Debug using `with-ssh` for Github Actions

Context

`with-ssh` is a feature for Github Actions that was created to replicate the features set of CircleCI's *Re-run with SSH* feature that a lot of the development team is used to.

Platform availability:

☒ Linux (https://github.com/pytorch/pytorch/pull/62280)
☒ Windows (https://github.com/pytorch/pytorch/pull/63440)

Comparisons to CircleCI's Re-run with SSH

Similarities

- 2 hour time limit after job has finished for when SSH sessions will be closed out and runner will return to the regular runner pool
- Public keys for ssh are pulled from Github using `https://github.com/${github.actor}.keys`
  - Example: https://github.com/seemethere.keys

Differences

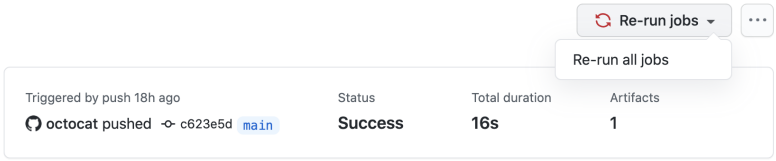
- Only works for users who are connected to the Meta VPN
- Requires a label (*with-ssh* in this case) to be applied to the Pull Request
  - As opposed to clicking *Re-run with SSH* through the CircleCI UI
  - **NOTE:** SSH keys will not be added to jobs ran *before* the label is applied so workflows will need to be re-ran *after* the *with-ssh* label has been applied
    * This is true even if the job is re-run using GitHub's "re-run jobs" button
    * **A completely new workflow run needs to be triggered after the label is applied**
- Only works for `pull_request` events and will not work on *main branch* `push` events

Known limitations

- Only works for users who are connected to the Meta VPN
  - No current planned support for outside collaborators yet

Workflow for users

1. Label your PR with the *with-ssh* label
   1.
2. Push a new commit / re-run completed workflows, see below for re-running jobs through the Github UI



1.

3. Traverse to logs for a `build` or `test` job that runs the `add-github-ssh-key` step added (currently all of our linux workflows have this enabled)
    1.
4. Use the SSH command provided to log into the node (do this immediately, as the job will start cleaning up if it reaches the end without any active SSH session):
    1.

Notes for users

General

- The default timeout for these jobs is **2 hours after workflows have completed**, but only if you SSH in before the end of the job
    - Users will be kicked after workflows have either *timed out* or have been *cancelled*

VSCode

VSCode's remote SSH extension works with both Linux and Windows CI machines. Add them as you would any other SSH remote, with `cmd + shift + P > SSH: New Remote` and use the address `runneruser@https://ec2....compute-1.amazonaws.com` as provided to you in the GitHub CI logs. On Windows you can directly open VSCode to the `pytorch` folder with the command:

```
[your mac] $ export HOST=runneruser@ec2-3-238-198-8.compute-1.amazonaws.com
[your mac] $ code --folder-uri vscode-remote://ssh-remote+$HOST/c:\\actions-runner\\_work\\p
```

Linux

Once you are connected through ssh, you may need to enter a docker container. Run `docker ps` to check if there are any docker containers running. Note that your CI job might be in the process of initiating a docker container, which means it will not show up yet. It is best to wait until the CI job reaches a step where it is building pytorch or running pytorch tests. If the job does have a docker container, run `docker exec -it IMAGE_ID /bin/bash` to connect to it.

Now you can find the pytorch working directory, which could be ~/workspace or ~/project, and run commands locally to debug the failure.

Windows

- To set-up the same CI environment run: `C:\actions-runner\_work\pytorch\pytorch\build\win_tmp\`
- The *Windows* workspace is currently located at `C:\actions-runner\_work\pytorch\pytorch`
- To use other shells for *Windows* just append the shell you'd like to run to your ssh command like:
  - `ssh runneruser@ec2-3-238-136-38.compute-1.amazonaws.com -- bash.exe`
- To run a particular test use the Miniconda Python: `C:\jenkins\miniconda3\python.exe test_profiler.py`

RDP

For certain Windows failures, it may be useful to have a full Remote Desktop connection. To use remote desktop, get a RDP client (Mac App Store) and: 1. The Windows EC2 machines only allow SSH connections on port 22, so make a tunnel for the RDP port (default 3389) to your local machine and set a password:

```
# change the EC2 hostname to the one given to you in the GitHub Actions logs
[your mac] $ ssh -L 3389:localhost:3389 runneruser@ec2-3-238-198-8.compute-1.amazonaws.com

# set a password for the 'runneruser' account (it must have upper
# and lowercase letters, a number, and be at least 8 characters)
[windows] $ net user runneruser <some password>
```

2. Open the RDP client, add a new PC with the hostname `localhost`
3. Connect to it by double clicking, use the username `runneruser` with the password you just created. You should see the remote desktop window open.

- **Old reference**: Detailed instructions for debugging Windows with SSH on CircleCI.

(For META Employees) Debugging using AWS SSM

AWS SSM can be used to log into *any* currently running EC2 instances whether or not your SSH key has been added to the particular instance.

This can be used to debug currently running jobs on PRs or trunk.

Pre-requisites

- Network connection on META VPN
- Cloud SSO Access (`bunnylol cloud fbossci`)

Usage

```
aws ssm start-session --target "<instance_id>" --region "<region>"
```

Example:

```
aws ssm start-session --target i-0099d5a07d34e8904 --region us-east-1
```

Caveats

- Nodes will still be reaped on their regular schedule
- Jobs will not wait for your session to finish before concluding
- There is no extra hold on the machine and the 2 hour timeout does not apply here, sessions will most likely end immediately at the next reap cycle