Migrate site from aws to bunny.net

Moving websites away from aws S3 and cloudshare To bunny.net

My Makefile generates all the site content from restructuredText markup files to a local directory on my laptop. For the amazon aws hosted site, I had 3 simple bash scripts to publish that content. One "deploy.aws" syncs all the content to an s3 Bucket. One "invalidate.sh" invalidates the files in cloudshare cache that are now changed, and "wait.sh" which blocks until the invalidation has completed.

To make the transition from AWS to Bunny.net, I only need to implement alternate versions of deploy scripting. My entire blog maintenance work cycle remains the same.

Set up bunny.net account and resources to service the site

- First I obtained a bunny.net account, and logged into that.
- I created and named (e.g blog-example-com) a standard storage zone.
- I created and named (e.g. blog-example-com-zone) a pull zone (under CDN tab) associated that with the storage zone name. I also added a hostname to the pull zone. It must correspond to my existing site domain name (e.g blog.example.com).
- Using my current Domain name service (in my case squarespace) I need to replace the old CNAME record that points to my old aws s3 bucket, with something like this:

  blog.example.com CNAME log-example-com-zone.b-cdn.net

Deploy scripting

Every time I modify, remove or add content during my blog maintenance work cyle, I need to publish those changes to my bunny.net storage zone, and purge the pull zone to push out the changes to the CDN.

- To do that I needed to create a new deploy script called **deploy.bunny**.
- For my script to work I have to setup credentials on my laptop so I wrote another short script that does that called **login.bunny**. It only needs to be invoked once. Afterwards the creditials will reside in $HOME/.bunny/credentials.

deploy.bunny

```
#!/usr/bin/env bash
#
#
set -euo pipefail

MYPNAME=$(basename ${0})
ARG1NAME=${1:-NOARG}

# --- Configuration ---
STNAME="blog-bernatchez-net"
LOCDIR="$(pwd)/bucket"
PULLZONENAME="blog-bernatchez-net-zone"
# --- End Configuration ---
echo MYPNAME $MYPNAME
echo LOCDIR $LOCDIR
echo STNAME $STNAME
echo PULLZONEID $PULLZONEID
echo PULLZONENAME $PULLZONENAME
```

```
# 1. Sync up to bunny zone
echo ${MYPNAME}: npx --yes bunny-transfer@latest sync ${LOCDIR} ${STNAME}
npx --yes bunny-transfer@latest sync ${LOCDIR} ${STNAME}
#  use move instead of sync if you dont want --delete

# 2. purge invalidates all cached objects
echo ${MYPNAME}: npx --yes bunny-transfer@latest purge $PULLZONENAME
npx --yes bunny-transfer@latest purge $PULLZONENAME

echo ${MYPNAME}: "Deployment complete."
```

deploy.login

```
#!/usr/bin/env bash
#
#
set -euo pipefail

MYPNAME=$(basename ${0})
ARG1NAME=${1:-NOARG}

# --- Configuration ---
STNAME="blog-bernatchez-net"
PULLZONENAME="blog-bernatchez-net-zone"
APIKEY="xxxxxxxx-xxxx-etc"
# --- End Configuration ---
echo MYPNAME $MYPNAME
echo STNAME $STNAME
echo PULLZONENAME $PULLZONENAME
echo APIKEY $APIKEY

# login
echo $MYPNAME: Login creates the file "~/.bunny/credentials"
echo $MYPNAME: Until you do that, your other bunny commands wont work
echo $MYPNAME: npx --yes bunny-transfer@latest login default $APIKEY
npx --yes bunny-transfer@latest login default $APIKEY

# verify
echo $MYPNAME: This verifies we have got it right
echo $MYPNAME: npx --yes bunny-transfer@latest who-am-i
npx --yes bunny-transfer@latest who-am-i
```

Final touch

To incorporate the deploy.bunny script into my workflow, I added an invocation of it to the end of the "pubhtml" target of my makefile. That way it gets invoked whenever I regenerate the content for publication.