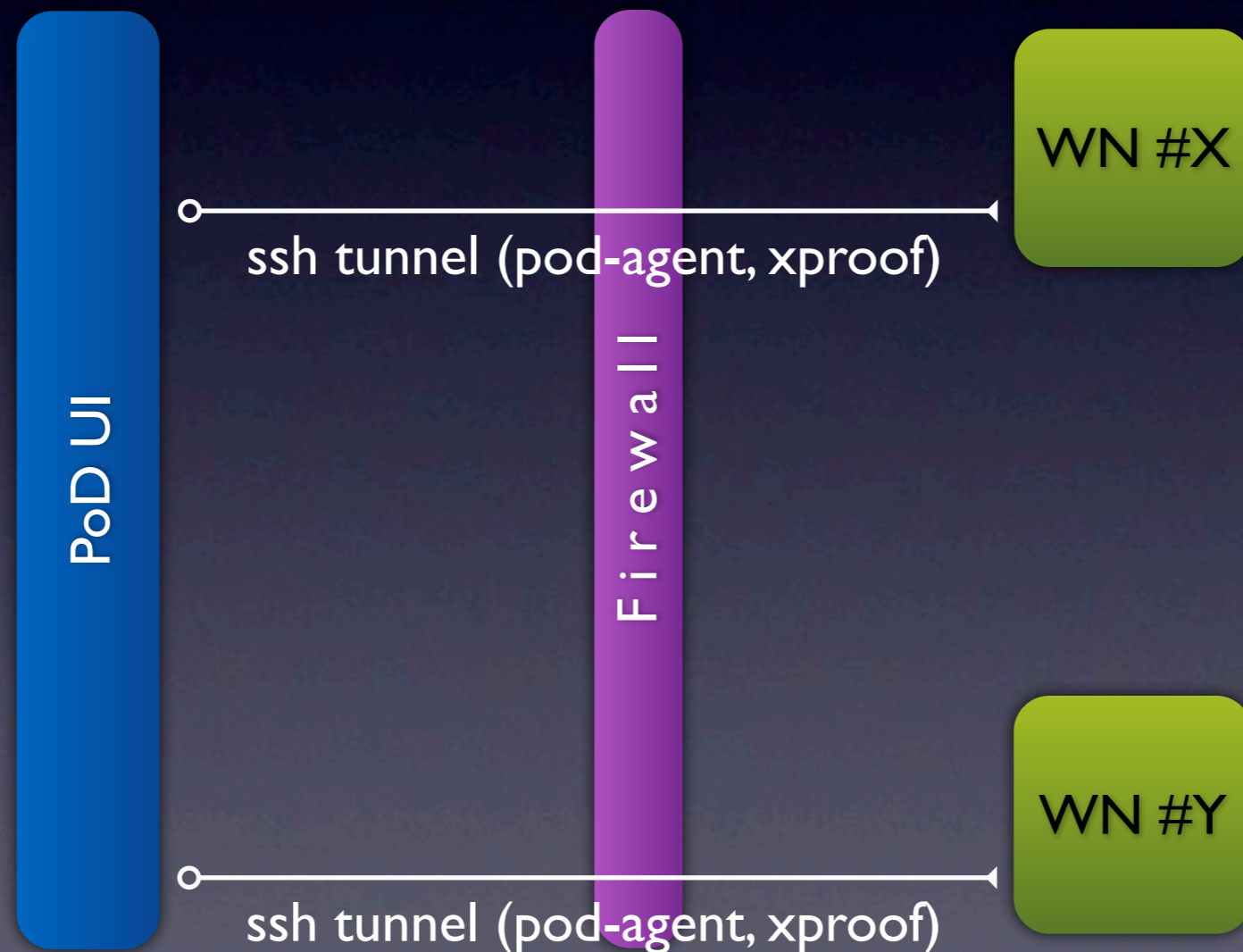
# SSH plug-in & tunnel

PoD will optionally create ssh tunnels for nodes behind a firewall.



# SSH plug-in & tunnel

PoD will optionally create ssh tunnels for nodes behind a firewall.



# SSH plug-in & cloud

Possible workflow:

- distribute data files,
- prepare an OS image, which includes PoD's worker package (made by pod-prep-worker),
- send the image to a cloud provider, requesting an ssh access to nodes,
- pass the list of workers to PoD ssh plug-in.

Enjoy your cloud based PROOF cluster.

# ToDo

- “out of server” UI,
- a native Mac OS X implementation of UI,
- an AliEn plug-in,
- a Condor plug-in.

<http://pod.gsi.de>