



CVE-2021-23827: Sakura Samurai discover cleartext pictures in Keybase Desktop Client; Windows, macOS, Linux

Cleartext Storage in a File or on Disk in Keybase Desktop Clients for Windows, macOS, and Linux allows attacker who can locally read user's files obtain private pictures in the Cache and uploadtemps directories. Keybase Client fails to effectively clear cached pictures, even after deletion via normal methodology within the client, or by utilizing the "Explode message/Explode now" functionality.

Published on Feb 22, 2021

Reading time: 3 minutes.

Credits

John Jackson

Twitter: <https://twitter.com/johnjhacking>

Aubrey Cottle

Twitter: <https://twitter.com/Kirtaner>

Jackson Henry

Twitter: <https://twitter.com/JacksonHHax>

Robert Willis

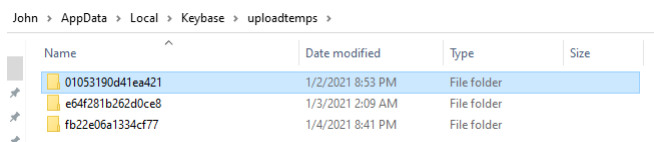
Twitter: https://twitter.com/rej_ex

Identification

During security research, John Jackson stumbled upon the Keybase Client directories and decided to take a look considering Keybase operates a Bug Bounty Program. Within several minutes, John noticed a directory named "uploadtemps"

C:\Users\yourusername\AppData\Local\Keybase\uploadtemps

The directory contained randomized folders:

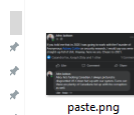


Name	Date modified	Type	Size
01053190d41ea421	1/2/2021 8:53 PM	File folder	
e64f281b262d0ce8	1/3/2021 2:09 AM	File folder	
fb22e06a1334cf77	1/4/2021 8:41 PM	File folder	

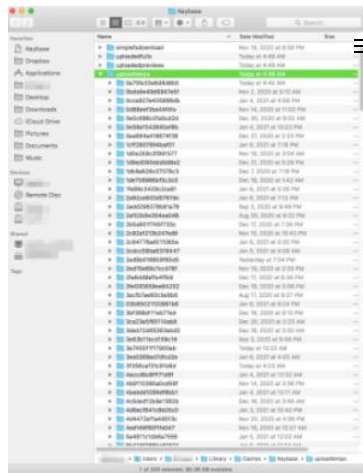
John noticed that inside of these folders, photos that had been previously pasted into conversations [but deleted through either normal means or exploded] remained, unencrypted.

Windows Client

John > AppData > Local > Keybase > uploadtemps > 01053190d41ea421



macOS Client



At this point, John decided to call in [Aubrey Cottle](#), [Jackson Henry](#), and [Robert Willis](#) to investigate the issue further. Robert investigated the issue further while Cottle & Henry quickly spun up an instance of Keybase on macOS and were able to determine that the issue also existed on these platforms, but utilizing a slightly different filesystem path:

/Users/usernamehere/Library/Caches/Keybase/uploadtmps

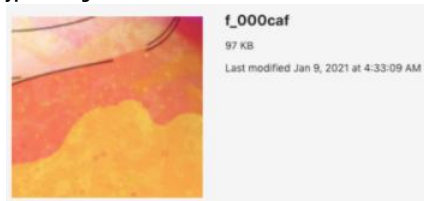
Additional Escalation

Sakura Samurai began to investigate further and that's when a similar issue was discovered within the "Cache" directory of the Keybase Client for macOS:

~/Library/Application Support/Keybase/Cache

The issue was similar in the sense that images were also being stored in this directory, unencrypted, however – the amount of images stored were far more in quantity than in the "uploadtmps" directory. In addition, the directory even included images that other users had sent us. In other words, a user could send a photo to another user via a private conversation, and click on the "delete" or "explode" button and the photo could still be recovered via the "Cache" directory due to the insufficient cache clearing issue/lack of encryption of the content. It was easily seen within mac because of the fileviewer functionality, but on windows a user has to change the file extension from its native extension to .png or .jpg

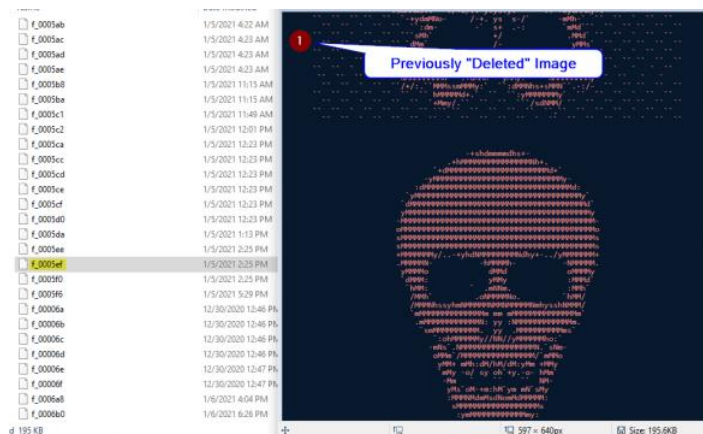
Recovered, unencrypted image on macOS



The vulnerability was quickly replicated on Windows by navigating to the Roaming folder of the AppData directory:

C:\Users\usernamehere\AppData\Roaming\Keybase\Cache

Recovered, unencrypted image on Windows



Specifically, the image of the skull above was sent to John Jackson by Robert Willis, and Robert had deleted the image – yet it remained behind in the Cache.

Impact

Normally, cached and cleartext photos wouldn't be considered a vulnerability, but in the instance of Keybase, it is. Keybase is known as an End-to-End Encryption (EE2E) Communication Application. In non-technical terms, this means that conversations between individuals or groups within the app shouldn't be persistently stored unencrypted on disk.

An attacker that gains access to a victim machine can potentially obtain sensitive data through gathered photos, especially if the user utilizes Keybase frequently. A user, believing that they are sending photos that can be cleared later, may not realize that sent photos are not cleared from the cache and may send photos of PII or other sensitive data to friends or colleagues. In addition, there are legal ramifications to such storage of information. For example, keybase is presenting itself as a secure end-to-end encryption solution. A vulnerability in such a sense could lead to private data being used in court cases against individuals, destroying Keybase's reputation as a secure and private communication platform.

Remediation

After reporting the vulnerability through Keybase's Bug Bounty Program, the vulnerability was swiftly accepted and a bounty was awarded. Keybase began working on the issue nearly immediately and Keybase addressed this issue on January 23, 2021 in versions 5.6.0 for Windows and macOS, and 5.6.1 for Linux. By updating to the latest version of Keybase, any unintentionally cached files are removed. No other action is required at this time.

Check out our website

<https://sakurasamurai.org>

Twitter Links:

Main Page

<https://twitter.com/SakuraSamuraii>

Founders

<https://twitter.com/johnjhacking>

<https://twitter.com/nicksahler>

Members

<https://twitter.com/JacksonHHax>

<https://twitter.com/Kirtaner>

https://twitter.com/rej_ex

<https://twitter.com/endingwithali>