Wiz Technical Exercise Intro / Overview

Welcome to the Wiz Technical Exercise! This is your opportunity to demonstrate compelling technical proficiency as well as your domain expertise in modern DevOps practices, cloud architecture, and cybersecurity.

We will provide the detailed requirements below, but broadly you are being asked to:

1. Deploy a three-tier web application using several cloud services with intentional configuration weaknesses and to demonstrate detection and remediation of these weaknesses with a CSP security tool (**this is a key element of the exercise**)
2. Leverage modern DevOps tools/best practices to automate the building of your cloud infrastructure and application deployment
3. Implement technical security controls before your application reaches production. These controls may cover a variety of different categories, such as compliance, audit, or security monitoring.

You can choose any of the "Big 3" CSPs that you are most comfortable/familiar with (either AWS, GCP, or Azure).

The Presentation

You will be presenting your solution (leveraging the provided template) to 2-3 Wiz panelists via Zoom. The panel will assess what you built, your methodology, challenges faced, your insights on the environment's security & the quality of your overall presentation. Your approach, the challenges faced, and your solutions will be essential components for the presentation portion of this task, along with your ability to effectively communicate the technical details.

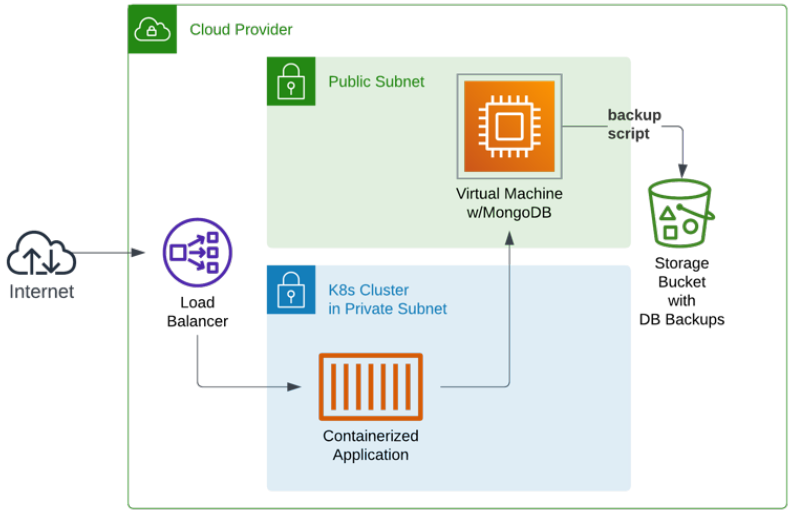
Excellent presentations will:

- Incorporate a mix of slides and live walkthrough
- Evidence the proper function of the infrastructure
- Discuss your approach to the build-out, including challenges and adaptations
- Detail your grasp of weak configurations and their potential consequences
- Demonstrate the value of the security tool in reducing risk

Exercise Layout / Timing

When you are ready, we'll schedule you with a Wiz expert panel. Plan for your initial presentation to take about 15 minutes. In the following 30 minutes, our panel will be asking questions about what you built as they will be leveraging a scoring rubric and need all the details. With the remaining 15 minutes, we can have a discussion about how Wiz works and answer any questions you may have about our solution.

Technical Exercise Requirements/Details



Environment Requirements:

- **Web Application Tier:** A containerized web application deployed on Kubernetes.
- **Database Tier:** A database server set up on a VM in a public subnet.
- **Storage Tier:** Cloud object storage configured as anonymous public readable.

Kubernetes Application Exercise

Your web application will be publicly accessible and set up to communicate with the database server. The database backups should be automated and stored in the public-readable cloud object storage.

At the end of basic setup, you should have a **working web application** which can be accessed from the web, and a VM instance running MongoDB. Before the presentation begins, ensure that every component of the exercise is operational and ready for demonstration.

- **Database:** Set up a VM using an outdated Linux version. On top of this VM, install an outdated database server. (DB example – MongoDB). Configure the VM to allow SSH connections from the internet. Configure the Database and/or associated networking to only allow connections to the Database from the **applications deployed to your Kubernetes cluster.**
- **Database authentication**: Ensure the database is configured for local authentication so you can construct a database connection string.
- **Highly Privileged DB VM**: Configure the VM in a way that it is granted overly permissive CSP permissions.
- **Object Storage**: Create a cloud object storage resource which will store the database backups. Modify the permissions on the cloud object storage resource to allow public read access to the backups stored within, as well as listing contents. *You will be asked to validate during the review that the backup is accessible via an external URL*.
- **DB Backups**: Create automation which regularly backs up the database(s) to the created bucket.
- **Kubernetes Cluster**: Deploy a Kubernetes cluster to host a containerized web application.
- **Containerized Web Application**:
    - Build and deploy a containerized web application to the Kubernetes cluster. You can develop your own, utilize open-source solutions, or try this sample: https://github.com/jeffthorne/tasky
    - Ensure the container employs database authentication. This typically uses a connection string format.
    - Confirm the built container image includes a file named "wizexercise.txt" with content.

- **Public Access**: Set up the containerized web application to be reachable from the public internet.
- **Container Admin Configuration**: Configure the web application container to run with cluster-admin privileges.
- Implement Cloud Service Provider native tooling that will evaluate misconfigurations of the infrastructure built through this exercise, and detect misconfigurations or risks introduced through the specified architecture above.

DevOps Exercise

As part of the exercise, we would like you to demonstrate leveraging modern DevOps practices to automate the building of your cloud infrastructure and the deployment of your application.

- **VCS/SCM**: Push your code to a VCS/SCM of your choice (GitHub, Gitlab, Azure DevOps, etc.)
- **CI Pipelines**: Setup two CI pipelines:
  - One CI pipeline to securely deploy your cloud infrastructure using IaC (Terraform, CloudFormation, etc.)
  - One CI pipeline to build & push your containerized application to a container registry of your choosing.

Cloud Detection & Response Exercise

- **Technical Security Controls:** Please have technical security controls in place before your application reaches production. These controls may cover a variety of different categories, such as compliance, audit, or security monitoring. The security controls may be preventative, detective, or responsive.
  - Include at least one preventative and detective control
- **Control Plane Audit Logging:** Please setup control plane audit logging for your CSP and show a sample event of the activity produced during the tech task
- **(Optional):** Setup cloud native detective controls for your CSP
- **(Optional):** Run a simulated attack or simulated behavior to showcase the efficacy of your preventative and detective controls

Extra Credit

Additions that demonstrate your technical skills are welcome, provided they fit into the time allotted. Efficient, repeatable approaches to buildouts are celebrated, as are any pains taken to make evidencing your buildout clear and intuitive.

Additional Resources:

- CI Systems
  - Github Actions - Docs
  - Jenkins - Docs
  - CircleCI Pipeline - Docs

- Identity and Access Management Policies:
  - AWS Identity and Access Management (IAM) Documentation: What is IAM?
  - Google Cloud IAM Documentation: Cloud Identity and Access Management (IAM)
  - Azure Identity and Access Management: Azure Active Directory Documentation
- Kubernetes RBAC Documentation: Kubernetes RBAC Documentation
- Infrastructure-as-Code tooling for Kubernetes deployment:
  - Terraform Kubernetes Provider: Terraform Kubernetes Provider Documentation
  - Helm – The Kubernetes Package Manager: Helm Official Site

- Cloud Detection & Response
  - Datadog - CI Docs
  - Splunk - CI/CD Monitoring
  - AWS Guard Duty - Docs
  - GCP Security Command Center - Best Practices
  - Azure Defender for Cloud - Docs