**AWS Lambda**

Here are some common AWS Lambda interview questions and answers for experienced professionals:

**1. What is AWS Lambda?**

**Answer**:  
AWS Lambda is a serverless compute service that automatically manages the underlying infrastructure and allows you to run code in response to events without provisioning or managing servers. It automatically scales the application by running the code in parallel and handles requests as they come in.

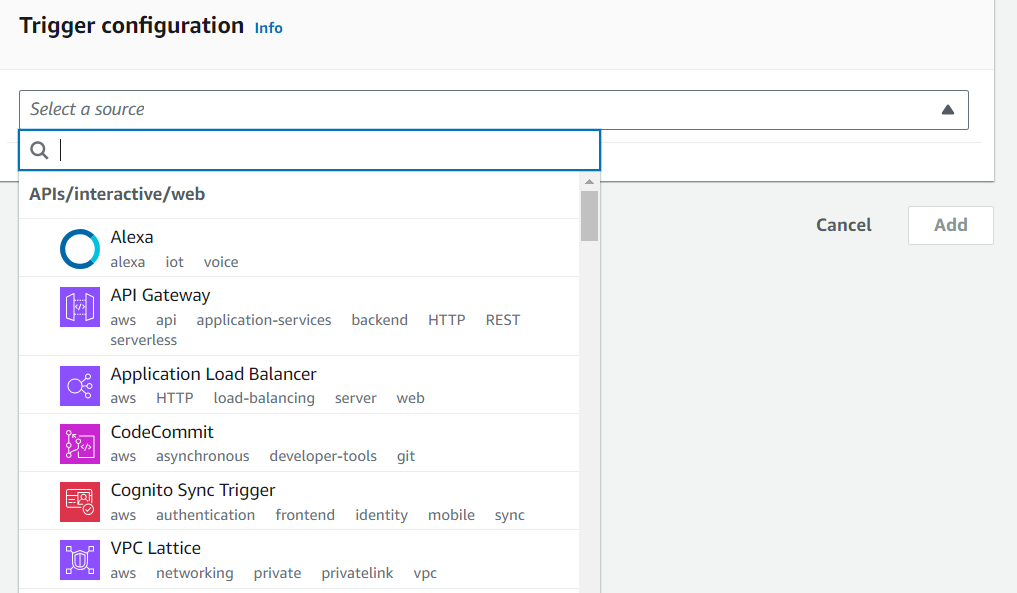
**2. How does AWS Lambda scale with increasing traffic?**

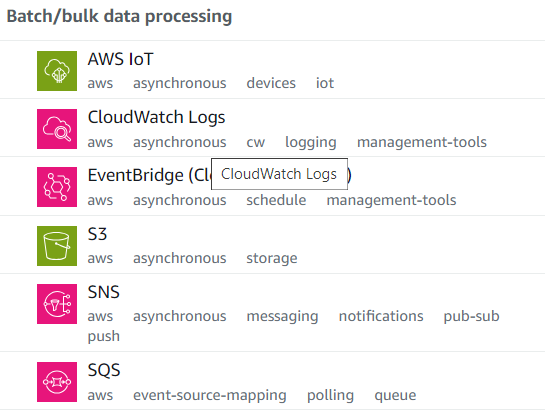
**Answer**:  
AWS Lambda automatically scales by creating new instances of the function as requests come in. Each request is handled by a separate invocation of the Lambda function, and AWS handles the scaling based on the number of incoming requests. However, it is limited by concurrency limits, which can be adjusted.

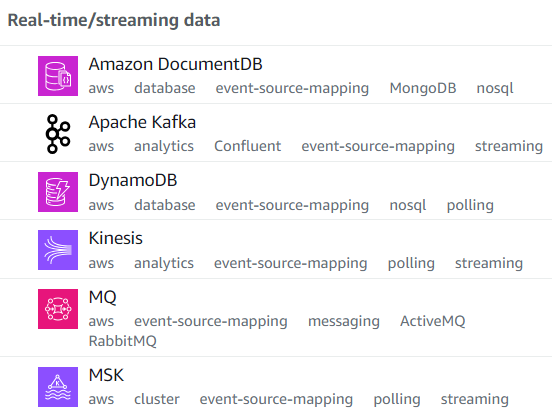
**3. What are the Lambda triggers?**

**Answer**:  
Lambda triggers are event sources that invoke Lambda functions. Common triggers include:

* **Amazon S3**: Triggered by events like file uploads or modifications.
* **Amazon DynamoDB Streams**: Triggered when new data is added or modified.
* **Amazon SNS**: Lambda can be invoked when an SNS notification is published.
* **Amazon API Gateway**: Triggered by HTTP requests.
* **Amazon CloudWatch Events**: Triggered by scheduled events or system events.
* **AWS SQS**: Triggered by new messages in an SQS queue.

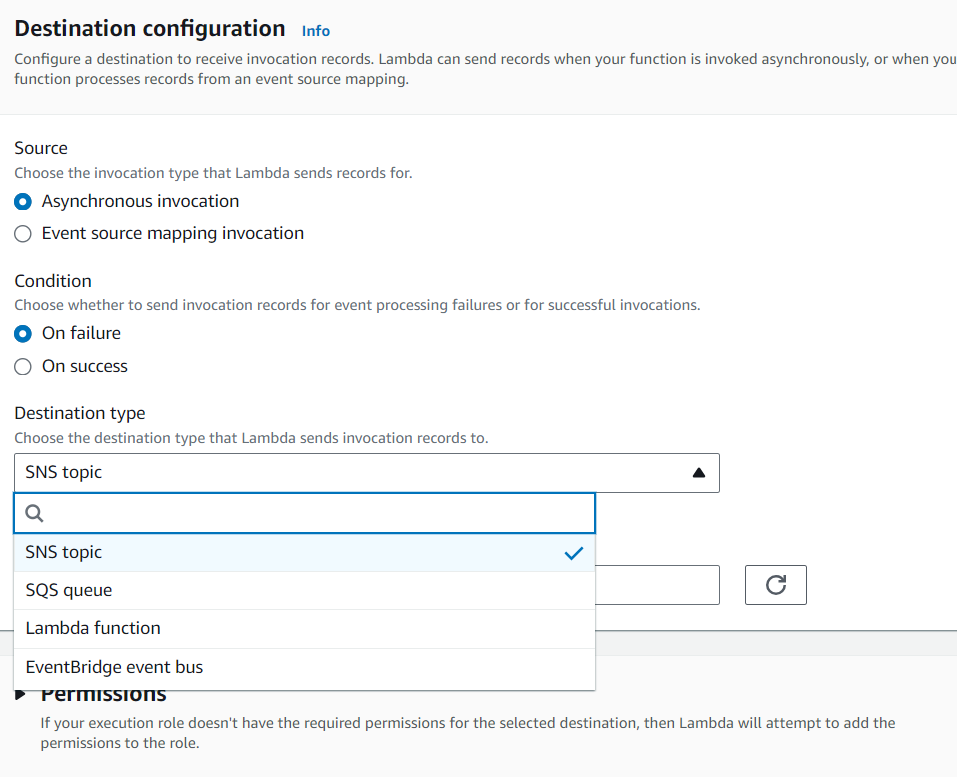






**Supported Destination Targets:**

1. **Amazon SNS (Simple Notification Service)**: Sends a notification when the function succeeds or fails.
2. **Amazon SQS (Simple Queue Service)**: Places a message in a queue when the function succeeds or fails.
3. **Amazon EventBridge**: Sends an event to EventBridge for further processing or automation.
4. **Another AWS Lambda Function**: Invoke another Lambda function for further processing or error handling.



**4. What are the limitations of AWS Lambda?**

**Answer**:

* **Execution Timeout**: The maximum execution timeout is 15 minutes.
* **Memory Limits**: Lambda functions can be allocated between 128 MB and 10 GB of memory.
* **Package Size**: The maximum deployment package size (including layers) is 50 MB for zipped files, or 250 MB unzipped.
* **Concurrency Limits**: By default, Lambda allows 1000 concurrent executions per account, but this can be increased via a support request.

**5. How do you handle errors in AWS Lambda?**

**Answer**:  
Lambda functions handle errors using the following methods:

* **Retry Mechanism**: For asynchronous invocations, Lambda automatically retries the function twice after the first failure.
* **Dead Letter Queue (DLQ)**: You can configure a DLQ to store messages when a Lambda function fails after retries.
* **AWS Lambda Destinations**: Allows you to route results (success or failure) to different AWS services like SNS, SQS, or another Lambda function.
* **Custom Error Handling**: You can use custom exception handling within your code and return structured error messages.

**6. What are Lambda Layers?**

**Answer**:  
Lambda Layers are a way to manage and share common code or libraries across multiple Lambda functions. Layers are used to store external libraries, custom runtimes, or any shared code so that each function doesn't have to include the same code in its deployment package, thus reducing the package size.

**7. What is the cold start problem in AWS Lambda? How do you mitigate it?**

**Answer**:  
A **cold start** occurs when a Lambda function is invoked after a period of inactivity, and it takes longer to start because AWS needs to initialize the execution environment. Cold starts primarily affect functions using languages with higher initialization times like Java or .NET.  
**Mitigations**:

* **Provisioned Concurrency**: Pre-warms a number of Lambda instances to avoid cold starts.
* **Use Lightweight Runtimes**: Languages like Python and Node.js have faster cold start times.
* **Optimize Function Code**: Reduce the initialization overhead by lazy-loading resources or minimizing dependencies.

**8. How do you secure AWS Lambda functions?**

**Answer**:  
Lambda security can be managed in various ways:

* **IAM Roles**: Each Lambda function should have an IAM role with the least privilege, granting it only the permissions it needs to perform its tasks.
* **Environment Variables Encryption**: Store sensitive data like API keys using encrypted environment variables and access them securely via the AWS KMS.
* **VPC Integration**: Lambda functions can be placed in a VPC to access internal resources securely.
* **Lambda Authorizers**: For API Gateway, Lambda can act as an authorizer to authenticate and authorize API requests.

**9. What is AWS Lambda@Edge?**

**Answer**:  
AWS Lambda@Edge allows you to run Lambda functions at AWS locations closer to the user using CloudFront. This reduces latency for the end user and enables serverless execution of code at edge locations in response to CloudFront events such as request/response processing.

**10. What is Provisioned Concurrency in AWS Lambda?**

**Answer**:  
Provisioned Concurrency ensures that Lambda functions start instantly by pre-warming the execution environments. When enabled, AWS keeps a specified number of execution environments running, ready to serve requests. This is particularly useful for performance-sensitive applications where cold starts are unacceptable.

**11. How do you monitor Lambda functions?**

**Answer**:

* **Amazon CloudWatch Logs**: Lambda automatically logs output and errors to CloudWatch Logs. This can be used for debugging.
* **Amazon CloudWatch Metrics**: Tracks Lambda performance metrics such as invocation count, duration, errors, and throttles.
* **AWS X-Ray**: Provides distributed tracing, allowing you to analyze performance bottlenecks and errors within the function and any upstream or downstream services.

**12. What is the pricing model for AWS Lambda?**

**Answer**:  
AWS Lambda pricing is based on:

* **Requests**: $0.20 per 1 million requests.
* **Duration**: $0.00001667 for every GB-second (the amount of memory allocated to the function multiplied by the execution time).
* **Provisioned Concurrency**: Charged based on the number of instances pre-warmed and the duration they are provisioned.

**13. How do you debug AWS Lambda functions?**

**Answer**:  
You can debug Lambda functions using:

* **CloudWatch Logs**: Reviewing log output from Lambda to see any errors or debug information.
* **AWS X-Ray**: Used for tracing Lambda requests and debugging performance bottlenecks.
* **Locally Testing**: Tools like AWS SAM (Serverless Application Model) or AWS Cloud9 can be used for local testing of Lambda functions before deploying.

**14. Can AWS Lambda functions call other AWS services? How?**

**Answer**:  
Yes, Lambda functions can call other AWS services by using the appropriate AWS SDKs (available in the Lambda environment). You need to ensure that the Lambda function’s IAM role has the necessary permissions to invoke or interact with the service.

**15. What are some common use cases for AWS Lambda?**

**Answer**:

* **Real-time File Processing**: Triggered by S3 events like uploading a file (e.g., for image processing).
* **Data Transformation**: Transforming data before storing it into another service (e.g., S3 or DynamoDB).
* **API Backends**: Running serverless APIs using AWS Lambda with API Gateway.
* **Automation**: Triggering workflows or processes in response to AWS CloudWatch events or SNS notifications.
* **Stream Processing**: Real-time processing of streaming data from services like Kinesis or DynamoDB streams.