Best practices for securely using Google Map API v3

levin.lin@2016.2.19

1. Get API key (browser key)

Reference:

https://developers.google.com/maps/documentation/javascript/get-api-key#key

Alternatively, follow these steps to get an API key:

(1) Go to the Google Developers Console.

(2) Create or select a project.

(3) Click **Continue** to Enable the API.

(4) On the **Credentials** page, get a **Browser key** (and set the API Credentials).
Note: If you have an existing **Browser key**, you may use that key.

(5) To prevent quota theft, secure your API key following these best practices.

(6) (Optional) Enable billing. See Usage Limits for more information.

2. Best pratices for securely using API keys

Reference:

https://support.google.com/cloud/answer/6310037

When you use API keys in your applications, take care to keep them secure. Publicly exposing your credentials can result in your account being compromised, which could lead to unexpected charges on your account. To keep your API keys secure, follow these **best practices**:

Do not embed API keys directly in code:

API keys that are embedded in code can be accidentally exposed to the public—for example, if you forget to remove the keys from code that you share. Instead of embedding your API keys in your applications, store them in environment variables or in files outside of your application's source tree.

Do not store API keys in files inside your application's source tree:

If you store API keys in files, keep the files outside your application's source tree to help ensure your keys do not end up in your source code control system. This is particularly important if you use a public source code management system such as GitHub.

Restrict your API keys to be used by only the IP addresses, referrer URLs, and mobile apps that need them:

By restricting the IP addresses, referrer URLs, and mobile apps that can use each key, you can reduce the impact of a compromised API key. You can specify the hosts and apps that can use each key from the Google Developers Console by opening the Credentials page and then either creating a new API key with the settings you want, or editing the settings of an API key.

Delete unneeded API keys:

To minimize your exposure to attack, delete any API keys that you no longer need.

Regenerate your API keys periodically:

You can regenerate API keys from the Google Developers Console by opening the Credentials page and clicking **Regenerate key** for each key. Then, update your applications to use the newly-generated keys. Your old keys will continue to work for 24 hours after you generate replacement keys.

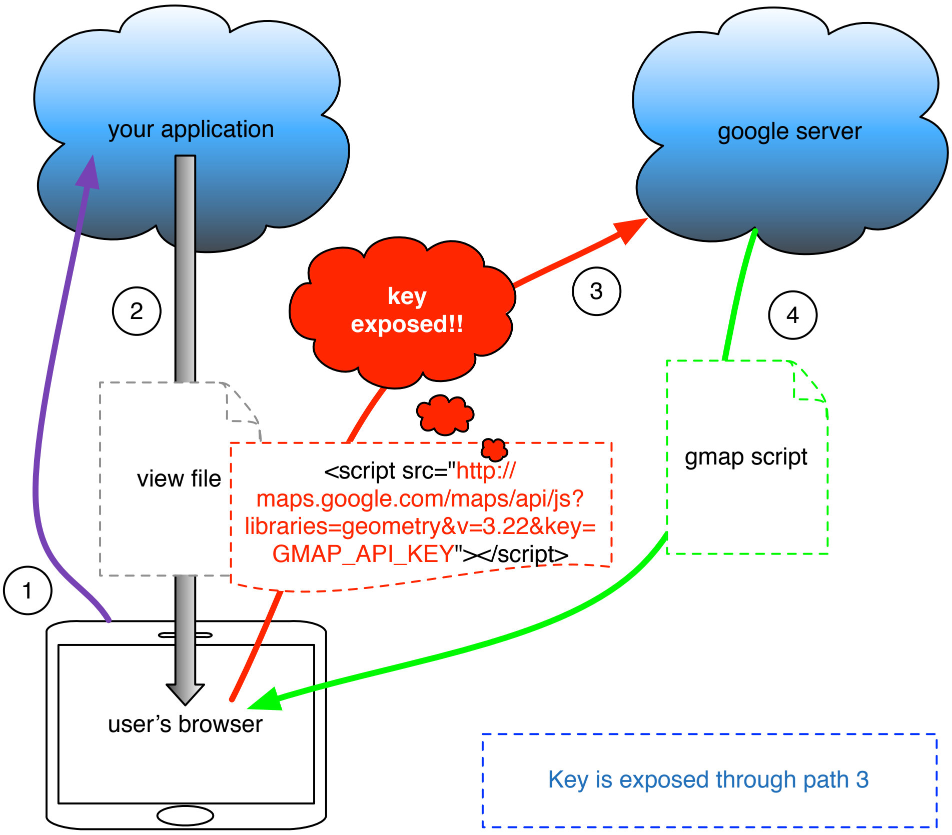
Review your code before publicly releasing it:

Ensure that your code does not contain API keys or any other private information before you make your code publicly available.

3. Beyond the "Best practices"

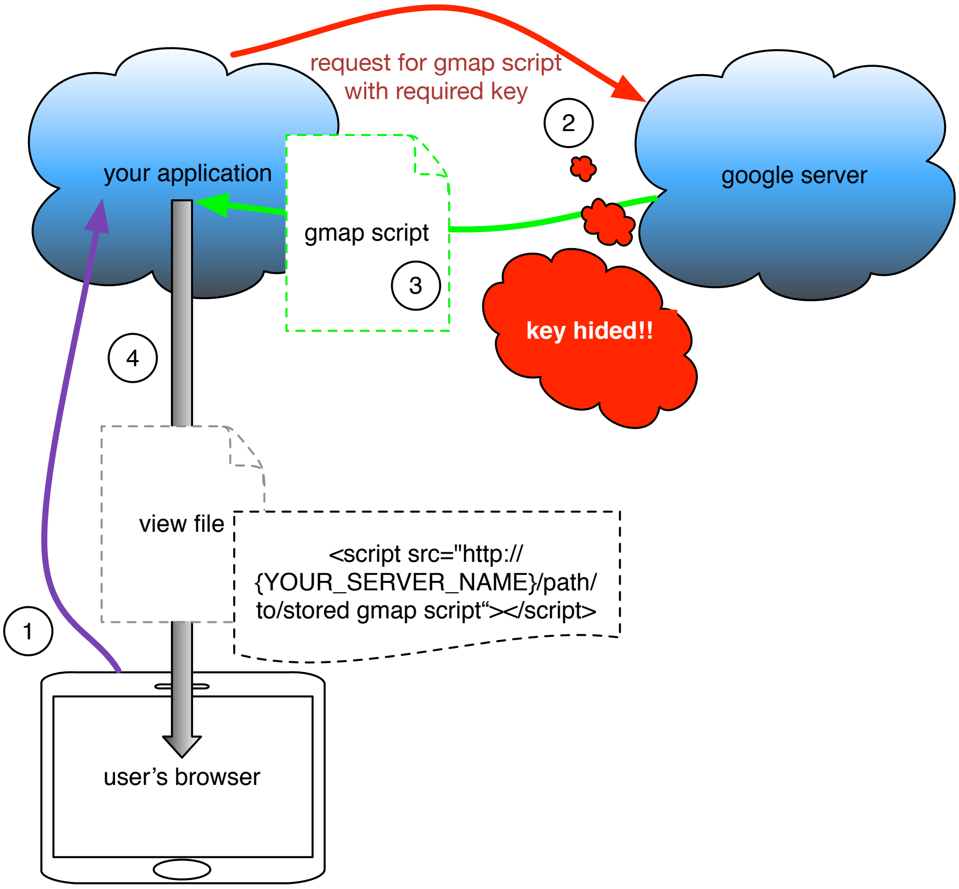
Following step 2 is good practice for our application, but we can still do more beyond that:

(1) It is easy to be noticed that, even when we keep API keys out of application's source tree and restrict the referrer URLs, they still can be exposed from the request sent from browser to google server.


your application
google server
key exposed!!
2
3
4
1
view file
<script src="http://maps.google.com/maps/api/js?libraries=geometry&v=3.22&key=GMAP_API_KEY"></script>
gmap script
user's browser
Key is exposed through path 3

best practices by google's advice

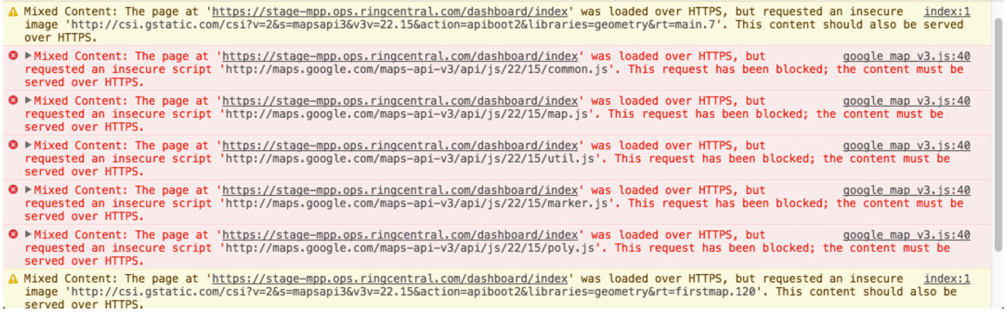
The better practice is hiding requests for Gmap from user's browser to our servers, which means our application do such requests in backend then give the scripts to browser on its needs, so that API key is naturally hided.



beyond best practices by google’s advice

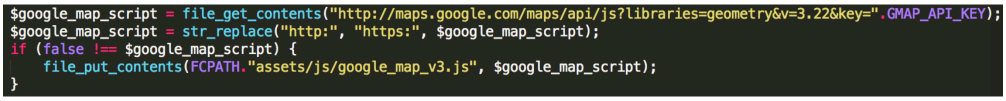
(2) If you choose to call gmap api from server side, you can do more thing while storing the gamp script file.

Here is a example, there are many API URLs in gmap script using HTTP protocol, which doesn’t work where using HTTPS, that result in many errors like:



```
⚠ Mixed Content: The page at 'https://stage-mpp.ops.ringcentral.com/dashboard/index' was loaded over HTTPS, but requested an insecure   index:1
  image 'http://csi.gstatic.com/csi?v=2&s=mapsapi3&v3v=22.15&action=apiboot2&libraries=geometry&rt=main.7'. This content should also be served
  over HTTPS.
⊗ ▶Mixed Content: The page at 'https://stage-mpp.ops.ringcentral.com/dashboard/index' was loaded over HTTPS, but        google_map_v3.js:40
  requested an insecure script 'http://maps.google.com/maps-api-v3/api/js/22/15/common.js'. This request has been blocked; the content must be
  served over HTTPS.
⊗ ▶Mixed Content: The page at 'https://stage-mpp.ops.ringcentral.com/dashboard/index' was loaded over HTTPS, but        google_map_v3.js:40
  requested an insecure script 'http://maps.google.com/maps-api-v3/api/js/22/15/map.js'. This request has been blocked; the content must be
  served over HTTPS.
⊗ ▶Mixed Content: The page at 'https://stage-mpp.ops.ringcentral.com/dashboard/index' was loaded over HTTPS, but        google_map_v3.js:40
  requested an insecure script 'http://maps.google.com/maps-api-v3/api/js/22/15/util.js'. This request has been blocked; the content must be
  served over HTTPS.
⊗ ▶Mixed Content: The page at 'https://stage-mpp.ops.ringcentral.com/dashboard/index' was loaded over HTTPS, but        google_map_v3.js:40
  requested an insecure script 'http://maps.google.com/maps-api-v3/api/js/22/15/marker.js'. This request has been blocked; the content must be
  served over HTTPS.
⊗ ▶Mixed Content: The page at 'https://stage-mpp.ops.ringcentral.com/dashboard/index' was loaded over HTTPS, but        google_map_v3.js:40
  requested an insecure script 'http://maps.google.com/maps-api-v3/api/js/22/15/poly.js'. This request has been blocked; the content must be
  served over HTTPS.
⚠ Mixed Content: The page at 'https://stage-mpp.ops.ringcentral.com/dashboard/index' was loaded over HTTPS, but requested an insecure   index:1
  image 'http://csi.gstatic.com/csi?v=2&s=mapsapi3&v3v=22.15&action=apiboot2&libraries=geometry&rt=firstmap.120'. This content should also be
  served over HTTPS.
```

It is easy to resolved that you just need to replace all the “http:” in gmap script whith “https:” before store it to file.



```
$google_map_script = file_get_contents("http://maps.google.com/maps/api/js?libraries=geometry&v=3.22&key=".GMAP_API_KEY);
$google_map_script = str_replace("http:", "https:", $google_map_script);
if (false !== $google_map_script) {
    file_put_contents(FCPATH."assets/js/google_map_v3.js", $google_map_script);
}
```

Enjoy it ☺