Advisory

Gmail - Google Docs Cookie Hijacking through PDF Repurposing Attack

Aditya K Sood, (C) SecNiche Security
Email: adi_ks [at] secniche.org

Disclaimer

Responsible Disclosure

_____

May 5 2009 –

Responsible Disclosed to Google Security Team

May 5 2009 –

Google started investigation of the vulnerability.

May 6 2009 –

Proof of Concept was shared with Google Security Team

May 7 2009 –

Google reproduced the issue and started working on it.

May 8 2009 –

A non disclosure notification was sent as per Google requirement

May 9 2009 –

Google deployed the requisite recommendation.

May 11 2009 –

Advisory Released

_____

Points to Consider:

[1] The whole network of Google docs has been changed and there will be no use of Adobe Acrobat Plugin.

[2] The custom designed application should avoid using acrobat plugin in opening PDF documents.

[3] Number of applications are still vulnerable to these type of inherent attacks.

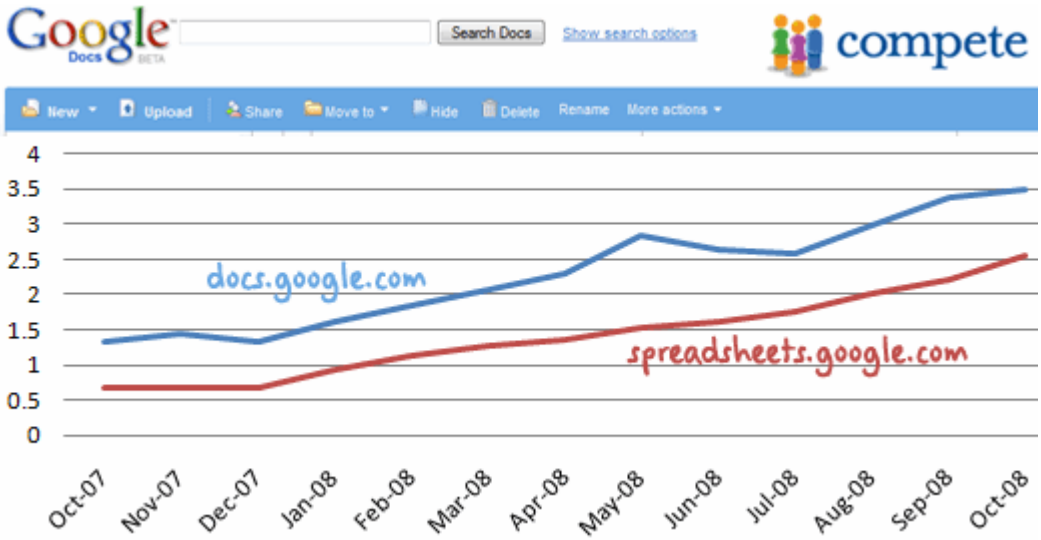
Discussion:

This attack depends on Adobe plugin used in browsers for opening of PDF files. There is another modified approach to trigger web attacks through JavaScript protocol handler in the context of browser when a PDF is opened in it. As we have seen, the kind of security mechanism implemented by Adobe in order to remove the insecurities that originate directly from the standalone PDF document in order to circumvent cross domain access. The attack is targeted on the web applications that allow PDF documents to be uploaded on the web server. Due to ingrained security mechanism in PDF Reader, it is hard to launch certain attacks. But with this technique an attacker can steal generic information from website by executing the code directly in the context of the domain where it is uploaded. The attack surface can be diversified by randomizing the attack vector. On further analysis it has been observed that it is possible to trigger phishing attacks too. Successful attacks have been conducted on number of web applications mainly to extract information based on DOM objects. The paper exposes a differential behavior of Acro JS and Brower JavaScript. More details on this type of attack can be seen here:

Detailed Paper
http://secniche.org/papers/SNS_09_03_PDF_Silent_Form_Re_Purp_Attack.pdf

This attack can be used to hijack Gmail/ Google doc cookies efficiently if certain conditions are met. The Google docs are an integrated service provided by Google for online viewing the document. A user logged in to Gmail will have the same cookie used for if any document. The interdependency can be exploited through this attack vector.

According to Compete, Google Docs had around 4.4 million unique visitors in September with docs.google.com attracting nearly twice as many unique users as spreadsheets.google.com on average



Last year stats about usage of Google Docs

The attack can be structured as:

1. An Attacker sends a well crafted PDF file to victim containing the execution code.

2. Victim opens the file in the default PDF Viewer by Google. There is a proper conversion take place and document is converted into different format so that no intermediate DOM calls can be executed. The Google has done a good work in this by keeping the human interaction to minimum.

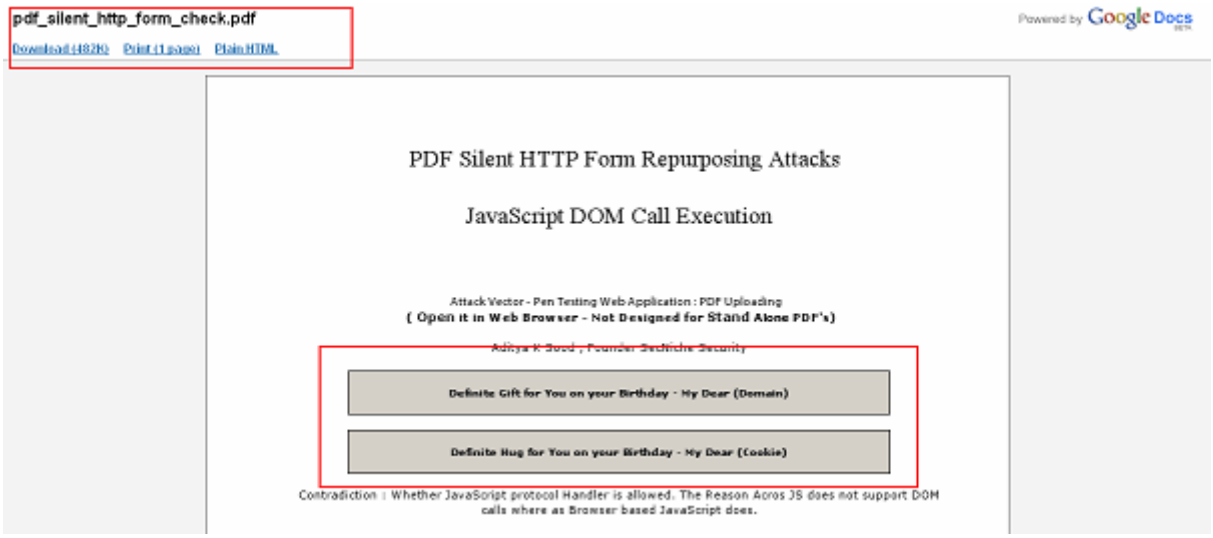
3. If a victim chooses to open that PDF file directly from Google PDF viewer for print, the attack is successful. This is because the PDF during print process is converted back into original format and opened in the browser. So there is no differential check is present and PDF becomes as a active and dynamic content having an appropriate interface with the browser. So now it is possible to extract cookies from it.

The attack is successfully triggered in number of situations and cookie can be extracted easily while user is logged into his/her account. The problem that favors this attack in concern to Gmail is the rendering of PDF directly into browser which should not be allowed.

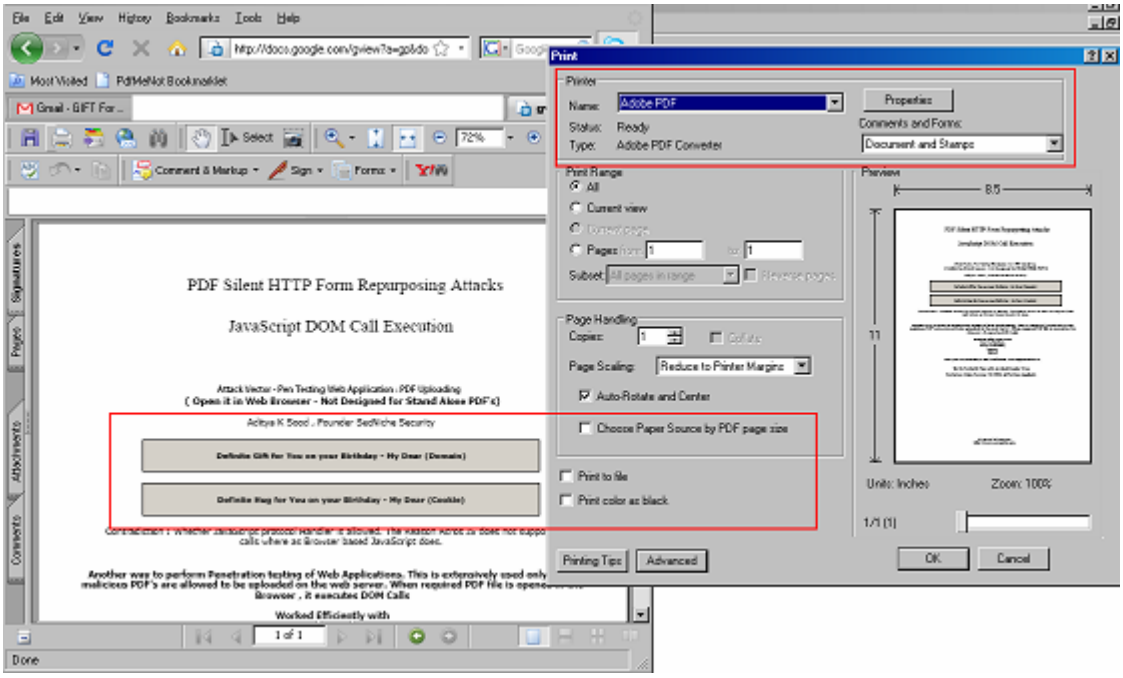
We will demonstrate this attack with appropriate steps as under mentioned:

Step 1: An attacker sends an Email with malicious PDF as an attachment.

Step 2: Victim opens the PDF directly in the Google Doc viewer.

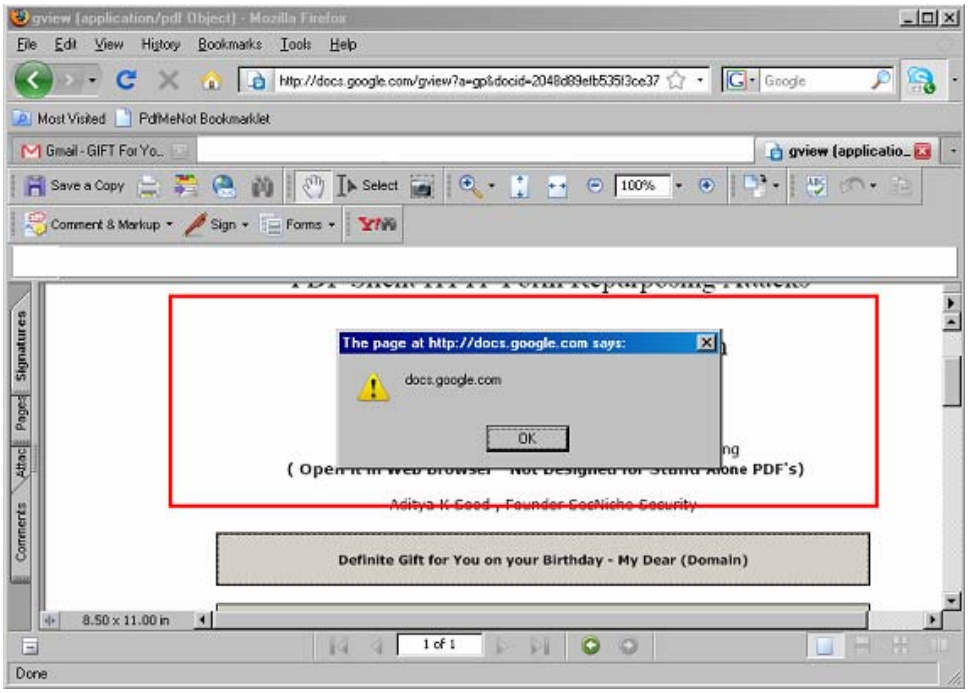


Step 3: Victim tries to print the document directly from viewer.

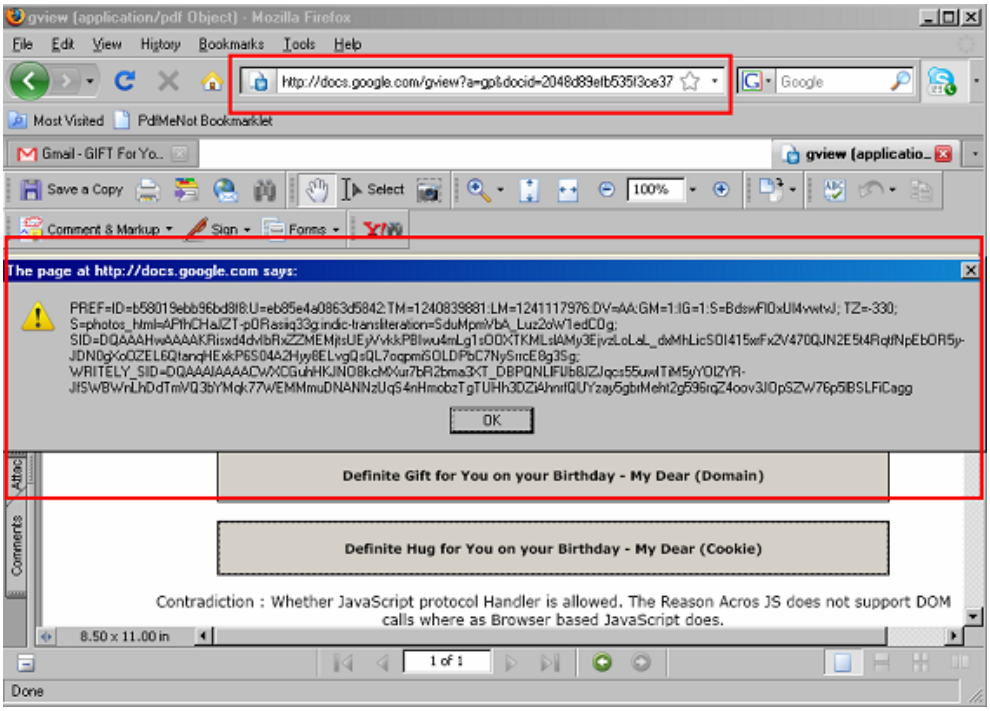


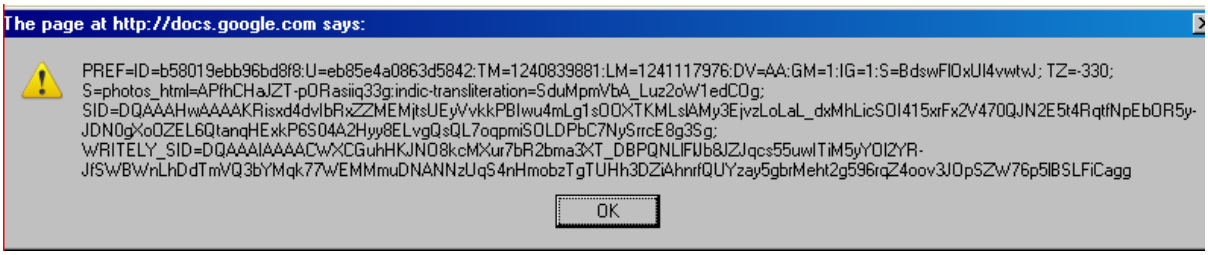
Once the PDF is opened in the browser it depends on the attacker the way he has designed the PDF. In our POC a form has been designed which worked as mentioned below:

[3.1] Gmail/Google Doc - Domain Check



[3.2] Gmail/Google Doc - Cookie Extracted





That's how the PDF Repurposing Attack can be triggered.