Spectra Inter-Web / LAN Intranet Ecosystem (Fedora 42 + Ubuntu 24.04 LTS)

> **Objective** Unify the *spectra-playground-server*, *spectra-backend*, *spectra-frontend*, *spectra-storage* and the **Afrhône** data cluster into a cross-cloud, cross-OS game-dev platform that serves Xbox Cloud builds while doubling as a creative-coding lab (WebGPU/WebGL/shaders) and data-viz sandbox.

---

0 Glossary

Name	Role
spectra-playground-server	Node-JS / Python FastAPI gateway for Playground UI & Neural-Lab WebSocket streams
spectra-backend	PostgreSQL 15 + Redis + Task Workers (Celery / Huey)
spectra-storage	MinIO S3-compatible blob store, off-site replicate to GCS bucket
Afrhône	GPU/FPGA edge nodes (Fedora 42 Server) handling ML inference + SDR capture
Lab Neural Map	Python module that auto-discovers federated workers via WireGuard overlay and registers them in etcd

---

1 Physical & Cloud Layout

```
       +-----------------+            Google Cloud (eu-w4)
+-----------------+
       |  Fedora VRB    |-----------------------------------------|  GCP VM
(hub)   |
 LAN <-> eno8303 10.10 |  WireGuard     10.44.1.0/24               |
10.44.1.2       |
 WAN <-> eno8403 /27   |-----------------------------------------| Azure VM
(hub)  |
       |  vbr1s0 bridge |  10.44.0.0/16 overlay        |-----------|
10.44.2.2       |
       +-------+--------+
+-----------------+
               |
               | libvirt: dev-net 10.44.10.0/24 (staging/prod nets similar)
               |
```

```
          +-----+----------+
          | win-gdk  VM    |  (Xbox GDK, UE5, Visual Studio 2025)
          +----------------+
```

*WireGuard keeps the **Fedora 42 VRB hub**, **Ubuntu 24.04 GCP node**, and **Ubuntu 24.04 Azure node** in one flat /16 so Nomad jobs & CI runners can float anywhere.*

---

2 Baseline VRB (see *Vrb-apply* script)

- WAN NIC `/128`, bridge `/64`, dnsmasq zones loaded from your 01-/02-/03-/20 files.
- `vrb-apply.sh` now also pulls WireGuard & libvirt nets (Section 3 & 4 below).

---

3 Cross-cloud WireGuard overlay (`configs/wireguard/wg0.conf`)

Already shown in **VRB-Cloud-Gaming** doc → install on each node and enable `wg-quick@wg0`.

---

4 Dev / Staging / Prod libvirt nets (`configs/libvirt-networks/*.xml`)

- `dev-net` → 10.44.10.0/24 (agents + win-gdk)
- `staging-net` → 10.44.20.0/24 (autoscaled game servers)
- `prod-net` → 10.44.30.0/24 (public matchmaker ELB backends)

---

5 Continuous Integration / Deployment

5·1 GitHub → Azure DevOps dual-runner

```
name: spectra-pipe
on: [push]
jobs:
  build-container:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - run: podman build -t ghcr.io/org/spectra-server:$GITHUB_SHA .
      - run: podman push ghcr.io/org/spectra-server:$GITHUB_SHA
  xbox-build:
    runs-on: [self-hosted, win-gdk]
    steps:
```

```
      - uses: actions/checkout@v4
      - run: msbuild spectra-xbox.sln /p:Configuration=Shipping /
p:Platform="GameCoreXboxSeries"
      - run: AzureBlobUpload.exe build\*.pkg
```

The **win-gdk** runner registers via the *Azure Pipelines* agent so the same script can be triggered from Azure-DevOps Release stage.

5·2 GCP ephemeral runner for GPU shaders

```
gcloud beta compute instances create-with-container shader-runner-$SHORT_SHA \
  --zone=us-west4-b --container-image=ghcr.io/org/spectra-shader:$SHA \
  --container-restart-policy=never --preemptible
```

---

6 Runtime stack (Kubernetes optional, Nomad simpler)

Tier	Runtime	Placement
Gateway	`spectra-playground-server` (Podman)	on-prem VRB hub & GCP zone
API	FastAPI gunicorn containers	on-prem + Azure
Storage	MinIO (stand-alone) + GCS R/O mirror	on-prem + GCP
Redis / Celery	Docker Swarm or Nomad group	anywhere (state ≈ none)
Data-viz	JupyterLab + Panel served via Spectra UI	dev-net only
Creative-coding	Vite + WebGPU sample bundles	dev-net only

---

7 Frontend wiring

- **PlaygroundModal component** in *spectra-frontend* calls `POST /api/v1/sandbox` to spin a disposable container with GPU on `dev-net`; returns WS URL for shader hot-reloader.
- **Lab Neural Map** fetches `/api/v1/nodes` (served by *spectra-backend* from etcd) and draws edges via d3-force.
- All WebGPU / WebGL demos compiled with Vite + `@webgpu/types` → shipped as static assets behind `spectra-playground-server`.

---

8 Bash deployment helpers (`scripts/`)

8·1 `vrb-apply.sh` – base + overlay + nets *(already live)*

8·2 `deploy-spectra.sh`

```
#!/usr/bin/env bash
podman-compose -f infra/docker-compose.yml pull
podman-compose -f infra/docker-compose.yml up -d
```

8·3 `create-win-gdk.sh` – virt-install wrapper *(see Section 4)*

---

9 Benchmark Workflow

1. **k6** hits `https://play.example.net/api/v1/ping` 500 rps.
2. **Flames** plugin on win-gdk records GPU frame time while streaming.
3. Results pushed to `spectra-storage/bench/$ISO8601.json`, Grafana dashboards auto-refresh.

---

10 Xbox-specific Lab hooks (recap)

See Section 5 of *VRB-Cloud-Gaming* doc in canvas.

✅Outcome

A reproducible, script-driven ecosystem where:

- Fedora 42 VRB hub + WireGuard overlay bind on-prem, Azure and GCP.
- Windows LTSC VM handles Xbox builds; CI runners spawn where compute is cheapest.
- spectra-* micro-services route internally via 10.44.0.0/16 but expose selected endpoints through Nginx Ingress on WAN /27.
- Creative coders open *PlaygroundModal* → get live WebGPU canvas; data engineers open Jupyter on dev-net; gamers hit staging-net for previews.

Everything lives in Git, deploys with `./scripts/vrb-apply.sh && ./scripts