

PROYECTO INTEGRADO

“El futuro del trabajo remoto con IsardVDI: Análisis detallado”



Autor: Mario García Salado

Tutor: Fernando

I.E.S. Francisco Romero Vargas (Jerez de la Frontera)

Administración de Sistemas Informáticos en Red
--

Curso: 2022/2023

Tabla de contenido

Contenido

Introducción.....	2
Finalidad.....	2-3
Objetivos.....	3
Medios necesarios.....	3-4
Realización del Proyecto.....	5-28
Problemas encontrados.....	28-29
Modificaciones sobre el proyecto planteado inicialmente.....	29
Posibles mejoras al proyecto.....	29
Bibliografía.....	29

Introducción.

En el mundo empresarial actual, la rapidez y la eficiencia son clave para el éxito. Sin embargo, el proceso de preparar dispositivos portátiles o sobremesa para los empleados puede ser costoso y llevar mucho tiempo. Además, si los trabajadores están en remoto y necesitan cambiar su equipo por cualquier motivo, pueden pasar semanas hasta que reciban uno de sustitución. Pero, ¿y si hubiera una solución que agilizara este proceso y eliminara el problema de raíz?

Es aquí donde entra en juego IsardVDI, una plataforma que permite a los empleados acceder a un escritorio virtual desde cualquier dispositivo electrónico simplemente con una conexión a internet, eliminando la necesidad de dispositivos específicos y acelerando los procesos de incorporación al puesto de trabajo. En esta introducción, exploraremos cómo IsardVDI puede transformar el mundo empresarial y mejorar la eficiencia y productividad de las empresas.

Finalidad.

¿Qué podemos esperar de IsardVDI? Son varias las ventajas por las que una empresa querría implementar el uso de esta plataforma en su organización y estas se pueden resumir en 5 puntos:

1. **Flexibilidad y movilidad:** IsardVDI permite a los empleados acceder a su escritorio virtual desde cualquier dispositivo electrónico. Para ello solo será necesario una buena conexión a internet. Esto aporta una gran flexibilidad y movilidad. Los empleados podrán trabajar desde cualquier lugar, lo que aumenta la productividad y mejora la calidad de vida laboral.
2. **Reducción de costos:** Al utilizar IsardVDI, las empresas pueden reducir los costos de hardware y software, ya que los empleados pueden acceder al escritorio virtual desde cualquier dispositivo electrónico de bajo costo.
(Ej: acceder desde el teléfono móvil en lugar de un pc de altas especificaciones)
3. **Seguridad:** IsardVDI ofrece un alto nivel de seguridad, ya que los datos se almacenan en servidores seguros y los empleados acceden a ellos mediante conexiones encriptadas.
4. **Facilidad de administración:** La administración de escritorios virtuales a través de IsardVDI es sencilla y centralizada, lo que permite a las empresas administrar fácilmente todos los escritorios virtuales de sus empleados desde una sola ubicación.

5. **Escalabilidad:** IsardVDI es escalable y se adapta a las necesidades de crecimiento de las empresas. Los empleados adicionales pueden incorporarse fácilmente a la plataforma sin necesidad de invertir en hardware y software adicionales.

En resumen, IsardVDI proporciona una solución rentable, segura y flexible para la gestión de escritorios virtuales en las empresas, lo que aumenta la eficiencia y la productividad de la misma.

Objetivos.

Con este proyecto conseguiremos desplegar de forma remota y virtual los puestos de trabajo o escritorios de los empleados. Estos escritorios se organizarán por departamentos, contando cada uno con un sistema operativo diferente según las necesidades de su puesto. Todo esto lo lograremos mediante IsardVDI que estará levantado gracias a Docker y Docker Compose.

En Docker Compose podremos configurar todos los parámetros respecto a la infraestructura del servidor y nuestra plataforma IsardVDI. Indicaremos a qué IP deben acceder los diferentes usuarios. Estos iniciarán sesión con sus credenciales individuales y tendrán acceso únicamente al escritorio virtual que les corresponda según su departamento o grupo.

Ya que este proyecto simula la falta o avería del equipo de empresa, las pruebas serán llevadas a cabo en un PC de sobremesa, un portátil y un teléfono móvil.

Medios necesarios.

IsardVDI es una plataforma de virtualización de escritorio remoto de código abierto basada en KVM (Kernel-based Virtual Machine) y QEMU (Quick Emulator). A continuación, se presentan los requisitos mínimos y recomendados para ejecutar IsardVDI:

Requisitos mínimos:

- CPU de 64 bits con soporte de virtualización (Intel VT o AMD-V)
- 4 GB de RAM
- 30 GB de espacio libre en disco duro
- Conexión de red (Ethernet o Wi-Fi)

Requisitos recomendados:

- CPU de múltiples núcleos con soporte de virtualización
- 8 GB o más de RAM
- 100 GB o más de espacio libre en disco duro
- Conexión de red de alta velocidad (Ethernet)

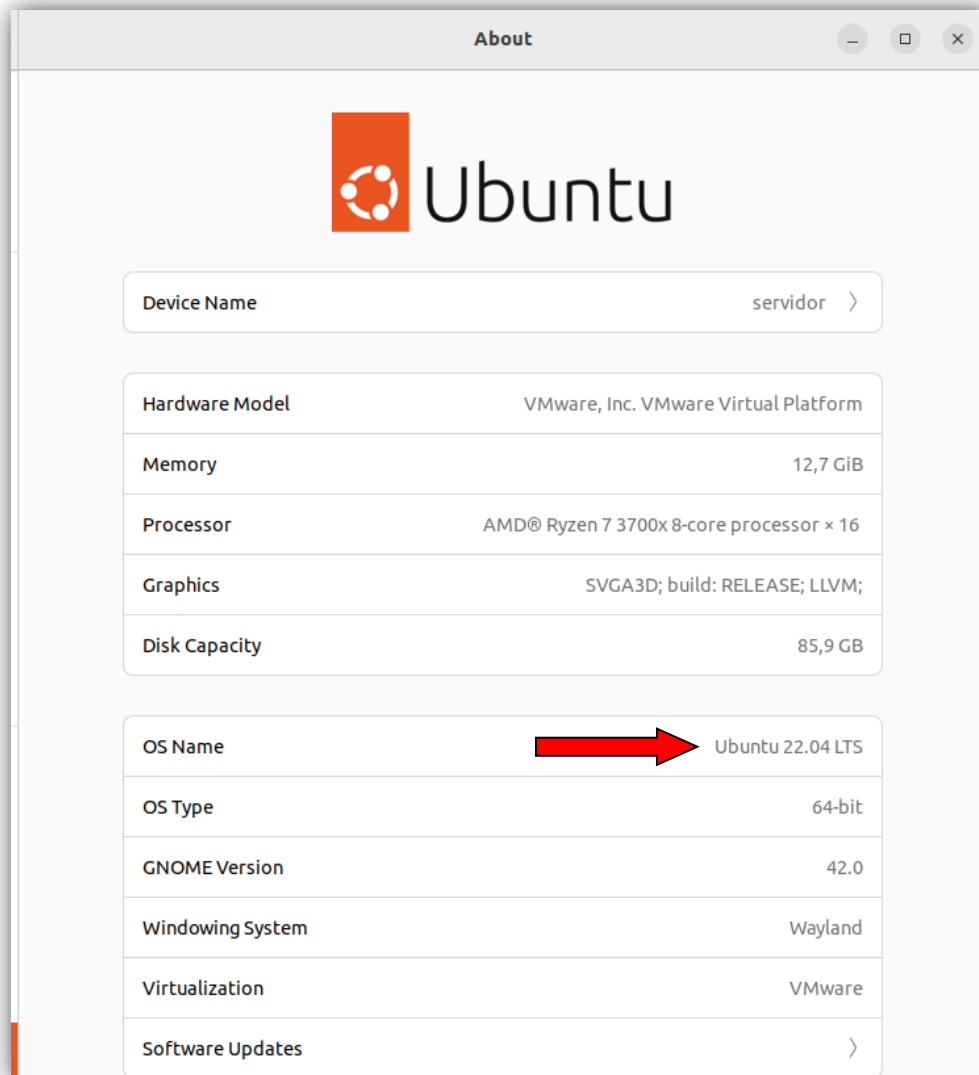
Es importante tener en cuenta que los requisitos pueden variar según la cantidad de usuarios concurrentes y las cargas de trabajo específicas que se ejecuten en la plataforma IsardVDI. Si se espera un alto nivel de actividad y uso intensivo de recursos, se recomienda un hardware más potente.

Para la realización de este proyecto necesitaremos lo siguiente en cuanto a Hardware y Software:

- Un PC con capacidad de virtualización sobre el que instalaremos un Ubuntu 20.04 (en una máquina virtual en este caso) que hará de servidor.
- VMware WorkStation: Máquina virtual de Ubuntu 20.04.
- Remote Viewer: para lanzar desde spice nuestro escritorio.
- Programas: Docker, Docker-Compose, IsardVDI
- Portátil que hará de empleado con conexión remota

Realización del Proyecto.

Para comenzar, debemos saber que IsardVDI funciona en distribuciones Linux. Por ello, este proyecto se ha realizado en una Máquina Virtual con **Ubuntu 22.04**:



Como preparación previa, tendremos que borrar las versiones anteriores de Docker que existan en nuestro equipo para posteriormente instalar las últimas versiones:

```
servidor@servidor-pc:~$ sudo apt-get remove docker docker-engine docker.io containerd runc
```

Una vez desinstalado cualquier versión antigua de Docker, actualizaremos el repositorio de Ubuntu de la siguiente manera:

```
servidor@servidor-pc:~$ sudo apt-get update

servidor@servidor-pc:~$ sudo apt-get install \
ca-certificates \
curl \
gnupg \
lsb-release
```

Agregamos la **clave GPG** oficial de Docker:

```
pc:~$ sudo mkdir -m 0755 -p /etc/apt/keyrings
pc:~$ curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

Configuramos el repositorio:

```
servidor@servidor-pc:~$ echo \
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/debian \
$(lsb release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Y ahora, para instalar Docker, volvemos a actualizar el repositorio tras su configuración:

```
servidor@servidor-pc:~$ sudo apt-get update
Obj:1 http://es.archive.ubuntu.com/ubuntu jammy InRelease
Obj:2 http://security.ubuntu.com/ubuntu jammy-security InRelease
Obj:3 http://es.archive.ubuntu.com/ubuntu jammy-updates InRelease
Obj:4 http://es.archive.ubuntu.com/ubuntu jammy-backports InRelease
Ign:5 https://download.docker.com/linux/debian jammy InRelease
Err:6 https://download.docker.com/linux/debian jammy Release
      404 Not Found [IP: 18.67.240.29 443]
Leyendo lista de paquetes... Hecho
E: El repositorio «https://download.docker.com/linux/debian jammy Release» no tiene un fichero de Publicación.
N: No se puede actualizar de un repositorio como este de forma segura y por tanto está deshabilitado por omisión.
N: Vea la página de manual apt-secure(8) para los detalles sobre la creación de repositorios y la configuración de usuarios.
```

Instalamos **Docker Engine**, **containerd** y **Docker Compose**:

```
pc:~$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

Verificamos que la instalación de Docker Engine ha salido bien ejecutando la **hello-world** imagen:

```
servidor@servidor-pc:~$ sudo docker run hello-world
[sudo] contraseña para servidor:
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:fffb13da98453e0f04d33a6eee5bb8e46ee50d08ebe17735fc0779d0349e889e9
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

servidor@servidor-pc:~$
```

También podemos comprobar que Docker está levantado ejecutando un **“systemctl status”**:

```
servidor@servidor-pc:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2023-03-19 15:25:19 CET; 34min ago
     TriggeredBy: ● docker.socket
   Docs: https://docs.docker.com
  Main PID: 8428 (dockerd)
    Tasks: 14
   Memory: 33.6M
      CPU: 625ms
   CGroup: /system.slice/docker.service
           └─8428 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
```


Hasta ahora hemos visto la instalación de **Docker** y **Docker compose** detallada en la página web oficial. Sin embargo, IsardVDI añade una serie de comandos con paquetes de configuración complementarios junto con la instalación de Docker y Docker compose:

```
servidor@servidor-pc:~$ sudo apt-get install -y \
  apt-transport-https \
  ca-certificates \
  curl \
  gnupg-agent \
  software-properties-common
```

```
servidor@servidor-pc:~$ curl -fsSL https://download.docker.com/linux/debian/gpg | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
```

```
servidor@servidor-pc:~$ sudo add-apt-repository \
  "deb [arch=amd64] https://download.docker.com/linux/debian \
  $(lsb_release -cs) \
  stable"
```

```
servidor@servidor-pc:~$ sudo apt-get update -y
```

```
*INSTALACIÓN*
servidor@servidor-pc:~$ sudo apt-get install -y docker-ce docker-ce-cli containerd.io
```

ALTERNATIVA DE DOCKER COMPOSE

Ante **posibles fallos** en la instalación, la web oficial de Docker recomienda este método de instalación específicamente para Docker compose, si los métodos anteriores no hubieran funcionado:

```
servidor@servidor-pc:~$ sudo apt-get update
```

Se necesita Docker compose 1.12 o posterior para Docker compose.yml v3.5. Se puede instalar la última versión usando **pip3**:

```
servidor@servidor-pc:~$ sudo apt install python3-pip -y
```

```
servidor@servidor-pc:~$ sudo pip3 install docker-compose
```

El hardware debe tener habilitada la virtualización. Podemos verificarlo desde la BIOS o en este caso desde CLI:

```
servidor@servidor-pc:~$ egrep '(vmx|svm)' /proc/cpuinfo
```

Para terminar la instalación, clonamos el repositorio y activar IsardVDI:

```
servidor@servidor-pc:~$ sudo git clone https://gitlab.com/isard/isardvdi
Clonando en 'isardvdi'...
```

Crearemos una nueva carpeta **isardvdi**, accederemos y haremos una copia de **isardvdi.cfg.example**:

```
servidor@servidor-pc:~$ cd isardvdi
servidor@servidor-pc:~/isardvdi$ sudo cp isardvdi.cfg.example isardvdi.cfg
```

Ejecutamos el siguiente script, siempre con privilegios ***sudo***:

```
servidor@servidor-pc:~/isardvdi$ sudo ./build.sh
```

Y activamos todo el despliegue de Docker compose con un **pull**:

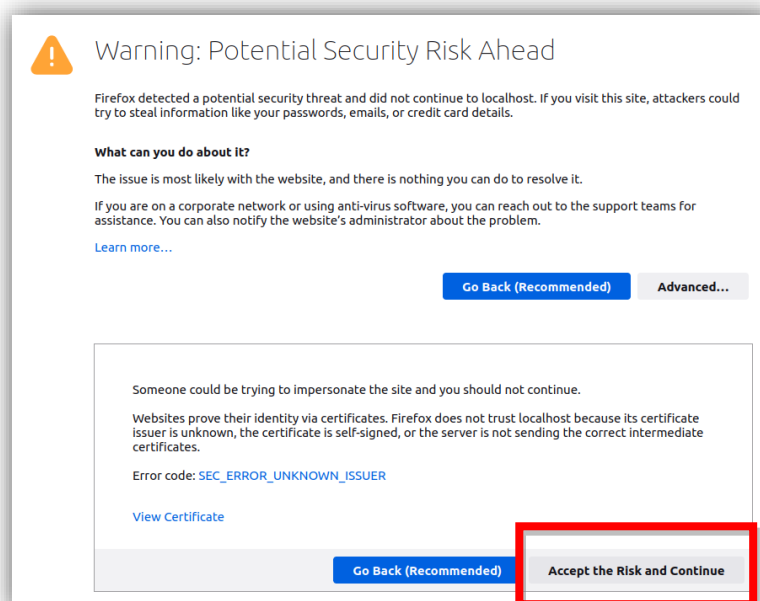
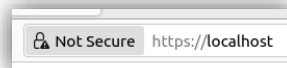
```
servidor@servidor:~/isardvdi$ sudo docker compose pull
[sudo] password for servidor:
[+] Running 233/22
:: isard-vpn Pulled 16.3s
:: isard-stats-rethinkdb Pulled 2.9s
:: isard-squid Pulled 107.7s
:: isard-stats-cadvisor Pulled 118.3s
:: isard-grafana Pulled 43.4s
:: isard-engine Pulled 127.5s
:: isard-grafana-agent Pulled 127.5s
:: isard-loki Pulled 128.4s
:: isard-webapp Pulled 102.8s
:: isard-storage Pulled 184.7s
:: isard-prometheus Pulled 146.3s
:: isard-websocketify Pulled 14.1s
:: isard-api Pulled 52.1s
:: isard-scheduler Pulled 39.6s
:: isard-static Pulled 18.3s
:: isard-guac Pulled 30.7s
:: isard-stats-node-exporter Pulled 128.4s
:: isard-db Pulled 39.9s
:: isard-stats-go Pulled 80.8s
:: isard-hypervisor Pulled 82.5s
:: isard-portal Pulled 146.8s
:: isard-authentication Pulled 15.2s
```

Y finalmente, levantamos Docker con: **sudo Docker compose up -d** (usaremos sudo Docker compose down para tirarlo)

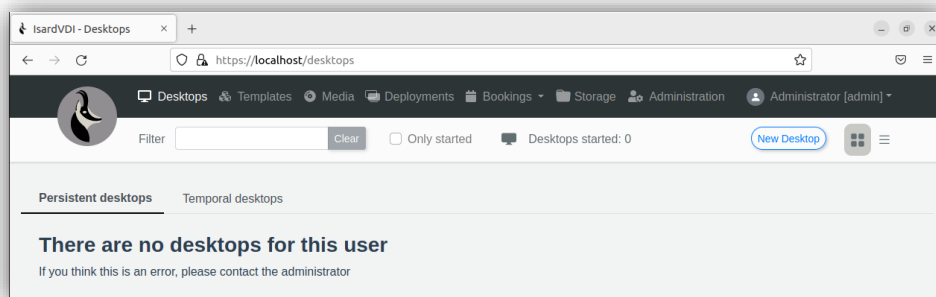
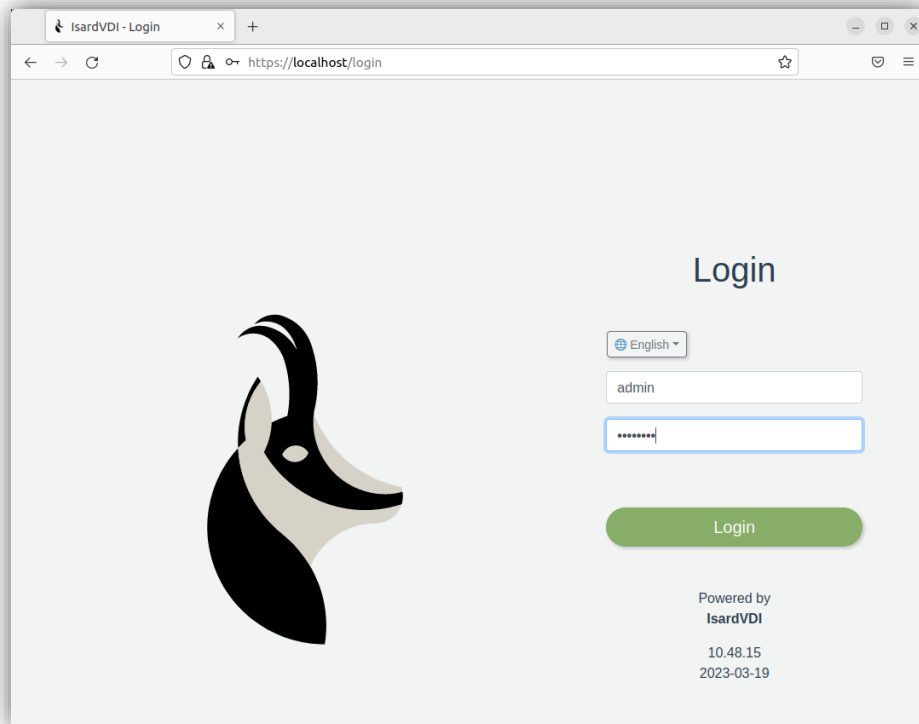
```
servidor@servidor:~/isardvdi$ sudo docker compose up -d
[+] Running 23/23
:: Network isard-network          Created                                0.1s
:: Container isard-portal          Started                             5.6s
:: Container isard-stats-cadvisor  Started                             5.6s
:: Container isard-db              Healthy                            20.3s
:: Container isard-hypervisor      Started                             5.7s
:: Container isard-grafana-agent   Started                             4.8s
:: Container isard-websockify      Started                             3.9s
:: Container isard-storage         Started                             3.9s
:: Container isard-static          Started                             4.1s
:: Container isard-guac            Started                             5.5s
:: Container isard-squid           Started                             5.5s
:: Container isard-grafana         Started                             4.0s
:: Container isard-stats-rethinkdb Started                             5.5s
:: Container isard-stats-node-exporter Started                          2.7s
:: Container isard-loki            Started                             4.0s
:: Container isard-prometheus      Started                             3.8s
:: Container isard-engine          Healthy                            50.7s
:: Container isard-api             Healthy                            60.7s
:: Container isard-scheduler       Started                             50.1s
:: Container isard-authentication  Started                             62.0s
:: Container isard-vpn             Started                             62.1s
:: Container isard-webapp          Started                             61.9s
:: Container isard-stats-go        Started                             61.8s
```

La primera vez que lo iniciamos esperamos 1 minuto aproximadamente ya que la base de datos se completará antes de que el inicio de sesión de IsardVDI esté disponible.

Una vez levantado, iremos a <https://localhost>. El usuario predeterminado es **admin** y la contraseña predeterminada es **IsardVDI**. Cuando accedemos, nos reconocerá como página NO segura. Simplemente continuaremos:



Esta es la interfaz principal de IsardVDI. Entonces, pondremos nuestras credenciales de administrador:



CONFIGURACIÓN

Para crear el archivo **docker-compose.yaml** tenemos que configurar "**isardvdi.cfg**":

```
servidor@servidor:~/isardvdi$ sudo nano isardvdi.cfg
```

```
GNU nano 6.2 isardvdi.cfg
# Isard main config v2.5.1

# ----- Docker Compose -----

## This configuration should generate docker-compose file for the following
## flavour
## Values: (first value is the default one)
## - all-in-one: all services in one docker-compose file
## - hypervisor: hypervisor and related service to access to their desktops
## - hypervisor-standalone: hypervisor without related services
## - video-standalone: services to access to desktops
## - storage: service to manage desktops disks
## - storage-base: base image for storage, just to improve CI/CD build
## - web: services to manage desktops
## - monitor: monitoring only host (grafana/loki/influxdb)
## - backupninja: standalone backups
#FLAVOUR=all-in-one
```

En este archivo podremos editar entre otras cosas:

- **Access web**

```
## Isard Engine
DOMAIN=192.168.214.133

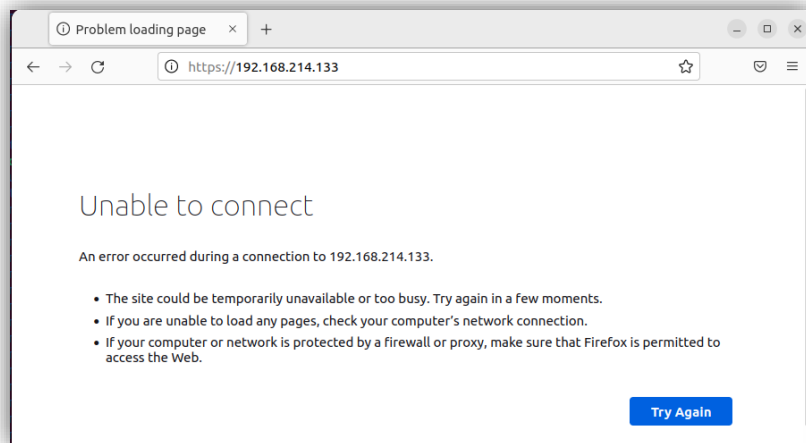
# ----- Admin password -----
## Initial WEB admin user password
## and authenticated backend password
WEBAPP_ADMIN_PWD=PruebaSrvr00
```

En la primera sección, cambiaremos localhost de **DOMAIN** y pondremos la IP del servidor. En **WEBAPP_ADMIN_PWD=** viene la contraseña por defecto "IsardVDI". También la cambiaremos.

Para efectuar los cambios y acceder con nuestra ip, tiramos Docker compose y lo volvemos a levantar:

```
servidor@servidor:~/isardvdi$ sudo docker compose down
[+] Running 23/23
  Container isard-stats-rethinkdb      Removed

servidor@servidor:~/isardvdi$ sudo docker compose up -d
[+] Running 22/23
  Network isard-network                Created
```



- Secret-Keys:

Podemos crear nuevas claves usando el comando "openssl rand -base64 32":

```
servidor@servidor:~/isardvdi$ sudo openssl rand -base64 32
[sudo] password for servidor:
oA2KYKBhdY4zJuxUC3pOj3z4SONHPc3kCrvkcRBMr08=
```

En isardvdi.conf:

```
WEBAPP_SESSION_SECRET=xq0Z3MP5ujxrQxtMGxgPiijH9xpuxkyP04R6At/V+g4=
API_ISARDVDI_SECRET=kpwPdT0ntI2XCEfzMp36hdSV9S42E7axS8D5TvP9c0A=
INFLUXDB_ADMIN_TOKEN_SECRET=9eFW/Qi29hL3hFGUP8wIGH89XKCH8s1k0il44GCRF2g=
API_HYPERVISORS_SECRET=B5/bUEUzIC+AjnQRmFh3vXR3VeIKirwdeL/xuHPV0+E=
```

- Backup:

Podemos programar Backups respecto a día, localización, etc:

```
# ----- Backups -----

Automated backups (https://0xacab.org/liberate/backupninja)
# If BACKUP_NFS_ENABLED is not enabled it will use this directory to create backups
# If BACKUP_NFS_ENABLED is enabled then this variable should be commented
BACKUP_DIR=/opt/isard-local/backup

# If nfs enabled you need to set server and folder also
BACKUP_NFS_ENABLED=false
BACKUP_NFS_SERVER=172.16.0.10
BACKUP_NFS_FOLDER=/remote/backupfolder

BACKUP_DB_ENABLED=false
BACKUP_DB_WHEN="everyday at 01"
BACKUP_DB_PRUNE="--keep-weekly=8 --keep-monthly=12 --keep-within=14d --save-space"

BACKUP_DISKS_ENABLED=false
BACKUP_DISKS_WHEN="everyday at 01"
BACKUP_DISKS_PRUNE="--keep-weekly=4 --keep-monthly=3 --keep-within=7d --save-space"
BACKUP_DISKS_TEMPLATES_ENABLED=false
BACKUP_DISKS_GROUPS_ENABLED=false
BACKUP_DISKS_MEDIA_ENABLED=false
```

- Certificate SSL:

La carpeta de certificados SSL es: **"/opt/isard/certs/default"**.

Para configurar el certificado srv.cat, tenemos que crear el directorio srv.cat en **"/opt/isard/certs/"**

```
servidor@servidor:~/isardvdi$ cd /opt/isard/certs/default
servidor@servidor:/opt/isard/certs/default$ ls
ca-cert.pem  ca-key.pem  chain.pem  server-cert.pem  server-key.csr  server-key.pem
```

```
servidor@servidor:/opt/isard/certs/default$ sudo cp -r ./* /opt/isard/certs/srv.cat/
```

Cuando tengamos los archivos SSL, ejecutamos:

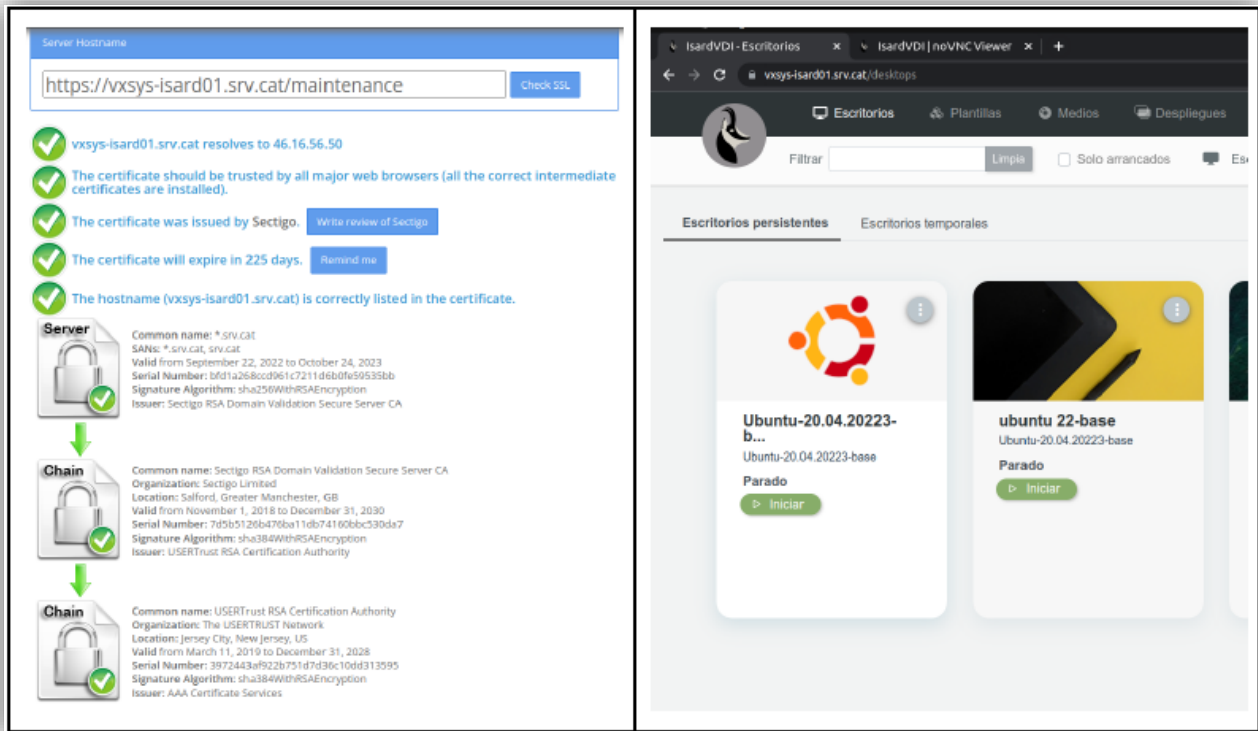
```
"sudo cat server-cert.pem chain.pem server-key.pem>
/opt/isard/certs/default/custom-portal-chain.pem"
```

```
servidor@servidor:/opt/isard/certs/srv.cat$ sudo su
root@servidor:/opt/isard/certs/srv.cat# sudo cat server-cert.pem chain.pem server-key.pem> /opt/isard/certs/default/custom-portal-chain.pe
```

Para actualizar los certificados SSL es necesario tirar y levantar docker:

```
servidor@servidor:~$ cd isardvdi/
servidor@servidor:~/isardvdi$ sudo docker compose down
[+] Running 23/23
```

```
Network isard-network removed
servidor@servidor:~/isardvdi$ sudo docker compose up -d
[+] Running 23/23
:: Network isard-network Created
```



<https://www.sslshopper.com/ssl-checker.html>

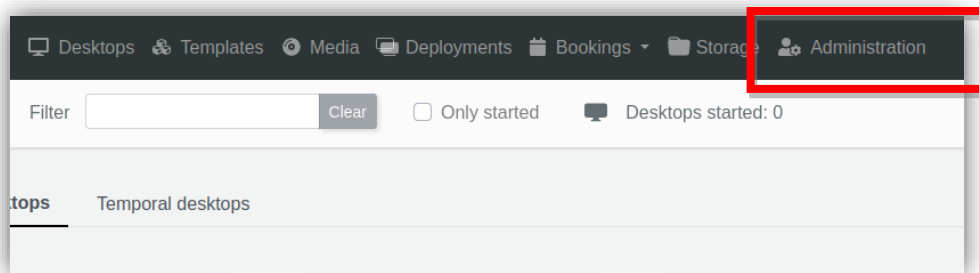
<https://vxsys-isard01.srv.cat/>

CREACIÓN DE GRUPOS Y USUARIOS

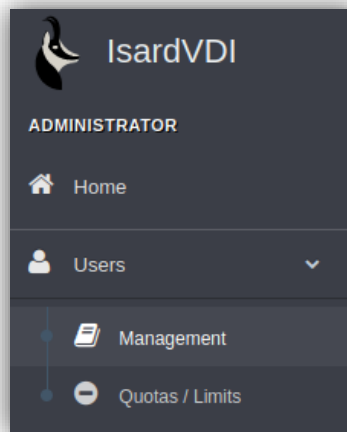
Para tener una buena organización, debemos gestionar los escritorios por departamentos:

Windows-10	Ubuntu-22.04.2
Develops	Develops
Marketing	Customer Care
Administrador	

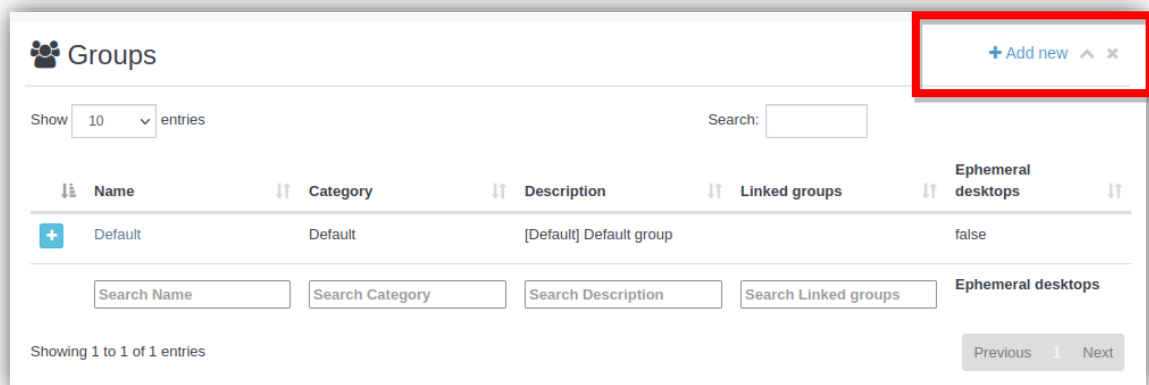
Una vez accedemos a IsardVDI, ahora sí, con nuestra IP, nos iremos a **Administration**:



Allí, nos vamos a la barra lateral izquierda en **Users > Management**:



En la sección Groups, le damos a **Add new**:



Aquí añadiremos los diferentes departamentos, pudiendo elegir si queremos que se creen los escritorios automáticamente o si queremos hacerlos temporales y por cuanto tiempo:

Add new Group

Group name and description

Name *
Customer Care

Description
Customer Care

Parent category *
Default - Default category

Linked groups ⓘ It will influence shared items.

Auto desktops creation
Enabled

Desktops list to be created at user login

Ephemeral desktops
Enabled

Minutes
5 34 62 91 120

Action on time: *
Stop

Cancel
Create Group

Groups

+ Add new

Show 10 entries
Search:

	Name	Category	Description	Linked groups	Ephemeral desktops
+	Customer Care	Default	[Default] Customer Care		false
+	Develops	Default	[Default] Develops		false
+	Marketing	Default	[Default] Marketing		false
+	Default	Default	[Default] Default group		false

Search Name

Search Category

Search Description

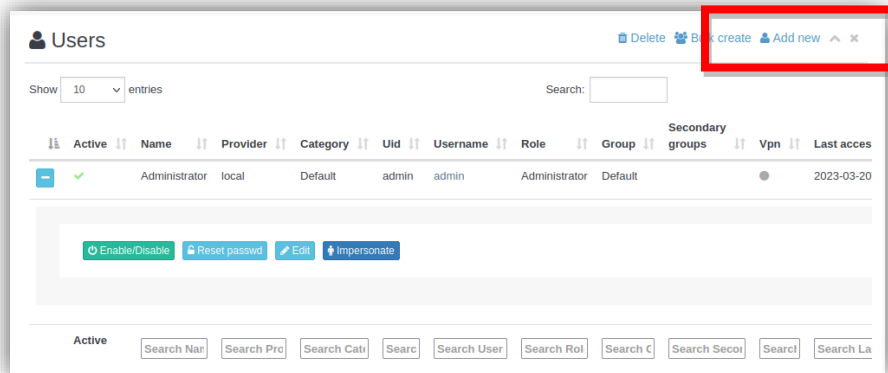
Search Linked groups

Ephemeral desktops

Showing 1 to 4 of 4 entries

Previous
Next

Más arriba, podremos crear a los usuarios en **Add new**:



+ User

Name *

Username *

Email

Password *

Repeat Password *

Role: *

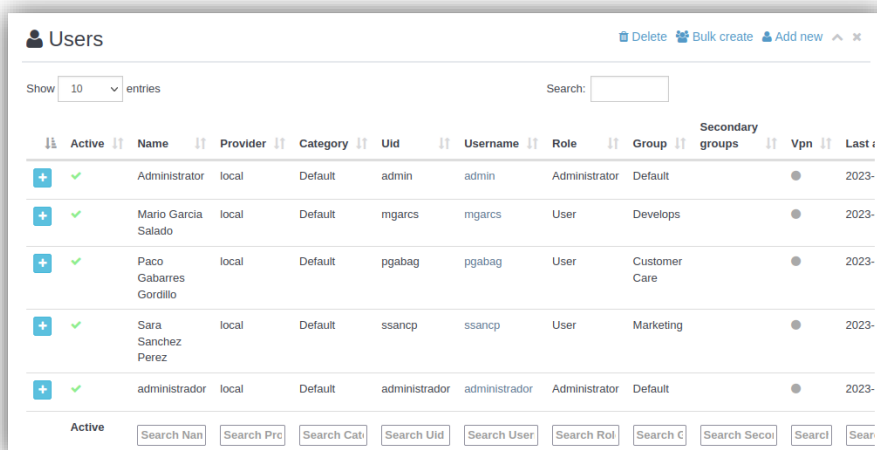
Category: *

Group: *

Secondary group:

It will influence both shared items and deployments creation.

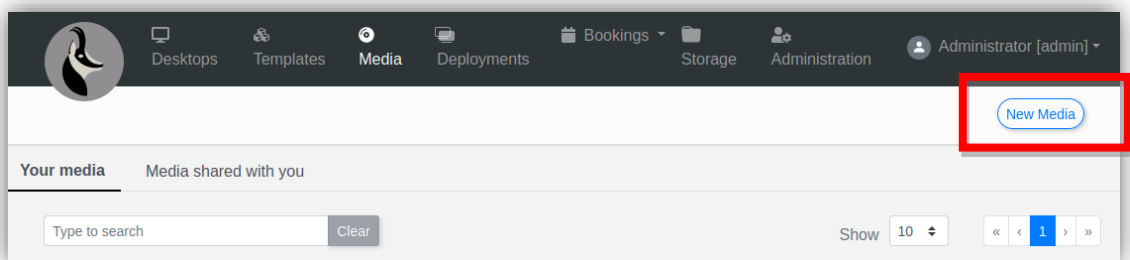
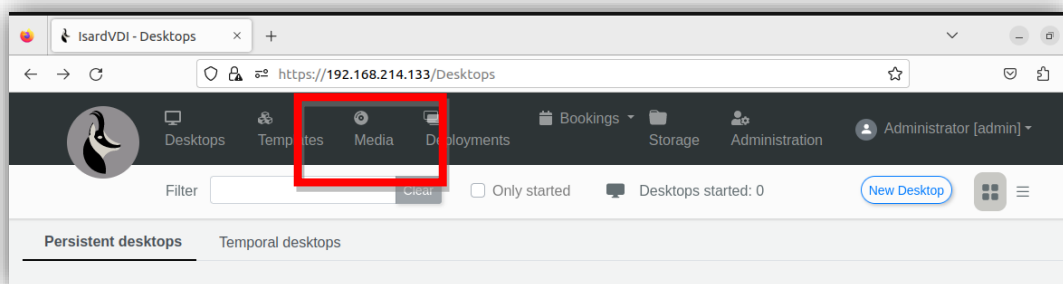
Hemos creado 4 usuarios. Uno por cada departamento y además del usuario Administrator, está el usuario **Administrador** que **pertenece a todos los departamentos**, por lo que tiene que poder ver todos los escritorios.



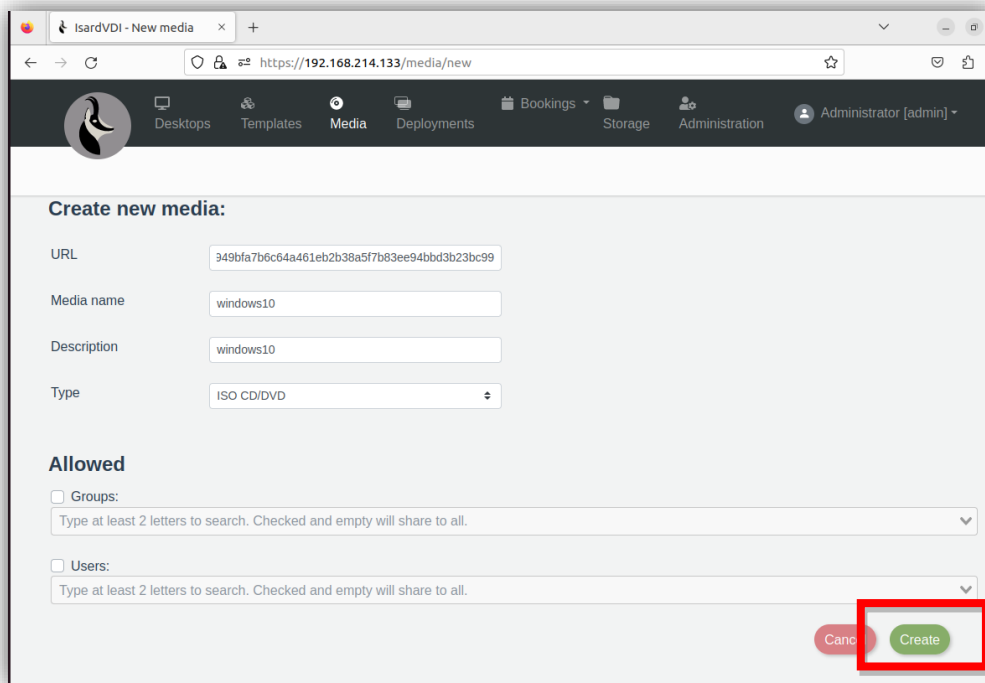
CREACIÓN DE TEMPLATES

Podemos crear escritorios a partir de ISOs descargadas desde la opción del menú **Descargas** o podemos subir nuestras propias ISOs desde la opción **Medios**. Una vez instalamos el sistema operativo y las aplicaciones, creamos una **Plantilla** y la asociamos al grupo o los usuarios que deseemos para que tengan un escritorio idéntico a la plantilla.

Para comenzar, subiremos una **ISO mediante URL**. Desde **Home** vamos a **Media > New Media**:



Ponemos **URL, nombre, descripción y el tipo de media**. Mas tarde añadiremos a los grupos y usuarios, una vez hayamos creado el Template:

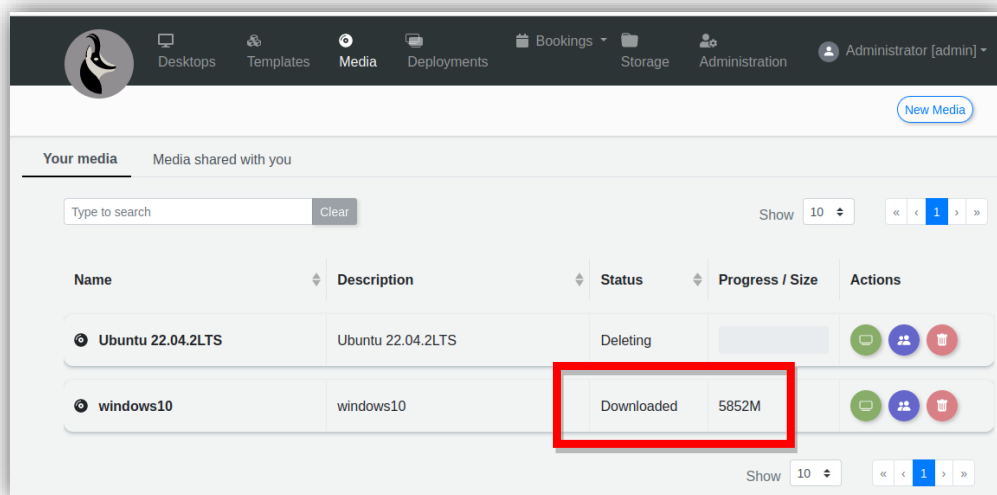


The screenshot shows the 'Create new media' form in the IsardVDI web interface. The form has the following fields:







- URL:** 349bfa7b6c64a461eb2b38a5f7b83ee94bbd3b23bc99
- Media name:** windows10
- Description:** windows10
- Type:** ISO CD/DVD

Below the form, there are sections for 'Allowed' groups and users, each with a search input and a dropdown menu. At the bottom right, there are 'Cancel' and 'Create' buttons. The 'Create' button is highlighted with a red rectangle.

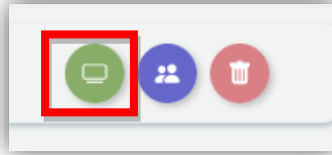
Le damos a **Create** y vemos como se ha subido, marcando el espacio ocupado en disco:



The screenshot shows the 'Your media' table in the IsardVDI web interface. The table has the following columns: Name, Description, Status, Progress / Size, and Actions. The 'windows10' media is highlighted with a red rectangle, showing its status as 'Downloaded' and its size as '5852M'.

Name	Description	Status	Progress / Size	Actions
Ubuntu 22.04.2LTS	Ubuntu 22.04.2LTS	Deleting		  
windows10	windows10	Downloaded	5852M	  

Una vez subida correctamente nuestra imagen, para lanzarla como escritorio le daremos al **botón verde de Objeto**:



Desde aquí podremos ponerle a nuestro escritorio virtual un nombre, una descripción, seleccionar con qué visor queremos verlo:

Create new desktop:

Info

Name

Description

Viewers

☒ Fullscreen disabled ☒ Fullscreen enabled

☒ ☒ ☒

☒ ☒

Browser SPICE RDP Browser RDP RDP VPN

Más abajo podremos cambiar las credenciales de acceso iniciales al escritorio, customizar sus recursos indicando el número de CPUs, RAM, drivers de video, el inicio, el Disk Bus y el tamaño del disco, además de agregar tarjeta de red y VPN:

RDP Login

⚠ **WARNING: Guest password will be stored in plain text!!**

Username:

Password:

Hardware

vCPUS: x v

Memory (GB): x v

Videos: x v

Boot: x v

Disk Bus: x v

Networks: v

Bookables

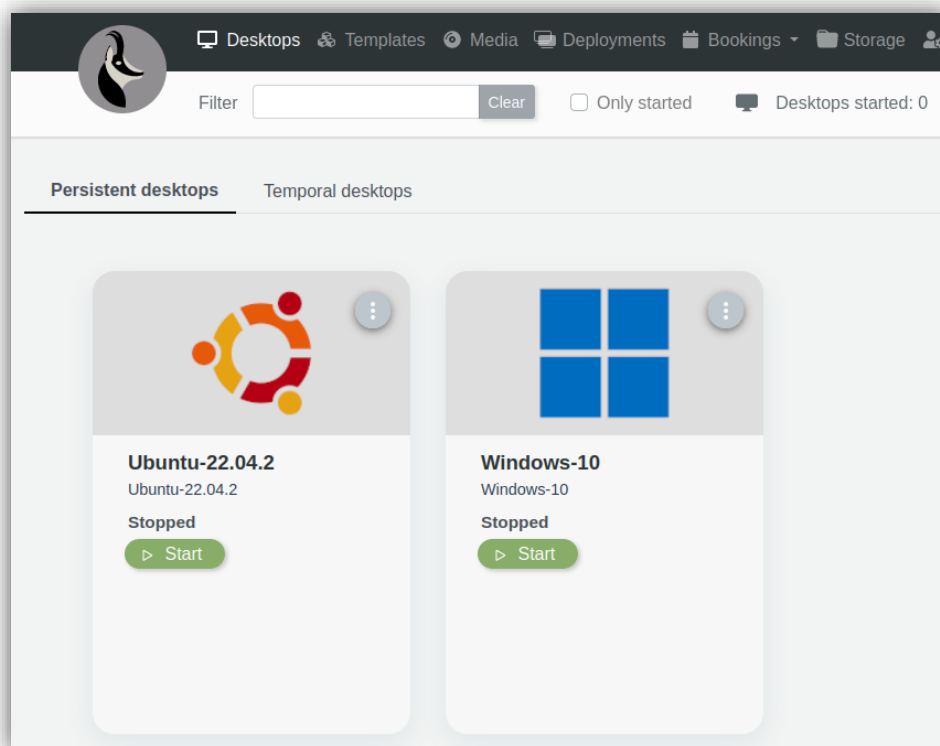
GPU: x v

Media

Isos: v

Floppies: v

Y vemos como resultado nuestro escritorio creado ***también añadimos el de ubuntu***:



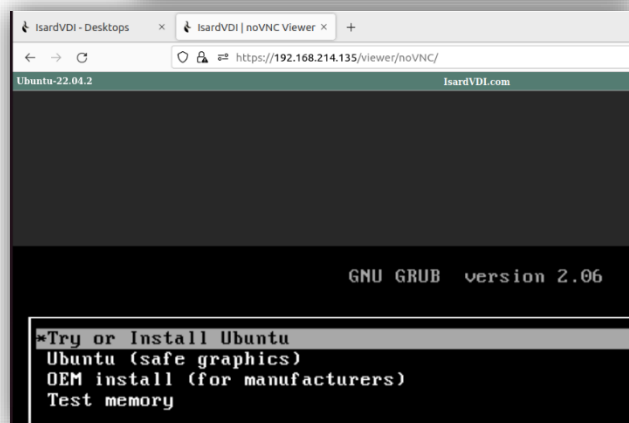
Le damos a **Start** y elegimos como abrirlo. En este caso empezaremos con el Navegador y luego con el visor Spice.

Desde el **navegador**, se nos abrirá una nueva ventana con nuestro escritorio, donde haremos la instalación de Windows 10 y los programas que queramos dejar para la posterior plantilla para los usuarios:

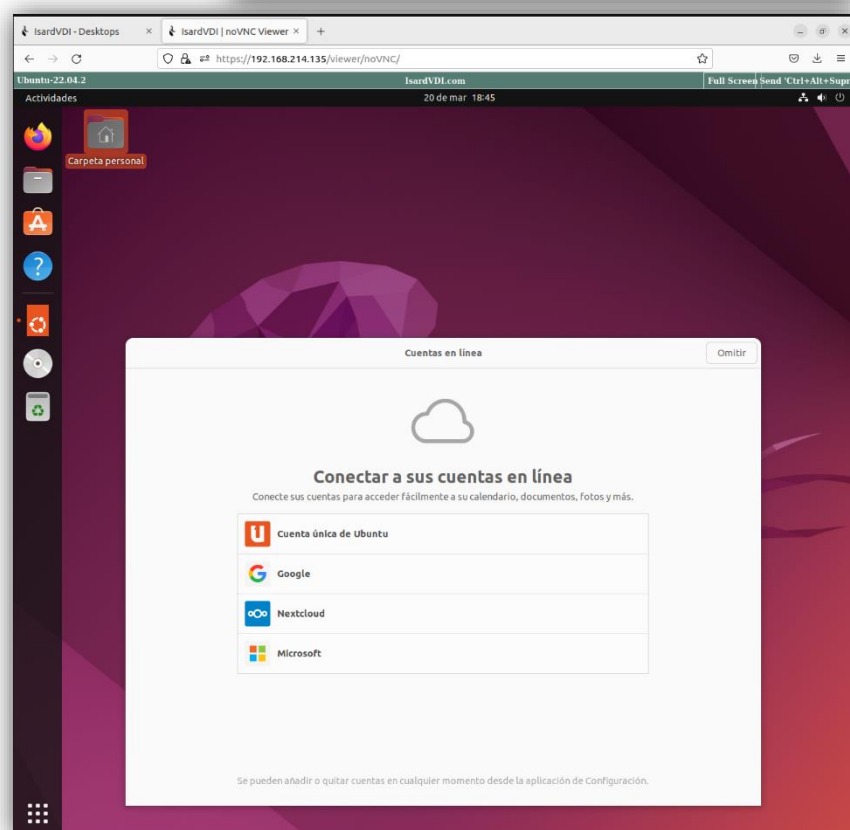
1º:



2º:



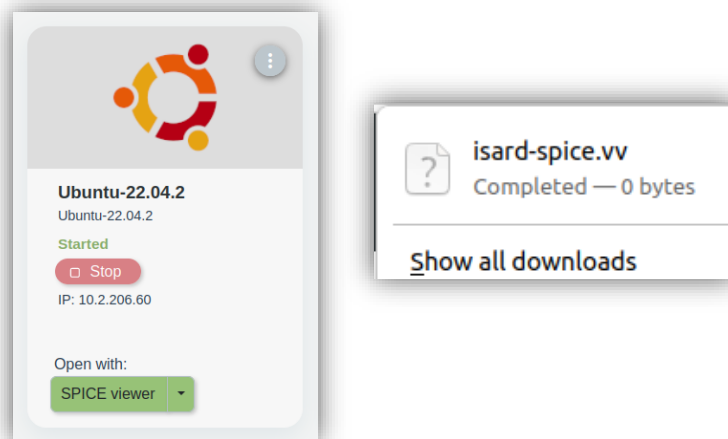
3º:



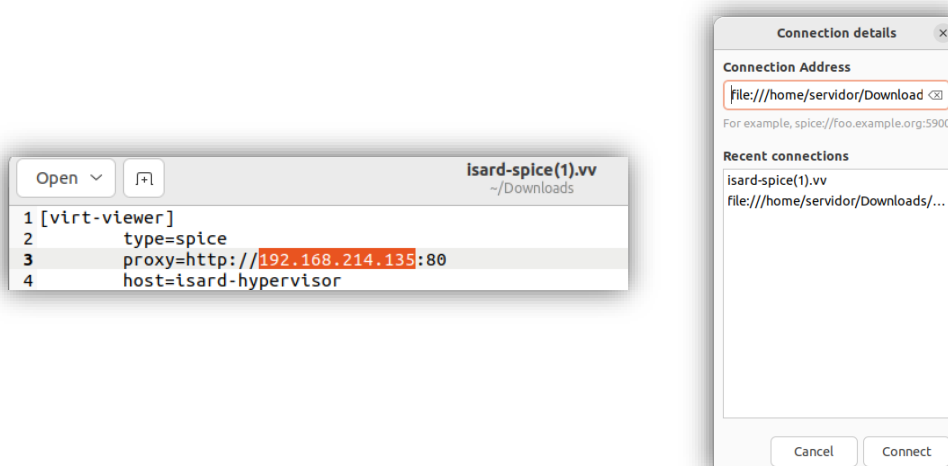
Creamos una carpeta de prueba para ver como se pasa al template y los usuarios pueden ver los cambios realizados:



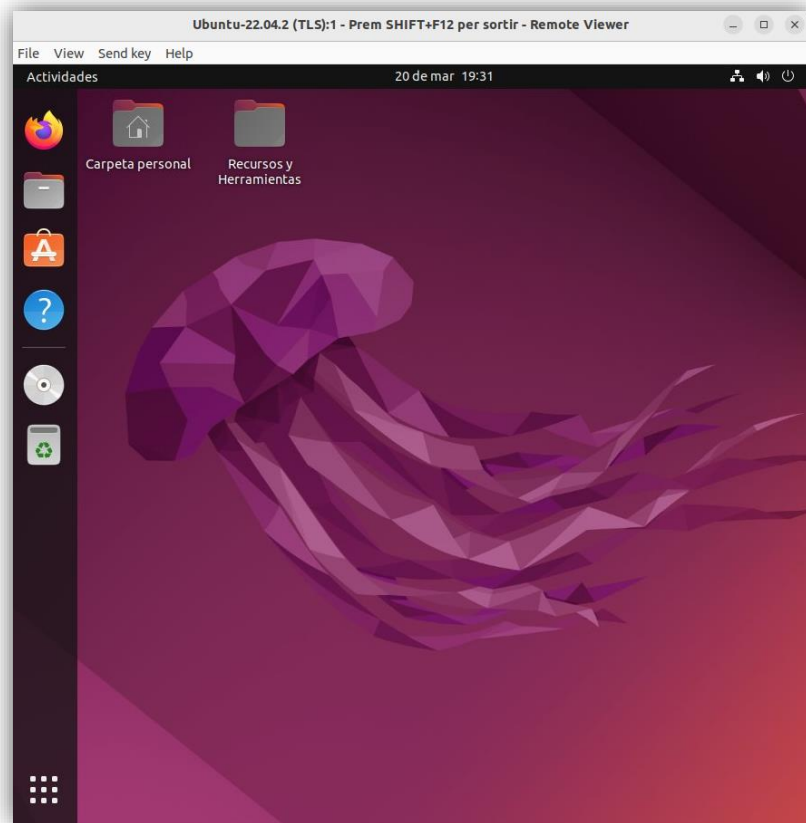
Ahora, con la opción de Spice viewer nos descargará un **archivo .vv** para arrancar nuestro escritorio en **Remote Viewer**:



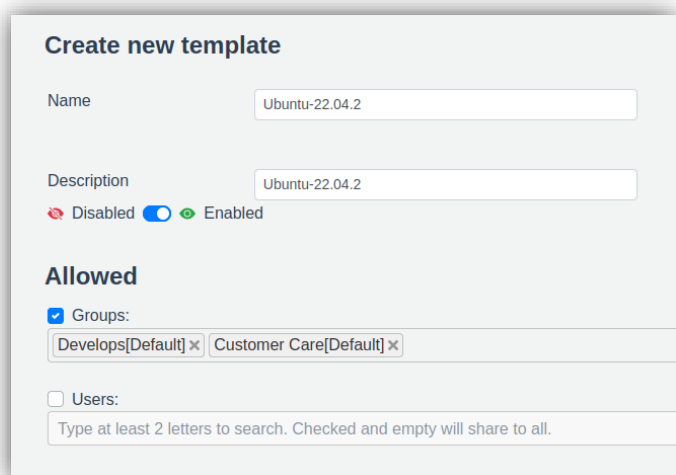
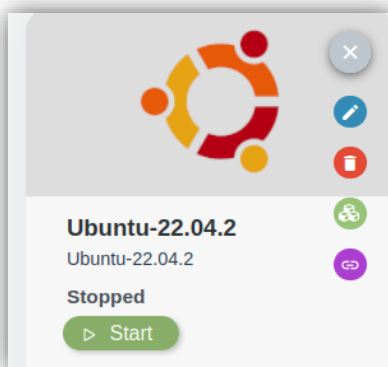
Editamos el archivo, ponemos la **IP del servidor** en localhost y abrimos con **Remote Viewer**:



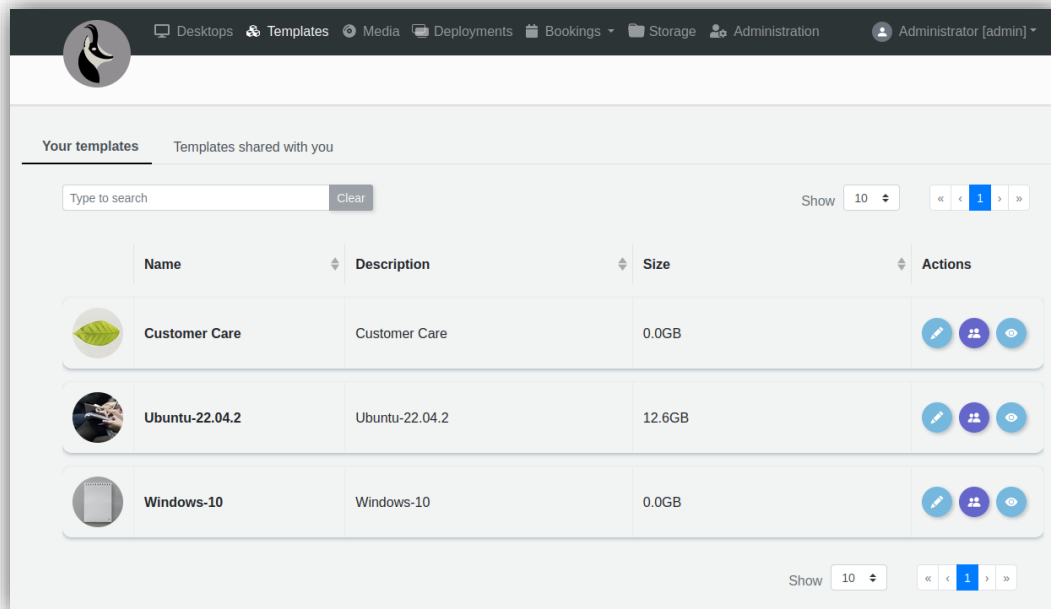
En **Remote Viewer** podemos notar una leve optimización comparado al navegador:



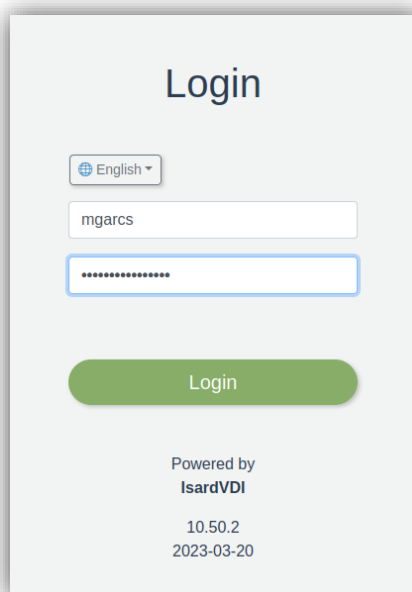
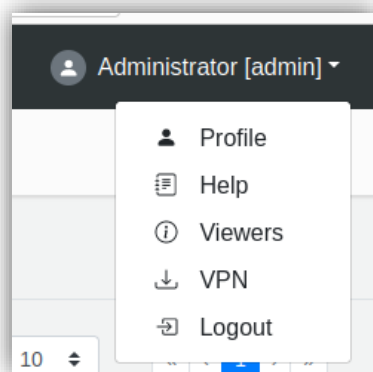
Una vez dejamos nuestro escritorio con la configuración deseada, editamos la instancia y le damos a **crear template**:

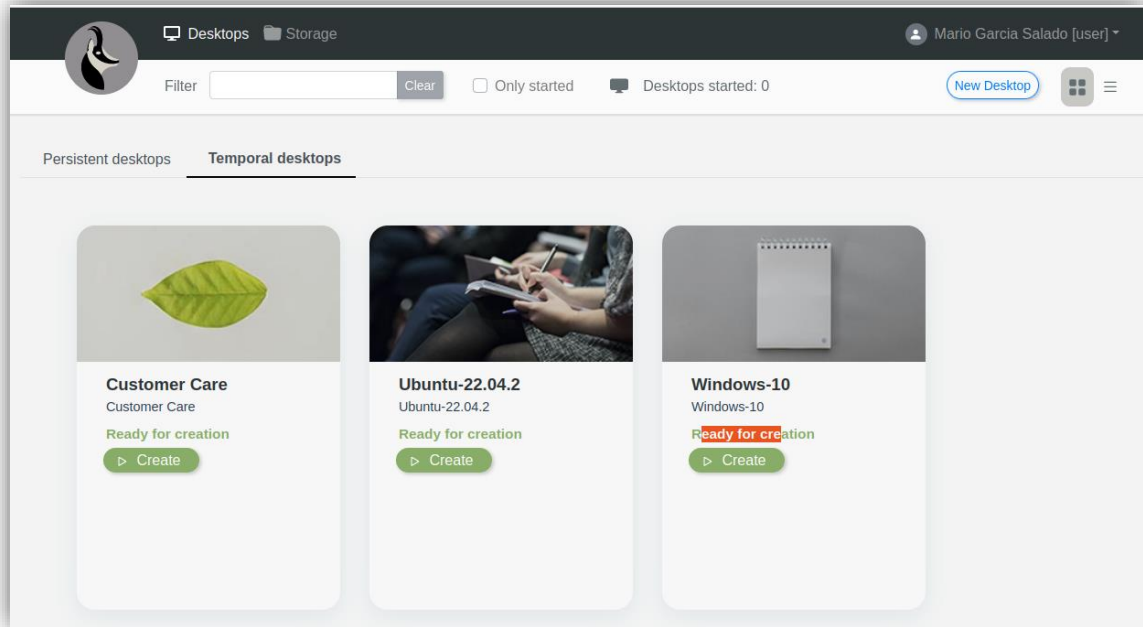


Y aquí vemos los **templates** creados:

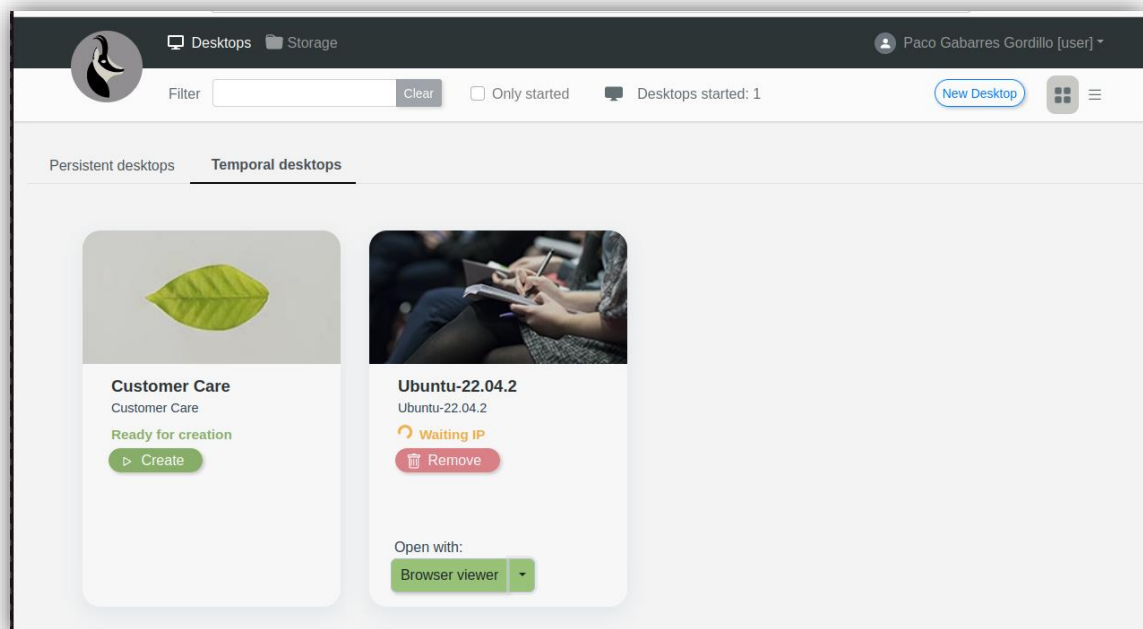


Ahora accederemos con diferentes usuarios de distintos departamentos para ver si las restricciones de las plantillas funcionan correctamente:

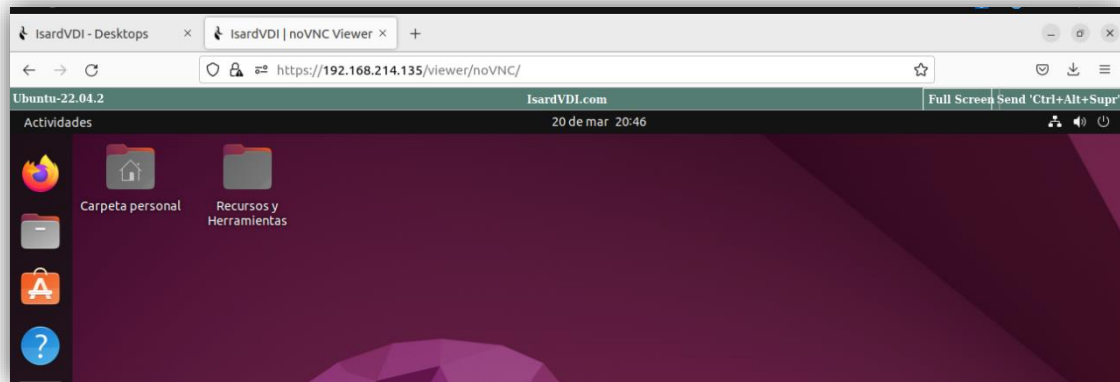




Mario tiene acceso a **todos los templates** porque el departamento de **Developers trabaja con todas las plantillas**. Veámos que pasa si iniciamos sesión con **Paco**, empleado de **Customer Care**:



De esta manera, Paco y cualquiera de los empleados puede acceder a la plantilla hecha específicamente para su departamento, desde cualquier sitio:



Problemas encontrados.

1.- Uno de los primeros fallos que he tenido al realizar este proyecto ha sido el siguiente:

Después de haber instalado Docker y Docker-Compose, seguir las instrucciones de la página de IsardVDI y lanzar Docker-Compose, sale el siguiente error:

```
servidor@servidor-pc:~/isardvdi/isardvdi$ sudo docker-compose pull
ERROR: The Compose file './docker-compose.yml' is invalid because:
'name' does not match any of the regexes: '^x-'
```

Tras mucha investigación, descubrí que el fallo era muy simple. En el tutorial inicial de IsardVDI la página te manda como pasos finales lanzar Docker-Compose pull y up -d.



El fallo está en que la última versión de compose no tiene “-”, por lo que en simplemente en vez de lanzar Docker-Compose deberíamos ejecutar docker compose, sin “-”.

2.- También, he tenido problemas a la hora de instalar Docker Compose. En la guía de instalación he dejado la solución.

3.- Y el último de las dificultades que he tenido ha sido a la hora de configurar correctamente los recursos para los escritorios virtuales, pues hacia falta tirar un poco mas de potencia para levantarlos correctamente y bien optimizados.

Modificaciones sobre el proyecto planteado inicialmente.

En principio la idea era montar una VPN con OpenVPN para proteger la red. Pero adentrándonos en IsardVDI vemos que ya proporciona su propia VPN. No está demás añadir seguridad con una red más privada.

Posibles mejoras al proyecto.

Además de la VPN, hay una gran cantidad de parámetros para configurar con grandes ventajas para servidores de gran magnitud. IsardVDI está pensado para trabajar con LDAP por lo que cuenta con muchas facilidades para esta.

Bibliografía.

IsardVDI Documentación: <https://isard.gitlab.io/isardvdi-docs/>

Instalación de IsardVDI: <https://www.isardvdi.com/>

Instalación de Docker: <https://docs.docker.com/engine/installation/>

Instalación de docker-compose: <https://docs.docker.com/compose/install/>