Links for CVE:
https://nvd.nist.gov/vuln/detail/CVE-2022-32777
https://nvd.nist.gov/vuln/detail/CVE-2022-32282

Team:
Kangning Zhang U31078191
Michael Matta U39019639
Jiahe Zhang U82392079
John Antony U87410712
HyunOh Jeon U72239124

Short description:

AVideo is a video sharing platform where users can share and access videos online anonymously. The vulnerabilities exist due to the missing HttpOnly and Secure flag of the session cookie and pass value, as well as improper authentication of the hash value of the password. An adversary can launch a downgrade attack to downgrade a victim's browser to use http. Here a victim would be a logged-in user on AVideo. After downgrading or even without downgrading, the adversary can use XSS to steal the session cookie and the hash value of the password of the logged-in user. Due to the improper authentication of the password, the adversary can now login on the AVideo using the victim's session cookie and hash of the password.

Who can launch the attack?

Below we list the requirements an attacker needs in order to exploit this vulnerability:

CVE-2022-32777 (insecure cookies):

1) There exists an XSS vulnerability (some POST form or the like) on AVideo
2) Attacker sends user AVideo link with JavaScript executed on link open
3) Script provides the attacker with session and pass cookies

CVE-2022-32282 (improper password check):

1) Attacker uses XSS in order to access hashed password from insecure 'pass' cookie

   OR

   Attacker uses SQL injection or MITM to gain access to hashed passwords in website database
2) Attacker uses password hash to login to victim's account

Why does the attack matter?

AVideo is an online, open-source audio & video sharing website. It is also available on iPhone and Android and it advertises video organization tools, live streaming, and channel analytics.

Firstly, CVE-32282 describes AVideo's improper password check, which allows users to login using both their password, and the hashed version of the password. This vulnerability is exacerbated by the insecure cookies described in CVE-32777, which deems that the session cookie is without the Secure or HttpOnly attributes, and the "pass" cookie (which contains the hashed user password) doesn't have the HttpOnly attribute. Although the 'pass' cookie is not vulnerable to non-HTTPS requests, as it has the Secure attribute, both the session and password cookies will be vulnerable to JavaScript via an XSS attack.

Because of these cookie vulnerabilities and improper authentication at login, attackers could launch a cross-site scripting attack in order to steal the user's password hash and then login to the site successfully with the hashed value. In the worst case, this could result in videos on the platform being manipulated or deleted permanently, which would be a disaster for content creators and would also shatter trust between AVideo and its users.

Additionally, because of the improper password check, a SQL injection that exposes password hashes could easily be utilized to login to any account in the AVideo system, including administrator accounts. Again, the risk of this could be the permanent loss of video files, account details and analytics. This could be for a single hacked account, or in the case of an administrator account, this could impact multiple accounts and channels on the site.

Attack in Further Details:

There are two ways to launch an attack. First is by XSS to use javascript to access the cookies saved in an active user's browser. The second is to be MITM and downgrade the TLS version, which allows brute force the keys used to encrypt communication between a user and the server. And thus, all the information needed for login can be obtained.

1. XSS attack
   Launch an XSS attack by tricking an active user of the website to open a malicious page that contains javascript to access the cookies saved in the browser. Since the session cookie and the pass cookie is not HttpOnly, they are accessible. Due to the improper authentication of the password, using the pass cookie, which is the hash of the user's password, is sufficient to log in to the victim's account.

2. MITM - downgrade attack
   Intersect the first GET request sent from the victim to the server. Change the header to request lower versions (less secure) of TLS running broken crypto from the server - establishing MITM. Then brute force the shared key between the server and the victim. With this key, an adversary can decrypt and modify all the communications between the

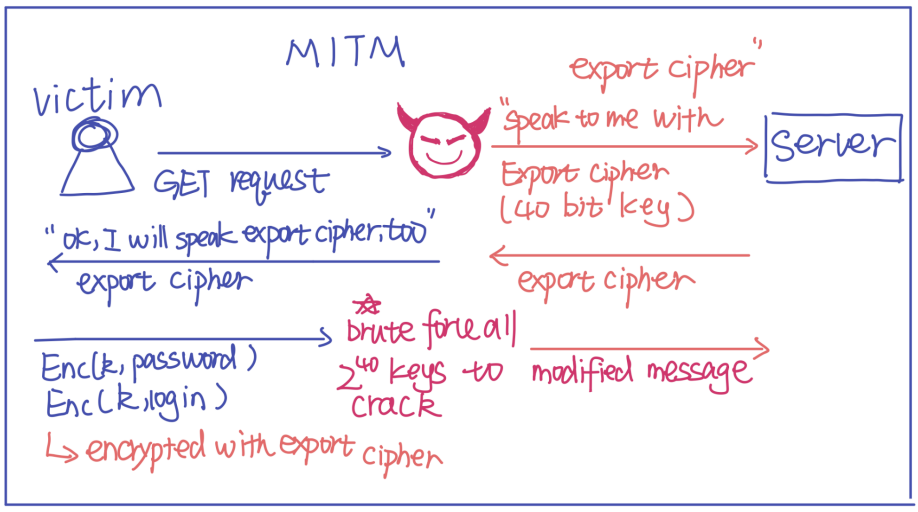
victim and the server, including the ones that contain the victim's password and login information. Therefore, gain access to the victim's account.

How is it fixed?

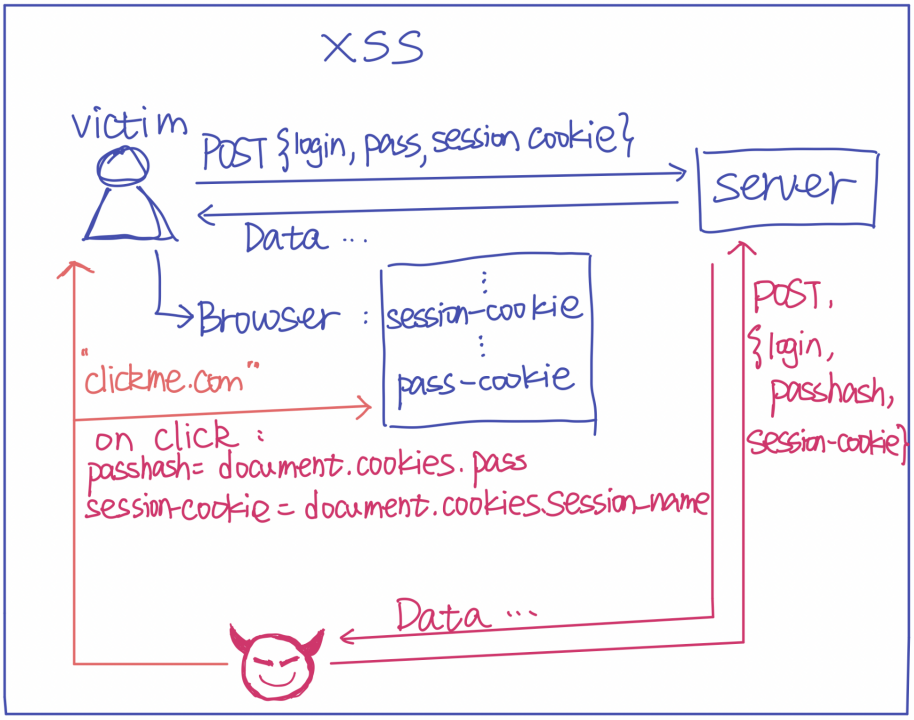
The problem is fixed by
1. Setting HttpOnly flag to true for the session cookie and the pass cookie; changing the browser to which supports HttpOnly. (CVE-2022-32777) By setting the HttpOnly flag to be true in the HTTP response header, the cookies are set to be inaccessible through client-side script. By making sure that a browser supports HttpOnly, when client-side script code attempts to read the cookie, the browser returns an empty string as the result as it detects the HttpOnly flag. This causes the attack to fail by preventing XSS code from sending the data to an attacker's website. Therefore, even if a cross-site scripting (XSS) flaw exists, and a user accidentally accesses a link that exploits this flaw, the browser will not reveal the cookie to a third party. The vulnerability is fixed.

2. Started using the PassHash instead of database pass in all sites, and soon disabled the login with the password hash (database password field) directly. (CVE-2022-32282) According to the official post from Cisco Talos, PassHash works differently from the original approach using a database to store passwords. It is an encrypted JSON and has an expiration time, and also will be automatically rejected if the original password is updated. Since the vulnerability is a consequence delivered from the improper password check, by replacing the original password storage method with PassHash, an attacker that owns a user's password hash will not be able to use it to directly log into the account since the password is encrypted with an expiration time. PassHash would reject the attacker if he tries to use the old hash to log in.

Threat Model



Consider an adversary who can intercept HTTP traffic sent by a AVideo user. When the adversary sees the GET request, which is usually the very first message sent from a user to the server in order to log in, intercept the request, change the header where it specifies the cipher to Export Cipher (40 bit key) or other broken ciphers and ask for lower versions of TLS (for example TLS 1.2). Send the modified request to the server. Upon recipient of the response from the server, send the same response to the victim. At this point, the victim and the server should have established a shared key (using the broken cipher) for communication. Since both the pass cookie and the session cookie don't have the flag secure set, they will be sent over to the lower version of TLS. The adversary can intercept the next message from the victim, brute force all 2^{40} keys to crack the message and also recover the key. From this moment on, the adversary has established MITM, and is able to decrypt and modify messages between the user and the server. Naturally have access to the victim's credentials and thus have access to the victim's account.



With cross-site-scripting, the approach is straightforward. An adversary who is able to construct a "legit-looking" website link, only needs to trick a user into clicking the link. Within the "legit-looking" website, there will be javascript embedded that will steal the cookies saved in the user's browser. Since neither the pass cookie nor the session cookie has the HttpOnly flag set, they are accessible via javascript. Thus, the adversary login to the website using the victim's pass cookie and session cookie (thanks to the improper design of password authentication of the website).

CVE Vector Justification

CVE-2022-32777 (insecure cookies):

Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

Attack vector: Network
Since this CVE doesn't require local or adjacent access, the adversary is able to launch an attack as long as they have network access.

Attack complexity: Low
A successful attack of this CVE would mainly depend on the attacker's capability. So we would expect repeated success on the vulnerable component using the same approach.

Privileges required: None
An adversary doesn't need any privilege in order to exploit this CVE. Launching an attack doesn't require the adversary to be able to sign in to the website as a normal user.

User interaction: None
The CVE can be exploited without user interaction if the attack is via MITM. However, if using XSS, the user will need to click a link that was manufactured by the adversary in order to launch the attack. The rating is reasonable since there exist exploits that don't require user interaction.

Scope: Unchanged
Upon a successful exploit, only the victim's account is affected. The adversary won't be able to change anything outside the victim's account.

Confidentiality impact: High
Upon a successful exploit, the adversary now has full access to the victim's account. This results in a total loss of confidentiality and all resources inside the victim's account are divulged to the attacker.

Integrity impact: None
This categorization is a bit misleading. Since a successful attack would grant the adversary access to the victim's account, naturally, the adversary would be able to change anything in this account. This is breaking integrity completely.

Availability impact: None

Again, this categorization is also misleading. Once an adversary has access to the victim's account, they can change anything, including the password. In this case, the victim's account won't be available to the victim anymore.

CVE-2022-32282 (improper password check):

Vector: CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H

Attack vector: Network
This CVE does not need any access to an adjacent or local network, which means that the attacker is able to attack the vulnerable component remotely as long as they have internet access. Thus the attack vector is accurate.

Attack complexity: Low
The attack can be launched if the attacker obtains the username and password hash of a user, and we expect repeated success on the vulnerable component using this method. Thus, attack complexity is indeed low.

Privileges required: Low
NVD has scored this category as "Low" but Talos scored this category as "High." We believe rating "high" for this field is more reasonable. In order to launch an attack, the adversary will need to know the username and password hash of a user. However, administrative power over the vulnerable database is normally needed to obtain password hashes.

User interaction: None
This is correct as there is no need for interaction from the user to exploit this vulnerability.

Scope: Unchanged
Although the hashed password can be used to breach users' accounts without permission, there is no change to how the system is intended to work and therefore the scope is unchanged.

Confidentiality impact: High
The victims will have a total loss of confidentiality by a successful attack. All contents of their account will be compromised, and whatever the user has (or had) access to will be in control of the attacker.

Integrity impact: High
Again, the victim will lose all control of their account, and the account can easily be wiped or have the password changed. In the case of a larger-scale database leak, the entire site's contents

could be erased or made unavailable. Therefore the integrity impact of this vulnerability is high for any potential victim.

Availability impact: High
Similar to the confidentiality and the integrity impacts, the availability impact should be high as well because any accounts breached using this vulnerability can be completely stolen, deleted, or have the username & password changed such that the victim would never be able to access their account again. Once again, in the case of an admin account hack or SQL database breach, this could affect the entire site rather than just an individual account.

Resources Used:
https://talosintelligence.com/vulnerability_reports/TALOS-2022-1542
https://talosintelligence.com/vulnerability_reports/TALOS-2022-1545

https://cvss.js.org/#CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N
https://owasp.org/www-community/HttpOnly