END USER AUTHENTICATION (EUA)

George S. Oreku, Jianzhong Li
Department of Computer Science and Engineering HIT, P.O.Box 773, 92 Xi Dazhi Street
Nangang District, Harbin 150001 China

Fredrick J. Mtenzi
School of Computing, Dublin Institute of Technology, Dublin 8, Ireland

Keywords: Authentication, Tickets, End-User, Authorization, Kerberos.

Abstract: Authentication is one among a set of services that constitute a security sub-system in a modern computing or communications infrastructure. End User Authentication flexibility model would proposed in this paper allow the user to have multiple authentication mechanisms with varying levels of guarantee, and for suppliers to request and rely on mechanisms appropriate to the service requested. Authentications to end-user in a simple three level ticket request model algorithms solution on open distributed environment. This paper describes the ticket used by clients, servers, and Kerberos to achieve authentication toward prevention of unauthorized access to in sourced data on applications level. However we explore an approach to end user authentication that generalizes the notion of a textual password that, in many cases, improves the security. Our approach is based on the use of Kerberos authentication technique and Diffie-Hellman Key exchange.

1 INTRODUCTION

For the vast majority of computer systems, authentication for users and passwords are the method of choice for access control security mechanism. In consumer applications as diverse as financial transactions, remote computer login, building access control, and keyless entry is extremely important for prove. With that idea in mind in distributed environment, three approaches to security can be envisioned:

1. Rely on each individual client workstations to assure the identity of its user or users and rely on each server to enforce a security policy based on user based identifications (ID).
2. Require that client system authenticate themselves to server, but trust the client system concerning the identity of its user.
3. Require the user to prove identity for each service invoked. Also require that servers prove their identity to client.

Raising questions to eCommerce security this paper presents authentications and authorization service model algorithms to an end user by the use of textual password. We extend the use of the last models by Diffie-Hellman Key Exchange A Non-Mathematician's Explanation (Palmgren,2005) and *Kerberos* authentication model (Steiner et al,1998) quoting at length to place authentication in proper systems context use.

2 EUA MODEL ARCHITECTURE

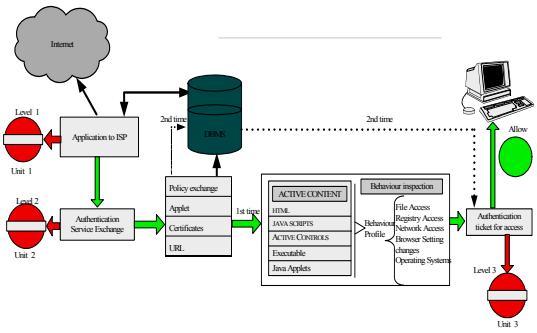

Figure 1: Architecture of a EUA access model.

S. Oreku G., Li J. and J. Mtenzi F. (2007).
END USER AUTHENTICATION (EUA).
In *Proceedings of the Third International Conference on Web Information Systems and Technologies - Internet Technology*, pages 406-409
DOI: 10.5220/0001278604060409

Our model is built on the premise that defensive, or exclusionary, security must be aligned with inclusionary tools and practices that allow users to access systems and information anytime, anywhere.

We illustrate the architecture of the access for an end User Authentication as follows:

1 The "early request ticket for the key exchange authentication from the DBMS unit" that provides identification that lives beyond the span of single first level authorizations or interaction
2 Level two makes it possible to associate successive message or request to third level.
3 The single use authorization, which conceals user's identities and also limits linkage among given users successive actions is in third level.
4 The three different units (Unit 1, Unit 2 and Unit 3) verify the user in application-level by sending different request in each unit.

2.1 Message Exchange Algorithm

We analyze the semantics basing on *Kerberos* authentication technique key exchange (Palmgren, 2005) and Diffie-Hellman Key exchange of each security transcoding by using an online contracting scenario.

Basic notations

C = Client, *ADBMS* = Authentication database Management System, *WS* = Web Source, *IDc* = Identifier of user on *C*, *Idws*= Identifier of Web source, P_C = Password of user on *C*, *Kws* = secret encryption key shared by *ADBMS* and *WS*, *TS* = timestamp, || = concatenation

Steps

(6) With the ticket, *C* can now apply to *(WS)* for service by sending a message to *(WS)* containing *C*'s *ID* and the ticket.
  (6.1) *(Ws)* decrypts the ticket and verifies that the user *ID* in the ticket is the same as the unencrypted user *ID* in the message.
  (6.2) If the two matches, the server considers the user authenticate and grant the requested service.

(7) *Simply stated:*
  (6.1) C Ú ADB : IDc || Pc || IDws
  (6.2) ADBMS Ú C: Ticket
  (6.3) C Ú WS: IDc || Ticket

Authentication Service Exhange: To obtain Ticket-Granting Ticket.

(1) C Ú ADBMS: IDc || IDtgs ||TS1

(2) ADBMS Ú C:EKc [Kc,tgs|| IDtgs || TS2 || Lifetime2 || Tickettgs]

Ticket-Granting Service Exchange: To obtain Service-Granting Ticket.

(3) C Ú TGS: IDv ||Tickettgs ||Authenticator C

(4) TGS Ú C: EKc [Kc, ¨WS || IDws || TS4 || Ticketv]

Client/WebService Authentication Exhange: To Obtain Service

(5) C Ú WS: Ticketv || Authenticator C

(6) WS Ú C: EKc,v[TS5 +1]

3 EUA ASSESSMENT

Considering today's pervasiveness of malicious software (Viruses, Trojan horses) and phishing attacks, any authentications solution must be resistant against offline credential stealing attacks. For this we propose a Challenge/response-based one-time passwords authentication.

3.1 Passwords

We represent these requirements (Jermyn et al 1999) in an authentication system consisting of five components. Components are defined inductively as follow:

1. The set A of *authentication Information* is the set of specific information with which entities prove their identities.
2. The set C of *complementary Information* is the set of information that the system stores and uses to validate the authentication information.
3. The set F of *complementation functions* that generate the complimentary from the authentication information that is: -
   For $f \in F, F : A \to C$

The set L of *authentication functions* that verify identity. That is for $| \in L, | : A \times C \to$ *{true, false}*

The set S of *selection functions* that enable an entity to create or alter the authentication and complementary information.

The goal is to find an $a \in A$ such that, for $f \in F$, $f(a) = c \in c$ *and c* is associated with a particular entity (or any entity).Because one can determine weather *a* is associated with and entity only by computing *f (a)* or by authenticating via | *(a)* we have two approaches for protecting the password, used simultaneously

1. Hide enough information so that one of *a, c,* or *f* can not be found.
2. Prevent access to the authentication functions *L*

3.2 Security Considerations

In both approaches, the goal of the defenders is to maximize the time needed to guess the password.

- Let *P* be the probability that an attacker guess a password in specified period of time
- Let *G* be the number of guesses that can be tested in one time unit
- Let *T* be the number of time units during which guessing occurs
- Let *N* be the number of possible passwords

Then

$$P \geq \frac{TG}{N} \quad (1)$$

Example1: Lets password be composed of characters drawn from an alphabet of 96 characters. Assume that 10^4 guesses can be tested each second. We wish the probability of a success guess to be 0.5 once a 365-days period. What is the minimum password length that will give us this probability?

From the formula above, we want

$$N \geq \frac{TG}{P} = \frac{(365 \times 24 \times 60 \times 60)10^4}{0.5} = 6.31 \times 10^{11} \quad (2)$$

Thus we must choose an integer s such that $\sum_{i=0}^{s} 96^i \geq N$

This holds when $s \geq 6$ so, to meet the desired conditions password of at least length *6* must be required.

Assumptions underlie example:

1. The time required to test a password is constant
2. All passwords are equally likely to be selected

Proof:

(a) The first is reasonable, because the algorithms used to validate password are fixed and either the algorithm are independent of the password's length or the variation is negligible.

(b) The second assumption is a function of the password selection mechanism. (Morris and Thomson, 1979)

3.3 One – time Passwords

One time password is a password that is invalidated as soon as it is used. It uses techniques first suggested to generate the password. (Lamport, 1981)

With this technology our System takes the "seed" user enters and generates a list of *n* passwords. The implementation presents each password as a sequence of *six* shorts words (but the internal presentation is an Integer).

1. User supplies his name to the server
2. The server replies with the number *i* stored in the *ticket* file
3. User supplies the corresponding password *pi*
4. The server computes $h(p_i) = h(k_{ni+1}) = k_{ni+2} = p_{il}$ and compares the results with the stored password. If they match, the ticket file to *i1* and stores *Pi* in the file. If the authentication fails the ticket file is left unchanged

Note: *h* – one-way hash function, *K* – Seed.

3.4 Performance Analysis

We will derive and analyze the robustness of passwords capabilities by probability letting unauthorized end-user guessing capability to allow its login within a particular time.

Parameters:

X: password length, Z: time.

The probability that randomly guessed capability will pass a particular authorization is given by

$$P(X, Z) = 1 - \left(1 - \frac{1}{2^Z}\right)^X \quad (3)$$

and the probability that a randomly guessed capability will pass all d authorization in a system is simply $p(x,z)^d$.

Recall that the capability password authorization *must* be > than *6* symbols (characters). Figure 2 and 3 shows login performance over time and different password length.

We leave the exact timing decisions to $s \geq 6$ symbols, and simply assume in our experiments that *x* is likely to be from two to *six*. From the graph the probability log in comes close to matches as the numbers of input values increases within different time. The performance is excellently, 97.14% of the attack using short key length per log in from $(z = 1)$, and increasing probability to 100% of login match to key length ≥ six with a marking of running time $(z \geq 4)$. As we show in figure 3, *X* must be at least *2*, because a valid capability is 100% that matches any of the *X* capabilities in the DBMS, small value of *X* provide the smallest probability that a randomly chosen capability will match the password.

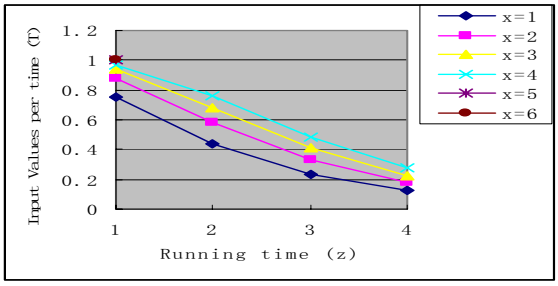

Figure 2: Inputs per time.

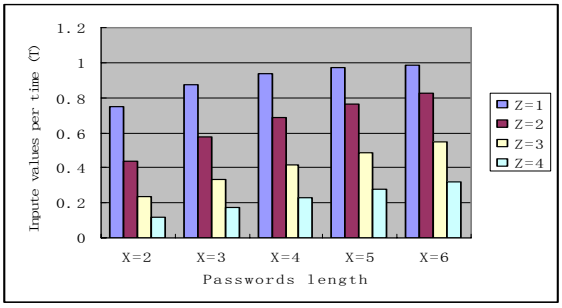

Figure 3: Passwords length per input values.

The value of X also affect the validity time of capability. The minimum validity time $(\min_m)$ *is* $(\min_m) = (x-1).T_k$, where T_k denotes the time a key is typed in. The maximum validity time $(\max_m) = X.T_k$. Ideally, we would like to get a small intervals for the validity time, so that we can tightly control the validity period, so we would like large value of X to minimize the difference between $(\min_m)$ *and* $(\max_m)$. We can determine X from Min_m and Max_m

$$X = \frac{\max_m}{\max_m - \min_m} \quad (4)$$

The $\max_m$ metric defines the longest amount of time that passwords matches can remain idle and still have a valid capability. Put in another way, Max_m defines the maximum amount of time that an attacker can login with particular capability before the capability is rejected by authentications.

Because the (login denial) rejection probabilities are independent *Bernoulli trials* the probability that the end user/client and server will be able to establish correct password after one try (by exchanging password is:

$$P\ (connect\ after\ 1\ try) = (1 - \in_i)^i$$

The probability that client will connect after K tries is:-

$$P\ (connection\ after\ K\ tries) = 1 - (1 - P\ (connect\ after\ 1\ try))^k$$

$$= 1 - (1 - (1 - \in_i)^i)^k$$

For a given desire connection probability, *P (connect)* the required numbers of connection attempts is:-

$$K = \frac{\log\left(1 - P\ (connect)\right)}{\log\left(1 - (1 - \in_i)^i\right)} \quad (5)$$

A nice feature of this formula is that the expected number of connections attempts depends logarithmically on the matches' probability, which indicates that even for large $\in_i$; a determined client can get a connected after a moderate waiting time.

4 CONCLUSION

We have presented an algorithm that explains the ticket message exchange for access to an end user in distributed systems. Using this model as a basis, we also present a countermeasure that may be deployed in the current passwords used, assuming that client and server software is updated.

Our approaches exploit the input capabilities of login that allow us to determine inputs time from the passwords length in which it occurs. We prove our arguments to assess the security mathematically.

REFERENCES

Palmgren, K., **"**Diffie-Hellman Key Exchange – A Non-Mathematician's Explanation" February 2005.

Steiner et al., J.G., *Kerberos:* An Authentication Service for Open Network Systems March 1988.

Ian Jermyn et al, "The Design and Analysis of Graphical Passwords," *Proceeding of the 8th USENIX Security Symposium* August 1999].

Morris, R., and Thomson, K., "Password Security, A case History," *Communications of the ACM 22 (11),* pp.594597 (Nov.1979).

Lamport, L., "Password Authentication with insecure Communication, "*Communication of the ACM 24 (11)*, pp.770771 (Nov. 1981).