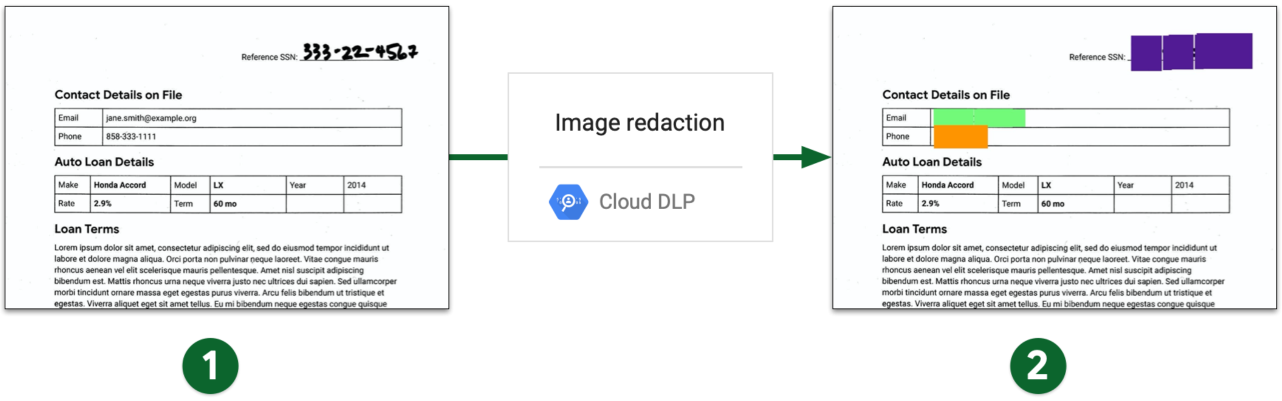
Redacting sensitive data from images

Cloud Data Loss Prevention (DLP) can redact sensitive text from an image. Using infoType detectors (/dlp/docs/infotypes-reference) and Cloud Vision (/vision), Cloud DLP inspects a base64-encoded image for text, detects sensitive data within the text, and then returns a base64-encoded image with any matching sensitive data obscured by an opaque rectangle.

Cloud DLP can redact sensitive data from many image types, including JPEG, BMP, and PNG. For more inform
pported file types (/dlp/docs/supported-file-types). Note that content redaction is currently not supported for
r DOCX files.

For example, consider the following "before" and "after" images. The original image is an example of a typical image file generated from a scan of a paper document. In this example, Cloud DLP has been configured to redact US Social Security numbers, email addresses, and telephone numbers using rectangles of different colors, depending on the content.



(/dlp/docs/images/redacting-sensitive-data-images-beforeafter.png)

1. Scanned image before image redaction
2. Scanned image after image redaction

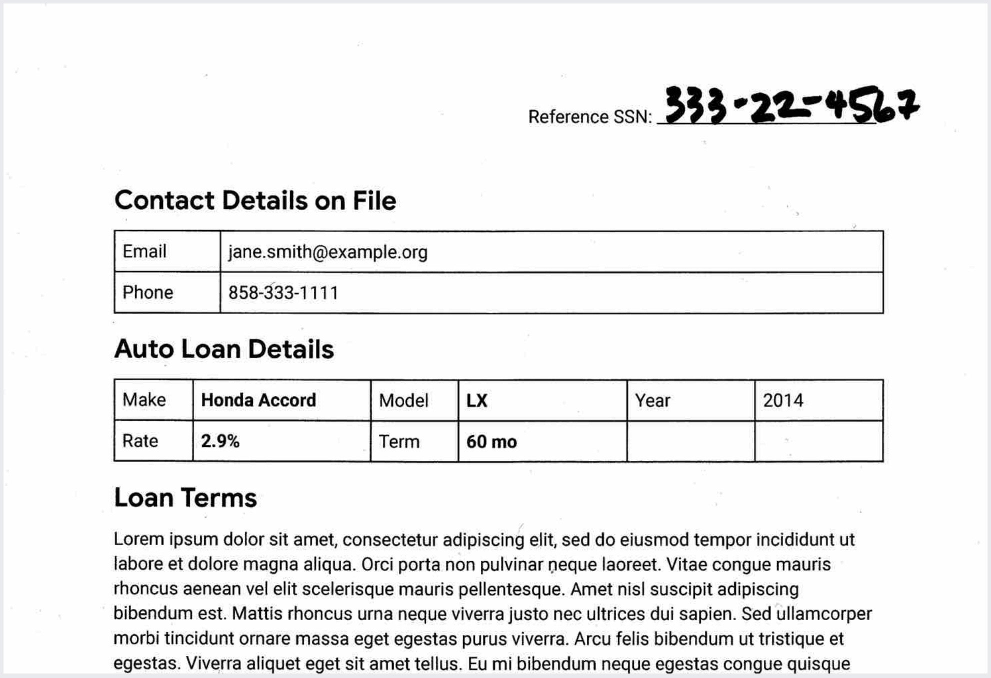
Redacting all default infoTypes from an image

To redact sensitive data from an image, you submit a base64-encoded image to the DLP API's `image.redact` (/dlp/docs/reference/rest/v2/projects.image/redact) method. Unless you specify specific information types (infoTypes (/dlp/docs/infotypes-reference)) to search for, Cloud DLP searches for the most common infoTypes.

To redact default infoTypes from an image:

1. Encode the image as base64.
2. Submit a request to the DLP API's `image.redact` method. The request need only contain the base64-encoded image if you want to redact default infoTypes.

For example, consider the following image. This image is an example of a typical image file generated from a scan of a paper document.



Reference SSN: 333-22-4567

Contact Details on File

Email	jane.smith@example.org
Phone	858-333-1111

Auto Loan Details

Make	**Honda Accord**	Model	**LX**	Year	2014
Rate	**2.9%**	Term	**60 mo**		

Loan Terms

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Orci porta non pulvinar neque laoreet. Vitae congue mauris rhoncus aenean vel elit scelerisque mauris pellentesque. Amet nisl suscipit adipiscing bibendum est. Mattis rhoncus urna neque viverra justo nec ultrices dui sapien. Sed ullamcorper morbi tincidunt ornare massa eget egestas purus viverra. Arcu felis bibendum ut tristique et egestas. Viverra aliquet eget sit amet tellus. Eu mi bibendum neque egestas congue quisque

(/dlp/docs/images/redacting-sensitive-data-images-dlpbefore.jpg)

To redact the default infoTypes from this image, send the following request to the DLP API's `image.redact` (/dlp/docs/reference/rest/v2/projects.image/redact) method:

tant: The code on this page requires that you first set up a Cloud DLP client. For more information about instal eating a Cloud DLP client, see Cloud DLP client libraries (/dlp/docs/reference/libraries). (Sending JSON to Clou EST endpoints does not require a client library.)

ProtocolJava (#java)

Important: Before sending this request, use a base64 utility to encode the image into ASCII text data, and then replace the **`[BASE64-ENCODED-IMAGE]`** placeholder with the base64-encoded text. Most development environments contain a native **`base64`** utility. To encode an image file:

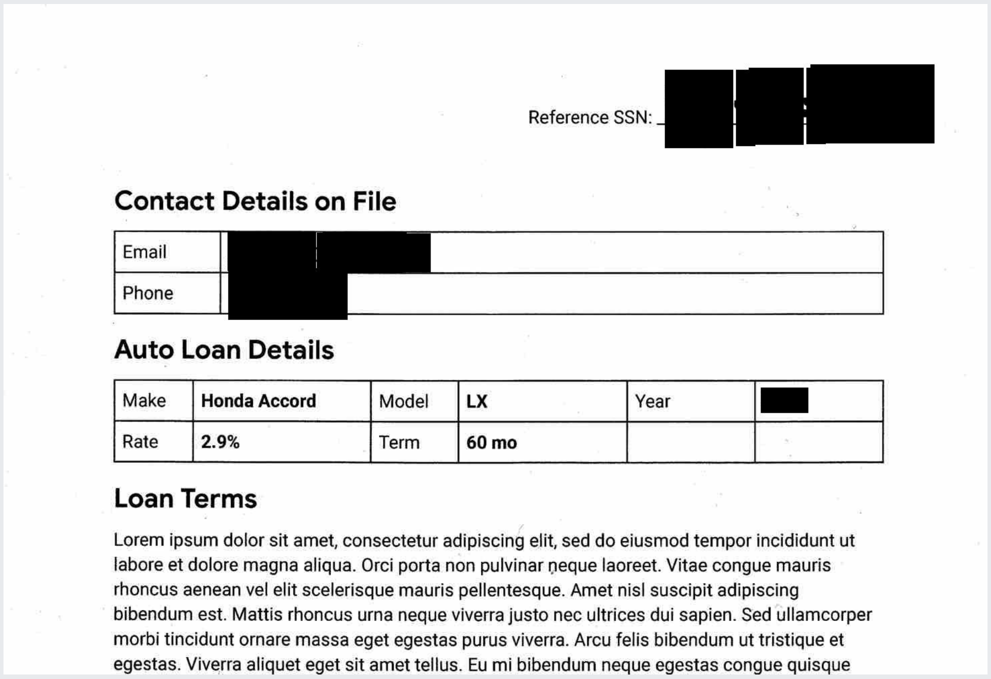
- Linux: **`base64 input.jpg > output.txt`**
- macOS: **`base64 -i input.jpg -o output.txt`**
- Windows: **`C:> Base64.exe -e input.jpg > output.txt`**
- PowerShell: **`[Convert]::ToBase64String([IO.File]::ReadAllBytes("./input.jpg")) > output.txt`**

```
{
  "byteItem": {
    "data": "[BASE64-ENCODED-IMAGE]",
    "type": "IMAGE_JPEG"
  }
}
```

Cloud DLP returns the following:

```
{
  "redactedImage": "[BASE64-ENCODED-IMAGE]"
}
```

After decoding the base64-encoded image, the image appears as follows:



Reference SSN: [redacted]

Contact Details on File

Email	[redacted]
Phone	[redacted]

Auto Loan Details

Make	**Honda Accord**	Model	**LX**	Year	[redacted]
Rate	**2.9%**	Term	**60 mo**		

Loan Terms

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Orci porta non pulvinar neque laoreet. Vitae congue mauris rhoncus aenean vel elit scelerisque mauris pellentesque. Amet nisl suscipit adipiscing bibendum est. Mattis rhoncus urna neque viverra justo nec ultrices dui sapien. Sed ullamcorper morbi tincidunt ornare massa eget egestas purus viverra. Arcu felis bibendum ut tristique et egestas. Viverra aliquet eget sit amet tellus. Eu mi bibendum neque egestas congue quisque

(/dlp/docs/images/redacting-sensitive-data-images-dlpafter1.jpg)

Note that, in addition to masking the handwritten Social Security number, the email address, and the phone number, Cloud DLP has also redacted the year. Assuming that's not optimal behavior, the next example demonstrates how to redact only certain infoTypes.

ong with the redacted image, you can get a report of pixel coordinate and length values that indicate the areas
ion boxes were drawn. This is the same information that image inspection (/dlp/docs/inspecting-images) retu
ble this functionality, set **`includeFindings`** (/dlp/docs/reference/rest/v2/projects.image/redact) to **`true`** in
st to **`image.redact`**.

Redacting specific infoTypes from an image

If you want to redact only certain sensitive data from an image, specify their corresponding built-in infoTypes.

You can also redact data matching your custom infoType detectors (/dlp/docs/creating-custom-infotypes).

To redact specific infoTypes from an image:

1. Encode the image as base64.

2. Submit a request to the DLP API's `image.redact` method. The request must include:
    - The base64-encoded image.
    - One or more infoType detectors (/dlp/docs/infotypes-reference).

Consider the original image from the previous section. To redact only US Social Security numbers, email addresses, and telephone numbers, send the following JSON to the DLP API's `image.redact` (/dlp/docs/reference/rest/v2/projects.image/redact) method:

tant: The code on this page requires that you first set up a Cloud DLP client. For more information about instal
eating a Cloud DLP client, see Cloud DLP client libraries (/dlp/docs/reference/libraries). (Sending JSON to Clou
EST endpoints does not require a client library.)

ProtocolJava (#java)C# (#c)

Important: Before sending this request, use a base64 utility to encode the image into ASCII text data, and then replace the **`[BASE64-ENCODED-IMAGE]`** placeholder with the base64-encoded text. Most development environments contain a native **`base64`** utility. To encode an image file:

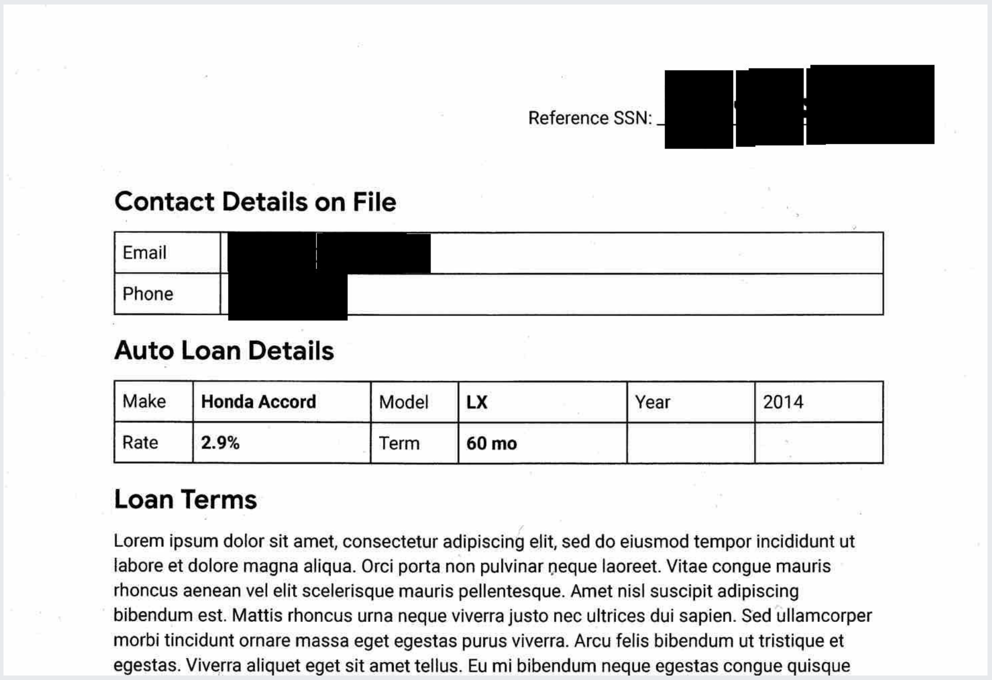
- Linux: **`base64 input.jpg > output.txt`**
- macOS: **`base64 -i input.jpg -o output.txt`**
- Windows: **`C:> Base64.exe -e input.jpg > output.txt`**
- PowerShell:
  `[Convert]::ToBase64String([IO.File]::ReadAllBytes("./input.jpg")) > output.txt`

```
{
  "byteItem": {
    "data": "[BASE64-ENCODED-IMAGE]",
    "type": "IMAGE_JPEG"
  },
  "imageRedactionConfigs": [
    {
      "infoType": {
        "name": "US_SOCIAL_SECURITY_NUMBER"
      }
    },
    {
      "infoType": {
        "name": "EMAIL_ADDRESS"
      }
    },
    {
      "infoType": {
        "name": "PHONE_NUMBER"
      }
    }
  ]
}
```

Cloud DLP returns the following:

```
  {
    "redactedImage": "[BASE64-ENCODED-IMAGE]"
  }
```

After decoding the base64-encoded image, the image appears as follows:



Reference SSN: [redacted]

Contact Details on File

Email	[redacted]
Phone	[redacted]

Auto Loan Details

Make	**Honda Accord**	Model	**LX**	Year	2014
Rate	**2.9%**	Term	**60 mo**		

Loan Terms

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Orci porta non pulvinar neque laoreet. Vitae congue mauris rhoncus aenean vel elit scelerisque mauris pellentesque. Amet nisl suscipit adipiscing bibendum est. Mattis rhoncus urna neque viverra justo nec ultrices dui sapien. Sed ullamcorper morbi tincidunt ornare massa eget egestas purus viverra. Arcu felis bibendum ut tristique et egestas. Viverra aliquet eget sit amet tellus. Eu mi bibendum neque egestas congue quisque

(/dlp/docs/images/redacting-sensitive-data-images-dlpafter2.jpg)

You can color code redacted information by infoType when you want to tell at a glance what's been redacted. See the following section for more information.

Redacting infoTypes from an image with color coding

To color code redacted information by infoType, you pair infoType detectors with RGB color space values.

To color code infoTypes redacted from an image:

1. Encode the image as base64.
2. Submit a request to the DLP API's `image.redact` method. The request must include:
    - The base64-encoded image.
    - One or more infoType detectors (/dlp/docs/infotypes-reference), each of which is assigned a color using RGB color space values (/dlp/docs/reference/rest/v2/projects.image/redact#Color).

Color space values are expressed in the three chromaticities of red, green, and blue. Each value is between 0 ar
ve. If you're used to expressing RGB chromaticity values between 0 and 255, inclusive, you'll need to divide eac
 before including it in the call to `image.redact`. For example, a purple RGB value of (77,26,153) would be
kimately (0.3,0.1,0.6).

Consider the original image from the first section. To redact US Social Security numbers with a purple box, email addresses with a green box, and telephone numbers with an orange box, send the following JSON to the DLP API's `image.redact` (/dlp/docs/reference/rest/v2/projects.image/redact) method:

tant: The code on this page requires that you first set up a Cloud DLP client. For more information about instal
eating a Cloud DLP client, see Cloud DLP client libraries (/dlp/docs/reference/libraries). (Sending JSON to Clou
EST endpoints does not require a client library.)

ProtocolJava (#java)

Important: Before sending this request, use a base64 utility to encode the image into ASCII text data, and then replace the `[BASE64-ENCODED-IMAGE]` placeholder with the base64-encoded text. Most development environments contain a native `base64` utility. To encode an image file:

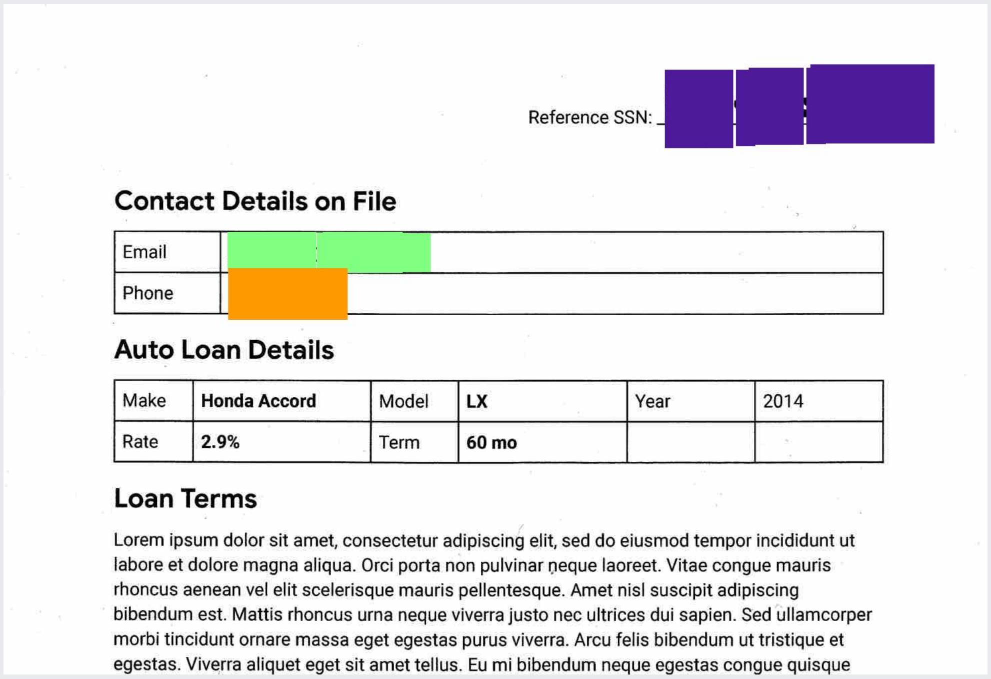
- Linux: `base64 input.jpg > output.txt`
- macOS: `base64 -i input.jpg -o output.txt`
- Windows: `C:> Base64.exe -e input.jpg > output.txt`
- PowerShell: `[Convert]::ToBase64String([IO.File]::ReadAllBytes("./input.jpg")) > output.txt`

```
{
  "byteItem": {
    "data": "[BASE64-ENCODED-IMAGE]",
    "type": "IMAGE_JPEG"
  },
  "imageRedactionConfigs": [
    {
      "infoType": {
        "name": "US_SOCIAL_SECURITY_NUMBER"
      },
      "redactionColor": {
        "red": 0.3,
        "green": 0.1,
        "blue": 0.6
      }
    },
    {
      "infoType": {
        "name": "EMAIL_ADDRESS"
      },
      "redactionColor": {
        "red": 0.5,
        "blue": 0.5,
        "green": 1
      }
    },
    {
      "infoType": {
        "name": "PHONE_NUMBER"
      },
      "redactionColor": {
        "red": 1,
        "blue": 0,
        "green": 0.6
      }
    }
  ]
}
```

Cloud DLP returns the following:

```
  {
    "redactedImage": "[BASE64-ENCODED-IMAGE]"
  }
```

After decoding the base64-encoded image, the image appears as follows:



Reference SSN: [redacted]

Contact Details on File

Email	[redacted]
Phone	[redacted]

Auto Loan Details

Make	**Honda Accord**	Model	**LX**	Year	2014
Rate	**2.9%**	Term	**60 mo**		

Loan Terms

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Orci porta non pulvinar neque laoreet. Vitae congue mauris rhoncus aenean vel elit scelerisque mauris pellentesque. Amet nisl suscipit adipiscing bibendum est. Mattis rhoncus urna neque viverra justo nec ultrices dui sapien. Sed ullamcorper morbi tincidunt ornare massa eget egestas purus viverra. Arcu felis bibendum ut tristique et egestas. Viverra aliquet eget sit amet tellus. Eu mi bibendum neque egestas congue quisque

(/dlp/docs/images/redacting-sensitive-data-images-dlpafter3.jpg)

Redacting all text from an image

Cloud DLP also contains an option to redact all detected text in an image.

To redact all text in an image:

1. Encode the image as base64.
2. Submit a request to the DLP API's `image.redact` method. The request must contain:
   - The base64-encoded image.
   - The `redactAllText` option set to `true`.

Consider the original image from the first section. To redact all text, send the following JSON to the DLP API's `image.redact` (/dlp/docs/reference/rest/v2/projects.image/redact) method:

ant: The code on this page requires that you first set up a Cloud DLP client. For more information about instal
eating a Cloud DLP client, see Cloud DLP client libraries (/dlp/docs/reference/libraries). (Sending JSON to Clou
EST endpoints does not require a client library.)

ProtocolPython (#python)Java (#java)

Important: Before sending this request, use a base64 utility to encode the image into ASCII text data, and then replace the **`[BASE64-ENCODED-IMAGE]`** placeholder with the base64-encoded text. Most development environments contain a native **`base64`** utility. To encode an image file:

- Linux: **`base64 input.jpg > output.txt`**
- macOS: **`base64 -i input.jpg -o output.txt`**
- Windows: **`C:> Base64.exe -e input.jpg > output.txt`**
- PowerShell: **`[Convert]::ToBase64String([IO.File]::ReadAllBytes("./input.jpg")) > output.txt`**

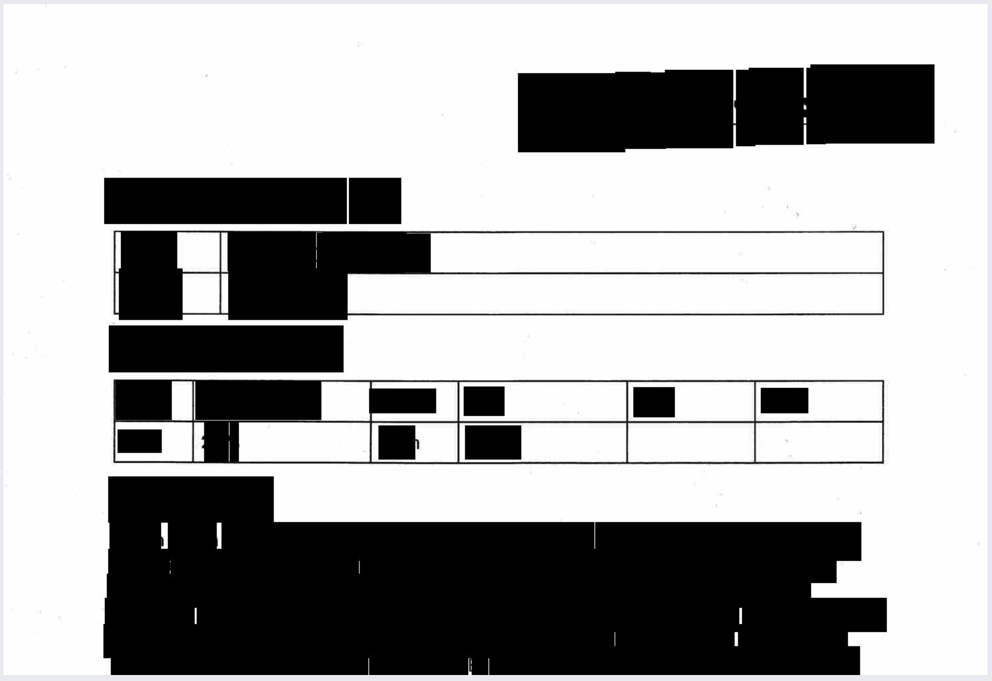
```
{
  "byteItem": {
    "data": "[BASE64-ENCODED-IMAGE]",
    "type": "IMAGE_JPEG"
  },
  "imageRedactionConfigs": [
    {
      "redactAllText": true
    }
  ]
}
```

Cloud DLP returns the following:

```
  {
    "redactedImage": "[BASE64-ENCODED-IMAGE]"
  }
```

The API returns the same image(s) you gave it, but any text identified as containing sensitive information according to your criteria has been redacted.

After decoding the base64-encoded image, the image appears as follows:



(/dlp/docs/images/redacting-sensitive-data-images-dlpafter4.jpg)

Code examples

Following is sample code in several languages that demonstrates how to use Cloud DLP to redact sensitive text from an image.

:ant: The code on this page requires that you first set up a Cloud DLP client. For more information about instal
eating a Cloud DLP client, see Cloud DLP client libraries (/dlp/docs/reference/libraries). (Sending JSON to Clou
EST endpoints does not require a client library.)

Node.jsPython (#python)Go (#go)PHP (#php)C# (#c)

View on GitHub (https://github.com/googleapis/nodejs-dlp/blob/master/samples/redactImage.js)

```
// Imports the Google Cloud Data Loss Prevention library
const DLP = require('@google-cloud/dlp');

// Imports required Node.js libraries
const mime = require('mime');
const fs = require('fs');

// Instantiates a client
const dlp = new DLP.DlpServiceClient();

// The project ID to run the API call under
// const projectId = 'my-project';

// The path to a local file to inspect. Can be a JPG or PNG image file.
// const filepath = 'path/to/image.png';

// The minimum likelihood required before redacting a match
// const minLikelihood = 'LIKELIHOOD_UNSPECIFIED';

// The infoTypes of information to redact
// const infoTypes = [{ name: 'EMAIL_ADDRESS' }, { name: 'PHONE_NUMBER' }];

// The local path to save the resulting image to.
// const outputPath = 'result.png';
async function redactImage() {
  const imageRedactionConfigs = infoTypes.map(infoType => {
    return {infoType: infoType};
  });
  // Load image
  const fileTypeConstant =
    ['image/jpeg', 'image/bmp', 'image/png', 'image/svg'].indexOf(
      mime.getType(filepath)
    ) + 1;
  const fileBytes = Buffer.from(fs.readFileSync(filepath)).toString('base64');

  // Construct image redaction request
  const request = {
    parent: `projects/${projectId}/locations/global`,
    byteItem: {
      type: fileTypeConstant,
      data: fileBytes,
    },
    inspectConfig: {
      minLikelihood: minLikelihood,
      infoTypes: infoTypes,
    },
    imageRedactionConfigs: imageRedactionConfigs,
  };

  // Run image redaction request
  const [response] = await dlp.redactImage(request);
  const image = response.redactedImage;
  fs.writeFileSync(outputPath, image);
  console.log(`Saved image redaction results to path: ${outputPath}`);
}
redactImage();
```

Try it out

You can try out each of the examples on this page yourself—or experiment with your own images—in the APIs Explorer on the reference page for `image.redact`:

Go to APIs Explorer (/dlp/docs/reference/rest/v2/projects.image/redact?apix=true)

or experiments with non-sensitive materials, it's sufficient to uncheck **Google OAuth 2.0** and enter `cts/testff` in the **parent** field.

What's next

- Learn more about image inspection and redaction (/dlp/docs/concepts-image-redaction).
- Learn how to inspect images for sensitive data (/dlp/docs/inspecting-images).