Security Analysis of Github

Sean Smith, Kyle Holzinger, Amalia Safer

April 29, 2015

1 Introduction

INSERT INTRODUCTION HERE

2 Cookies

A session cookie called `user_session` is stored which contains a seemingly random nonce. When a get request is made to `https://github.com`, the cookie is sent and the database is queried to see if the cookie is valid. If the cookie is valid it will return user data as if the user is logged in. There are two more cookies of importance `logged_in` which is a yes/no value and `dotcom_user` which is the user's username. To impersonate a user, only the `user_session` is needed, the `logged_in` cookie will always be yes and the `dotcom_user` will be filled by the server if the `user_session` cookie is valid. When the user logs out, the cookie is invalided by the server in a post request to `https://github.com/logout` that contains the content type and an authenticity token. The authenticity token is a random nonce that github uses to prevent against CSRF. If an attacker is using the user's cookie and the user logs out, the attacker's cookie will be invalidated.

When the user logs in, a post request is sent to `https://github.com/session` with the username, password and authenticity token.

One attack is to guess a random cookie and query to see if it's valid. There are approximately 8 million active github users at a time so roughly 8 million valid cookies. Since you don't need the `logged_in` cookie to be set correctly, you can construct a random cookie and check if it's valid. The length of the cookie is 80 characters and each character is from the universe

Figure 1: Browser Fingerprint

(a-z, A-Z, 0-9, -, _) which has a size of 64. Say the set of correct cookies S has size $|S| = 8,000,000$, the universe U has a size of $|U| = 64^{80}$. The probability of guessing a correct cookie c is so low that it's not a reasonable attack.

$$Pr[c \in S] = \frac{|S|}{|U|} \approx 10^{-138}$$

3 User Tracking

Github does not serve ads as it's business model revolves around selling premium subscriptions. However it does track users for analytics purposes via Google analytics. It does this in two ways:

1. Google analytics sticks cookies on the user's browser. First it sticks a `_ga` cookie that expires in two years to distinguish between individual users, then it sticks a `_utma` cookie that expires in 30 years that is updated every time a request is sent to google analytics.

2. In the rare event that the user removes these cookies, google analytics tries to fingerprint the user. It collects the browser, operating system, extensions installed, model of the computer and a couple other distinguishing factors. This is collected from the `user_agent` header which contains information such as `Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/42.0.2311.90 Safari/537.36`. Users who do not allow cookies are even easier to identify than other users as the server identifies that they don't allow cookies and then it's a smaller pool of people that they may be. The remaining identifying information is enough to identify the user. From my browser setup 22 bits of unique identifying information is available. Enough to make me distinct in a pool of $5,000,000$ people.

4 References

1. We used this this online tool to fingerprint the browser much in the same way that Google Analytics does.
`https://panopticlick.eff.org/index.php?action=log&js=yes`