Google Secret Manager

说明：秘密资料加密存储管理云端工具

Google Cloud Platform

Google Cloud Platform lets you build, deploy, and scale applications, websites, and services on the same infrastructure as Google.

https://console.cloud.google.com/security/secret-manager

项目创建页面

Google Cloud Platform

Google Cloud Platform lets you build, deploy, and scale applications, websites, and services on the same infrastructure as Google.

https://console.cloud.google.com/projectselector2/home/dashboard

> 所有 cloud服务使用前提：创建并使用“服务账号密钥“

Authenticating as a service account | Authentication | Google Cloud

"type": "thumb-down", "id": "hardToUnderstand", "label":"Hard to understand" },{ "type": "thumb-down", "id": "incorrectInformationOrSampleCode", "label":"Incorrect information or sample code" },{ "type": "thumb-down", "id": "missingTheInformationSamplesINeed", "label":"Missing the

https://cloud.google.com/docs/authentication/production

- 进入以上页面，往下翻到“创建服务帐号”位置，创建或选择已有的服务账号密钥，下载密钥json文件
- 然后选择使用“通过环境变量传递凭据“来使用这个密钥，比如mac上使用方法跟文档说明略微不同，仅仅export命令不行，有效的方法为:

```
nano ~/.bash_profile
export GOOGLE_APPLICATION_CREDENTIALS="[上面的JSON_PATH]" //添加此行并保存退出
source ~/.bash_profile
```

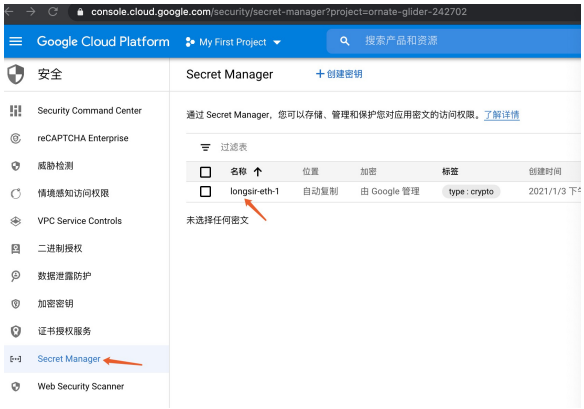
- 点击以下链接，使用secret manager的控制台创建需要保护的密钥文本, 取名→输入文本/上传文件→默认选项→创建

Google Cloud Platform

Google Cloud Platform lets you build, deploy, and scale applications, websites, and services on the same infrastructure as Google.

https://console.cloud.google.com/security/secret-manager

- 然后在密钥列表中



- 点击刚创建的“名称”，进入密钥详情页面

- 点击某个版本最右边的操作按钮（3个点），复制资源id给程序使用

- 点击以下链接，然后选择node.js代码示例，name更换为上一步的资源id，运行脚本即可获得受保护的密钥

Access secret version | Secret Manager Documentation | Google Cloud

Whether your business is early in its journey or well on its way to digital transformation, Google Cloud's solutions and technologies help chart a path to success.

https://cloud.google.com/secret-manager/docs/samples/secretmanager-access-secret-version#secretmanager_access_secret_version-nodejs

示例代码

```
/**
 * 官方样码：https://cloud.google.com/secret-manager/docs/samples/secretmanager-access-secret-version#secretmanager_access_secret_versio
 */

const {SecretManagerServiceClient} = require('@google-cloud/secret-manager');

const client = new SecretManagerServiceClient();

const mySecretProject = 'projects/661232270843'

//full secret path: projects/661232270843/secrets/longsir-eth-1/versions/4
const mySecretProjectPath = mySecretProject + '/secrets'

/**
 * 创建一个名为secretId的密钥，设置其值需要使用addSecretVersion
 * @param {string} secretId https://console.cloud.google.com/security/secret-manager 页面的名称栏所表示的id
 */
async function createSecret(secretId) {
  const [secret] = await client.createSecret({
    parent: mySecretProject,
    secretId: secretId,
    secret: {
      replication: {
        automatic: {},
      },
    },
  });

  console.log(`Created secret ${secret.name}`);
}

/**
 * 添加一个secretStr到secretId所表示密钥中，版本自动递增1
 * @param {string} secretId https://console.cloud.google.com/security/secret-manager 页面的名称栏所表示的id
 * @param {string} secretStr 设定值
 */
async function addSecretVersion(secretId, secretStr) {
  const payload = Buffer.from(secretStr, 'utf8');
  const [version] = await client.addSecretVersion({
    parent: mySecretProjectPath + '/' + secretId,
    payload: {
      data: payload,
    },
  });
  console.log(`Added secret version ${version.name}`);
}

/**
 * 获取名为secretId的密钥的versionId对应的版本值
 * @param {string} secretId https://console.cloud.google.com/security/secret-manager 页面的名称栏所表示的id
 * @param {string} versionId 第几个版本, "latest" 表示最新版本，版本号最大的那个
 */
async function accessSecretVersion(secretId, versionId = 1) {
  const [version] = await client.accessSecretVersion({
    name: mySecretProjectPath + '/' + secretId + '/versions/' + versionId
  });

  let v = version.payload.data.toString();
  console.info(`The secret version is: ${v}`);
  return v;
}

/**
 * 删除名为secretId的密钥, 如： projects/my-project/secrets/my-secret 中的 my-secret
 * @param {string} secretId https://console.cloud.google.com/security/secret-manager 页面的名称栏所表示的id
 */
async function deleteSecret(secretId) {
  let name = mySecretProjectPath + '/' + secretId
  await client.deleteSecret({
    name: name,
  });

  console.log(`Deleted secret ${name}`);
}

/**
 * 查询是否有特定的secretId, 比如：projects/661232270843/secrets/longsir-eth-1
 * @param {string} secretId
 */
async function getSecret(secretId) {
  try{
      const [secret] = await client.getSecret({
          name: mySecretProjectPath + '/' + secretId,
```

```
        });
      return secret && secret.name
  } catch(e) {
      return false;
  }
}

/**
 * 列出所有的secretId
 */
async function listSecrets() {
  const [secrets] = await client.listSecrets({
    parent: mySecretProject,
  });
  let arr = []
  secrets.forEach(secret => {
    arr.push(secret.name);
  });

  return arr;
}

// createSecret('test1')
// addSecretVersion('test1', 'my super secret data');
// accessSecretVersion('test1', 1);
// deleteSecret('test1');
// getSecret('longsir-eth-1');
// listSecrets();

module.exports = {
  createSecret,
  addSecretVersion,
  accessSecretVersion,
  deleteSecret,
  getSecret,
  listSecrets
}
```