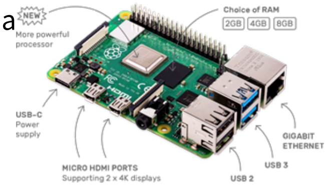
Article
Bob Schat · Feb 10, 2022 7m read

The InterSystems Iris Fhirserver running on a Raspberry Pi Raspberry running as a FHIRserver

The InterSystems Iris Fhirserver running on a Raspberry Pi Raspberry running as a FHIRserver

Raspberry running as FHIRserver

About a year ago I wrote some articles about the installation of the HAPI FHIRserver on a



Raspberry Pi. At that time, I only knew the basics of the FHIR standard, little about the technology behind FHIR-servers and not much more about the Raspberry. By trying, failing, giving up and trying again I learned a lot.More than a year later I ’ m still a “ perseverant amateur ” , but meanwhile I built up a full FHIR environment with Raspberries, a number of applications in HTML/JavaScript/jQuery/Bootstrap and some applications in PHP. I did some work with authentication- and authorization servers on the Raspberry and some experiments with SMART on FHIR.

Most of it works, although sometimes without me knowing what the difference is between the last non-working version and the success I obviously created. I ‘ m still surprised daily. Looking back, I should properly have studied systems development (I ’ ve a degree in forestry and currently volunteer a lot in that area. But that is different story).

Please be aware that a Raspberry Pi is ok to experiment with, but is not suitable for productional situations. So don ’ t use the Raspberry as a productional system in healthcare or any other situation in which reliability, continuity, security, privacy etc is needed or expected. It ’ s a perfect and cheap computer to experiment and it exceeds my expectations time after time, but it isn ’ t of “ industrial strength ” .

The InterSystems FHIRserver

At the time I have chosen the HAPI FHIR server because of its availability and because I succeeded quite soon to get some results with it. A former colleague, who is working for InterSystems now already suggested to experiment with InterSystems ’ FHIRserver too.

I know InterSystems quite well (or should I say “ I.know ”) and it sounded like fun, but at that time, more than a year ago, a suitable version of Iris wasn ’ t available for ARM -processors (the Raspberry runs on ARM). Apart from that is the community version of Iris distributed as a Docker-container and I knew less than nothing about Docker.

From Raspbian towards Ubuntu

Meanwhile a year has passed. There is a new and even more powerful Raspberry Pi available (while stocks last…). The new Raspberry 4, like the Raspberry 3, supports 64-bits Unix. T

he IRIS community version is available for ARM too and I learned somewhat more about containers and Docker.

With all these improvements, the nonsensical “ lock down ” we had in the Netherlands at the end of last year became an opportunity to start a new experiment in which I would try to get the community version of IRIS running on a Raspberry Pi .

Looking back this was quite easy. In fact, there are only three attention points:

1. You ’ ll need a Raspberry supporting 64 bits. This means either a Raspberry 3 or 4 .
2. I didn ’ t succeed with (64 bits) Raspbian (which is quite new by the way). But I succeeded with 64 bits Ubuntu.
3. The InterSystems FHIRserver seems to be more judgemental about the FHIR-calls than the HAPI FHIRserver. I will explain this later in an additional publication, but I don ’ t think this is bad. Being the central repository in an (health-)care environment you should be strict regarding the standards, otherwise it will be “ garbage in – garbage out ” .

It took however some time for me to understand while applications that ran ok with the HAPI FHIRserver failed running with IRIS. Spoiler alert: the problem was in my applications.

It is however fun and more than worthwhile to see a little computer like a Raspberry Pi, at costs less than 100 Euros, running the full InterSystems IRIS platform, complete with management portals, Ensemble, Caché and a FHIRserver.

In the next paragraph I ’ ll explain step-by-step what I did to get there. That will not be a lengthy story about the history of Ubuntu, the philosophy behind Docker etc. If you ’ re interested in that, please visit the Internet. I will focus on the “ happy flow ” and only when relevant mention where I firstly went wrong.

Installing the InterSystems FHIRserver

Phase 1. Ubuntu 64-bits

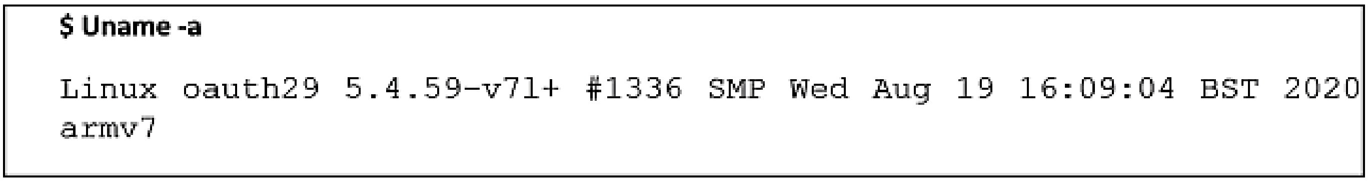
1. *Format your SD-card* and put the 64-bits version of Ubuntu for ARM-processors on it. The compressed version was named “ ubuntu-20.04.3-preinstalled-server-arm64+raspi.img.xz ” in my case and you ’ ll find it on http://cdimage.ubuntu.com . (Of course you have to decompress the .img file before putting it on the SD-card!).

Don ’ t use RaspberryPi-imager because that gives an incorrect version. I used Win32DiskImager.

2. *Start your Raspberry with the new image*. Change (and memorize!) the password for the ubuntu-account. wachtwoord van het ubuntu-account.

3. *Your Raspberry is running Ubuntu now, but we are not ready yet.*

Look at this:



```
$ Uname -a

Linux oauth29 5.4.59-v71+ #1336 SMP Wed Aug 19 16:09:04 BST 2020
armv7
```

This Ubuntu-version still is 32-bit !

We ’ re going to cure that now

1. Log in on your Pi

2. **sudo rpi-update**

3. **sudo reboot now**

5. **sudo copy /boot/config.txt /boot/config-ok.txt** (in case of errors we ’ ve got an escape)

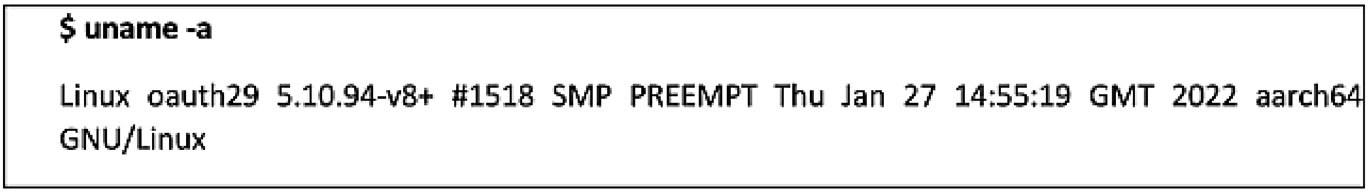
6. **sudo nano /boot/config.txt** Add under [pi4] the text : arm64bit=1

8. Save boot/config.txt (in nano: <ctr>O)

9. Exit from nano (<ctrl>X)

10. **sudo reboot now**

And now we ’ re running 64 bits. See for yourself:



```
$ uname -a

Linux oauth29 5.10.94-v8+ #1518 SMP PREEMPT Thu Jan 27 14:55:19 GMT 2022 aarch64
GNU/Linux
```

Make a notation of your Pi ’ s IP-address (**sudo hostname -I** where I is the I from "Island") and you can put it away since in Ubuntu SSH is enabled by default.

Phase 2: Docker

1. **sudo curl -fsSL** https://get.docker.com **-o get-docker.sh**

2. **sudo sh get-docker.sh**

This will result (after a while) in an extensive notification of Docker about using it as a “ non-priviliged user ” .

3. **sudo usermod -aG docker $USER**

4. Log off and log on again

5. **docker container run hello-world**

Docker now looks for a “ hello-world ” image on your system. It will not find it there and than will download it from the Docker-repository. After that Docker puts the image in a container on the Pi and starts the “ hello-world ” image.

You can see that confirmed by another block of text from Docker, including the “ Hello from Docker ” welcome message.

Fase 3: The Docker image for InterSystems’ IRIS FHIRserver

1. Give the following command in one time:

```
sudo docker run --name my-iris -d --publish 9091:1972 --publish 9092:52773
containers.intersystems.com/intersystems/irishealth-community-arm64
```

Now a lot of events happen after eachother:

- Docker will get the “community edition” from Irishealth and puts it in your local Docker repository.
- Docker starts the community edition in a docker container named “my-iris”.
- Docker maps port 1972 of IRIS, running in the container, to port 9091 of your Raspberry.
- Following the same pattern, port 52773 of IRIS (in the container) becomes port 9092 on your Raspberry.

You can see what happened by:

2. **sudo docker container ls** But there is another, much more impressive way to experience it.

Phase 4. Start the InterSystems IRIS for Health Fhirserver

Start a webbrowser in your network, for example on your PC en go to:

<ip-address-of-your-Raspberry>:9092/csp/sys/UtilHome.csp .

You’ll see the managementportal of IRIShealth. The portal automatically adjusts to the language on your PC.

The standard account is _SYSTEM (with a leading underscore) and the initial password is SYS (without underscore). IRIS requires to change the password immediately.

Now you’re able to configure the IRIS for Health FHIRserver and start it up, following the instructions on: https://docs.intersystems.com/irisforhealthlatest/csp/docbook/DocBook.UI...

You have to configure a “Foundation” and to define an “Endpoint” (FHIR 3 or 4).

The “endpoint” will result in the URI on which you can access the FHIRserver. In my case the “Patient” resource is on http://192.168.1.29:9092/csp/healthshare/fhironpi/fhir/r4/Patient (doesn’t work from outside!)

Conclusion

Has this been worth the trouble? Yes, more than that! To start with: there wasn't much trouble. I didn't succeed with Raspbian, but that was solved easily. And having the IRIS-for-health platform running on device like a Raspberry Pi is impressive. All the dashboards (DeepSee), dataconversion-tools, Ensemble (an Enterprise Service Bus plus a lot of extra tools), CSP-pages and Caché (a multi-dimensional database) seem to be available to experiment with and even I.know for natural language processing is visible in the management portal. The IRIS-for-health platform offers much more than a FHIRserver allone and far too much to describe in one article.

For me personally IRIS-for-health has an extra bonus since I worked (as an architect) with Ensemble almost as soon as it was there and apart from that IRIS-for-health's earliest start was in the second half of the 2-th century. The MUMPS development environment for healthcare applications (MUMPS: Massachusetts General Hospital Utility Multi-Programming System) ran mostly on VAX computer systems of Digital Corp. And my professional career in IT started at that time with VAXes.

MUMPS, Caché, Ensemble and today the IRIS-for-Health platform have great stability and require relatively modest support and maintenance. The documentation of the InterSystems products is extensive and easy to find.

I will experiment more with IRIS and the FHIRserver and I promise to at least publish one more article: about my "struggle" with a strict adherence to the FHIR-standard!

Stay healthy, keep thinking, play on!

Bob

#Beginner #Containerization #FHIR #InterSystems IRIS

Source URL:https://community.intersystems.com/post/intersystems-iris-fhirserver-running-raspberry-pi-raspberry-running-fhirserver