The case for Nix on the home server

By Anthony Tarbinian & Samir Rashid

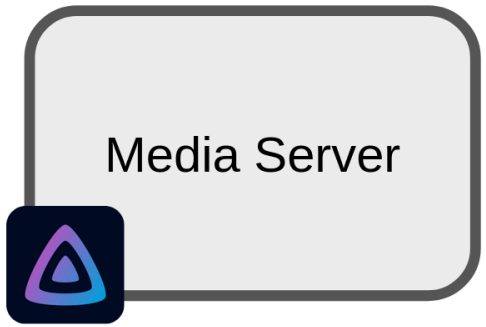
Nixcon NA, March 15, 2024

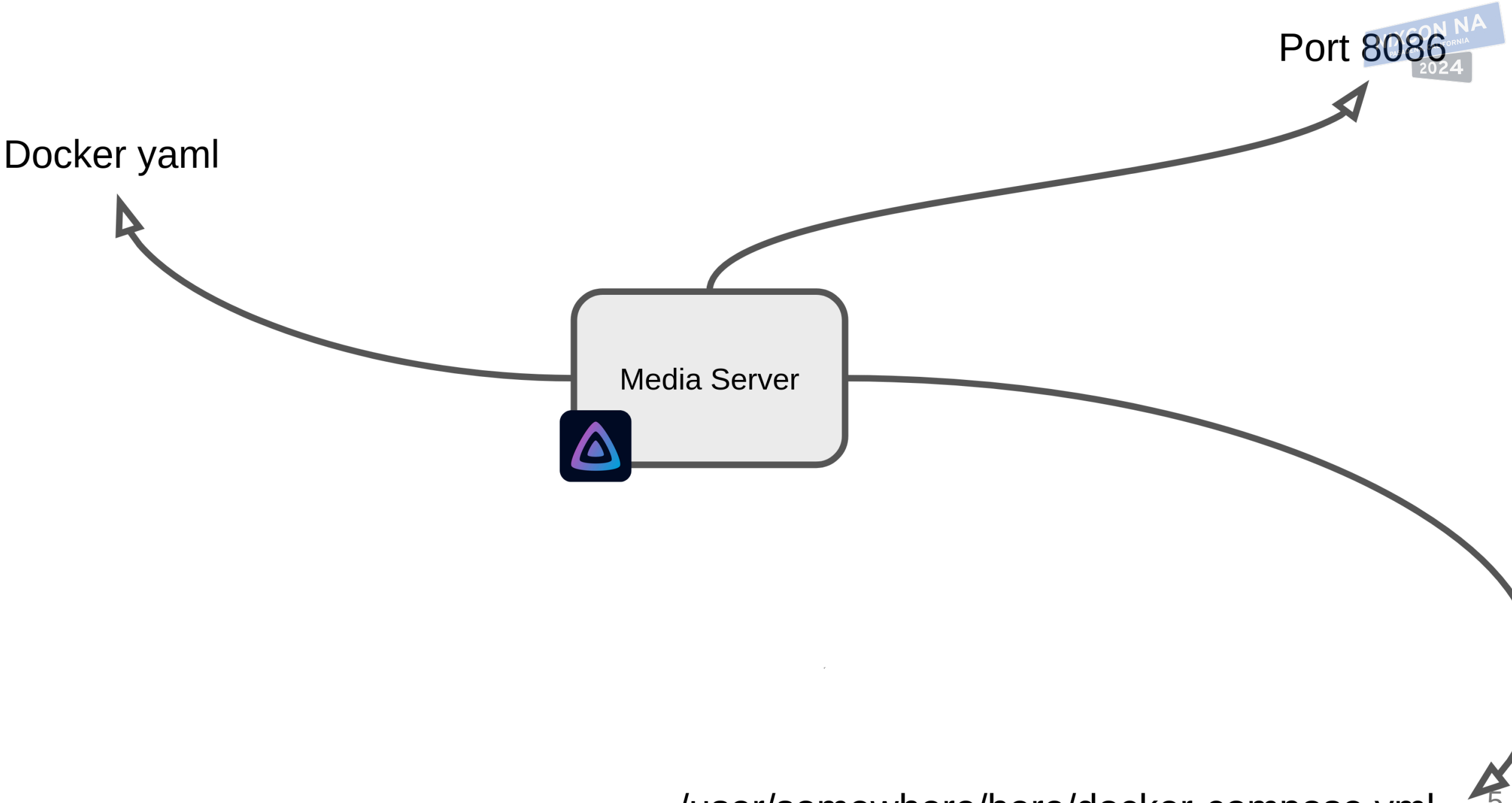
Presentation permalink:

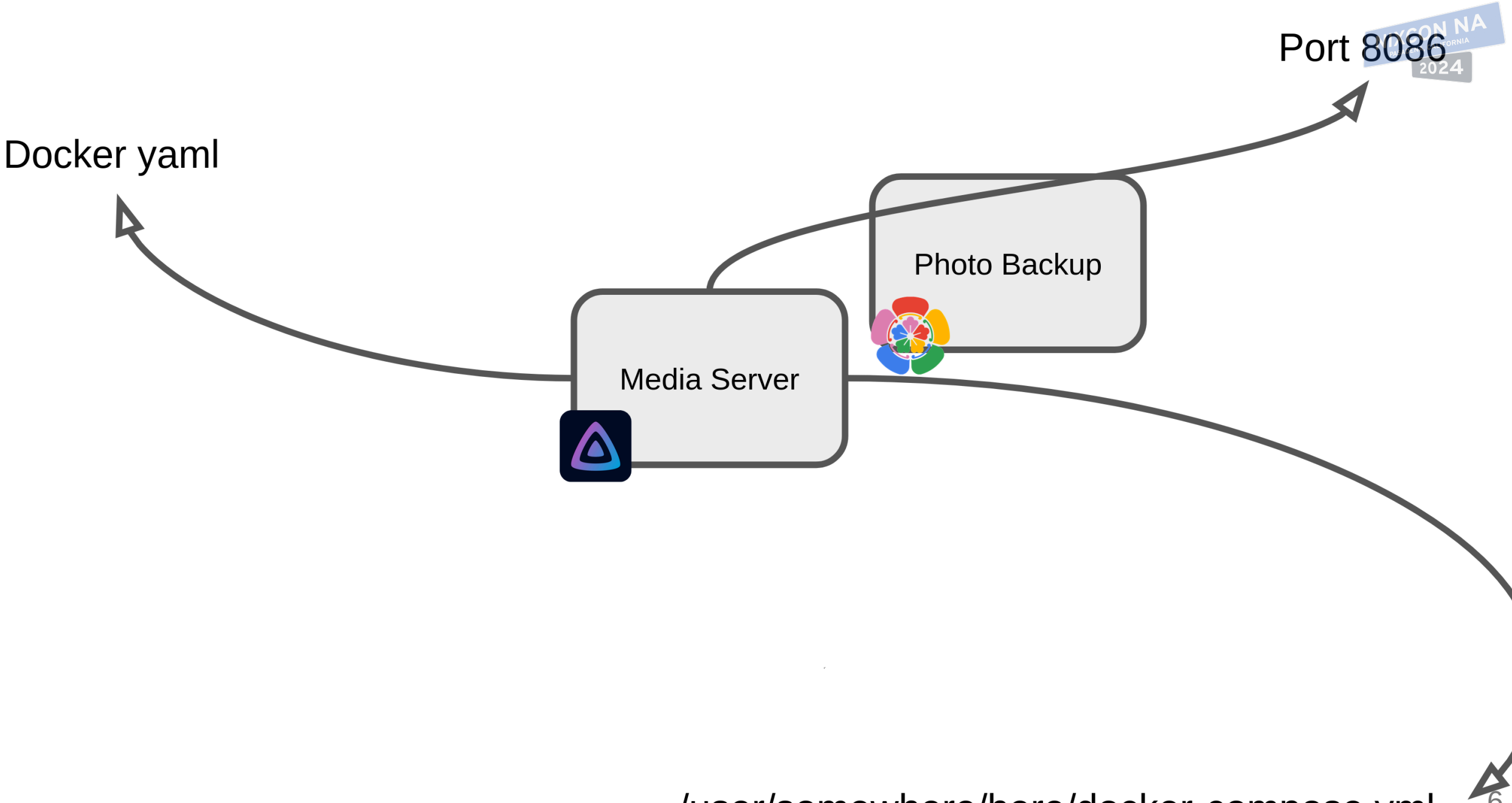
godsped.com/nix-homeserver

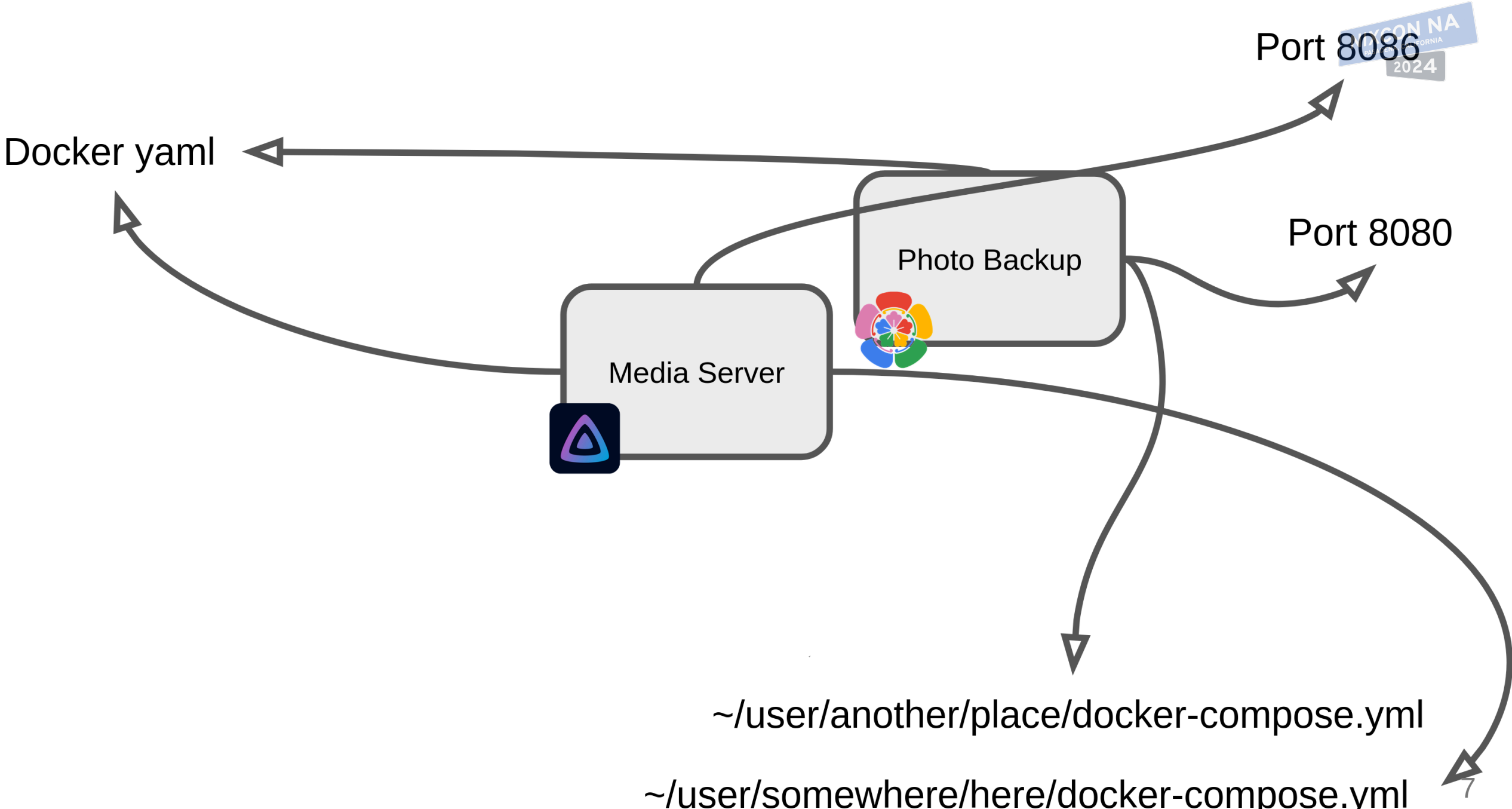

Beelink®
EQ

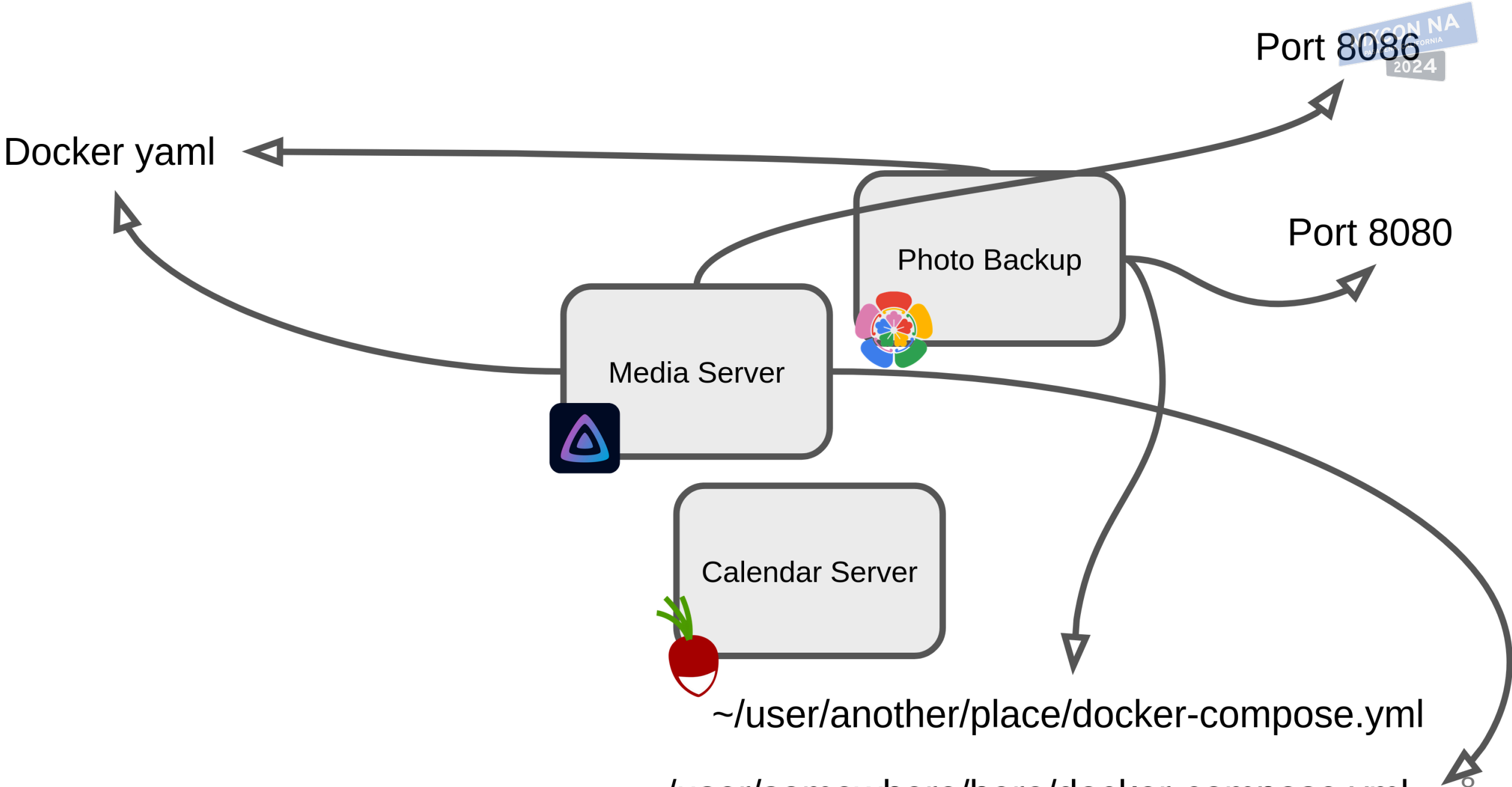

NIXCON NA
PASADENA, CALIFORNIA
2024

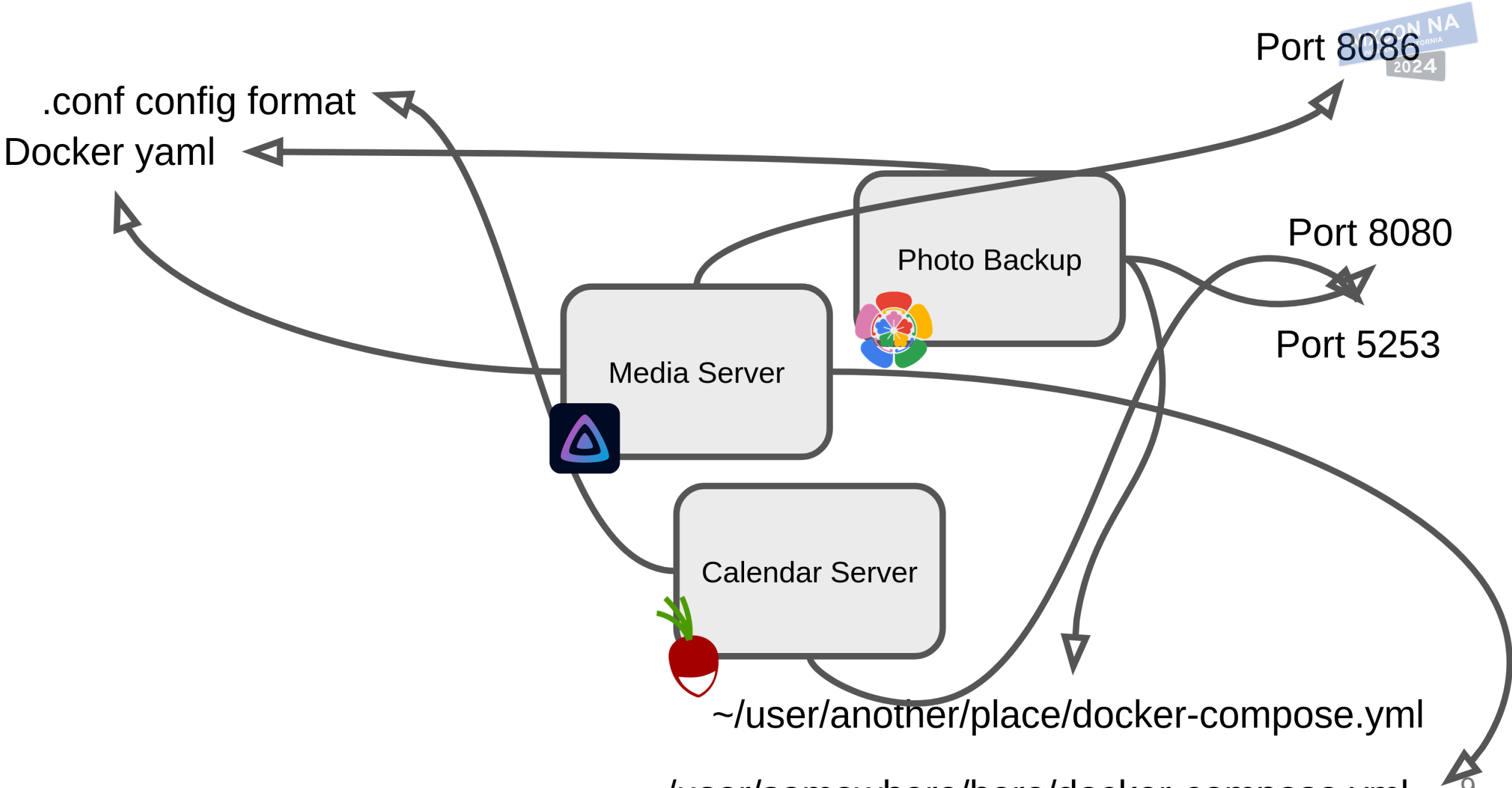

Media Server

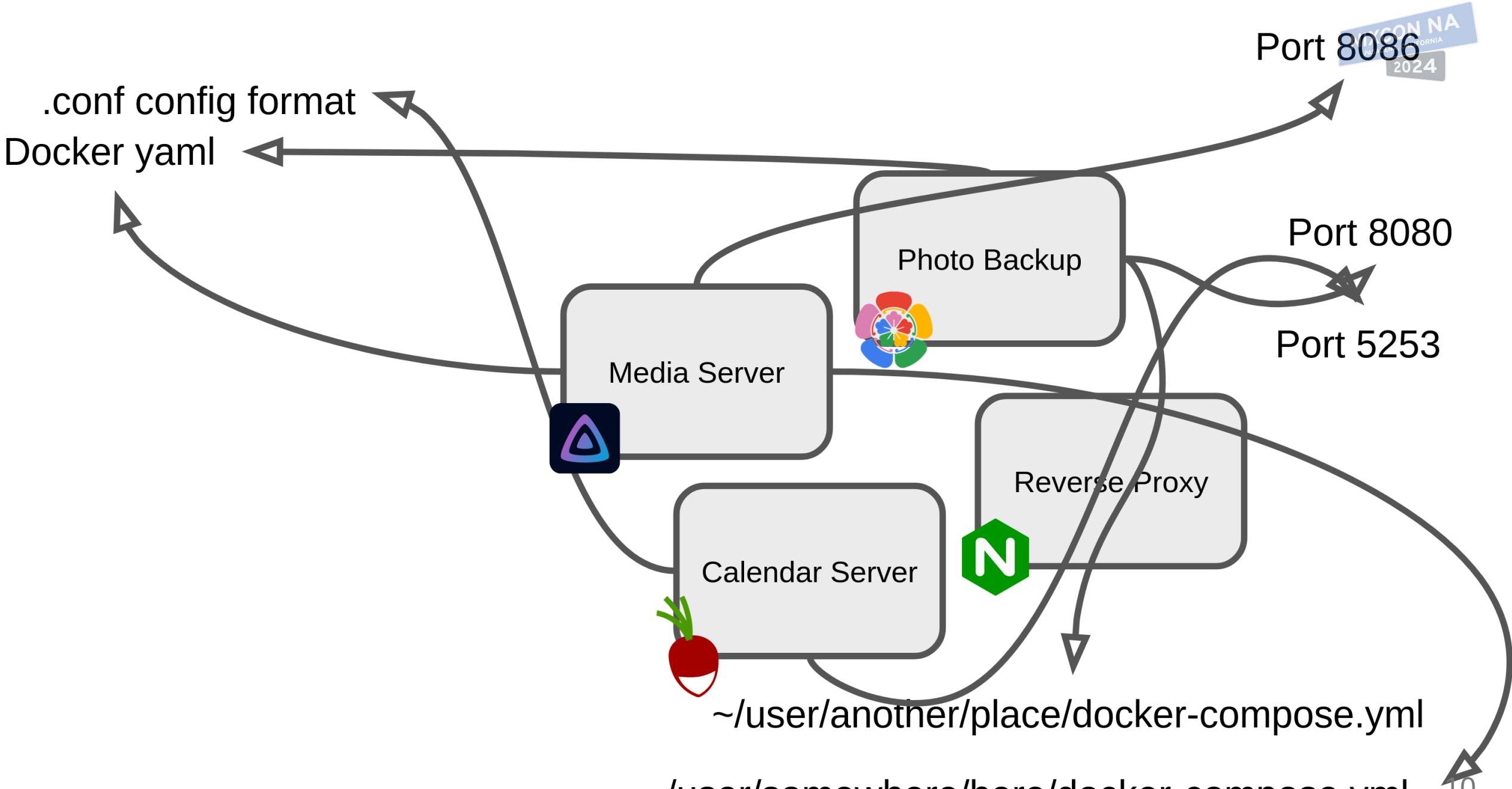

Port 8086
Docker yaml
Media Server
~/user/somewhere/here/docker-compose.yml

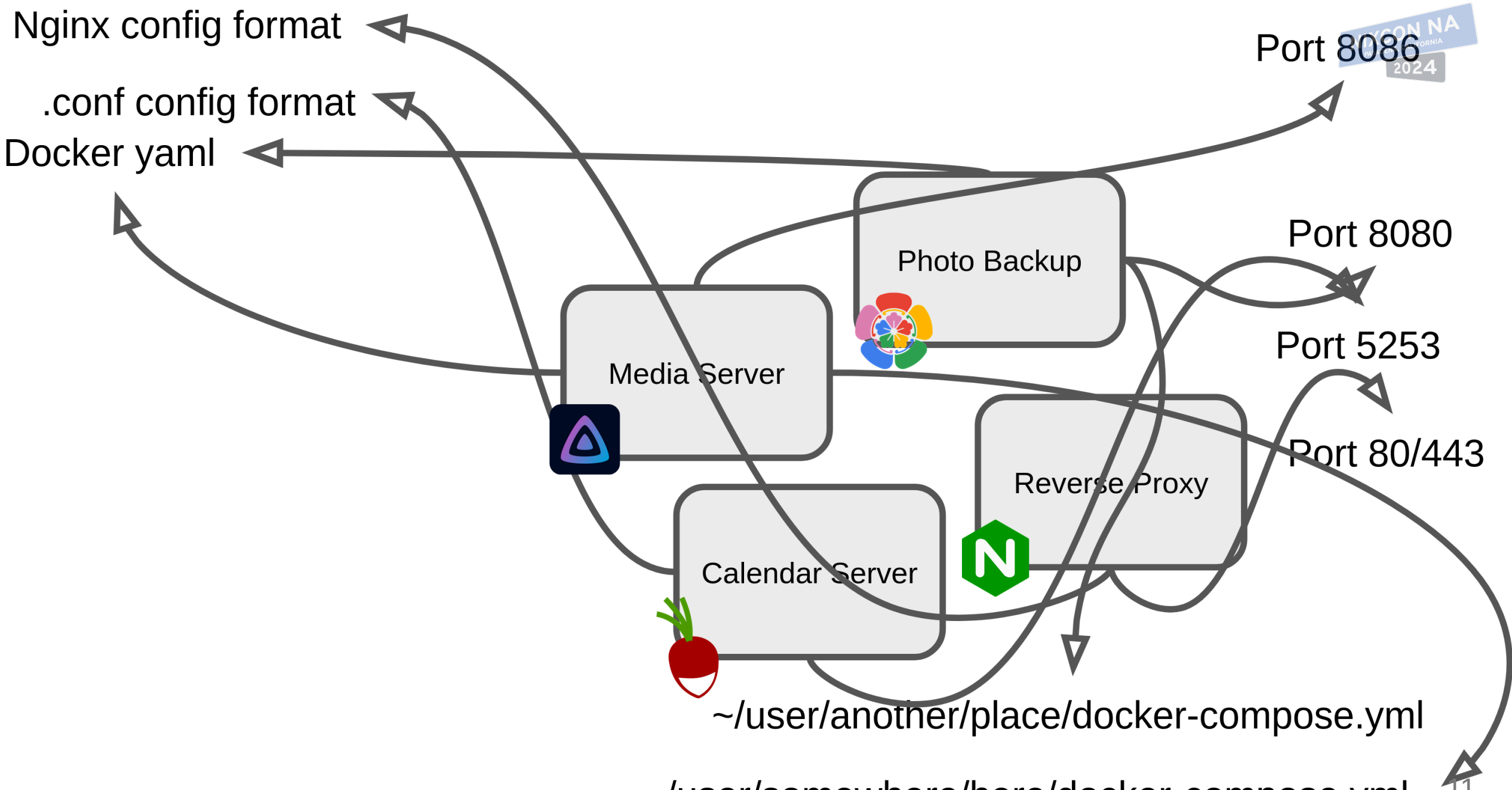

Port 8086
Docker yaml
Photo Backup
Media Server
~/user/somewhere/here/docker-compose.yml

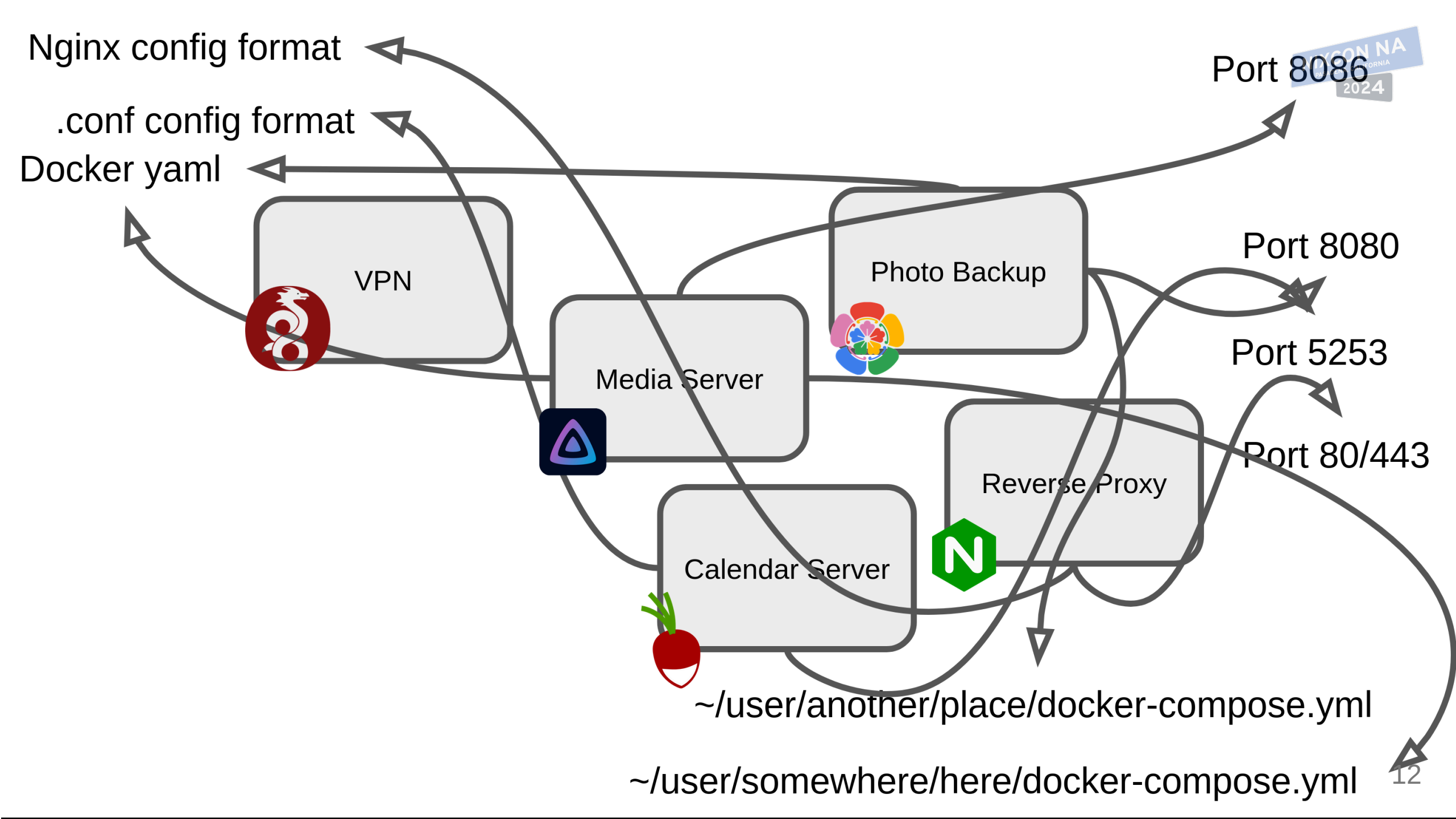

Port 8086
Docker yaml
Port 8080
Photo Backup
Media Server
~/user/another/place/docker-compose.yml
~/user/somewhere/here/docker-compose.yml

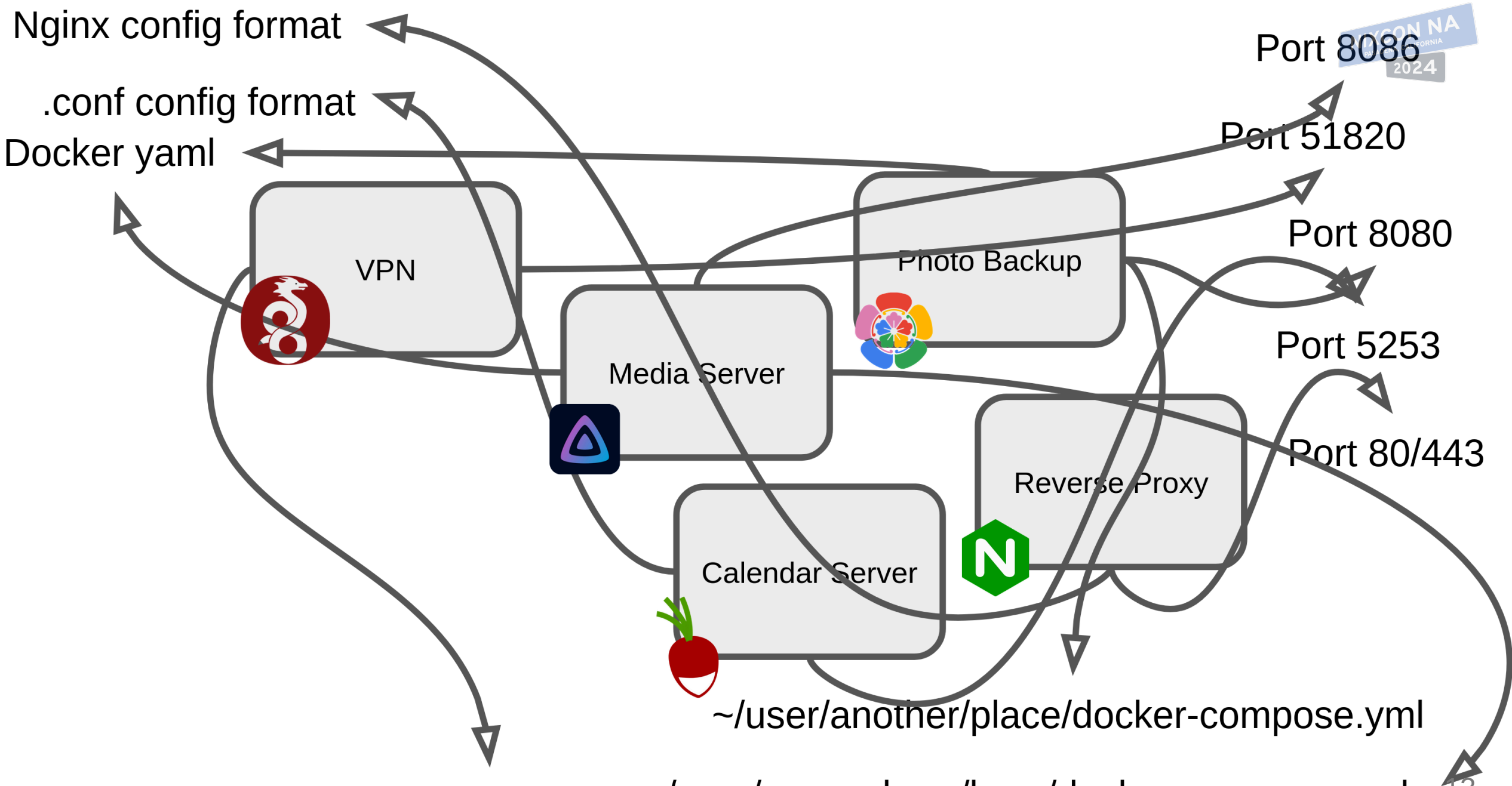

Port 8086
Docker yaml
Port 8080
Photo Backup
Media Server
Calendar Server
~/user/another/place/docker-compose.yml
~/user/somewhere/here/docker-compose.yml

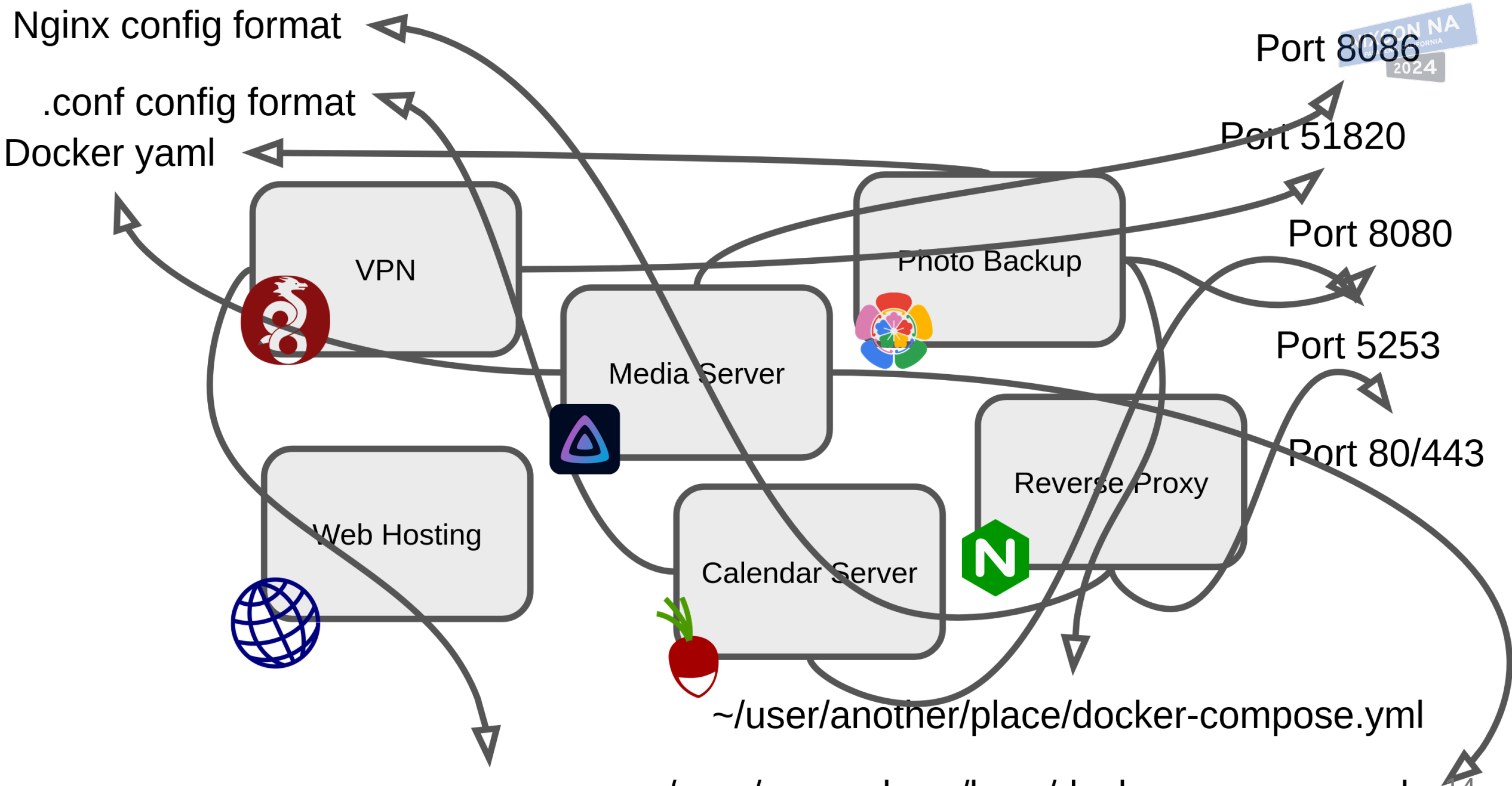

Port 8086
.conf config format
Docker yaml
Photo Backup
Port 8080
Port 5253
Media Server
Calendar Server
~/user/another/place/docker-compose.yml
~/user/somewhere/here/docker-compose.yml

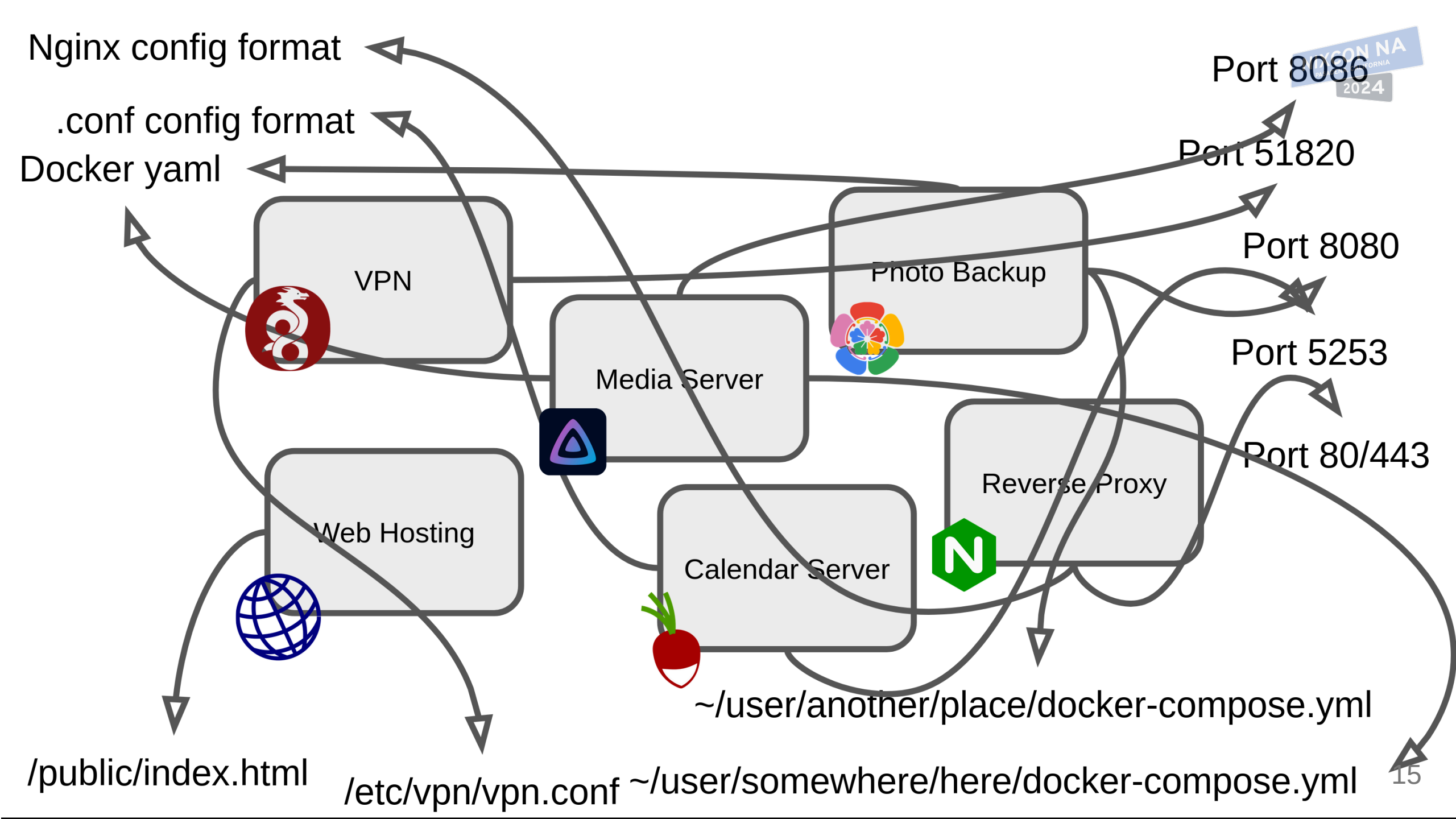

Port 8086
.conf config format
Docker yaml
Photo Backup
Port 8080
Port 5253
Media Server
Reverse Proxy
Calendar Server
~/user/another/place/docker-compose.yml
~/user/somewhere/here/docker-compose.yml

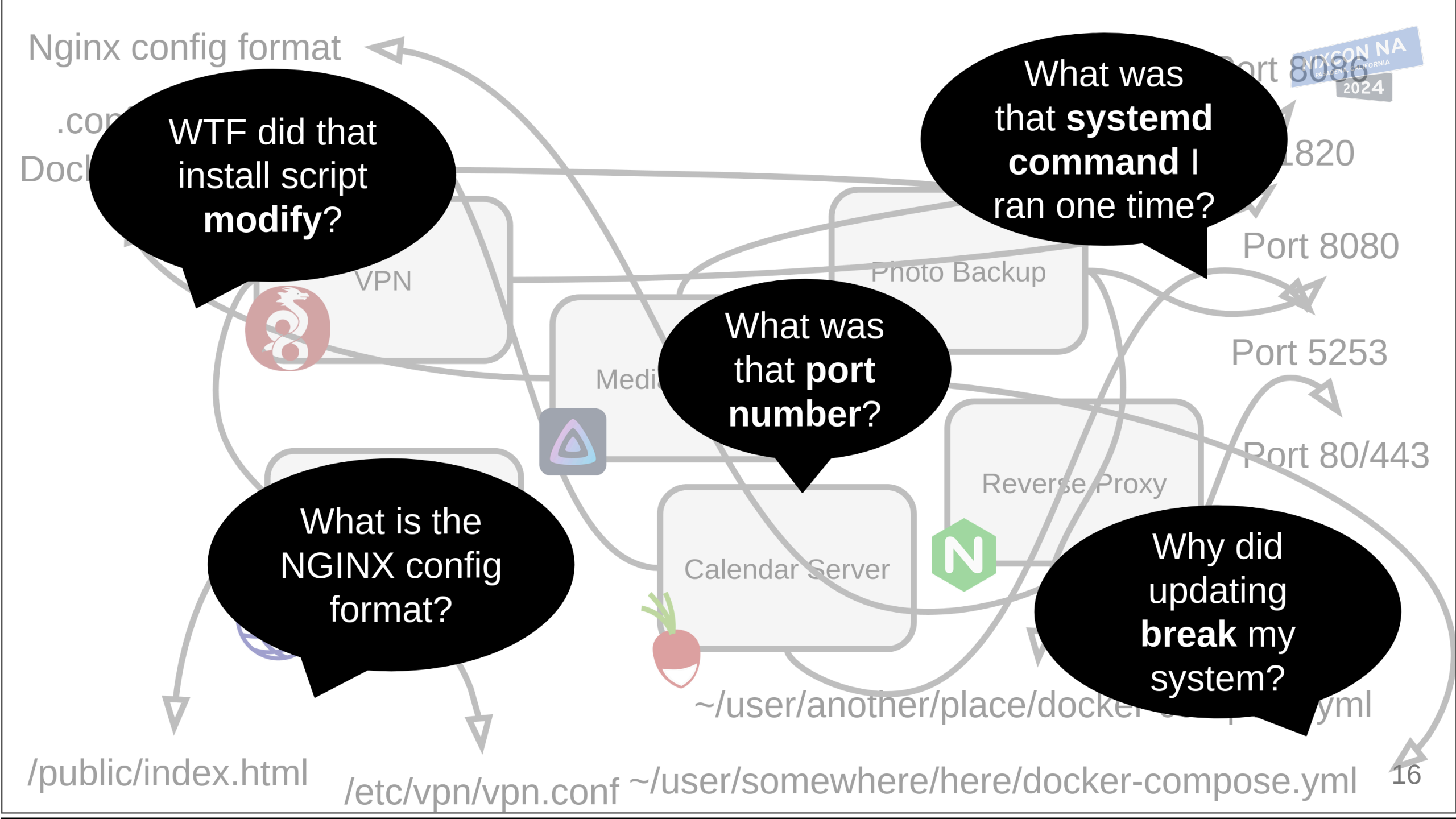

Nginx config format
.conf config format
Docker yaml
Port 8086
Port 8080
Port 5253
Port 80/443
Photo Backup
Media Server
Reverse Proxy
Calendar Server
~/user/another/place/docker-compose.yml
~/user/somewhere/here/docker-compose.yml


Nginx config format
.conf config format
Docker yaml
VPN
Media Server
Photo Backup
Reverse Proxy
Calendar Server
Port 8086
Port 8080
Port 5253
Port 80/443
~/user/another/place/docker-compose.yml
~/user/somewhere/here/docker-compose.yml


Nginx config format
.conf config format
Docker yaml
VPN
Media Server
Photo Backup
Calendar Server
Reverse Proxy
Port 8086
Port 51820
Port 8080
Port 5253
Port 80/443
~/user/another/place/docker-compose.yml
/etc/vpn/vpn.conf
~/user/somewhere/here/docker-compose.yml


Nginx config format
.conf config format
Docker yaml
VPN
Media Server
Photo Backup
Web Hosting
Calendar Server
Reverse Proxy
Port 8086
Port 51820
Port 8080
Port 5253
Port 80/443
~/user/another/place/docker-compose.yml
/etc/vpn/vpn.conf
~/user/somewhere/here/docker-compose.yml


Nginx config format
.conf config format
Docker yaml
VPN
Media Server
Photo Backup
Web Hosting
Calendar Server
Reverse Proxy
Port 8086
Port 51820
Port 8080
Port 5253
Port 80/443
~/user/another/place/docker-compose.yml
/public/index.html
/etc/vpn/vpn.conf
~/user/somewhere/here/docker-compose.yml


Nginx config format
WTF did that install script **modify**?
What was that **systemd command** I ran one time?
Port 8080
VPN
Photo Backup
What was that **port number**?
Port 5253
Port 80/443
Reverse Proxy
What is the NGINX config format?
Calendar Server
Why did updating **break** my system?
~/user/another/place/docker
/public/index.html
/etc/vpn/vpn.conf
~/user/somewhere/here/docker-compose.yml

The case for Nix on the home server

By Anthony Tarbinian & Samir Rashid

Nixcon NA, March 15, 2024

Presentation permalink:
godsped.com/nix-homeserver

Who are we

Samir Rashid is a student at UC San Diego. I'm working towards a future of understandable software.

Anthony Tarbinian is a student at UC San Diego. I'm a fan of open source software and public transit.

Benefits of a Nix home server

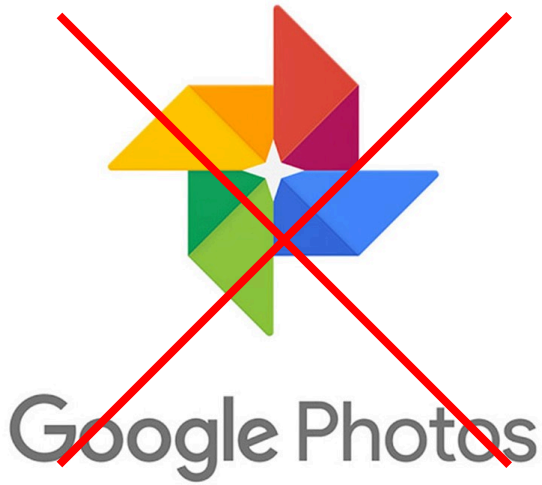
Fearless modifications

System-wide transparency

Maintenance freeing

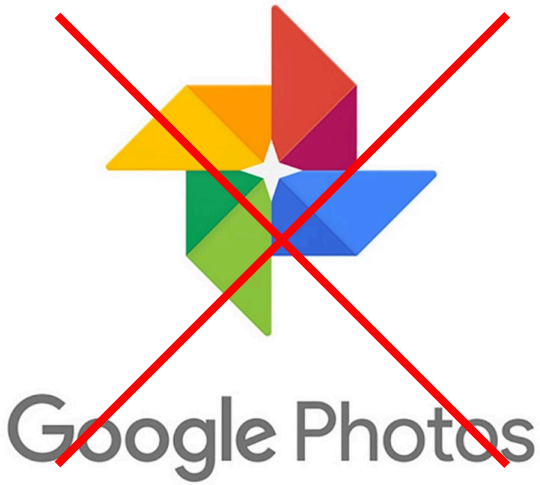
What is a home server?

- Host your own services
  - Photo Backup
  - Media Library
  - Website Hosting
  - Backups

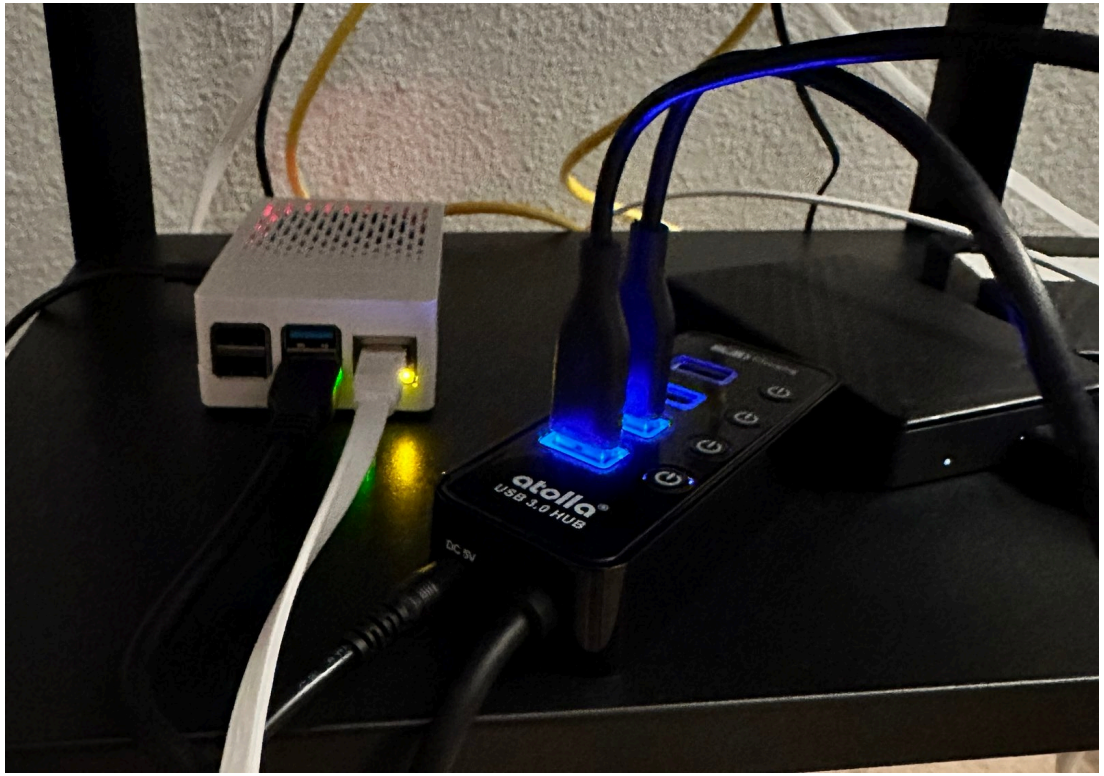


What is a home server?

- Host your own services
  - Photo Backup
  - Media Library
  - Website Hosting
  - Backups



Expensive!?



Why a home server?

- **Sovereignty**: own your own data
  - Does not rely on cloud services
  - The Internet was meant to be distributed
- **Fun**: play with software

Netflix Help Center
https://help.netflix.com › node

Netflix Ad-supported Plans

Learn more about **Netflix ad-supporte**

TV shows at a lower price.

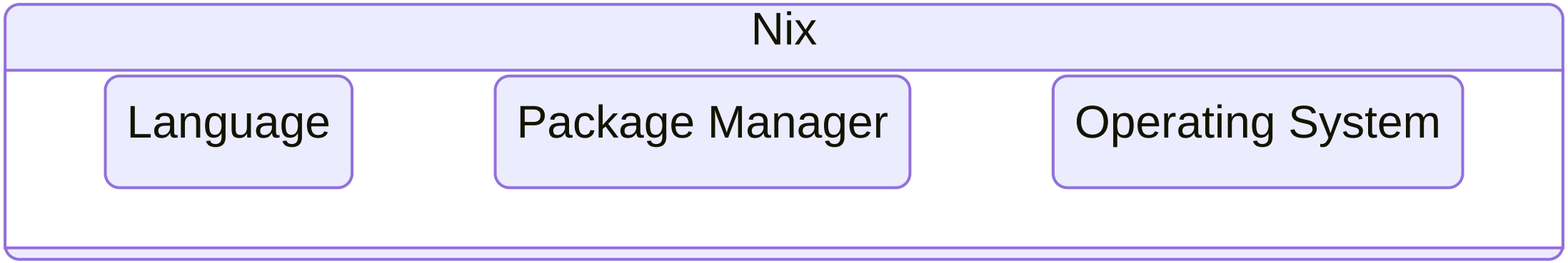
The Verge
https://www.theverge.com › google-stadia-shu

Google is shutting down Stadia

Sep 29, 2022 — Google is shutting down Sta

remain live for players until January 18th,

What is Nix?



Why Nix?

- A way to configure your **entire system**
  - Write configuration in the Nix **language**
  - Build your system from your configuration
- Create declarative, reproducible environments
  - Write once, build anywhere

What can you do in your Nix config?

Installing packages

Just add this to `configuration.nix`

```
environment.systemPackages = with pkgs; [
    vim
    git
];
```

What can you do in your Nix config?

Installing packages

Just add this to `configuration.nix`

```
 environment.systemPackages = with pkgs; [
     vim
     git
+    tmux
];
```

What can you do in your Nix config?

Setup SSH Server

Just add this to `configuration.nix`

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

Installing Home Server Applications

NIXCON NA
PASADENA, CALIFORNIA
2024

Jellyfin

- Popular open source media management software
- Host your own movies and music

Install Jellyfin attempt

NIXCON NA PASADENA, CALIFORNIA 2024

Debian and Ubuntu Official

Install Jellyfin via our APT repository or via manual archives (.deb).

All Debian Versions | All Ubuntu Versions

```
curl https://repo.jellyfin.org/install-debuntu.sh | sudo bash
```

If you do not have `curl` installed, you can use `wget -O-` instead of `curl`.

For more advanced users, the full steps can be found in the docs.

Once installed, Jellyfin will be running as a service. Manage it with `sudo systemctl`

In Nix

```
services.jellyfin.enable = true;
```

What does this do?

```
services.jellyfin.enable = true;
```

- Download and install Jellyfin
- Create a systemd service to startup Jellyfin
- Start systemd service on boot

Samba

- Networked file sharing

Setting up Firewall

Without Nix

Need to manually run a `ufw` command

```
sudo ufw allow 139/tcp
```

With Nix

```
services.samba = {
    enable = true;
    openFirewall = true;
}
```


192.168.1.50:7878
jellyfin.local

Nginx

- Web server & reverse proxy

Where is Nginx's config file located?

Problem: too many config locations

```
/etc/
/opt/
/usr/local/etc/
~/.config/
~/.dotfile
~/.ssh/config
~/.local/share/
~/random-folder/docker-compose.yml
...
```

What format is `nginx.conf`?

- Why does Nginx use a bespoke configuration language?

nginx.com

Problem: too many config formats

There's dozens of configuration formats:

- Nginx, SSH, JSON, YAML, TOML, CONF, INI, etc.

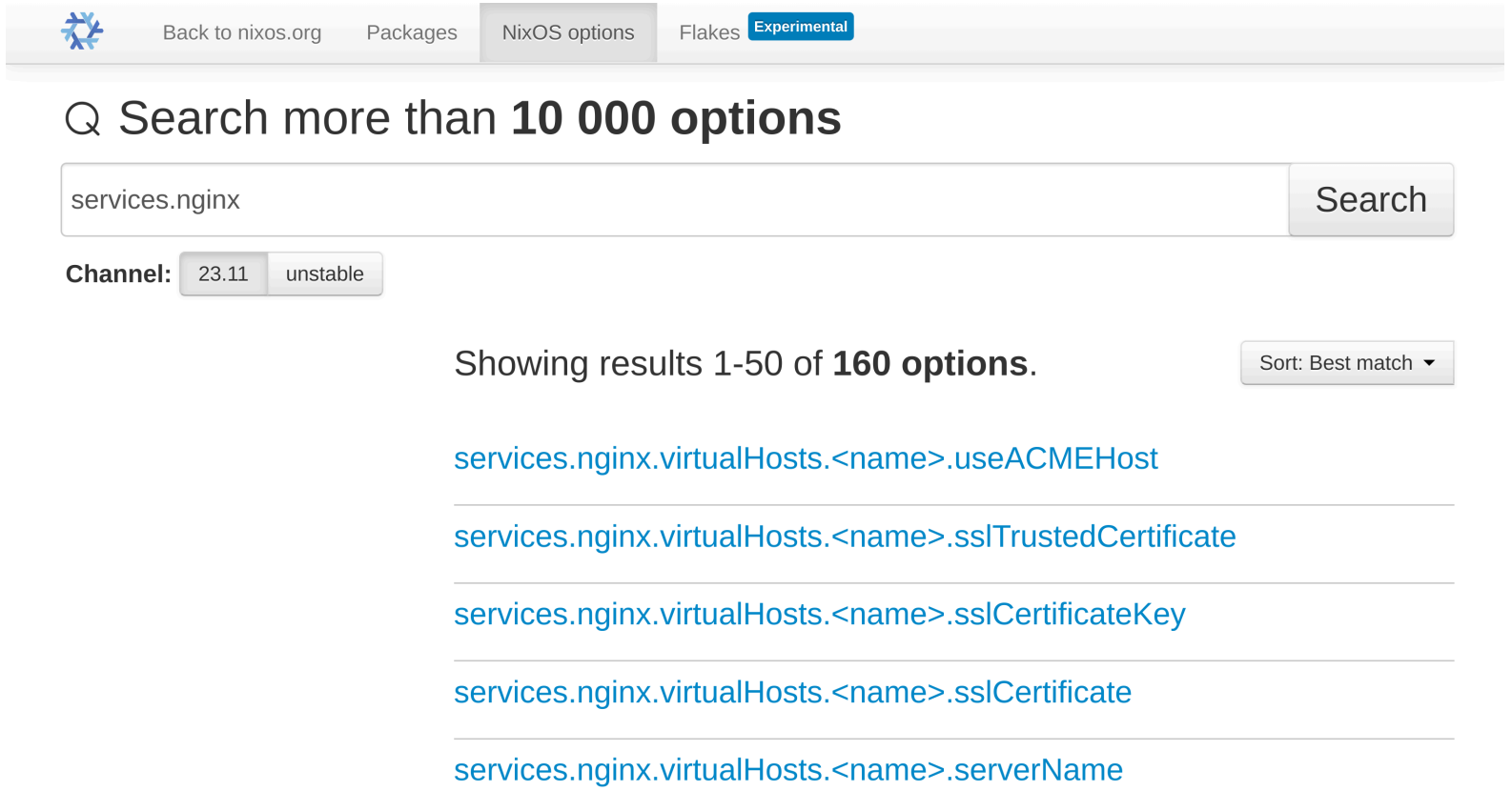
Nix simplifies this by having all your services share the same config language

Configure Nginx in Nix

```
services.nginx = {
    enable = true;
    virtualHosts."jellyfin.local" = {
        locations."/" = {
            proxyPass = "http://127.0.0.1:8096/";
        };
    };
};
```

- We don't have to learn how to install Nginx, learn its bespoke configuration language.

Nginx Options



search.nixos.org/options

Nix: Less Cognitive Overload

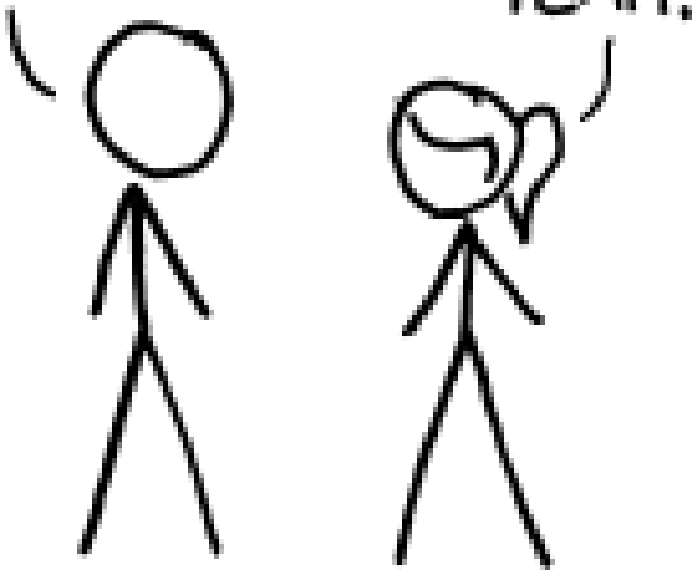
- All ports opened in **one** place
- All systemd services defined in **one** place
- **One** config location
- **One** config format

HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC.)

NIXCON NA
PASADENA, CALIFORNIA
2024

SITUATION:
THERE ARE
14 COMPETING
STANDARDS.

14?! RIDICULOUS!
WE NEED TO DEVELOP
ONE UNIVERSAL STANDARD
THAT COVERS EVERYONE'S
USE CASES.



SOON:

SITUATION:
THERE ARE
15 COMPETING
STANDARDS.

XKCD

Reusable

Reuse building blocks from online

- Someone has made a NixOS config for whatever you want
- Just as easy as copying a docker compose

Reusable: Mail Server

- Only a few lines!

nixos-mailserver.readthedocs.io

Setup the server

The following describes a server setup that is fairly complete. Even though there are more possible options (see the `default.nix` file), these should be the most common ones.

```
{ config, pkgs, ... }:
{
  imports = [
    (builtins.fetchTarball {
      # Pick a commit from the branch you are interested in
      url = "https://gitlab.com/simple-nixos-mailserver/nixos-mailserver/-/archive/A-COMMIT-ID/
      # And set its hash
      sha256 = "0000000000000000000000000000000000000000000000000000";
    })
  ];

  mailserver = {
    enable = true;
    fqdn = "mail.example.com";
    domains = [ "example.com" ];

    # A list of all login accounts. To create the password hashes, use
    # nix-shell -p mkpasswd --run 'mkpasswd -sm bcrypt'
    loginAccounts = {
        "user1@example.com" = {
            hashedPasswordFile = "/a/file/containing/a/hashed/password";
            aliases = ["postmaster@example.com"];
        };
        "user2@example.com" = { ... };
    };

    # Use Let's Encrypt certificates. Note that this needs to set up a stripped
    # down nginx and opens port 80.
    certificateScheme = "acme-nginx";
  };
}
```

After a `nixos-rebuild switch` your server should be running all mail components.

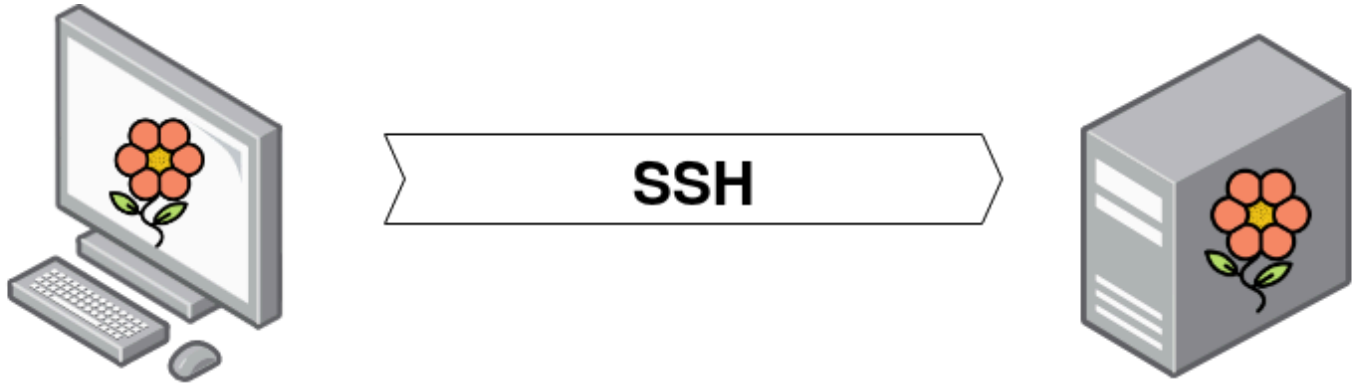
Reusable: Wireguard VPN


```
networking.wireguard.interfaces = {
wg0 = {
    ips = [ "10.100.0.1/24" ];
    listenPort = 51820;
    ...
    privateKeyFile = "path to private key file";
    peers = [
        {
            publicKey = "{client public key}";
            allowedIPs = [ "10.100.0.2/32" ];
        }
        {
            publicKey = "{john doe's public key}";
            allowedIPs = [ "10.100.0.3/32" ];
        }
    ];
};
};
```

nixos.wiki/wiki/WireGuard

Reusable across machines

- Configure software once across your machines
  - Even on different architectures



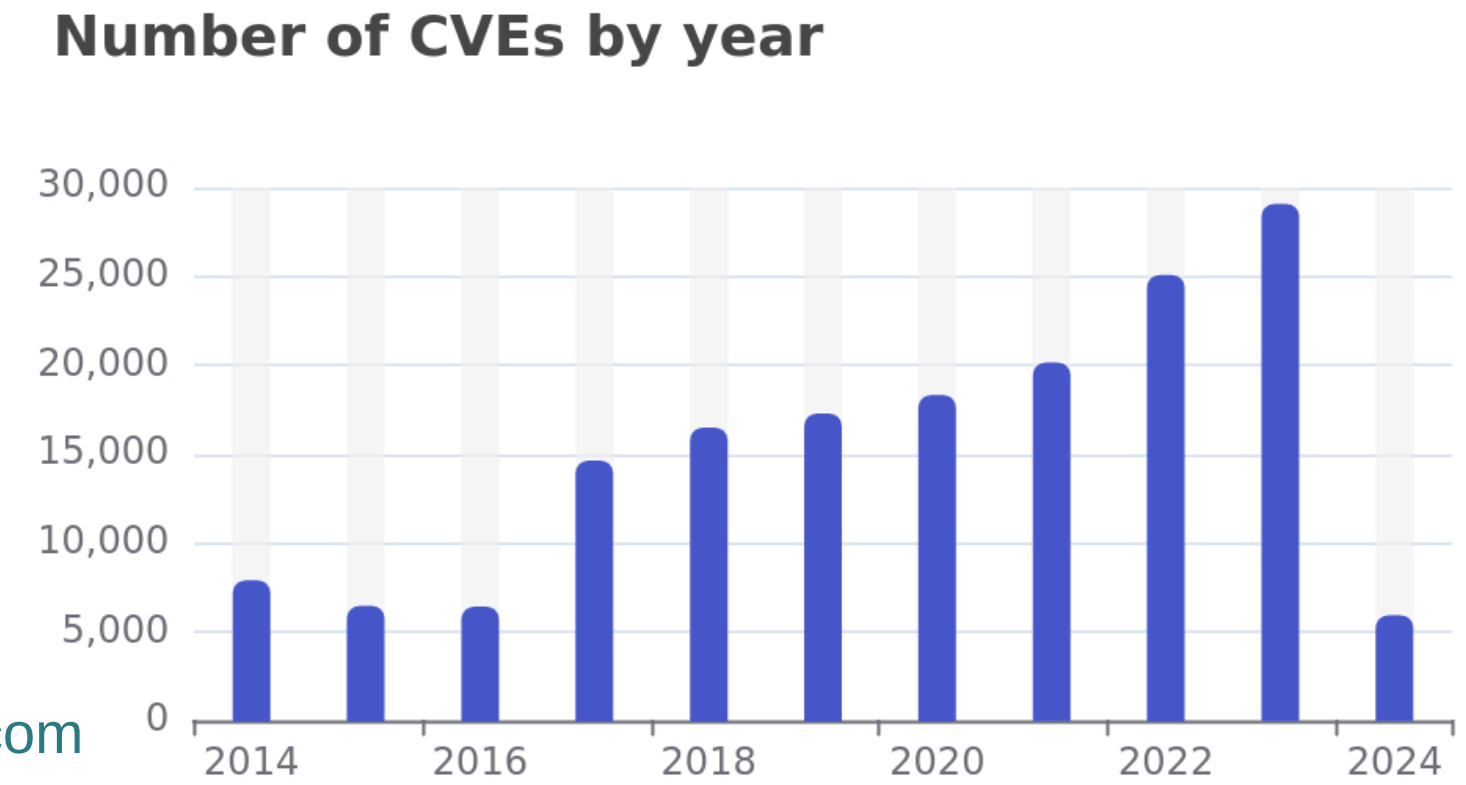

```
10  const profileJsonPath = `${workspaceDirec
11  fs.readFile(profileJsonPath, "utf8", (err
12    if (err) {
13      vscode.window.showErrorMessage(
```

```
-  fs.readFile(
-    `${workspaceDirectory}/profile.json`,
-    "utf8",
-    (err: Error | null, data: string) => {
-      if (err) {
-        vscode.window.showErrorMessage(
+  const profileJsonPath = `${workspaceDirectory}/profile.js
+  fs.readFile(profileJsonPath, "utf8", (err: Error | null,
+    if (err) {
+      vscode.window.showErrorMessage(
```

Upgrading Server Software

Upgrade Early, Upgrade Often

- Important to keep home server up-to-date
  - New application features
  - Security



cvedetails.com

Upgrading

APT (Debian/Ubuntu)

```
sudo apt-get upgrade -y
```

What could go wrong while upgrading?

- Dependency conflict
- Unwanted changes
- Power gets cut mid-upgrade

... system gets left in broken state

Does this really happen? Yes ...

ask Ubuntu

Home

Questions

Unable to correct problems, you have held broken packages

Asked 11 years, 3 months ago Modified 1 month ago Viewed 1.5m times

Ubuntu upgrade to 20.04 broken by sudden restart of system

Asked 3 years, 8 months ago Modified 3 years, 8 months ago Viewed 1k times

Unable to correct problems, you have held broken packages

Asked 11 years, 3 months ago Modified 1 month ago Viewed 1.5m times

Fearless upgrades

APT (Debian/Ubuntu)

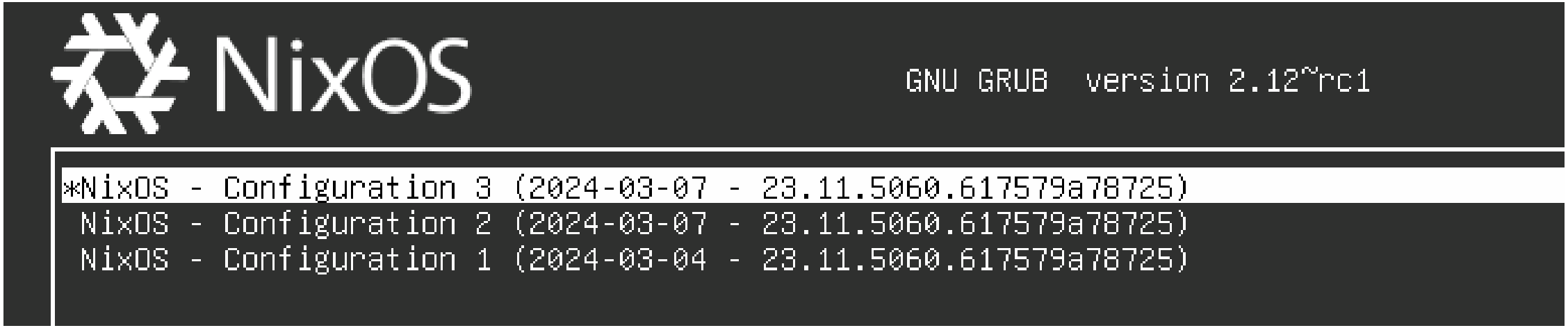
- `sudo apt-get upgrade -y`
- How to undo: impossible with APT *

Nix (NixOS)

- `sudo nixos-rebuild switch`
- How to undo: `sudo nixos-rebuild --rollback`

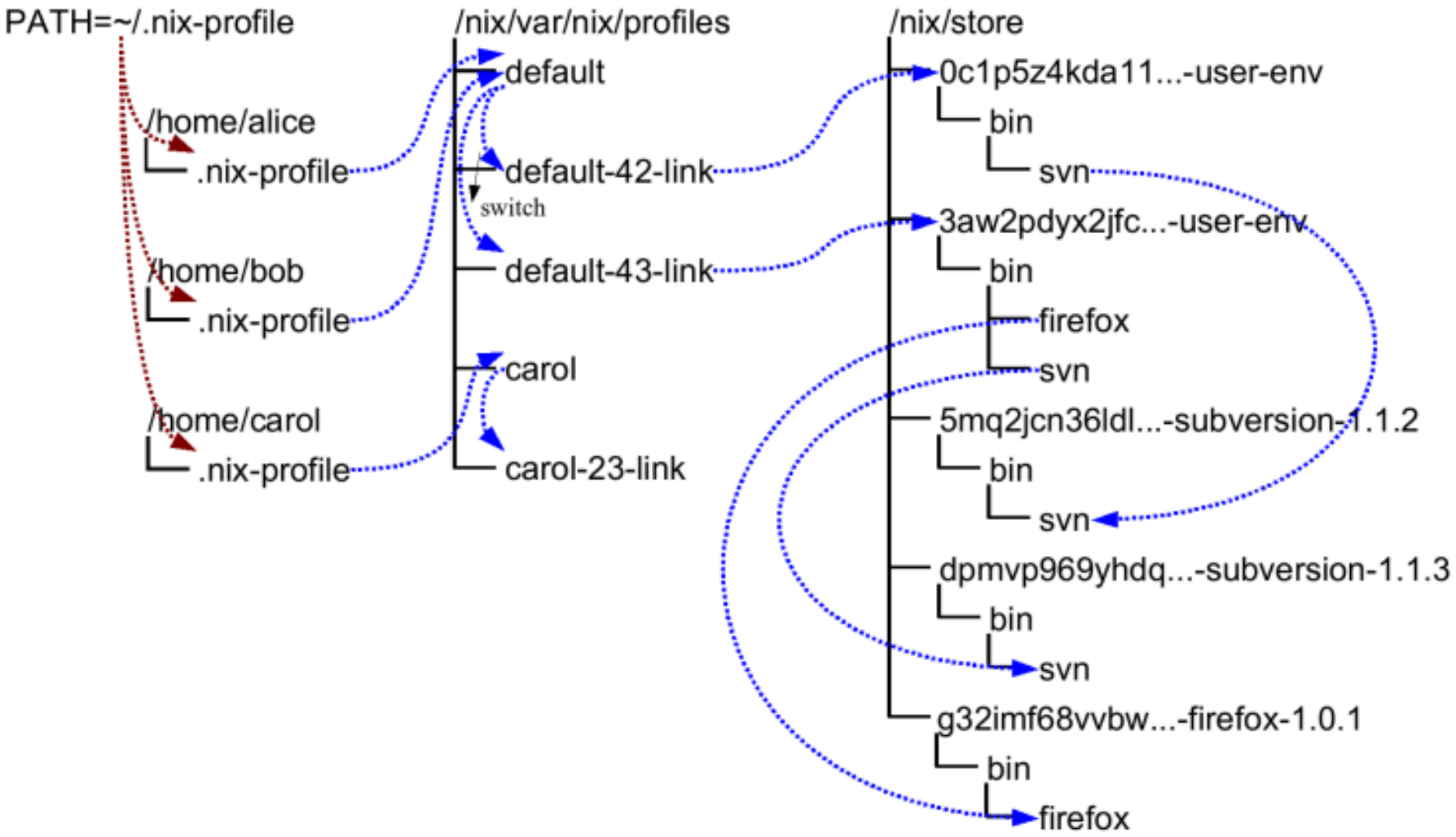
Rollbacks

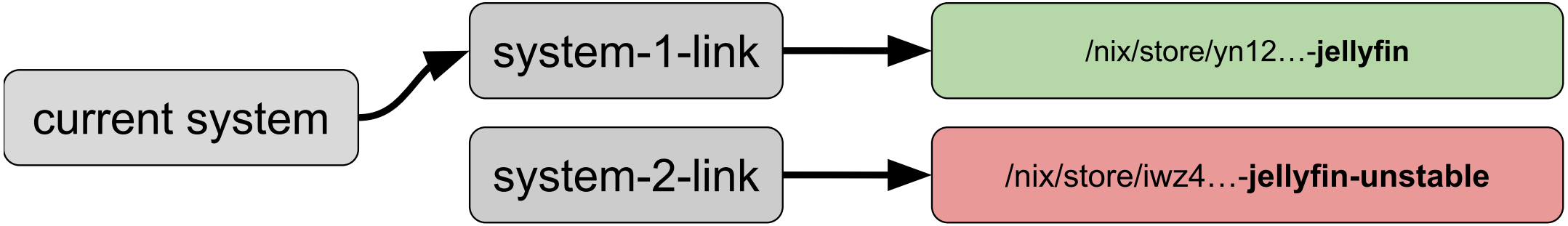
- NixOS maintains a version of your system each time you make a change
- Rollback to a previous version if something breaks
- Can fearlessly upgrade

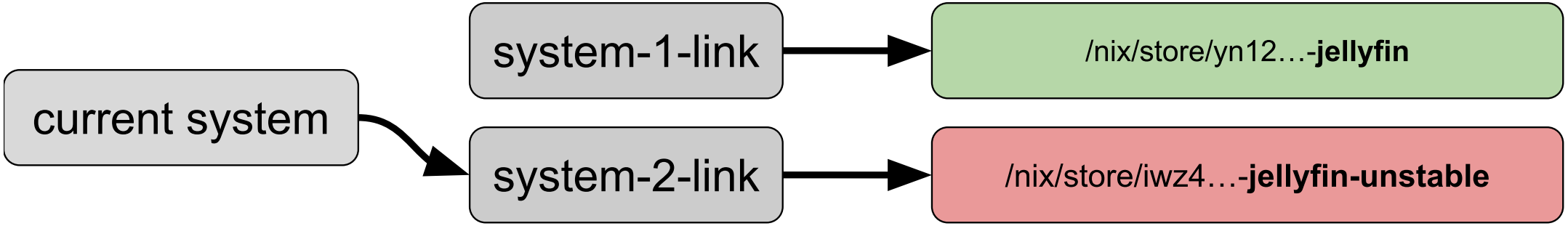


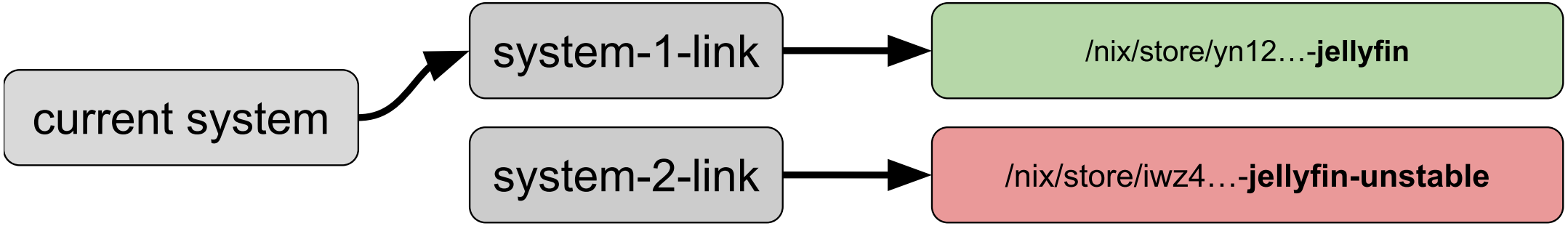
Like git for your entire system

What's the secret sauce?



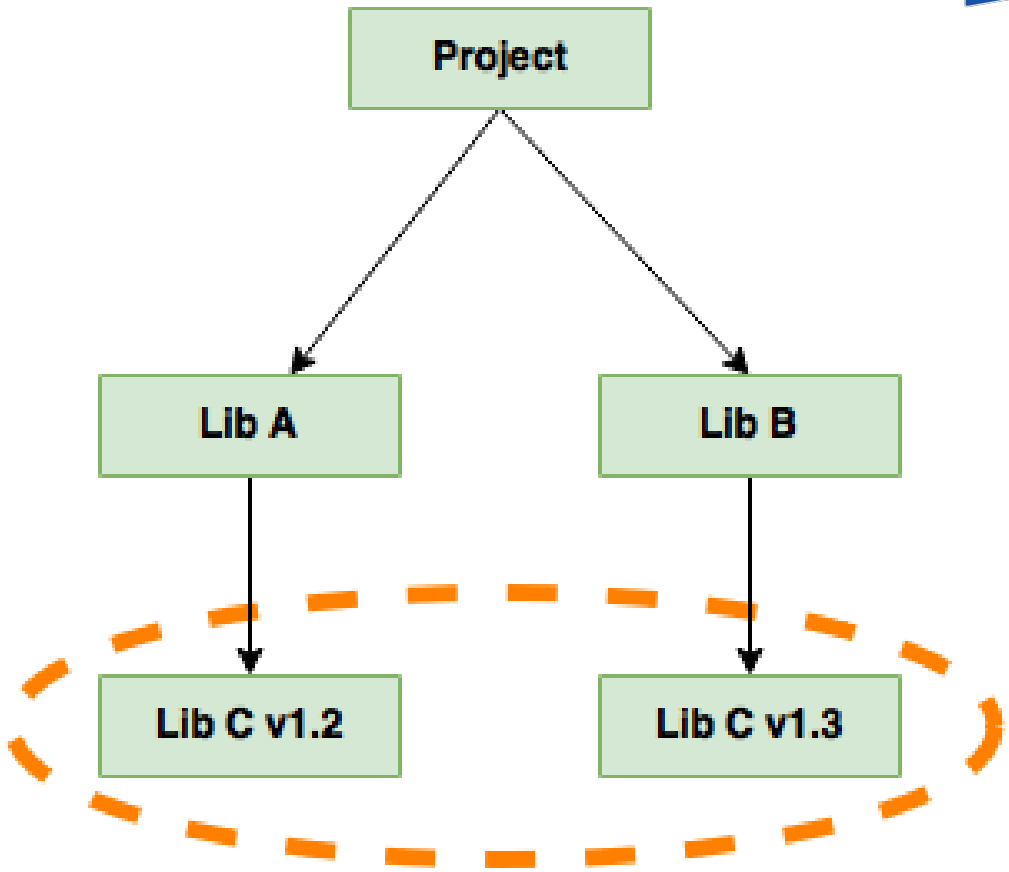

current system
system-1-link
system-2-link
/nix/store/yn12…-jellyfin
/nix/store/iwz4…-jellyfin-unstable


current system
system-1-link
system-2-link
/nix/store/yn12…-jellyfin
/nix/store/iwz4…-jellyfin-unstable


current system
system-1-link
system-2-link
/nix/store/yn12…-jellyfin
/nix/store/iwz4…-jellyfin-unstable

Version Pinning

- Install **both** bleeding edge, stable, and legacy software
- No conflicting dependency versions



vsvankhede.github.io

```
services.jellyfin.package = unstable.jellyfin;
```

Docker vs Nix

Docker

✅ Ubiquitous adoption

❌ Leaves `docker-compose.yml` files on your system

❌ Versioning is opt-in

❌ Not integrated with host system

Nix

❌ Requires effort to package

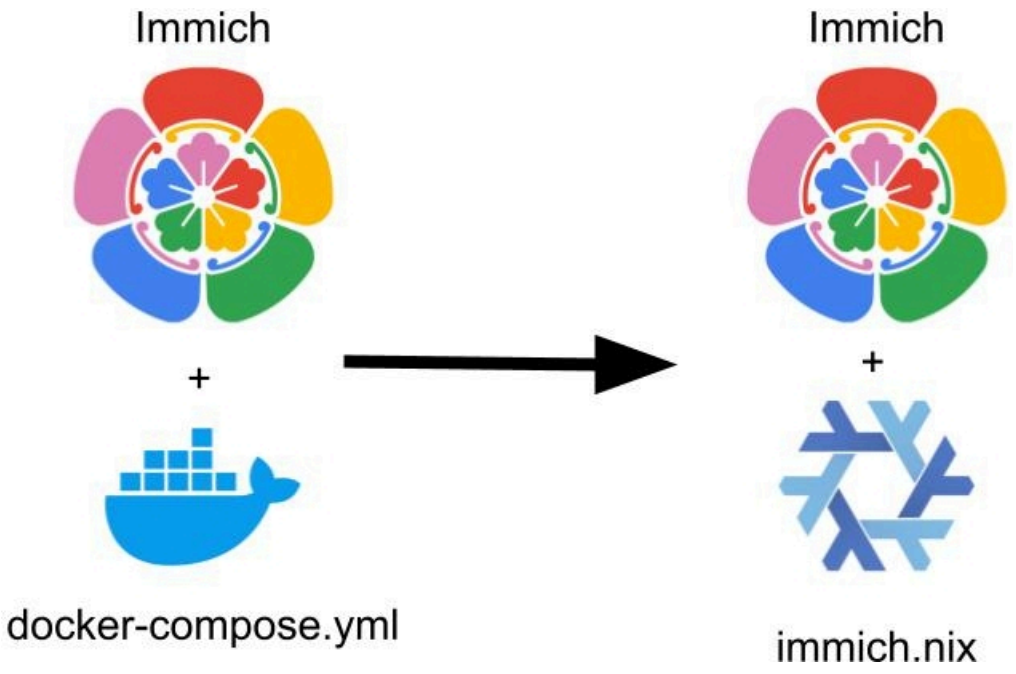
✅ Nice to have all your services defined in one place

✅ Easy to rollback

✅ Manages system state

Docker vs Nix: Why choose one?

- Can startup Docker containers on boot from within Nix
- Can pin images to Docker image hash
- Can use existing `docker-compose.yml` with `compose2nix` tool
  - github.com/aksiksi/compose2nix



Ansible vs Nix

Ansible

- ❌ No undo if OS update breaks
- ❌ Ansible is notoriously slow
- ❌ Interrupting Ansible can break system
- ❌ Easy to drift away defined Ansible state

Nix

- ✅ Rollbacks
- ✅ Nix is highly cached
- ✅ Nix builds are atomic
- ✅ Nix encourages reproducibility

Drawbacks

- Inconsistent configuration options
- Sparse documentation
- Confusing error messages
- Fragmented standards
- Different line of thinking
  - "Normal" things may not work the way you expect and are used to

Is Nix better?

Nix is (Maintenance) Freeing

- **You don't want managing your home server to be your full time job**
- `configuration.nix` is self documenting and gives you an easy way to see an overview of the system
- You don't have to worry about breaking things when you make changes.

Benefits of a Nix home server

- Fearless modifications: **rollbacks make upgrades risk free**
- System-wide transparency: **configuration.nix is just a text file with everything specified about your system**
- Maintenance freeing: **reusable components, don't have to memorize commands or config formats**

Try it at home!

See the following GitHub repo for a simple example NixOS configuration:

`github.com/atar13/nixcon24-home-server`

Contact us: nixcon@godsped.com