Research Paper Engineering

User Authentication to Provide Security against Online Guessing Attacks

*A. Sai Kumar **P. Subhadra

* M.Tech (C.S.E), VCE,Hyderabad

** Associate Professor in CSE Dept, VCE,Hyderabad

ABSTRACT

Passwords are one of the most common causes of system break-ins, because the low entropy of passwords makes systems vulnerable to brute force guessing attacks (dictionary attacks). Automated Turing Tests (ATTs) continue t o be an effective, easy-to-deploy approach to identify automated malicious login attempts with reasonable cost of inconvenience to users. In this paper, we discuss the inadequacy of existing and proposed login protocols designed to address large- scale online dictionary attacks (e.g., from a botnet of hundreds of thousands of nodes). We propose a simple scheme that strengthens password-based authentication protocols and helps prevent online dictionary attacks as well as many-to-many attacks common to 3-pass SPAKA protocols.

Keywords: online password guessing attacks, brute force attacks password dictionary, ATTs

1. Introduction

Online guessing attacks on password-based systems are inevitable and commonly observed against web applications and SSH logins. In recent report SANS identified password guessing attacks on websites as a top cyber security risk. As an example of SSH password-guessing attacks, one experimental Linux honey pot setup has been reported 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, However online 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 limited number of guesses from a single machine before being locked out, delayed, or challenged to answer Automated Turing Tests. 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 weak passwords and attackers currently control large botnets online attacks are much easier than before. However, this inconveniences the legitimate 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.

Due to successful attacks which break ATTs without human solvers. 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, feeling 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 but logins to more ATTs, to drive up the economic cost to attackers. We present a scheme that strengthens existing authentication protocols against online dictionary attacks. We integrate our scheme with SPAKA protocols to strengthen SPAKA protocols against online dictionary attacks as well as many-to-many guessing attacks. We call the modified SPAKA protocols as SPAKA+. Overview of Our approach there is a fundamental difference between the login attempts performed by the legitimate users and the login attempts performed by the attackers trying to launch online dictionary attacks. A user who knows the password can successfully login within a couple of trials, while an attacker is expected to perform several magnitudes more trials than legitimate users do. In general, one of the main factors that limit the success of an attacker attempting to launch dictionary attacks is the amount of time required by a program (password cracker) to guess a user's password. The threat of parallel attacks can be eliminated by requiring the client to send a "proof of work" with an aim to keep the attacker busy and reduce the number of sessions that an attacker can initiate. Incentives to buy such network of zombies.

Fig: Example of screen

130 SPAKA+ strengthens SPAKA protocols against online dictionary attacks and many-to-many attacks by asking clients to solve a cryptographic puzzle (proof of work). The scheme is designed to distinguish between legitimate users and attackers and puts negligible computational burden on the legitimate users. Attackers are forced to solve puzzles, which increases the complexity of online dictionary attack ap-

proximately by the hardness of puzzles. If under attack, the authentication server can self-adjust the hardness of the puzzle. Therefore, our protocol will impose significant computation burden on sophisticated attackers using rented zombies, and, thus, greatly increase the amount of time required to break passwords. The computational burden on the authentication server is negligible and the server has to maintain only one long term state information (near-stateless) if users' computers are assumed to be secure. In case users' com- putters are not secure, we suggest a way to minimize the success of the attacker by maintaining some state information on the server. Our generic puzzle-based scheme can be generalized to non-SPAKA protocols, such as the authentication protocols used with SSL or SSH as long as the basic protocols generate shared secrets (e.g., session keys) between the client and the server.

2. SPAKA Protocols

Since Loma set al. introduced LGSN in 1989 there have been considerable research efforts on Strong Password-based Authentication and Key Agreement (SPAKA) protocols, such as EKE, SPEKE, SRP, AMP, AuthA, PAK etc. SPAKA protocols are remote password-only protocols. They can provide authentication and Key agreement over insecure channel without the support of previously shared cryptographic keys or a Public Key Infrastructure (PKI). We list security properties of SPAKA protocols below.

If a user tries to login from a computer without a valid (cookie, ticket) pair, the client must solve a puzzle.

In other words, an attacker cannot verify her guesses without solving a puzzle, even if she launch many-to-many guessing attacks to 3-pass protocols.

A client with a valid (cookie, ticket) pair does not have to solve a puzzle. In this case, our protocol adds negligible computation on the legitimate client side.

They add negligible computation on the server side.

3. Discussion

Usability given a computationally intensive cryptographic puzzle, deferent machines may spend deferent amount of time to solve it. If a legitimate user is using a slower machine, she has to spend more time solving a puzzle. However, after a successful login she will have a (cookie, ticket) pair, and, therefore, she does not need to solve puzzles as long as she keeps using the same set of machines. The usability of the system is not sacrificed. Client-side cookies in our approach, cookies are stored in users' computers. Once the client receives a new cookie (after successful login), the client can simply delete the stale cookies for that account therefore, the maximum number of cookie stored on a user's computer is equal to the number of accounts the user has. If cookies are stored in the server, then the server has to store all cookies that have not expired. If the authentication service is heavily used and if the lifetime of cookies is long, then the server has to store a large number of cookies. No information leakage. Comparing to the 3-pass or 4-pass general protocols, more information is sent in our new protocols, namely puzzle, k cookie.

These pieces of information do not help Eve to find valuable information, such as the password, the verifier previous session keys, or the new session key. Analyzing the possibility of ticket theft. If an attacker is able to steal a ticket, then she may be able to bypass our puzzle. We now analyze the scenarios in which a client's ticket can be stolen. If a user is always using her home (or personal) computer or a well administered lab computer with appropriate user accounts, then the chances of stealing the client's ticket are slim, since the ticket is protected with a strong access control mechanism. Even in the case where a user is using a well administered lab computer, which is using a network file system, the chances that an attacker successfully steals a client's ticket are low as the attacker will have to run a sni□er on the local network to eavesdrop the ticket stored on the remote file server, which typically is di□cult (or to some extent easy to detect) in a well administered lab. Here we are going to use hardware kit called as gsm modem for generating otac code to mobile and more for every successful login system need to be generate the code to the mobile as well as mail id then the user need to be validate the account while user login he need to be enter the username, password, ip address and the code which was sent to the mobile.

4. Conclusion and Future Work

We introduced SPAKA+ that strengthens SPAKA protocols against online dictionary attacks using cryptographic puzzles. SPAKA+ significantly increases the complexity of online dictionary attacks as well as many-to-many guessing attacks. The server can self-adjust the computational burden of an attacker by tuning the hardness of the puzzles in real-time based on the server's estimate of ongoing online dictionary attacks. SPAKA+ is secure and adds negligible load on the legitimate clients. We designed a simple cryptographic puzzle that utilizes the nice structure of the SPAKA protocols for generating puzzles, transferring puzzles and solutions, and verifying solutions. The puzzle is designed such that attackers cannot relay it to others. The puzzle does not require any function other than those used in the SPAKA protocols. As a result, the new scheme can be used without requiring additional support from the underlying system and is easy to implement. If the users' computers are secure, the server's load in our scheme is low. Whereas if the users' computers are assumed to be insecure we have presented an approach that resists stolen tickets by maintaining state information on the server.

Future work includes integrating the new scheme into available SPAKA implementations, such as SRP and PAK. We plan to implement an experimental system to perform online dictionary attacks and many-to-many guessing attacks in order to evaluate the success of our scheme. A detailed evaluation both from performance perspective as well as usability perspective will be performed. We also plan to apply our idea to SSH.

REFERENCES

[1]. M. Bellare and P. Rogaway. The AuthA protocol for password-based authenticated key exchange, 2000. Submission to IEEE P1363.2. | [2]. S. M. Bellovin and M. Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In IEEE Symposium on Security and Privacy, | 1992. | [3]. V. Boyko, p. MacKenzie, and S. Patel. Provably secure password authentication | and key exchange using di□e-hellman. In EUROCRYPT, 2000. | [4]. P. Buxton. Egg rails at password security. Netimperative, June, 24, 2002. | [5]. CERT. TCP syn flooding and ip spoofing attack. CERT Advisory CA-96.21, November 1996. | [6]. D. Dean and A. Stubblefield. Using client puzzles to protect TLS. In the 10th | Annual USENIX Security Symposium, 2001. | [7]. D. Denning and G. Sacco. Timestamps in key distribution systems. Communications. | | [8]. C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In | CRYPTO, 1993. | [9]. IEEE P1363 Working Group. IEEE P1363-2: Standard specifications for password- | based public key cryptographic techniques. http://grouper.ieee.org/groups/ | 1363.10. D. P. Jablon. Strong password-only authenticated key exchange. Computer Com- munication Review, 1996.