From Code to Compromise:
Analyzing Threats in GCP Cloud Functions

Anirban Das

- **Security Engineer** at **ExaWizards Inc.**
- Former **Security Researcher** at **Zscaler**
- **Loves** Threat Detection & Deception, Security Research (Cloud, DevSecOps, IdPs etc)
- **Loves** travelling and playing games.

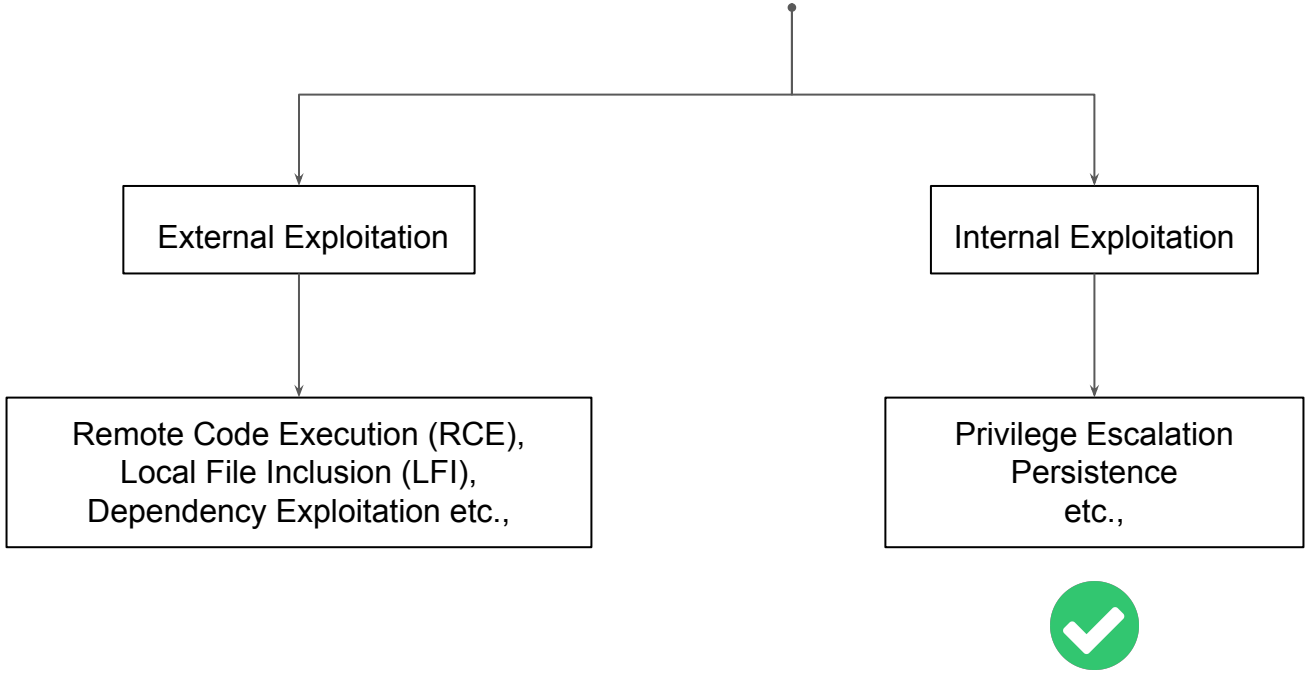
We will be integrating Cloud Functions into Cloud Run UI in the upcoming months. GO TO CLOUD RUN

From Code to Compromise:
Analyzing Threats in GCP Cloud Run Functions

Overview

1. Cloud Run Function Overview
2. Services and APIs
3. Cloud Run Function Build Process (Gen-1 and Gen-2)
4. Points of Attack
5. Hiding the Source Code
6. Detection

Cloud Run Function Abuse



What is Cloud Run Function?

A Google Cloud Run Function is a **serverless**, **event-driven** compute service that allows you to run **code** in response to specific events, known as **triggers**, from an **event source**.

File Upload in Cloud Storage

Message Publish in Pub/Sub

HTTP Request to the function's endpoint etc.,

Cloud Storage, Pub/Sub, or HTTP requests

Scenario

You want to allow users to **upload images** to your platform.

However, to maintain a consistent look and feel across your website, all uploaded images need to be **resized** and optimized before they are displayed on the site.

Solution

1. Deployment:

   Deploy a Cloud Run Function that is configured to listen for a **Trigger Event** from a **Cloud Storage Bucket**.

   The trigger event is a **File Upload** to the specified **Cloud Storage Bucket**.
   The function will automatically execute whenever a new image is uploaded.

2. User Action: A user **uploads** an image to your platform.

3. Event Source: The image is uploaded to a specific **Cloud Storage Bucket**.

4. Trigger: The upload event **triggers** the Cloud Run Function automatically.

5. **Function Execution**:

   The function executes code to resize the image to the desired dimensions and saves the optimized image back into a Cloud Storage Bucket.

6. Outcome: The resized and optimized image is now ready to be displayed on your website.

- Cloud Run Function have two types of Environment: **Generation 1** and **Generation 2**

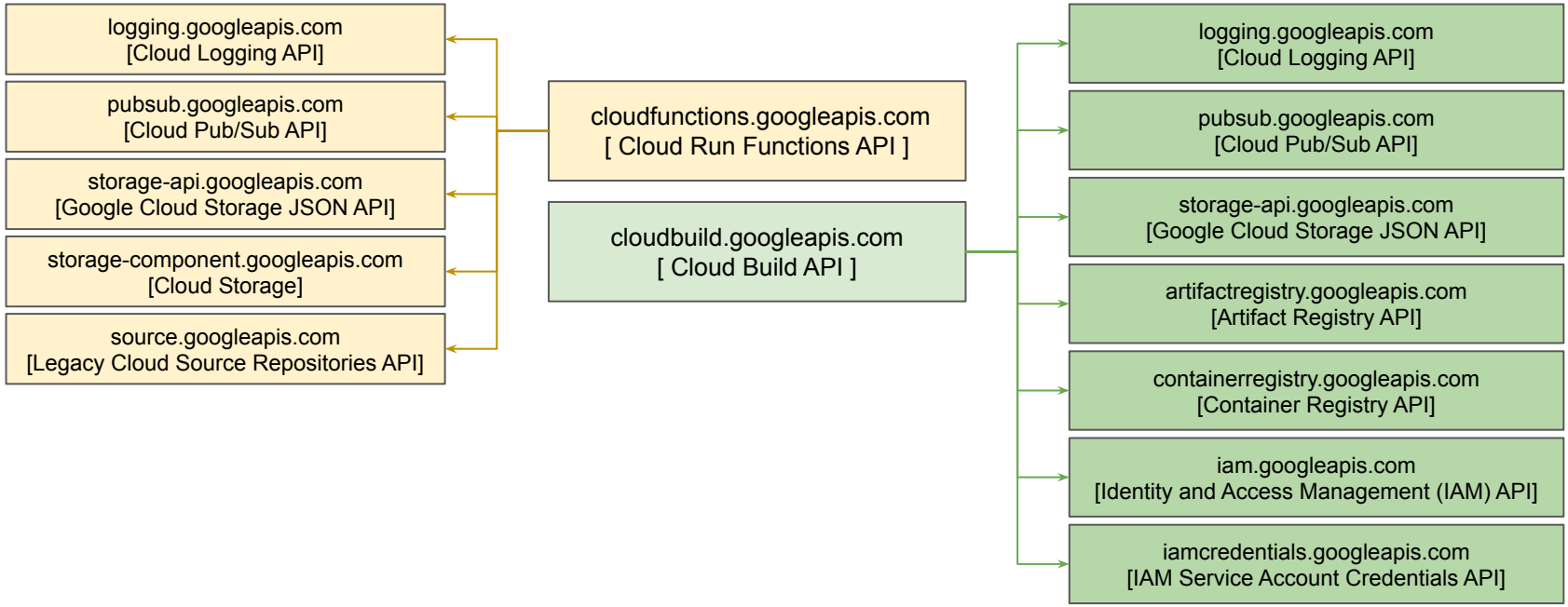
- Cloud Run Functions can be deployed / updated using **source code** from three sources:

  Local machine, **Cloud Storage Bucket**, or **Cloud Source Repository**.

Services / APIs that need to be enabled (Gen - 1)

cloudfunctions.googleapis.com	Cloud Run Functions API
cloudbuild.googleapis.com	Cloud Build API

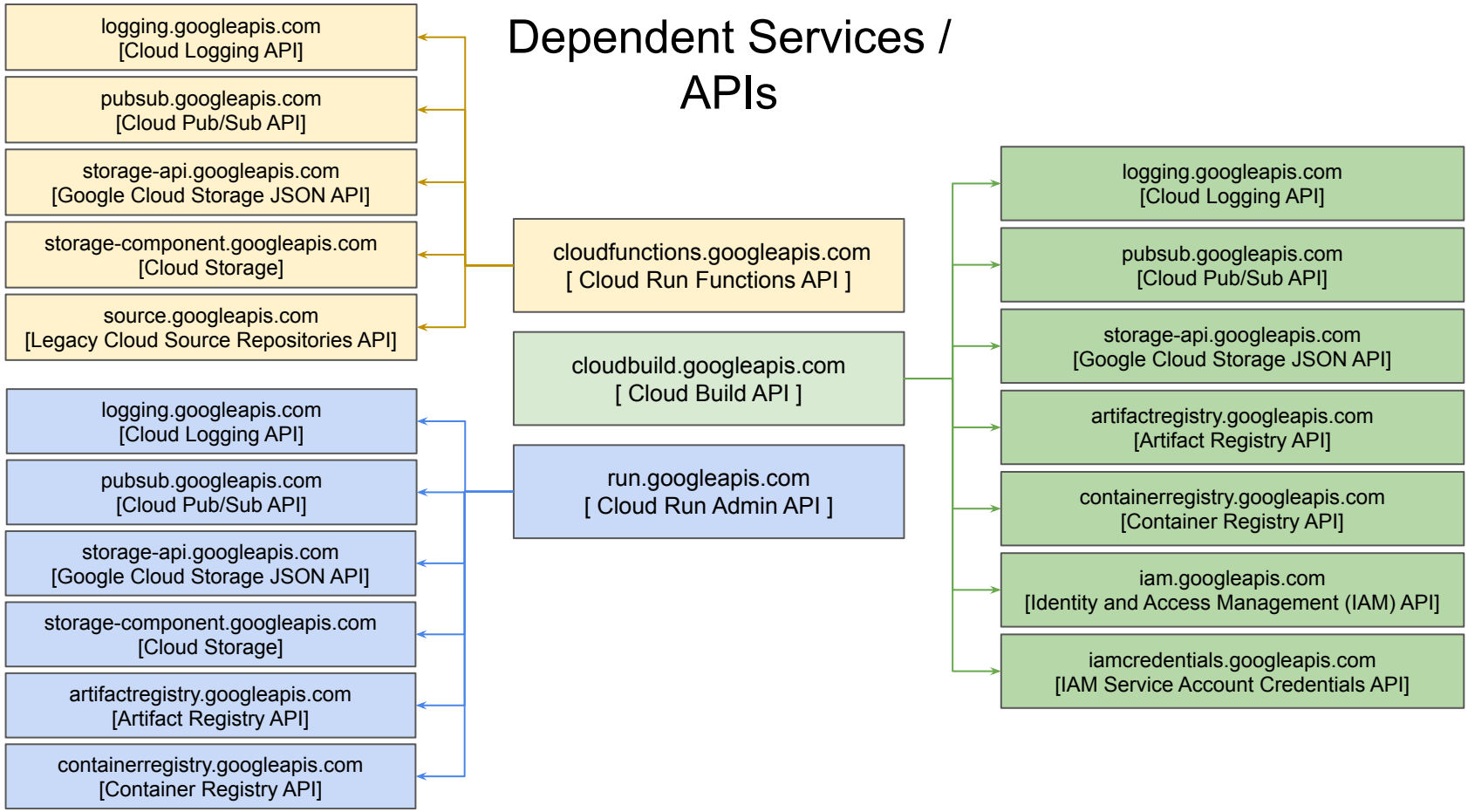
Dependent Services / APIs



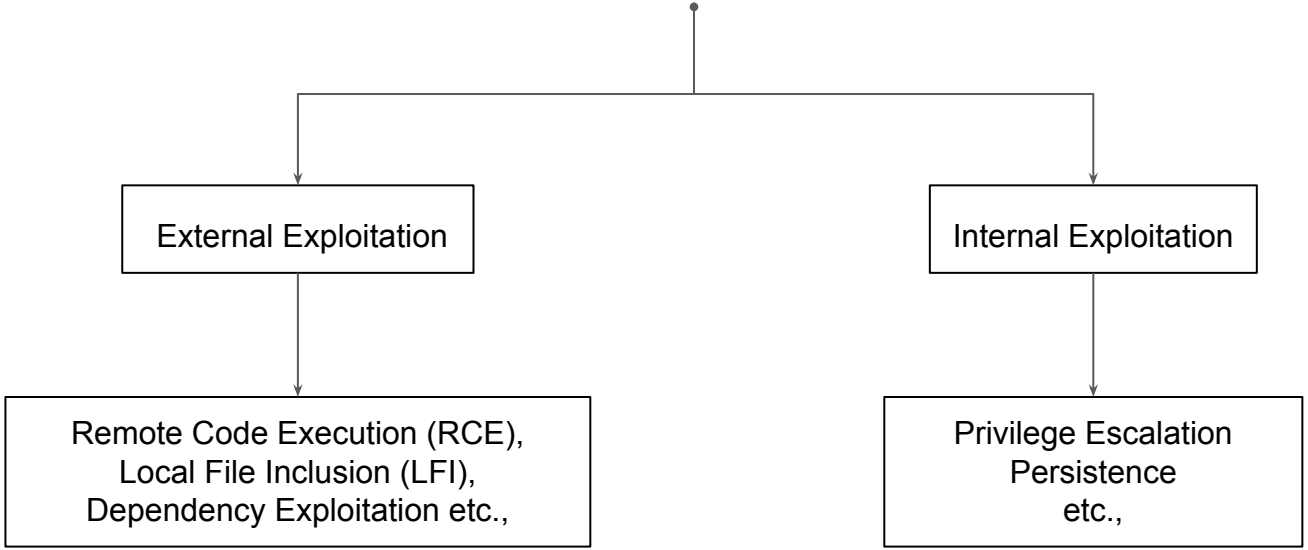
Services / APIs that need to be enabled (Gen - 2)

cloudfunctions.googleapis.com	Cloud Run Functions API
cloudbuild.googleapis.com	Cloud Build API
run.googleapis.com	Cloud Run Admin API

Dependent Services / APIs



Cloud Run Function Abuse



Privilege Escalation:

Deploying a Cloud Run Function which queries IMDS for **Runtime Service Account's** Access Token and prints it as an output.

Persistence:

Deploying a Cloud Run Function which creates new service accounts when a certain event (Changes in a Cloud Storage bucket, Pub/Sub messages etc) triggers it.

Questions:

1. Is uploading malicious code the only way to abuse Cloud Run Functions?

2. Is there any way to abuse other Services which Cloud Run Function interacts with?

3. Is Runtime Service Account the only Service Account that comes in play?

Answers

1. Is uploading malicious code the only way to abuse Cloud Run Functions?

   No, you can use Dependency management file for abuse as well. (*Check Referenced Link*)

2. Is there any way to abuse other Services which Cloud Run Function interacts with?

   Yes, other services that support Cloud Run Function or Cloud Run Function depends on.

3. Is Runtime Service Account the only Service Account that comes in play?

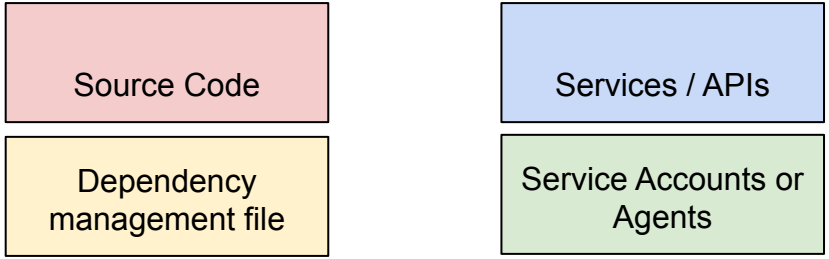
   No, there are various Service Accounts that are used.

Reference: https://www.tenable.com/blog/confusedfunction-a-privilege-escalation-vulnerability-impacting-gcp-cloud-functions

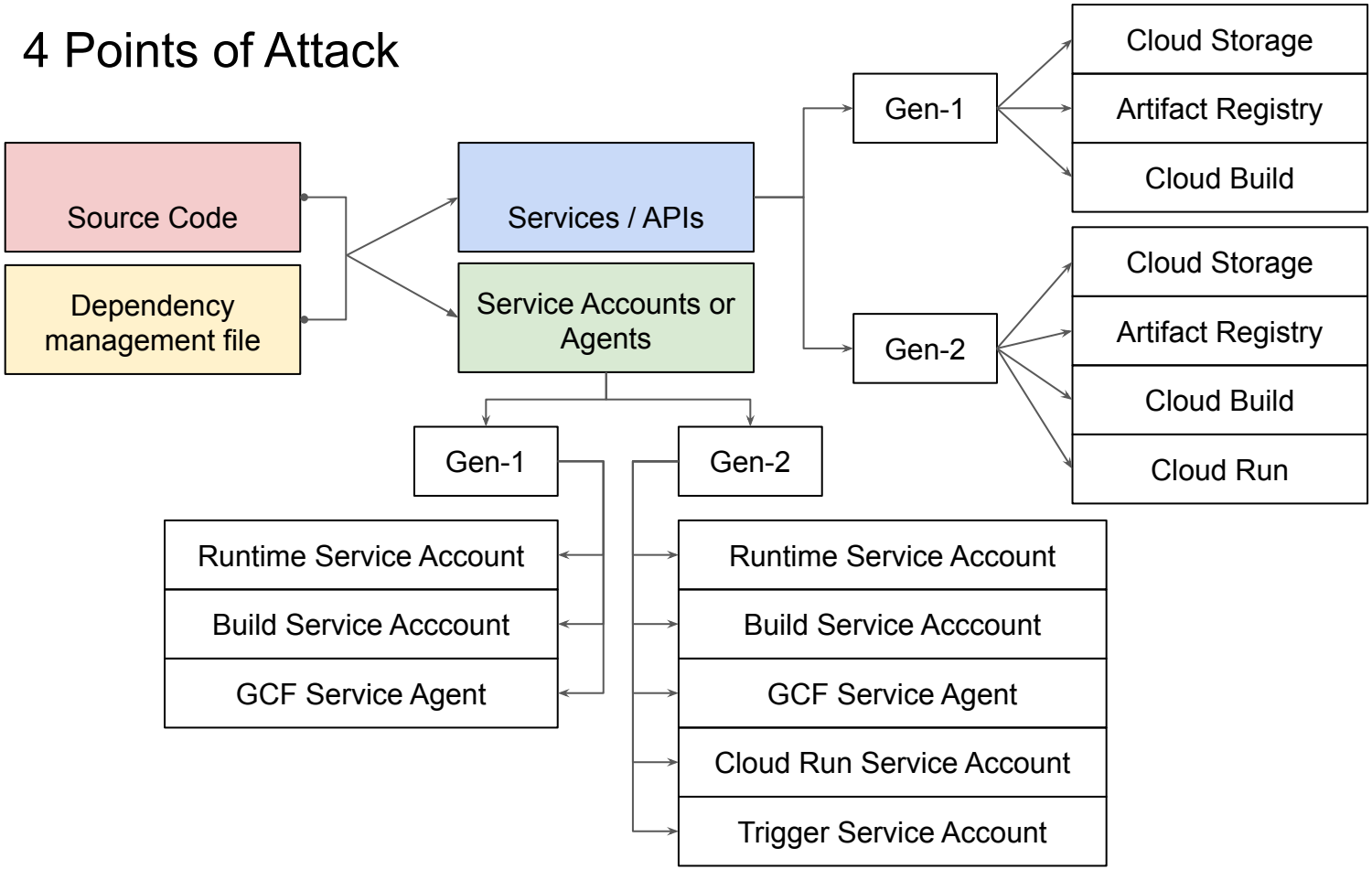
Cloud Run Function Build Process (Generation 1 and 2)

https://docs.google.com/spreadsheets/d/1i5aBG9bkJ1o8RH_FgiobFffOs_QvzcS-dSu_ozz8LhI/edit?usp=sharing

4 Points of Attack



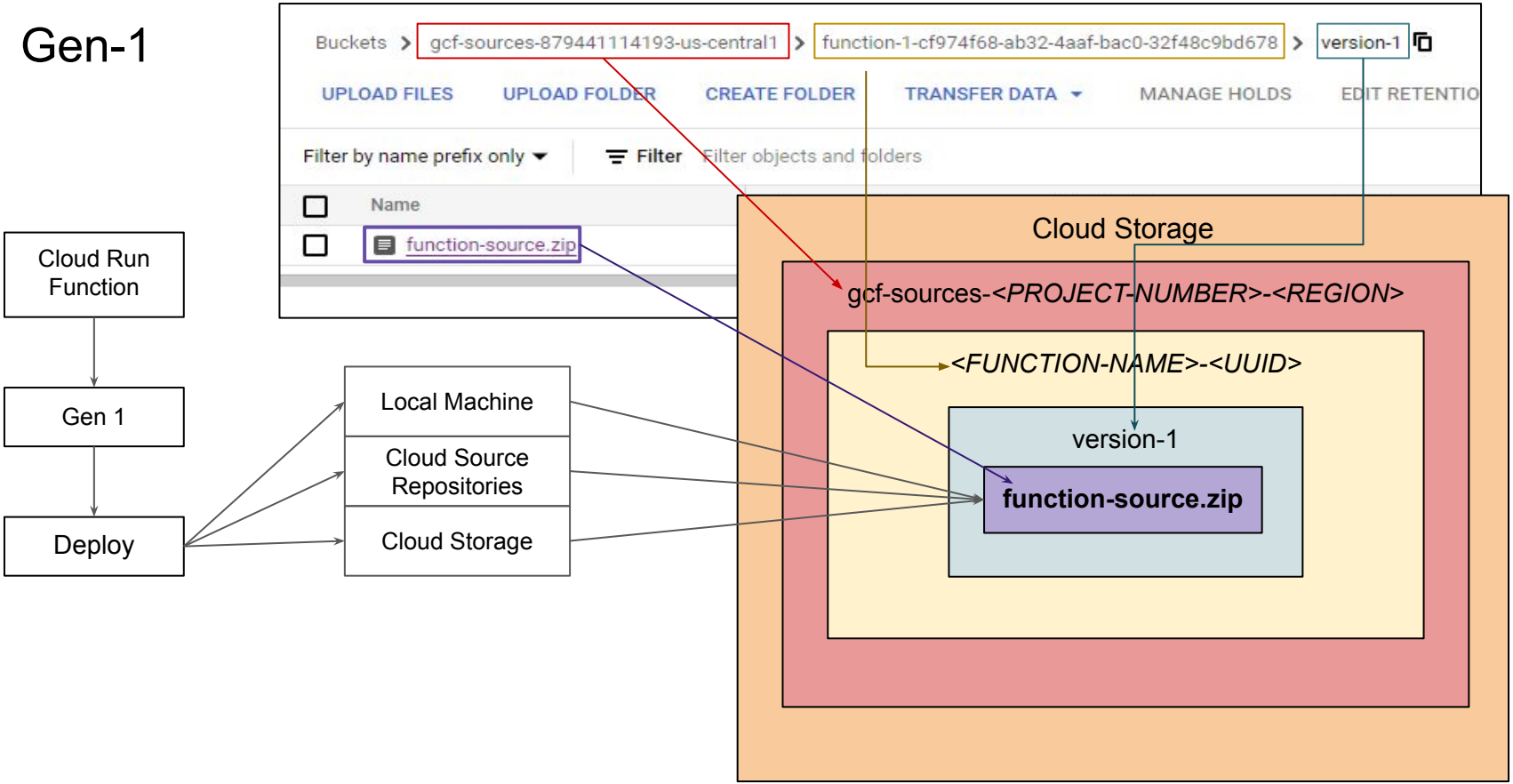
4 Points of Attack



The **Main Source** file and **Dependency Management** file are prime targets and only way for attackers to exploit Cloud Functions.

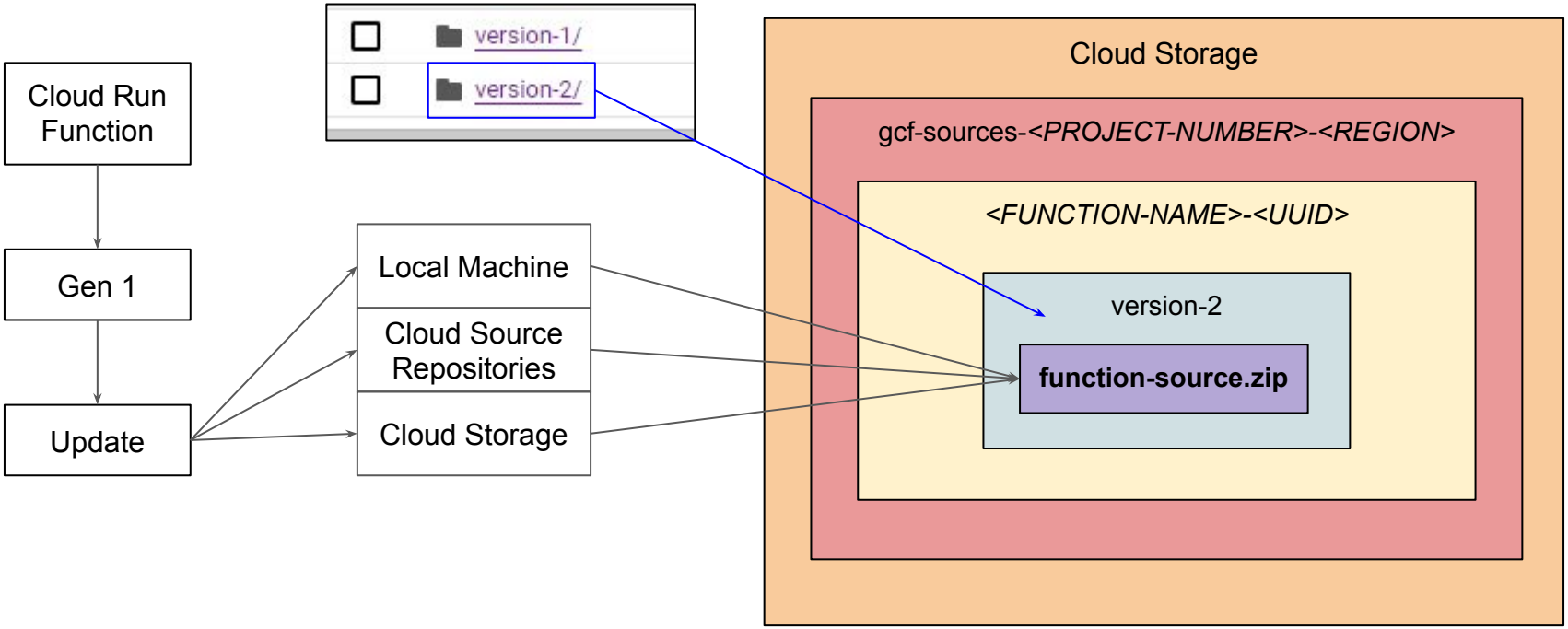
Where is the Source Code and Dependency management file stored?

Cloud Storage	Gen-1	gcf-sources-<PROJECT_NUMBER>-<REGION>/<FUNCTION_NAME>-<UUID>/version-<NUMBER>/function-source.zip *gcf-sources-879441114193-us-central1/function-1-888ebd01-a7b0-4234-99e5-1bbc89ca1377/version-1/function-source.zip*
	Gen-2	gcf-v2-sources-<PROJECT_NUMBER>-<REGION>/<FUNCTION_NAME>/function-source.zip *gcf-v2-sources-879441114193-us-central1/function-2/function-source.zip* gcf-v2-uploads-<PROJECT_NUMBER>-<REGION>/UUID.zip *gcf-v2-uploads-879441114193-us-central1/90898fa1-7db3-4ed6-ac48-cd3c3ebf76f9.zip*



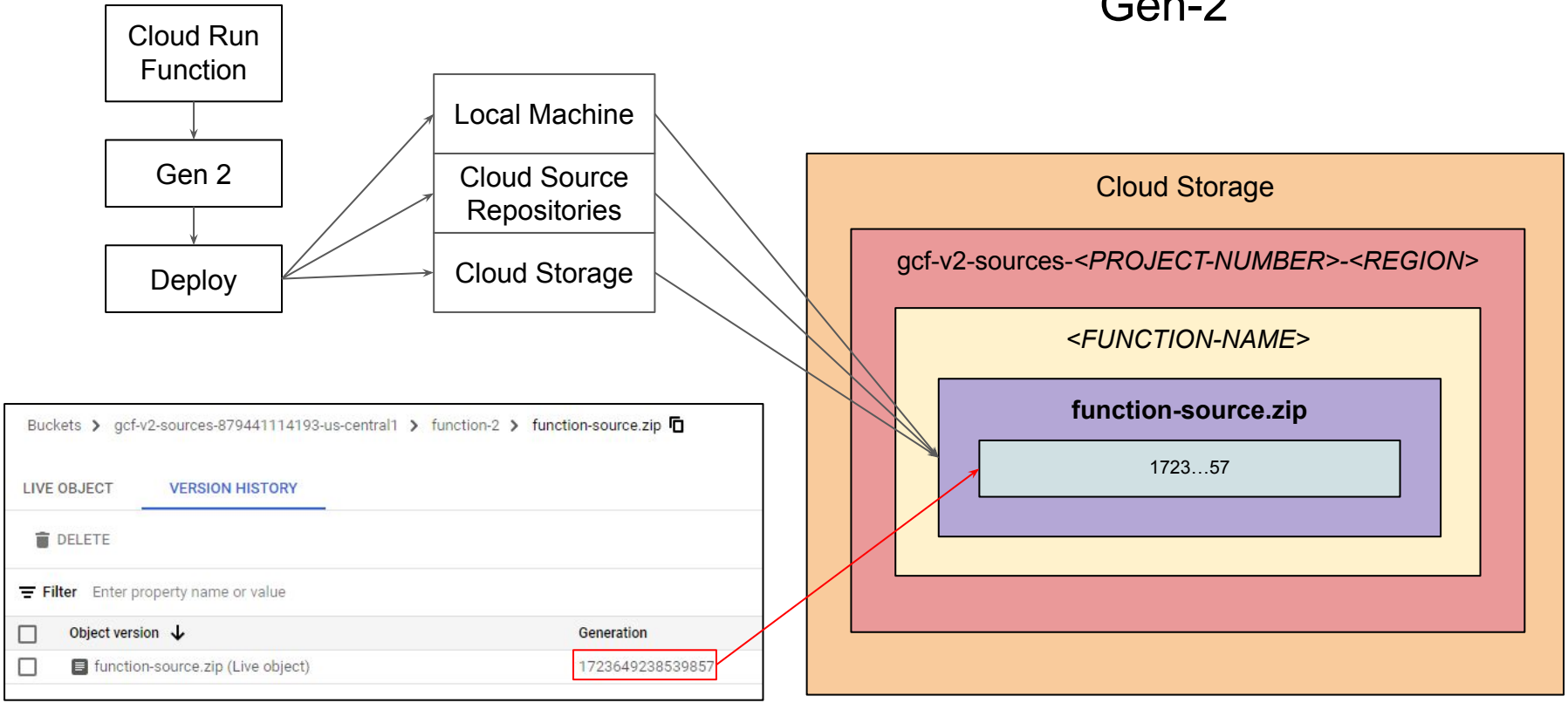
Gen - 1: **function-source.zip** versioning during function deployment.

Gen-1

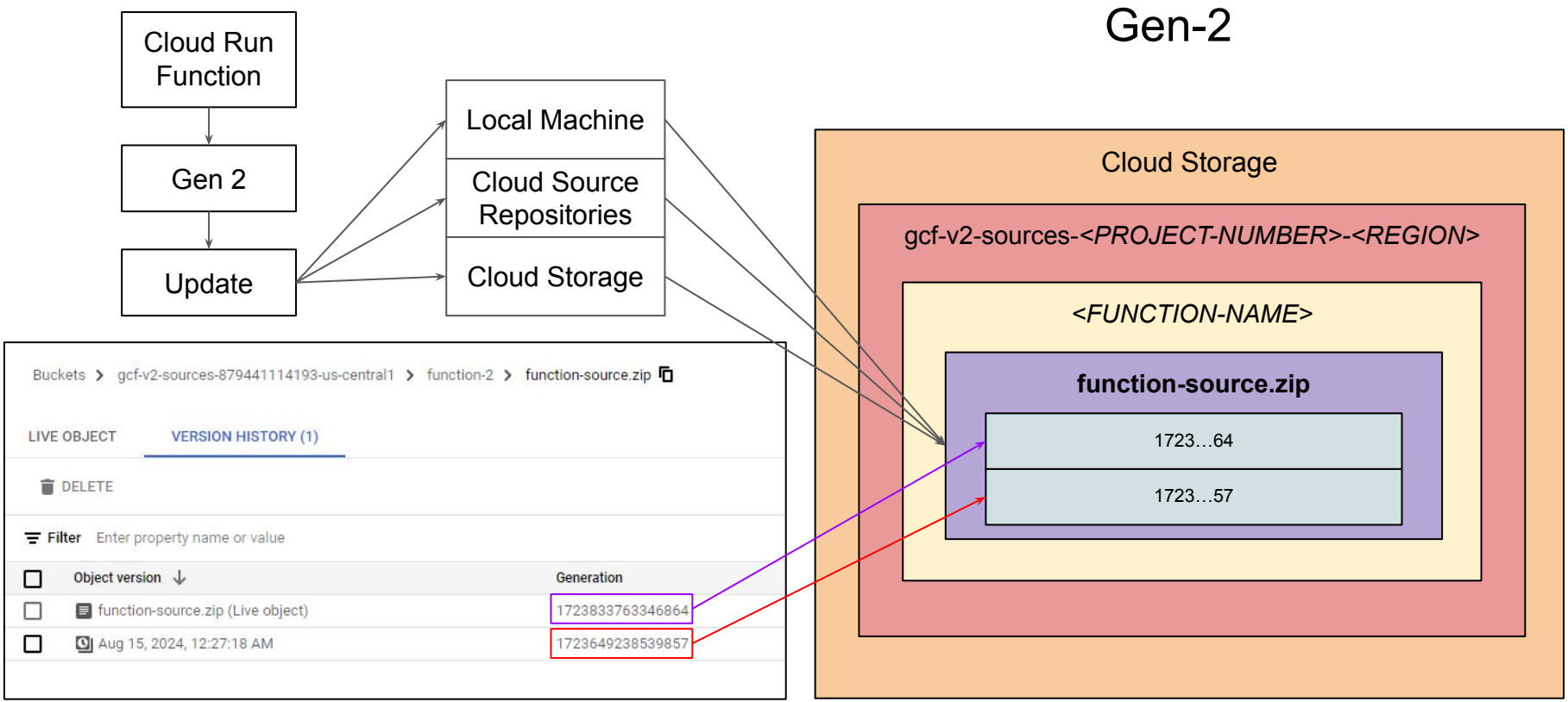


Gen - 1: **function-source.zip** versioning during function update.

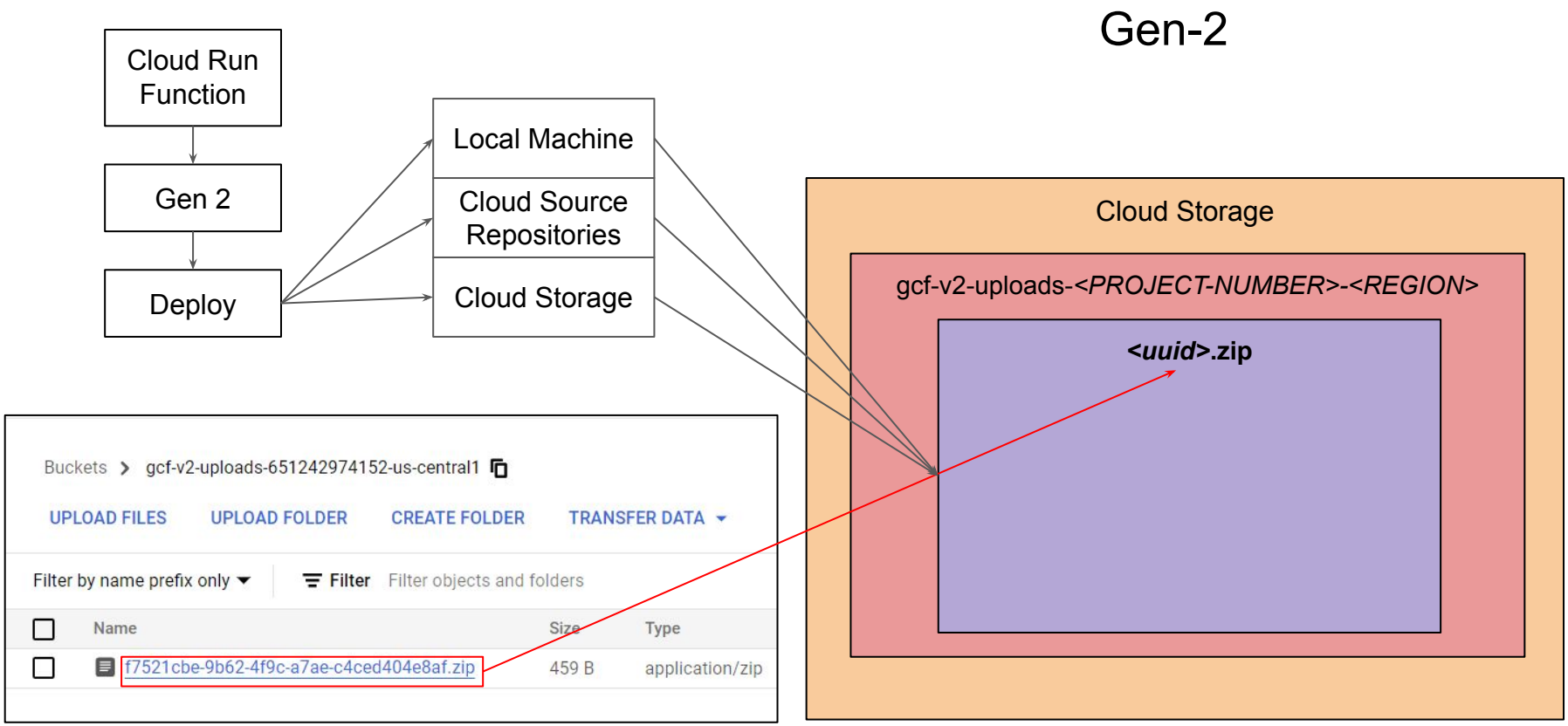
Gen-2



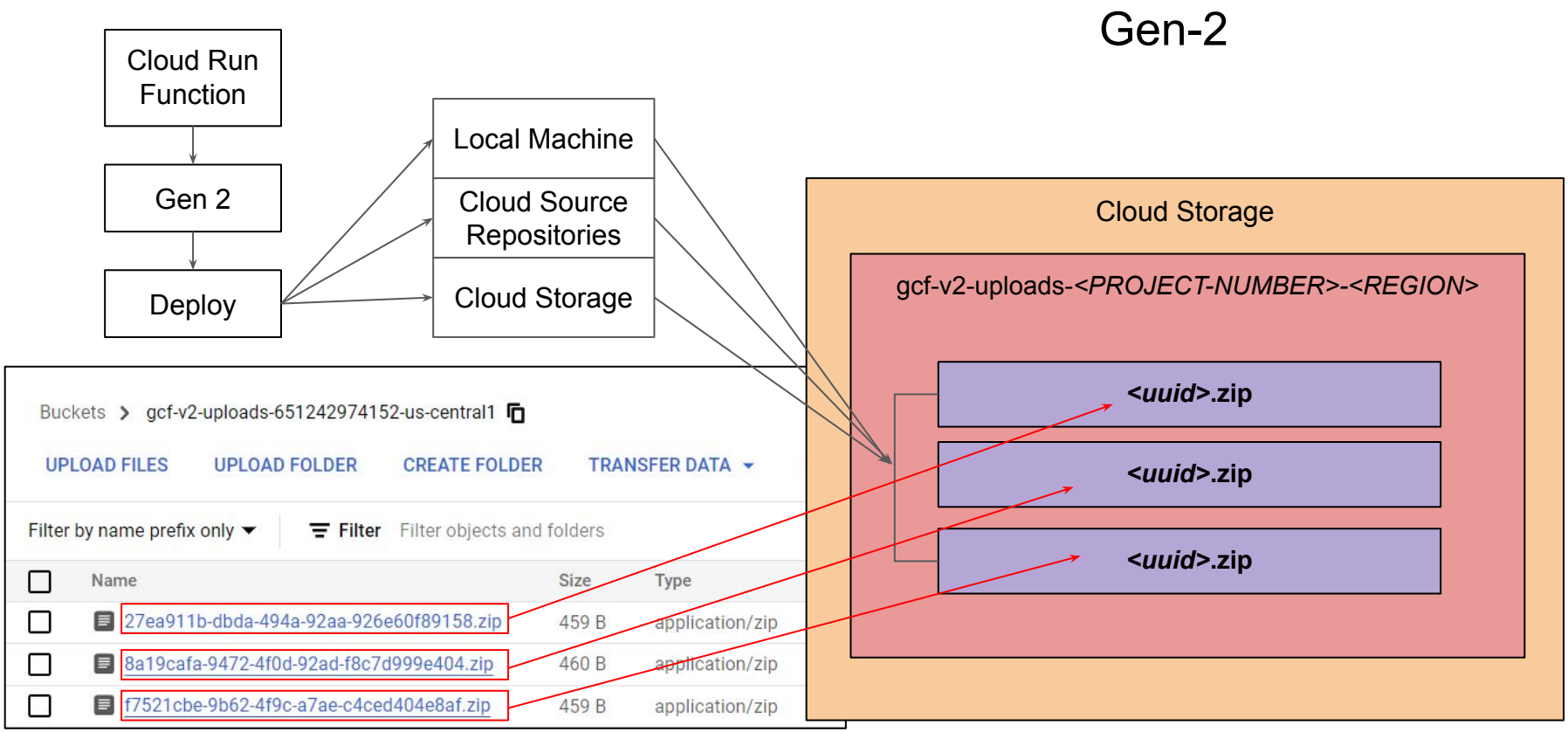
Gen - 2: **function-source.zip** versioning during function deployment.



Gen - 2: **function-source.zip** versioning during function updation.



Gen - 2: ***<uuid>*.zip** versioning during function deployment.



Gen - 2: ***<uuid>*.zip** versioning during function update.

Where is the Source Code and Dependency management file stored?

Artifact Registry	Gen-1	gcf-artifacts/<FUNCTION_NAME> *gcf-artifacts/function--1* gcf-artifacts/<FUNCTION_NAME>/cache *gcf-artifacts/function--1/cache*
	Gen-2	gcf-artifacts/<PROJECTNAME>__<REGION>__<FUNCTIONNAME> *gcf-artifacts/cloudfunc--research__us--central1__function--2* gcf-artifacts/<PROJECTNAME>__<REGION>__<FUNCTIONNAME>/cache *gcf-artifacts/cloudfunc--research__us--central1__function--2/cache*

Detection

- Analyze the source code and dependency file in function-source.zip for Gen-1.
- Analyze the source code and dependency file in function-source.zip & *uuid.zip* for Gen-2.
- Log Ingress traffic to IMDS (Instance Metadata Service - 169.254.169.254)

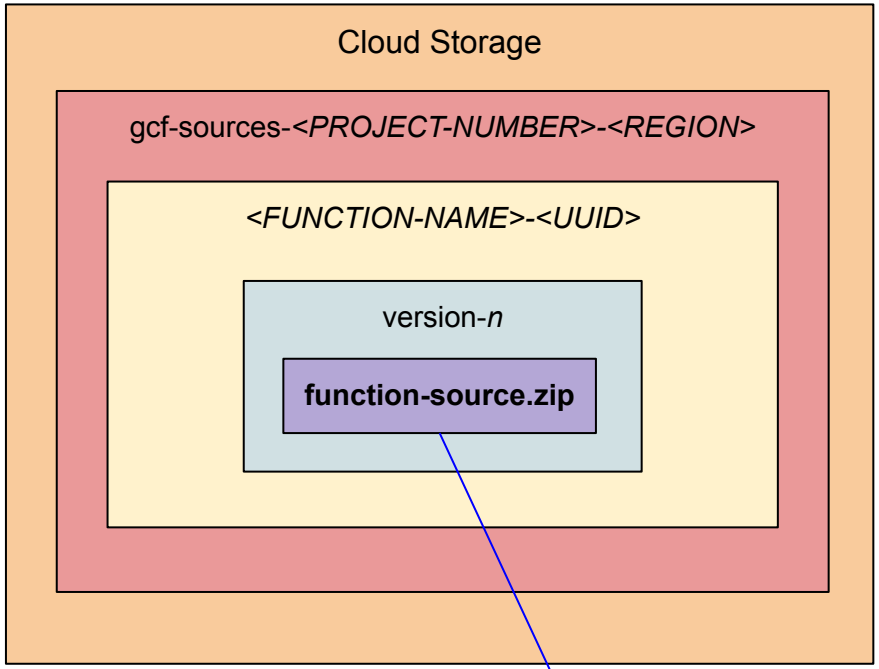
Questions:

What if the Attacker deletes the Cloud Storage **Object** (function-source.zip & <*uuid*>.zip) or whole **Bucket** (gcf-sources-<PROJECT_NUMBER>-<REGION> OR gcf-v2-sources-<PROJECT_NUMBER>-<REGION>)?

What if the Attacker deletes the **Image** in **Artifact Registry**?

Hiding the Source Code!

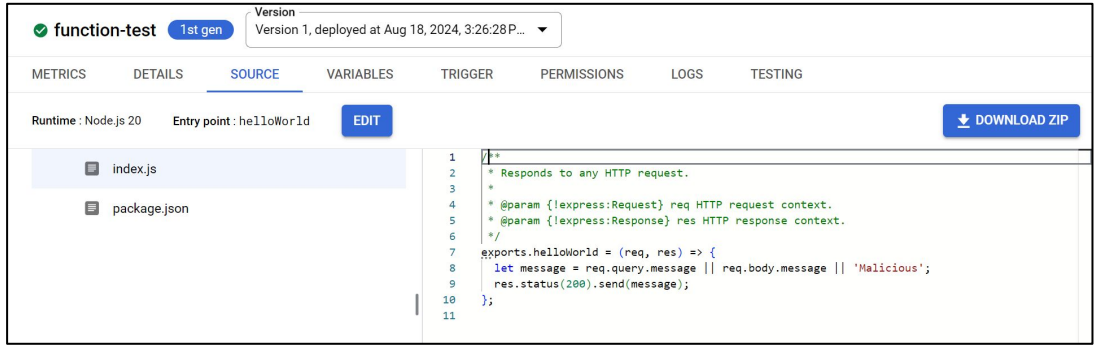
Gen-1



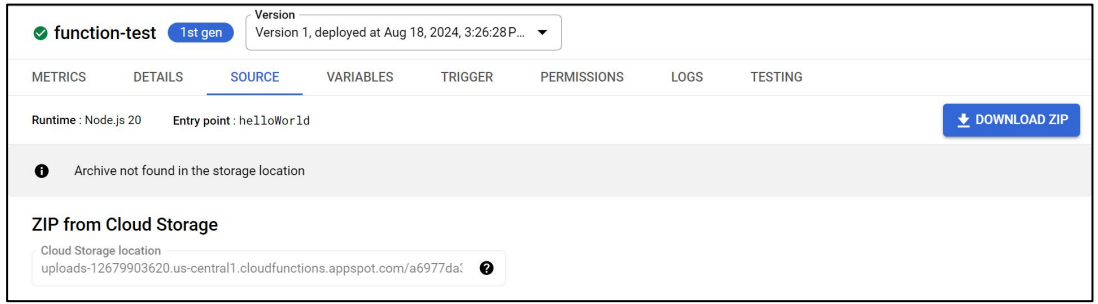
https://storage.googleapis.com/**gcf-sources-651242974152-us-central1/function-24-39a67aa2-3756-49f8-83a3-becb1095c713/version-1/function-source.zip**?GoogleAccessId=service-651242974152@gcf-admin-robot.iam.gserviceaccount.com&Expires=1723962679&Signature=YE%2Bmcv..%3D

Attack Flow - Gen 1

1. Attacker Deploys a Malicious Cloud Run Function.

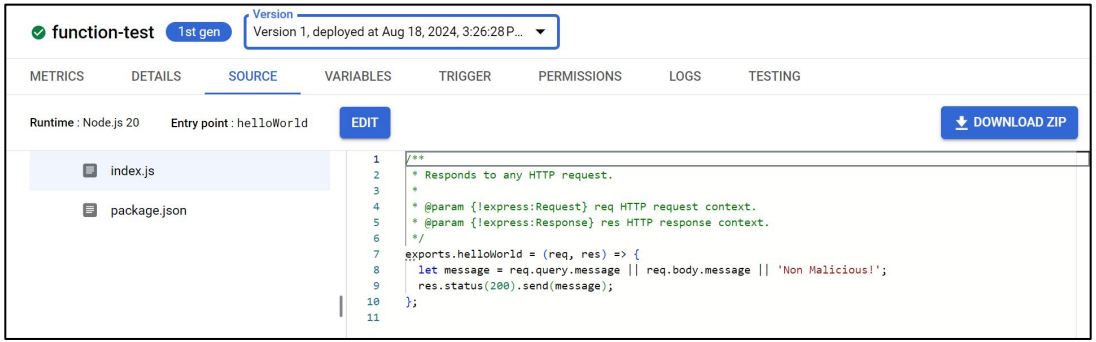


2. Attacker Deletes the Malicious **function-source.zip** object file from the *gcf-sources-<PROJECT-NUMBER>-<REGION>* Bucket.

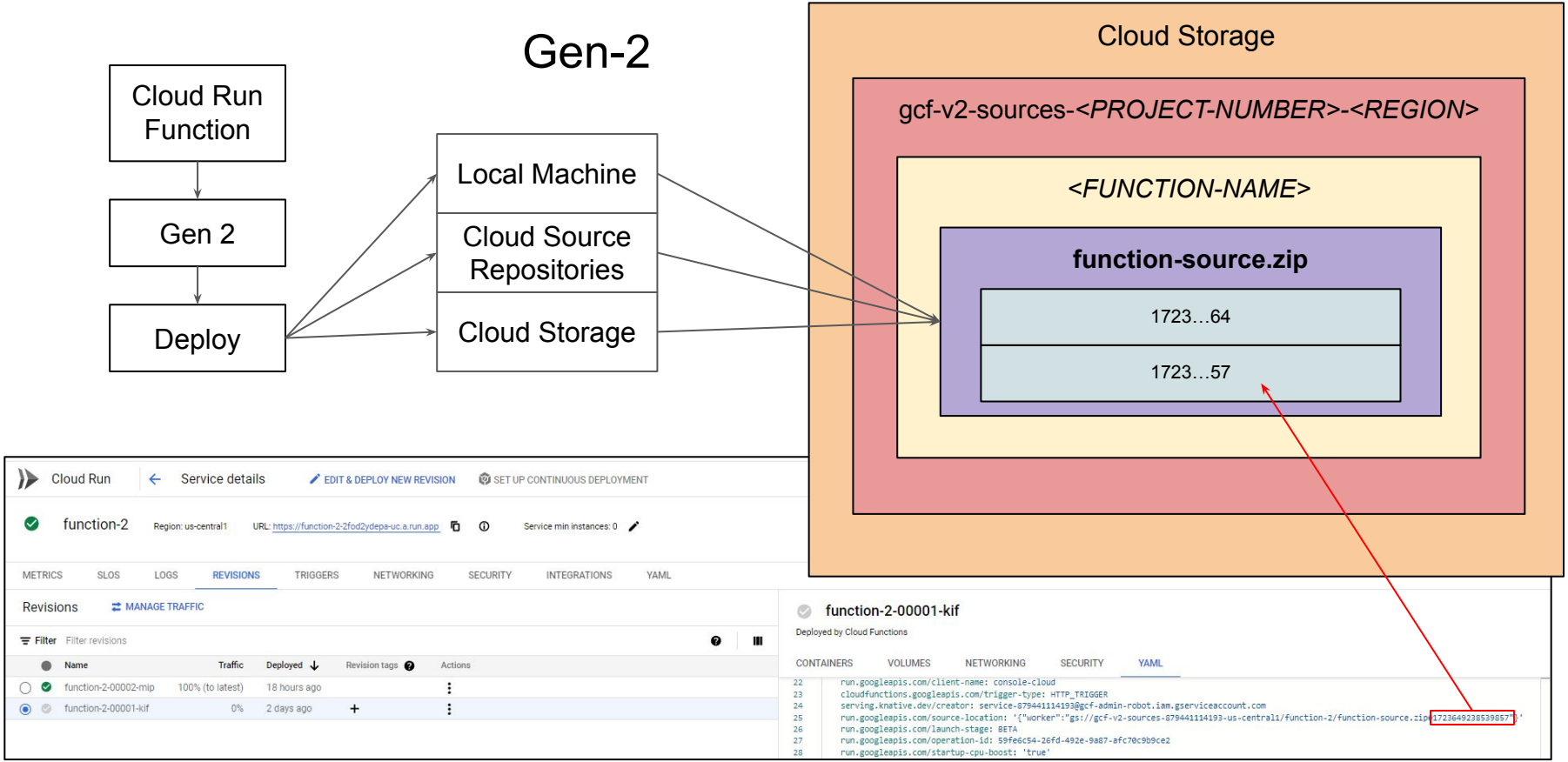


The Function still can be triggered without any problems

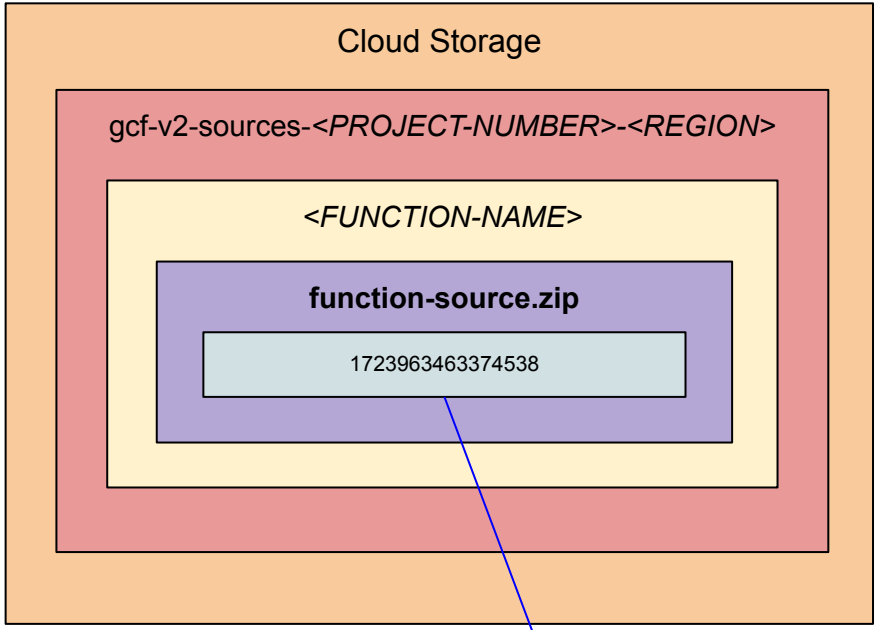
3. Attacker uploads a Non-Malicious function-source.zip object file.



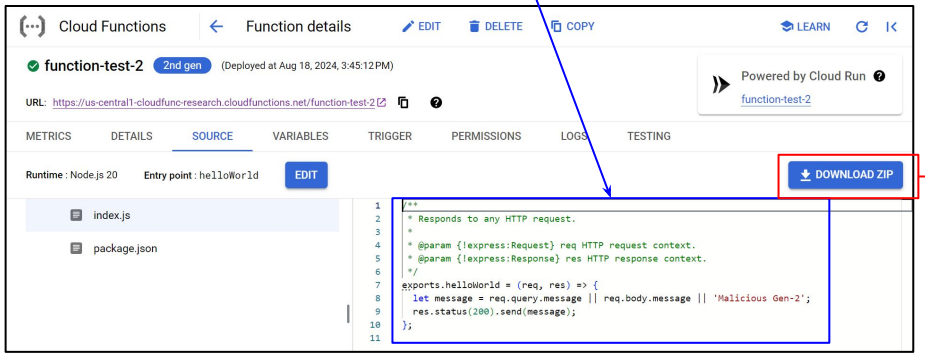
The Malicious code still executes upon trigger!



Gen - 2: **function-source.zip** generation ID being set in Cloud Run.

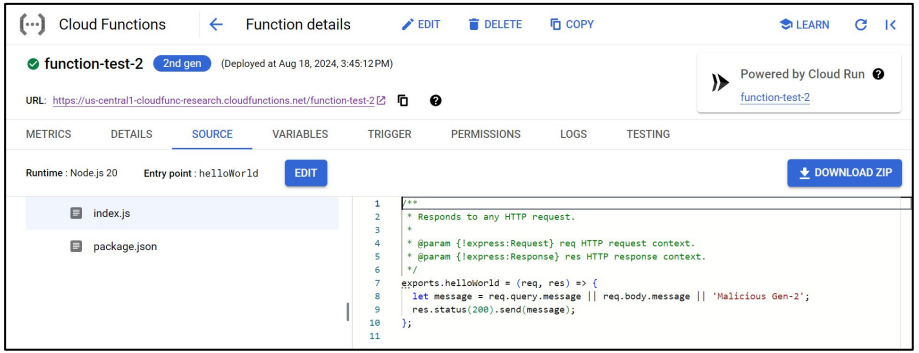


https://storage.googleapis.com/**gcf-v2-sources-651242974152-us-central1/function-test-2/function-source.zip**?GoogleAccessId=service-651242974152@gcf-admin-robot.iam.gserviceaccount.com&Expires=1723965317&Signature=xp0BnU..%3D%3D&generation=**1723963463374538**



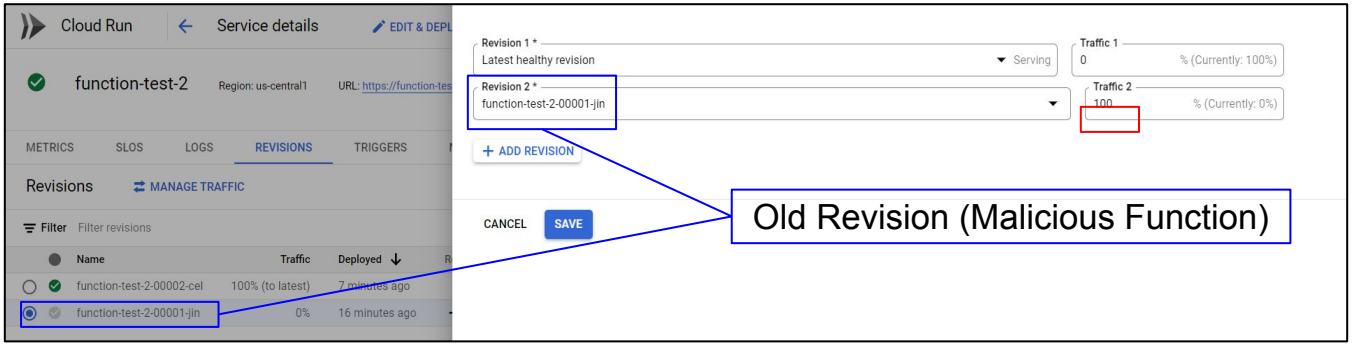
Attack Flow - Gen 2

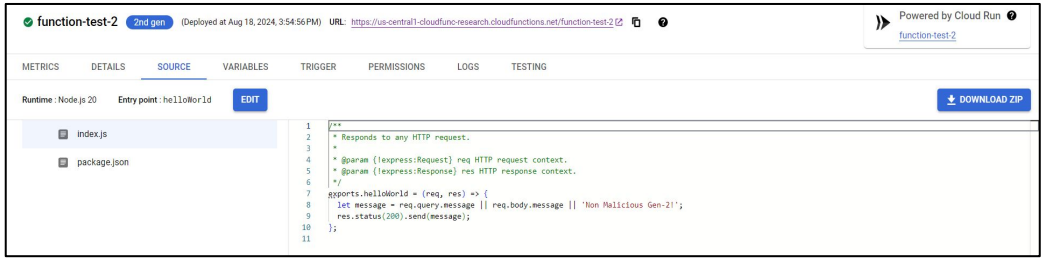
1. Attacker Deploys a Malicious Cloud Run Function.



2. Attacker Updates the Malicious Cloud Run Function with Non-Malicious Code.

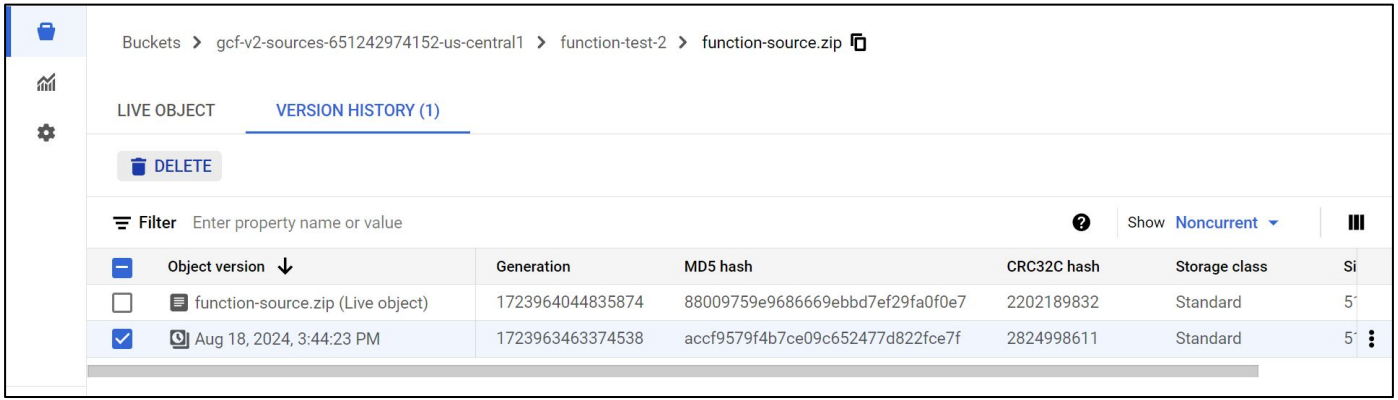
3. In Cloud Run, Attacker directs 100% of traffic from Latest Revision to Old Revision (Malicious Function).





The Malicious code executes upon trigger!

4. ***(Optional)*** Attacker Deletes the malicious object version of *function-source.zip (1723963463374538) from the gcf-v2-sources-<PROJECT-NUMBER>-<REGION>* Bucket.



Gen-1: Indicator of Attack (IOA)

Deleting and Uploading “**function-source.zip**”

- The Bucket and all of the it’s Folder & File objects “gcf-sources-*<PROJECT-NUMBER>-<REGION>/<FUNCTION-NAME>-<UUID>* /version-*<n>*/**function-source.zip**” are created, uploaded and deleted by Service Agent:

  service-*<PROJECT-NUMBER>*@gcf-admin-robot.iam.gserviceaccount.com

  Service Agents **cannot** be **Impersonated** in Google Cloud Platform. (As of August 2024)

- Therefore, if any other principal (user or service account) deletes these folder and file objects it could be an indication of potentially malicious activity.

Gen-1: Indicator of Attack (IOA)

```
-- Show malicious "function-source.zip" object deletion inside "gcf-sources-<PROJECT-NUMBER>-<REGION>" bucket.

protoPayload.methodName="storage.objects.delete" AND
protoPayload.resourceName:(
  "projects/_/buckets/gcf-sources-"
  AND "/objects/"
  AND "/version-"
  AND "function-source.zip"
) AND
NOT protoPayload.authenticationInfo.principalEmail =~ "service-.*@gcf-admin-robot.iam.gserviceaccount.com"
```

```
-- Show malicious "function-source.zip" object upload inside "gcf-sources-<PROJECT-NUMBER>-<REGION>" bucket.

protoPayload.methodName="storage.objects.create" AND
protoPayload.resourceName:(
  "projects/_/buckets/gcf-sources-"
  AND "/objects/"
  AND "/version-"
  AND "function-source.zip"
) AND
NOT protoPayload.authenticationInfo.principalEmail =~ "service-.*@gcf-admin-robot.iam.gserviceaccount.com"
```

Gen-1: Indicator of Attack (IOA)

```
-- Show malicious "function-source.zip" object deletion inside "gcf-sources-<PROJECT-NUMBER>-<REGION>" bucket
for a specific Function.

protoPayload.methodName="storage.objects.delete" AND
protoPayload.resourceName:(
  "projects/_/buckets/gcf-sources-"
  AND "/objects/<FUNCTION_FOLDER_NAME>"
  AND "/version-"
  AND "function-source.zip"
) AND
NOT protoPayload.authenticationInfo.principalEmail =~ "service-.*@gcf-admin-robot.iam.gserviceaccount.com"
```

```
-- Show malicious "function-source.zip" object upload inside "gcf-sources-<PROJECT-NUMBER>-<REGION>" bucket
for a specific Function.

protoPayload.methodName="storage.objects.create" AND
protoPayload.resourceName:(
  "projects/_/buckets/gcf-sources-"
  AND "/objects/<FUNCTION_FOLDER_NAME>/"
  AND "/version-"
  AND "function-source.zip"
) AND
NOT protoPayload.authenticationInfo.principalEmail =~ "service-.*@gcf-admin-robot.iam.gserviceaccount.com"
```

Gen-2: Indicator of Attack (IOA)

Deleting “**function-source.zip**” Object Version

- Both the Buckets and all of it’s Folder, File objects & File Object Versions

  “gcf-v2-sources-<*PROJECT-NUMBER*>-<*REGION*>/<*FUNCTION-NAME*>/**function-source.zip**”

  “gcf-v2-uploads-<PROJECT-NUMBER>-<REGION>/***<UUID>*.zip**”

  are created, uploaded and deleted by Service Agent:

  service-*<PROJECT-NUMBER>*@gcf-admin-robot.iam.gserviceaccount.com

  Service Agents **cannot** be **Impersonated** in Google Cloud Platform. (As of August 2024)

- Therefore, if any other principal (user or service account) deletes these folder and file objects it could be an indication of potentially malicious activity.

Gen-2: Indicator of Attack (IOA)

```
-- Show malicious "function-source.zip" object version deletion inside "gcf-v2-sources-<PROJECT-NUMBER>-<REGION>"
bucket.

protoPayload.methodName="storage.objects.delete" AND
protoPayload.resourceName:(
  "projects/_/buckets/gcf-v2-sources-"
  AND "/objects/"
  AND "function-source.zip"
) AND
NOT protoPayload.authenticationInfo.principalEmail =~ "service-.*@gcf-admin-robot.iam.gserviceaccount.com"
```

```
-- Show malicious "function-source.zip" object version deletion inside "gcf-v2-sources-<PROJECT-NUMBER>-<REGION>"
bucket for a specific Function.

protoPayload.methodName="storage.objects.delete" AND
protoPayload.resourceName:(
  "projects/_/buckets/gcf-v2-sources-"
  AND "/objects/<FUNCTION_NAME>/"
  AND "function-source.zip"
) AND
NOT protoPayload.authenticationInfo.principalEmail =~ "service-.*@gcf-admin-robot.iam.gserviceaccount.com"
```

Gen-2: Indicator of Attack (IOA)

```
-- Show malicious "<UUID>.zip" object version deletion inside "gcf-v2-uploads-<PROJECT-NUMBER>-<REGION>" bucket.

protoPayload.methodName="storage.objects.delete" AND
protoPayload.resourceName:(
  "projects/_/buckets/gcf-v2-uploads-"
  AND ".zip"
) AND
NOT protoPayload.authenticationInfo.principalEmail =~ "service-.*@gcf-admin-robot.iam.gserviceaccount.com"
```

Artifact Retention

Gen-1		
Policy	**State**	**Default**
Soft Delete	Enabled	7 Days
Object Versioning	Disabled	
Bucket Retention	Disabled	
Object Retention	Disabled	
Event-based hold	Disabled	

Gen-2		
Policy	**State**	**Default**
Soft Delete	Enabled	7 Days
Object Versioning	Enabled	
Bucket Retention	Disabled	
Object Retention	Disabled	
Event-based hold	Disabled	

Artifact Retention

Gen-1				
Policy	**State**	**Recommended State**	**Default**	**Recommended**
Soft Delete	Enabled	Enable	7 Days	90 Days
Object Versioning	Disabled	Disable	-	-
Bucket Retention	Disabled	Enable	-	90 Days
Object Retention	Disabled	-	-	-
Event-based hold	Disabled	-	-	-

Artifact Retention

Gen-2				
Policy	**State**	**Recommended State**	**Default**	**Recommended**
Soft Delete	Enabled	Enable	7 Days	90 Days
Object Versioning	Enabled	Enable	-	-
Bucket Retention	Disabled	Enable		90 Days
Object Retention	Disabled	-	-	-
Event-based hold	Disabled	-	-	-

Thanks, let’s take some Questions!