Mansoura Engineering Journal

Volume 33 | Issue 3 Article 6

12-5-2020

New Secret Key Exchange Based on Recent Cryptographic Schemes.

Amina Rashad
Electronics, Communication and Computers Engineering., Faculty of Engineering at Helwan., Helwan University, eng_amina82@yahoo.com

Maged Ibrahim
Assistant Professor., Electronics, Communication and Computers Engineering Department., Faculty of Engineering at Helwan., Helwan University., mhii72@yahoo.com

Zaki Nossair
Assistant Professor., Electronics, Communication and Computers Engineering Department., Faculty of Engineering at Helwan., Helwan University., mossalr@yahoo.com

Recommended Citation

Rashad, Amina; Ibrahim, Maged; and Nossair, Zaki (2020) "New Secret Key Exchange Based on Recent Cryptographic Schemes.," *Mansoura Engineering Journal*: Vol. 33 : Iss. 3 , Article 6.
Available at: https://doi.org/10.21608/bfemu.2020.127484

Mansoura Engineering Journal, (MEJ), Vol. 33, No. 3, September 2008. E. 1

New Secret Key Exchange Based On Recent Cryptographic Schemes

تبادل المفتاح السرى بين المستخدمين باستخدام طرق التشفير الحديثة

Amina Abd El Fattah Rashad Dr:Maged H. Ibrahim Dr:Zaki B.Nossair
Department of Electronics, Communications and Computers engineering,
Faculty of Engineering at Helwan,
Helwan University
E-mail: (eng_amina82@yahoo.com) (mhii72@yahoo.com) (znossair@yahoo.com)

ملخص:
يعتبر بروتوكول تبادل المفتاح الموثوق فيه من اهم الموضوعات لبناء شبكة الاتصالات السريه بين اثنين او اكثر والهدف من هذا البحث هو اقتراح بروتوكول لتوسيع تبادل المفتاح بين اثنين من المستخدمين. فى هذا البروتوكول كل مستخدم يتبادل مفتاح سرى مع الخادم. كل مستخدم يستعمل هذا المفتاح السرى ليتبادل مفتاح سرى اخر مع المستخدم الاخر. وقد اثبتنا فى هذا البحث ان هذا البرتوكول فعال واكثر سريه بالنسبه الى بروتوكولات تبادل المفتاح السرى

Abstract:
Authenticated key exchange protocols have an important role for building secure communications amongst two or more entities over the networks. Two-party authenticated key exchange protocols where each pair of parties must share a secret with each other; a three-party protocol does not cause any key management problem for the parties. In this paper, an extension of two-party key exchange protocol, which is based on Diffie-Hellman key exchange, is proposed. In this protocol each user exchanges secret key with server then each user uses this secret key to exchange session key with each other. The efficiency and the security analysis of this new key exchange protocol are proven in this paper.

Keywords:
Two party, three party, authenticated key exchange, network security

Introduction:
Key establishment protocols are mechanisms that allow any two or more users to establish shared keys amongst themselves. There are two fundamental types of key establishment protocols, *Key transport* and *key exchange*. Key transport protocols, are those in which a single entity is trusted to choose the key and securely transfer it to the other entities.

Key Exchange Protocols Properties [1]

1) Links between key exchange and mutual authentication
   a) Key exchanges must be authenticated to prevent attacks.
   b) A session key makes it possible to extend an initial authentication to the whole communication.
   c) "Authentication and key exchange protocols" provide direct authentication and

Accepted September 29, 2008

authenticated key exchange all-in-one.

2) Forward Secrecy (FS)
   a) Even if an attacker discovers long-term secret(s), he will not be able to recover the session keys (past and future).
   b) Provided when long-term secrets are only used for authentication and do not take part in session keys generation.
3) Identity Protection
   a) No identity is transmitted in the clear, so a spy can't know who the communicating peers are.

There are many different ways to analyze key exchange protocols:

1) Known key security: a protocol run should result in a unique secret session key. If this key is compromised, it should have no impact on other session keys.
2) Forward secrecy: The fact that long-term private keys are compromised, should have no impact on the secrecy of previously established session keys.
3) Key-compromise impersonation resilience: If entity A's long-term private key is compromised, an adversary is able to impersonate A. But this should not enable him to impersonate other entities to A.
4) Key control: Neither of the entities should be able to force the session key to a value of his choice.

Paper organization, this paper is organized as follows: Section 2 represents the previous work of key exchange and the motivations and contributions. The model assumptions are given in section 3. Section 4 describes the proposed protocol. Section 5 represents the security analysis. Section 6 represents Performance discussions. Finally, the conclusions are given in section 7.

2. Previous work:

A *key exchange protocol* is a series of steps used by two or more parties in order to securely agree on a shared secret, such as a session key, in an unprotected network. A protocol that establishes a shared key between two entities is called a *two-party* key exchange protocol. Sometimes it's also useful to consider three parties, and thus the protocol is called a *tripartite* key exchange protocol. If a protocol has more than three participants, it is called a *group* or *conference* key exchange protocol. These kinds of protocols have a long history; the first known protocol was Diffie-Hellman [2, 3, and 4]. In 1976, Whitfield Diffie and Martin Hellman [5] proposed the earliest example of an asymmetric key establishment technique, but this protocol does not provide any authentication of parties or the exchanged information, the scheme is vulnerable to a *man-in-the-middle* attack.. Since then, many key exchange protocols have been proposed. In 2004, Popescu [6] proposed a protocol based on elliptic curve but this protocol does not meet key-compromise impersonation resilience. In 2005, He Ge [7] proposed a protocol based on hidden exponent RSA, but this protocol does not meet unknown key-share resilience. In 2005, Fuw-Yi Yang and Jinn-Ke Jan [8] proposed a protocol based on Diffie-Hellman key exchange called H-protocol. In 1992, A refinement and extension of encrypted key exchange scheme was proposed by Steiner et al. [9], which was extended to three-party. In 2005, Anish Mathuria and Vipul Jain [10] proposed some three party key exchange protocols using trusted server, but one of this protocols does not meet key confirmation then he proposed new protocol to solve this problem. In 2006, Wen, Lin and Hwang [11] proposed a protocol based on hybrid key architecture. A hybrid key architecture means that one entity (often a server) stores a pair of matching public/private keys while the other entity shares a secret with the server. This protocol does not meet forward secrecy. In 2006, Brita Vesteras [12] improve the security of Wen-Lin-Hwang's protocol.

The motivation: Two-party authenticated key exchange protocols where each pair of parties must share a secret with each other; a three-party protocol does not cause any key management problem for the parties.

The contribution: the proposed protocol is an extension from two parties to three parties. In this protocol each user exchange secret key with server then uses this secret key to exchange session key with each other. The efficiency and security of the proposed protocol are proven in this paper.

3. The Model Assumptions:

In this section, we precisely state the assumptions of the adversary and the communication models.

The Communication Model:

In this protocol, two parties, Alice and Bob connect to a server then Alice and Bob connect to each other. The three parties will then be connected on a private and authenticated channel.

The Adversary Model:

Assume a passive adversary, which means that this adversary can see and learn all information sent to or from the corrupted party without compromising the correct behavior of this party. The parties follow the execution steps of the protocol word for word but they are willing to learn any information leaked during execution. This commonly used security model is well-known as the honest-but-curious scenario.

4. The proposed Protocol:

In this section, the complete description of the proposed protocol is given. Alice and Bob want to agree on two session keys using trusted party (server). Alice and Bob share secret key with server then use this key to agree on two session keys.

Notations:

Descriptions for the notations used in this protocol are as follows:

U_A, U_B, U_S: The identity of Alice, Bob and Server.

S_A, S_B: master key that Alice &Bob stores.

pk_s, sk_s : A public/private key pair held by server

$E_{pk_s}(x)$: Encryption of x using the server's public key pk_S

$D_{sk_s}(x)$: Decryption of y using the server's private key sk_S

$f(), H()$: One-way hash functions

The Protocol:

From the beginning, Alice (U_A) and Bob (U_B) store their master keys S_A and S_B. The server U_s holds the private key pair pk_s, sk_s, and maintains a public table which contains all identities (like U_A, U_B) and their corresponding verifiers (like $f(U_A,S_A), f(U_B,S_B)$). The table record for clients U_A and U_B will be $(U_A, f(U_A,S_A))$, $(U_B, f(U_B,S_B))$. Alice and Bob select a random numbers r_a and r_a then they compute the cipher text from $y_a = E_{pk_S}(U_A,U_B,S_A,r_a)$ and $y_b = E_{pk_S}(U_A,U_B,S_B,r_b)$. Then they store (r_a, y_a) and (r_b, y_b).

Alice and Bob select k_a and k_b, then they compute ($n_a = y_a^{k_a} \bmod p$) and ($n_b = y_b^{k_b} \bmod p$). Alice sends y_a, n_a and Bob sends y_b, n_b to the server. The server decrypts y_a to obtain (U_A,U_B,S_A,r_a). The server then checks if $f(U_A,S_A)$ matches with the value in the table. The server decrypts y_b to obtain (U_A,U_B,S_B,r_b). The server then checks if $f(U_B,S_B)$ matches with the value in the table. If there is no match, the server terminates the protocol. If there is a match, the server then selects two random numbers r_s and k_s. The Server computes ($n_{s_a} = y_a^{k_s} \bmod p$) and ($n_{s_b} = y_b^{k_s} \bmod p$) then computes $Sk_a' = H((n_a)^{k_s}, r_a, r_s)$ and the server creates an authentication value $Auth_{s_a} = H(Sk_a', 2)$ with Alice. The server computes $Sk_b' = H((n_b)^{k_s}, r_b, r_s)$ and the server create an authentication value $Auth_{s_b} = H(Sk_b', 2)$ with Bob. Server sends $r_s, n_{s_a}, Auth_{s_a}$ to Alice and $r_s, n_{s_b}, Auth_{s_b}$ to Bob. Alice and Bob compute $Sk_a' = H((n_{s_a})^{k_a}, r_a, r_s)$ and $Sk_b' = H((n_{s_b})^{k_b}, r_b, r_s)$ Alice and Bob verify $Auth_{s_a}$ and $Auth_{s_b}$ then create an authentication value $Auth_a = H(Sk_a', 1)$ and $Auth_b = H(Sk_b', 1)$. Alice and Bob send $Auth_a$ and $Auth_b$ to server. Alice and Bob compute secret keys $Sk_a = H(Sk_a', 0)$ and $Sk_b = H(Sk_b', 0)$. Server verifies $Auth_a$ and $Auth_b$. If it is okay, the server computes the secret keys $Sk_a = H(Sk_a', 0)$ and $Sk_b = H(Sk_b', 0)$. Alice encrypts $(U_B, (n_{s_a})^{k_a}$ by secret key which computed between Alice and server and send it to the server. The server decrypts this cipher and encrypts $(U_A, (n_{s_a})^{k_a})$ by secret key which computed between Bob and server and send it and n_{s_a} to Bob. Bob decrypts this message and computes session key ($K_{AB} = (n_{s_a})^{k_a k_b} \bmod p$) then sends $(n_{s_a})^{k_b}$, $MAC_{K_{AB}}(n_{s_a})^{k_a}$ to Alice. Alice computes session key ($K_{AB} = (n_{s_a})^{k_a k_b} \bmod p$) then sends

$MAC_{K_{AB}}(n_{s_a})^{k_b}$ to Bob. Bob encrypts $(U_A,(n_{s_b})^{k_b})$ by secret key which computed between Bob and the server and send it to the server. The server decrypts this cipher and encrypts $(U_B,(n_{s_b})^{k_b})$ by secret key which computed between Alice and the server and send it and n_{s_b} to Alice. Alice decrypts this message and computes session key $(K_{BA}=(n_{s_b})^{k_b k_a} \bmod p)$ then sends $(n_{s_b})^{k_a}, MAC_{K_{BA}}(n_{s_b})^{k_b}$ to Bob. Bob computes session key $(K_{BA}=(n_{s_b})^{k_b k_a} \bmod p)$ then sends $MAC_{K_{BA}}(n_{s_a})^{k_a}$ to Alice.

Alice (S_A)	Server	Bob (S_B)
Select $r_a \rightarrow \{0,1\}^k$		Select $r_b \rightarrow \{0,1\}^k$
Compute $y_a = E_{pk_S}(U_A,U_B,S_A,r_a)$		Compute $y_b = E_{pk_S}(U_A,U_B,S_B,r_b)$
Select $k_a \rightarrow \{0,1\}^k$		Select $k_b \rightarrow \{0,1\}^k$
Compute $n_a = y_a^{k_a} \bmod p$		Compute $n_b = y_b^{k_b} \bmod p$
$\xrightarrow{y_a,\,n_a}$		$\xleftarrow{y_b,\,n_b}$

$D_{sk_S}(y_a)$

Check on $f(U_A,S_A)$

$D_{sk_S}(y_b)$

Check on $f(U_B,S_B)$

Select $r_s \rightarrow \{0,1\}^k$

Select $k_s \rightarrow \{0,1\}^k$

Compute $n_{s_a} = y_a^{k_s} \bmod p$

Compute $n_{s_b} = y_b^{k_s} \bmod p$

$Sk_a' = H((n_a)^{k_s}, r_a, r_s)$

$Auth_{s_a} = H(Sk_a',2)$

$Sk_b' = H((n_b)^{k_s}, r_b, r_s)$

$Auth_{s_b} = H(Sk_b',2)$

Alice		Bob
$\xleftarrow{r_s,\,n_{s_a},\,Auth_{s_a}}$		$\xrightarrow{r_s,\,n_{s_b},\,Auth_{s_b}}$
$Sk_a' = H((n_{s_a})^{k_a}, r_a, r_s)$		$Sk_b' = H((n_{s_b})^{k_b}, r_b, r_s)$
$H(Sk_a',2) =^? Auth_{s_a}$		$H(Sk_b',2) =^? Auth_{s_b}$
$Auth_a = H(Sk_a',1)$		$Auth_b = H(Sk_b',1)$
$\xrightarrow{Auth_a}$		$\xleftarrow{Auth_b}$

$H(Sk_a',1) =^? Auth_a$

$H(Sk_b^{'},1) =^{?} Auth_b$

$Sk_a = H(Sk_a^{'},0)$ $Sk_a = H(Sk_a^{'},0)$

$Sk_b = H(Sk_b^{'},0)$ $Sk_b = H(Sk_b^{'},0)$

$E_{Sk_a}(U_B,(n_{s_a})^{k_a})$

$\xrightarrow{E_{Sk_a}}$

$D_{Sk_a}(E_{Sk_a}(U_B,(n_{s_a})^{k_a}))$

$E_{Sk_b}(U_A,(n_{s_a})^{k_a})$

$\xrightarrow{E_{Sk_b},\, n_{s_a}}$

$D_{Sk_b}(E_{Sk_b}(U_A,(n_{s_a})^{k_a}))$

$K_{AB} = (n_{s_a})^{k_a k_b} \bmod p$

$K_{AB} = (y_a^{k_s})^{k_a k_b} \bmod p$

$\xleftarrow{(n_{s_a})^{k_b},\, MAC_{K_{AB}}(n_{s_a})^{k_a}}$

$K_{AB} = (n_{s_a})^{k_a k_b} \bmod p$

$K_{AB} = (y_a^{k_s})^{k_a k_b} \bmod p$

$\xrightarrow{MAC_{K_{AB}}(n_{s_a})^{k_b}}$ $E_{Sk_b}(U_A,(n_{s_b})^{k_b})$

$\xleftarrow{E_{Sk_b}}$

$D_{Sk_b}(E_{Sk_{ba}}(U_A,(n_{s_b})^{k_b}))$

$E_{Sk_a}(U_B,(n_{s_b})^{k_b})$

$\xleftarrow{E_{Sk_a},\, n_{s_b}}$

$D_{Sk_a}(E_{Sk_a}(U_B,(n_{s_b})^{k_b}))$

$K_{BA} = (n_{s_b})^{k_b k_a} \bmod p$

$K_{BA} = (y_b^{k_s})^{k_b k_a} \bmod p$

$\xrightarrow{(n_{s_b})^{k_a},\, MAC_{K_{BA}}(n_{s_b})^{k_b}}$

$K_{BA} = (n_{s_b})^{k_b k_a} \bmod p$

$\xleftarrow{MAC_{K_{BA}}(n_{s_b})^{k_a}}$ $K_{BA} = (y_b^{k_s})^{k_b k_a} \bmod p$

5. Security analysis:

Known key security:

The session key K_{AB}, K_{BA} is computed from $(y_a^{k_s})^{k_a k_b}$ and $(y_b^{k_s})^{k_b k_a}$. All the values of k_a, k_b and k_s change each session. This means that even if the session key is compromised, it will have no effect on other session keys. So the protocol meets the known key security goal.

Key-compromise impersonation resilience:

If the Alice's long-term private key S_A is compromised, an attacker can impersonate Alice and create the message y_a. This is because the encryption algorithm, the server's public key pk_s and U_A is publicly known. All the attacker then needs in order to create the message $y_a = E_{pk_S}(U_A, U_B, S_A, r_a)$ is a random value r_a'. But because the attacker does not know the server's secret key sk_S, he is not able to decrypt the value y_a and get the information he needs in order to compute the hash value $Sk_a' = H((n_{s_a})^{k_a}, r_a, r_s)$. He cannot get the correct value of r_a without decrypting y_a.

Now look at it in the other way, and assume that the server's private key sk_S is compromised. Then an attacker can decrypt the message y_a, y_b from Alice and Bob. He can then complete the protocol and create a secret key SK between Alice and the attacker. But he still cannot impersonate another client to the server. He needs to know one of the client's private keys s_x in order to do this. So the protocol meets the key compromise impersonation goal.

Forward secrecy:

In this protocol, the long-term private keys do no directly affect the session key. So if the attacker wants to learn a previous session key, he must drive k_a, k_b. These values transmit in discrete logarithm problem. So the protocol meets Forward secrecy goal.

Unknown key-share resilience:

Because of the Auth messages that the two parties exchange with server, they prove their identity to each other. As long as the server's private key sk_S is not compromised, only the server could decrypt y_a, y_b and get the r_a, r_b which it needs to create an $Auth_{s_a}$, $Auth_{s_b}$ value that the clients would accept. And still, as long as sk_S is not compromised, only the client who sent the first message will know the value of r_a and create an $Auth_a$ that the server would accept. So the protocol meets the unknown key-share resilience goal.

Key control:

The session keys $K_{AB} = (y_a^{k_s})^{k_a k_b}$ and $K_{BA} = (y_b^{k_s})^{k_b k_a}$ consist of three random values, one from each entity. Alice and Bob select k_a, k_b. In the same time. So, each of Alice and Bob can not control in their random values to result session keys as they want. So, the protocol meets key control goal.

Key confirmation:

Because both client and server verify each other's Auth values, they confirm that the other principal is computing the same secret key. Each party verify each other's MAC values, they confirm that the other principal is computing the same session key. Hence the protocol provides strong key confirmation.

The protocol's specific goal:

The goal of this protocol is to extend the two-party key exchange protocol into three-party key exchange and to achieve mutual authentication and secure communication. The two parties, Alice and Bob are authenticated by sending their IDs (U_A, U_B) and master keys (S_A, S_B), encrypted with the server's public key pk_S in $y_a = E_{pk_S}(U_A, U_B, S_A, r_a)$ and $y_b = E_{pk_S}(U_A, U_B, S_B, r_b)$. The server is authenticated by sending back $Auth_{sa}$=H (H $((n_a)^{ks}$, ra, r_s), 2) and $Auth_{sb}$=H (H $((n_b)^{ks}$, r_b, r_s), 2). Assume that the server's private key sk_S is kept secret, which it should be, only the server could decrypt y_a, y_b and retrieve the value r_a and r_b. Because of the Auth messages, Alice and server know that they are using the same values y_a, r_a and r_s. Bob and server know that they are using the same values y_b, r_b and r_s. Therefore, the protocol achieves mutual authentication and secure communication. The proposed protocol is secure in standard model.

6. Performance discussion:

In this section, examine the performance of the proposed protocol in terms of two perspectives: communication cost and on-line computation cost.

1: Communication cost:

Comparisons of communication cost in terms of round efficiency and message-transmitted size between the proposed protocol and the related schemes are given as follows:

A: Round efficiency: the proposed protocol only requires three rounds, which is less than it is required by other round-efficient 3PAKE schemes (related to table 1).

B: Message-transmitted size: Assume that the block size in secure secrete key cryptosystems is 128 bits, the output size in public key cryptosystems is 1024 bits, the output size of one-way hash functions is 128 bits. The transmitted message size of the proposed protocol is 128 * 4 + 1024 * 2 bits in Round 1. The cost is 128 * 8 + 128 * 2 bits in Round 2. In Round 3, the cost is 128 * 2 bits. Therefore, the total size of transmitted message in the proposed protocol is 4096 bits. From table 1, the proposed protocol has less message transmitted size than LSH and SCH protocol.

2: On-line computation cost:

From table 1, the proposed protocol required suitable modular exponentiation secret key en(de)cryption and public key en(de)cryption. LSH protocol and SCH protocol required secret key en(de)cryption and public key en(de)cryption more than required in the proposed protocol.

Table 1

	Communication cost		Computation cost				
	Round efficient	Message size	Random Number A/B/S	public key en(de)cryption A/B/S	secret key en(de)cryption A/B/S	Modular exponentiation A/B/S	Hash function A/B/S
The proposed protocol	**3**	**4096**	**2/2/2**	**1/1/0**	**1/1/2**	**3/3/2**	**3/3/6**
LSH [13]	**5**	**6400**	**2/3/0**	**1/1/2**	**2/2/2**	**2/2/0**	**1/1/0**
SCH [14]	**5**	**6016**	**1/1/3**	**1/1/2**	**2/2/0**	**2/2/4**	**0/0/0**

Table 1: comparison between the proposed protocol and 3PAKE schemes

The values of the random numbers have no effect on computation cost. The computation of hash functions has very light cost. Public and secret key encryption and exponentiation have a large computational cost. From table 1, the proposed protocol involves the fewest number of rounds than the other protocols. The proposed protocol has larger number of hash function than LSH and SCH protocols. However, LSH and SCH protocols have larger number of public and secret key encryption than the proposed protocol. Therefore, the proposed protocol has light total computation cost. This implies that our protocol is efficient and particularly suitable for resource-limited network environments, such as networks for mobile and wireless communication.

7. Conclusion:

In this paper, the proposed protocol is an extension of two-party key exchange protocol into three-party key exchange. The proposed protocol can fulfill the following security analysis: Known key security, Key-compromise impersonation resilience, Forward secrecy, Unknown key-share resilience, Key control and Key confirmation. Besides, compared with other schemes, the protocol not only needs fewer rounds to perform the protocol but also has considerably lower computational cost. In sum, this paper proposes more efficient and secure protocol.

Reference:

[1] "Key Exchange Protocols Properties" http://www.hsc.fr/ressources/presentations/ike/img5.htm

[2] D. A. Carts. "A Review of the Diffie-Hellman Algorithm and its Use in Secure Internet Protocols". http://www.sans.org/rr/whitepapers/vpns/751.php, 2001.

[3] RSA Security. What is Diffie-Hellman. http://www.rsasecurity.com/rsalabs/node.asp?id=2248.

[4] Wikipedia - Diffie-Hellman. http://en.wikipedia.org/wiki/Diffie-Hellman.

[5]. W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, Vol. 22, pp. 644-654, 1976.

[6] C. Popescu. "A Secure Authenticated Key Agreement Protocol". *Proceedings of the 12th*

IEEE Mediterranean Electrotechnical Conference, Dubrovnik, Croatia, pp. 783-786, May 2004.

[7] He. Ge and S. R. Tate. "Efficient authenticated key-exchange for devices with a trusted manager". Technical Report 2005- 02, University of North Texas, Computer Privacy and Security (CoPS) Lab, 2005.

[8] F. Y. Yang and J. K. Jan "An Enhanced and Secure Protocol for Authenticated Key Exchange" 2005.

[9] M. Steiner, G. Tsudik, M. Waidner,. "Refinement and extension of encrypted key exchange". Operating Systems Review 29 (3), 22–30, 1995.

[10] A. Mathuria and V. Jain "On Efficient Key Agreement Protocols" February 28, 2005

[11] H.-A. Wen, C.-L. Lin and T. Hwang. "Provably secure authenticated key exchange protocols for low power computing clients". *Computers & Security*, vol. 25, pp. 106-113, 2006.

[12] Brita Vesteras "Analysis of Key Agreement Protocols" Department of Computer

[13] Lin, C.L., Sun, H.M., Hwang, T., 2000. "Three-party encrypted key exchange: attacks and a solution". ACM Operating Systems Review 34, pp.12–20, 2006.

[14] H.M., Sun, B.C., Chen, Hwang, T. "Secure key agreement protocols for three-party against guessing attacks". The Journal of Systems and Software 75, pp.63–68, 2005.