**OpenHack – Microsoft Azure Well-Architected Framework**

**Overview**

The **Microsoft Azure Well-Architected Framework OpenHack** attempts to test the participants' understanding of the Microsoft Well-Architected pillars. During the OpenHack, participants will demonstrate and practice a standardized process of conducting a Well-Architected assessment across all five of its pillars against a customer's workload.

The **Microsoft Azure Well-Architected Framework OpenHack** simulates a real-world scenario of a bank who has attempted to deploy a production workload to Azure. Unfortunately, the bank was unaware of the Well-Architected pillars and proven design patterns, therefore, the production workload fails in many areas.

During the "hacking," attendees will focus on:

1. Analyzing and identifying enhancements to the customer's environment
2. Strengthening the customer's operational landscape
3. Ensuring that the customer's experience with the Microsoft cloud is of the utmost quality, value, and excellence

By the end of the OpenHack, it is expected that attendees will be familiar with the Microsoft Azure Well-Architected Framework and able to conduct an assessment against a customer's workload in the Azure cloud.

**Key Technologies**

* [Azure DevOps](https://docs.microsoft.com/azure/devops/" \o "https://docs.microsoft.com/azure/devops/)
* [Azure Bicep/ARM Templates](https://docs.microsoft.com/azure/azure-resource-manager/bicep/)
* [Azure Key Value](https://docs.microsoft.com/azure/key-vault)
* [Azure Traffic Manager](https://docs.microsoft.com/azure/traffic-manager)
* [Virtual Machines](https://docs.microsoft.com/azure/virtual-machines/windows/overview)
* [Azure App Service](https://docs.microsoft.com/azure/app-service/)
* [Azure SQL](https://docs.microsoft.com/azure/azure-sql/)
* [Azure Load Balancer](https://docs.microsoft.com/azure/load-balancer/)
* [Application Gateway - WAF v2](https://docs.microsoft.com/azure/application-gateway/)
* [Azure Front Door](https://docs.microsoft.com/azure/frontdoor/)
* [Azure Cost Management](https://docs.microsoft.com/azure/cost-management-billing/)
* [Azure vNets](https://docs.microsoft.com/azure/virtual-network)
* [Azure Service Endpoints](https://docs.microsoft.com/azure/virtual-network/virtual-network-service-endpoints-overview)
* [Azure Monitor](https://docs.microsoft.com/azure/azure-monitor/)
* [Azure Defender](https://docs.microsoft.com/azure/security-center/azure-defender)
* [Azure Sentinel](https://docs.microsoft.com/azure/sentinel/)
* [Grafana](https://grafana.com/docs/)

**Prerequisites**

**Knowledge Prerequisites**

To be successful and get the most out of this OpenHack, it is highly recommended that participants have a deep understanding (L400+) of the following:

* [Microsoft Azure Well-Architected Framework](https://docs.microsoft.com/azure/architecture/framework/)
* [Cloud Design Patterns](https://docs.microsoft.com/azure/architecture/patterns/index-patterns)

Additionally, teams should be deeply familiar with the key technologies listed above.

Finally, participants should have operational knowledge (L200) of the following technologies:

* [Azure DevOps Build/Release Pipelines](https://docs.microsoft.com/azure/devops/pipelines/get-started/what-is-azure-pipelines)
* [Azure Bicep](https://docs.microsoft.com/azure/azure-resource-manager/templates/bicep-overview)
* [Azure CLI](https://docs.microsoft.com/cli/azure/)

Be prepared to roll up your sleeves, learn, and participate in an interactive team environment.

**Tooling Prerequisites**

To be successful and get the most out of this OpenHack and to avoid any delays with downloading or installing tooling, you are encouraged to have the following ready to go and to review the links and resources listed (as needed for upskilling).

* A modern laptop running Windows 10 (1703 or higher), Mac OSX (10.12 or higher), or Ubuntu (16.04 or higher)
* Install your choice of Integrated Development Environment (IDE) software, such as [Visual Studio](https://visualstudio.microsoft.com/) or [Visual Studio Code](https://visualstudio.microsoft.com/)
* Download the latest version of [Azure CLI](https://docs.microsoft.com/cli/azure/install-azure-cli)
* Download the latest version of [Azure Bicep CLI](https://github.com/Azure/bicep/blob/main/docs/installing.md)
* For performance tests, install [k6 CLI](https://k6.io/docs/getting-started/installation/) (to install k6 CLI on Windows, you'll first need to install [Chocolatey Package Manager](https://chocolatey.org/install)) or install [Docker Desktop](https://docs.docker.com/engine/install/)
* To build applications locally, participants will need to download and install [.NET Core 5.0 SDK](https://dotnet.microsoft.com/download)

**Development Environment Configuration**

Each team will need the following resources for the OpenHack:

* An Azure subscription with Owner privileges. The teams will be required to create Service Principals to connect Azure DevOps for deploying templates and to connect Grafana to Azure Monitor.
* An Azure DevOps tenant with Project Collection Administrator privileges. The tenant will contain two projects along with source code for the OpenHack.

A deployment guide can be found in the public [BYOS repo](https://github.com/microsoft/OpenHack/blob/main/byos/waf/deployment.md).

**Language-Specific Prerequisites**

The following languages/technologies are not required of *each participant*. However, the team, as a whole, should be able to cover the following together.

* **C#** - Some understanding is necessary for comprehending source code and completing later challenges.
* **JavaScript** - An extremely elementary knowledge is required if participants wish to understand the code. Otherwise, they will only be updating a few strings at the beginning of a script.
* **SQL (Entity Framework) with SQL Profiler** - Teams should be able to identify under performing queries and have the ability to optimize, as necessary.
* **YAML for Azure DevOps Build/Release Pipelines** - Initial YAML files are provided. However, teams must be able to update the scripts, as necessary, to complete challenges. (NOTE: Teams may wish to leverage GitHub workflows but doing so is outside the scope of this OpenHack.)
* **Azure Bicep** - The infrastructure is built using Azure Bicep, along with some desired state configurations (DSC). Teams will need to know how to develop Azure Bicep scripts in order to complete the Infrastructure-as-Code (IaC) components of this OpenHack. (NOTE: Teams may also leverage pure ARM templates or Terraform. However, both are outside the scope of this OpenHack and may prove to cost the team valuable time.)

**Links & Resources**

Besides the foundation knowledge requirements outlined above, below are a few additional resources that may be helpful prior to beginning this OpenHack.

**General**

* [Microsoft Assessments: Azure Well-Architected Review](https://docs.microsoft.com/assessments/?mode=pre-assessment&id=azure-architecture-review)

**DevSecOps**

* [DevSecOps in Azure](https://docs.microsoft.com/azure/architecture/solution-ideas/articles/devsecops-in-azure) (NOTE: While some content centers around AKS and Terraform, the general concepts are the same.)
* [Design a CI/CD pipeline using Azure DevOps](https://docs.microsoft.com/azure/architecture/example-scenario/apps/devops-dotnet-webapp)
* [End-to-end governance in Azure when using CI/CD](https://docs.microsoft.com/azure/architecture/example-scenario/governance/end-to-end-governance-in-azure)
* [DevTest and DevOps for IaaS solutions](https://docs.microsoft.com/azure/architecture/solution-ideas/articles/dev-test-iaas)
* [CI/CD for Azure VMs](https://docs.microsoft.com/azure/architecture/solution-ideas/articles/cicd-for-azure-vms)
* [DevTest and DevOps for PaaS solutions](https://docs.microsoft.com/azure/architecture/solution-ideas/articles/dev-test-paas)
* [CI/CD for Azure Web Apps](https://docs.microsoft.com/azure/architecture/solution-ideas/articles/azure-devops-continuous-integration-and-continuous-deployment-for-azure-web-apps)
* [DevOps in a hybrid environment](https://docs.microsoft.com/azure/architecture/solution-ideas/articles/devops-in-a-hybrid-environment)
* [Implementation reference for load testing pipeline solution](https://docs.microsoft.com/azure/architecture/example-scenario/banking/jmeter-load-testing-pipeline-implementation-reference) (NOTE: Documentation is specific to JMeter but can be adapted to k6.)
* [Hybrid availability and performance monitoring](https://docs.microsoft.com/azure/architecture/hybrid/hybrid-perf-monitoring)
* [Centralized app configuration and security](https://docs.microsoft.com/azure/architecture/solution-ideas/articles/appconfig-key-vault)
* [Secure and govern workloads with network level segmentation](https://docs.microsoft.com/azure/architecture/reference-architectures/hybrid-networking/network-level-segmentation)
* [Security considerations for highly sensitive IaaS apps in Azure](https://docs.microsoft.com/azure/architecture/reference-architectures/n-tier/high-security-iaas)
* [Threat indicators for cyber threat intelligence in Azure Sentinel](https://docs.microsoft.com/azure/architecture/example-scenario/data/sentinel-threat-intelligence)
* [Web application monitoring on Azure](https://docs.microsoft.com/azure/architecture/reference-architectures/app-service-web-app/app-monitoring)

**Core Networking**

* [Zero-trust network for web applications with Azure Firewall and Application Gateway](https://docs.microsoft.com/azure/architecture/example-scenario/gateway/application-gateway-before-azure-firewall)
* [Multi-region load balancing with Traffic Manager and Application Gateway](https://docs.microsoft.com/azure/architecture/high-availability/reference-architecture-traffic-manager-application-gateway)
* [Protect APIs with Application Gateway and API Management](https://docs.microsoft.com/azure/architecture/reference-architectures/apis/protect-apis)
* [Azure Firewall architecture overview](https://docs.microsoft.com/azure/architecture/example-scenario/firewalls/)
* [Securely managed web applications](https://docs.microsoft.com/azure/architecture/example-scenario/apps/fully-managed-secure-apps)

**Infrastructure**

* [IaaS: Web application with relational database](https://docs.microsoft.com/azure/architecture/high-availability/ref-arch-iaas-web-and-db)
* [Basic Web Application](https://docs.microsoft.com/azure/architecture/reference-architectures/app-service-web-app/basic-web-app)
* [Multi-region web app with private connectivity to database](https://docs.microsoft.com/azure/architecture/example-scenario/sql-failover/app-service-private-sql-multi-region)
* [Enterprise deployment using App Services Environment](https://docs.microsoft.com/azure/architecture/reference-architectures/enterprise-integration/ase-standard-deployment)
* [High availability enterprise deployment using App Services Environment](https://docs.microsoft.com/azure/architecture/reference-architectures/enterprise-integration/ase-high-availability-deployment)
* [High availability and disaster recovery scenarios for IaaS apps](https://docs.microsoft.com/azure/architecture/example-scenario/infrastructure/iaas-high-availability-disaster-recovery)
* [Build high availability into your BCDR strategy](https://docs.microsoft.com/azure/architecture/solution-ideas/articles/build-high-availability-into-your-bcdr-strategy)
* [Highly available multi-region web application](https://docs.microsoft.com/azure/architecture/reference-architectures/app-service-web-app/multi-region)

**Applications**

* [Unified logging for microservices applications](https://docs.microsoft.com/azure/architecture/example-scenario/logging/unified-logging)
* [Modernize .NET applications](https://docs.microsoft.com/azure/architecture/solution-ideas/articles/net-app-modernization)
* [Scalable Web Apps](https://docs.microsoft.com/azure/architecture/solution-ideas/articles/scalable-web-apps)
* [Simple branded website](https://docs.microsoft.com/azure/architecture/solution-ideas/articles/simple-branded-website)
* [Serverless web application](https://docs.microsoft.com/azure/architecture/reference-architectures/serverless/web-app)

**Post Learning Recommendations**

*TBD.*   
*CSU is currently determining a “post-learning” process. I can complete this section once those details are finalized.*

**Challenges**

## Challenge 1: Assess the environment

In this challenge, you will assess a customer's production environment that has been deployed to Azure and currently hosting a live workload. You will gain an understanding of the customer's current process for deployment and management of the applications and supporting cloud resources. Part of this understanding will include performing a cost analysis of the Azure environment and establishing some baseline performance metrics. Based upon the assessment, you will need to craft a plan to assist the customer in adoption and implementation of the Microsoft Well-Architected Framework. You will use this plan to govern how you proceed in future challenges.

### **Learning Objectives**

* Inventory current components
* Identify potential areas of improvement
* Perform a cost analysis of the environment
* Conduct an initial performance test
* Conduct a load test against current business SLAs
* Conduct a stress test
* Establish and document performance baselines
* Understand the difference between the three types of performance tests
* Demonstrate a basic understanding of WAF
* Document any initial concerns for business meeting SLAs

## Challenge 2: Operationalize automated deployments

In this challenge, you will assist the customer in adopting automated deployments of both the application and infrastructure. You will explore the Azure Bicep scripts included in the Azure DevOps project and learn how to convert them to ARM templates for environment configuration and deployments. When helping the customer automate infrastructure and application changes, you will configure build and release pipelines. You will also attach deployment gates to release pipelines for security and governance of cloud resources. Deployments should ensure that only the correct resources should be removed by reporting changes prior to being executed. It is important that you have a firm understanding of Azure Bicep and that your pipelines are configured correctly as **all future changes to the Azure environment must be conducted through the pipelines**.

### **Learning Objectives**

* Create build/release pipelines in Azure DevOps
* Determine best strategy for deployments (conditions, types, etc.)
* Demonstrate "what-if" scenarios with deployments
* Configure gated deployments
* Discuss deployment models to maximize uptime

## Challenge 3: Planning for failure

It is not a matter of if an environment will fail, but when. Therefore, prior to any changes being introduced into the application or its supporting infrastructure, it is important that proper monitoring and alerting be configured. In this challenge, you will determine what types of monitoring, along with the necessary data points, should be conducted. You will also determine and configure any necessary alerting of activities within your environment. These activities could include, but are not limited to, resource availability (or lack thereof), security threats, etc. Finally, you will determine a BCDR strategy in which you will implement in a future challenge.

### **Learning Objectives**

* Discuss potential alerting options and processes
* Compose a list of metrics that should be monitored
* Compose a list of conditions that should generate an alert
* Configure alerting for determined conditions
* Demonstrate a successful alert
* Configure Azure Monitor and Security Center
* Enable Azure Sentinel
* Discuss BCDR strategy, including the varying costs of potential solutions relative to RPO/RTO/SLAs/KPIs
* Document BCDR strategy
* Identify points (roles) of contact for failures in the environment and construct a RACI chart

## Challenge 4: Visualizing operations

Now that significant planning and documentation has been completed, it is time to begin executing those plans. However, one last thing to consider is visualizing changes across your environment. In this short challenge, you will take the composed list of metrics and conditions from the last challenge and build some operational dashboards. You will create a free Grafana account and link your Azure Monitor instance to it. You will then build some time-series reports and create an operational dashboard.

**Learning Objectives**

* Creating demo Grafana account
* Linking Azure Monitor to Grafana
* Creating operational dashboard in Grafana
* Reporting on IaaS resources
* Discuss the benefits and use cases of a real-time operational dashboard

## Challenge 5: Improve the environment

Before any changes are introduced into the current application architecture, the customer is interested in hardening what is currently deployed in terms of security, redundancy, and overall optimization. In this challenge, you will leave the application structure as-is—leaving it running on VMs—but you will implement some changes to the deployments and configurations of the environment. You will need to address secrets, potential areas of attack, and resiliency within the application. You will also need to make sure the telemetry is captured from the application and the virtual machine.

### **Learning Objectives**

* Right-sizing VMs
* Redundancy of VMs using Availability Zones and proper storage configurations
* Securing network resources using NSGs
* Implementing JIT on VMs
* Discuss and properly configure RBAC based on RACI matrix
* Discuss options for securing database connection strings and implement a solution
* Adding Application Insights to an application
* Configuring VM to send telemetry via Application Insights or Log Analytics agent
* Replacing ELB with Application Gateway and configuring OWASP
* Comparing performance to baselines and ensuring that business SLAs are met with changes in architecture
* Discuss potential cost improvements (Azure Reserved Instances)

## Challenge 6: Leveraging PaaS services

Often, customers choose to build their own environments for services already managed by Microsoft or its partners. This customer is no different. In this challenge, you will begin migrating the components of the web application to Azure's various PaaS offerings. As you migrate the application and configure its services, you must consider all pillars of the Microsoft Well-Architected Framework. With your implementation, you will need to ensure that all of the customer's SLAs are able to be met at all times while maintaining adherence to the Microsoft Azure Well-Architected Framework. You will also need to make sure that your Grafana dashboards are updated to report desired metrics from the new resources. Finally, all alerts should be updated based on the new resources.

### **Learning Objectives**

* Configuring Azure App Service for autoscale
* Securing Azure App Service
* Configuring Azure SQL for failover
* Using Key Vault in Azure App Service
* Enabling SQL Always Encrypted
* Leveraging private endpoints
* Using Azure Batch
* Continuing to compare performance to baselines and ensuring that business SLAs are met with changes in architecture
* Failing a region while maintaining performance SLAs
* Reporting on PaaS services

## Challenge 7: Optimizing the API

In the previous challenge, you may have elected to move the API to Azure Functions. If so, great! If not, here is your chance. In this challenge, you will separate the web front-end (WFE) from its services layer (API). Upon doing so, you will move the service layer to Azure Functions. In light of the Microsoft Azure Well-Architected Framework, you will also need to configure Azure Functions accordingly.

### **Learning Objectives**

* Separating an application's front-end from its API
* Securing of Azure Functions
* Configuring autoscaling of Azure Functions
* Instrumenting Azure Functions with Application Insights
* Discuss the various billing models of Azure Functions and the reasoning behind which one was chosen in the case of this challenge
* Discuss the benefits of separating the two layers of the application
* Continuing to compare performance to baselines and ensuring that business SLAs are met with introduction of Azure Functions

## Challenge 8: Tightening database security

The application, unfortunately, is still using legacy database connection strings. This means that there are still identities (SQL or Azure AD) that are used to connect the application to the database. In this challenge, you will refactor your application to leverage managed identities for database connectivity and remove application-specific named users from the database.

### **Learning Objectives**

* Using managed identities for database connectivity
* Understand and discuss the benefits of managed identities in terms of the Microsoft Azure Well-Architected Framework

## Challenge 9: Capturing correlation data

You have already introduced Application Insights into your application. However, the customer is finding it difficult to correlate errors between the web front-end and the backend. In this challenge, you will discuss approaches to implementing the capturing of correlation data when an error occurs in the application. Remember, solutions should consider all pillars of the Well-Architected Framework.

### **Learning Objectives**

* Discuss techniques for capturing correlation data
* Understand costs and security when storing correlation data
* Understand how processing correlation data can impact application performance
* Understand what data to capture in a distributed architecture

**Value Proposition**

* Understand and clearly articulate the Microsoft Well-Architected Framework and each of its pillars, along with the respective benefits
* Systematically analyze a given workload under the individual lenses of the Well-Architected Framework pillars
* Deliver appropriate recommendations based on standard proven practices in accordance with the Well-Architected Framework
* Successfully implement and migrate existing workloads to optimize and strengthen said workloads in all pillars of the Well-Architected Framework
* Increase the overall technical intensity of technical team members

**Technical Scenarios**

* Performance - test and maintain, if not improve, performance of the application and its supporting infrastructure in order to meet specified SLAs
* Resiliency - develop a BCDR strategy and architect an environment to which to migrate a customer's workload that will maximize uptime
* Security - ensure that all connections to and from the application, including the database and its data, are secure, and verify access to cloud resources are restricted to necessary teams via appropriate RBAC policies
* Modernization - transition a traditional, monolithic web application to modern PaaS services

**Audience**

* Target Audience:
  + Microsoft - CE, CSE, CSA, GBB, ATT, CE, SE, TPM
  + Customer - Developers, Architects, SREs, DevOps Engineers, Systems Integrators, Systems Administrators
* Target Verticals: All
* Customer Profiles:
  + Customers who are looking to deploy new cloud-native applications and/or infrastructure into Azure and wish to have an assessment conducted on the workload prior to it going live.
  + Customers who currently have cloud-native applications running in production and are seeking visibility into optimization in terms of cost, security, reliability, and/or performance.
  + Customers who have migrated traditional, on-premises workloads to Azure and are seeking additional assistance in optimization for cloud operations.

**Registration questions**

|  |  |
| --- | --- |
| Topic specific question | Response Options |
| What is your level of understanding using Azure today? | None  Some understanding  I have some pilot work on Azure  I heavily rely on Azure today for cloud |
| What is your level of understanding of the Azure Architecture Center? | None  Some understanding  I use it occasionally  I use if frequently |
| What is your level of understanding of Cloud Design Patterns? | None  Some understanding  I use them occasionally  I use them frequently |
| What is your level of understanding of the Microsoft Azure Well-Architected Framework? | None  Some understanding  Intermediate knowledge  Expert |
| Which area most closely aligns to your expertise/role? | Developer  Infrastructure  Data  Security  Other |
| How many years of experience do you have in your area? | < 1 year  1-5 years  5-10 years  10+ years |
| What level of understanding do you have in: |  |
| Performance testing | None  Introductory  Fundamental  Intermediate  Advanced |
| Azure DevOps Build/Release Pipelines | “” |
| Azure Bicep, ARM Templates | “” |
| Core Azure IaaS services | “” |
| Core Azure PaaS services | “” |
| Azure Networking | “” |
| Cost Management | “” |
| Azure Monitor | “” |
| Grafana | “” |
| Cloud & application security | “” |
| Business continuity and disaster recovery (BCDR) | “” |

**FAQ**

Q: Are there any pre-requisites to attending this OpenHack?

A: Yes. Internal attendees should have completed all MS Learn modules for WAF and any hands-on labs for the individual pillars.

Q: Are there any OpenHacks that I should complete or would be considered advantageous to complete before participating in Microsoft Azure Well-Architected Framework OpenHack?

A: Yes. While the following OpenHacks are not *required*, they could provide some fundamental understanding in many of the principles covered in the Microsoft Azure Well-Architected Framework OpenHack.

* + App Modernization with NoSQL
  + DevOps
  + Migration (Line of Business)
  + Serverless

Q: I have been working in Azure for years. Is this OpenHack really necessary?

A: Yes. There are elements of the Microsoft Azure Well-Architected Framework with which you may be unfamiliar. This OpenHack will help solidify and validate your knowledge of all five pillars.

Q: For which technology vertical is this OpenHack designed?

A: The Microsoft Azure Well-Architected Framework is not for any single role or vertical but for all technology professionals. Furthermore, while this OpenHack is designed to challenge any IT professional who works in the cloud, many insights can be gleaned for those who work with on-premises environments.