|  |
| --- |
|  |
| VCP toolbox  Git-Helper |
| Viacom content platform toolbox, part of the continuous delivery workflow  Engineering tool creation procedure  Content platform initiative  Viacom Bridge |

|  |
| --- |
| Content platform engineering team - Viacom  David Sorbona  April-2017  v1.0 WIP RFC#2 |

Contents

[Procedure to create the git-helper tool 3](#_Toc480468471)

[Setup the git-helper toolbox repository 3](#_Toc480468472)

[Create the git-helper container image with new identity and SSH keys. 5](#_Toc480468473)

[Push git-helper to container image registry 13](#_Toc480468474)

[Objective 13](#_Toc480468475)

[Pre-requirements 13](#_Toc480468476)

[Procedure 13](#_Toc480468477)

# Procedure to create the git-helper tool

This document is part of the continuous delivery workflow for the Content platform initiative / Viacom Bridge. You can find more information in the “VIACOM - Content Platform - Continuous Delivery Workflow” document.

Just for a knowledge preservation matter, we will describe the procedure that was used to create this tool. You don’t need to do it again since the container image is available in bitbucket and AWS ECR.

In summary, in the following steps we are going to create a container image to execute git commands against Bitbucket (stash) using only SSH protocol.

## Setup the git-helper toolbox repository

1. Create a repository in Bitbucket to host our git helper container dockerfile and dependencies.
   1. Open the VCP Bitbucket dashboard: <https://stash.mtvi.com/projects/VCP>
   2. Create new repository named "vcp-toolbox-docker-githelper"

Bitbucket Projects 
C 
Repositories 
Name 
Repositories 
vcp-aws-resourcedefinitions 
vcp-netcore-webappdem01 
vcp-toolbox-docker-githelper 

1. Clone the repository locally, execute this commands:

mkdir C:\Development\Viacom

cd C:\Development\Viacom\Viacom.ContentPlatform

git clone <https://sorbonad@stash.mtvi.com/scm/vcp/vcp-toolbox-docker-githelper.git>

1. Clone the “git-pusher” repository created by the M&E team

cd C:\Development\Viacom

git clone <https://sorbonad@stash.mtvi.com/scm/mdu/git-pusher.git>

WARNING

The git-pusher repository contains a bash script. I had to re-save the "scripts/add\_commit\_push.sh" to change the EOL for Linux (notepad++\edit\EOL Conversion\Unix, then save file) otherwise the script failed. Probably if you are in a MAC or Linux this is not needed.

File Edit Search Viev 
Windows (CR LF) 
unix (IF) 
Macintosh 
Undo 
Copy 
Delete 
Select All 
Begin/End Select 
Copy to Clipboard 
Indent 
Convert Case to 
Line Operations 
CommentAJncomment 
Auto-completion 
EOL Conversion 
Blank operations 
paste 
Ctrl + Z 
Ctrl* 
Ctrl.X 
Ctrl*C 
ctrl*V 
ctrl+A 
echo 
exit 
echo - 
exit 
$1 Il en 
if [ $ 

 We are going to use this repo as baseline to add our SSH keys and after that we are going to create our own "Docker based git helper"

1. Copy all files and folders from git-pusher to vcp-toolbox-docker-githelper

cd c:\Development\Viacom\git-pusher

xcopy \* /s C:\Development\Viacom\Viacom.ContentPlatform\vcp-toolbox-docker-githelper\

WARNING: Note that we are not copying hidden files, in this way I've explicitly avoided the copy of the ".git" folder of the "git-pusher" repo.

1. Delete your local copy of the repo "git-pusher"

cd C:\Development\Viacom\

Remove-Item -Recurse -Force .\git-pusher\

Now we are ready to create our own git helper container image.

## Create the git-helper container image with new identity and SSH keys.

The git-helper container will need a custom identity and a set of private/public SSH keys that will allow git to execute commands, against Bitbucket, from the Bamboo deploy pipeline, without providing user/password credentials.

Firstly, we are going to create new “identity file” and a pair of SSH keys. Secondly, we will use those artifacts to build our custom "git helper" container image that we will name as "vcp-toolbox-docker-githelper"

**Remember**: To speed up this process we will use some files from the "git-pusher" repository, a similar tool used by the M&E team.

Do you need additional info? Check those links:

* <https://stash.mtvi.com/projects/MDU/repos/git-pusher/browse>
* <https://confluence.atlassian.com/bitbucket/set-up-ssh-for-git-728138079.html>

1. Build a local Docker image using the dockerfile we've copied from the git-pusher repository

cd C:\Development\Viacom\Viacom.ContentPlatform\vcp-toolbox-docker-githelper

docker build -t vcp-toolbox-docker-githelper .

1. Open a command line interactive session to execute commands in the Docker container using the “vcp-toolbox-docker-githelper” image we've build in the previous step.

Machine generated alternative text:
Settings 
General 
Shared Drives 
Advanced 
Network 
Proxies 
Daemon 
Diagnose & Feedback 
Reset 
Docker is running 
Shared Drives 
Select the local drives you want to be available to 
your containers. 
Shared 
Microsoft PowerShell 
> docker run 
--rm 
Reset credentials... 
Drive 
-v alpine Is /data 
x 
x 
Apply WARNING:

If you are in windows, check that the "C" drive is shared in Docker. We need this to copy the new SSH files outside the container.

docker run -v C:\Development\Viacom\Viacom.ContentPlatform\vcp-toolbox-docker-githelper:/vcp-toolbox-docker-githelper -it vcp-toolbox-docker-githelper /bin/sh

1. Delete the existent identity and ssh keys stored in .ssh folder

cd /root/.ssh

ls

id\_rsa id\_rsa.pub known\_hosts

rm id\_rsa

rm id\_rsa.pub

**Result**: After that the /root/.ssh folder should contains only the known\_hosts file.

Full interaction:

Machine generated alternative text:
ps docker 
run -it 
git-push /bin/sh 
/ input # cd 
N/.ssh # 
N/.ssh # Is 
i d_rsa 
N/.ssh # 
N/.ssh # rm 
N/.ssh # rm 
N/.ssh # Is 
known _ hosts 
N/.ssh # 
/ root/ . ssh 
i d_rsa . pub 
i d_rsa 
i d_rsa . pub 
known _ hosts 

We are going to save the new SSH and identify files in this ~/.ssh folder.

1. Create our new default identity. For doing that we have to create a new identity and private/publish keys using the "*ssh-keygen*" command.

You can found more detailed information of this process the section "Setup SSH for Mac OS/Linux" here: <https://confluence.atlassian.com/bitbucket/set-up-ssh-for-git-728138079.html>

Machine generated alternative text:
YAtIassian Documentation 
Documentation / Bitbucket Cloud Documentation Home / 
Bitbucket Cloud Documentation Home 
Get started with Bitbucket Cloud 
Bitbucket Tutorials: Teams in Space Training 
Ground 
Bitbucket Cloud basics 
Bitbucket Cloud Teams 
Projects 
Plans and billing 
Manage an individual account or a team 
Log in to manage an individual account or a 
team 
Set up SSH for Git 
Set up SSH for Mercurial 
Two-step verification 
App passwords 
Manage groups 
Manage email notifications 
Set email preferences 
Set email aliases 
Associate an existing domain with an account 
Rename or convert an account 
Delete an account or a team 
Upgrade to Atlassian account 
Control access to your private content 
Use and administer repositories 
Add-ons, integrations, and services 
Search... 
Q 
Log in 
Manage an individual account or a team 
Set up SSH for Git 
If you came to this page because you don't have SSH set up, then you have been using the secure hypertext transfer protocol 
(HTTPS) to communicate between your local system and Bitbucket Cloud. When you use HTTPS, you authenticate (supply a 
username and password) each time you take an action that requires a connection with Bitbucket. Who wants to do that? This 
page shows you how to use secure shell (SSH) to communicate with the Bitbucket server and avoid having to manually type a 
password all the time. 
Set up SSH 
Setting up an SSH identity can be prone to error. Allow yourself some time, perhaps as much as an hour 
depending on your experience, to complete this page. If you run into issues, check out Troubleshoot SSH 
Issues for extra information that may help you along. You can even skip this whole page and continue to use 
HTTPS if you want. 
To use SSH with Bitbucket, you create an SSH identity containing a private key (on your local computer) and a public key 
(uploaded to Bitbucket) which create a key pair. After setting up SSH between your local system and Bitbucket, your system 
uses the key pair to authenticate you automatically to anything to which the associated account has access. 
For security reasons, we recommend that you generate a new SSH key and replace the existing key on your 
account at least once a year. 
There are a few important concepts you need when working with SSH identities and Bitbucket. 
• You cannot reuse an identity's public key across accounts. You must create SSH identities for each individual Bitbucket 
account. 
• You can associate multiple identities with a Bitbucket account. 
> Tell me why I would do that. 
RSA (R. Rivest, A. Shamir, L. Adleman are the originators) and digital signature algorithm (DSA) are key encryption 
algorithms. Bitbucket supports both types of algorithms. You should create identities using whichever encryption method 
is most comfortable and available to you. 
The following sections cover how to set up SSH for Git. 
> Set up SSH for Windows 
> Set up SSH for Mac OS/Linux 

Using the same interactive bash session we already have against our "vcp-toolbox-docker-githelper" container, lets execute:

ssh-keygen

When requested:

* "Enter file in which to save the key…" just press enter (we will use the default one)
* "Enter passphrase" just press enter (we are not going to use a key locked by a passphrase)

**Result**: The command creates two files, your default identity (id\_rsa) with its public SSH key (id\_rsa.pub).  Here the full interaction:

Machine generated alternative text:
N/ .ssh # ssh-keygen 
Generating public/ rivate rsa kev pair. 
Enter file in whic to save the key (/root/.ssh/id_rsa): 
Enter passphrase (empty for no passphrase) : 
Enter same passphrase again: 
your identifi cation has been saved in / root/.ssh/id_rsa. 
your public key has been saved in / root/ .ssh/id_rsa.pub. 
The key fingerprint is: 
SHA256: 36cwyqdZV3uk/Nr4BZYRStSLBCLz1WOUDOBwwvuVTPQ root@a0815a2cb6b9 
The key's randomart image is: 
2048]----+ 
o . E..o..o. 
CSHA256]- 
i d_rsa 
N/ .ssh # Is / root/.ssh 
i d_rsa . pub 
N/.ssh # 
known _ hosts 

1. Load our keys. This will add our new identity to the Linux container. Using the same interactive bash command line, execute:

ssh-agent /bin/sh

ssh-add /root/.ssh/id\_rsa

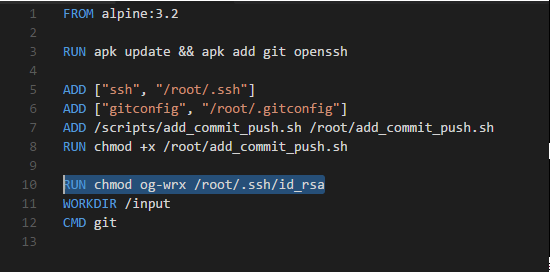
ssh-add -l

**Result**: after that you will see the key that the ssh-agent is managing. Here the full interaction:

/input # sshia ent /bin/sh 
/input # ssh-agd /root/ .ssh/id_rsa 
Identity added: /root/ .ssh/id_rsa (/root/ .ssh/id_rsa) 
/ input # ssh-add -l 
2048 Idgsrzl /root/ .ssh/id_rsa (RSA) 

WARNING

* If you don’t see that SHA256 key, then stop here and review your steps.
* The id\_rsa file must have the correct POSIX permissions, otherwise it won’t work. This is defined in the dockerfile. Make sure the “*chmod og-wrx /root/.ssh/id\_rsa”* command is present:



1. Now we have to install the public key in our Bitbucket account. Using the same interactive bash command line, execute

cat /root/.ssh/id\_rsa.pub

**Result**: the command will show the public key as a string.

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC97JeS7d6k/LZVShmVpxpt6KL4mycXwjasQHxqkiLTusTLcARmrkWPecplc9sv8YVCCmrLBWsQasxVIRQp6YYN/Xhsno5FbCp+K36vZRQezOPNwoCrLMwq9F6qQEQx/AbNw8Vb607G9taYXnZ2lJLF/qFjFFyWE0VuLIqqFjOhxdpW3/wpzf91S66oD8whVmmIa12m8hAB4PBz1YNYlunRURF2D0C+Ijhm/uE3TMuS72WRKP1dw6RcLkcskdH7MRYGkKCoeqHWj4CV5SaSiI8NiCqtuCG7c9eZmDbMuZAHWOTermegCCAlVNSxSF4j2Iib6C/3TbGpKjLM7j4lhDvJ meenaksa@NY-MEENAA-MV

**CONTEXT: Why are we doing this?**

If the Bitbucket server/repository knows our new public key, then the git client will be able to encrypt and sign the communication between the "git-helper container" and the Bitbucket server, therefore there will be no need to provide (or hardcode) a username/password to execute git commands. Bitbucket will also know who is sending the command since the communication will be signed with our private key.

1. Copy they public key hash string to the clipboard and go to the "vpc-aws-resourcedefinitions" repository in Bitbucket:

<https://stash.mtvi.com/projects/VCP/repos/vcp-aws-resourcedefinitions>

WARNING: Verify that the string has no new lines or additional empty spaces that could have been added during the copy/paste operation.

1. Go to “settings \ access keys” and click on "Add key"

C 
19 
Settings 
Repository details 
Web Pages 
SECURITY 
Repository permissions 
Branch permissions 
Access keys 
Audit log 
Subversion Mirror 
WORKFLOW 
Hooks 
Pull requests 
HipChat integration 
Branching model 
Script Pre Hooks 
Access keys Add key 
Access keys provide a simple way for other systems to access repositories using SSH without storing user 
credentials. 
Label 
https://bam... 
https://bam... 
Key 
ssh-rsa AAAAB3NzaC1 yc2EAAAADAQABAAABAQDNnvRxH... 
ssh-rsa AAAAB3NzaC1 yc2EAAAADAQABAAABAQC5USNna.. 
Bitbuckefs RSA fingerprint: 
Permission 
Read / Write 
Read / Write 

1. Set as read/write, paste the public key hash and click on “Add key”

Add public key 
Permission 
Key 
C) Read 
@ Read / Write 
ssh-rsa AAAAB3NzaC1yc2EAAAADAQAEAAAPAOC97JeS7d6k/LZVShmVpxpt6KL4mycXwiasQ 
6vZR0ezo 
OhxdpW3 /wpzf9 
1 g 6 EaD8whVmm1a12mdhAg4PPz1 uE3TMuS72WRKP1dw6RcLkcskdH7 
MRYGkKCoeqHWj 4 cvssasi18NicqtuCG7c9eZmDbMuZAHROTermegccAIVNsxSF4j21ib6C/3T 
bGpKjLM7j41hDvJ meenaksaßNY-MEENß„A-MV 
Read our help doc for creating SSH keys 
Add key 
Cancel 

1. Now we have to test if this works. Let's clone the vcp-aws-resourcesdefintion inside the container (use the SSH repo URL variant). Then let’s create a dummy file and finally let’s commit and push the changeset. If it works as expected, you should see the dummy file in the Bitbucket repository. Here the full interaction using the same interactive bash session against the "vcp-toolbox-docker-githelper" container.

cd /input

git clone **ssh**://git@stash.mtvi.com/vcp/vcp-aws-resourcedefinitions.git

cd vcp-aws-resourcedefinitions/

echo "hi world" >> ./test.txt

git add .

git commit -m "Hello world test"

git push

In the Bitbucket repo we should see the file:

g Bitbucket Projects 
Viacom content platform 
Source 
master 
Repositories Pull requests 
/ vcp-aws-resourcedefinitions 
vcp-aws-resourcedefinitions 
Search for code or repositories... 
Browse 
Filter 
vcp-netcore-webappdem01 
[i] test.txt 
Hello world test 
5 mins ago 

In addition, if you open the file you will see that the committer is "Mr Robot". That name was defined in the "gitconfig" file of our repository (copied from the M&E git-pusher tool). I like the name so we will keep it.

Machine generated alternative text:
Viacom content platform / vcp-aws-resourcedefinitions 
Source 
master 
Source view 
hi world 
1 
vcp-aws-resourcedefinitions 
test.txt 
Mr Robot committed 58436cc947f 6 mins ago 
Diff to previous 
History v 
Blame 
Raw file 

gitconfig 
[user] 
name Mr Robot 
email mr.robot@viacom.com 
4 (push] 
default simple V This pc 
Windows (C:) Development Viacom git-pusher 
084pr.17 5:36 
17 5:36 
Od4pr.17 5:39 
OB-Apr-17 5:36 
QB-Apr-17 5:35 
Name 
scripts 
Dockerfile 
g itconfig 
read m end 
Date modified 

**NOTE**

We can execute the same test with the bash script "/root/add\_commit\_push.sh" that is provided by the "git-pusher" container image. Here an example:

cd /input

git clone ssh://git@stash.mtvi.com/vcp/vcp-aws-resourcedefinitions.git

cd vcp-aws-resourcedefinitions/

echo "hi world" >> ./test.txt

/bin/sh **/root/add\_commit\_push.sh** /input/vcp-aws-resourcedefinitions hi-world-3 ssh://git@stash.mtvi.com/vcp/vcp-aws-resourcedefinitions.git

The command will execute 3 operations in one: add + commit + push.

Viacom content platform I vcp-aws-resourcedefinitions 
Commits 
master 
Author 
All branches 
Commit 
Message 
hi-world-3 
Commit date 
24 seconds ago 
Issues 
Builds 
Mr Robot 
4e95e4fc200 

1. Copy the Identity and SSH keys outside the container:

cd /root/.ssh

cp \* -f /vcp-toolbox-docker-githelper/ssh/

**REMEMBER**: The **/vcp-toolbox-docker-githelper** folder is a mounted directory that points to your local file system outside the container

**Result**: The new identity and SSH keys were saved in the path:

C:\Development\Viacom\Viacom.ContentPlatform\vcp-toolbox-docker-githelper\ssh

1. Update the readme file "C:\Development\Viacom\Viacom.ContentPlatform\vcp-toolbox-docker-githelper\readme.md"

Viacom content platform toolbox 
## Docker git helper 
April 2ø17 
Contains git inside, allows docker workers make git operations 
Based on project "git-pusher" : 
# scripts 
- allows add files inside target dir (first param), commit changes with commit message 
(second param) and push to remote url(third param) 

1. Commit the changes of the "vcp-toolbox-docker-githelper" repo and push them to Bitbucket

cd C:\Development\Viacom\Viacom.ContentPlatform\vcp-toolbox-docker-githelper

git add .

git commit –m “version ready”

git push

1. Done!

# Push git-helper to container image registry

## Objective

The objective in this section is to publish our git-helper container image to the Viacom AWS ECR instance

## Pre-requirements

In order to publish a container image to AWS you will have to use the AWS command line client and use temporal IAM credentials.

* Aws command line client installed: <https://aws.amazon.com/cli/>
* Local clone of the repo “vcp-toolbox-docker-githelper”
* Python 2.7 installed: <https://www.python.org/downloads/>

## Procedure

1. Create a new container registry in the Viacom private instance of AWS ECR (non-prod / us-east-1)

Name: **viacomcontentplatform/vcp-toolbox-docker-githelper**

Services v 
Resource Groups 
VMNDevOps/david.sorbona@vi... 
N. Virginia 
Support 
Get started with EC2 Container Registry 
I Step 1: Configure repository 
Step 2: Build, tag, and push Docker image 
Configure repository 
This wizard will guide you through the steps of creating a repository in EC2 Container Registry. Learn more 
Repository name* 
Repository URI 
Permissions 
orm/vcp-toolbox-docker-githel 
o 
Namespaces are optional, and they can be included in the repository name with a 
slash (for example, namespace/repo) 
310827469077.dkr.ecr.us- 
east-1.amazonaws.com/viacomcontentplatform/vcp- 
toolbox-docker-githelper 
As the owner, you have access to this repository by default. After completing this wizard, you can grant others 
permission to access this repository in the console. 
*Required 
Cancel 
Next step 

**Result:** you should see the aws instructions to build, tag and publish your image. Keep this information handy.

Get started with EC2 Container Registry 
Step I : Configure repository 
Build, tag, and push Docker image 
I Step 2: Build, tag, and push Docker image 
NOW that your repository exists, you can push a Docker by following these steps: 
Successfully created repository 
310827469077. dkr.ecr. us-east, I. 
To install the AWS CLI and Docker and for more information on the steps below, visit the ECR documentation page. 
I) Retrieve the docker login command that you can use to authenticate your Docker client to your registry: 
awg ecr get—login 
region us—east—I 
2) Run the docker login command that was returned in the previous step. 
3) Build your Docker image using the following command. For information on building a Docker file from scratct-l see the instructions here. 
You can skip this Step if your image already built: 
docker maild —t 
4) After the build completes, tag your image so you can push the image to this repository: 
docker tag 31082746903 Q .dkr.ecr .us— 
east -I 
5) Run the following command to push this image to your newly created AWS repository: 
docker push 
docker—qithelper: la test 
*Required 

1. If you don’t have it, clone the repository “vcp-toolbox-docker-githelper” locally.
2. If you don’t have it, build the Docker image of the “vcp-toolbox-docker-githelper”

cd C:\Development\Viacom\Viacom.ContentPlatform\vcp-toolbox-docker-githelper

docker build -t vcp-toolbox-docker-githelper .

1. If you have never used the aws command line, we have to configure it first, so execute this command

aws configure

The command will request the following parameters:

* + AWS Access Key ID [None]: leave it empty
  + AWS Secret Access Key [None]: leave it empty
  + Default region name [None]: us-east-1
  + Default output format [None]: json

**IMPORTANT**

The “access key id” and “secret key” are temporal values that are retrieved on demand based on our AD credentials (SSO integration). To get them we have to execute a python script created by the MCS team.

The Next steps explain how to execute the script. If you need more info, go to check these links:

<https://confluence.mtvi.com/display/ISG/Using+the+AWS+Command-Line+Interface>

<https://confluence.mtvi.com/display/ISG/AWS+Knowledge+Base>

1. If you don’t have the aws-login.py script needed to get temporal credentials to interact with AWS, follow these steps:
   1. Download aws-login.py script from here:

<https://stash.mtvi.com/projects/ISG/repos/aws/browse/tools/aws-login.py>

* 1. Install pip (the package manager for python)
     1. Download this file <https://bootstrap.pypa.io/get-pip.py>
     2. Then in cmd (as Admin) execute:

python get-pip.py

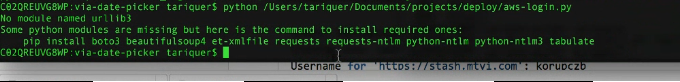
More info: <http://stackoverflow.com/questions/4750806/how-do-i-install-pip-on-windows>

* 1. Install script dependencies (as admin)

pip install boto3 beautifulsoup4 et-xmlfile requests requests-ntlm python-ntlm python-ntlm3 tabulate urllib3

WARNING

Python will show you the complete list of dependencies to install after executing the aws-login.py script. Re-execute pip install if needed to install missing dependencies:



1. Run the aws-login.py script and provide the information required:

python.exe .\aws-login.py

Full interaction

ps c: folder> python-exe .\aws-login.py 
Hint: set AD_USERNANE to Domain\username to get rid of the username prornpt 
Domai n\Username: pl ayasur\sorbonad 
password: 
Assertion not found, trying form login... 
Fetching account names... 
please choose the role you would like to assume: 
account i d account name 
role 
0 124619196548 vmn -ea -nonprod 
VMNDevops 
1 310827469077 vmn-pl atform-nonprod VNNDeVOps 
2 627274808695 vmn—pl atform-prod 
VMNDevops 
Selection I 
credentials for account vmn-platform-nonprod role VMNDevops stored under saml profile, 
You need to call aws --profile saml or export in your .bash_profile 
credentials stored to c: .aws/credentials 
ps c: folder> cd —/.aws 
ps c: dir 
Di rectory: c: 
expires at 2017-04-09 
Mode 
Lastwri teTime 
09-Apr—17 
09-Apr-17 
1:33 AM 
1:25 AM 
2:30 AM 
Length Name 
3510 
awstest . PSI 
config 
46 
credentials 
1684 

WARNING: In MacOS the script has to be executed with SUDO

**Result:** This script creates temporal AWS keys stored in the ~/.aws/credentials file

More info: <https://confluence.mtvi.com/display/ENT/How+to+Setup+Native+Docker>

1. Push the “vcp-toolbox-docker-githelper” image to the container registry “viacomcontentplatform/vcp-toolbox-docker-githelper” created previously. To do that execute the following command:

aws ecr get-login --region us-east-1 --profile saml| iex

IMPORTANT

The result of “aws ecr get-login” is a docker command returned as string, so we need to “pipes” the string to the Interpret-Expression cmdlet. The pipeline commands powershell to redirect the output of a command as the input for another command. In Linux and MacOs a similar command would be ($aws ecr get-login --region us-east-1 --profile saml)

WARNING

For some OS configurations the modifier “--profile saml” is not needed.

After that execute the following commands to TAG and PUSH the image:

docker tag vcp-toolbox-docker-githelper 310827469077.dkr.ecr.us-east-1.amazonaws.com/viacomcontentplatform/vcp-toolbox-docker-githelper:latest

docker push 310827469077.dkr.ecr.us-east-1.amazonaws.com/viacomcontentplatform/vcp-toolbox-docker-githelper:latest

**NOTE**

The docker tag and push parameters were provided by the AWS ECR web portal. You can see it again thru the "View Push Command" button available in the AWS ECR container image page.

**Result:**

PS docker push 310827469077.dkr.ecr.us-east-l.amazonaws.com/viacomcontentplatform/vc 
The push re rs to a repository (3108 74 9077.dkr.ecr.us-east-l.amazonaws.com/viacomcontentplatform/vcp-toolbox-docker-githelper 
-toolbox-docker-githelper: latest 
3322c787a35b: pushed 
a6d99e8323d3: pushed 
d25b2b4beIa2: pushed 
3a9f554b16S9: pushed 
1b96213e7b2b: pushed 
IOf4bfaa902S: pushed 
3d0207fSdcff: pushed 
latest: di est: sha756:20937a6a64cb4272f40dhef2e7cb67egbcgd842a55fcf13e18dId12edd381hf3 size; 
1983 

In the AWS ECR repository you should see a new entry:

All repositories : viacomcontentplatform/vcp-toolbox-docker-githelper 
Repository ARN 
Repository URI 
:310827469077: repository/viacomcontentplatform/vcp-toolbox-docker-g ithelper 
310827469077.dkr.ecr.us-east-I.amazonaws.com/viacomcontentplatform/vcp-toolbox-docker-githelper 
View Push Commands 
Images 
Permissions 
Amazon ECR limits the number of images to 1,000 per repository. Request a limit increase. 
Image sizes may appear compressed. Learn more 
Delete 
Filter in this page 
Image tags 
Last updated on April 9, 2017 AM (0m ago) 
Tag status: 
All 
view al 
Digest 
Size (MiB)... 
13.34 
1-1 > page size 100 
Pushed at 
2017-04-09 02:5820 -0300 

1. Done!