2011 International Conference on Network and Electronics Engineering
IPCSIT vol.11 (2011) © (2011) IACSIT Press, Singapore

Web Security Improving by using Dynamic Password Authentication

Detchasit Pansa, Thawatchai Chomsiri

Faculty of Informatics, Mahasarakham University, Thailand
detchasit@msu.ac.th, thawatchai@msu.ac.th

Abstract. To detect a password while an internet user is logging in the website has been viewed as one of the significant problems. At the present time, although the HTTPS is broadly used, there are still some kinds of tools and equipments, e.g. Cain and BackTrack, which assist the hackers to detect and decode the HTTPS. At this point, this research aims to design a New Web Authentication in which it will switch the password that browser is sent to the server to be Dynamic Password. Through this process, the passwords that the hackers detected for each time will not be the same, so that the hackers are not able to validate those passwords. The researchers designed the Password Authentication Model as well as analyzed its functional procedure. It was indicated that the Model contains a complete security system. Furthermore, the researchers established the website in order to prove that the web programmers are able to practically apply the model into the authentic works. Indeed, the researchers also assessed the speed capability of the model and found that it could easily be implemented and fully secured. Meanwhile, the Delay and CPU Load values on the server were hardly increased.

Keywords: Web Security, Authentication, Password, Hash, Login

1. Introduction

At the present lives, the use of services via websites has become popular and necessary for people's daily life. It is undeniable that people need to use Web Mail to order some products via the websites as well as to register as members of various online-service websites. Hence, the online security has also become very essential. Nowadays, it is very easy to detect the internet user's password, especially for the websites that do not use the HTTPS in sending the password. The hackers on the same LAN of the victims are able to detect the password via the use of ARP spoof [1] and detect the packet by using Ethereal, WireShark, or other related tools. Even though the majority of the websites uses the HTTPS (during the process of authentication), the hackers are still able to detect and decode the passwords via the use of various tools e.g. Cain [2] and BackTrack [3] to disguise as "MITM: Man In The Middle" and distract the browsers from the websites. After that, it will issue a fake certificate (fake public key) for the browsers in order to detect and decode the user's password. After obviously seeing the problem, the researchers initiated with a conception that the password sent by the browsers to the server during the process of authentication should not be only a plain text or the encoding that could be re-encoded. In fact, the password should be one-way encoded in which it is unable to be re-encoded as well as it should be dynamic, which will change after a single authentication. Therefore, the researchers decided to create the New Authentication Model on the basis of the Dynamic Password. The model must not contain any complication and can be simply implemented, with full security and without slowing down the functional system.

2. Related works

Although the use of authentication system via Hashing Function has previously been done e.g. by Lamport [4] and CINON [5], it indicated that Lamport still encountered some problem in which it required the password resetting. Later, Shimizu [5] solved Lamport's problem by proposing CINON without password resetting, together with PERM protocol [6] in finding the solution for the random number

memorizing problem of CINON protocol. Nevertheless, all of [4], [5] and [6] had no direct focus on the solution for the problematic web-authentication, and still contained the complicated procedure which was difficult to be practically implemented. Lei et al. [7], [8] proposed solution toward the password detection and stealing via virtual password system with the application of the randomized linear generation function. Besides, there was an existence of other related researches with the application of virtual password system e.g. [9] used "Secret Little Functions" and "Codebooks" for hiding the passwords from the hackers. These processes were similar to [10] and "Codebook-Based Solution" proposed in [11]. N.M. Haller [12] suggested the S/Key one-time password system that used a number of hashing functions, whereas the server will verify the password by using hashing function with an equal number, before making a comparison. The mentioned researches differed from the Model proposed in this research. In this research, the researchers merely made a single use of hashing with time-variable involved.

3. Design

In this model, Dynamic password was originated by appending the time string to the password string, before making a single hashing. The user would keep using the same password (until the password was changed by the user).

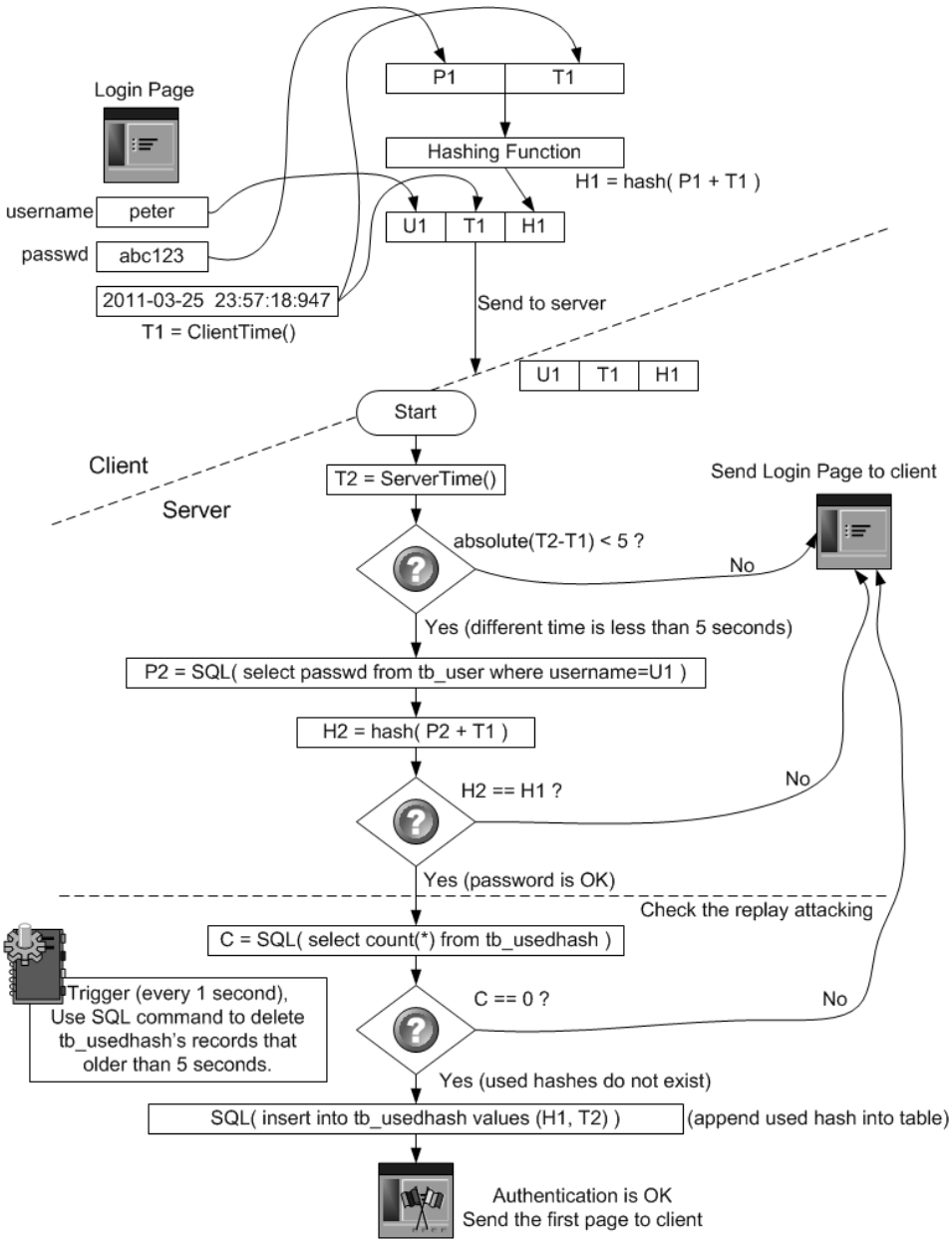

Fig. 1: Design of Web Authentication.

In the Figure 1, the hashing of the password string + the time string (P1+T1) were H1. When compared with an original-version of the authentication, it was apparent that the login form would send password string into the password's field; meanwhile, the value in the user's field was the Username. The two mentioned fields were similar to the original authentication; the only difference was that the value in the password's field was a variance. From the figure, there was an additional field exist; it was the time field (T1). It assisted the server in verifying the time duration of H1 creation (In practical, the browser could send T1 to the server via the Hidden Field). In this model, the time setting of the client and the server must be well-synchronized, by using NTP: Network Time Protocol, or by creating the program on the server in order to postpone the time. If using the latter process, the client must initially inform its time; for example, sending the client's time along with the login-form request in the first place. In fact, in this research, the researchers intended to create the prototype program for the first process which was the time synchronization between the client and the server by using NTP. When U1, T1, and H1 were sent to the server, the server would verify T1. If T1 (Client's Time) differed from the server's time more than 5 seconds, it could be concluded that U1+T1+H1 were captured and undertook the Replay Attack. In that case, the browser would be redirected to the login page. If T1 and the server's time differed not over 5 seconds, the server would verify the user's password via SQL command order to verify the user's password in database (DB) under the username's condition that matched with U1. Through this process, the password would be P2. Later, the researchers would not make a direct comparison between P2 and H1 since H1 was not the password, but only a hashing value. Thus, the researchers needed to make a comparison through the same process by appending T1 to P2 and did hashing first. If the received hashing value resulted with same outcome, it mean that the password sent by the user was confirmedly correct.

Nonetheless, if the hackers created a new program that could launch the Replay Attack within 5 second, the attack would be blocked at the final phase by an verification whether H1 was already used or not. During this phase, the successfully authenticated H1 would be recorded and remained in DB for at least 5 seconds (it would be deleted after that). If there was the use of H1 previously contained in DB (already used), it was concluded that this was the Replay Attack and the request would be redirected to the login form.

4. Analysis

The model was analyzed for 3 aspects including security, speed, and ease of implementation.

4.1. Security

It was known that the main objectives of the model was to make the websites more secured by using Dynamic Password, in which it would not be detected. The hackers would not be able to use the password since it was replaced by H1. Indeed, H1 was Hash (P1+T1), which mean P1 was the password keyed in by the user, whereas T1 was the client's time. The security of this model was that the password field (H1) could be changeable because each time the user logged in, the client's time would change so that the Hash (P1+T1) also changed along. In case that the client was sending U1+T1+H1 to the server and the hacker wanted to detect U1+T1+H1, he would make ARP Spoof on the LAN, as well as run the Ethereal or WireShark to detect the data. After that, it would take several minutes to receive the data, stopped detecting, and started searching for the password. This process would take quite a long period to be completed. In general, it would take approximately 1-5 minutes (60-300 seconds). It surely would not be completed within 10 seconds (by manual processing). After the hacker received U1+T1+H1 and undertook the Manual Replay Attack, it still could not attack the system since the model provided the security system against the attack by verifying the difference between the server's time and T1 (the client's time). If the time differed more than 5 seconds, it was admitted to be the replay attack.

However, it was possible that the hackers created some special program to launch an automatic replay attack with speedier functioning and took less than 5 seconds or possibly less than1 second. For that reason, the researchers developed the model at the final phase with the security system against this kind of attacks. Actually, the model was designed to make a record of the used H1, especially for the one that passed the authentication. If the hackers launched an automatic replay attack and sent the used H1 to the server (or logging in), it would be possible for the server to run verification because it previously had the record of the

used H1. On the other hands, if the hackers were able to sent the detected H1 before the authenticating user did, it would be called "a Perfect Man in the Middle" which was harder to prevent (in future work, the researchers still are able to create the further security system against the Perfect Man In The Middle).

4.2. Performance

The increased overhead was to get the client's time and the hashing on the client. Getting the client's time was determined as a simple task, so it was not the main problem. For hashing, when making a comparison with the RSA encoding which required a heavy processing, it could be stated that hashing was a simple task as well as it was not the problem of the performance.

On the server, the time comparison was a simple functioning whereas the password retrieval in DB was a regular task already contained in an original authentication. Consequently, the difficult task of this model might definitely be the verification of the used hash. On the surface, there was a number of the used hash records collected during the 5-second period when a huge amount of the users existed. For example, with 1,000,000 user's account, the amount of the used hash would be increased to 1,000,000 records or more. In fact, it could be roughly estimated that there was 1,000,000, user's accounts on the system and each user approximately logged in 2 times a day i.e. in the morning and the afternoon (e.g. via Gmail or hotmail). If the correct password of a user's account was keyed in, the used hash of that user would stay in DB for 5 seconds -- 5 seconds x 2 times = 10 seconds (for one day). Therefore, the possibility that a user's used hash would remain in DB for one day would be Prop = 10/ (60*60*24) = 10/86400 = 0.000011574. This mean that if there were 1,000,000 account users logged in randomly (with different timing), the average number of the records in the used hash table would result as 11.574 or around 12 records. In case that the SQL command was applied to verify whether there were the repeated H1 found in 12 records in DB was considered as a simple task of the 1,000,000 user's accounts.

4.3. Ease of Implementation

The function used to get time and hash was executed in the Java Script on the client, as well as in PHP, ASP, and other computing languages on the server. The time synchronization between the client and the server could be done via NTP, though the NTP contained a delay value (according to an average of the researchers' experiments = 100-300 ms) and it was impossible to take more than 1 second. As a consequence, to compensate the overlapped time of the implementation, the researchers should reserve the time by keeping the used hash in DB for one more second, so it would totally be 6 seconds for the security. On the contrary, another choice of the implementation was not to use NTP by sending the client's time in the first place that it was redirected to the login page, in order that the server could recognize an offset value and automatically compensate the overlapping between the client's and server's time.

5. Implementation and Performance Evaluation

Table 1: The Response Time and CPU values before implementing the model

Request / sec	Response Time (ms)	CPU load (%)
10	671	5.73
20	809	12.75
30	973	34.27
40	1,147	76.81
50	1,369	100.00
60	1,568	100.00
70	2,106	100.00
80	2,423	100.00
90	2,781	100.00
100	3,359	100.00

The researchers undertook the implementation by making use of PHP5, Java Script, CentOS 5.3, Intel Core 2 Duo CPU (1.8GHz), 2GB RAM, 1Gbps LAN as well as testing to assess the response time value (counting time on the client) and CPU load value (assessed on the server) for the load with a concurrent connection of 10, 20, 30, …, 100 request/sec respectively. The test was carried out for 30 times and the averages were presented on the table. The testing results were as shown on Table 1 and Table 2 respectively.

Table 2: The Response Time and CPU load values after implementing the model

Request / sec	Response Time (ms)	CPU load (%)
10	831	7.26
20	1,180	19.87
30	1,302	47.07
40	1,721	100.00
50	2,578	100.00
60	3,184	100.00
70	4,211	100.00
80	5,013	100.00
90	6,478	100.00
100	8,312	100.00

6. Conclusion

In this research, the researchers proposed the Model of Dynamic Password Authentication for the websites. It was suggested that in each time the browser is sending the password to the server, the hash (password + time) must be sent along with to replace the authentic password. The model contains the security system against the Replay Attack as well. In details, the researchers proposed the model, the functional procedure, as well as created the prototype website to prove that this model can be practically used with high accuracy. The researchers also assessed the model's effectiveness by making a comparison with the original authentication and found that the proposed model originated less than 3% of the response times (delay) and required less than 1% of the CPU capability on the server. In the future work, the researchers assure to create the model with the security system against the Perfect Man In The Middle e.g. in case of the hackers created a special software for launching an automatic replay attack, and able to send the detected H1 to the server before the authenticating user.

7. References

[1] S. Whalen. *An Introduction to ARP Spoofing*. 2001., http://packetstorm.securify.com/papers/protocols/intro_to_arp_spoofing.pdf

[2] *Cain & Abel*., http://www.oxid.it/cain.html

[3] *BackTrack*., http://www.backtrack-linux.org

[4] L. Lamport. Password authentication with inse-cure communication. *Communications of the ACM*. 1981, 24 (1): 770–772.

[5] A. Shimizu. A dynamic password authentication method by one-way function. *IEICE Trans. Inform. Syst*. 1990, 73 (1): 630–636.

[6] A. Shimizu, T. Horioka, and H. Inagaki. A password authentication method for contents communication on the Internet. *IEICE Transactions on Communications*. 1998: 81 (2): 1666–1763.

[7] M. Lei, Y. Xiao, S. V. Vrbsky, C.-C. Li, and L. Liu, A virtual password scheme to protect passwords. *In Proceedings of IEEE International Conference on Communications (ICC'2008)*. Beijing. 2008, pp. 1536–1540.

[8] M. Lei, Y. Xiao, S. V. Vrbsky, and C.-C. Li. Virtual password using random linear functions for on-line services, ATM machines, and pervasive computing. *Computer Communications*. 2008, 31 (18): 4367–4375.

[9] Y. Xiao, C.-C. Li, M. Lei, and S. V. Vrbsky. Secret little functions and codebook for protecting users from password theft. *In Proceedings of IEEE International Conference on Communications (ICC'2008)*. Beijing. 2008, pp. 1525–1529.

[10] J. A. Haskett. Pass-algorithms: A user validation scheme based on knowledge of secret algorithms. *Communications of the ACM*. 1984, 27 (8): 777–781.

[11] M. Szydlowski, C. Kruegel, and E. Kirda. Secure input for web applications. *In Proceedings of 23rd Annual Computer Security Applications Conference (ACSAC'2007)*. Florida. 2007, pp. 375–384.

[12] N.M. Haller. The S/KEY one-time password system. *In: Symposium on Network and Distributed Systems Security*. San Diego. 1994, pp. 151–157.