can I use "net/http" with lambda

Yes, you can use Go's `net/http` package with AWS Lambda, but you need an adapter to translate between API Gateway events and standard `http.Request` objects. A Lambda function cannot directly run a standard HTTP server because it operates on an event-driven model, not a long-running web server.

The key is to use the official AWS Lambda Go API Proxy library (`aws-lambda-go-api-proxy`), which acts as the intermediary between an incoming HTTP request and your Go application's HTTP handler.

How to use `net/http` with Lambda

1. Create a standard Go HTTP server

Write your application using Go's standard `net/http` library or a compatible framework like Gin or Gorilla Mux. For example, here is a basic HTTP server with a single handler:

go

```go
package main

import (
        "fmt"
        "net/http"
)

func myHandler(w http.ResponseWriter, r *http.Request) {
        fmt.Fprintf(w, "Hello from a Go Lambda!")
}
```

Ask anything

2. Use the `aws-lambda-go-api-proxy` library

The `aws-lambda-go-api-proxy` library is the bridge that allows your Lambda function to use the standard HTTP handler. It wraps your `http.Handler` and starts it with the Lambda runtime.

Here is how you would use it with the `myHandler` from the previous step:

go

```go
package main

import (
	"context"
	"fmt"
	"net/http"

	"github.com/aws/aws-lambda-go/lambda"
	"github.com/awslabs/aws-lambda-go-api-proxy/httpadapter"
)

func myHandler(w http.ResponseWriter, r *http.Request) {
	fmt.Fprintf(w, "Hello from a Go Lambda!")
}

func main() {
	// Create a standard Go ServeMux and register your handler
	mux := http.NewServeMux()
	mux.HandleFunc("/", myHandler)

	// Wrap the mux in the httpadapter for V2 API Gateway events
	adapter := httpadapter.NewV2(mux)
```

Use code with caution.

3. Set up the Lambda and API Gateway

To complete the process, you need to configure your AWS infrastructure to trigger your Go Lambda function on an HTTP request:

1. **Build and deploy the Go binary**: Compile your Go code for a Linux environment and create a zip file for deployment.
2. **Create an API Gateway**: Set up an API Gateway endpoint that integrates with your Lambda function.
3. **Use a Lambda Function URL (simpler)**: For a quick and easy HTTP endpoint, you can enable a function URL directly on your Lambda function. This automatically provides an HTTPS endpoint for your function.

Advantages of using this approach

- **Code portability**: You can use the same `net/http` handler code for both local development and production Lambda environments.
- **Rich ecosystem**: You gain access to the wide variety of standard Go libraries and web frameworks that are built around the `http.Handler` interface.
- **Simpler migration**: It makes it easy to convert existing Go web applications to run on Lambda without a complete rewrite.

AI can make mistakes, so double-check responses

↪ What are some benefits of Lambda Function URLs over API Gateway?

Jul 18, 2023 — To use with the standard library, the httpadapter.New function takes in a http.Handler (which has the route defined) a...

AWS Builder Center

The Lazy engineer's guide to running your Go web application ...

May 20, 2023 — When we run this command as-is, we get a web server running that we can interact with as we would expect. If we shippe...

Jamie Tanna

AWS API Gateway V2 to Go Lambda HTTP Handler Adaptor

Dec 19, 2021 — Lambda handlers. A Go Lambda handler that's designed to handle API Gateway V2 requests will need to accept the events.

Adrian Hesketh

Show all