Of course! Here is the complete, step-by-step command guide for all tasks in the lab, with the specific API key replaced by a placeholder.

Task 1: Deploying an API Backend

This task deploys the Cloud Function that will act as the backend service for your API.

1. **Set the region and enable the necessary APIs.**
   Bash
   ```
   gcloud config set compute/region us-central1 && gcloud services enable apigateway.googleapis.com servicemanagement.googleapis.com servicecontrol.googleapis.com cloudfunctions.googleapis.com cloudbuild.googleapis.com
   ```

2. **Clone the code repository**.
   Bash
   ```
   git clone https://github.com/GoogleCloudPlatform/nodejs-docs-samples.git
   ```

3. **Navigate into the function's directory**.
   Bash
   ```
   cd nodejs-docs-samples/functions/helloworld/helloworldGet
   ```

4. **Deploy the Cloud Function**. This process will take a few minutes to complete. ⏳
   Bash
   ```
   gcloud functions deploy helloGET \
     --runtime nodejs20 \
     --trigger-http \
     --allow-unauthenticated \
     --region us-central1
   ```

Task 2: Test the API Backend

This step verifies that the Cloud Function was deployed correctly.

1. **Set your Project ID as an environment variable**.
   Bash
   ```
   export PROJECT_ID=$(gcloud config get-value project)
   ```

2. **Test the function's endpoint directly**.
   Bash
   ```
   curl -s -w "\n" "https://us-central1-${PROJECT_ID}.cloudfunctions.net/helloGET"
   ```
   The expected output is Hello World!.

Task 3: Creating a Gateway

Here, you will create the API Gateway and connect it to your backend function.

1. **Navigate back to your home directory**.
   Bash
   ```
   cd ~
   ```
2. **Create the OpenAPI specification file** that defines your API.
   Bash
   ```
   cat << EOF > openapi2-functions.yaml
   swagger: '2.0'
   info:
     title: hello-world-api
     description: Sample API on API Gateway with a Google Cloud Functions backend
     version: 1.0.0
   schemes:
     - https
   produces:
     - application/json
   paths:
     /hello:
       get:
         summary: Greet a user
         operationId: hello
         x-google-backend:
           address: https://us-central1-$PROJECT_ID.cloudfunctions.net/helloGET
         responses:
           '200':
             description: A successful response
             schema:
               type: string
   EOF
   ```
3. **Create the API, the API Config, and the Gateway**. Gateway creation takes about 5-10 minutes. ⏳

Bash
```
gcloud api-gateway apis create hello-world-api --project=$PROJECT_ID
gcloud api-gateway api-configs create hello-world-config --api=hello-world-api
--openapi-spec=openapi2-functions.yaml --project=$PROJECT_ID
--backend-auth-service-account=$(gcloud iam service-accounts list
--filter="displayName:'Compute Engine default service account'" --format='value(email)')
gcloud api-gateway gateways create hello-gateway --api=hello-world-api
--api-config=hello-world-config --location=us-central1 --project=$PROJECT_ID
```

4. **Get the gateway's URL and test it**.

   Bash
   ```
   export GATEWAY_URL=$(gcloud api-gateway gateways describe hello-gateway --location
   us-central1 --format 'value(defaultHostname)')
   curl -s -w "\n" "https://$GATEWAY_URL/hello"
   ```

   The expected output is Hello World!.

Task 4: Securing Access by Using an API Key

This step involves creating an API key in the console.

1. In the Cloud Console, navigate to **APIs & Services > Credentials**.
2. Click **+ CREATE CREDENTIALS** and select **API key**.
3. A pop-up will show your key. **Copy this key** and click **CLOSE**.
4. In Cloud Shell, **save the key to an environment variable**, replacing the placeholder text with the key you just copied.

   Bash
   ```
   export API_KEY="REPLACE_WITH_YOUR_API_KEY"
   ```

Task 5: Create and Deploy a New API Config

Now, you'll update your API definition to require the key you just created.

1. **Create a new, secure OpenAPI specification file**.

   Bash
   ```
   cat << EOF > openapi2-functions2.yaml
   swagger: '2.0'
   ```

```
info:
  title: hello-world-api
  description: Sample API on API Gateway with a Google Cloud Functions backend
  version: 1.0.0
schemes:
  - https
produces:
  - application/json
paths:
  /hello:
    get:
      summary: Greet a user
      operationId: hello
      x-google-backend:
        address: https://us-central1-$PROJECT_ID.cloudfunctions.net/helloGET
      security:
        - api_key: []
      responses:
        '200':
          description: A successful response
          schema:
            type: string
securityDefinitions:
  api_key:
    type: "apiKey"
    name: "key"
    in: "query"
EOF
```

2. **Create a new API config from the secure file**.
   Bash
   ```
   gcloud api-gateway api-configs create hello-secure-config --api=hello-world-api
   --openapi-spec=openapi2-functions2.yaml --project=$PROJECT_ID
   --backend-auth-service-account=$(gcloud iam service-accounts list
   --filter="displayName:'Compute Engine default service account'" --format='value(email)')
   ```

3. **Update the gateway to use the new secure config**. This update will also take a few minutes. ⏳
   Bash
   ```
   gcloud api-gateway gateways update hello-gateway --api=hello-world-api
   --api-config=hello-secure-config --location=us-central1 --project=$PROJECT_ID
   ```

Task 6: Testing Calls Using Your API Key

The final step is to test the newly secured gateway.

1. **Test the call WITHOUT the API key**. This should now fail with an UNAUTHENTICATED error.
   Bash
   ```bash
   curl -sL "https://$GATEWAY_URL/hello"
   ```

2. **Test the call WITH the API key**. This will succeed.
   Bash
   ```bash
   curl -sL -w "\n" "https://$GATEWAY_URL/hello?key=$API_KEY"
   ```

   The successful output will be Hello World!. Congratulations! 🎉