Tectonic: An Academic Cyber Range

Cheatsheet

Tectonic is a cyber range designed to provide realistic cybersecurity scenarios for education and training through the deployment of networks, systems and applications that can be used to train users on cybersecurity topics. Key functionalities include customizable network configurations, real-time monitoring and automated attack simulations.

Scenario Management

Scenarios are defined using a scenario description yml file (usually `description.yml` inside the scenario directory) plus a lab edition file (usually `<lab_name>.yml`.

- Create base images:

  `tectonic -c ~/tectonic.ini <lab_edition_file> create-images`

- Deploy scenario:

  `tectonic -c ~/tectonic.ini <lab_edition_file> deploy`

- Destroy scenario [and base images]:

  `tectonic -c ~/tectonic.ini <lab_edition_file> destroy [--images]`

- Show cyber range information (access IP addresses, credentials):

  `tectonic -c ~/tectonic.ini <lab_edition_file> info`

Operations on machines

Operations done on machines in the scenario, after it is deployed.

- Get a console on a *single* machine:

  `tectonic -c ~/tectonic.ini <lab_edition_file> console <machine_spec>`

- Reboot, shutdown or start machines in the scenario:

  `tectonic -c ~/tectonic.ini <lab_edition_file> [reboot|shutdown|start] <machine_spec>`

- Recreate machines (go back to the initial state):

  `tectonic -c ~/tectonic.ini <lab_edition_file> recreate <machine_spec>`

- Run an arbitrary ansible playbook:

  `tectonic -c ~/tectonic.ini <lab_edition_file> run-ansible -p <playbook> <machine_spec>`

Machine names Machines in the cyber range are identified as follows:

`<institution>-<lab_name>-<instance>-<guest>[-<copy>]`

The copy value is optional and only appears in the name if there is more than one copy for the same guest.

For example, copy 2 of the `server` guest of instance 3 of the lab `test_lab` and institution `test_inst` is:

`test_inst-test_lab-3-server-2`

As another example, the attacker guest, which consists of a single copy, of instance 2 of the lab test_lab and institution test_inst is:

`test_inst-test_lab-2-attacker`

Machine specification Most commands accept machine specification options, which can be a combination of: instance number (`-i`), guest (base) name (`-g`), and copy number (`-c`).

For example, to reboot all copies of the machine `victim` of instances 3 and 4, one can run:

```
tectonic -c ~/tectonic.ini <lab_edition_file> reboot -g victim -i 3,4
```

this will reboot machines `test_inst-test_lab-3-victim` and `test_inst-test_lab-4-victim`.

Instance and copy numbers can be specified either as a list: `1,2,3`, as a range: `5-10`, or as a combination: `2,4-6,8`.

Connectivity to the scenario

- Teacher access:
  Use `tectonic console`, or connect through `ssh`:

  `ssh -J ubuntu@<teacher_access_ip> <machine_ip>`

  `teacher_access_ip` is shown after scenario deployment and in the output of `tectonic info`.

- Student access:

  `ssh -J traineeXX@<student_access_ip> <entry_point_ip>`

  `student_access_ip` is shown after scenario deployment and in the output of the `tectonic info` command.

  By default student usernames are of the form `traineeXX`, where `XX` is the instance number. Credentials can be either SSH public keys, generated passwords or both.

  Only machines declared as entry points are accessible to the students.

- Copy files to/from machines:
  Use the `-J` option to `scp` in the same way as above.

  ```
  scp -J traineeXX@<student_access_ip> <source> <dest>
  ```

Port forwarding It is possible to forward ports to access services withing the scenario. To do that, use the `-L` option to `ssh`:

```
ssh -L localhost:<local-port>:<remote-ip>:<remote-port> <ssh-connection-options>
```

where `<ssh-connection-options>` connects to the scenario either as student or teacher, as above.

For example, to forward local port 80443 to port 443 on machine 10.0.1.5, use:

```
ssh -L localhost:80443:10.0.1.5:443 <ssh-connection-options>
```

You can then connect to localhost:80443 to access port 443 on machine 10.0.1.5.

File edition

For editing files within a scenario, you can use a console based text editor or run locally a text editor that supports remote connections, such as VSCode. Using the above ssh connection commands, VSCode can edit remote files, open a console and configure port forwardings. See https://code.visualstudio.com/docs/remote/ssh for more details.