Run qemu in a docker container and qemu runs a riscv debian image

1.- Install Qemu and the binfmt_misc module.

To run a RISC-V Docker image on an Intel machine, you'll need to use QEMU user-mode emulation to allow the RISC-V binary to run on the Intel architecture.

```
apt-get install qemu-user-static
```

2.- Register the RISC-V emulator with binfmt_misc:

You can register QEMU as the handler for RISC-V binaries using

```
update-binfmts --install riscv64 /usr/bin/qemu-riscv64-static --magic
'\x7fELF\x02\x01\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02\x00\xf3\x00' --mask
'\xff\xff\xff\xff\xff\xff\xff\xfc\xff\xff\xff\xff\xff\xff\xff\xff\xfe\xff\xff\xff'
```

This tells the kernel to use the RISC-V QEMU binary when it encounters a RISC-V executable.

3.- Pull or build a RISC-V Docker image

4.- Run the RISC-V container

This has been tested to work in WSL2.

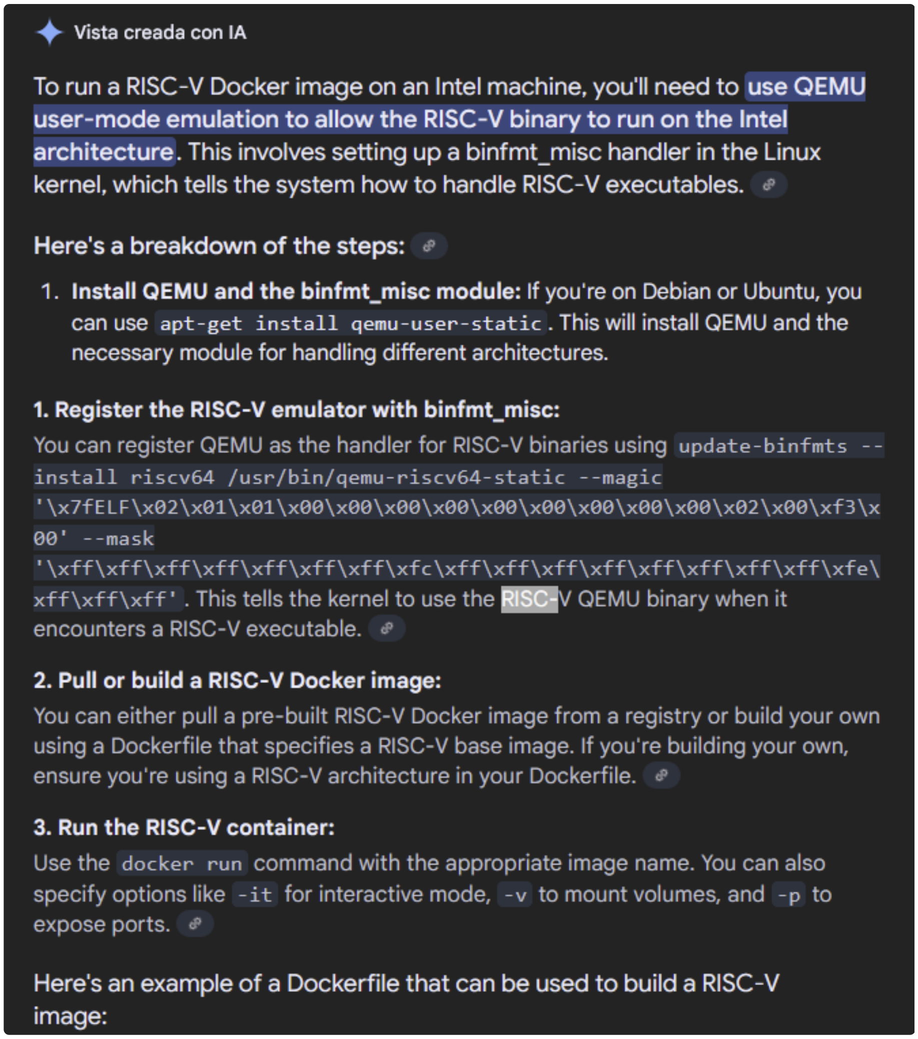
How to know the platform of a docker image? I.e.

```
docker run --rm mplatform/mquery drujensen/riscv-ubuntu:latest
```

Where drujensen/riscv-ubuntu is an image name.

Sources:

1. DavidBurela/riscv-emulator-docker-image
2. Emulating RISC-V Debian on WSL2
3.



Vista creada con IA

To run a RISC-V Docker image on an Intel machine, you'll need to use QEMU user-mode emulation to allow the RISC-V binary to run on the Intel architecture. This involves setting up a binfmt_misc handler in the Linux kernel, which tells the system how to handle RISC-V executables.

Here's a breakdown of the steps:

1. **Install QEMU and the binfmt_misc module:** If you're on Debian or Ubuntu, you can use `apt-get install qemu-user-static`. This will install QEMU and the necessary module for handling different architectures.

1. Register the RISC-V emulator with binfmt_misc:
You can register QEMU as the handler for RISC-V binaries using `update-binfmts --install riscv64 /usr/bin/qemu-riscv64-static --magic '\x7fELF\x02\x01\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02\x00\xf3\x00' --mask '\xff\xff\xff\xff\xff\xff\xff\xfc\xff\xff\xff\xff\xff\xff\xff\xff\xfe\xff\xff\xff'`. This tells the kernel to use the RISC-V QEMU binary when it encounters a RISC-V executable.

2. Pull or build a RISC-V Docker image:
You can either pull a pre-built RISC-V Docker image from a registry or build your own using a Dockerfile that specifies a RISC-V base image. If you're building your own, ensure you're using a RISC-V architecture in your Dockerfile.

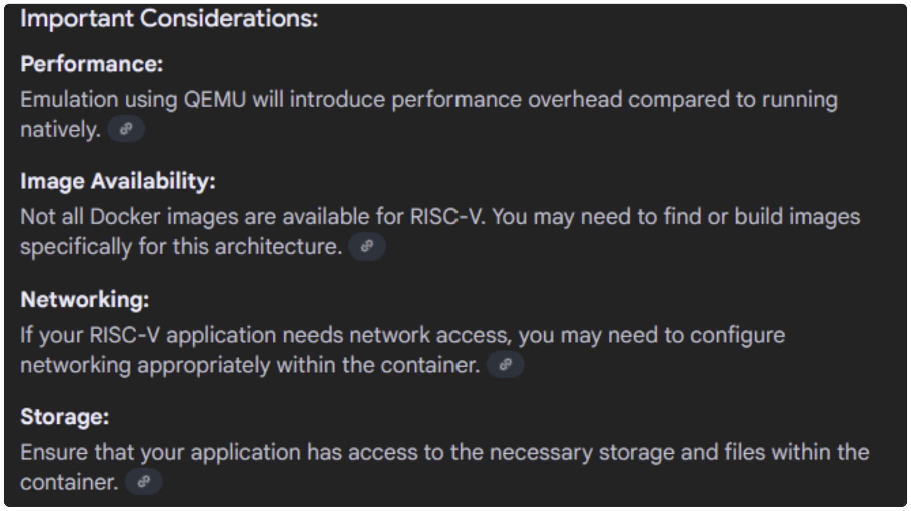
3. Run the RISC-V container:
Use the `docker run` command with the appropriate image name. You can also specify options like `-it` for interactive mode, `-v` to mount volumes, and `-p` to expose ports.

Here's an example of a Dockerfile that can be used to build a RISC-V image:

```
update-binfmts --install riscv64 /usr/bin/qemu-riscv64-static --magic
'\x7fELF\x02\x01\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02\x00\xf3\x00' --mask
'\xff\xff\xff\xff\xff\xff\xff\xfc\xff\xff\xff\xff\xff\xff\xff\xff\xfe\xff\xff\xff'
```

```
FROM ubuntu:latest

RUN apt-get update && apt-get install -y --no-install-recommends \

    qemu-user-static \

    && rm -rf /var/lib/apt/lists/*

# Install other necessary packages, e.g., build tools

RUN apt-get update && apt-get install -y --no-install-recommends \

    gcc \

    g++ \

    make \

    && rm -rf /var/lib/apt/lists/*
```

```

# Set the architecture

ENV ARCH riscv64

# Add entrypoint or command to run your application

# For example, if you have a compiled RISC-V executable:

# ENTRYPOINT ["/path/to/your/riscv/executable"]
```



Important Considerations:

Performance:

Emulation using QEMU will introduce performance overhead compared to running natively.

Image Availability:

Not all Docker images are available for RISC-V. You may need to find or build images specifically for this architecture.

Networking:

If your RISC-V application needs network access, you may need to configure networking appropriately within the container.

Storage:

Ensure that your application has access to the necessary storage and files within the container.

4. How to do something similar for ARM 🔗 How to Build and Run ARM Docker Containers on x86 Hosts · MatchboxBlog
5. How to get supported platforms for docker image with docker