

experiment Lab schedule 1 hour 30 minutes universal_currency_alt No cost

show_chart Introductory

Overview

JSON payloads can be crafted to cause problems when loaded into JSON parsers. In this lab, you learn how to protect against this type of payload.

Objectives

In this lab, you learn how to perform the following task:

- Use the JSONThreatProtection policy to protect your API proxy against malicious JSON payloads.

Setup

For each lab, you get a new Google Cloud project and set of resources for a fixed time at no cost.

1. Sign in to Qwiklabs using an **incognito window**.

2. Note the lab's access time (for example, **1:15:00**), and make sure you can finish within that time.
There is no pause feature. You can restart if needed, but you have to start at the beginning.
3. When ready, click **Start lab**.
4. Note your lab credentials (**Username** and **Password**). You will use them to sign in to the Google Cloud Console.
5. Click **Open Google Console**.
6. Click **Use another account** and copy/paste credentials for **this** lab into the prompts.
If you use other credentials, you'll receive errors or **incur charges**.
7. Accept the terms and skip the recovery resource page.

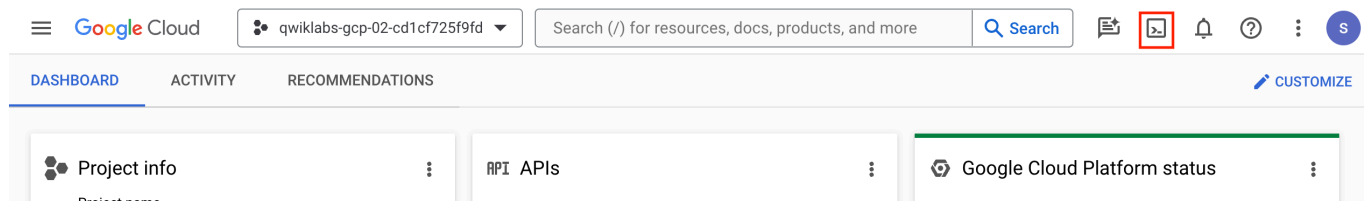
Note: Do not click **End Lab** unless you have finished the lab or want to restart it. This clears your work and removes the project.

Activate Google Cloud Shell

Google Cloud Shell is a virtual machine that is loaded with development tools. It offers a persistent 5GB home directory and runs on the Google Cloud.

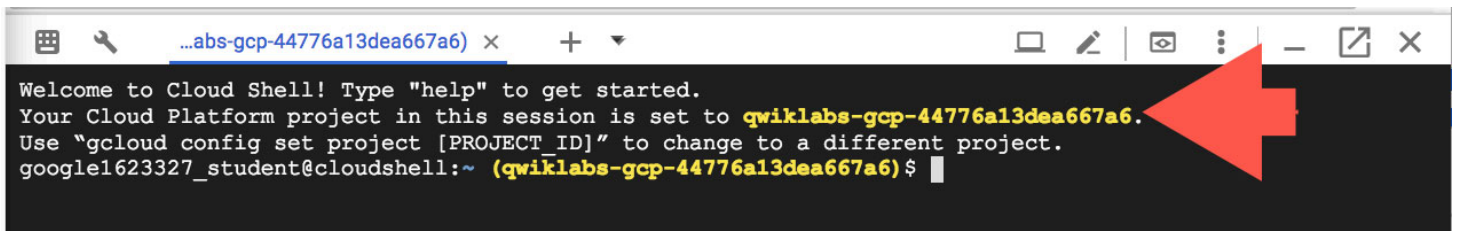
Google Cloud Shell provides command-line access to your Google Cloud resources.

1. In Cloud console, on the top right toolbar, click the Open Cloud Shell button.



2. Click **Continue**.

It takes a few moments to provision and connect to the environment. When you are connected, you are already authenticated, and the project is set to your *PROJECT_ID*. For example:



```
...abs-gcp-44776a13dea667a6) x + ▾
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to qwiklabs-gcp-44776a13dea667a6.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
google1623327_student@cloudshell:~ (qwiklabs-gcp-44776a13dea667a6) $
```

gcloud is the command-line tool for Google Cloud. It comes pre-installed on Cloud Shell and supports tab-completion.

- You can list the active account name with this command:

```
gcloud auth list
```

content_c

Output:

```
Credentialed accounts:
- @.com (active)
```

Example output:

```
Credentialed accounts:
- google1623327_student@qwiklabs.net
```

- You can list the project ID with this command:

```
gcloud config list project
```

content_c

Output:

```
[core]
project =
```

Example output:

```
[core]
```

```
project = qwiklabs-gcp-44776a13dea667a6
```

Note: Full documentation of **gcloud** is available in the gcloud CLI overview guide .

Preloaded assets

These assets have already been added to the Apigee organization:

- The **retail-v1** API proxy
- The oauth-v1 API proxy (for generating OAuth tokens)
- The TS-Retail target server in the eval environment (used by retail-v1)

These assets will be added to the Apigee organization as soon as the runtime is available:

- The **API products**, **developer**, and **developer app** (used by retail-v1)

The **highlighted** items are used during this lab.

Note: Revision 1 of the retail-v1 proxy is marked as deployed, and is immutable. If you ever make a mistake in your proxy code that you can't recover from, you can select revision 1 and restart editing from there.

Task 1. Add a JSONThreatProtection policy

In this task, you add a JSONThreatProtection policy to protect against malicious JSON payloads.

JSON payloads can be crafted to cause issues when they are parsed. The JSONThreatProtection policy can check a JSON payload against configured limits without loading the payload into a parser.

Note: The JSONThreatProtection policy in this API proxy will only be used to validate the JSON payload when an order is being created. In a typical API, you should consider using JSON threat protection any time you allow JSON as an input.

Add the JSONThreatProtection policy

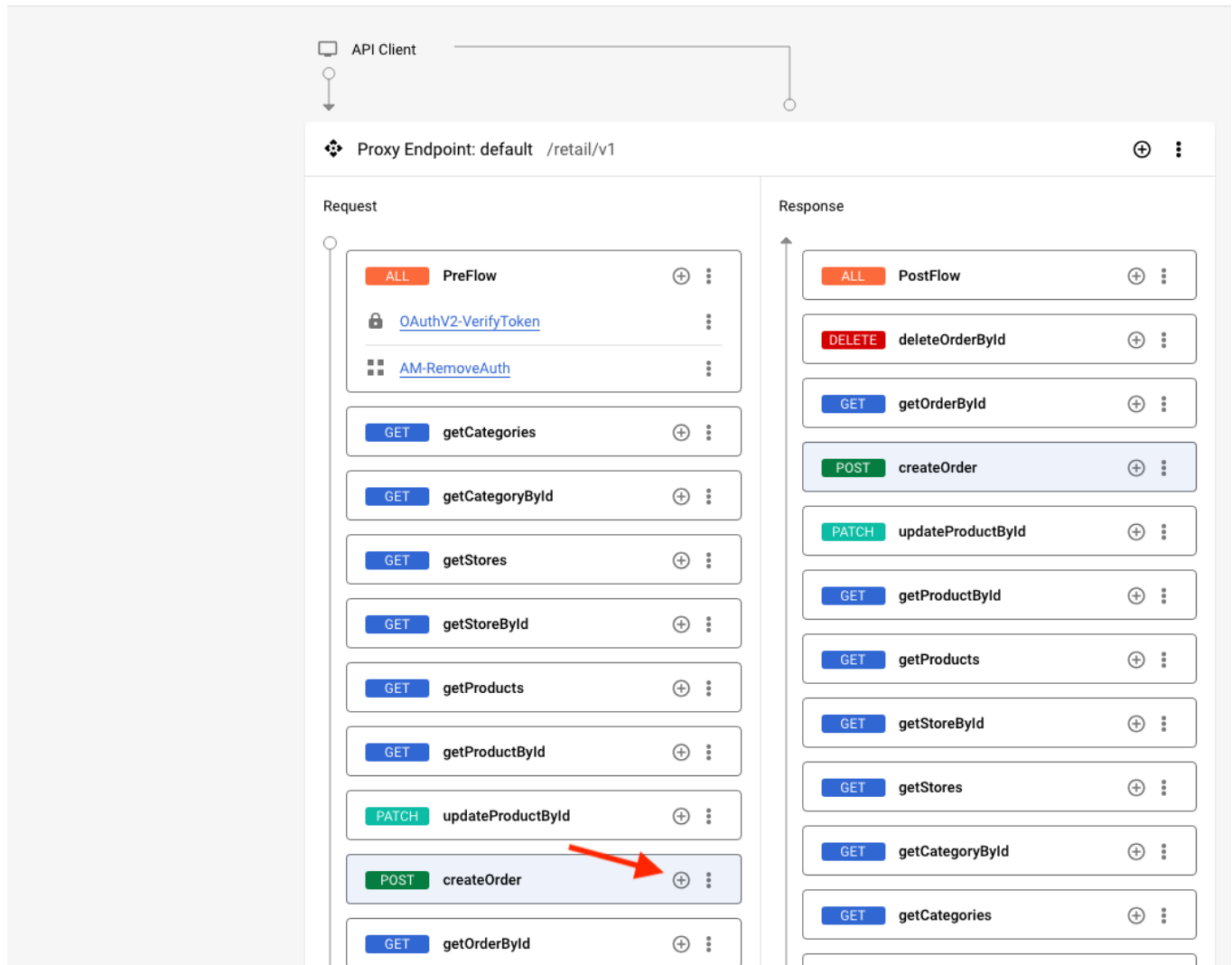
1. In the Google Cloud console, on the **Navigation menu** (≡), select **Integration Services > Apigee > Proxy Development > API proxies**.
2. Select the **retail-v1** proxy.
3. Click the **Develop** tab.

You are modifying the version of the retail-v1 proxy that was created during Labs 1 through 4.

4. Select **Proxy endpoints > default > createOrder**.

The **createOrder** flow is selected.

5. On the **Request createOrder flow**, click **Add Policy Step (+)**.



Note: The JSONThreatProtection policy checks payloads without loading the payload into a JSON parser. This policy should be run before any policies that use the JSON parser. For example, an ExtractVariables policy that uses JSONPath to retrieve information from the request should be placed later in the flow than the JSONThreatProtection policy.

6. In the **Add policy step** pane, select **Create new policy**, and then select **Security > JSON Threat Protection**.

7. Specify the following values:

Property	Value
Name	JSNTTP-Protect
Display name	JSNTTP-Protect

8. Click **Add**.
9. Click on **Policies > JSONTP-Protect**.
10. Replace the policy's default configuration with:

```
<JSONThreatProtection continueOnError="false" enabled="true"
name="JSONTP-Protect">
  <ArrayElementCount>3</ArrayElementCount>
  <ContainerDepth>2</ContainerDepth>
  <ObjectEntryCount>5</ObjectEntryCount>
  <ObjectEntryNameLength>10</ObjectEntryNameLength>
  <Source>request</Source>
  <StringValueLength>20</StringValueLength>
</JSONThreatProtection>
```

content_co

The **Source** is set to **request**, so the incoming request's payload will be validated.

The numbers indicate limits that should be enforced for the structure of JSON payloads. For example, **ContainerDepth** indicates how many levels deep your objects and arrays can go. An object (1) containing an array (2) which contains objects (3) would have a ContainerDepth of 3. This policy configuration only allows a ContainerDepth of 2, so the request would be rejected.

To understand each of the parameters, refer to the JSONThreatProtection policy documentation.

Note: The values for this policy are set very low so that you can easily test JSON payloads against the limits. In practice, your values would be significantly higher.

11. To save the updates, click **Save**, and then click **Save as New Revision**.
12. Click **Deploy**.
13. To specify that you want the new revision deployed to the eval environment, select **eval** as the **Environment**, and then click **Deploy**.
14. Click **Confirm**.

Task 2. Store the API key in a shell variable

In this task, you store the application's API key in a shell variable.

Wait for app to be available

- Navigate to **Distribution > Apps**.

Check runtime status

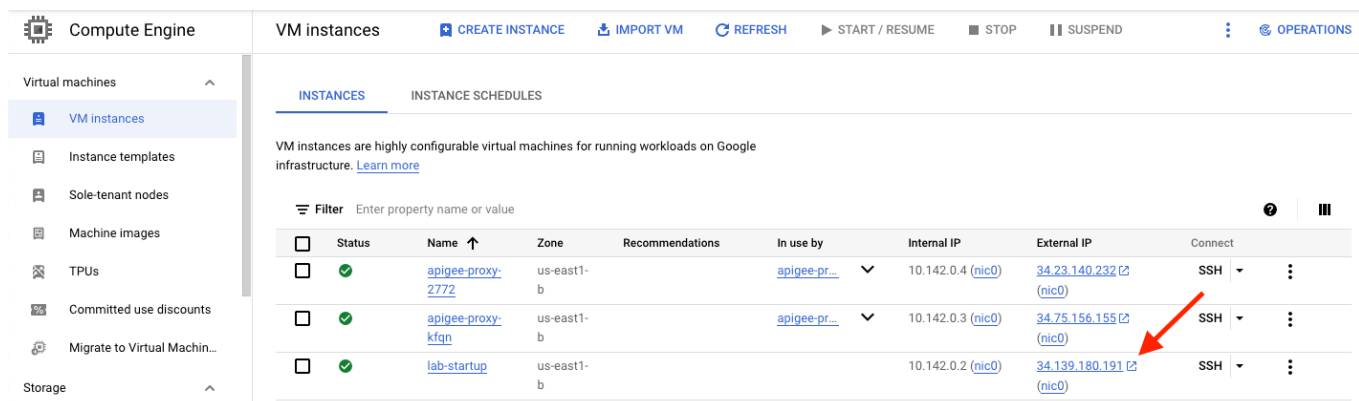
Certain assets, including API products, developers, developer apps, and KVMs, cannot be saved until the runtime is available.

For example, when navigating to the API products page, you might see an error message that reads "Products were not loaded successfully."

This is an error you should see when you are waiting for the runtime instance to be available. Once the runtime is available, refreshing the page will remove the error.

If you get this type of error, you can check the status of provisioning by navigating to the Lab Startup Tasks dashboard at the external IP address of the **lab-startup** VM instance.

1. In the Google Cloud Console navigate to **Compute Engine > VM instances**.
2. To open the Lab Startup Tasks dashboard, click on the **External IP** for the **lab-startup** VM.



Compute Engine									
VM instances									
CREATE INSTANCE IMPORT VM REFRESH START / RESUME STOP SUSPEND OPERATIONS									
INSTANCES INSTANCE SCHEDULES									
VM instances are highly configurable virtual machines for running workloads on Google infrastructure. Learn more									
Filter Enter property name or value									
<input type="checkbox"/>	Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP	Connect	
<input type="checkbox"/>	✓	apigee-proxy-2772	us-east1-b		apigee-pr...	10.142.0.4 (nic0)	34.23.140.232 (nic0)	SSH	⋮
<input type="checkbox"/>	✓	apigee-proxy-kfqj	us-east1-b		apigee-pr...	10.142.0.3 (nic0)	34.75.156.155 (nic0)	SSH	⋮
<input type="checkbox"/>	✓	lab-startup	us-east1-b			10.142.0.2 (nic0)	34.139.180.191 (nic0)	SSH	⋮

3. If you see a redirect notice page, click the link to the external IP address.

A new browser window will open. Lab startup tasks are shown with their progress.

- *Create proxies, shared flows, target servers* should be complete when you first enter the lab, allowing you to use the Apigee console for tasks like proxy editing.
- *Create API products, developers, apps, KVMs, KVM data* indicates when the runtime is available and those assets may be saved.
- *Proxies handle API traffic* indicates when the eval environment has been attached to the runtime and the deployed proxies can take runtime traffic.

- Lab Startup Tasks -			
Progress	Time	State	Task
<div></div>	05:02	completed	Create proxies, shared flows, target servers (environment available)
<div></div>	30:49	completed	Create API products, developers, apps, KVMs, KVM data (runtime is available)
<div></div>	31:11	started	Proxies handle API traffic (environment attached to runtime)
<div></div>	03:34	completed	Provide access to lab
<div></div>	30:05	started	Full provisioning of Apigee org qwiklabs-gcp-02-d23d90c73c5a in us-west4
<div></div>	01:41	completed	Create Apigee load balancer at api-test-qwiklabs-gcp-02-d23d90c73c5a.apigee-api
<div></div>	00:14	completed	Connect load balancer to runtime instance

**Before continuing, you need to wait for Create API products, developers, apps, KVMs, KVM data to complete.*

While you are waiting

- JSONThreatProtection policy
- XMLThreatProtection policy — This policy provides the same type of protection for XML payloads

Store the app's key in a shell variable

The API key may be retrieved directly from the app accessible on the **Publish > Apps** page. It can also be retrieved via Apigee API call.

- In **Cloud Shell**, run the following command:

```
export API_KEY=$(curl -q -s -H "Authorization: Bearer $(gcloud auth
print-access-token)" -X GET
"https://apigee.googleapis.com/v1/organizations/${GOOGLE_CLOUD_PROJECT
app" | jq --raw-output '.credentials[0].consumerKey'); echo "export
API_KEY=${API_KEY}" >> ~/.profile; echo "API_KEY=${API_KEY}"
```

content_co

This command retrieves a Google Cloud access token for the logged-in user, sending it as a Bearer token to the Apigee API call. It retrieves the **retail-app** app details as a JSON response, which is parsed by **jq** to retrieve the app's key. That key is then put into the **API_KEY** environment variable, and the export command is concatenated onto the **.profile** file which runs automatically when starting a Cloud Shell tab.

Note: If you run the command and it shows **API_KEY=null**, the runtime instance is probably not yet available.

Task 3. Send requests and modify the limits

In this task, you send requests and relax the limits of your policy until the request is allowed past the JSONThreatProtection policy.

Send a request

1. In Cloud Shell, execute this curl command:

```
curl -H "Content-Type: application/json" -H "apikey: ${API_KEY}"
-X POST "https://api-
test-${GOOGLE_CLOUD_PROJECT}.apiservices.dev/retail/v1/orders" -
d \
'{
```

content_co

```
"orderNumber": 342345,
"lineItems": [
  { "productId": "ME089LLA", "quantity": 1 },
  { "productId": "MD388LLA", "quantity": 2 }
],
"promisedDeliveryDate": "30 Oct 2024",
"deliveryNotes": "If not home, please place inside backyard
gate",
"destination": {
  "addressType": "home",
  "address": {
    "streetAddr1": "1 Main St."
  }
}
}' | json_pp
```

This returns the error message *Invalid ApiKey for given resource*.

Joe's app is associated with the read-only API product. We are now trying to do a POST, which requires the full access product.

2. Return to the Apigee console tab.
3. Navigate to **Distribution > Apps**.
4. Select **Joe's retail app**.

Update the app's API product

1. Click **Edit**.
2. In the **Credentials** section, for **Credential**, click the edit icon (✎).
3. In the **Edit Credential** pane, for **Product 1**, select **retail (full access)**.
4. To update the credential, click **Done**.
5. Click **Save**.

Update the policy configuration until the request passes

1. Navigate to **Proxy development > API Proxies**.
2. Select the **retail-v1** proxy, and then click the **Develop** tab.
3. Click **Policies > JSONTP-Protect**.
4. In Cloud Shell, resubmit this command:

```
curl -H "Content-Type: application/json" -H "apikey: ${API_KEY}" \
-X POST "https://api-
test-${GOOGLE_CLOUD_PROJECT}.apiservices.dev/retail/v1/orders" -
d \
'{
  "orderNumber": 342345,
  "lineItems": [
    { "productId": "ME089LLA", "quantity": 1 },
    { "productId": "MD388LLA", "quantity": 2 }
  ],
  "promisedDeliveryDate": "30 Oct 2024",
  "deliveryNotes": "If not home, please place inside backyard
gate",
  "destination": {
    "addressType": "home",
    "address": {
      "streetAddr1": "1 Main St."
    }
  }
}' | json_pp
```

content_co

Note: If you continue to receive the "Invalid ApiKey for given resource" error, this may be because the allowed operations for an API key are cached for about 3 minutes after they are used. If you have correctly updated the app, and you continue to try the request, you will eventually receive the error from the JSONTP-Protect policy.

You should see a 500 error that looks like this:

```
{
  "fault": {
    "detail": {
      "errorcode": "steps.jsonthreatprotection.ExecutionFailed"
    },
    "faultstring": "JSONThreatProtection[JSONTP-Protect]: Execution fai
  }
}
```

Note: The default status code for the error, 500 Internal Server Error, is typically not appropriate, because this error is not a server error.

The policy is generally used to validate incoming JSON payloads, so rewriting the status code to 400 Bad Request is usually appropriate.

The **object entry name** on line 2 is **orderNumber**, which is 11 characters long. Looking at the policy configuration, the **ObjectEntryNameLength** is set to 10. Note that only the first error is returned when a limit is exceeded, even though there might be multiple errors. The longest object entry name in the payload is **promisedDeliveryDate**, which is 20 characters long.

Note: You typically should not set limits based exactly on the maximum values that exist now. It is better to be less strict in the limits, so that future valid payloads are less likely to be rejected.

5. Set the **ObjectEntryNameLength** in the **JSONThreatProtection** policy to 30.
6. Click **Save**, and then click **Save as New Revision**.
7. Click **Deploy**.
8. To specify that you want the new revision deployed to the eval environment, click **Deploy**, and then click **Confirm**.

Wait for the updated proxy to be deployed.

9. Click **Policies > JSONTP-Protect**.

10. Resubmit the curl command.

You should get a new error:

```
{
  "fault": {
    "detail": {
      "errorcode": "steps.jsonthreatprotection.ExecutionFailed"
    }
    "faultstring": "JSONThreatProtection[JSONTP-Protect]: Execution
failed. reason: JSONThreatProtection[JSONTP-Protect]: Exceeded container
depth at line 4",
  }
}
```

The maximum depth in the payload is 3 (object containing an array which contains objects), but the policy only allows a container depth of 2.

11. Set the **ContainerDepth** in the **JSONThreatProtection** policy to 5.

If you look at the payload, you can also see that the maximum string length of 20 is exceeded by the deliveryNotes field.

12. Set the **StringValueLength** to 100.

13. Click **Save**, and then click **Save as New Revision**.

14. Click **Deploy**.

15. To specify that you want the new revision deployed to the eval environment, click **Deploy**, and then click **Confirm**.

Wait for the updated proxy to be deployed.

16. Resubmit the curl command.

This time, you should receive a 200 response that resembles this:

```
{  
  "name": "-MXBcl-WqjD8tr_Z_9th"  
}
```

The request successfully passed the JSONThreatProtection policy and was sent to the backend.

17. You can retrieve the order by sending the following curl call, with *REPLACE* set to the name that was returned by the previous request:

```
curl -H "apikey: ${API_KEY}" -X GET "https://api-  
test-${GOOGLE_CLOUD_PROJECT}.apiservices.dev/retail/v1/orders/REPLA content_co  
| json_pp
```

Congratulations!

In this lab, you protected the createOrders resource against malicious JSON by using the JSONThreatProtection policy and configuring it with the correct limits.

End your lab