We can also publish built artifacts from GitLab somewhere that's easy for users to consume. For this, I'm using Cobalt Strike's Arsenal Kit as an example. For instance, a team may want to source control customisations to their kit(s) and publish complete builds for its members to pick up and use from a central location.

Rasta Mouse 👌 arsenal-kit					
A arsenal-kit ⊕ Project ID: 5 ि - ○ 1 Commit	Tags 🗔 2.4 MB Project Storage				
main v arsena	I-kit / + ~	Find file Web IDE 🖌 🗸 Clone 🗸			
Rasta Mouse authored ju	ist now	aee168dc [
README & Auto DevOps	enabled 🕑 Add LICENSE 🕀 Add CHANGELOG	G → Add CONTRIBUTING → Add Kubernetes cluster			
Name	Last commit	Last update			
🖹 kits	Initial commit	just now			
🗅 templates	Initial commit	just now			
M+ README.md	Initial commit	just now			
🕸 arsenal_kit.config	Initial commit	just now			
🔈 build_arsenal_kit.sh	Initial commit	just now			
🖹 releasenotes.txt	Initial commit	just now			

This can be built on a standard Ubuntu Docker image using the following:

image: ubuntu		
stages:		
- build		
build:		
stage: build		

```
variables:
output_path: "dist/artifact"
```

script:

- "apt -y update"
- "apt -y install mingw-w64 cifs-utils"
- "chmod +x kits/artifact/build.sh"
- "chmod +x build_arsenal_kit.sh"
- "./build_arsenal_kit.sh"

artifacts:

paths:

- "\$output_path/"

<pre>\$ chmod +x kits/artifact/build.sh</pre>
<pre>\$ chmod +x build_arsenal_kit.sh</pre>
<pre>\$./build_arsenal_kit.sh</pre>
/ / / //_/(_) /_
// /// _ / _ / _ `/ / .< / / _/
/ / / ()/ / / / / / / / / /_
Cobalt Strike Arsenal Kit
(c) 2022 HelpSystems LLC
[arsenal kit] [+] Building Artifact Kit
[Artifact kit] [+] You have a x86_64 mingwI will recompile the artifacts
[Artifact kit] [*] Using allocator: HeapAlloc
[Artifact kit] [*] Using STAGE size: 271360
[Artifact kit] [*] Using RDLL size: 5K
[Artifact kit] [*] Using stack spoofing technique
[Artifact kit] [+] Artifact Kit: Building artifacts for technique: pipe
[Artifact kit] [*] Compile src-main/resource.rc
[Artifact kit] [*] Recompile artifact32.dll with src-common/bypass-pipe.c
[Artifact kit] [*] Recompile artifact32.exe with src-common/bypass-pipe.c
[Artifact kit] [*] Recompile artifact32svc.exe with src-common/bypass-pipe.c
[Artifact kit] [*] Recompile artifact32big.dll with src-common/bypass-pipe.c
[Artifact kit] [*] Recompile artifact32big.exe with src-common/bypass-pipe.c

This will output everything to a new directory called **dist**.

However, specifying the output path under **artifacts** will tell GitLab to pull those out of the job and store them in the UI. This is also mandatory to ensure that the built artifacts are passed to any future jobs (such as deploy) without being deleted.

O passed Job #16 in pipeline #11 for f6f20f74 from main by Rasta Mouse 3 minutes ago				
Artifacts / dist / artifact	🕁 Download artif	acts archive		
Name	Size			
Ê				
🖹 artifact.cna	11.4 KB			

🖹 artifact32.dll	37.5 KB
🕒 artifact32.exe	40 KB
🖹 artifact32big.dll	304 KB
🕒 artifact32big.exe	306 KB
artifact32svc.exe	42 KB
artifact32svcbig.exe	306 KB
🕒 artifact64.exe	41.5 KB
🗈 artifact64.x64.dll	38.5 KB
🕒 artifact64big.exe	307 KB
🗈 artifact64big.x64.dll	304 KB
🕒 artifact64svc.exe	43 KB
🖹 artifact64svcbig.exe	307 KB

I shall now deploy these to an SMB file server, using the following addition to .gitlab-ci.yml:



• You cannot usually mount an SMB share from inside a Docker container. That's why we had to set the privileged setting to true in the runners.docker configuration.

There are several variables defined here, including the credentials to authenticate to the share. These should not be hardcoded in your YAML file, but rather defined inside the GitLab project. With the project, go to **Settings > CI/CD > Variables** and click the **Expand** button.

Here, you can click **Add variable** and provide any key/value pair that you need. These will then be passed down into the Docker runner as environment variables.

Variables

Variables store information, like passwords and secret keys, that you can use in job scripts. Learn more.

Variables can be:

- Protected: Only exposed to protected branches or protected tags.
- Masked: Hidden in job logs. Must match masking requirements. Learn more.

Environment variables are configured by your administrator to be protected by default.

Collapse

Туре	↑ Key	Value	Protected	Masked	Environments	
Variable	Domain [°	****************************	~	×	All (default)	Ø
Variable	PASSWORD	***************************	~	×	All (default)	Ø
Variable	SHARE_IP [**************************** [<mark>0</mark> 1	~	×	All (default)	Ø
Variable	SHARE_NAME	***************************** [<mark>0</mark> 1	~	×	All (default)	Ø
Variable	USERNAME 🖺	*****************************	~	×	All (default)	Ø

Add variable Reveal values

After running the pipeline, all the files are present.

PS C:\> ls -r \\172.23.156.140\artifacts\$\				
Directory: \\172.23.156.140\artifacts\$				
Mode	LastW	riteTime	Length	Name
d	17/11/2022	10:24		artifact-kit
Directory	: \\172.23.15	6.140\artifacts	\$\artifa	ct-kit
Mode	LastW	riteTime	Length	Name
-a	17/11/2022	10:24	11624	artifact.cna
-a	17/11/2022	10:24	38400	artifact32.dll
-a	17/11/2022	10:24	40960	artifact32.exe
-a	17/11/2022	10:24	311296	artifact32big.dll
-a	17/11/2022	10:24	312832	artifact32big.exe
-a	17/11/2022	10:24	43008	artifact32svc.exe
-a	17/11/2022	10:24	313344	artifact32svcbig.exe
-a	17/11/2022	10:24	42496	artifact64.exe
-a	17/11/2022	10:24	39424	artifact64.x64.dll
-a	17/11/2022	10:24	313856	artifact64big.exe
-a	17/11/2022	10:24	311296	artifact64big.x64.dll
-a	17/11/2022	10:24	44032	artifact64svc.exe
-a	17/11/2022	10:24	314368	artifact64svcbig.exe

You're obviously not constrained to SMB - you can transfer files over whatever means works best for you.