**Security in Web Transfer Report**

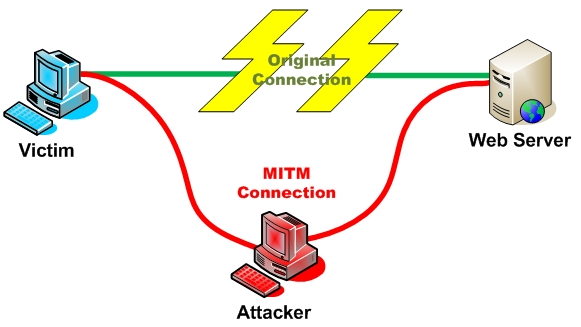
------ **Man-in-the-Browser (MitB)**

**1. Introduction**

This is a project that tests the secure level and reliability of the “web transfer method”. In this project, I tried to attack the websites which use “web method” to transfer data from client end to the server end. In other words, the data in the database on the server side will rely on what the clients send to them by, maybe, browser.

For different websites, they usually have different secure levels and they may use different methods to prevent users cheating their servers. That means, the owners of the websites will suppose there will be many potential attacks which come from the users and they should take some necessary defense methods.

Actually, the main idea of my attack method will be “man-in-the-middle” (man-in-the-browser):



“Intercept a communication between two systems. For example, in an http transaction the target is the TCP connection between client and server. Using different techniques, the attacker splits the original TCP connection into 2 new connections, one between the client and the attacker and the other between the attacker and the server, as shown in figure above. Once the TCP connection is intercepted, the attacker acts as a proxy, being able to read, insert and modify the data in the intercepted communication.” What I will do in this paper is, act the bad guy who is called “attacker” in the figure above, intercept the requests which come from the client end and the responses which come from the server end. Check them, modify them and then send them again to the server (client) end so that I can cheat both sides of the client-server.

In this paper, I found different secure levels’ websites. Analyze the different methods they use, design different attack methods to cheat on them and try to implement the attacks successfully. I will show the details of the attack design, implementation, the strength and weakness of different defense methods, the analysis of them and the probable way that can avoid them.

**2. Tools**

Before I start, I should introduce some tools I may use in the next tests.

First, Sothink SWF Decompiler

This is software that can decompile the flash file. That is, if we have the “.swf” file, we can use this tool to decompile the “.swf” file into “.fla” file so that we can see the source code of the “.swf” file. I may need it is because some data transfer methods may be hidden in the source code of the flash file but not in the webpage’s source code.

Second, WebScarab

This is the core tool which will be used in the following tests. WebScarab is a web security application testing tool. It can intercept the request and response packages of the web browser so that it makes the data modification possible. I will cheat the website’s server in my next tests, so I need such a tool to help me intercept and modify the data which is sent to the server.

Thirdly, D5Calc or www.cmd5.com

MD5Calc is a local tool that can compute the MD5 value of a file or a string. And “www.cmd5.com” is a website that can compute the MD5, SHA1 and many other values online. I need these tools is because in most cases, the data is not transferred in plaintext but in ciphertext which is encrypted with MD5 or SHA. If I have these tools, I can check the MD5 or SHA values so that I can establish my own fake values to instead of them.

**3. Tests**

**3.1 Test 1:**

**3.1.1 Target:**

www.baidu.com

**3.1.2 Description:**

“www.baidu.com” is the “Google” in China. And “http://baike.baidu.com” is the “Wikipedia” in China. At the bottom of the webpage, there is a button called “If this item is helpful”. It means if you feel this item is helpful, you can click it so that someone others can know the item is good. In other words, it likes the “like” function on FaceBook, “if you like it, then click it!”. “445” in the figure means there were already 445 people think this item is good.



**“Like” button**

**Means “445 liked”**

However, you can click this button only once because the owner of the webpage wants to prevent cheating. That is, the number “445” can be only added “1” to “446”.

**3.1.3 Attack goal:**

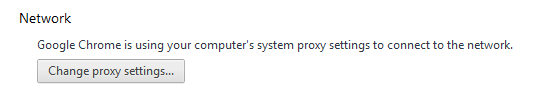
I want to break the rule so that I can make it not only “446” but any number I want.

**3.1.4 Attack Design:**

Because the action “click the button” relies on users which means if users doesn’t click the button there will be no data transferred to the server; and if the user click the button, there will be something transferred to the server to let the server know the number should be added “1” or something. Thus, I plan to use webscarab to intercept the request which comes from the broswer when I click the button and see what broswer tries to tell the server. And then I will see if I can establish a fake data to instead the real data so that to achieve my goal.

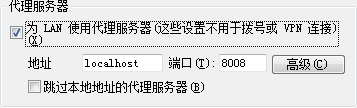
**3.1.5 Test Begin:**

Before I use the WebScarab, I need to set the proxy so that I can make it work normally.



🡪

**“Proxy”**



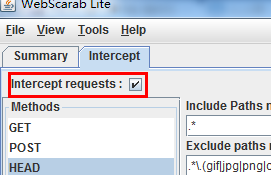
**“Use proxy for LAN (All these setting are not used for dialing or VPN connection)”**

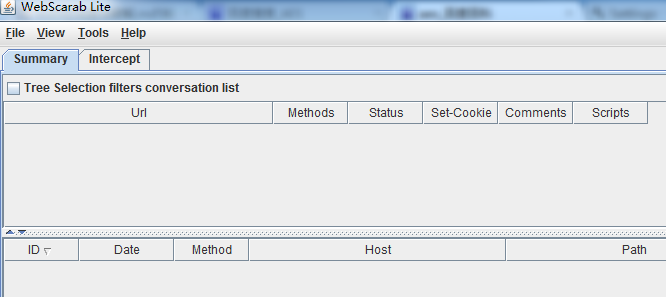
**“Port”**

**“Address”**

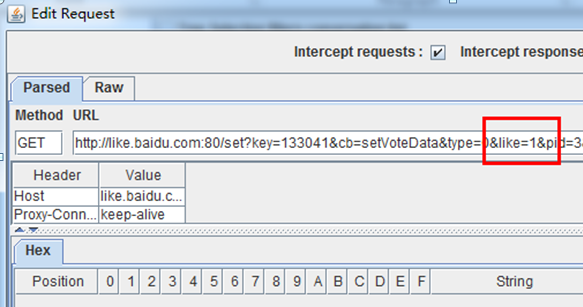
I’m sorry for the chinese words shown in this figure above. It’s because I used my own computer to do the tests, and my system language is chinese. But I have translated them into english(red words) so that it could be easier to understand.

First, check the “intercept requests” box to make sure WebScarab can help me intercept the information sent to the server.



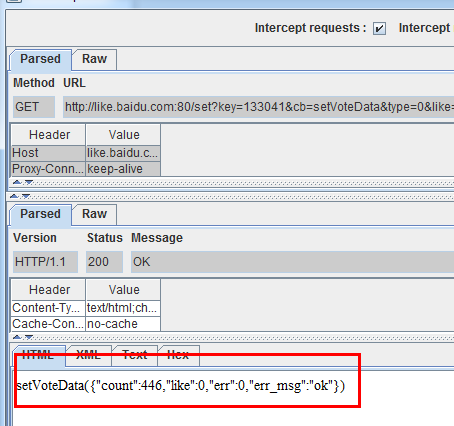


Then come back to the webpage “http://baike.baidu.com/view/133041.htm”. Click the “like” button. When I click the button, there is a window pop out. It is because when I click the button, I will trigger the action of this button, some messages will be sent to the server and WebScarab will help me intercept them:



We can see the red part, “like = 1”. I guess it may mean that the “445” will be added with “1”. But I want it to add with “100”, thus, I changed it to “100” and send it.

Then another window pops out:



We can see the red part, “count” is 446 which has been added “1” from “445”, consequently, “like” is “0”. But what I want is to make the count with “545”, then I change the data and send it.



After that, come back to the webpage to see the result:



We can see the result has become “545” which has been added “100” from “445”.

**3.1.6 Conclusion:**

From this test, we can learn that if the broswer simply use the plaintext of data to transfer, it will be very unsecure because it’s easy for a “chrome” user to see the source code of the webpage so that bad guys can easily know the actions of buttons or links. And if some bad guys can check the source code of the webpage, then they may also intercept the packages so that they can easily know what information the broswer send to the server by using very simple tools like WebSacarb. Consequently, for a owner and designer, hide the critical part of the source code or use some encryption methods are pretty necessary.

**3.2 Test 2:**

I am a guy who is very interested in the computer games, but sometimes it seems I am not a very good player. That is because on the “Top 10 list of the score”, many people’s grades are so high that I cannot believe they can really do that well. Consequently, I think there could be only one way for them to get such high scores --- cheating! So I am just curious how they can cheat. And that’s also why I come up with this Online Flash Game test.

**3.2.1 Target:**

http://flash.17173.com/flashfile/2012-09-28/20120928144541431.shtml

**3.2.2 Description:**

This is an online flash game which is called “catching moles”. After playing the game, users can know their scores and rank.

**3.2.3 Attack Goal:**

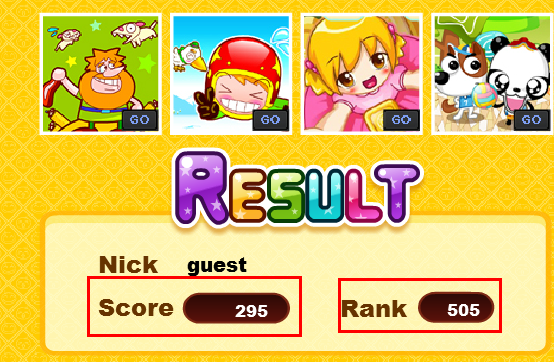
Make my rank to “No. 1”.

**3.2.4 Attack Design:**

I will try the method which is used in the first test first because the WebScarab can simply intercept any packages that sent to server. However, because it’s a flash game, there may be some information transferred inside the flash that we can hardly know the details. But anyway, I will try the method which is used in the first test first. That is, I will check the information which is sent to the server when I finish the game to see if there are some plaintexts that can be easily understood.

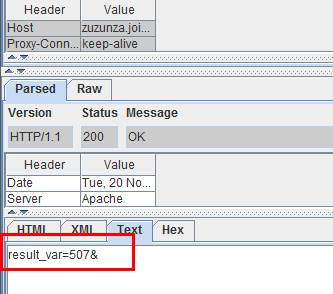
**3.2.5 Test Begin:**

First, to establish the compared group, I played the game normally:



From the red parts, we can see that when I get 295 scores, my rank is 505.

Then I open the WebScarab, and play the game again. When I played the game this time, I intended to get only few scores because I want to make it comparing with the normal one. WebScarab popped a window which intercepted the packages which will be sent to the server when game was over:



We can see that there is a variable called “result\_var=507&”, I guess it may be my rank of this game, thus, I modified it with “1” and come back to the game. Then I found my rank was “1” even my score was only “5”!



Then I want to test what if I don’t change the rank while my score is only 5.



From the figure above, we can see that the rank is not “1” anymore. That is, my guess about the variable is correct.

**3.2.6 Conclusion:**

Though it is an online flash game and it has chance to make the transferred information more secure. But it still uses the plaintext of the rank simply as a readable variable to transfer to the server, it is full of risk if there is a bad guy like me, the server will be cheated easily. But I believe there must be some others are much better than this one.

**3.3 Test 3:**

**3.3.1 Target:**

www. sockandawe.com

**3.3.2 Description:**

It’s also an online flash game website. When finish the game, it will let you file your name and email address. If your score is high enough, you can be in the Top 10 list of this website.

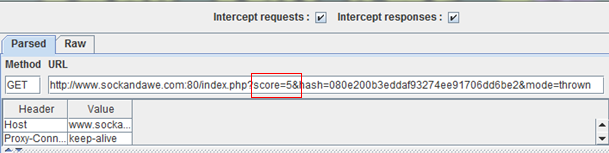
**3.3.3 Attack Goal:**

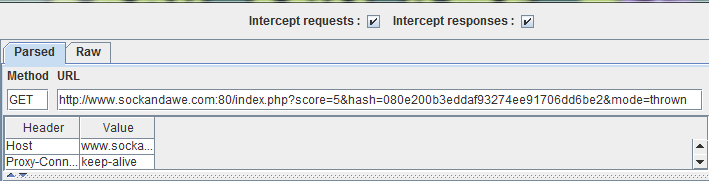
Make me get the highest scores

**3.3.4 Attack Design\_primary:**

Try the simplest method just as what I did in the first two tests to see what happens.

I open the WebScarab to intercept the packages:





Hmm, it seems there is no difference with the second one…The score is also in plaintext which is “5”.

But wait! There is also a hash value! What is that?

Obviously, I cannot guess what the hash value is because it is a string which is meaningless!

Consequently, I need to design my attack again!

**3.3.5 Attack Design\_Advanced:**

First, we can see the player who get the highest scores will be shown on “Top 10 list”, thus, there must be some information sent to the server of the website. And because it is a flash game, how my performance is can be only recorded by the flash program itself. Consequently, I will need to know how this flash program works, that is, I need to know the source code of this flash program. What I am thinking now is I can download this flash program to my local device and then decompile it so that I can get the source code of it.

Secondly, when I get the source code, I will try to find which part of the code is controlling recording my score, which part of the code is responsible for sending my score to the server and which part tells server the current information is authorized and correct.

Thirdly, only know how the flash program record my score will be not enough, because we need to know how it will let the server know what my score is. I can see there is a plaintext “score = 5”, however, there are still some hash value as well. I need to figure out what the hash values are if I want to do a successful modification. What I know is, currently, many website would like to use MD5 or SHA to keep their messages secret and secure. But of course, this website may not use these kinds of encryption methods, but I will also try to find some ways to know what they are.

**3.3.6 Begin Attack:**

According to the Attack Design, I think I need to figure out what is the value before the hash function because the server may use the hash value to judge if the data is authentic when it comes from the client side!

Consequently, my job next is to try to figure out what is the hash value.

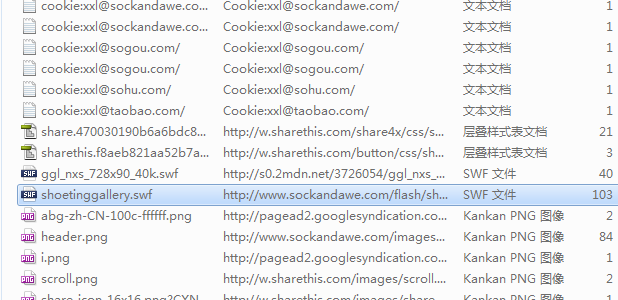
I tried to check the source code of the website first,



But there is not any “action” in the code but only the flash file. Consequently, I guess the action may be inside the flash file.

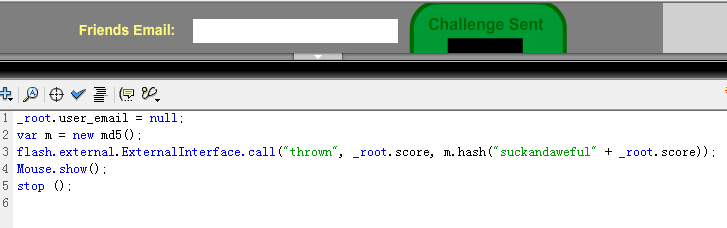
Thus, I will try to find something in the source code of the flash file.

But, first step is to download the flash file:



I checked my cookies and the temporary files. Then I found the “shoetinggallery.swf”.

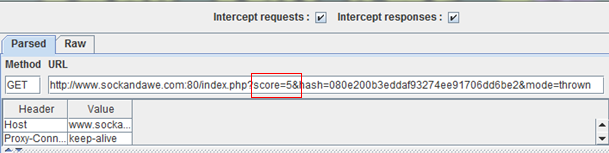
After finding it, I use the “Sothink SWF Decompiler” to decompile it so that I can see the source code.



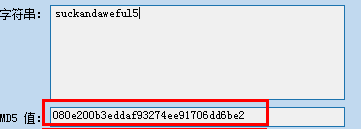
When I check the source code of the flash file, I found that at the “send” button, there is such piece of code “m.hash(“suckandaweful” + \_root.score)”.

That’s what I am looking for!

It seems like the hash value which shown in the intercepted package is established with hashing the string “suckandaweful” and the user’s score. To verify my guess, I use the MD5 calculator to compute the MD5 value of “suckandaweful5” (because my score is 5)



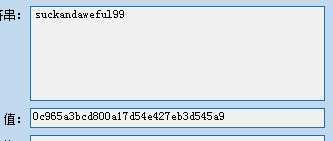
**1st hash value**



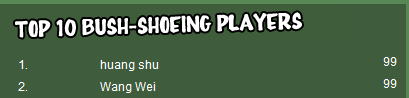
**2nd hash value**

From figures above, we can see that the 1st and the 2nd MD5 values are the same! Consequently, my guess is correct!

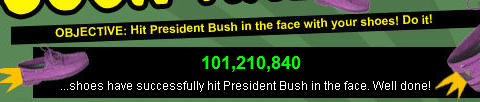
Then I can establish the fake hash value which can match my score and then send it to the server.



However, I found the “Top 10 list” didn’t change.



Then I think there may be something wrong with the server, because there is a total number at the top of the webpage that count all the hits.



However, after playing several times this game, the number didn’t change either. Consequently, I think the owner of the website may forbid the server receive more data from the client side.

**3.3.7 Conclusion:**

Though I didn’t put my name on the “Top 10 list” of the website, but I think it’s not my method’s problem but the server side. Because even I can fake the data correctly and send it to the server, if the server doesn’t receive it, I can do nothing. But it is still a very good example because though it uses the “score” as the plaintext to transfer to the server, it added something which is hash value. I believe, there must be a judge at the server side that check if the hash value can match the “suckandaweful” + “score”, actually, I think if they match, then the server accept the information and save them into the database, or the server will discard them. It is different from the first two test, because, first, we cannot find the action in the source code of the webpage but have to decompile the flash file to get the action; second, we cannot use WebScarab directly to modify the plaintext value, but we need to establish the hash value so that we can create the fake information successfully. Obviously, this one is more secure than the website in the second test; however, we still can break it and cheat the server. Consequently, there is no absolute method that can make the transmission secure but only methods can make it relative secure.

**4. Security Leverl**

From above, we can see that different websites have different methods of defending the MitB. And I can generalize them into several security levels according to their security methods.

**1st level: Transferring in plaintext and with “action” in HTML.**

This is the method which is used in my test 1 and some other primary and personal website. For an attacker, it’s easy to gain the information which is from the user. So it’s the lowest level of security method in data transferring.

**2nd level: Transferring in plaintext but not simply in HTML.**

This is another full of risk security level used in a lot of websites. Usually, some small flash game website will use this. Because there is nothing very important but the scores gained in the game. The bad guy may hardly find the plaintext directly in the HTML source code. However, he can check the source of the flash file and get the plaintext easily, too. Thus, it’s still full of risk.

**3rd level: Transferring in ciphertext. (i.e. MD5, SHA and etc)**

My third test’s security level is similar with this one. The bad guys cannot find any meaningful things in the source code of HTML. They may have some powerful tools like “WebScarab” so they can intercept the packages which are sent to server. However, they will find there is no meaningful information but only the values of MD5 (or SHA and etc). It’s hard for them to decrypt them, so they have to try to get some clues in the source code of flash files. For the test 3, unfortunately, they can find the useful information in the flash file’s source code.

**4th level: Transferring in ciphertext and also salt added**

I didn’t do test for such kind of website. But what I know is there are some website using ciphertext and salt to keep their data transmission secure. For example, some websites add salt to the username and password of the user and then encrypt them by using MD5 or SHA or something. And then transfer them to the server. Though there is a bad guy can intercept the information, he can hardly decrypt it because salt added. Of course we don’t need to concern how the server can decrypt it, because in the database, the username and the password are in salted - MD5 (SHA) types.

**5. Additional Test**

**5.1 Introduction**

Now we know how the MitB works. But in the first several tests, there are some websites that people may never want to attack because the attackers cannot get benefit. The real attacks happen only when the attackers can get benefit. For example, when we try to get someone’s username and password of a website, we would like to attack because we can get benefit. Consequently, I did this test to prove MitB can be used to steal people’s account information when they login.

**5.2 Additional Test 1**

**5.2.1 Target:**

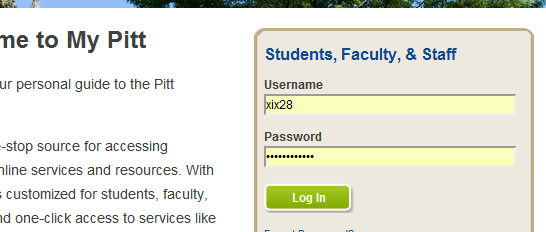
www.my.pitt.edu

**5.2.2 Attack Goal:**

Get the user’s username and password

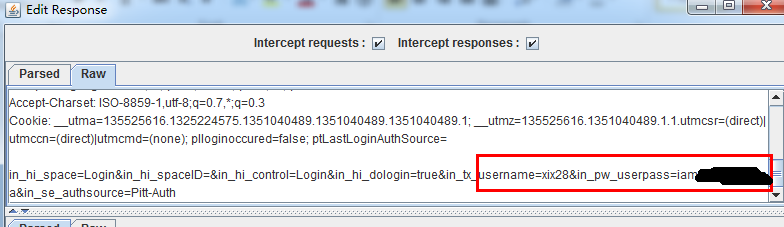
**5.2.3 Attack Design:**

When I click the “login” button, run WebScarab and try to intercept the information the browser submit to the server.



**5.2.4 Attack Begin:**

When I click the “Log In” button, this window pops out. After checking it, I found the username and my password is transferred in plaintext! I can read my password directly! It’s really a bad thing. Not only no salt, but also there is no “HASH” convert in this transmission. No more steps, I can read my password directly by only using a simple tool.



**5.2.5 Conclusion:**

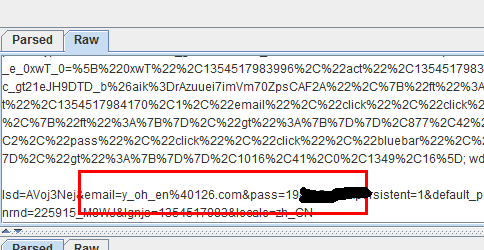
Password transferred in plaintext is really not a good thing. Actually, it can do better with using hash function or even salted – hash function. If an attacker wants to steal someone’s password, he just needs to make “WebScarab” run in the victim’s PC by using “Trojan” or something. And then the attacker can intercept the information as what I got so that he can easily know the password of the victim. As mentioned, the website can add some “salt” byte at the end of the password and then hash them and send the hash value to the server so that the attacker can hardly get the password.

**5.3 Additional Test 2**

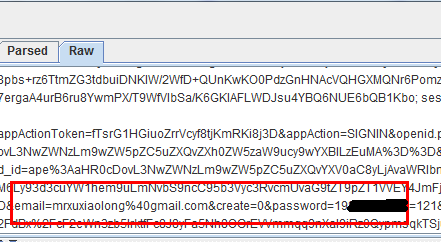
**5.3.1 Introduction**

Before I did the first additional test, I never think I can get the password in plaintext because it’s really full of dangerous. Thus, I thought maybe the most famous website like Facebook.com, Amazon.com and Google.com could be better. Consequently, I want to try this website.

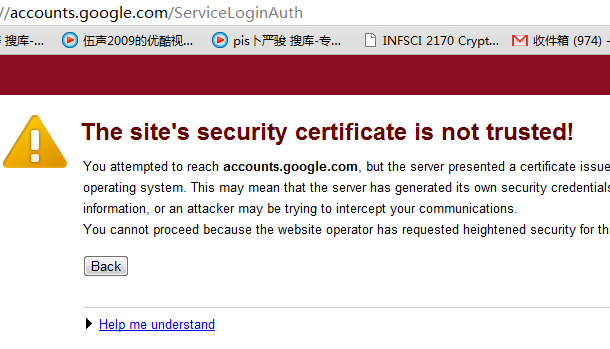
**5.3.2 Facebook.com**



**5.3.3 Amazon.com**



**5.3.4 Google.com**



**5.3.5 Result**

From the figures above, we can get the result:

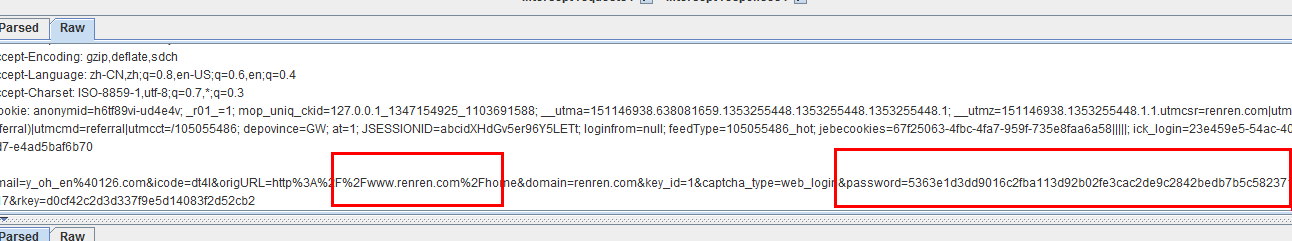
|  |  |  |
| --- | --- | --- |
|  | Encrypted? | Method? |
| My.pitt.edu | No | Plaintext |
| Facebook.com | No | Plaintext |
| Amazon.com | No | Plaintext |
| Google.com | Unknown | Cannot be intercepted by WS |

**5.4 Additional Test 3**

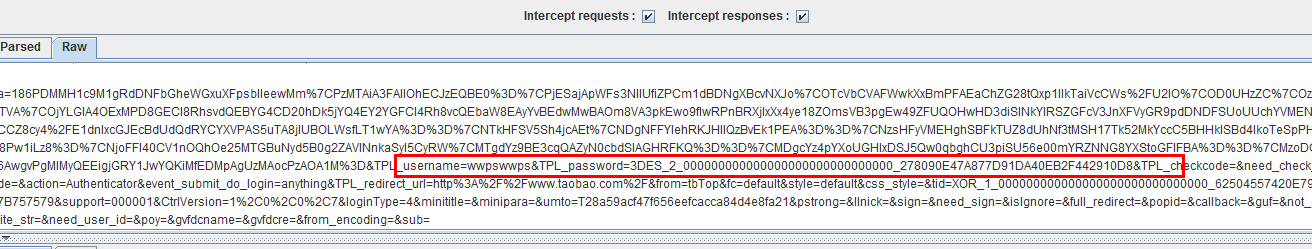
**5.4.1 Introduction**

In this part, I will show some Chinese websites, they may not as good as Facebook, Amazon or Google. However, they did do some efforts in hiding users’ password when transmission happens.

**5.4.2 Renren.com**



**5.4.3 Taobao.com**



**5.4.4 Weibo.com**



**5.4.5 Result**

|  |  |  |
| --- | --- | --- |
|  | Encrypted? | Method? |
| Renren.com | Yes | Unknown |
| Taobao.com | Yes | 3 – DES Encryption |
| Weibo.com | Yes | RSA Encryption |

**5.4.6 Reasons**

I don’t think these three websites are better than Facebook, Amazon and Google, because they are not creative but only learn or, in another word, simulate Facebook, Amazon and Twitter. However, obviously they do better at the point of transmission of users’ passwords. That is because there is a bad event which is called “CSDN leak” in recent years. In this event, more than 6 million users’ account information was leaked and everybody shocked that they have to update their passwords. Then, more and more websites realized that they have to do something to make the users’ information more secure so that they can keep their users. Because they all know users are the life of the website. That’s why they all did some efforts on hiding the users’ passwords in when data is transferred from the client to the server.

**6. Analysis**

According to the part 5 – additional tests, we can see that some of the websites transfer users’ passwords in plaintext and some are in ciphertext. We can hardly judge which one is better because there is not only one thing for us to consider. For example, users’ experience maybe as important as keeping users’ personal information secures. However, in this project, the key point is the security; consequently, users’ information security will be put at the first place.

I have to say that the passwords are transferred in plaintext is very dangerous. Because it may make the passwords leak very easily. A bad guy needs no more steps but only intercept the packages which are sent from browser to the server, he can read the password directly! And this password can be used to login to the website as a normal user! What’s worse, he can give the password to someone else who has no ability to launch an attack but want to get others’ password. Is it the worst situation? Of course not! Because the bad guy may even guess the victim’s other passwords! Let’s come back to the real world, it’s an internet world. We have passwords for our debit card, Facebook, online games, MSN and almost everything. However, how many of us would like to set different passwords for each of them? Most of us won’t do that because we will have to remember more than 100 passwords. It’s too hard for us. Consequently, we can always see that one person Bob may use the same password A for his online game accounts, password B for his Facebook and Twitter accounts, password C for his mypitt and PNC accounts. Obviously, problem comes. If a bad guy Oscar knows Bob’s password for mypitt by intercepting the packages which is from Bob’s browser when he login, Oscar will know Bob’s PNC account’s password directly!

However, what if we do something on the password to make it not show in plaintext but in ciphertext when transmission happens? Let’s see an example, we have a website: www.abc.com, Bob has an account. And when Bob login, Oscar intercept the package, and part of the package which is intercepted is “usr = BOBpwd = a906449d5769fa7361d7ecc6aa3f6d28”. Obviously, Oscar cannot read Bob’s password directly. So what Oscar can know it only “use the username ‘BOB’ and the password string ‘a906449d5769fa7361d7ecc6aa3f6d28’ can login successfully”. Of course, Oscar can still login the website with the intercepted information. However, he can do nothing more.

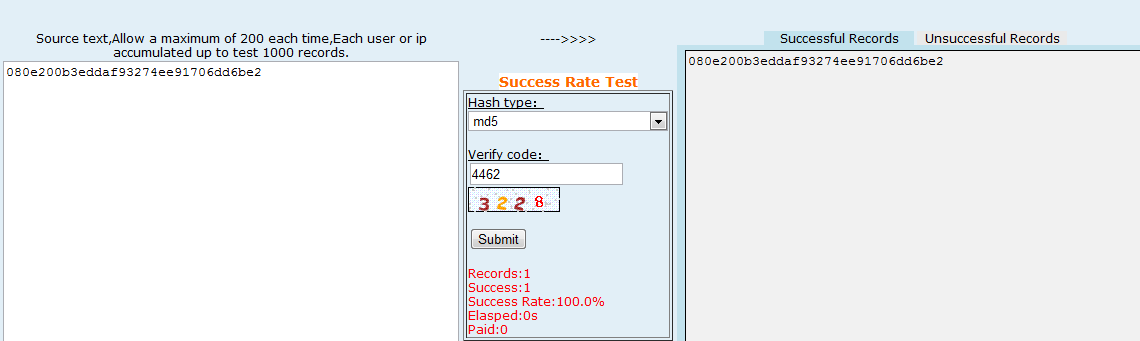
Now let’s suppose the password is shown in plaintext and then see what Oscar can do. The unfortunate guy is still Bob, but at this time the part of the package which is intercepted by Oscar is “usr = BOBpwd = 123abc”. Of course, Oscar can use them to login without using WebScarab. It may save his 2 minutes. And also, he can share this password and username with his friend Ostruck who has never heard about the WebScarab so that Ostruck can login with Bob’s account anytime and anywhere. Moreover, poor Bob uses the same password on his PNC bank card and online apple account. Then Oscar steals all of Bob’s money and deletes all his contacts from apple cloud…

Of course, poor Bob is not a real guy and we can see his story as a joke. But it is possible and completely could happen. We know there is no method which is absolutely secure; like the MD5 value, we still didn’t find a way to crack it, but we can rely on a very huge database to find the value out; however, even we know that we are still on our way of trying to figure out more secure one. Consequently, even if we cannot prevent password from leaking completely, we still have some ways to make Oscar spend more time, money and energy on attacking. And here, it’s encryption but not plaintext.

**7. How could we defend them**

Of course, there are still many other ways to help a website keep the data transmission secure, but there is no shield can ever be broken. New defense methods appear, new attack methods will be come up.

For example, there is a website can decrypt the MD5, SHA or some other ciphertext. I’ve tried the MD5 value in test 3, and this website can decrypt it in only one second!



Consequently, what we need to consider is not to avoid attacks but how to make the cost of attacks most.

Come back to this project, there are some tests for websites which are in different security levels. However, we can hardly judge which website is better and which one is worse because they have different user groups. For the website in test 1 and test 2, how many people would like to spend 1 day, a week or even a month to study how to attack them or cheat their servers? I think the answer is “few”. Because they are just flash game websites, people who open the website just want to play games! They have no need or even no abilities to attack them! Though they can attack successfully, they get no benefit. Thus, the security level of these websites is fine. However, what if it is a Bank’s online system? Then the answer should be different. There must be a lot of potential attackers that want to spend 1 year, 1 decade or even whole life to attack it. Because if they are successful, they get money, they get benefit! Consequently, for the question “how could we defend”, what we should consider first is “which security level we should use”.

And then, we need to consider how we can defend the attacks. We just need to talk about the third test because for the first one, it can use the method in test 2 to make the website more secure; and for the test 2, it can use the method in test 3 to make it more secure. All the tests in this project are local, but the real attack will not like this. In a real attack, nobody wants you to run a “WebScarab” in his machine when he surfs on the internet. So what the attacker needs is “Trojan” which can be run in the backstage of the user. Of course the user may not know there is such a “Trojan”. But the “Trojan” can work like the “WebScarab” so that the bad guy can launch the “MitB” attack. To prevent this kind of attack from happening, we can use the firewall which can make the browser run independently. That is, we use a virtual firewall in our browser, when we open the browser, the firewall runs, too. And this firewall can isolate the browser’s activities with other applications’ activities so that the “Trojan” cannot get the data from local browser. And the attack will be failed.

**Reference**

1. https://www.owasp.org/index.php/Man-in-the-middle\_attack

2. http://en.wikipedia.org/wiki/Man-in-the-browser

3. http://en.wikipedia.org/wiki/WebScarab

4. The Ghost in the Browser Analysis of Web-based Malware, Niels Provos, Dean McNamee, Panayiotis Mavrommatis, Ke Wang and Nagendra Modadugu Google, Inc.

5. Making Sense of Man-in-the-browser attacks – Threat Analysis and Mitigation for Financial Institutions

6. Review of Browser Extensions, a Man-in-the-Browser Phishing Techniques Targeting Bank Customers, Nattakant Utakrit

7. Hardening Web Browsers Against Man-in-the-Middle and Eavesdropping Attacks, Haidong Xia and Jose Carlos Brustoloni