stashcat

Cleartext Storage of Sensitive Information (CWE 312)

Jens van Laak

May 27, 2020

Stashcat is an instant messenger that is advertised to be safe and conformant to data protection regulations of the European Union. According to the vendor stashcat is used by German State Police in Niedersachsen and in Hessen and the German Armed Forces. It is as well the foundation for the application schul.cloud.

The webapplication of Stashcat stores cleartext session credentials and the private key for the end-to-end encryption persistent on the local system without user interaction and without the possibility for the user to opt-in or to opt-out.

A copy of or access to the local storage database file enables an attacker to login and get unlimited access to all data that the user has access to.

1 Overview

Vendor Stashcat Gmbh (owned by heinekingmedia GmbH)

Vendor Website https://www.stashcat.com

Product stashcat

Version $\leq$ 3.9.2

Product Website https://app.stashcat.com

Impact Information Disclosure

Attack Vector copy of persistent stored data from the user's system

2 Description

Stashcat uses an API to communicate with the system's server. Most of the requests are handled via HTTPS GET, POST and OPTIONS requests.

Two parameters – client_key and device_id – are send in each request and build the session key.

These parameters along with the unencrypted private key for the end-to-end encryption, the salt value for crypto functions, the emailadress of the user and a complete set of userinformation are stored *persistent* in the local storage[1] without any user interaction. The respective keys in the database are deviceID, clientSecret, im_privateKey, im_iv, dd_username and im_userInfo.

A logout function is provided, but the user has to navigate to the respective webfunction. If and only if the user logs off[2] via that function, the locally stored data is deleted. If a user just terminates the browser, all data is kept unprotected in the storage, because it is stored in the local storage database file by default[3].

A user can simply proof the fact that all access data is stored locally in cleartext by switching off the mobile device (and removing the battery if applicable). After the restart of the device a start of the app will immediately give access to all data (including the end to end encrypted text messages) without any password. For desktop browsers the same can be done by closing the browser (additionally restart the computer) and open the browser again. Retrieving the URL `http://app.stashcat.com` gives immediately access to all data (including the end to end encrypted text messages) without any password.

3 Description of the attack

A copy of or access to the local database file of the browser or app gives sufficient information to access Stashcat in the user context without any password authorization. The system does not ask for a login password nor for the key encryption passphrase.

4 Impact

The access to the persistent stored data opens the possibility to get full access to the user account.

In addition to the access to the not end-to-end encrypted files, the cleartext storage of the unencrypted private key enables to access all end-to-end encrypted text communication.

[1] e.g. webappsstore SQLITE database in case of Firefox, the respective .ldb file and the database logfile in case of Chromium

[2] this needs three clicks, two in the menu and one on a button

[3] The data is stored persistent all the time. It is deleted when the user uses the logout functionallity.

This violates *among others* even *basic*[4] rules of BSI[5] Grundschutz for secure webapplications.

In particular:

APP.3.1.A1 If the webapplication stores authentication data the user has to opt-in and needs to be informed about the assosiated risks.

The session id and the private key are stored without any opt-in (and without any possibility to opt-out). No information is given to the user about the risk of storage.

APP.3.1.A3 the session ID needs to be protected if stored on client. If the session gets invalid the session data needs to be deleted both on server and client side.

The session id as combination of clientSecret and deviceID parameter is stored unprotected in the local storage database file. The data on the client is not deleted if the session gets invalid by time, because the client is not necessarily connected to stashcat at that time.

[4]basic being the lowest of the three catagories: basic requirements, standard requirements and requirements for applications with higher security needs

[5]Federal Office for Information Security https://www.bsi.bund.de