PASSWORD ATTACK RESISTANT USING PGRP

S.Vasanthakumari
PG Scholar
Knowledge Institute of Technology
Salem
vasanthuvijay@gmail.com

G.KeerthanaPriyadharshini
PG Scholar
Knowledge Institute of Technology
Salem
gkeerthi0406@gmail.com

S.Karthikeyan
PG Scholar
Knowledge Institute of Technology
Salem
kiotkarthik@gmail.com

M.Sujitha
PG Scholar
Knowledge Institute of Technology
Salem. msujinavi@gmail.com

T.Karthikeyan
Assistant Professor, Dept. of CSE
Knowledge Institute of Technology
Salem
tkcse@kiot.ac.in

M.Sharmili
Assistant Professor Dept. of CSE
Knowledge Institute of Technology
Salem
mshcse@kiot.ac.in

Abstract: Remote login services are used in web applications. Web interface and secure shall login (SSH) methods are used for the remote login process. Remote login services are attacked with Brute force and dictionary attacks. Password guessing attacks are initiated by the Botnets. Automated Turing Tests (ATTs) is conducted to identify automated malicious login attempts. Pinkas and Sander (PS) and van Oorschot and Stubblebine proposals (VS) are used to limit the accessible password guessing attacks based on ATTs. The PS proposal reduces the number of ATTs sent to appropriate users. The VS proposal reduces the security overhead with a significant cost to usability. Security, usability and user interface factors are considered in the remote login process. Login attacks are controlled by Password Guessing Resistant Protocol (PGRP). PGRP limits the total number of login attempts from unknown remote hosts. PGRP enforces ATTs after a few failed login attempts are made from unknown machines. PGRP allows a high number of failed attempts from known machines without answering any ATTs. Known machines are systems with a successful login have occurred within a fixed period of time. White-listed IP address and client cookie are used to identify the known machines. PGRP supports both graphical user interfaces (browser-based logins) and character-based interfaces (SSH logins). User name and IP address are used to detect appropriate users. Cookie thefts are handled with enhanced PGRP protocol. Black lists are used to manage the attacker addresses under login verification. Compromised machine attacks are handled with the user name and IP address associations. Concurrent login verification is applied with session details. The password communications are secured with RSA algorithm.

Keywords: RSA algorithm, PGRP protocol, secure shall login (SSH)

1. INTRODUCTION

Accessible guessing attacks on password-based systems are inevitable and commonly observed against web applications and SSH logins. In a recent report, SANS recognized password guessing attacks on websites as a top cyber security risk. As an example of SSH password-guessing attacks, one experimental Linux honeypot setup has been reported [5] to suffer on average 2,805 SSH malicious login attempts per computer per day. Interestingly, SSH servers that disallow standard password authentication may also suffer guessing attacks, e.g., through the exploitation of a lesser known/used SSH server configuration called keyboard interactive authentication [6]. However, accessible attacks have some inherent disadvantages compared to offline attacks: attacking machines must engage in an interactive protocol, thus allowing easier detection; and in most cases, attackers can try only restricted number of guesses from a single machine before being locked out, delayed, or challenged to answer Automated Turing Tests [1]. Consequently, attackers often must employ a large number of machines to avoid detection or lock-out. On the other hand, as users generally choose common and relatively feeble passwords and attackers currently control large botnets, accessible attacks are much easier than before.

One active resistance against automated accessible password guessing attacks is to restrict the number of miscarried trials without ATTs to a very small number, limiting automated programs as used by attackers to three free password guesses for a targeted account, even if different machines from a botnet are used. However, this inconveniences the appropriate user who then must answer an ATT on the next login attempt.

Several other techniques are deployed in practice, including: allowing login attempts without ATTs from a different machine, when a certain number of failed attempts occur from a given machine; allowing more attempts without ATTs after a time-out period; and time-limited account locking. Many existing techniques and proposals involve ATTs, with the underlying assumption that these challenges are sufficiently difficult for bots and easy for most people. However, users increasingly dislike ATTs as these are perceived as an extra step; see Yan and Ahmad [8] for usability issues related to commonly used CAPTCHAs. Due to successful attacks which break ATTs without human solvers (e.g.,[2]) ATTs perceived to be more difficult for bots are being deployed. As a consequence of this arms-race, present-day ATTs are becoming increasingly difficult for human users [3], fueling a growing tension between security and usability of ATTs. Therefore, we focus on reducing user annoyance by challenging users with fewer ATTs, while at the same time subjecting bot logins to more ATTs, to drive up the economic cost to attackers [11].

Two well-known proposals for limiting accessible guessing attacks using ATTs are Pinkas and Sander and van Oorschot and Stubblebine. The PS proposal reduces the number of ATTs sent to appropriate users, but at some meaningful loss of security; for example, in an example setup PS allows attackers to eliminate 95 percent of the password space without answering any ATTs. The VS proposal reduces this but at a significant cost to usability; for example, VS may require all users to answer ATTs in certain circumstances. The proposal in the present paper, called Password Guessing Resistant Protocol (PGRP), significantly improves the security-usability trade-off, and can be more generally deployed beyond browser-based authentication.

PGRP builds on these two previous proposals. In particular, to limit invaders in control of a large botnet, PGRP enforces ATTs after a few failed login attempts are made from unknown machines. On the other hand, PGRP allows a high number of miscarried attempts from known machines without answering any ATTs. We define known machines as those from which a successful login has occurred within a fixed period of time. These are identified by their IP addresses saved on the login server as a white list, or cookies stored on client machines. A white-listed IP address and/or client cookie expires after a certain time.

PGRP accommodates both graphical user interfaces and character-based interfaces, while the previous protocols deal exclusively with the former, requiring the use of browser cookies. PGRP uses either cookies or IP addresses, or both for tracking appropriate users. Tracking users through their IP addresses also allows PGRP to increase the number of ATTs for password guessing attacks and meanwhile to decrease the number of ATTs for appropriate login attempts. Although NATs and web proxies may reduce the utility of IP address information, in practice, the use of IP addresses for client identification appears feasible. In recent years, the trend of logging in to accessible accounts through multiple personal devices is growing. When used from a home environment, these devices often share a single public IP address which makes IP-based history tracking more user friendly than cookies. For example, cookies must be stored, albeit transparently to the user, in all devices used for login.

2. PROBLEM STATEMENT

1. STRICT BUT USER-FRIENDLY ATT-BASED SCHEME. The proposed PGRP scheme is more restrictive against attackers than commonly used countermeasures and two earlier proposals. At the same time, PGRP requires answering fewer ATTs for all appropriate users, including those who occasionally require multiple attempts to recall a password.
2. FIRST REPORTED EMPIRICAL ANALYSIS OF ATTBASED SCHEMES. We compare PGRP's performance and usability to previous such schemes, using two data sets from a university environment.
3. APPLICABILITY TOWEB AND TEXT LOGINS. PGRP is not limited to web only login, as it uses IP address and/or other methods to identify a remote machine in addition to optionally using cookies. By using text-based ATTs, SSH login can be adapted to use PGRP.

3. RELATED WORK

Although accessible password guessing attacks have been known since the early days of the Internet, there is little academic literature on prevention techniques. Account locking is a customary mechanism to prevent an adversary from attempting multiple passwords for a particular username. Although locking is generally temporary, the adversary can mount a DoS attack by making enough failed login attempts to lock a particular account. Delaying server response after receiving user credentials, whether the password is correct or incorrect, prevents the adversary from attempting a large number of passwords in a reasonable amount of time for a particular username. However, for adversaries with access to a large number of machines, this mechanism is ineffective. Similarly, prevention techniques that rely on requesting the user machine to perform extra nontrivial computation prior to replying to the entered credentials are not effective with such adversaries.

ATT challenges are used in some login protocols to prevent automated programs from brute force and dictionary attacks. Pinkas and Sander [4] presented a login protocol based on ATTs to protect against accessible password guessing attacks. It reduces the number of ATTs that appropriate users must correctly answer so that a user with a valid browser cookie will rarely be prompted to answer an ATT. A deterministic function of the entered user credentials is used to decide whether to ask the user an ATT. To improve the security of the PS protocol, van Oorschot and Stubblebine [10] suggested a modified protocol in which ATTs are always required once the number of failed login attempts for a particular username exceeds a threshold; other modifications were introduced to reduce the effects of cookie theft.

For both PS and VS protocols, the decision function AskATT()) requires careful design. He and Han [9] pointed out that a poor design of this function may make the login protocol vulnerable to attacks such as the "known function attack" and "changed password attack". The authors proposed a secure nondeterministic keyed hash function as Ask-ATT() so that each username is associated with one key that should be changed whenever the corresponding password is changed. The proposed function requires extra server-side storage per username and at least one cryptographic hash operation per login attempt.

4. PASSWORD GUESSING RESISTANT PROTOCOL

We present the PGRP protocol, including the goals and design choices.

4.1. Objectives of PGRP

Our objectives for PGRP include the following:

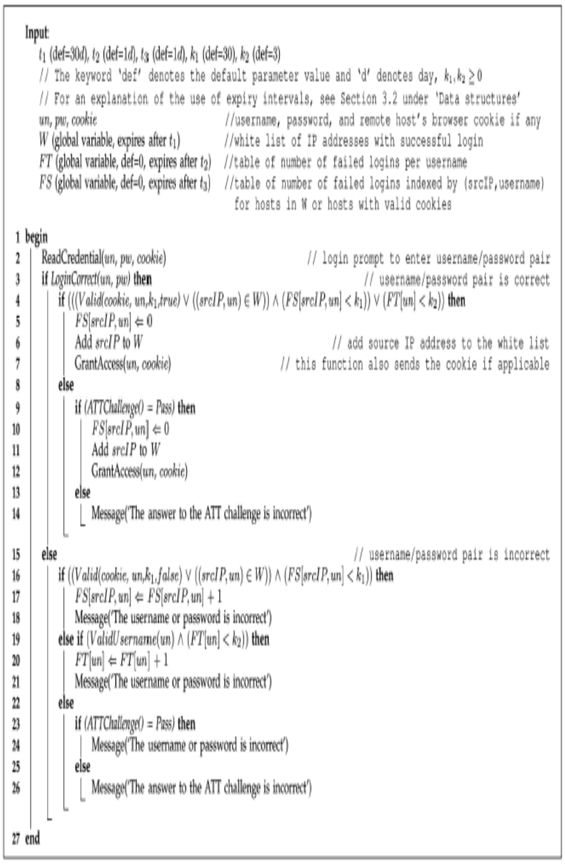
1. The login protocol should make brute force and dictionary attacks ineffective even for adversaries with access to large botnets.
2. The protocol should not have any significant impact on usability. For example: for appropriate users, any additional steps besides entering login credentials should be minimal. Increasing the security of the protocol must have minimal effect in decreasing the login usability.
3. The protocol should be easy to deploy and scalable, requiring minimum computational resources in terms of memory, processing time, and disk space

4.2. PGRP Outline

The general idea behind PGRP (see Fig. 1) is that except for the following two cases, all remote hosts must correctly answer an ATT challenge prior to being informed whether access is granted or the login attempt is unsuccessful: 1) when the number of failed login attempts for a given username is very small, and 2) when the remote host has successfully logged in using the same username in the past.

In contrast to previous protocols, PGRP uses either IP addresses, cookies, or both to identify machines from which users have been successfully authenticated. The decision to require an ATT challenge upon receiving incorrect credentials is based on the received cookie and/or the remote host's IP address. In addition, if the number of failed login attempts for a specific username is below a threshold, the user is not required to answer an ATT challenge even if the login attempt is from a new machine for the first time.

Fig. 1. PGRP: Password Guessing Resistant Protocol



```
Input:
  $t_1$ (def=30d), $t_2$ (def=1d), $t_3$ (def=1d), $k_1$ (def=30), $k_2$ (def=3)
  // The keyword 'def' denotes the default parameter value and 'd' denotes day, $k_1, k_2 \geq 0$
  // For an explanation of the use of expiry intervals, see Section 3.2 under 'Data structures'
  un, pw, cookie                              //username, password, and remote host's browser cookie if any
  W (global variable, expires after $t_1$)        //white list of IP addresses with successful login
  FT (global variable, def=0, expires after $t_2$) //table of number of failed logins per username
  FS (global variable, def=0, expires after $t_3$) //table of number of failed logins indexed by (srcIP,username)
                                                for hosts in W or hosts with valid cookies

1  begin
2    ReadCredential(un, pw, cookie)                 // login prompt to enter username/password pair
3    if LoginCorrect(un, pw) then                   // username/password pair is correct
4      if $(((Valid(cookie, un, k_1, true) \vee ((srcIP, un) \in W)) \wedge (FS[srcIP, un] < k_1)) \vee (FT[un] < k_2))$ then
5        $FS[srcIP, un] \Leftarrow 0$
6        Add $srcIP$ to $W$                            // add source IP address to the white list
7        GrantAccess(un, cookie)                    // this function also sends the cookie if applicable
8      else
9        if (ATTChallenge() = Pass) then
10         $FS[srcIP, un] \Leftarrow 0$
11         Add $srcIP$ to $W$
12         GrantAccess(un, cookie)
13       else
14         Message('The answer to the ATT challenge is incorrect')

15   else                                           // username/password pair is incorrect
16     if $((Valid(cookie, un, k_1, false) \vee ((srcIP, un) \in W)) \wedge (FS[srcIP, un] < k_1))$ then
17       $FS[srcIP, un] \Leftarrow FS[srcIP, un] + 1$
18       Message('The username or password is incorrect')
19     else if $(ValidUsername(un) \wedge (FT[un] < k_2))$ then
20       $FT[un] \Leftarrow FT[un] + 1$
21       Message('The username or password is incorrect')
22     else
23       if (ATTChallenge() = Pass) then
24         Message('The username or password is incorrect')
25       else
26         Message('The answer to the ATT challenge is incorrect')

27 end
```

4.3. PGRP Operations

PGRP uses the following functions:

1. ReadCredential(OUT: un,pw,cookie). Shows a login prompt to the user and returns the entered username and password, and the cookie received from the user's browser (if any).
2. LoginCorrect(IN: un, pw; OUT: true/false). If the provided username-password pair is valid, the function returns true; otherwise, it returns false.
3. GrantAccess(IN: un, cookie). The function sends the cookie to the user's browser and then enables access to the specified user account.
4. Message(IN: text). Shows a text message.
5. ATTChallenge(OUT: Pass/Fail). Challenges the user with an ATT and returns "Pass" if the answer is correct; otherwise, it returns "Fail."
6. Valid Username. If the provided username exists in the login system, the function returns true; otherwise, it returns false.
7. Valid. First, the function checks the validity of the cookie where it is considered invalid in the following cases: 1) the login username does not match the cookie username; 2) the cookie is expired; or 3) the cookie counter is equal to or greater than k1. The function returns true only when a valid cookie is received. If state =true, a new cookie is created including the following information: username, expiry date, and a counter of the number of failed login attempts. Notice that if state = true, the function does not send the created cookie to the user's browser. Rather, the cookie is sent later by the GrantAccess() function. If state = false and a valid cookie is received, the cookie counter is incremented by one and the cookie is sent back to the user's browser. No action is performed for all the other cases.

4.4.Cookies versus Source IP Addresses

Similar to the previous protocols, PGRP keeps track of user machines from which successful logins have been initiated previously. Browser cookies seem a good choice for this purpose if the login server offers a web-based interface. Typically, if no cookie is sent by the user browser to the login server, the server sends a cookie to the browser after a successful login to identify the user on the next login attempt. However, if the user uses multiple browsers or more than one OS on the same machine, the login server will be unable to identify the user in all cases. Cookies may also be deleted by users, or automatically as enabled by the private browsing mode of most modern browsers. Moreover, cookie theft might enable an adversary to impersonate a user who has been successfully authenticated [7]. In addition, using cookies requires a browser interface.

Consequently, we choose to use both browser cookies and source IP address in PGRP to minimize user inconvenience during the login process. Also, by using IP addresses only, PGRP can be used in character-based login interfaces such as SSH. An SSH server can be adapted to use PGRP using text-based ATTs. For example, a prototype of a text-based CAPTCHA for SSH is available as a source code patch for OpenSSH [12].

The security implications of mistakenly treating a machine as one that a user has previously successfully logged in from is limited by a threshold such that after a specific number of failed login attempts an ATT challenge is imposed. For identification through a source IP address, the condition FS[srcIP;] un < k_1 in line 4 limits the number of failed login attempts an identified user can make without answering ATTs). Also, the function Valid in line 4 updates a counter in the received cookie in which the cookie is considered invalid once this counter hits or exceeds k_1. This function is also called in line 16 to check this counter in case of a failed login attempt.

4.5. Decision Function for Requesting ATTs

Below we discuss issues related to ATT challenges as provided by the login server in Fig. 1. The decision to challenge the user with an ATT depends on two factors: 1) whether the user has authenticated successfully from the same machine previously; and 2) the total number of failed login attempts for a specific user account.

5. PROTOCOLS COMPARISON

We analyze the security, usability, and required system resources of PGRP as compared to a strawman protocol and the PS and VS protocols. This section also provides a comparative summary of major limitations in each protocol.

5.1. Security Issues

Following the previous analysis of PS, assume a fixed password space of cardinality N, assume passwords are equi-probable, and that the delay between when the {username, password} pair is entered and the ATT challenge is presented to the user is identical whether or not the credentials are correct. Also assume that cookie theft, and adversaries using appropriate users' IP addresses occur rarely.

5.2. Usability and ATT Challenges

Our main security goal is to restrict an attacker who is in control of a large botnet from launching accessible single-account or multiaccount password dictionary attacks. In terms of usability, we want to reduce the number of ATTs sent to appropriate users as much as possible. A user receives ATTs when the total number of failed attempts exceeds threshold k_2, and the login attempt is initiated from 1) an unknown machine or 2) a known machine from which the user has already failed k1 times. This happens for both cases of correct and incorrect username-password pairs, assuming the provided username is valid. Below we discuss different login scenarios and the extra effort as required from users by PGRP. The analysis below indicates that only limited usability impact may be expected from our proposal; the same can also be inferred from our real-world data analysis.

5.3. Resource Utilization

No lists are maintained in the PS protocol, thus no extra memory overhead is imposed on the login server. In the VS protocol, only FT is maintained. The number of entries in this list grows linearly with unique usernames used in failed login attempts. An attacker may try to exhaust a login server's memory by failed login attempts for many usernames. For any cookie-based login protocol, the login server may also need to store information regarding each generated cookie to ameliorate cookie theft attacks. Note that neither the PS nor VS protocol uses IP addresses. The most expensive server operation in PS, VS, and PGRP is generating an ATT.

6. PGRP WITH CRYPTOGRAPHY SUPPORT

The Password Guessing Resistant Protocol (PGRP) is enhanced to control cookie thefts. Black lists are used to manage the attacker addresses under login verification. Compromised machine attacks are handled with the user name and IP address associations. Concurrent login verification is applied with session details. The password communications are secured with RSA algorithm. The system is designed to protect the user login process from attacks. The PGRP is adapted to secure SSH and web login interfaces. The PGRP is enhanced with compromised machine attack handling schemes. The system is constructed with six modules. They are Authentication server, Client communication, Address handler, Turing tests, Interface controller and Security process.

The authentication server maintains the users and their password details. The client communication is designed for remote login process. The address handler management module is designed to maintain white list and black list addresses. The turing tests are initiated to protect user attacks. The interface controller is designed to control session and cookie based attacks. The security process module is designed to test the user account server with automatic attack generator.

6.1. Authentication Server

The authentication server maintains the user details with password information. User accounts are maintained with their login history information. User accounts are created under the client application. User verification is performed in the authentication server.

6.2. Client Communication

The system supports two types of client interfaces. They are SSH login interface and web login interface models. The secure shell (SSH) login interface allows console based access models. The web login interface is designed for browser based access. The RSA algorithm is used to secure user name and password details.

6.3. Address Handler

The system maintains two types of IP lists to manage user login details. They are white list and black list addresses. The appropriate user addresses are maintained under white list. Attacker addresses are maintained under black list.

6.4. Turing Tests

The automatic turing test (ATT) is used to handle continuous login attempts. ATT process initiates user responses for query process. They system uses two types of ATT process. Audio and image based ATT operations are used in the system.

6.5. Interface Controller

The interface controller verifies the cookies and user sessions. The cookies are used to store the login information under the client machine. The cookie values are integrated with time based signature values to protect cookie thefts. The session verification is performed to protect concurrent login process. Historical user access sessions are verified with login attempts.

6.6. Security Process

The attacker application is designed to test the password protection system with automatic login attempts. The system is tested with brute force attacks and password dictionary attacks. Character combinations are used in brute force attacks. Password dictionary is used in the dictionary based attacks.

7. CONCLUSION

Remote login schemes are used to access system resources on the Intranet and Internet environment. Password Guessing Resistant Protocol is used to manage password attacks. The PGRP is enhanced to protect cookie theft based attacks. Session based attacks are also handled by the system. The client authentication is secured using RSA algorithm. The system handles single account attacks and multi account attacks. Graphical and console based login interfaces are supported by the system. The system usability is controlled. The system provides high security on remote login applications.

REFERENCES

[1] Mansour Alsaleh, Mohammad Mannan, and P.C. van Oorschot, "Revisiting Defenses against Large-Scale Accessible Password Guessing Attacks" IEEE transactions on dependable and secure computing, vol. 9, no. 1, january/February 2012

[2] “Botnet Pierces Microsoft Live through Audio Captchas,” TheRegister.co.uk, http://www.theregister.co.uk/2010/03/22/microsoft_live_captcha_by pass/, Mar. 2010.

[3] E. Bursztein, S. Bethard, J.C. Mitchell, D. Jurafsky, and C. Fabry, “How Good Are Humans at Solving CAPTCHAs? A Large Scale Evaluation,” Proc. IEEE Symp. Security and Privacy May 2010.

[4] B. Pinkas and T. Sander, “Securing Passwords against Dictionary Attacks,” Proc. ACM Conf. Computer and Comm. Security, Nov. 2002.

[5] D. Ramsbrock, R. Berthier, and M. Cukier, “Profiling Attacker Behavior following SSH Compromises,” Proc. 37th Ann. IEEE/IFIP Int’l Conf. Dependable Systems and Networks (DSN ’07), pp. 119-124, June 2007.

[6] SANS.org, “Important Information: Distributed SSH Brute Force Attacks,” SANS Internet Storm Center Handler’s Diary, http://isc.sans.edu/diary.html?storyid=9034, June 2010.

[7] K. Fu, E. Sit, K. Smith, and N. Feamster, “Dos and Don’ts of Client Authentication on the Web,” Proc. USENIX Security Symp., pp. 251-268, 2001.

[8] J. Yan and A.S.E. Ahmad, “Usability of CAPTCHAs or Usability Issues in CAPTCHA Design,” Proc. Symp. Usable Privacy and Security (SOUPS ’08), pp. 44-52, July 2008.

[9] Y. He and Z. Han, “User Authentication with Provable Security against Accessible Dictionary Attacks,” J. Networks, vol. 4, no. 3, pp. 200-207, May 2009.

[10] P.C. van Oorschot and S. Stubblebine, “On Countering Accessible Dictionary Attacks with Login Histories and Humans-in-the-Loop,” ACM Trans. Information and System Security, vol. 9, no. 3, pp. 235-258, 2006.

[11] M. Motoyama, K. Levchenko, C. Kanich, D. Mccoy, G.M. Voelker, and S. Savage, “Re: CAPTCHAs Understanding CAPTCHASolving Services in an Economic Context,” Proc. USENIX Security Symp., Aug. 2010.

[12] C. Namprempre and M.N. Dailey, “Mitigating Dictionary Attacks with Text-Graphics Character Captchas,” IEICE Trans. Fundamentals of Electronics, Comm. and Computer Sciences, vol. E90-A, no. 1, pp. 179-186, 2007.

[9]. Kundur, D., & Hatzinakos, D. (1999), “Digital watermarking for telltale tamper proofing and authentication”. In Proceedings of the IEEE special issue on identification and protection of multimedia information (pp. 1167–1180).

[10]. Li, C.-T., & Yuan, Y. (2006). “Digital watermarking scheme exploiting nondeterministic dependence for image authentication”. Optical Engineering, 45(12), 127001.

[11]. Lin, C. Y., & Chang, S. F. (2000). “Semifragile watermarking for authenticating JPEG visual content”. In Proceedings of SPIE conference on security and watermarking of multimedia contents II (pp. 140–151).

[12]. Lin, E. T., Podilchuk, C. I., & Delp, E. J. (2000). “Detection of image alterations using semi-fragile watermarks”. In Proceedings of SPIE conference on security and watermarking of multimedia contents II (pp. 152–163)