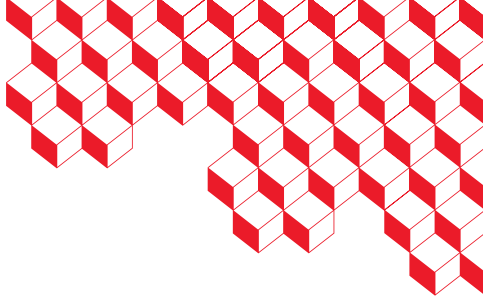


The testing infrastructure of EPNix

Rémi NICOLE /  minijackson

CEA/DRF/IRFU/DIS/LDISC

1. What's EPNix?

Tagline

Build, package, deploy IOCs and EPICS-related software using the Nix package manager.

EPNix logo

Tagline

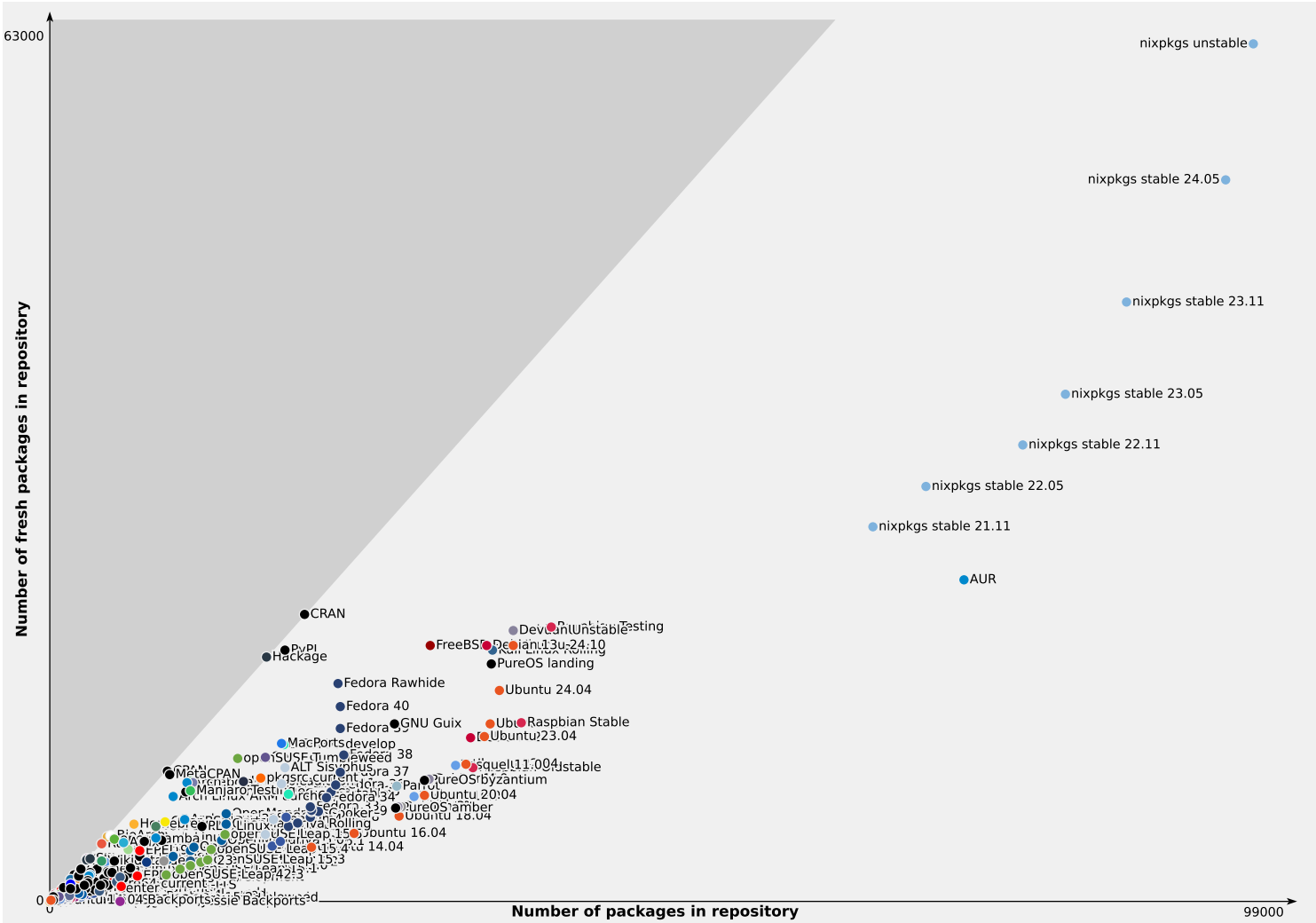
Build, package, deploy IOCs and EPICS-related software using the Nix package manager.

EPNix logo

Documentation: https://epics-extensions.github.io/EPNix/

Nix' buzzwords

- Infrastructure as code (part of it)
- Reproducibility
- Software supply chain security (SLSA)
- Software Bill of Materials (SBOM)


63000
Number of fresh packages in repository
0
Number of packages in repository
99000
nixpkgs unstable
nixpkgs stable 24.05
nixpkgs stable 23.11
nixpkgs stable 23.05
nixpkgs stable 22.11
nixpkgs stable 22.05
nixpkgs stable 21.11
AUR
CRAN
PyPI
Hackage
FreeBSD
PureOS landing
Fedora Rawhide
Ubuntu 24.04
Fedora 40
GNU Guix
Raspbian Stable
Ubuntu 23.04
MacPorts devel
Fedora 38
ALT Sisyphus
Fedora 37
MetaCPAN
Ubuntu 20.04
Fedora 34
PureOS amber
Ubuntu 18.04
Ubuntu 16.04
Ubuntu 14.04
openSUSE Leap 42.3

2. Nix concepts

Definitions

Nix the package manager
Nix the programming language
Nixpkgs the Repository
NixOS the Linux Distribution

Definitions

Nix the package manager

Nix the programming language

Nixpkgs the Repository

NixOS the Linux Distribution

declarative, reproducible

Example NixOS configuration

```
{pkgs, ...}:
{
  environment.systemPackages = [pkgs.htop pkgs.vim];

  users.users.admin = {
    isNormalUser = true;
    extraGroups = ["wheel"];
  };

  services.openssh = {
    enable = true;
    settings.PermitRootLogin = "no";
  };
}
```

Example NixOS configuration with EPNix

```
{
  # After importing EPNix, you can do:
  services.archiver-appliance = {
    enable = true;

    stores.lts.location = "/data/lts";
    stores.mts.location = "/data/mts";
    stores.sts.size = "20g";

    openFirewall = true;
  };
}
```

What this does:

- Creation of the Linux user and group `archappl`
- Installation and configuration of MariaDB:
  - Creation of the `archappl` MariaDB user, with UNIX socket authentication
  - Creation of the Archiver Appliance database
  - Creation of the various tables in that database
  - Giving access rights to this database for the `archappl` user
- Installation and configuration of Tomcat:
  - Installation of the WAR files of Archiver Appliance
  - Installation of the MariaDB connector and its dependencies
  - Configuring the MariaDB connector to authenticate to the database
  - Logging configuration to `journald`
- Configuring mounts so that:
  - `/arch/lts` and `/arch/mts` are bind mounts to the configured locations, with some added security options, such as `nodev` and `noexec`
  - Mounting `/arch/sts` as a new `tmpfs` with the configured size

3. Testing

Implications for testing

We have a configuration that declares an entire system.

Implications for testing

We have a configuration that declares an entire system.

Wouldn’t it be nice if we could generate images with that?

Available NixOS images

- amazon
- azure
- cloudstack
- do
- docker
- gce
- hyperv
- install-iso
- install-iso-hyperv
- iso
- kexec
- kexec-bundle
- kubevirt
- linode
- lxc
- lxc-metadata
- openstack
- proxmox
- proxmox-lxc
- qcow
- qcow-efi
- raw
- raw-efi
- sd-aarch64
- sd-aarch64-installer
- sd-x86_64
- vagrant-virtualbox
- virtualbox
- vm
- vm-bootloader
- vm-nogui
- vmware

Available NixOS images

- amazon
- azure
- cloudstack
- do
- docker
- gce
- hyperv
- install-iso
- install-iso-hyperv
- iso
- kexec
- kexec-bundle
- kubevirt
- linode
- lxc
- lxc-metadata
- openstack
- proxmox
- proxmox-lxc
- qcow
- qcow-efi
- raw
- raw-efi
- sd-aarch64
- sd-aarch64-installer
- sd-x86_64
- vagrant-virtualbox
- virtualbox
- vm
- vm-bootloader
- vm-nogui
- vmware

NixOS tests

Given one or more NixOS configurations:

1. Generate VM images
2. Start those VMs
3. Run commands on it

NixOS tests

Given one or more NixOS configurations:

1. Generate VM images
2. Start those VMs
3. Run commands on it
   - Check the state of configured services
   - Simulate user interaction
   - Assert invariants

4. EPNix' CA gateway test

1st IOC machine

```
# Test two IOC on their own network,
# but only one in the ADDR_LIST of the gateway
ioc = {
  imports = [
    (softIoc ''
      record(ai, "PV_CLIENT") { }
    '')
  ];
  virtualisation.vlans = [1];
};
```

2nd IOC machine

```
invisible_ioc = {
  imports = [
    (softIoc ''
      record(ai, "PV_INVISIBLE_CLIENT") { }
    '')
  ];
  virtualisation.vlans = [1];
};
```

3rd IOC machine

```
# Test one IOC in its own network,
# but put the broadcast address
# in the ADDR_LIST of the gateway.
# Useful for testing the openFirewall option
ioc_broadcast = {
  imports = [
    (softIoc ''
      record(ai, "PV_FROM_BROADCAST") { }
    '')
  ];
  virtualisation.vlans = [2];
};
```

Gateway machine

```
gateway = {
  services.ca-gateway = {
    enable = true;
    openFirewall = true;
    settings = {
      # One unicast, one broadcast
      cip = ["ioc" "192.168.2.255"];
    };
  };

  virtualisation.vlans = [3 1 2];
};
```

Client machine

```
client = {
  environment.systemPackages = [pkgs.epnix.epics-base];
  virtualisation.vlans = [3];
};
```

Setup summary

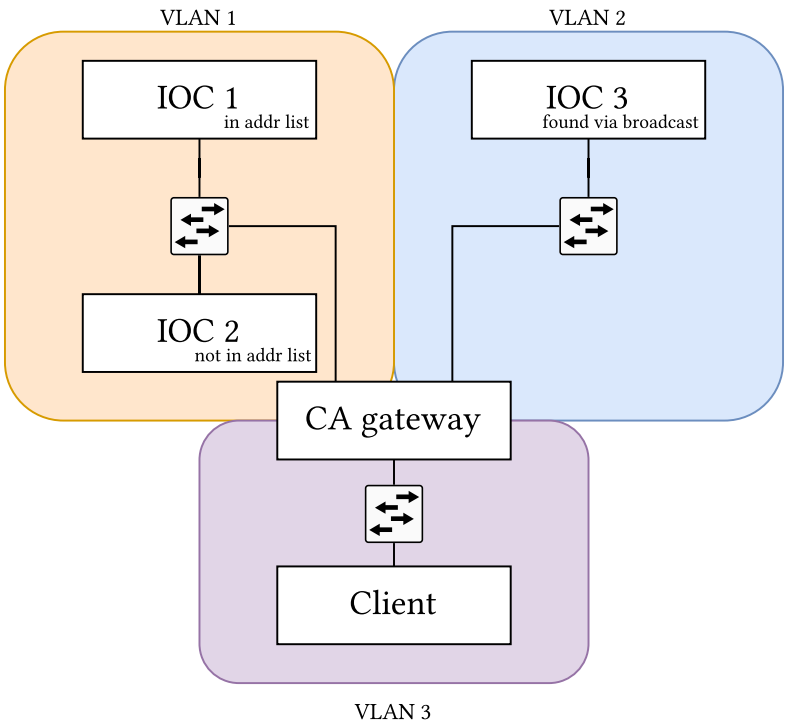


Figure 2: Channel Access gateway test setup

Test script

```
start_all()

gateway.wait_for_unit("ca-gateway.service")
ioc.wait_for_unit("ioc.service")
invisible_ioc.wait_for_unit("ioc.service")
ioc_broadcast.wait_for_unit("ioc.service")
client.wait_for_unit("multi-user.target")

def caget(pv: str) → str:
    return f"EPICS_CA_ADDR_LIST=gateway caget {pv}"

client.wait_until_succeeds(caget("PV_CLIENT"))
client.wait_until_succeeds(caget("PV_FROM_BROADCAST"))
client.fail(caget("PV_INVISIBLE_CLIENT"))
```

How to run the test

- Install Nix
  - Works on any Linux distribution, and WSL2
- Clone the EPNix repository
- `nix build -L '.#checks.x86_64-linux.ca-gateway'`

Tests currently in EPNix

- Channel Access gateway
- Phoebus alarm (server & logger)
- Phoebus Olog
- Phoebus save-and-restore
- Archiver Appliance
- ChannelFinder

- default IOC from `makeBaseApp.pl`
  - for EPICS base 3 and 7
- example IOC from `makeBaseApp.pl`
- cross-compiling to various architectures
- pyepics
- StreamDevice
- autosave
- pvxs (IOC, qsrv2, and standalone)
- sncseq

Tests currently in EPNix

- Channel Access gateway
- Phoebus alarm (server & logger)
- Phoebus Olog
- Phoebus save-and-restore
- Archiver Appliance
- ChannelFinder

- default IOC from `makeBaseApp.pl`
  - for EPICS base 3 and 7
- example IOC from `makeBaseApp.pl`
- cross-compiling to various architectures
- pyepics
- StreamDevice
- autosave
- pvxs (IOC, qsrv2, and standalone)
- sncseq

Example tests in Nixpkgs

- Test of OpenArena, a first person shooter game
  1. Start 3 VMs: a game server, and 2 clients
  2. Wait for the machines to boot, and for the game server to start
  3. Start the game on the 2 clients, and connect them to the server
  4. Disconnect the “ethernet cable” of client1, wait for 10 seconds, and reconnect
  - Make screenshots throughout the process
- BitTorrent test
  1. Start 4 VMs: a tracker in VLAN 1, a router in VLAN 1 & 2, a client in VLAN 1, and a client in VLAN 2
  2. Make client1 download a torrent from the tracker
  3. We stop the tracker
  4. Make client2 download a torrent from client1, through the router

Some advantages

- You can run them locally!
- Easy to integrate in CI
- You can run them interactively
- You can run them in parallel
- They run in VMs, so no pollution
- Tests scripts are in Python, very versatile

Some drawbacks

- Slow
  - For most Phoebus tests, you have to wait for ElasticSearch to start on each test run
  - Tests always run from the beginning
- Issues with WSL due to hardware acceleration taken by Hyper-V

That's it!

Always happy to receive questions

Documentation: https://epics-extensions.github.io/EPNix/