Research Article

An Anonymous User Authentication with Key Agreement Scheme without Pairings for Multiserver Architecture Using SCPKs

Peng Jiang, Qiaoyan Wen, Wenmin Li, Zhengping Jin, and Hua Zhang

State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China

Correspondence should be addressed to Peng Jiang; jiangpengbanana@163.com

Received 5 April 2013; Accepted 30 April 2013

Academic Editors: G. A. Gravvanis and G. Wei

Copyright © 2013 Peng Jiang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With advancement of computer community and widespread dissemination of network applications, users generally need multiple servers to provide different services. Accordingly, the multiserver architecture has been prevalent, and designing a secure and efficient remote user authentication under multiserver architecture becomes a nontrivial challenge. In last decade, various remote user authentication protocols have been put forward to correspond to the multi-server scenario requirements. However, these schemes suffered from certain security problems or their cost consumption exceeded users’ own constrained ability. In this paper, we present an anonymous remote user authentication with key agreement scheme for multi-server architecture employing self-certified public keys without pairings. The proposed scheme can not only retain previous schemes’ advantages but also achieve user privacy concern. Moreover, our proposal can gain higher efficiency by removing the pairings operation compared with the related schemes. Through analysis and comparison with the related schemes, we can say that our proposal is in accordance with the scenario requirements and feasible to the multi-server architecture.

1. Introduction

In modern society, people’s life is highly dependent on the Internet, but the exposure of networks often causes great loss to users, which brings about that a secure user authentication mechanism has become the key issue to preserve valid remote clients in safety from being attacked. There is no doubt that the user authentication with smart card is one of the most widely used and the simplest approaches. When taking only one sort of service into account, some password authentication schemes for single-server environment have been proposed [1, 2].

Later with the rapid development of technology, different servers are needed to offer service via the network, and conventional methods need users to register with various servers repetitively and remember different identities and passwords. It is obvious that these traditional schemes make authentication inconvenient and cost much. Consequently, an appropriate multiserver user authentication mechanism has turned into a concern. In 2001, Li et al. [3] gave a remote user authentication scheme in neural networks for the first time, which opened up the gateway access to the multiserver architecture.

Considering the system environment without loss of generality, the multiserver architecture consists of multiple distributed service servers and remote clients with limited resource and capability. The service servers offer different access services such as e-commerce, online conference, network game, and remote medical system. If a remote client wants to access to these services, he/she needs to login these service servers through cellular network or wireless local area networks (WLANs).

Due to multiserver environment special characteristics and information security problem in public networks, designing a feasible user authentication scheme under multiserver architecture is a key issue, which can ensure the access of legitimate users and prevent invalid user from interfering with the service server. A practical user authentication
scheme under the multiserver environment must address the following requirements. They consist of both the previous criteria [1] and new user anonymity issue.

1. No repetitive registration is needed for the multiserver environments.
2. No verification table is stored in the server.
3. Mutual authentication and session key agreement can be achieved between the users and the service servers to carry on subsequent communications.
4. Various possible attacks can be resisted.
5. User can choose identity and password freely and change his/her password freely.
6. The computational and communication cost is low since the energy resources and computing capability of a smart card are limited.
7. The user is not allowed to expose his identity privacy information to eavesdroppers. Assume that the adversary obtains a valid user's identity, he/she can masquerade the user to enjoy the regular service without registration, which can cause losses for the valid user or even worse consequences. So the anonymous authentication should be implemented.

In order to satisfy all of these criteria, this paper proposes an anonymous remote user authentication scheme without pairings for multiserver architecture using self-certified public keys (SCPKs). We present public key-based user anonymous authentication scheme under the multiserver environment. Meanwhile, our proposal heightens efficiency increasingly accompanied by the removal of pairings operation; in contrast, the existing public key-based authentication schemes generally employ pairings function. Moreover, our proposal can avoid the server spoofing attack since the verification process relies on the server’s private key. Through security and performance analysis, our proposal not only achieves anonymous authentication with key agreement securely but also results more efficiently, remedying the weaknesses of previous authentication schemes which either encounter some attacks or fail to protect user privacy or cost relatively more energy. Compared with other related achievements, ours is more suitable for the remote user whose resources and capability are constrained under multiserver architecture.

The rest of this paper is organized as follows. Section 2 briefly describes some related works. Some preliminaries are given in Section 3. Our proposed secure and efficient user authentication scheme for multiserver architecture and corresponding analysis are presented in Sections 4 and 5, respectively. Finally, some conclusions are drawn in Section 6.

2. Related Work

Until now, two categories of improved multiserver user authentication schemes, hash-based authentication and public key based authentication, have emerged successively. To hash-based authentication, some user password authentication suggestions [4–7] based on static ID have been proposed to conquer the weaknesses of Li et al.’s, yet these were proven easy to be traced. In 2009, Liao and Wang [8] raised a dynamic identity authentication protocol for multiserver environment to advance previous work. In the following years, many researchers [9–12] have developed and enhanced the user authentication scheme step by step. To public key-based authentication, employing public key cryptosystem into the password authentication, Das et al. [13] first proposed a remote user authentication protocol with smart card using bilinear pairings. Yet theirs had an obvious disadvantage: no mutual authentication and key agreement. To improve the security, a series of user authentication schemes [14–16] with bilinear pairings have been presented. To improve the efficiency, Tseng et al. [17] gave a low-cost pairing-based user authentication protocol for wireless users and claimed that theirs was efficient, easy password changing, and suitable for multiserver environment in distributed networks. Unfortunately, in 2013, Liao and Hsiao [18] pointed out that Tseng et al.’s scheme also lacked mutual authentication with session key agreement, suffered from insider attack, password guessing attack, and replay attack, and advanced a pairings-based user authentication scheme using self-certified public keys. Liao and Hsiao claimed that their proposal could withstand various possible attacks and was well suited for multiserver environment.

Regrettfully, most of the existing related public key based authentication schemes under multiserver architecture mentioned previously did not pay attention to user anonymity issue. Moreover, their authentication schemes needed excessive energy consumption employing pairings operation and suffered from the server spoofing attack, which was not conducive to communication running and trapped in DoS attack easily.

3. Preliminaries

We now briefly review some basic concepts used in this paper, including bilinear pairings [19], related complexity assumptions [20], and self-certified public keys [21, 22].

3.1. Admissible Bilinear Pairing. Let \( \mathbb{G} \) be an additive group generated by \( P \) with prime order \( q \) and let \( \mathbb{G}_T \) be a multiplicative group of the same order. A map \( \hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T \) is said to be an admissible bilinear pairing if the following three conditions hold true.

1. Bilinearity: for all \( a, b \in \mathbb{Z}_q^* \), we have \( \hat{e}(aP, bP) = \hat{e}(P, P)^{ab} \).
2. Nondegeneracy: \( \hat{e}(P, P) \neq 1_{\mathbb{G}_T} \).
3. Computability: \( \hat{e} \) is efficiently computable.

We refer readers to [19] for more details of such pairings.

3.2. Complexity Assumption

1. Computational discrete logarithm (CDL) assumption: given \( Q = k \cdot P \), where \( P, Q \in \mathbb{G} \), there exists no probabilistic polynomial-time algorithm which can determine \( k \).
(2) Computational Diffie-Hellman (CDH) assumption: given two elements \(aP, bP\) in a group \(G\), where the unknown numbers \(a, b \in \mathbb{Z}_q^*\) are selected at random, there exists no probabilistic polynomial-time algorithm which can compute \(abP\).

(3) Elliptic curve factorization (ECF) assumption: given two elements \(P, Q\), where \(Q = aP + bP\) and \(a, b \in \mathbb{Z}_q^*\), there exists no probabilistic polynomial-time algorithm which can obtain \(aP\) and \(bP\).

3.3. Self-Certified Public Key. Here, we describe a self-certified public key process briefly; more details can be found in [21, 22].

(1) Initialization: given a group \(G\) on an elliptic curve \(E, P\) is a based point generator of prime order \(q\), the system authority (SA) selects a random value \(s \in \mathbb{Z}_q^*\) as its private key and computes the public key \(P_{pub} = s \cdot P\). Publish the related parameters and keep \(s\) secret.

(2) Partial private key and private key generation: the user \(U_i\) chooses a number \(k_i\) randomly, computes \(K_i = k_i \cdot P\), and sends \((ID_i, K_i)\) to SA over a secure channel. SA calculates \(W_i = K_i + w_i \cdot P\) as the witness using a random number \(w_i\). Then, SA computes the user's partial private key \(\bar{s}_i = H(ID_i || W_i) \cdot s + w_i\) and submits \((\bar{s}_i, W_i)\) to \(U_i\). \(U_i\) can obtain its private key \(s_i = \bar{s}_i + k_i\).

(3) Public key extraction: \(U_i\)'s public key can be computed by \(Pub_i = S_i \cdot P\). Any entity, who communicates with \(U_i\) and receives the witness \(W_i\), can authenticate \(U_i\)'s public key, as long as he/she calculates the equation: \(Pub_i = H(ID_i || W_i) \cdot P_{pub} + W_i\).

4. The Proposed Scheme

In this section, we propose an anonymous remote user authentication scheme for multiserver environment without pairings, which consists of five phases: server registration phase, user registration phase, login phase, verification phase, and password change phase. Three entities are involved: user \((U_i)\), service server \((S_j)\), and registration center \((RC)\). The system uses the private/public key pair \(s/P_{pub}\), where \(s\) is a random number in \(\mathbb{Z}_q^*\) and \(P_{pub} = s \cdot P\). Then publish the system parameters \(Params = \{G, P, q, P_{pub}, H(\cdot)\}\) and keep \(s\) secret. The notations used in this section are listed in Table 1. Some detailed steps will be described as follows and shown in Figure 1.

4.1. Server Registration Phase. When the service server wants to access to the multiserver architecture, it needs to register first. In this phase, RC uses the self-certified public key (SCPK) to generate the related credentials.

Step S1. \(S_j\) chooses a random value \(k_j \in \mathbb{Z}_q^*\), computes \(K_j = k_j \cdot P\), and sends \((SID_j, K_j)\) to RC.

Step S2. After receiving the message \((SID_j, K_j)\), RC generates a \(w_j \in \mathbb{Z}_q^*\) randomly, calculates \(W_j = K_j + w_j \cdot P\), \(\bar{s}_j = H(SID_j || W_j) \cdot s + w_j\), and issues \((W_j, \bar{s}_j)\) to \(S_j\).

Step S3. \(S_j\) can obtain its private key with \(s_j = \bar{s}_j + k_j\) and verify the validity of the message by computing \(Pub_j = s_j \cdot P = H(SID_j || W_j) \cdot P_{pub} + W_j\).

If the equation holds, the issued values are valid, and vice versa.

4.2. User Registration Phase. Supposing that the user \(U_i\) wants to get service granted only from \(S_j\), he/she needs to register to the same RC that \(S_j\) did, by submitting his identity \(ID_i\) and password \(PW_i\) to RC. Then, RC returns the smart card back to \(U_i\). The communication between \(U_i\) and RC is through a secure channel. The steps are performed as follows.

Step U1. \(U_i\) freely chooses a password \(PW_i\) and a random number \(b_i\) to compute \(A_i = H(PW_i || b_i)\) and \(I_i = H(ID_i || b_i)\). Then, \(U_i\) submits \((ID_i, A_i, I_i)\) to RC for user registration via a secure channel.

Step U2. RC calculates

\[
\begin{align*}
B_i &= s \cdot H(ID_i || I_i) + x, \\
C_i &= B_i \oplus H(A_i), \\
D_i &= B_i \oplus H(ID_i) \oplus A_i, \\
X &= x \cdot P, \\
E_i &= H(ID_i || A_i) \cdot P - X,
\end{align*}
\]

stores \((C_i, D_i, E_i, H(\cdot))\) in \(U_i\)'s smart card, and submits it to \(U_i\). Then \(U_i\) keys \(b_i\) into the smart card.

4.3. Login Phase. When \(U_i\) wants to login to the server \(S_j\), he/she first inserts his/her own smart card to a card reader and then inputs the identity \(ID_i\) and password \(PW_i\). The login details with respect to this smart card are as follows.

Step L1. The smart card computes \(A_i' = H(PW_i || b_i), B_i' = D_i \oplus H(ID_i) \oplus A_i', \) and \(C_i' = B_i' \oplus H(A_i')\) and checks whether
Figure 1: The proposed scheme.

C\text{\textprime}_i = C_i. If the answer is yes, it means that the smart card matches to \text{} U_i.

Step L2. The smart card generates a random value \( r_j \in \mathbb{Z}_q \) and computes
\[
\begin{align*}
W_j &= H(\text{ID}_j \parallel I_j), \\
\text{CID}_i &= W_i \oplus B_i, \\
R_i &= r_i \cdot P.
\end{align*}
\]

Step L3. The smart card submits the login request message (\text{CID}_i, R_i) to S_j over a public channel.

4.4. Verification Phase. After receiving the login request message from U_i, S_j performs the following tasks to authenticate the user.

Step V1. S_j checks whether \text{CID}_i conforms to the fixed format. If the format is wrong, S_j outputs the reject message; otherwise it calculates
\[
\begin{align*}
K_{ji} &= s_j \cdot R_j, \\
R_j &= r_j \cdot P,
\end{align*}
\]

\( T_{ji} = r_j \cdot R_j \),

\( M_j = H(\text{CID}_i \parallel K_{ji} \parallel R_i \parallel R_j) \),

where \( r_j \) is a random value, chosen by S_j. Then S_j sends (\text{SID}_j, W_j, R_j, M_j) to U_j.

Step V2. Receiving the message (W_j, R_j, M_j), U_i first verifies the public key of S_j by the equation Pub_j = \( H(\text{SID}_j \parallel W_j) \cdot P_{\text{pub}} + W_j \). Only under the case the equation holds, U_i continues to calculate K_{ji} = r_j \cdot Pub_j, T_{ji} = r_j \cdot R_j. Then U_i needs to check whether \( M_j = H(\text{CID}_j \parallel K_{ji} \parallel R_i \parallel R_j) \). When the verification can pass, U_i authenticates S_j and computes
\[
\begin{align*}
P_{ij} &= B_i \oplus H(K_{ji} \parallel T_{ji}), \\
X &= H(\text{ID}_i \parallel A_i) \cdot P - E_i.
\end{align*}
\]

Then U_i transmits (P_{ij}, X) to S_j. 

Step V3. Next, S_j undoes \( B_i = P_{ij} \oplus H(K_{ji} \parallel T_{ji}), W_i' = \text{CID}_i \oplus B_i' \) and examines \( X = B_i' \cdot P - W_i' \cdot P_{\text{pub}} \). If it is not the case,
Scheme Computational cost

5. Analysis of Our Scheme

In this section, we first analyze the functionality features of our proposed scheme based on the requirements of the remote user authentication for multiserver architecture, which have been presented in Section I. Then we evaluate the performance of the proposed scheme and make comparisons with some related works [8, 9, 11, 12, 17, 18].

5.1. No Repetitive Registration. In our scheme, before the user wants to login to the server under multiserver environment, they must run the user registration with his/her information to the registration center. Then, the user can access to all the service without submitting registration request once again.

5.2. No Verification Table. Throughout the protocol process, it is not difficult to find that RC and S_j have no need to maintain any verification or password table, which can cost much and whose leakage may causes serious disruption. Meanwhile, our scheme does not need to store the user’s password or public key with certificate, too.

5.3. Mutual Authentication with Session Key Agreement. In the verification phase of the proposed scheme, the service server S_j can authenticate the validity of U_i by checking if X = B'_j ⋅ P − W'_j ⋅ P_pub holds. U_i can verify the public key of S_j Pub_j = H(SID_j || W_j) ⋅ P_pub + W_j with W'_j to confirm that S_j is the objective service server; meanwhile check the equation M'_j = M_j to affirm that the login message is received by S_j. Only when all previous equations are satisfied, the session continues and the communication parties agree on a shared session key SK = H(W'_j || K_j || T_j). For the aforementioned analysis, our scheme can achieve mutual authentication with session key agreement.

5.4. No Synchronization Clock. In our scheme, both the user and the service server employ the random points R_i, R_j to interactive with each other. The timestamp does not appear in the proposed scheme; therefore the synchronization clock problem can also be abstained in the session key.

5.5. Anonymity. In the user registration phase, the identity of the remote user can be protected from disclosure by the secure channel between U_i and RC. In the login and authentication phase, U_i’s identity is submitted with CID_i substituting ID_i, nobody can learn the user’s real identity, and S_j can only verify the user’s validity cannot obtain the real ID_i with the received message. To general adversary, he/she can extract the smart card and intercept the login message, but he can do nothing to crack the user’s identity due to the resistance to collision of the hash function. Therefore, we claim that our scheme can provide the user anonymity.

5.6. Security of the Session Key

(1) Perfect Forward Secrecy and Backward Secrecy. In this scheme, the session key is established by W'_j, K_ij, T_ij, where K_ij and T_ij rely on the random values r_i and r_j, r_i and r_j are independently generated in each session, and are changed for each authentication phase and are not correlated. The adversary cannot use current session key to derive forward and backward session key. Hence, we claim that our scheme achieves perfect forward secrecy and backward secrecy.

(2) Known Session Key Security. In this scheme, the session key SK = H(W'_j || K_j || T_j) is composed of W'_j, K_ij and T_ij. Assume that the adversary can seize a session key SK, he cannot obtain the parameters W'_j, K_ij, and T_ij attributed to the one-way hash function H(⋅). Since K_ij and T_ij consist of R_i, R_j, which are
independent for each session, no session keys rely on each other. Furthermore, though the adversary can intercept the current transmitted message $R''_mx, R''_nx$, he cannot compute the new session key $SK_{mn}$'s components $K''_{mn}$ without the server's private key or $T''_{mn}$ due to the CDH problem's difficulty.

(3) No Key Control. In this scheme, the session key consists of $W_i, K_{ij}, T_{ij}$, where partial parameters $K_{ij}, T_{ij}$ are generated by Diffie-Hellman key exchange form; thereby the fairness of the session key can be guaranteed. More specifically, $K_{ij} = s_j \cdot R_i = r_j \cdot \text{Pub}_j, T_{ij} = r_j \cdot R_i = r_j \cdot R_i, R_i$ and $R_j$ are respectively provided by the user and the server; therefore either party is in vain attempting to preselect or control the session key.

5.7. Various Common Attacks. Our proposed remote user authentication scheme for multiserver architecture cannot only meet the previous security features, but also be against various known attacks, such as impersonation attack, and stolen smart card attack. We will discuss the following extra four attacks, the others can refer to [11, 18].

(1) Impersonation Attack. If an adversary tries to impersonate as a legitimate user to log into the server, he/she must first forge a valid login request message (CID, $R_i$). However, the adversary cannot compute a new and legal login message without knowing ID, or $B_i$. Suppose that the adversary can steal the smart card of the user $U_i$ by virtue of some approaches, he is still unable to calculate $B_i$ for the reason that he has no information about $A_i$ and ID. Moreover, even if the adversary utilizes (CID, $R_i$) to log into $S_j$, he cannot pass the verification $X = B_i^T \cdot P - W_i^T \cdot P_{pub}$, because he is unable to provide correct $P_{pub}$ without $B_i$ or $K_{ij}$. The adversary cannot obtain the valid session key. Under the situation, our proposed scheme can withstand the impersonation attack.

(2) Stolen Smart Card Attack. We assume that $U_i$'s smart card is stolen or lost; the adversary picks it and has the ability to breach the information stored in the smart card $(C_i, D_i, E_i, H(\cdot), b_i)$. Yet on the one hand, it is impossible to guess $A_i$ and ID, correctly at the same time, on the other hand, $s$ and $x$ are, respectively, private key and secret value of $RC$, so the adversary cannot derive $B_i$. Consequently, the adversary cannot fabricate a valid login message or compute the session key. That is the reason that our proposed protocol is secure against the stolen smart card attack.

(3) Off-Line Password Guessing Attack. Assume that the adversary guesses a password $PW'$ from the dictionary; he can compute $A_i = H(PW' || b_i), B_i = C_i \oplus H(A_i)$ but fails to calculate other information without ID, or $K_{ij}$. The adversary cannot examine whether the guessed password $PW'$ is correct without comparing parameters. Hence, the adversary can extract the smart card information and intercept the transmitted message in public channel, but our proposed scheme can resist the off-line password guessing attack.

(4) Man-in-the-Middle Attack. When an adversary wants to perform the man-in-the-middle attack, he can intercept the login message, communicate, and share the session key with the server. In the proposed scheme, even if the adversary gets the message in public channel, he cannot calculate $W_i, K_{ij}, or T_{ij}$ without ID or other random values $r_j, r_j$. Consequently, our scheme can resist the man-in-the-middle attack.

(5) Server Spoofing Attack. When a valid but malicious server $S_m$ wants to cheat $U_i$ on behalf of $S_j$ and obtain the session key, he needs to know both the witness and private key of $S_j$. In our scheme, $S_m$ cannot provide the correct witness, and the user $U_i$ cannot pass the server's public key verification. Even if $S_m$ intercepts $W_j$, he cannot check the equation $X = B_i^T \cdot P - W_i^T \cdot P_{pub}$ since he does not obtain $K_{ij}$ without knowing the private key $s_j$. Finally, the adversary fails to share the session key with the user $U_i$. Therefore, our scheme can resist the server spoofing attack.

5.8. Local Password Verification. In our scheme, $U_i$ can account whether the used smart card matches with himself by checking $C_i' = C_i$ before logging into $S_j$, and thus accomplish the user password verification locally. Through the previous equation, $U_i$ can avoid network resource wasting caused by wrong password. Because until the authentication phase $S_j$ can authenticate user's validity and password appropriateness; in other words, wrong password cannot be detected until the authentication phase. Therefore, our scheme can achieve local password verification.

At last, the functionality comparisons among our and other previously proposed schemes, such as [8, 9, 11, 12, 17, 18], are listed in Table 2. In particular, we can clearly see that the other schemes do not assist in the impersonation attack except our proposed scheme. Thus, it is obvious that our proposed scheme is superior to the others in accordance with all of essential comparative items. In addition, unlike the other related public key-based multiserver authentication schemes [17, 18], ours can achieve the user anonymity and local password verification. On the whole, our proposal is the only one that can satisfy all the functionalities for the multiserver architecture.

5.9. Performance. Under multiserver architecture, the computational cost is a key issue to evaluate whether a remote user authentication scheme is efficient because of mobile devices’ constrained resources and computing capability. Before analyzing the computational cost of each phase, define some notations and equivalence relationship first:

(i) $T_m$: the time to compute a bilinear pairing map;
(ii) $T_M$: the time to compute a point multiplication on the elliptic curve group;
(iii) $T_A$: the time to compute a point addition on the elliptic curve group;
The XOR operation, modular multiplication, and modular addition operation are negligible during evaluating the performance. In the following, we will give the computational cost of five phases individually. In the server registration phase, the computational cost is $5T_M + 4T_A + 2T_H$. The user registration phase consumes $2T_M + 2T_A + 6T_H$. When the user logs into the server, it costs $T_M + 4T_H$. During verification of each other between the server and the user, $9T_M + 3T_A + 6T_H$ is demanded. The computational cost of the password change phase is $2T_M + 2T_A + 8T_H$. The detailed cost comparisons with the related authentication schemes [17, 18] are illustrated in Table 3. At the same time, we show the implementation result in Figure 2, which can show the computational cost contrast more intuitively. Table 3 and Figure 2 can clearly indicate that our proposal needs no pairing operation, while [18] contains $4T_c$ and [17] contains $2T_c$. Because the relative computational cost of a pairing is approximately 20 times higher than that of the point multiplication over elliptic curve group, we can find that the computational cost of ours is obviously much less than that of others by removing pairing operation.

From Tables 2 and 3, we can make a conclusion that our remote authentication scheme has more security features and lower computational cost among the existing related works, which satisfies the requirements for the multiserver architecture.

6. Conclusions

An anonymous and efficient remote user authentication scheme for the multiserver architecture is proposed in this paper and the self-certified public keys are employed. Our scheme can satisfy all of the requirements needed for achieving secure authentication in multiserver environments, as compared with the previously proposed schemes. Moreover, the proposal succeeds to both achieve the user’s identity anonymity and remove the pairing operation, which makes that the proposed scheme can provide more advantages and be more practical for the actual applications. Additionally, we analyze the security and performance of our proposal and make comparisons with other related works. From these analysis and comparisons, we can reach a conclusion that our proposed scheme owns more functionalities and attains higher efficiency.

Acknowledgments

This work is supported by NSFC (Grant nos. 61272057, 61202434, 61170270, 61100203, 61003286, and 61121061) and the Fundamental Research Funds for the Central Universities (Grant nos. 2012RC0612 and 2011YB01).
References


Submit your manuscripts at
http://www.hindawi.com