Dissecting an Untracked Policy

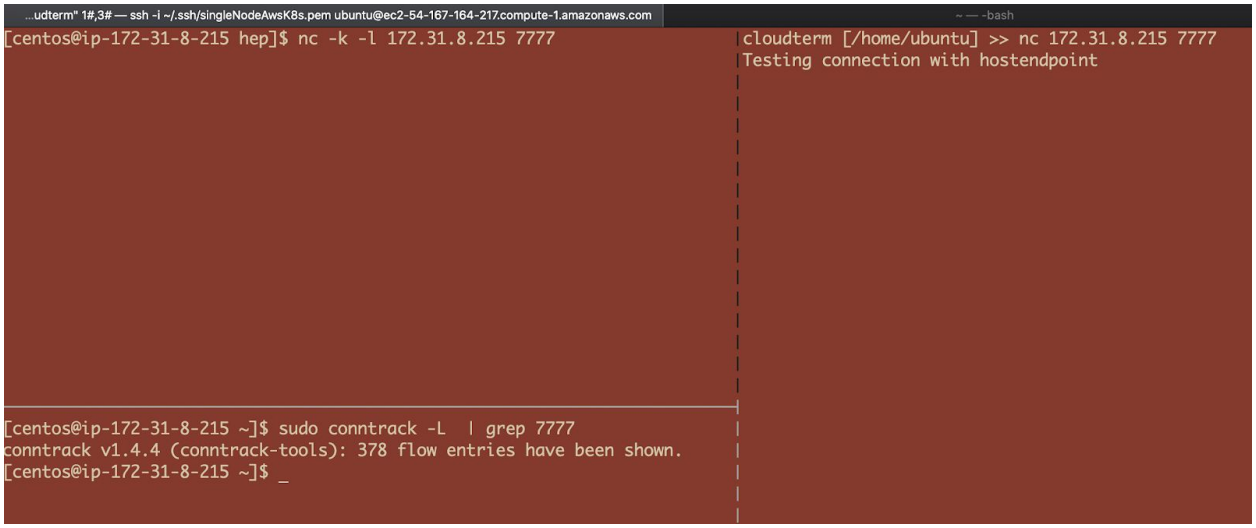
Untracked policy is very simple, yet can be deceiving. Let us start with a clean slate and trace a simple untracked policy. The files referred in this example are in this calico policies by example repo. We will use ncat (nc) command for client/server connections.

Objective is to create an untracked policy to permit traffic into port 7777 of the host. We will verify that conntrack is not created when untracked policy is in use.

First, start with a clean slate. As you can see below, nc is listening on port 7777 on left, and nc client can connect (on the right). Also the conntrack table creates an entry (left bottom).

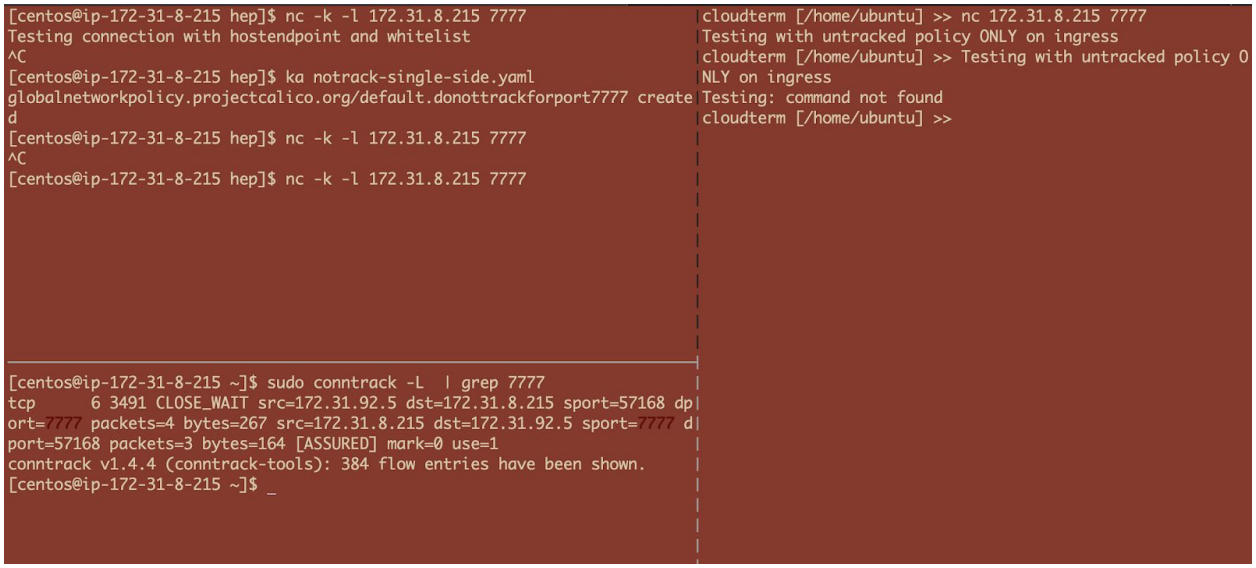
It takes about 120 seconds (default) for conntrack entry to expire after the connection has been terminated. As part of the next step, let's create a calico host endpoint. Note: Creating a host endpoint (if you follow the example below) will make your host network blind, except for the fail-safe ports in Calico. So do not attempt to try this exercise on a production system.

Apply the configuration (kubectl apply -f hep0.yaml) hep0.yaml. Note that you will need to customize the configuration per your host specification. Now let us see the behavior.



So there's no connection! This is expected, as the hostendpoint creates a default-deny policy for all traffic into the host. Let us whitelist all traffic (apply manifest whitelist.yaml) into the host. This will allow us to move to an untracked policy next.

Now we are ready to create an untracked policy on host endpoint. Apply the policy untrack-singleside.yaml. It basically applies the untracked policy in a single direction (permit into port 7777 of the host).



So it did not work! And there's some unexpected behavior. Conntrack table shows an entry stuck in CLOSE_WAIT mode. What actually happened here? Looking at the packet processing chain diagram above, the incoming flow bypassed the conntrack (because of untracked policy), but the return flow (in OUTPUT chain) hit the conntrack. And conntrack did not find any matching connection for the response packet. We do not want this kind of behavior. Hence it is important to test and deploy untracked policies properly.

Let us now deploy the untracked policy on both sides (apply file). This applies untracked allow policy to and from port 7777 on the host.

As you can see above, the traffic went through just fine. And no conntrack entry was created.

Finally, let us see how to navigate the untracked policy in the iptables itself. We know that it's hooked into raw table in PREROUTING chain. So start traversing until you hit the policy itself. The following diagram shows the snapshot of starting from the root upto the policy.

Untracked policies can be very powerful, if used properly.

This completes the review of untracked policy in calico.