A Security Analysis of Qik and LiveCast

December 6, 2010

Dillon Yang, Tsung-Ying Tsai, Nick Hanssmann, Chia-En Pan

Department of Electrical and Computer Engineering
University of British Columbia
Vancouver, Canada

dillonykyang@gmail.com, nickhanssmann@gmail.com, jonestsai512@gmail.com, penguin_1986@hotmail.com

***Abstract* — As the number of smart phones in the North American market increases, so does the potential for security risks. Increasingly more smart phone users are enjoying the ability to stream live video capture from their portable devices. The websites that host these services need to provide privacy and security for their clients. In this paper, we analyzed the overall security of two popular live streaming websites: Qik and LiveCast. As a result of our analysis, vulnerabilities such as brute force password attacks, denial of service, and user location extraction were present in the systems. These vulnerabilities threaten the availability of the two websites as well as putting the users' private information at risk. Countermeasures for these vulnerabilities are suggested according to the principle of designing secure systems.**

I. INTRODUCTION

QIK and LiveCast are mobile streaming platforms that allow users to stream live videos from their cell phones to the internet. The capability of live streaming must be easily accessible by the users and at the same time provide security. With more than 3 million users utilizing such services, the risk increases if these websites are vulnerable to attacks. By performing security analysis on both Qik and LiveCast, we can find out the likelihood of such systems being compromised by attackers.

We used techniques such as brute forcing passwords, SQL injection, cross-site scripting, social engineering, and denial of service when we carried out our analysis on Qik and LiveCast. Our result shows that both website are subject to brute force password attacks, social engineering, and denial of service attacks. If a successful password brute force and social engineering attack is carried out, users' private data can be disclosed or modified. Denial of service attacks can be used to disable access to user accounts or resources of the websites.

According to our analysis, Qik has violated the principle of separation of privilege because the website does not authenticate the user requesting a password reset. As a counter measure, Qik can ask the user for more information before performing a password reset. The system designers for both Qik and LiveCast also violated the principle of question assumptions by incorrectly assuming that users will perform normal operations through the graphical user interface. Various HTTP attacks such as brute-forcing passwords can be carried out. As a countermeasure, unlimited HTTP post messages from malicious users must be blocked.

II. RELATED WORK

Since the birth of the World Wide Web (WWW) in 1990, the number of Internet-based firms such as Amazon and Google has increased rapidly [1]. These companies rely on the Web servers to provide services to clients around the globe through the Internet; therefore, many security experts and system designers have done detailed analyses on the security of these servers to ensure

that the confidentiality, integrity, and the availability of the customers and companies are well-protected.

Many web security analysis conducted by these experts involved scanning the servers for vulnerability by launching different types of known attacks at the server [2]. For example, some of the attacks are SQL Injection, Cross-site scripting, parameters tampering, spoofing packets, and spamming HTTP messages. We first perform these existing attacks on both Qik and LiveCast to see how both systems respond to these attacks and to see if they are more vulnerable to specific types of attacks. Then, combining these results and our own strategies, we tailor our attacks to exploit the specific security loopholes of these servers.

III. FAILED ATTACKS

A. *Structured Query Language (SQL) Injection*

Structured Query Language (SQL) injection is a code injection technique that exploits security vulnerabilities in the database. If user input is not sufficiently validated, the malicious user can use SQL injection to bypass authentication, disclose information, and compromise data integrity and the availability of data. After attempting SQL injection on user input fields of both Qik and LiveCast, the sites returned with generic error messages. This result shows that user inputs were correctly filtered by the servers.

B. *Cross-Site Scripting (XSS)*

Cross-site scripting (XSS) is a security vulnerability that enables attackers to inject client-side script into web pages viewed by other users. This attack occurs when a user sends a malicious script to a text input. Another end user's browser cannot detect if the script is from a trusted source. As a result, it is likely for an unsuspecting user to execute the script. XSS can be used to steal cookies, session tokens, or other sensitive data retained by the user's browser [3]. Both Qik and LiveCast would not display scripts as inputs. Various types of XSS attacks were attempted without any success. This result led us to conclude that user inputs were correctly filtered by both sites.

C. *Parameter Tampering*

Parameter tampering is a form of attack that manipulates the exchange of parameters between the client and the server. Such attacks are performed by malicious users in order to modify application data such as user credentials and user permissions. WebScarab, an application that allows users to modify data created by the browser before sending them to the server, was used to check if Qik and LiveCast were vulnerable to these attacks. However, both websites blocked any parameter tampering attempts.

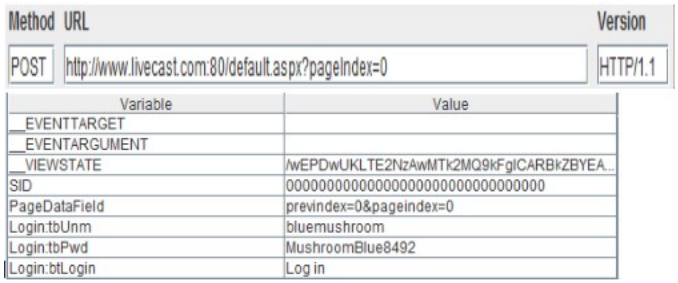
IV. SUCCESSFUL ATTACKS

A. *Brute Force Password*

Brute force password finding refers to actively guessing the password of a target account or accounts. The time it takes to successfully guess the password of a single target account depends on the length of the password chosen, and the speed at which attempts can be made. We explored the possibility of an online brute force attack being exercised on Qik and LiveCast. On both sites, there were no security measures taken to directly reduce the possibility of an online brute force attack. With enough time, any account could potentially be broken into. A password dictionary can be used to increase the likelihood of guessing the password quickly. If the attacker's goal is to get into *any* account, the password dictionary could be used on one account, and if unsuccessful, used again on the next account. Since each site allows for a password of length one, there is a good chance that many of the accounts are protected by a very weak password, and could be hacked easily.

We experimented on this prediction by setting up a test account with one of the passwords from our dictionary. Publicly available brute forcing software such as Brutus and Hydra did not work as we had expected, so we wrote a simple program specifically for Qik and LiveCast. Firstly, we analyzed the outgoing HTTP POST messages during login using WebScarab. Then we were able to reconstruct these messages in our program and

execute them. The program can be run on several machines with each testing separate parts of the dictionary, checking each password with the given account name. Our results showed that the program guessed the password in a few hours with a common password. This demonstration shows that accounts with a weak password are vulnerable to an online brute force attack.



Method	URL	Version
POST	http://www.livecast.com:80/default.aspx?pageIndex=0	HTTP/1.1

Variable	Value
__EVENTTARGET	
__EVENTARGUMENT	
__VIEWSTATE	/wEPDwUKLTE2NzAwMTk2MQ9kFgICARBkZBYEA...
SID	00000000000000000000000000000000
PageDataField	previndex=0&pageindex=0
Login:tbUnm	bluemushroom
Login:tbPwd	MushroomBlue8492
Login:btLogin	Log in

Fig. 1. This is a captured HTTP POST message during LiveCast Login using WebScarab

Another option we looked into was using the reset password function to generate a password of known length and complexity and try a brute force attempt on that. On either site, a password is reset to length 6 with both lower case characters and numbers. This leaves 36^6 possibilities. To perform an online attack using our program would take as long as two months [4]; therefore, it is beneficial to not use the password reset function, and instead hope that the victim's original password is of a reduced complexity, or a commonly used term.

B. Social Engineering

Social Engineering is a type of hacking that, rather than forcing entrance into a system or using technical cracks, exploits the trust a user may have for some known organization [5]. A common example of social engineering is email phishing. This technique is no different for Qik or LiveCast than other examples of social engineering; however, some of the security flaws found on the site make it easier to carry out a phishing attack.

A vulnerability we found was that Qik does not provide proper transparency for their email list. It is possible, through the password reset function on their site, to retrieve email addresses of Qik users. These addresses can then be targeted directly for phishing, which would greatly increase the "hit rate" of the scam.

On LiveCast, alternatively, an attacker is able to spam comments to any live stream chat. The attacker can gain access to a list of webcast IDs, which identify different live streams, and run a small program to cycle through those streams and post a link to their phishing site on each. We were able to successfully spam comments into any chat using any username. Interestingly, we found that using our program we were able to spoof a username that had special characters in it as well, even though they aren't allowed on account creation. The most effective use of this flaw would be to spam a link to a phishing site using a username like "LiveCast" or ">System" in every active chat room.

C. Denial of Service

Denial of service (DoS) attack is an attempt to depreciate one's computer resources such as network bandwidth, or processing capabilities. On both Qik and LiveCast, there were no security measures to determine how many times or connections at a time a user can access the web services or perform information retrieval. By sending multiple requests to a user's account information, the server's database may try to look up the same request sent by an attacker and becomes too busy to serve the actual users [6]. With enough computers and resources, an intruder may temporarily bring one of the websites down.

We also looked into the website features provided by Qik and LiveCast to determine if denial of service attacks were possible on a particular service. On both websites, DoS attacks can be initiated by spamming messages in the video comments section to prevent legitimate users from viewing posted comments. For Qik, we also found that it has no protection against automatic registration. An attacker may simply register multiple accounts to overload the server database. Moreover, malicious user can keep resetting a legitimate user's password multiple times to prevent the user from logging into the account. Qik did not authenticate the user before a password reset is performed.

D. View User Locations

In LiveCast, we found a security exploit that allows an attacker to potentially see the location of

any user who is streaming live. If a user has GPS activated on their streaming devices, or if they choose to use the "predefined location" option, live positions can be seen and updated while streaming. A video can be accessed through an ASHX extension server file that is used to verify client's identity if the video is set to private. The team has found another way to get a video's live location in private live streams. By inspecting packets from Wireshark, we were able to monitor a stream of keywords and the video identification that we are watching. After further investigation into these packets, we found that there is a constant 25 KB/s of data being transferred which contains that information. We attempted to break down the information in that stream and found out that it contains plain text formatted in JavaScript Object Notation (JSON), a lightweight data interchange format. If there is a live event, the stream notifies the website by embedding the video's GPS information and video's identification which we can get from inspecting the HTTP POST messages from FireBug, a FireFox plugin. In detail, we have used Microsoft Silverlight for the media playback and inspected the results each time there is a "MarkerReached" event. When a MarkerReached event happens, a full JSON formatted plain text will contain video's live location regardless of the video's privacy. This location is given in the form of GPS coordinates, and can be easily plotted on a map. Such exploit breaks the *confidentiality* rule in the CIA property in information security because users' locations are exposed while they are streaming regardless of the videos' privacy setting.

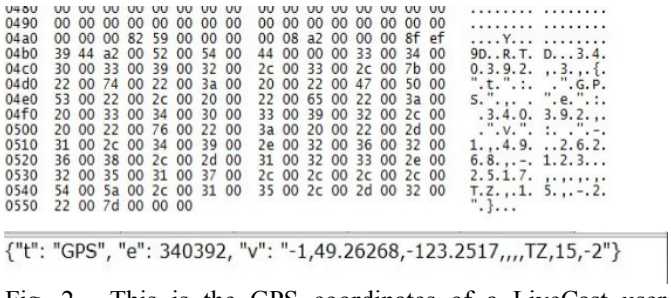


Fig. 2. This is the GPS coordinates of a LiveCast user captured using WireShark

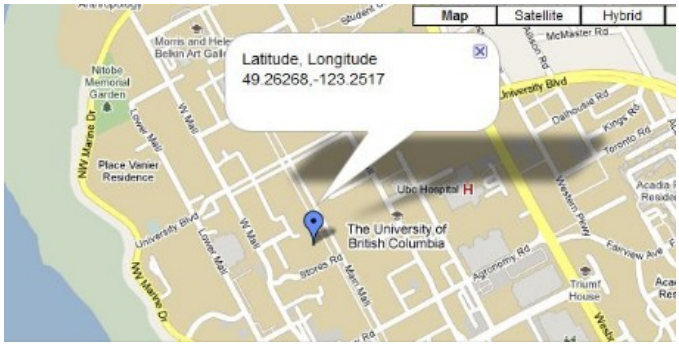


Fig. 3. This is the GPS coordinates of a LiveCast user viewed with Google Map

V. RESULTS

This section provides a brief summary on the vulnerabilities that we discovered and the attacks that failed.

TABLE I
VULNERABILITIES FOR QIK AND LIVECAST

Both Qik and LiveCast	Qik	LiveCast
Brute-force password finding	Allowing password reset with only phone number or email	Location exposure for live streams
Spamming comments	Email not verified upon account creation	Allowing weak passwords
Social engineering		
Denial of Service attacks		

This is a list of the vulnerabilities on Qik and LiveCast

The failed attacks include:

- SQL Injection
- Cross-site Scripting
- Parameter Tampering

VI. DISCUSSION

A. Security Principles Violated

1. Separation of Privilege

On Qik, the account password can be reset by providing either the phone number or the email address associated with that account. Based on our

finding, we were able to perform a denial of service attack by spamming the password reset page with HTTP Post messages filled with randomly generated phone numbers. This attack is successful because Qik's password reset function violates the "Separation of Privilege". Without another separated mechanism to authenticate the user requesting the password reset, the attacker can disrupt the daily operations of the public users and reduce the availability of the system.

On the other hand, LiveCast does not have this flaw in their system. The password resetting feature provided by LiveCast requires both the user name and its associated email address to proceed.

2. Question and Assumption

According to our experiment, both Qik and LiveCast are vulnerable to various HTTP attacks such as brute-forcing password on account log-in, and generating spam and phishing contents at videos. These attacks involve reconstructing the HTTP Post messages that are generated and sent to the server when the user presses certain buttons on the browser. The system designer violates the principle of "Question and Assumption" by incorrectly assuming that all users only perform these operation (log-in and post comments) through the graphical user interface (GUI) provided by the browser. The user's confidentiality and the company's integrity are threatened as a result of this vulnerability.

B. Proposed Solutions

1. Separation of Privilege – Countermeasure

A phone number or an email address alone is not secure enough to authenticate who the users really are; moreover, authorize them to reset their account passwords. In order to prevent the availability of the user accounts from being compromised, Qik should request another security token from the user performing the password reset. Similar to LiveCast, Qik can ask for the user to provide the name of their account, in addition to the existing information. Another way is to ask the user to create a couple "secret" questions and answers upon account creation; such that, the system can then use these questions and answers to verify the identity of the user. Another more defensive approach is to disable the password reset function on important accounts, such as accounts owned by organizations or companies. For these account, the password can only be modified by requesting the system administrator to manually reset it.

Even though these options provide better security of the system, they increase the number of operations that the users need to perform for the same task. The system designer must find a balance between the two to not violate the principle of "Psychological Acceptability".

2. Question and Assumption – Countermeasure

In order to prevent unlimited automated log-in and comment posting via HTTP Post messages, the system designer needs to implement mechanisms and establish policies to reduce or stop the attackers from sending these HTTP requests. First, the system can enforce an exponential back-off time interval between requests coming from the same IP address. In addition, the system can choose to temporarily ban hosts that are sending too many HTTP requests within a certain time frame, or HTTP messages containing malicious contents (posting comments about malicious sites or phishing contents). Furthermore, users should be asked to complete a CAPTCHA or other forms of challenge-response when they attempt to log in and post messages; such that, these operations are difficult to be automated by the attackers.

On the topic of account log-in, system should not allow more than a certain number of retries when the users continuously fail to provide the correct password. Upon account creation, the system should not allow weak passwords in order to increase the difficulty of brute-force password attack.

VII. Conclusion

Our analysis done on Qik and LiveCast has discovered several security vulnerabilities. This security analysis is focused on both consumer (Qik) and enterprise (LiveCast) based services. It has been shown that both websites violate

confidentiality, integrity and availability, the three information security principles. Furthermore, we have found these websites to be vulnerable to several attacks including: brute force password finding, denial of service, social engineering, comment spamming, account reset spamming, and location extraction. Weakness is also shown in many areas including password strength, lack of CAPTCHA, and general ease of account services such as password reset or account creation. With the countermeasures we have provided invoked, these sites can be safe from any further attacks.

References

[1] S. Garfinkel, "The Architecture of the World Wide Web," in *Web security, privacy and commerce*. Sebastopol, 2002, pp 13-33.

[2] A. Tiwana, "Maintaining a Security System," in *Web Security*. 1999, pp 319-326.

[3] Huseby, Sverre H. Innocent code : a security wake-up call for Web programmers. New York: John Wiley & Sons, 2003.

[4] Online Password Calculator [Online] http://lastbit.com/pswcalc.asp [Accessed: Nov 7, 2010]

[5] Logsdon, Tom. Computers & social controversy. Potomac, Md.: Computer Science Press, 1980.

[6] Association for Computing Machinery. Special Interest Group on Security, Audit, and Control. Proceedings of the ACM Workshop on XML Security. New York, N.Y. : ACM Press, 2003