Cryptanalysis and Security Enhancement of an Advanced Authentication Scheme using Smart Cards, and a Key Agreement Scheme for Two-Party Communication

Swapnoneel Roy, Amlan K Das and Yu Li
Department of Computer Science and Engineering
University at Buffalo, The State University of New York
Buffalo, NY 14260–2000
http://www.cse.buffalo.edu/

***Abstract*—In this work we consider two protocols for performing cryptanalysis and security enhancement. The first one by Song, is a password authentication scheme based on smart cards. We note that this scheme has already been shown vulnerable to the *off-line password guessing* attack by Tapiador et al. We perform a further cryptanalysis on this protocol and observe that it is prone to the *clogging* attack, a kind of denial of service (DOS) attack. We observe that all smart card based authentication protocols which precede the one by Song, and require the server to compute the computationally intensive modular exponentiation, like the one by Xu et al., or Lee at al., are prone to the clogging attack. We then suggest an improvement on the protocol to prevent the clogging attack.**
The other protocol we consider is a two-party identity-based authenticated key agreement protocol by Hölbl et al. They have devised two such protocols in their work. They call them *Protocol* 1 and *Protocol* 2. Both the protocols have already been shown vulnerable to the *insider* attack in a recent work by Chen et al. Here we consider Protocol 2 and show its vulnerability to a simple man-in-the-middle attack where the adversary does not know or calculate either party's private key, or the session key. Protocol 2 by Hölbl et al is an improvement over a previous work by Tseng. This makes the Tseng's protocol vulnerable to the attack we illustrate. We further suggest an additional step for these protocols to make them immune against the man-in-the-middle attack.

***Keywords*-Password Authentication; Key Agreement; Cryptanalysis; Protocols; Security; Clogging Attack; Denial of Service Attack; Man in the middle Attack; Smart Card;**

The motivation behind development password authentication schemes[1] is to help legitimate users or *clients* to acquire services from a legitimate provider or *server*. Whenever a user wants a service from a provider, it has to identify itself to the provider by some means. Password authentication has been one of the most convenient schemes for user identification over the years. Now a days millions of providers use password authentication schemes to identify legitimate users. Common examples include E-mail services, E-banking services, personal web albums services, blog services, participation on web forums, shopping on Internet, etc.

Basically any password authentication scheme has two phases:

- **Registration Phase:** In this phase, the user registers with an *identity* and *password* with the server. The password is stored by the server and remains a secret between the user and the server.
- **Authentication Phase:** In this phase, the user wants a service from the server. It sends its identity and password to the server to gain a service. The server then determines whether the user is legitimate by comparing the received values of identity and password to the stored values. The server extends the desired service to the user, if found legitimate.

During authentication processes, if the password is transmitted over a public communication channel in plain-text form, an adversary can intercept the message by eavesdropping the communication. He can impersonate the user by reusing the password obtained from the message. These matters threaten the user privacy. The organizations generally store the passwords of the users in a database for future verification and authentication. Directly storing the plain-text passwords in a password table in the server's database is vulnerable. It does not provide any security against privileged insiders of the server. Nor does it protect the passwords if the server's database is somehow hacked. In order to eliminate the problem of password table disclosure, the servers might encrypt the password and store them. However, the communication interception still remains a threat to the system security. Another problem is revoking a users identity and password. A compromise in the password for any user might be like losing a credit card. The adversary could take advantage before the organization is notified about the loss. Hence we need to develop really efficient password authentication schemes to make the authentication carried out in a secured manner.

Keeping in mind the above issues, and to make the system more secure, many smart card based password authentication protocols have been developed over the last decade [1], [3], [4], [5], [6], [7]. In such a scheme, the user (client) is provided with a smart card. Whenever the client wants a service, it provides its smart card with a password which it keeps secret. The smart card in turn then uses this password

[1]We use the terms *scheme* and *protocol* interchangeably in this paper

to construct a login message which is sent to the server. The server then authenticates this messages and provides the desired service is the password is found valid.

In this paper, we consider such a protocol based on smart cards by Song[2] [1]. He had designed this protocol as an improvement over another protocol by Xu et al. [3]. In a recent work by Song [1], the protocol by Xu et al. has been shown vulnerable to the *impersonation* attack. Song then designed this advanced protocol in the same work, to prevent the impersonation attack on the protocol in [3]. Here we show, that the Song's scheme is vulnerable to the *clogging* attack, a kind of the *denial of service* (DOS) attack. We had found out earlier in an unpublished work that the protocol by Xu et al. was prone to clogging attack. In that work, we had conjectured that Song's scheme is also vulnerable to clogging attack. In this attack, the adversary can simply stop the server from providing any service without having any knowledge of the user information like its identity or password. Nor does the adversary need to make any complicated calculations to launch a clogging attack on Song's protocol. We illustrate this in this work. We have come to know about a cryptanalysis of the Song's protocol by Tapiador et al [2]. But they did not consider performing the clogging attack on the protocol in their work. Its also worth mentioning here, that the chain of smart card based authentication protocols which make the server compute the computationally intensive modular exponentiation like [3], [5], are all vulnerable to the clogging attack. The adversary takes the advantage of the computation intensiveness of the modular exponentiation calculation in launching this attack. We then suggest a step to prevent clogging attack on these protocols.

Key agreement protocols are needed in various kinds of communication. Keys need to be securely exchanged before a communication could be established. There are some security threats to this namely intruder-in-the-middle, where the adversary pretends to be somebody else to both the communicating parties. Replays of old keys is another attack that is common in this aspect. Thus is the need to develop secure key exchanging protocols to establish secure communication.

The key agreement schemes generally have two steps:

- **Decide the public/private key pairs:** In this phase, both the parties calculate a pair of keys. A private key, which they keep secret and a public key which they let other users know. In some recent protocols, a part of this phase is done by a *key distribution center*, which stores the public keys of the users in a directory.
- **Decide the secret session key:** In this phase, the users exchange their public keys and/or some values. The secret session key for communication is calculated based on those values, and/or the private and public keys. A number of protocols also let one of the parties decide the secret key or *session key* and send it to the other party by encrypting it along with the message.

In this paper we consider such a protocol by Hölbl et al.[3] [8]. Actually there are two such protocols proposed in [8] termed as Protocol 1 and Protocol 2. We consider Protocol 2 and show that a man-in-the-middle attack can be made on it without any knowledge of the users' information, or without performing any sort of calculations. We would like to note that the Hölbl-Welzer scheme has been shown to be prone to the *insider* attack by Chen et al. [9]. They showed that the secret key could be somehow calculated by an insider to break the protocol. Here we show that the adversary would not have to be an insider, and need not have to calculate the key to break the protocol. A simple man-in-the-middle impersonation attack is enough to break the protocol. We further note that Protocol 2 of the Hölbl-Welzer schemes is an improvement over Tseng el al.'s [11] protocol. So that makes the protocol in [11] prone to the same kind of attack. We suggest a step at the end which immunes these protocols from this attack.

The remainder of the paper is organized as follows: In Section I, we review the Song's scheme, we then have a discussion on this scheme in Section II. We illustrate the clogging attack on this scheme in Section III. We then suggest a fix from this attack in Section IV. Section V introduces Protocol 2 of the Hölbl-Welzer scheme of key agreement. In a similar manner we provide a discussion on this protocol in Section VI. Section VII and VIII then illustrate the man-in-the-middle attack on the protocol, and a fix from the attack respectively.

I. Review of Song's scheme

We now briefly illustrate the password authentication scheme by Song [3]. This authentication scheme is based on a smart card. In Section II, we discuss the performance dependencies and security vulnerabilities of this scheme. In Section III, we present a clogging attack on the protocol. We then discuss a possible fix against the attack. At the end, we note that similar schemes by Xu-Zhu-Feng [3], and Tsaur-Wu-Lee [5], which make the server compute modular exponentiation, are prone to the same kind of attack and has the same performance dependencies as this scheme. The Song's scheme consists of 3 phases namely *registration*, *login*, and *authentication*. All the phases are illustrated in Algorithm 1.

The server selects two large prime numbers p and q such that $p = 2q + 1$. The server then selects an $x \in Z_q^*$ as its secret key, an appropriate one-way hash function $h(\cdot)$, and a symmetric key cryptography algorithm with $E(\cdot)$ and $D(\cdot)$

[2]We call this protocol Song's scheme in this paper.

[3]We call this protocol the Hölbl-Welzer scheme in this paper.

REGISTRATION PHASE

User A

1) Select ID_A, PW_A.
2) Send $\{ID_A, PW_A\}$ to the server.

Server S

1) Compute $B_A = h(ID_A^x modp) \oplus h(PW_A)$.
2) Store the tuple $\{ID_A, B_A, h(\cdot), E(\cdot)\}$ into a smart card.
3) Issue the smart card to the user.

LOGIN AND AUTHENTICATION

User A

1) Input ID_A and PW_A from the smart card.
2) Select a random R_A, and set $T_A \leftarrow$ *system current time*.
3) Compute $K_A = B_A \oplus h(PW_A)$, $W_A = E_{K_A}(R_A \oplus T_A)$, and $C_A = h(T_A \,||\, R_A \,||\, W_A \,||\, ID_A)$.
4) Send the message $\{ID_A, C_A, W_A, T_A\}$ to the server.

Server S

1) Verify ID_A, T_A.
2) Compute $K_A = h(ID_A^x modp)$, $R'_A = D_{K_A}(W_A) \oplus T_A$.
3) Verify: $C_A \stackrel{?}{=} h(T_A \,||\, R'_A \,||\, W_A \,||\, ID_A)$.
4) Compute $C_S = h(ID_A \,||\, R'_A \,||\, T_S)$.
5) Send $\{ID_A, C_S, T_S\}$ to the user.

User A

1) Validate ID_A and T_S.
2) Verify: $C_S \stackrel{?}{=} h(ID_A \,||\, R_A \,||\, I_S)$.

COMPUTE SESSION KEY

User A

- $sk = h(ID_A \,||\, T_S \,||\, T_A \,||\, R_A)$

Server S

- $sk = h(ID_A \,||\, T_S \,||\, T_A \,||\, R'_A)$

Algorithm 1: The Song's scheme of password authentication [1]

as the encryption and decryption algorithm respectively. p and x are both kept secret by the server.

A. Registration phase

The scheme performs the following steps:

1) The user sends $\{ID_A, PW_A\}$ to the server through a secure channel, where ID_A and PW_A are the user's identification and password respectively.
2) After it receives $\{ID_A, PW_A\}$, the server computes $B_A = h(ID_A^x modp) \oplus h(PW_A)$.
3) The tuple $\{ID_A, B_A, h(\cdot), E(\cdot)\}$ is then stored by the server into a smart card and issued to the user.

B. Login phase

The smart card is attached to a card reader by the user and it provides its ID_A and PW_A. The smart card then performs the following procedure:

1) Select a random R_A, and set $T_A \leftarrow$ *system current time*.
2) Compute $K_A = B_A \oplus h(PW_A)$, $W_A = E_{K_A}(R_A \oplus T_A)$, and $C_A = h(T_A \,||\, R_A \,||\, W_A \,||\, ID_A)$.
3) Transmit the message $\{ID_A, C_A, W_A, T_A\}$ to the server.

C. Authentication phase

In this phase, the following steps are performed by the server and the smart card for mutual authentication. At the end, a *session key* K is agreed by the two parties.

1) The following are performed by the server upon receiving the login message from the user:
   - Check whether ID_A is valid. If not, reject the login request.
   - Check whether the difference between $T^* - T_A \leq \delta$ where δ is predefined threshold, and T^* is the server time at which it receives the login request.
   - Compute $K_A = h(ID_A^x modp)$, $R'_A = D_{K_A}(W_A) \oplus T_A$ and check whether $C_A = h(T_A \,||\, R'_A \,||\, W_A \,||\, ID_A)$. If not, reject the login request.
   - If the verification in the previous step succeeds, set $T_S \leftarrow$ *current server time*, compute C_S $h(ID_A \,||\, R'_A \,||\, T_S)$, and send the message $\{ID_A, C_S, T_S\}$ to the user.
2) The following are performed by the user (smart card) upon the receipt of $\{ID_A, C_S, T_S\}$ from the server:
   - Validate ID_A and T_S.
   - Check whether $C_S = h(ID_A \,||\, R_A \,||\, I_S)$. If they are equal, the server is authenticated.
3) The user and the server then compute the session key $sk = h(ID_A \,||\, T_S \,||\, T_A \,||\, R_A) = h(ID_A \,||\, T_S \,||\, T_A \,||\, R'_A)$.

Note: If the user need to change his password PW_A to a new PW'_A, the smart card can first confirm the validity of PW_A

(by interacting with the server), and on success, resets PW_A to PW'_A, and replaces B_A with $B'_A = B_A \oplus PW_A \oplus PW'_A$.

II. Discussions on Song's scheme

Song's scheme has a big dependency on the server's and the user's clocks. For a connection-oriented application, this might be cumbersome. A protocol must be set up to maintain time synchronization between the clocks of various users and the servers. This protocol must be made fault tolerant, to cope with complicated network errors and also various kinds of attacks. In spite of the channel being secure, an opportunity of an attack can arise and an adversary might intercept a message and change its timestamp T_A. That way the adversary successfully denies the legitimate user a service since the server would reject the login request on the basis of timestamp difference. So this kind of attack is possible even when the scheme prevents replay attacks. Also networking delays might make the timestamp go past the threshold thus making the total service inherently slow.

III. Attack on Song's scheme

We show that Song's scheme is prone to the *clogging attack*. The clogging attack is a kind of *denial of service* attack in which the adversary C repeatedly sends messages to the server and *clogs* it with those messages [14]. Lets see how this could happen with the Song's scheme in place. The following is performed by the adversary C:

1) C intercepts the message $\{ID_A, C_A, W_A, T_A\}$ sent by the user to the server in the login phase.
2) Since the message is unencrypted, C can change the timestamp T_A to some T_C so that it meets the criterion $T^* - T_C \leq \delta$.
3) C changes C_A to any random garbage value C_C.
4) C then sends $\{ID_A, C_C, W_A, T_C\}$ to the server.

The following is performed by the server:

1) Check whether ID_A is valid. Here it is valid.
2) Check whether the difference between $T^* - T_C \leq \delta$. This step passes as well.
3) Compute $K_A = h(ID_A^x modp)$, $R'_A = D_{K_A}(W_A) \oplus T_C$ and check whether $C_C = h(T_C \mid\mid R'_A \mid\mid W_A \mid\mid ID_A)$. This fails, so the request gets rejected.

The point here is the adversary C would now repeat the steps several times and make the server compute the modular exponentiation step several times. Basically C can potentially change all the incoming login request messages from the legitimate user to the server. Since modular exponentiation is computationally intensive, the victimized server spends considerable computing resources doing useless modular exponentiation rather than any real work. Thus the adversary C *clogs* the server with useless work and therefore denies any legitimate user any service. The adversary just needs an ID of a single valid user to perform the clogging attack repeatedly.

Observation: It can be easily seen that the attack just illustrated can also be made on the protocols by Xu et al. [3] and by Tsaur et al. [5]. Hence our observation here is the clogging attack can be performed on all the smart card based authentication protocols based on computing modular exponentiation.

IV. A probable fix from the attack

A. The steps to avoid the clogging attack

At the beginning of the authentication phase, the server could check whether the network address of the user is valid. It has to know the network addresses of all the registered legitimate users. In spite of that, adversary C could spoof the network address of a legitimate user and replay the login message. To prevent it, we might add a *cookie exchange step* at the beginning of the login phase of Song's scheme. This step has been designed as in the well known *Oakley key exchange* protocol [15].

1) The user's smart card chooses a pseudo-random number n_1 and sends it along with the message $\{ID_A, C_A, W_A, T_A\}$.
2) The server upon receiving the message, acknowledges the message and sends its own cookie n_2 to the user.
3) The next message from the user must contain n_2, else the server rejects the message and the login request.

B. Security analysis of the fix

Had C spoofed the user's IP address, C would not get n_2 back from the server. Hence C only succeeds to have the server send back an acknowledgement, but not to compute the computationally intensive modular exponentiation. Hence the clogging attack is avoided by these additional steps. Saying this, we would note that this process does not *prevent* the clogging attack but only *thwarts* it to some extent.

V. Review of Hölbl-Welzer scheme

In this section we consider a key agreement protocol in a public key cryptography scheme [8]. As with the previous protocol, we describe the scheme briefly, and then we illustrate an attack on the underlying protocol. At the end we suggest a fix for the attack.

The Hölbl-Welzer Protocol 2 is a public key cryptography scheme in which the underlying protocol is a secret key agreement between two parties to communicate. It is an advanced and modified version of the well known *Diffie-Hellman Key Exchange* protocol [12]. Specifically, Protocol 2 in [8] is an improvement over the protocol in [11]. We observe that Protocol 2 is prone to the *man-in-the-middle* attack. Hence the protocol in [11] is also prone to the man-in-the-middle attack. Protocol 2 of the Hölbl-Welzer scheme assumes the existence of a Key Granting Center (KGC). We review the protocol in Algorithm 2. Note at the end of the procedure, we have $K_{AB} = K_{BA} = g^{(v_A + r_A)(v_B + r_B)}$. The

other key exchanging protocol (Protocol 1) is quite similar to Protocol 2, which is illustrated here.

VI. Discussion on Hölbl-Welzer scheme

The security of the scheme is based on the hardness of computing discrete logarithms. It is secured against replay attacks, key compromises etc. But since the protocol does not have an identity verification step at the beginning, it is vulnerable to the man-in-the-middle attack. We illustrate this in Section VII.

Global Public Elements

p a prime number for modulus
g a primitive root of p
a one-way function f
$y_s = g^{x_s} modp$, where $x_s \in \mathbb{Z}_{p-1}$
x_s is kept secret with the KDC

Keys Generation for User i by the KDC

Compute $I_i = h(ID_i)$, where ID_i is the identity of the user i
Choose a random number $k_i \in \mathbb{Z}_p$
Compute user i's public key as $u_i = g^{k_i} modp$
Compute user i's private key as $v_i = I_i k_i + x_s u_i (modp - 1)$

User A Sending Operation

Select private $r_A \in \mathbb{Z}_p$
Compute public $t_A = g^{r_A}$
Send $\{u_A, t_A, ID_A\}$ to B

User B Sending Operation

Select private $r_B \in \mathbb{Z}_p$
Compute public $t_B = g^{r_B}$
Send $\{u_B, t_B, ID_B\}$ to A

Calculation of Secret Key by User A

Compute $I_B = h(ID_B)$ and $w_A = r_A + v_A$
Compute $x_A = u_B y_s^{h(ID_B, u_B)} = g^{k_B} g^{x_s h(ID_B, u_B)} = g^{v_B}$
Compute the key $K_{AB} = (t_B x_A)^{w_B} = (g^{r_B} g^{v_B})^{w_A} = g^{w_A w_B} = g^{(v_A + r_A)(v_B + r_B)}$

Calculation of Secret Key by User B

Compute $I_A = h(ID_A)$ and $w_B = r_B + v_B$
Compute $x_B = u_A y_s^{h(ID_A, u_A)} = g^{k_A} g^{x_s h(ID_A, u_A)} = g^{v_A}$
Compute the key $K_{BA} = (t_A x_B)^{w_A} = (g^{r_A} g^{v_A})^{w_B} = g^{w_B w_A} = g^{(v_A + r_A)(v_B + r_B)}$

Algorithm 2: The Hölbl-Welzer Key Agreement Protocol 2 [8]

VII. Attack on Hölbl-Welzer scheme

Figure 1 illustrates an attack on the protocol.

- A sends message $\{u_A, t_A, ID_A\}$ to B to establish a key.
- The adversary C intercepts this message and sends $\{u_C, t_C, ID_C\}$ to both A and B.
- B sends back $\{u_B, t_B, ID_B\}$ to C.
- Since there are no identification verification done by any user communication proceeds uninterrupted.
- A and C establish and share key $K_{AC} = g^{(r_A + v_A)(r_C + v_C)}$.
- C and B establish and share key $K_{CB} = g^{(r_C + v_C)(r_B + v_B)}$.
- A believes that he is communicating with B and send messages meant for B, encrypted using K_{AC}.
- But those messages are now decrypted and read by the adversary C.
- C then encrypts the same message (modified or unmodified) with K_{CB} and relays it to B.
- B does not know that the messages being forwarded to him are originally from A and not from C.
- The attack on this protocol is possible only because the users do not verify the identity of the sender.

In this case, since ID_B or ID_C are just numbers, users A and B might not bother to verify them before calculating the secret session key. Thus it becomes easy for the adversary C to break the protocol.

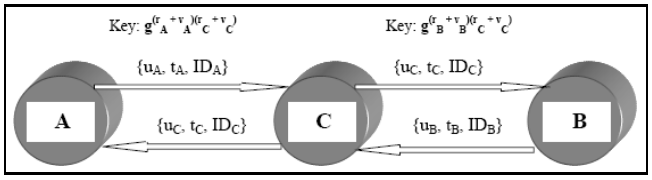

Figure 1. Man-in-the-middle attack on Hölbl-Welzer Protocol 2

VIII. A probable fix from the attack

A. The steps to avoid the man in the middle attack

The protocol would be free from the attack described in Section VII if there is a identity verification step before the users calculate the session key. This identity could be in the form of a digital signature. To implement this, an encryption/decryption function $E(\cdot)/D(\cdot)$ could be chosen. While sending any message, the sender encrypts it with his private key, which is decrypted with the sender's public key by the receiver.

B. Security analysis of the fix

In the example above,

1) A would send the message $E_{v_A}(\{u_A, t_A, ID_A\})$ to B.
2) B then performs $D_{u_A}(E_{v_A}(\{u_A, t_A, ID_A\}))$ to retrieve message $\{u_A, t_A, ID_A\}$.

3) B then does the same operation while replying to A.

It would convince A and B in Figure 1 that the messages actually come from each other and not from C. The assumption here is all users know each other's public keys beforehand.

IX. Conclusion

We have considered two protocols in this work. The first protocol is a password authentication scheme which we have found to be prone to the clogging attack. We showed that the attack on this protocol could be avoided by using an additional step of exchanging numbers. The second protocol we considered is a key exchange scheme. We showed it to be vulnerable to man-in-the-middle attack. Then we showed how to avoid this attack by using an encryption/decryption.

References

[1] Ronggong Song. *Advanced smart card based password authentication protocol*. Computer Standards & Interfaces, Volume 32, Issue 4, June 2010, Pages 321-325.

[2] Juan E. Tapiador, Julio C. Hernandez-Castro, Pedro Peris-Lopez, John A. Clark. *Cryptanalysis of Song's advanced smart card based password authentication protocol*. Unpublished manuscript, June 2010.

[3] Jing Xu, Wen-Tao Zhu, and Deng-Guo Feng. *An improved smart card based password authentication scheme with provable security.*Computer Standards & Interfaces, Volume 31, Issue 4, June 2009, Pages 723-728.

[4] C.C. Chang, T.C. Wu, *Remote password authentication scheme with smart cards*. IEEE Proceedings-Computers and Digital Techniques, vol.138, issue 3, pp.165–168, 1991.

[5] Woei-Jiunn Tsaur, Chia-Chun Wu, and Wei-Bin Lee. *A smart card based remote scheme for password authentication in multi-server Internet services.* Computer Standards & Interfaces, Volume 27, June 2004, Pages 39-51.

[6] Wen-Shenq Juang. *Efficient password authenticated key agreement using smart cards.* Computers & Security, Volume 23, Issue 2, March 2004, Pages 167-173.

[7] H.Y. Chien, J.K. Jan, and Y.M. Tseng. *An efficient and practical solution to remote authentication: smart card*. Computers and Security, vol.21, no.4, pp.372–375, 2002.

[8] Marko Hölbl, and Tatjana Welzer. *Two improved two-party identity-based authenticated key agreement protocols.* Computer Standards & Interfaces, Volume 31, 2009, Pages 1056-1060.

[9] Yalin Chen, Jue-Sam Chou, and Chun-Hui Huang. *Improvements on two password-based authentication protocols.* Cryptology ePrint Archive, 2010.

[10] B.T. Hsieh, H.M. Sun, T. Hwang, C.T. Lin. *An improvement of Saeednia's identity based key exchange protocol.* Information Security Conference 2002, 2002, pp. 41–43.

[11] Y.M. Tseng. *An efficient two-party identity-based key exchange protocol*. Informatica 18 (1) (2007) 125–136.

[12] W. Diffie, M.E. Hellman. *New directions in cryptography*. IEEE Transactions on Information Theory 22 (1976) 644–654.

[13] Hsi-Chang Shih. *Cryptanalysis on Two Password Authentication Schemes*. Master's Thesis. Laboratory of Cryptography and Information Security Department of Computer Science and Information Engineering. National Central University, Chung-Li, Taiwan 320, Republic of China.

[14] William Stallings. *Cryptography and Network Security, 4/E*. Prentice Hall 2006.

[15] Hilarie Orman. *The Oakley Key Determination Protocol*. University of Arizona. TR 97 02