Google UI-Redressing Bug That Discloses The User's Email Address

Written by Mazin Ahmed

In this post, I will be talking about an interesting bug that affects Google Blogger. This security bug has been left undiscovered since almost 2007.

The bug allows an attacker to trick the victim into revealing his email address using UI-Redressing techniques.

Background

We always see a header on blogs that is hosted on Google Blogger that looks like the following:

for unauthenticated users:

for authenticated users:

As you can see, in the screenshot of an authenticated user, the user's email address is shown. The challenge now is how can we use this feature on our side; to reveal users's email address.
After testing the issue, it appears that the response of the HTTP request holds the user's email address lacks the usage of X-Frame-Options header. This means that anyone can make an HTTP request within an Iframe, and craft it in a way that can help the attacker in making unintended actions.

A screenshot that shows the lack of X-Frame-Options HTTP response header within the Blogger header's request.

In most cases, we use UI-Redressing when we are allowed to perform “clicks”, and in that form, we would specify it as “Clickjacking”. This case is slightly different type of UI-Redressing from an exploitation point of view.

The idea that came to mind was to create a proof of concept that includes an Iframe that responds with the user's email address, and then force the user into pasting the email within the page, where it gets sent to an external server (eg.. the attacker's sever).

I have put this idea into code that does exactly that, you can watch the following demonstration video to see an example of the exploitation of the issue.

A second idea which apparently failed to be performed due to strong Same-Origin Policy of modern browsers was to have an Iframe that responses with the user's email address, and once the response is received, a Javascript within the same HTML page executes that take a screenshot of the Iframe, then

send it to an external server, where there will be function running that takes the image and transform it into plain text. This issue was not possible due to the Same-Origin Policy, as the API of getscreenshot() is restricted to not be able to read pixels of none-same-origin Iframe.

Vendor Response:

After a discussion with Google Security, Google has decided to not patch the issue.

Final Thoughts:

This was quite an interesting bug, yet, the impact of the issue is not very high. I would like to share it as this issue is almost 10 years old, and no one has discovered, exploited, or discussed it before.

I would like to thank Google Security Team for their prompted and quick response regarding the issue.