http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (24): 8269-8274, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.8269.8274

RFID Tag Ownership Transfer Protocol with Backtracking Authorization Ability

He Lei, Gan Yong and Yin Yi-feng
School of Computer and Communication Engineering,
Zhengzhou University of Light Industry, Zhengzhou, China

Abstract: It was proposed an ownership transfer protocol with backtracking authorization ability to transfer tag ownership among different owners. The protocol is divided into two sub-protocols, ownership transfer sub-protocol and backtracking authorization sub-protocol. The former provides mutual authentication and key update to transfer ownership among different owners. The latter provides limited authorization of tag to old owner. The security of our protocol was analyzed in GNY logic. The result indicates it has better security properties. It protects forward security and backward security of secrets stored in the tag. It resists against replay attack, man-in-middle attack and desynchronization attack. The ownership transfer sub-protocol was simulated and implemented. The experimental data shows the protocol is suitable for low-cost RFID tags.

Key words: Ownership transfer, RFID, authentication, backtracking authorization, GNY logic

INTRODUCTION

Radio Frequency Identification (RFID) is an automatic identification technology. It has been used in many fields, such as logistics management, access control, etc. Typically, RFID system consists of tag, reader and backend database. A tag is mainly made up of antenna and integrated circuit. A reader is responsible for communication with tag and backend database. A backend database stores the information about tag and provides some services, for example, authentication service (Li *et al*., 1990; Kapoor and Piramuthu, 2012).

However, RFID brings new security issues because it extends the security boundary. It is difficult to design security protocol of RFID system because tag is an extremely resource-constrained device. Many cryptography algorithms can not be implemented because of the limited memory and computation resource of tag. Researchers generally accepts that only some lightweight functions or operations can be executed, such as Hash, XOR operation, cyclic shift, etc.

In the logistics process, objects tagged may be owned by different entities. When the object is handed to a new owner, the ownership of tag attached also needs to be transfered to the new owner accordingly. It is important to guarantee the security of tag Ownership Transfer (OT) procedure (Molnar *et al*., 2006; Saito *et al*., 2005; Song, 2008) To secure ownership transfer, we should ensure that the Old Owner (OO) is not able to interrogate the tag any longer and the tag establishes new shared secrets with New Owner (NO). The old owner updates its secrets shared with the tag and sends them to the new owner. The new owner also updates the secrets received from old owner and establishes new shared secrets with the tag. During the procedure, it should provide authentication between owners and the tag and be resistant to man-in-middle attack, replay attack, desynchronization attack, etc. Moreover, it should protect forward security and backward security of the secrets. The former means that even if the new owner obtains the secrets updated, it still won't infer the secrets shared by the old owner and the tag. The latter means that the old owner won't infer the secrets shared by the new owner and the tag even if it obtains the secret before updating (Wang, 2011).

In this study, we propose an ownership transfer protocol for RFID tag. It meets the security requirements mentioned above (Zhou *et al*., 2011). Besides, in some cases, such as after-sale and maintenance service, the old owner needs to temporally recover the ownership. Our protocol can provide backtracking authorization capability to meet the requirement.

This study is organized as follows. The following section provides a brief overview on some related protocols. Afterwards, we propose a tag ownership transfer protocol with backtracking authorization ability and present a brief security analysis of our protocol by using GNY logic. We implement and simulate our protocol in Linux and obtain experimental data. The last section contains conclusions and suggestions for the future work.

Corresponding Author: He Lei, School of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou, China

RELATED WORK

One of the earlier research results in the field of RFID tag ownership transfer protocol is the pseudonym protocol enabling ownership transfer of RFID tags proposed by Molnar *et al*. In this protocol, new owner increases the counter value to prevent old owner from accessing tag. Another earlier research result is the reassignment scheme of an RFID tag's key for ownership transfer proposed by Saito *et al*., 2005. In this protocol, it proposed two ownership transfer protocols. One involves Trusted Third Party (TTP), the other doesn't involve TTP. These two protocols don't provide forward security of secrets stored in tag.

Song proposed an ownership transfer protocol which consists of two sub-protocols. Wang analyzed the ownership transfer protocol and found it does not provide froward security.

Kulseng *et al*. (2010) proposed two ownership transfer protocols. One contains TTP, the other doesn't contain TTP. The former does not specify how to resist desynchronization attack when tag and TTP update PIN. The latter is vulnerable to tracking attack.

Zhou *et al*. proposed an ownership transfer protocol in supply chain. It includes old owner, new owner, tag, TTP and third party logistics. We think it is vulnerable to desynchronization attack.

Kapoor and Piramuthu proposed two protocols. These two protocols implement symmetric key cryptography to encrypt confidential information. New owner obtains ownership by negotiating key with tag.

Protocol description: Our protocol is divided into two sub-protocols, namely ownership transfer sub-protocol and backtracking authorization sub-protocol. The former is used to transfer the ownership of tag from old owner to new owner. The latter provides old owner with temporary access to tag. In order to facilitate the research, we assume the channel between tag and reader is not secure, while other channel is secure. The notations in Table 1 are used throughout the study.

Ownership transfer sub-protocol: The ownership transfer sub-protocol contains authentication phase and key update phase. Old owner will update secrets shared by itself and tag when it verifies tag's identification. Afterwards, it sends the information updated to new owner to protect forward security. New owner also authenticates the tag and updates key in the same way after receiving the message to protect backward security. The protocol is illustrated as Fig. 1:

Table 1: Notations

Notation	Meaning
k	The current key shared by owner and tag
kold	The latest successful authentication key whose initial value is k
r_i	I-th random number
a, b	Concatenation of message a and b
H(a)	One way hash function of message a
"x"	A string x
□	XOR operation
Max	Maximum access time
k_{temp}	Temporary key
c	The value of counter in the tag

- Old owner generates a random number r_1 and sends Ownership Transfer Request (OTR) and r_1 to tag through reader
- Tag generates a random number r_2 and sends { r_2, H(k, r_1, r_2, OTR)} to reader. Reader forwards the message received with {OTR, r_1} to old owner
- Backend database of old owner performs an exhaustive search among all items to find an appropriate k or k_{old} which meets the requirement H (k, r_1, r_2, OTR). If no match is found, the protocol is stopped. If one match is found, namely k or k_{old}, old owner believes the tag authentic and performs different operations according to different situation.
- If there is a k meets the requirement, it proves that the keys respectively stored in tag and backend database are synchronous. It generates a new key and updates k_{old} = k, k =k_{new}
- If there is a k_{old} meets the requirement, it is possible to suffer desynchronization attack. It keeps k_{old} unchanged, generates a new key, k_{new} and updates k = k_{new}

After updating the key, old owner sends {status, H(status, r_1, r_2, k_{old})$\oplus k_{new}$, H(status, r_2, r_1, k_{old}, k_{new})} to tag in which the value of status is 00. Old owner appends its identification to the owner list which contains the identification of all owners. It sends k and owner list to the new owner in a secure channel.

The tag computes H(status, r_1, r_2, k_{old}) by using its key as k_{old} and further infers k_{new}. It verifies whether the k_{new} computed is correct according to the H (status, r_2, r_1, k_{old}, k_{new}) received. If it is correct, the tag updates k_{old} = k, k= k_{new} and sends H("update success", r_1, r_2, k_{old}, k) to old owner through reader in which "update success" is a string.

If the new owner receives the message sent by the old owner, it will obtain the key k updated and set k_{old} = k. The new owner performs the sub-protocol again to protect the backward security of tag's information.

Backtracking authorization sub-protocol: In the lifetime of tag, old owner of tag needs to temporarily recover

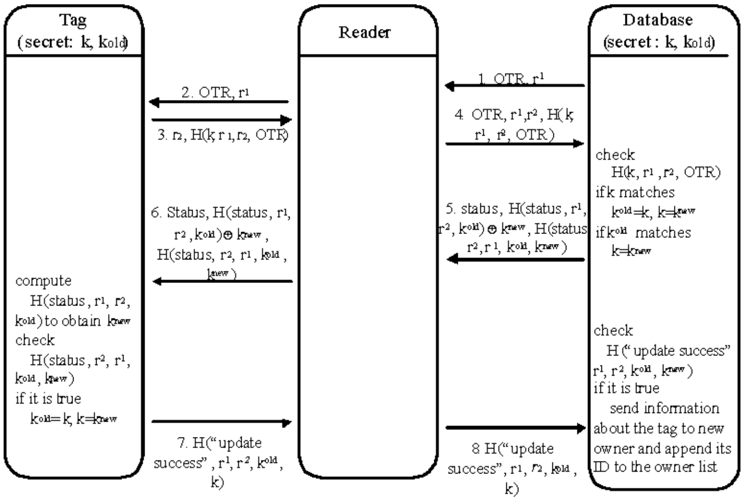
Tag
(secret: k, kold)
Reader
Database
(secret : k, kold)
1. OTR, r1
2. OTR, r1
3. r2, H(k, r1, r2, OTR)
4. OTR, r1, r2, H(k, r1, r2, OTR)
check
H(k, r1, r2, OTR)
if k matches
kold=k, k=knew
if kold matches
k=knew
5. status, H(status, r1, r2, kold) ⊕ knew, H(status r2, r1, kold, knew)
6. Status, H(status, r1, r2, kold) ⊕ knew, H(status, r2, r1, kold, knew)
compute
H(status, r1, r2, kold) to obtain knew
check
H(status, r2, r1, kold, knew)
if it is true
kold=k, k=knew
7. H("update success", r1, r2, kold, k)
8 H("update success", r1, r2, kold, k)
check
H (" update success" r1, r2, kold, knew)
if it is true
send information about the tag to new owner and append its ID to the owner list

Fig. 1: Ownership transfer sub-protocol

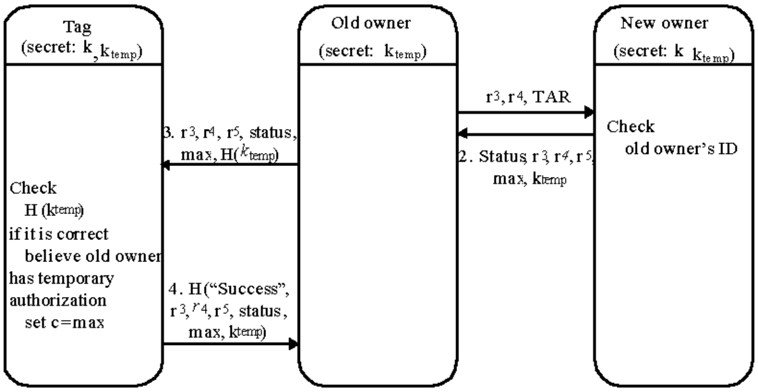
Tag
(secret: k, ktemp)
Old owner
(secret: ktemp)
New owner
(secret: k ktemp)
r3, r4, TAR
Check
old owner's ID
2. Status, r3, r4, r5, max, ktemp
3. r3, r4, r5, status, max, H(ktemp)
Check
H (ktemp)
if it is correct
believe old owner has temporary authorization
set c=max
4. H("Success", r3, r4, r5, status, max, ktemp)

Fig. 2: Backtracking authorization sub-protocol

access to tag for some special purposes, such as after-sales service. If the current owner, i.e., the new owner mentioned above agrees, it will give temporary access permissions of tag to the old owner. Moreover, it sets a maximum access times, max, for old owner and tag to guarantee the authorization will automatically expire. The protocol is shown in Fig. 2:

- Old owner will receive the random number r_3 generated by tag in its response when it initiates an access. It sends r_3, a random number r_4 generated by itself and a Temporary Authorization Request(TAR) to the new owner, namely $\{r_3, r_4, TAR\}$
- The new owner verifies whether the old owner's identification exists in the owner list. If it does and the new owner grants to authorization, the new owner will set max and generate a random number r_5. It sends {status, r_3, r_4, r_5, max, H(k, r_3, r_4, r_5, status, max)} to the old owner in a secure manner where status is 01
- The old owner sets H(k, r_3, r_4, r_5, status, max) as k_{temp} granted by the new owner. It sends {r_3, r_4, r_5, status, max, H(H(k, r_3, r_4, r_5, status, max))} to tag
- The tag knows old owner wants to communication with it by checking the value of status. It verifies whether H(H(k, r_3, r_4, r_5, status, max)) received is

correct. If it is correct, the tag believes the new owner grants the temporary authorization to the old owner. H(k, r_3, r_4, r_5, status, max), namely k_{temp}, is the temporary key shared by the old owner and tag. It sets c= max and sends H("Success", r_3, r_4, r_5, status, max, H(k, r_3, r_4 r_5 status, max)) to the old owner

- The old owner can commurication with tag when it verifies the message received. After each communication, c decreases by 1, until to 0. When c is zero, the tag sends status whose value is 10 to the old owner. The old owner needs to reapply for authorization from the new owner, or it abandons the commurication with the tag

Protocol security analysis: GNY logic is a logic method proposed by Li *et al.* to analyze the security of protocol. In this study, we use GNY logic to briefly analyze our protocol based on formal description and initialization assumptions. In order to facilitate the analysis, it is assumed that only the channel between tag and reader is not secure, while other channels are secure. Expressions and inference rules we used comply with the study achieved by Li *et al.*

Analysis for ownership transfer sub-protocol: There is a key update procedure in the ownership transfer sub-protocol. In order to maintain consistency in the inference process, we use k to represent the key before updating and k_{new} to represent the key updated. We proceed by formal description of messages in the protocol, followed by explicit initialization assumptions, the goals and inference process:

1. Formal description of protocol messages

OTM1: $T \triangleleft *OTR, *\ r_1$

OTM2: $OO \triangleleft OTR, r_1, *r_2, *H(k, r_1, r_2, OTR)$

OTM3: $T \triangleleft *status, *\{k_{new}\} \sim> OO \ni k, *H(status, r_2, r_1, k, k_{new}) \sim> OO|\equiv T \overset{k_{new}}{\longleftrightarrow} OO$

OTM4: $OO \triangleleft *\ H(\text{“update success”}, r_1, r_2, k, k_{new}) \sim> T|\equiv T \overset{k_{new}}{\longleftrightarrow} OO$

2. Initial assumptions

OTA1: $OO \ni k$

OTA2: $OO \ni r_1$

OTA3: $OO|\equiv \# r_1$

OTA4: $OO|\equiv T \overset{k_{new}}{\longleftrightarrow} OO$

OTA5: $T \ni k$

OTA6: $T|\equiv \# r_2$

OTA7: $T|\equiv \phi(k_{new})$

OTA8: $T \ni r_2$

OTA9: $T|\equiv OO|\Rightarrow OO|\equiv *$

OTA10: $T|\equiv OO|\Rightarrow OO \ni k$

OTA11: $T|\equiv T \overset{k_{new}}{\longleftrightarrow} OO$

OTA12: $T|\equiv OO|\Rightarrow T \overset{k_{new}}{\longleftrightarrow} OO$

OTA13: $OO|\equiv T|\Rightarrow T|\equiv *$

3. Security objectives and inference process

OTG1: $OO|\equiv T \ni k$ (OTM2, OTA1, OTA2, T1, P1, OTA3, F1, OTA4, I3, I6, P3)

OTG2: $T \ni k_{new}$ (OTM1, OTM3, T1, P1, OTA5, OTA8, P4, P6)

OTG3: $T|\equiv \# k_{new}$ (OTA7, R2, F10, F7)

OTG4: $T|\equiv OO \ni k$(OTA6, F1, OTA11, OTA7, I1, OTG3, F2, OTA9, J2, OTA10, J1)

OTG5: $T|\equiv T \overset{k_{new}}{\longleftrightarrow} OO$(OTM1, T1, P1, OTA5, OTM3, OTA8, OTG2, P2, OTA6, F1, F10, OTA11, I3, OTA9, J2, J3, OTA12, J1)

OTG6: $T|\equiv OO \ni k_{new}$ (I6, P3)

OTG7: $OO|\equiv T \ni k_{new}$ (OTM4, OTG2, P2, OTA3, F1, OTM4, OTA4, I3, I6, P3)

OTG8: $OO|\equiv T|\equiv T \overset{k_{new}}{\longleftrightarrow} OO$ (F10, OTA13, OTM4, J3)

From the above analysis we find that old owner and tag perform mutual authentication. Both of them have the key updated and believe it is the new key shared by them. New owner will perform authentication and key update procedure again in the same way when it receives the information about the tag from old owner. Therefore, the inference procedure of security objectives is similar with the procedure mentioned above.

Analysis for backtracking authorization sub-protocol: We use k_{temp} to replace H(k, r_3, r_4, r_5, status, max) because it is in the role of temporary key in this protocol:

1. Formal description of protocol messages

BM1: $NO \triangleleft *\ r_3, *\ r_4, *TAR$

BM2: $OO \triangleleft *status, *\ r_3, r_4, *\ r_5, *\ max, *\ H(k, r_3, r_4, r_5, status, max)$

BM3: $T \triangleleft r_3, *\ r_4, *\ r_5, *status, *\ max, *H(H(k, r_3, r_4, r_5, status, max)) \sim> (NO|\equiv T \overset{k_{new}}{\longleftrightarrow} OO, OO \ni H(k, r_3, r_4, r_5, status, max))$

BM4: $OO \triangleleft *H(\text{“Success”}, r_3, r_4, r_5, status, max, H(k, r_3, r_4, r_5, status, max))$

2. Initial assumptions∋

BA1: $T \ni k$

BA2: $T|\equiv \# r_3$

BA3: $OO|\equiv NO|\Rightarrow \# k_{temp}$

BA4: $OO|\equiv NO|\equiv \# k_{temp}$

BA5: $T|\equiv T \overset{k}{\longleftrightarrow} NO$

BA6: $T|\equiv NO|\Rightarrow NO|\equiv *$

BA7: $T|\equiv NO|\Rightarrow T \overset{k_{temp}}{\longleftrightarrow} OO$

BA8: $T|\equiv NO|\Rightarrow OO \ni k_{temp}$

BA9: $OO|\equiv NO|\equiv T \overset{k_{temp}}{\longleftrightarrow} NO$

BA10: $OO|\equiv NO|\Rightarrow T \overset{k_{temp}}{\longleftrightarrow} OO$

3. Security objectives and inference process

BG1: $OO \ni max$ (BM2, T1, P1)

BG2: $OO \ni k_{temp}$ (BM2, T1, P1)

BG3: $T|\equiv \# k_{temp}$ (BM3, BA1, T1, P1, BA2, F1, F10)

BG4: $OO|\equiv \# k_{temp}$ (BA3, BA4, J1)

BG5: $T|\equiv NO|\sim H(k, r_3, r_4, r_5, status, max)$ (BM3, T1, P1, BA5, I3)

BG6: $T|\equiv NO|\sim max$ (BG5, I7)

BG7: $T|\equiv T \overset{k_{temp}}{\longleftrightarrow} OO$ (BA6, J2, J3, BA7, J1)

BG8: $T|\equiv OO \ni k_{temp}$ (BA8, J1)

BG9: $OO|\equiv T \overset{k_{temp}}{\longleftrightarrow} OO$ (BA9, BA10, J1)

BG10: $T \ni max$ (BM3, T1, P1)

BG11: $T \ni k_{temp}$ (BM3, T1, P1, BA1,P4)

BG12: $OO|\equiv T \ni k_{temp}$ (BM4, BG9, I3, I6, BG4, P3)

BG13: $OO|\equiv T \ni max$ (The reasoning process is similar with BG11.)

We can see from the analysis procedure that old owner and tag will have a temporary key and maximum

Table 2: Comparison with other protocols

	Forward security	Backward security	Location privacy	Replay attack	Man-in-middle attack	Desynchronization attack	Backtracking limited authorization
Kapoor (withTTP)	√	√	√	√	√	√	×
Kapoor (without TTP)	√	√	√	√	√	√	×
Kulseng (with TTP)	√	√	√	√	√	×	×
Kulseng (without TTP)	√	√	×	√	√	√	×
Saito (with TTP)	×	√	√	√	√	√	×
Saito (without TTP)	×	√	√	√	√	√	×
Song	×	√	√	√	√	√	×
Zhou	√	√	√	√	√	×	×
Our protocol	√	√	√	√	√	√	√

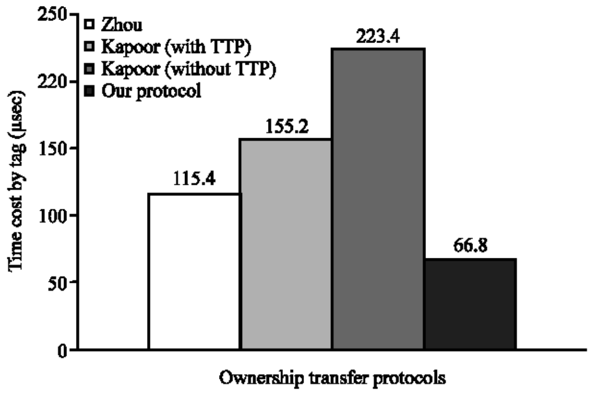

Fig. 3: Computation time cost by tag

access times when new owner authorizes old owner. Moreover, old owner and tag believe the k_{temp} generated by the new owner is the fresh key shared by them.

From the above analysis, we find that this protocol can provide mutual authentication and key update between owner and tag. It transfers tag ownership to new owner and protects the forward security and backward security of the information stored in the tag. It will provide limited access authorization when old owner wants to re-access the tag. This protocol is resistant against replay attack and man-in-middle attack. It also resists desynchronization attack because backend database stores key before updating and the key updated in the procedure of key update. We compare the security of our protocol with other existing research results, as shown in Table 2. The symbol, "√", means the security is satisfied. The symbol, "×", indicates the security is unsatisfied.

Simulation and implementation of the protocol: We implement our protocol in Linux, as well as some existing research results. We obtain experimental data including cost time when tag performs ownership transfer. It is shown in Fig. 3. The data shows the cost time of our protocol is shorter than others.

CONCLUSION

Most of RFID tags in their lifetime experience multiple owners. An old owner will transfer the ownership of tag to a new owner when it hands goods attached by a tag to a new owner. It is a serious problem for users to protect security of ownership transfer procedure. This study proposes an ownership transfer protocol with backtracking ability. We present a brief security analysis in GNY logic. The result indicates our protocol provides mutual authentication between an owner and a tag. It protects forward security and backward security. Moreover, it is resistant against replay attack, man-in-middle attack and desynchronization attack. It also provides temporary access authorization to an old owner of tag after ownership transfer. We implement this protocol in Linux and obtain some experimental data. The data shows the performance of our protocol is better than some other protocols. Next we will research how to further reduce the computation amount of tag.

ACKNOWLEDGMENT

This study is sponsored by National Natural Science Foundation of China No. 61272038.

REFERENCES

Kapoor, G. and S. Piramuthu, 2012. Single RFID tag ownership transfer protocols. IEEE Trans. Syst. Man Cybernet. C: Appl. Rev., 42: 164-173.

Kulseng, L., Z. Yu, Y. Wei and Y. Guan, 2010. Lightweight mutual authentication and ownership transfer for RFID systems. Proceedings of the IEEE Conference on Information Communication, March 14-19, 2010, San Diego, CA., USA., pp: 1-5.

Li, G., R. Needham and R. Yahalom, 1990. Reasoning about belief in cryptographic protocols. Proceedings of IEEE Computer Society Symposium on Research in Security and Privacy, May 7-9, IEEE, Oakland, CA., Piscataway, NJ., United States, pp: 234-248.

Molnar, D., A. Soppera and D. Wagner, 2006. A Scalable, Delegatable Pseudonym Protocol Enabling Ownership Transfer of RFID Tags. In: Selected Areas in Cryptography, Preneel, B. and S. Tavares (Eds.). Springer, Berlin, Germany, pp: 276-290.

Saito, J., K. Imamoto and K. Sakurai, 2005. Reassignment Scheme of an RFID Tag's Key for Owner Transfer. In: Embedded and Ubiquitous Computing, Enokido, T., L. Yan, B. Xiao, D. Kim, Y. Dai and L.T. Yang (Eds.). Springer, Berlin, Germany, pp: 1303-1312.

Song, B., 2008. RFID tag ownership transfer. http://events.iaik.tugraz.at/RFIDSec08/Studys/Publication/15%20-%20Song%20-%20Ownership% 20 Transfer%20-%20paper.pdf

Wang, S.H., 2011. Analysis and Design of RFID Tag Ownership Transfer Protocol. In: Proceedings of the 2011 International Conference on Informatics, Cybernetics and Computer Engineering, Jiang, L.Z. (Ed.). Springer, Berlin, Germany, pp: 229-236.

Zhou, W., E.J. Yoon and S. Piramuthu, 2011. Varying Levels of RFID Tag Ownership in Supply Chains. In: On the Move to Meaningful Internet Systems, Meersman, R., T. Dillon and P. Herrero (Eds.). Springer, Berlin, Germany, pp: 228-235.