1. Suppose you have a firewall that's supposed to block SSH connections, but instead lets them through. How would you debug it?

2.
    - Yes, in Project 1, I did allow SSH traffic to all of the VMs on my network.
    - My Jump-Box-Provisioner VM accepted SSH connection from my computer's IPV4 address (174.104.223.176). My Web-1 (10.0.0.5), Web-2 (10.0.06), and Elk1-VM VMs accepted SSH connection from the Ansible container within my Jump-Box-Provisioner VM.
    - If we try to connect to a VM that does not accept SSH connections, we will receive a "connection refused" error message because the VM's port 22 is either closed (stopped on Azure lab) or is being blocked (firewall).

3.
    - I would assume the source of the error is the Network Security Group.
    - I would double-check the Inbound Security Rules.
    - I would attempt to SSH into the VM thorough terminal to test if my configurations are effective.

4.
    - To investigate the problem, I would specifically look at Inbound Security Rules within Network Security Group.
    - I would check Action, Port, and Priority.
    - I would specifically look for whether the connection is allowed under Action, which Port (22, 80, etc.) is used, if the priority of an inbound rule is relatively higher or lower than the rule meant to block inbound SSH connections (should be higher because we want to deny SSH connection).
    - I would connect to my VMs by running SSH in the terminal to test that my fix is effective.

5.
    - No, this solution does not guarantee that the Project 1 network is now immune to all unauthorized access (phishing emails, drive-by-downloads, etc.); however, it should guarantee that the Project 1 network is immune to all SSH attempts.
    - I might add a detection/monitoring control and use a program such as Checkmk or Languardian to ensure I identify any suspicious authentication attempts.