VRSpace.org - Feature #156

Easily Deployable Container

09/11/2022 03:46 PM - Nate Lager

Status:	Closed	**Start date:**	09/11/2022
Priority:	Normal	**Due date:**	
Assignee:	Nate Lager	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour

Description

I've decided to take on the task of making a yaml definition that will deploy vrspace and the openvidu voice server in podman containers using podman play kube. This gives us a few things.

- this definition will with on podman, and kubernetes based orchestration.
- this will make it easy for folks who are not as familiar with the technologies in use to deploy a vrspace setup to toy with

This will however not give us:

- easy docker deployments

docker-compose could give us easier docker deployments, and this should also be looked at. I may do this once I have the podman setup worked out. Unfortunately docker-compose is not as easy to deploy on a podman container host. I use podman as my platform so thats why I am interested in making this work with podman. HOWEVER there's no reason not to repeat the work for docker-compose.

The proposed deployment:

- The definition will stand up the vrspace container that I maintain at: https://hub.docker.com/r/gangrif/vrspace it will assume that the administrator will handle routing of traffic to it (in my case i use an nginx proxy), and the port which it listens on will be configurable.
- It will also stand up openvidu, and vrspace will be configured out of the box to talk to openvidu on localhost. This is one of the pieces that I need to work out still. My understanding of the openvidu connection is that vrspace needs to connect to openvidu, not the end user, correct? If this is correct this should be easy to configure.
- It will define persistent volumes for things like worlds, config, and likely database.

Questions:

- My understanding is that vrspace maintains a database, my assumption is that this database should be persistent. What sort of data goes in there? and will it someday be split into a database service rather than a file?

History

#1 - 09/11/2022 05:33 PM - Nate Lager

First question...
The easiest way to pass in config (like.. vidu's location and password) would be on the command line.

I pass the config file on the command line so I can tell vrspace where to find the custom config if there is one, with

```
--spring.config.location=/config/application.properties
```

Can I do the same thing for other items in the config file? For example, could I add:

```
--openvidu.publicurl=http://127.0.0.1:4080 --openvidu.secret=somekindapss
```

?

#2 - 09/11/2022 05:35 PM - Nate Lager

still learning the markup used by redmine.. I think you get the point that my ```'s were meant to be the command line args im attempting to add to java.

#3 - 09/11/2022 05:49 PM - Nate Lager

RTFM

I now see the answer to two of my questions in the wiki. Client browsers need access to vidu directly if im reading your diagram properly. and the command line looks like

-Dopenvidu.publicurl=YOUR_URL and -Dopenvidu.secret=YOUR_SECRET

#4 - 09/12/2022 12:23 PM - Josip Almasi

Nate Lager wrote:

> still learning the markup used by redmine.. I think you get the point that my ```'s were meant to be the command line args im attempting to add to java.

Right, try < pre > and < /pre > tags, like

```
whatever
```

#5 - 09/12/2022 01:06 PM - Josip Almasi

OK to answer your questions:

See image at https://redmine.vrspace.org/projects/vrspace-org/wiki#Software-Architecture
So both browser and VRSpace server talk to OpenVidu.
VRSpace server handles chatroom creation and destruction on demand, when first user enters/last user exits a space.
For each user, a token is requested from OpenVidu, and passed to the client. The browser then uses the token to connect to chatroom for the world.

Then, see https://redmine.vrspace.org/projects/vrspace-devops/wiki - VRSpace DevOps project is not public, but you should have access.
Note plenty of ProxyPass/ProxyPassReverse pairs.

In vrspace-ssl.ssl, this is about static content, as we have 1G+ of it, and it grows with every model you get from sketchfab. So everything that's not static, needs to be explicitly ProxyPass-ed. We can't simply ProxyPass everything, as it would cause embedded tomcat to server static content. That in turn error instead of 3D canvas when vrspace server is down.

Then we have much the same in vidu-ssl.conf, but with a bit different purpose, in a word - hack :) It just wouldn't work without it, though I don't recall exact symptoms. Initially I attempted to run OpenVidu on the same web server, but ultimately failed. It's just not made for this, requires it's own FQDN. So I ended buying a wildcard cert. These ProxyPasses are are artifacts of my previous attempts, proxying everything might just work for you.

Also note special handling of websocket proxying.

Now that I mention certs, this is major hustle with both. I'd love to have functional Let's Encrypt setup, with auto-renewal and everything.

State of the system is persisted into the database. DB contains worlds, users, bots, content... everything. It's populated by VRSpace server, on demand. And is queried all the time, like select * from world where coordinates < my_coordinates+radius. Specifically, these are stored:
https://www.vrspace.org/docs/javadoc/org/vrspace/server/obj/package-summary.html
This embedded database is exactly the same Neo4J that you get if you download stand-alone server, listens on the same port and everything, except it doesn't come with fancy UI that comes with Neo4j distro. Meaning, you can't really manage it as it is. But you can connect fancy UI that you've downloaded with Neo4j distro to it.
So I think it's simply out of scope, keep it simple.

#6 - 09/12/2022 02:36 PM - Nate Lager

Josip Almasi wrote:

> OK to answer your questions:
>
> See image at https://redmine.vrspace.org/projects/vrspace-org/wiki#Software-Architecture
> So both browser and VRSpace server talk to OpenVidu.
> VRSpace server handles chatroom creation and destruction on demand, when first user enters/last user exits a space.
> For each user, a token is requested from OpenVidu, and passed to the client. The browser then uses the token to connect to chatroom for the world.
>
> Then, see https://redmine.vrspace.org/projects/vrspace-devops/wiki - VRSpace DevOps project is not public, but you should have access.
> Note plenty of ProxyPass/ProxyPassReverse pairs.
>
> In vrspace-ssl.ssl, this is about static content, as we have 1G+ of it, and it grows with every model you get from sketchfab. So everything that's not static, needs to be explicitly ProxyPass-ed. We can't simply ProxyPass everything, as it would cause embedded tomcat to server static content. That in turn error instead of 3D canvas when vrspace server is down.
>
> Then we have much the same in vidu-ssl.conf, but with a bit different purpose, in a word - hack :) It just wouldn't work without it, though I don't recall exact symptoms. Initially I attempted to run OpenVidu on the same web server, but ultimately failed. It's just not made for this, requires it's own FQDN. So I ended buying a wildcard cert. These ProxyPasses are are artifacts of my previous attempts, proxying everything might just work for you.

> Also note special handling of websocket proxying.
>
> Now that I mention certs, this is major hustle with both. I'd love to have functional Let's Encrypt setup, with auto-renewal and everything.
>
> State of the system is persisted into the database. DB contains worlds, users, bots, content... everything. It's populated by VRSpace server, on demand. And is queried all the time, like select * from world where coordinates < my_coordinates+radius. Specifically, these are stored:
> https://www.vrspace.org/docs/javadoc/org/vrspace/server/obj/package-summary.html
> This embedded database is exactly the same Neo4J that you get if you download stand-alone server, listens on the same port and everything, except it doesn't come with fancy UI that comes with Neo4j distro. Meaning, you can't really manage it as it is. But you can connect fancy UI that you've downloaded with Neo4j distro to it.
> So I think it's simply out of scope, keep it simple.

Thanks for all the info. Should this issue be under the vrspace-devops project then?

Once I have everything worked out, I want to make each piece of the vrspace deployment as automate-able as possible.

My usual deployment of any web based app is to not bother with ssl at the container level, and instead handle that at the proxy. Do you see any reason that wont work out here?

vidu appears to make its own internal self signed tls cert. So that container will have tls at its disposal, but vrspace does not, correct?

Letsencrypt is a great solution. Openvidu's docs suggest that it can use letsencrypt simply by enabling it and making sure the proper ports are open to the world. If we could do the same in vrspace, that would make it much simpler to keep the tls connections up from the user all the way to the service, encrypting the back-channel traffic as much as the public.

#7 - 09/12/2022 03:31 PM - Josip Almasi

Nate Lager wrote:

> Thanks for all the info. Should this issue be under the vrspace-devops project then?

No. DevOps is not public because it exposes configuration of a live server. Like, production :) The configuration wasn't peer reviewed so I'm not exactly sure it's safe. (what do you think?)
But since you intend to publish everything, it should be open.

> Once I have everything worked out, I want to make each piece of the vrspace deployment as automate-able as possible.
>
> My usual deployment of any web based app is to not bother with ssl at the container level, and instead handle that at the proxy. Do you see any reason that wont work out here?

No, but my knowledge of containers is poor. You'll need some way to access the server log, at least. That shouldn't require ssh.

> vidu appears to make its own internal self signed tls cert. So that container will have tls at its disposal, but vrspace does not, correct?

Yeah but vidu self-signed cert is not exactly a feature. Users get mysterious errors visible only in javascript console, until they open vidu link manually and accept the cert there.
You can also enable self-signed cert in vrspace server, just set server.ssl.enabled=true.
I've opted to terminate ssl on the proxy.

> Letsencrypt is a great solution. Openvidu's docs suggest that it can use letsencrypt simply by enabling it and making sure the proper ports are open to the world. If we could do the same in vrspace, that would make it much simpler to keep the tls connections up from the user all the way to the service, encrypting the back-channel traffic as much as the public.

Right.

Now, did you see that OpenVidu isn't just a program, it's bundle of bunch of servers behind nginx? OpenVidu is essentially chatroom manager for kurento, there's stun servers and whatnot in there. By setting up another reverse proxy in front of it all, like I did, you end up with too complex topology.
So I think one container with proxy+vrspace and another one with openvidu as it is may be the optimal solution. But that requires two FQDNs and valid certificates for both.

#8 - 09/20/2022 11:00 AM - Benjamin LIPERE

Hello Nate Lager.

I am doing something on Kubernetes, right now I have Infrastrucutre problems, but once fixed, it will only need OpenVidu.
Euh ....

Where do I restart ??
++

#9 - 02/12/2024 03:17 PM - Josip Almasi

- Status changed from New to Closed

done long time ago