Enhanced Hash Chain based Scheme for Security and Privacy in RFID Systems

Vaibhaw Dixit, Harsh K. Verma, Akhil K. Singh
Department of Computer Science and Engineering,
Dr B R Ambedkar National Institute of Technology,
Jalandhar, Punjab, India

ABSTRACT

The research Paper describes security and privacy issues in RFID systems and proposes a new enhanced algorithm. Our proposed protocol is an enhancement over Sang-Soo Yeo and Sung Kwon Kim' protocol and it provides better security and privacy. Comparative analysis of our proposed protocol and Yeo *et al.* protocol is done on the basis of various parameters like number of tags, length of hash chain, number of groups etc. The simulation is done using Sun Microsystems Java and results are analyzed using visual inspection.

Keywords

Security, Privacy, Hash Chain, Hash function, RFID.

1. INTRODUCTION

Radio systems have gained immense popularity during recent years. Radio frequency identification (RFID) is a technology with a great potential in many industries and a wide spectrum of possible uses. Main aim of Industries is to make it possible to trace an individual product through the whole process, so it is possible to see where the product has been at a specific time.
The motivation behind the pervasive use of RFID systems is the need to fully automate remote tracking and identification of objects by embedding cheap and low power RFID tags in the objects. RFID tags are composed of an antenna and a small microchip with some identification information encoded in it. The data transmitted by the tag may contain identification or location information or specifies about the product such as price, color, date of manufacturing etc.
The RFID technology is rife with problems related to security and privacy. There is a concern that information stored on RFID tags could be read by anyone with an RFID reader – data thieves, hackers, or forgers. In such a setup, the RFID system is exposed to a number of security attacks [1, 2] and privacy issues. Normally, RFID system is consist of four main components [3, 4] tag, reader, data processing subsystem, and back-end server. Possible privacy issues and security requirements are outlined below:

1.1 Security Requirements

Anonymity The information given out by the tag should not give any idea about the tag ID.
Anti-spoofing It should not be possible to spoof the response of a tag except through physical tampering and cloning of the tag.
Forward Security an adversary, who eavesdrops on the tag output, should not be able to associate the current output with the past output [3].
Untraceability the tag response should be varying and this variation should not be predictable, i.e. it should be truly random.

1.2 Privacy Issues

The RFID privacy problem has two components [3]. The first concern is the leakage of information about user belongings, and the second is related to tracking and identification by forming associations between tags and their holders. The two aspects are explained below:
Information Leakage in the absence of confidentiality and authentication mechanism, there is a possibility for unauthorized readers to gain access to tag information, which is a threat to user privacy. For example, banknotes labeled with RFID tags may reveal the cash balance of a person.This also involves the risk of corporate espionage, as lack of proper access control enables monitoring of competitor's inventory.
Behavioral Tracking [5] and Personal Identification another important privacy concern is the tracking of individuals by RFID tags carried by them. If the consumer buys an item using a credit card and an adversary can link the credit card details with that of purchased tagged item, the identity and movements of the consumer can be traced by Correlating data from multiple tag reader locations adversary could track movement, social interactions, and financial transactions of the user. Even if the tags are protected, i.e. they only contain product codes rather than unique serial numbers, individuals could still be tracked by the constellation of products they carry.

1.3 Requirements of Back-End Server

Efficiency and Scalability. Some privacy protection schemes [1, 6, and 7] require that back-end server performs a large amount of computation for each identification request. other schemes [8, 9, and 10] may require that the reader frequently writes new data on the tag. In these cases, computations of the reader or the back-end server must be efficient and scalable.
Flexibility. The database of back-end server must be flexible. In some approaches [11], back-end server must rebuild its database whenever a tag is registered or unregistered. It takes very long time to rebuild the database.

2. SYSTEM DESCRIPTION

2.1 Related Work

The scheme of Yeo *et al.* [12] provides a conceptually simple but elegant solution to defeat the tracing problem and to ensure forward security in tag transactions. In this scheme, each tag is required to support only 2 hash functions. When the tag is queried by a reader, it sends the hash (G) of its current identifier and then renews it using a different hash function (H). In addition, the group ID to which the tag belongs, is also hashed and sent. This is done to reduce the search space for tag identification. In order to identify a tag from the output, the backend starts computing hash chains starting from the initial secret for each tag belonging to the group, until the matching tag is found. This is shown in Fig.1 Only the system is able to link all the values sent by the tag while an attacker cannot. Although this scheme satisfies the requirements of anonymity, forward security and untraceability, it suffers from a few problems. Firstly this scheme is prone to spoofing and replay attacks. An adversary can simply eavesdrop on the communication between the tag and reader, and replay the previous tag responses to authenticate itself to the reader.

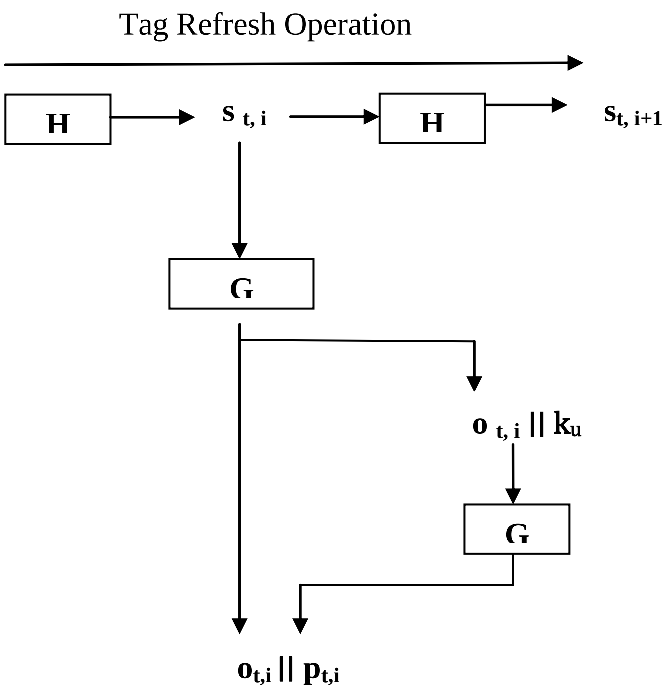

Fig: 1 The Hash Chain Based Scheme of Yeo *et al*..

An adversary may even Launch a *future-time attack*, in which the adversary may query a tag offline repeatedly, store these responses and replay them later to the authorized reader. This enables tag spoofing without amounting to actual physical tampering of the tag. This can be easily avoided if the reader sends a random challenge in each authentication session. This improvement has been used in our Proposed Protocol.

2.2 Proposed Protocol

In this section we are going to describe our improved protocol. The idea here is to reduce computational complexity of back end server by reducing the search space for each tag by introducing the concept of subgroup index within a group index for each tag and backend server. The subgroup index depends on the group index to which it belongs.

Our algorithm improves the security of RFID system by sending a random number r by reader for each query in order to randomize the tag response and to avoid replay attacks.

2.2.1 Notations

The notations used and construction are similar to that in [10]. For the sake of completeness, the same are detailed below:

T: the tag

R: the reader

B: the backend

G, H: $\{0; 1\}^* \longrightarrow \{0, 1\}^l$ 2 different hash functions with outputs of length l.

n: the number of tags in system.

m: the maximum length of hash chain for each tag, i.e. the maximum number of allowed reads on the tag before the tag secret is refreshed.

g: the number of tag groups.

Sg: the number of tag subgroups within a tag group.

r: the random number sent by reader during query process

k_u: the index value of tag group u, where u = 1, 2, - - - - ,g.

$k_{u,\,s}$: the index value of tag subgroup within a group u, where s=1, 2- - - -,sg.

$s_{t,1}$: the initial secret of tag t.

idt : the ID of tag t, where t = 1, 2, · ·,m.

$s_{t,i}$: the ith hash chain value of tag t, i.e. $H^{i-1}(s_{t,1})$, where i = 1, 2, - - - , m.

2.2.2 Protocol Design

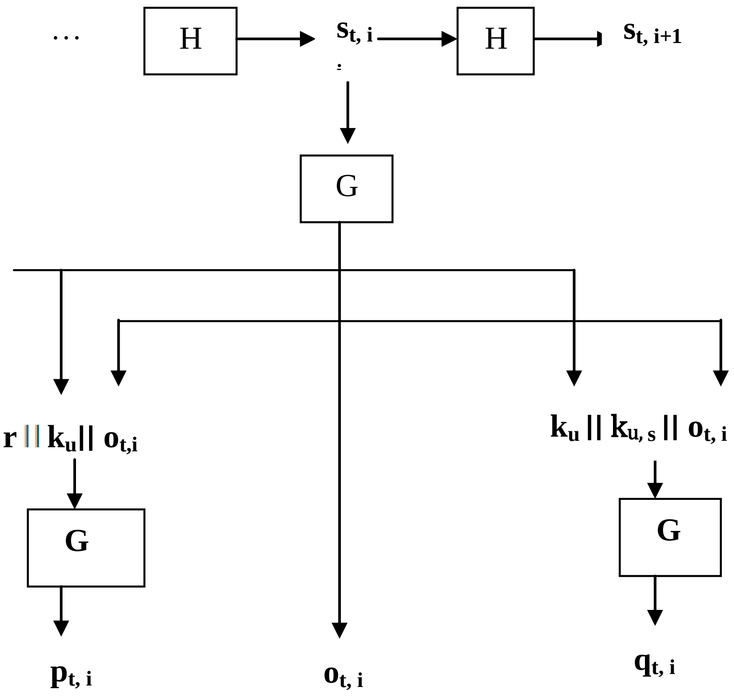

Fig: 2 Proposed Algorithm

2.2.3 Setup

Initially, for each tag t, a random identifier $s_{t,1}$,group identifier k_u ,subgroup identifier $k_{u,s}$ are initialized and stored in the tag memory. The same are also recorded in the system database. G and H are two hash functions supported on each tag

- *Scheme outline*

Main concern of our proposed protocol is satisfying the forward security and Indistinguishability. Our proposed protocol provide forward security by using two hash functions and Indistinguishability is provided by using a random variable r in each reader query. Whenever it is queried by a reader, the tag computes output data $p_{t,i}||o_{t,i}||\ q_{t,i}$for the reader using hash function *G* and the tag changes its secret data $s_{t,i}$ using hash function*H* (figure.1.2)..we have modified Yeo *et al*. protocol by adding two steps, in first step we concatenated random variable r and group index k_u with $o_{t,i}$ and in second step we concatenatied subgroup index $k_{u,s}$ with group index k_u and $o_{t,i.}$.Backend server finds group identifier k_u from r, $o_{t,i}$ and $p_{t,i}$ by checking $p_{t,i} = G(r||k_u||o_{t,i})$ for all u = 1,2, … , g.and finds subgroup identifier $k_{u,s}$ from k_u,$O_{t,i}$ and $q_{t,i}$ by checking $q_{t,i} = G(k_u||K_{u,s}||O_{t,,i})$ for all s=1,2…..,sg.it then finds (*idt,* $s_{t,i}$) by checking $o_{t,i}= G(Hi-1(s_{t,1}))$ for all $1 \leq i \leq n$ and all tags t in a subgroup within a group.

- *Interaction*

When the reader queries the tag with random number r, the tag performs the following steps:

1. Compute $o_{t,i} =G(s_{t,i})$.
2. Compute $p_{t,i} =G(r||k_u||o_{t,i})$.
3. Compute $q_{t,i}=G(k_u||k_{u,s}||o_{t,i})$.
4. Sends response $o_{t,i}||p_{t,i}||o_{t,i}$
5. Replace st,i by st,i_{+1} =H(st,i) and increments its counter i.

The reader R acts as a relay and passes on the tag output to the backend B through a secure channel. Backend B receives the tag output $r,O_{t,i}||P_{t,i}||Q_{t,i}$ from resder R and performs the following steps :

1. Finds group identifier k_u from r, $o_{t,i}$ and $p_{t,i}$ by checking $p_{t,i} = G(r||k_u||o_{t,i})$ for all u = 1,2, … , g.
2. Finds subgroup identifier $k_{u,s}$ from k_u,$O_{t,i}$ and $q_{t,i}$ by checking $q_{t,i} = G(k_u||K_{u,s}||O_{t,,i})$ for all s=1,2…..,sg.
3. Performs tag identification by matching all tags t in a subgroup s with in a group u for $o_{t,i}=G(s_{t,i})$.
4. Sends queried tag data to the authenticated R through a secure channel.

- *Analysis Of Our Proposed Algorithm :*

Security Analysis

The scheme satisfies all security requirements as follows:
Anonymity: G is a one-way function, so if the adversary obtains tag output $o_{t,i}$ he cannot know $s_{t,i}$ from $o_{t,i.}$
Untraceability: G outputs random values, so if the adversary watches the tag output, he cannot link $o_{t,i}$ and $o_{t,i+1}$.
Forward security: Even if the tag is tampered and $s_{t,i}$ is revealed to the adversary, there is no way to find $s_{t,j}$ for j < i, due to one way hash property of H.
Replay attack prevention is also guaranteed in our scheme, which is an added feature over the previously proposed hash chain based schemes. Consider the following replay attack scenario :
Attack Scenario (future-time attack): An adversary queries the tag offline several times when the tag is not in range of the authenticated reader and stores these responses.
These responses are played back later through a cheap cloned tag for online authentication. The random number r in the initial query by the reader guarantees that the tag response is different for each session. Hence capturing tag responses and using them later for online authentication is not possible.

Performance Analysis

The complexities of hashing for the different steps in tag identification are as follows:
Finding group identifier and subgroup identifier finding group identifier requires O(g) hash computations at the backend since it requires linear search over all the tag groups and finding subgroup identifier within a group require O(sg) hash computation since it requires linear search over all subgroup with in a group. So finding group identifier and subgroup identifier requires O(g*sg) hash computations.
Tag identification in a subgroup with in a group it requires O((m*n)/(g*sg)) hash computations at backend since hash chain length is m and number of tags in a subgroup with in a group is n/(g*sg).so we have to perform m hash computations for each tag in a subgroup with in a group.
Thus the total complexity for tag identification at the backend is given by O ((g*sg) + ((m*n)/(g*sg))). This is much less than the total complexity of tag identification in previous schemes. The OSK scheme has a complexity of O(mn)[3] and the scheme of Yeo *et al*.. has a complexity of O(g +((mn)/g))[10].
The flexibility of back-end server is good because it is easy to add a new tag into back-end server's database and to remove an useless tag.

3. RESULTS COMPARISON AND ANALYSIS

A comparative analysis of our proposed protocol and Yeo *et al*. is performed to provide some measurements on the number of hash computations required for each. Effects of several parameters such as number of tags, length of hash chain for a tag, and number of groups , on the number of hash computation required are investigated. This comparative analysis provides the comprehensive view of both protocols, how performance of these algorithms varies on changing these parameters.

3.1 Comparison on the Basis of Number of Tags

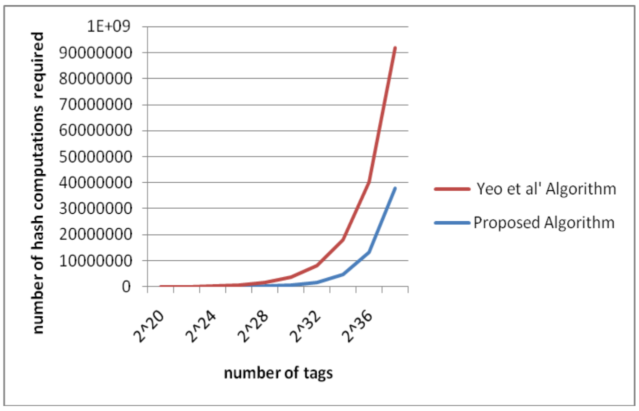

Fig: 3 Comparisons On The Basis Of Number of Tags

In this comparative analysis we are analysing the effect of number of tags over the performance of both protocols. We are changing the number of tags and taking other parameters as constant .we have analysed these results on the basis of graph shown in Fig 3. Effect of n can be seen from the graph, when number of tags is increasing number of hash computations are also increasing for both algorithms. Our proposed algorithm performed better than the Yeo *et al*. algorithm as in Yeo *et al*. algorithm we perform linear search for all tags in a group for tag identification but in our algorithm we linearly search the tags in a subgroup in a group for tag identification.

3.2 Comparison On The Basis Of Hash Chain Length (M)

In this comparative analysis we are analysing the effect of hash chain length over the performance of both protocols. We are changing the length of hash chain and taking other parameters as constant. These results are analysed by visual perception of graph shown in Fig 4.here in graph we can see that for our algorithm number of hash computations required are less than the Yeo *et al*. algorithm. In Yeo *et al*. algorithm for tag identification we have to perform linear search for all the tags in group and for each tag we have to perform m hash computations so when length of hash chain increases number of hash computations also increases but in our proposed algorithm we linearly search all tags in a subgroup within a group so required number of hash computations decreases. When length of hash chain increases then required number of hash computation increases very slightly in our algorithm because we linearly search n/(g.sg) tags in a subgroup for each tag we perform m hash computations.

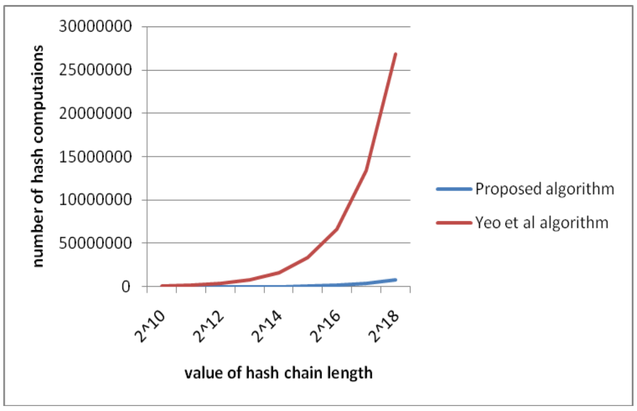

Fig: 4. Comparison on the basis of hash chain length

3.3 Comparison on The Basis of Number of Groups (g)

In this comparative analysis we are analysing the effect of number of groups over the performance of both protocols. We are changing the number of groups and taking other parameters as constant.

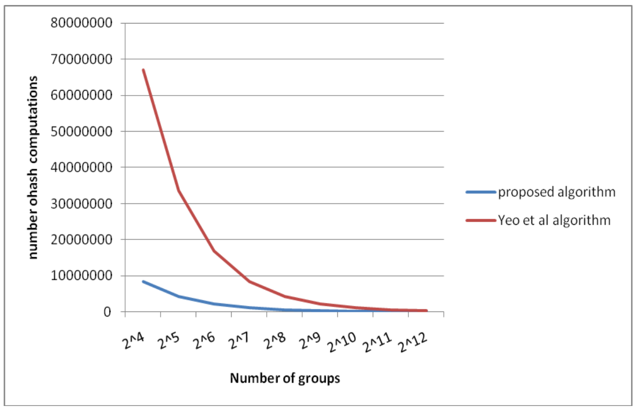

Fig: 5. Percentage decreases in number of hash computation

These results are analysed by visual perception of graph shown in Fig 5. Effect of g can be seen from the graph, when number of groups is increasing the number of hash computations are decreasing for both algorithms. In this graph we can see that number of hash computations for Yeo *et al*. algorithm decreasing fast in comparison to our algorithm because when we are increasing the number of groups then number of tags in a group decreases so the number of hash computations, basically we are dividing group into smaller groups. In our proposed protocol number of groups is increasing but number of subgroup remains constant so the number of tags decreases in a subgroup. Here is a trade-off between number of groups and subgroups, when group size is very small then dividing the group into subgroups does not yield any performance gain its only increases the complexity of protocol

3.4 Comparison On The Basis Of Number of Tags (n), Hash Chain Length (m), Number of Groups (g)

In this comparative analysis we are analysing the effect of n, m, g over the performance of both protocols. These results are analysed by visual perception of graph shown in Fig 6.Effect of n, m, and g can be seen from the graph over the number of hash functions required for both protocols

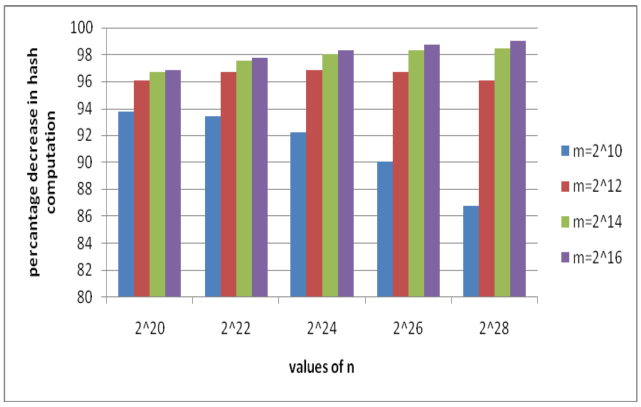

Fig: 6 Graph of Percentage Decrease in Hash Computations Required For Different Values of n, m, and g

These results are analyzed by visual perception of graph. Effect of n, m, and g can be seen from the graph over the number of hash functions required for both algorithms. For our proposed algorithm Percentage saving in number of hash computations required is maximum when value of number of tags and length of hash chain is maximum as, when number of tags increases then the number of tags in a group and subgroup also increases. So in case of Yeo *et al*. algorithm we have to perform search for all tags in a group and we have to do m hash computations for each tag in group but in our proposed algorithm we only search for tags in a subgroup within a group and perform m hash computation for each tag in subgroup.
When number of groups and subgroups increase then the search space for tag identification reduces so the number of hash computations required reduces. So from this graph we can conclude that when value of number of tags and hash chain length is maximum then percentage saving of our proposed protocol is also maximum.

4. CONCLUSIONS

This paper introduces an enhanced protocol. In our proposed protocol we have used the concept of subgroups in a group and one extra hash computation, in order to substantially reduce the computational load at the backend. It also prevents replay attacks, by using a random number in each reader query. A random number is sent by the reader for each query. The tag response is a function of this random number and the current secret. This feature not present in any of the previous hash chain based schemes.
We have done comparative analysis of our proposed protocol and Yeo *et al*.. protocol. Comparative analysis of both protocols are done using different parameters like number of

tags, length of hash chain, number of groups in a RFID system. We conclude on the basis of the results that this enhanced algorithm performs better than the Yeo *et al.* protocol due to usage of a random number 'r' and dividing elements of group into subgroups.

5. REFERENCES

[1] Damith Ranasinghe, Daniel Engels, and Peter Cole. Low-cost rfid systems: Confronting security and privacy. In *Auto-ID Labs Research Workshop*, Zurich, Switzerland, September 2004.

[2] Jeongkyu Yang, Jaemin Park, Hyunrok Lee, Kui Ren, and Kwangjo Kim. Mutual authentication protocol for low-cost RFID. Handout of the Ecrypt Workshop on RFID and Lightweight Crypto, July 2005.

[3] M. Ohkubo, K. Suzuki, and S. Kinoshita, "Cryptographic Approach to "Privacy-Friendly" Tags". *RFID Privacy Workshop*, MIT, November 2003 J. Breckling, Ed., *The Analysis of Directional Time Series: Applications to Wind Speed and Direction*, ser. Lecture Notes in Statistics. Berlin, Germany: Springer, 1989, vol. 61.

[4] S. Sarma, S. Weis, and D. Engels, "RFID Systems and Security and Privacy Implications",CHES 2002, vol. 2523 of LNCS, pp. 454-469, August 2002.

[5] G. Avoine, P.Oechslin, "RFID Tracebility: A Multilayer Problem". In Financial Cryptography, 2005.

[6] M. Ohkubo, K. Suzuki and S. Kinoshita, "Efficient Hash-Chain Based RFID Privacy Protection Scheme", Ubicomp 2004, 2004.

[7] Boyeon Song , Chris J. Mitchell,"Scalable RFID Pseudonym Protocol"Third International Conference on Network and System Security 2009.

[8] P. Golle, M. Jakobsson, A. Juels, and P. Syverson, "Universal Re-Encryption for Mixnets", CT-RSA '04, 2004

[9] J. Saito, J-C. Ryou and K. Sakurai, "Enhancing Privacy of Universal Re-Encryption Scheme for RFID Tags", EUC '04, pp. 879-890, August 2004

[10] Nai-Wei Lo, Kuo-Hui Yeh," Mutual RFID Authentication Scheme for Resource-constrained Tags",In Journal of Information Science and Engineering-IJSE vol. 26, no. 5, pp. 1875-1889, 2010

[11] G. Avoine and P. Oechslin. "A Scalable and Provably Secure Hash-Based RFID Protocol", IEEE PerSec 2005, Kauai island, Hawaii, USA, March 8th, 2005.

[12] Sang-Soo Yeo and Sung-Kwon Kim. Scalable and flexible privacy protection scheme for RFID systems. In Refik Molva, Gene Tsudik, and Dirk Westhof_, editors, European Workshop on Security and Privacy in Ad hoc and Sensor Networks ESAS'05, volume 3813 of Bibliography Lecture Notes in Computer Science, pages 153-163, Visegrad, Hungary, July 2005.