Individual Management of Mysql Server Data Protection and Time Intervals between Characters During the Authentication Process

Irakli Kardava, *Member, IAENG*, Nana Gulua, Beka Toklikishvili, Nugzar Meshveliani, Tamta Kvaratskhelia and Zygmunt Vetulani

***Abstract*—in the field of data protection, in general, many types of methodologies have been developed and implemented. However, there are a number of cases where information can be quite unprotected. Especially when data is unauthorized accessed, or information is 'transported' using any type of memory device. In such a situation, it is easy to steal data and use it illegally. This article describes our proposed approach that solves these problems, both in the case of unauthorized intrusions (invasions) and in the case of illegal 'transportation' of data.**

The implementation of this approach is applied and gives a positive practical result. In particular, the data will become unusable and will be given the face of a combination of chaotic characters if it in any way gets in touch with a person who does not have the right to own this data. And its syntax and semantics will be incomprehensible / indistinguishable. We have integrated this technique into databases for testing. In our case this system is mysql and the administration tool, - phpmyadmin. In general, it should be noted that it can be used in any direction where it is relevant to create, store and transport different types of data.

This article also describes our approach to more powerful protection mechanisms for the authorization process. We gave them a dependence on the time function and made the distinction of identical data input by different people without the help of additional sensors.

Manuscript received in February 01, 2021; Revised in May 24, 2021.

This work was supported by the following project: Engineering studies in IT (ESIT) - internationalized study program offered by the Faculty of Mathematics and Computer Science at Adam Mickiewicz University in Poznan. Project number: POWR.03.03.00-00-M149/16.

I. Kardava is a Doctor of informatics. He is a Senior Lecturer at the faculty of Mathematics and Computer Science at Adam Mickiewicz University in Poznan, Poland; He is also an assistant professor at Sokhumi State University in Tbilisi. (e-mails: kardava@sou.edu.ge; irakar@amu.edu.pl).

N. Gulua is a Doctor of Informatics. She is a professor at the faculty of Mathematics and Computer Science at Sokhumi State University in Tbilisi, Georgia. (e-mail: ngulua@sou.edu.ge).

B. Toklikishvili is a PhD student at the faculty of Mathematics and Computer Science at Sokhumi State University in Tbilisi, Georgia. (e-mail: toklikishvili@sou.edu.ge).

N. Meshveliani is a PhD student at the faculty of Mathematics and Computer Science at Sokhumi State University in Tbilisi, Georgia. (e-mail: n.meshveliani@sou.edu.ge).

T. Kvaratskhelia is a scholarship master student of the Young Scientist program at the faculty of Political Science and Journalism at Adam Mickiewicz University in Poznan, Poland (e-mail: tamkva@st.amu.edu.pl).

Z. Vetulani is a professor at the faculty of Mathematics and Computer Science at Adam Mickiewicz University in Poznan, Poland. He is a leader of the Artificial Intelligence team at the faculty (e-mail: vetulani@amu.edu.pl).

***Index Terms*— Additional sensors, authorization process, chaotic characters, data protection, distinction of identical data, mysql, powerful protection, syntax and semantics, time function.**

I. Introduction

THE main and general goal of our research for the initial stage was to create a new and flexible mechanism for the protection of textual information [1-6]. Then was the implementing the compiled algorithm in any particular system [7-12]. The phpmyadmin database administration system was selected by us according to certain criteria. As you know, it is quite reliable, sustainable, web-based and has open source code. It is these qualities that lead to its widespread use and popularity. However, with regard to the information security, it is possible to add a few new features. The new approach can be considered as: 1. only in the direction of the process of copying information (in this case, import and export); 2. As well as its presentation in the database. We assume that in the first case the textual data is stored in the database in the classical form of its representation, i.e. without any encryption. But when it is exported, it will pass the additional encryption function (section) we have compiled, and when it gets into another memory, it will be written as a modified form. That is, if another person uses it in other systems, the information will be incomprehensible to him, because the encryption algorithm is available only to the administrator of that system. The import process is based on a similar principle. The text first goes through an additional decryption algorithm and then enters the database; naturally it will again be given the appropriate form of the initial state. This means that the information in the database is actually protected after it is exported, as the morphology and semantics of its characters (graphemes) will be incomprehensible to others [13-14]. As we know, in addition to records, data exported through phpmyadmin also contain commands corresponding to sql syntax (select, insert, update, delete ...). Although basic information can be encrypted in an exported file, sql commands can be used to understand database name, table name, field names, type and other no less important information about databases, tables, triggers, and so on and about the general structure. It is known that in many cases even this data is enough for hackers to perform malicious actions (for example: injections). To avoid this case, we also use the principle described above here and encrypt / decrypt

ISBN: 978-988-14049-2-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

the sql commands as well. The entire contents of the exported file will be fully encrypted.

Now consider a second case, when a record is viewed in localhost from a table generated by phpmyadmin. Even in this case it is possible to steal-delete important information and so on. In our opinion, it is possible that successfully can be used our approach to avoid such insecurity or dangers. In particular, it is possible to "single change" the presentation of information in the memory of a given machine during the data viewing process. That is, the data are compiled by our encryption algorithm before it is presented as a table record. The presented information will again have incomprehensible syntax and semantics [15].

For more understanding it should be described the technical side of our approach. This approach allows all database administrators to create their own encryption and decryption function, place them in the appropriate place in the phpmyadmin source code, and activate one of the desired cases described above. If the exported information will get into another database via another phpmydmin, naturally, there won't be any function to decrypt it and it will become useless automatically. It is clear that it is possible that every database system can have its own independent encryption and decryption algorithms. Which precludes the transfer and use of information from one system to another without special permission. The same can be said for logic programming systems, such as Prolog. According to our approach, it is possible for a particular person to be able to change at least the standard syntax and create their own specific style so that the logic of the whole system remains unchanged. In this case, the syntactic inconsistency arises in the same systems of other machines, and this fact can be considered as one of the defense mechanisms.

In addition to the particular system, this approach, in terms of information security, can be implemented in any other similar tool, or even in the entire operating system. Two issues need to be emphasized: 1. our approach does not involve the development of encryption and decryption algorithms (it generally creates a new platform for information security). It depends on the desire and taste of the users; 2. This article describes our experiment and the approach is not officially implemented in phpmyadmin.

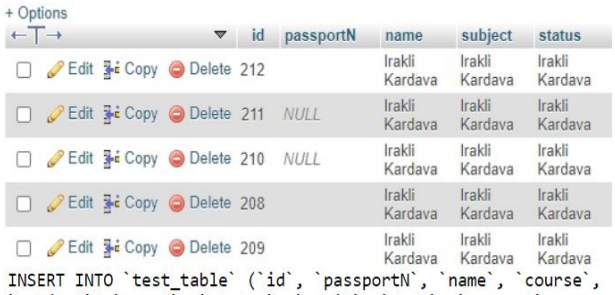
II. Description of the Approach Development and the Implementation

Before we made the changes to phpmyadmin, the first task for us was to study in detail the working principle and mechanism of phpmyadmin. As you know, it's software based on php language. Since its source code is open, the process of diagnosing the operation of a given system has not been difficult for us. For the initial stage we decided to find and study the software code of the file that provided the export of information from the database (as mentioned above, our approach is not officially implemented in the phpmyadmin system, so in this article we refrain from naming the relevant files). We have created an additional demonstration function for encryption, which must pass any record / stream from the database (regardless of type) and only then encrypt it and send it to the computer memory. This means that the contents of the file exported in this way contain encrypted information. See Fig 1 (for example, here is used the text encoded by the Enigma imitator. This mechanism was used to encrypt information by the Germans during World War II. It was hacked by the English mathematician Alan Turing. It was he who created the well-known Turing machine):

```
INSERT INTO `test_table` (`id`,
`passportN`, `name`, `course`, `subject`,
`count`, `status`, `email`, `date`,
`comment`, `password`) VALUES
(212, '', 'HCZHU LEXDZ KZB', '4', 'HCZHU
LEXDZ KZB', 1, 'HCZHU LEXDZ KZB', 'HCZHU
LEXDZ KZB', '2018-06-22', 'HCZHU LEXDZ
KZB', NULL),
(211, NULL, 'HCZHU LEXDZ KZB', '4', 'HCZHU
LEXDZ KZB', 1, 'HCZHU LEXDZ KZB', 'HCZHU
LEXDZ KZB', '2018-06-22', 'HCZHU LEXDZ
KZB', NULL),
(210, NULL, 'HCZHU LEXDZ KZB', '4', 'HCZHU
LEXDZ KZB', 1, 'HCZHU LEXDZ KZB', 'HCZHU
LEXDZ KZB', '2018-06-22', 'HCZHU LEXDZ
KZB', NULL),
(208, '', 'HCZHU LEXDZ KZB', '4', 'HCZHU
LEXDZ KZB', 1, 'HCZHU LEXDZ KZB', 'HCZHU
LEXDZ KZB', '2018-06-22', 'HCZHU LEXDZ
KZB', NULL),
(209, '', 'HCZHU LEXDZ KZB', '4', 'HCZHU
LEXDZ KZB', 1, 'HCZHU LEXDZ KZB', 'HCZHU
LEXDZ KZB', '2018-06-22', 'HCZHU LEXDZ
KZB', NULL);
```

Fig 1. The contents of a file with encrypted information (Enigma start position is AAA. Encrypted text is „Irakli Kardava").

It is shown here that the information is encrypted and cannot be used for unwanted actions. This result is especially important when the data contain people's personal and banking data. Also, even when they contain state secrets. For more illustrations of the achieved result see Fig 2:




```
INSERT INTO `test_table` (`id`, `passportN`, `name`, `course`,
`subject`, `count`, `status`, `email`, `date`, `comment`,
`password`) VALUES
(212, '', 'HCZHU LEXDZ KZB', '4', 'HCZHU LEXDZ KZB', 1, 'HCZHU
LEXDZ KZB', 'HCZHU LEXDZ KZB', '2018-06-22', 'HCZHU LEXDZ
KZB', NULL),
(211, NULL, 'HCZHU LEXDZ KZB', '4', 'HCZHU LEXDZ KZB', 1,
'HCZHU LEXDZ KZB', 'HCZHU LEXDZ KZB', '2018-06-22', 'HCZHU
LEXDZ KZB', NULL),
(210, NULL, 'HCZHU LEXDZ KZB', '4', 'HCZHU LEXDZ KZB', 1,
'HCZHU LEXDZ KZB', 'HCZHU LEXDZ KZB', '2018-06-22', 'HCZHU
LEXDZ KZB', NULL),
(208, '', 'HCZHU LEXDZ KZB', '4', 'HCZHU LEXDZ KZB', 1, 'HCZHU
LEXDZ KZB', 'HCZHU LEXDZ KZB', '2018-06-22', 'HCZHU LEXDZ
KZB', NULL),
(209, '', 'HCZHU LEXDZ KZB', '4', 'HCZHU LEXDZ KZB', 1, 'HCZHU
LEXDZ KZB', 'HCZHU LEXDZ KZB', '2018-06-22', 'HCZHU LEXDZ
KZB', NULL);
```

Fig 2. Information in phpmyadmin and its corresponding exported text in encrypted form in a given file.

ISBN: 978-988-14049-2-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

It's clear that the exported file contains encrypted information, the encryption code of which can only be created by the creator of the encryption function (because our approach assumes that anyone who is a database administrator can create and add their own code in the appropriate phpmyadmin environment. We can't be responsible for the reliability of the encryption and decryption functions created by others). To say more easily, before information can be exported from a database, it has to go through one additional instance. See Fig 3, which schematically describes this process before and after adding our function.

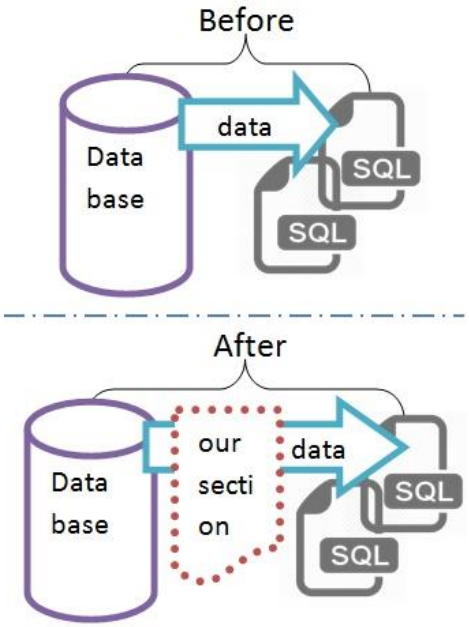


Fig 3. Scheme, where is given the function before and after adding.

Let's go back to Fig 1. Here, the contents of the exported file are encrypted, but only the parts that represent the records directly in the database from which they were downloaded. And sql requests are still given in unencrypted form without any changes. This happened because according to the working principle of phpmyadmin, it turns out that the records come as a separated stream while sql requests come as a separated export process. It became necessary for them to create additional functions for encryption and decryption (it is also possible to use the same functions for both streams by inserting them in the appropriate places. It depends on the desire of a particular administrator). See Fig 4, where the contents of the exported file are shown in full encrypted form.

The administrator can use one of the above mentioned functions or both of them simultaneously. Of course, this also depends on his wishes.

As for the changing the type of information presented in the database, which is considered as the second case in the introduction, it's done on a similar principle. Therefore, as a result of pre-assessment and analysis of the existing threats, it is possible to choose the appropriate approach and activate it / them. If necessary, of course, it can be deactivated by the administrator.

The generalization of this algorithm will be interesting in the various directions of artificial intelligence in general [16-18].

```
INSERT INTO `test_table` (`id`,
`passportN`, `name`, `course`, `subject`,
`count`, `status`, `email`, `date`,
`comment`, `password`) VALUES
(212, '', 'HCZHU LEXDZ KZB', '4', 'HCZHU
LEXDZ KZB', 1, 'HCZHU LEXDZ KZB', 'HCZHU
LEXDZ KZB', '2018-06-22', 'HCZHU LEXDZ
KZB', NULL),  Before
(211, NULL, 'HCZHU LEXDZ KZB', '4', 'HCZHU
LEXDZ KZB', 1, 'HCZHU LEXDZ KZB', 'HCZHU
LEXDZ KZB', '2018-06-22', 'HCZHU LEXDZ
KZB', NULL),
HQPWER PMHS `MTJA_UCLAO` (`CF`,
`EQPNQWXPY`, `FGUX`, `UBFUZT`, `GEJWVOA`,
`ETJGV`, `KEHRNO`, `PVZQS`, `ZOZZ`,
`YXHQHPM`, `YBIPNPFZ`) HYQEPC
(212, '', 'HCZHU LEXDZ KZB', '4', 'HCZHU
LEXDZ KZB', 1, 'HCZHU LEXDZ KZB', 'HCZHU
LEXDZ KZB', '2018-06-22', 'HCZHU LEXDZ
KZB', NULL),  After
(211, NULL, 'HCZHU LEXDZ KZB', '4', 'HCZHU
LEXDZ KZB', 1, 'HCZHU LEXDZ KZB', 'HCZHU
LEXDZ KZB', '2018-06-22', 'HCZHU LEXDZ
KZB', NULL),
```

Fig 4. The fully encrypted contents of the exported file (In this presented article we do not consider the encryption of digits and special characters. It is not important in this particular case.).

Here is given the part of the code, where encryption / decryption will be performed:

```
// Extended inserts case
            if      ($GLOBALS['sql_insert_syntax']      ==
'extended'
                || $GLOBALS['sql_insert_syntax'] == 'both'
            ) {
                if ($current_row == 1) {
                    $insert_line = $schema_insert . '('
                        . implode(', ', $values) . ')';
                } else {
                    $insert_line = '(' . implode(', ', $values) . ')';
                    $insertLineSize = mb_strlen($insert_line);
                    $sql_max_size                              =
$GLOBALS['sql_max_query_size'];
                    if (isset($sql_max_size)
                        && $sql_max_size > 0
                        &&  $query_size  +  $insertLineSize  >
$sql_max_size
                    ) {
                        if (!Export::outputHandler(';' . $crlf)) {
                            return false;
                        }
                        $query_size = 0;
                        $current_row = 1;
                      $insert_line=$schema_insert.$insert_line;
                    }
                }
```

ISBN: 978-988-14049-2-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

```
        $query_size += mb_strlen($insert_line);

        //This is the data that needs to be changed
    $insert_line=$this>replaceCoding($insert_line);

        // Other inserts case
    } else {
        $insert_line = $schema_insert
            . '(' . implode(', ', $values) . ')';
    }
```

III. Additional protection mechanism for the authorization process

This section of this article describes additional security features for user data. As you know, in order to log in to any system, the user must first register (in some cases, the system administrator himself creates accounts for other users) and then be able to log in. For this it needs to enter the username and password in the appropriate fields. This data is then checked. Successful entry is made if their correctness is determined. Here is a simple logic, if the data will be given incorrectly, then login is impossible (there is nothing new in this). However, if the other person (not the user) enters the same data correctly in the appropriate fields, logging will be successful. Although it is quite possible for a real user to whom this account belongs do not know anything about it. That means that these types of protection mechanisms cannot determine to whom belongs the entered data (the actual user or someone else, who knows the other person's identifying information).

This is why this data is the target of various types of information extortion tools (phishing and ect). We have developed an approach that solves the problem described in this section, i.e. it differentiates the input of the same data by different people and will only obey the real user as it "guesses" it. We do not use human biological data, such as fingerprints (bio-vibration, etc.), nor face-recognition and various types of scans to achieve this goal.

For more clarity, for example: x1 person's username is "User12345" and the password is "Password12345", this person writes this data in the appropriate fields and it becomes its successful (sanctioned) entry into the system. At the same time there is x2 person who knows the data of x1. It also writes exactly the same data in exactly the same fields, but it cannot log in.

No additional sensors or scanners that can distinguish people with certain characteristics from each other are used to achieve this result. We have introduced an additional concept which is time and we use its dynamism. In particular, during registration, a person is given the opportunity to choose the interval between entering the user and the password, i.e. the length of the delay.

For instance:
"P" -> 2 seconds and more;
"a" -> 3 seconds and more;
"s" -> 1 second and more;
"s" -> 2 seconds and more;
"w" -> 2 seconds and more;
"o" -> 1 second and more;
"r" -> 3 seconds and more;
"d" -> 4 seconds and more;
"1" -> 2 seconds and more;
"2" -> 1 second and more;
"3" -> 3 seconds and more;
"4" -> 1 second and more;
"5" -> 1 second and more.

We had given to those contents the form of the following formula:

$$SA = \left\{U, P, \left[Uc\langle Ti_{nu}\rangle\right], \left[Pc\langle Ti_{np}\rangle\right]\right\} \quad (1)$$

Where n u=U.lengh-1 and n p=P.lengh-1.

SA is Successful Access;
U - User; P - Password;
Uc<Tinu> - User Character <Time intervaln user>;
Pc<Tinp> - Password Character <Time intervaln password>.

The '[' and ']' symbols indicate that it is not necessary to realize the components given among them, although the use of at least one of them is precisely the practical application of our approach concept.

I.e. A specific example shows that the user will have to memorize the numbers 2312213421311 (these are the intervals between the password characters in seconds, that is, special cases of *Pc <Tin p>* according to the formula (1). Thus, *Pc <Ti1p> = 2; Pc <Ti2p> = 3; Pc <Ti3p> = 1; Pc <Ti4p> = 2* and so on.), but the user will never have to write them when logging in. The user only has to follow these intervals.

So these intervals when entering data are neither entered from the keyboard, nor we have to say, also we don't perform any movement such that it can be memorized and used by anyone. We made the process depending only for the duration of time and the intervals are calculated automatically. Because these intervals are inaccessible to anyone else (it is only in a particular person's brain and on the other hand as a sample is stored encrypted in a special section), even if the user & password data is entered correctly, the intervals will not match the samples and login will not be possible. This means that the entire system is protected from attacks by itself.

IV. Conclusion

The approaches presented in this article are very useful for protecting information and for the establishing inaccessibility of its content. Also, to increase the security quality of the login process in different systems. In addition, in our opinion, it is a general but a clear model of the algorithm of human brain activity and thought exchange between people. As you know, no one has access to the content of the thoughts in the human brain except themselves. They become understandable to others only

ISBN: 978-988-14049-2-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

when the owner expresses them, writes them, moves them or shares them with others. To say technically, thoughts are understood by others only when a given person allows access to them. And if we arbitrarily decide to understand human thoughts, it is both biologically and technically impossible because we do not know what kind of thoughts they are and where they are stored. Even if we do know, their semantics are known only to their owner. If we generalize, it turns out that every individual has his information, but only this person knows their content, until he shares it with others. Of course, this is a natural model, implemented biologically. That was the model, which inspired our work.

REFERENCES

[1] Hacigümüs H., Iyer B., Li C., Mehrotra S., Providing Database as a Service, International Conference on Data Engineering (ICDE), 2002, pp. 29-39.

[2] Luc Bouganim and Philippe Pucheral, Chip-secured data access: confidential data on untrusted servers, Proceedings of the 28th international conference on Very Large Data Bases. 2002, pp. 131-142.

[3] IBM corporation, IBM Database Encryption Expert: Securing data in DB2, 2007.

[4] Mattsson U., Transparent Encryption and Separation of Duties for Enterprise Databases: A practical implementation for Field Level Privacy in Databases, Protegrity Technical Paper, 2004, http://www.protegrity.com/whitepapers.

[5] Hakan Hacigumus, Balakrishna R. Iyer, and Sharad Mehrotra, Effcient execution of aggregation queries over encrypted relational databases, DASFAA, 2004, pp. 125-136.

[6] Sun S. Chung and Gultekin Ozsoyoglu, Anti-tamper databases: Processing aggregate queries over encrypted databases, Proceedings of the 22nd International Conference on Data Engineering Workshops, Washington, 2006, pp. 98-107.

[7] Rakesh Agrawal, Jerry Kiernan, Ramakrishnan Srikant, and Yirong Xu, Order preserving encryption for numeric data, Proceedings of the 2004 ACM SIGMOD international conference on Management of data, ACM, 2004, pp. 563-574.

[8] Tingjian Ge and S. Zdonik, Fast, secure encryption for indexing in a column-oriented dbms, IEEE 23rd International Conference on data engineering, 2007, pp. 676-685.

[9] HweeHwa Pang, Jilian Zhang, and Kyriakos Mouratidis, Scalable Verification for Outsourced Dynamic Databases, Proceedings of the 35th international conference on Very Large Data Bases. 2009, pp. 802-813.

[10] Sung Hsueh, Database Encryption in SQL Server 2008 Enterprise Edition, SQL Server Technical Article, 2008. http://msdn.microsoft.com/en-us/library/cc278098.aspx.

[11] RSA Security company, Securing Data at Rest: Developing a Database Encryption Strategy, whiter paper, 2002.

[12] Sybase Inc, Sybase Adaptive Server Enterprise Encryption Option: Protecting Sensitive Data, 2008. http://www.sybase.com.

[13] Antidze, J. and Gulua, N. Kardava, I. (2013). The Software for Composition of Some Natural Languages' Words Lecture Notes on Software Engineering, pp 96-100.

[14] Kardava, I. Tadyszak, K. Gulua, N. Jurga, S. (2017). The software for automatic creation of the formal grammars used by speech recognition, computer vision, editable text conversion systems, and some new functions. Proceedings Volume 10225, Eighth International Conference on Graphic and Image Processing (ICGIP 2016); 102251Q https://doi.org/10.1117/12.2267687.

[15] Irakli Kardava, Nana Gulua, Jemal Antidze and Beka Toklikishvili - Morphological Synthesis and Analysis of Georgian Words - Human Language Technologies as a Challenge for Computer Science and Linguistics - http://ltc.amu.edu.pl/content.en.html Poznan, Poland. 2019.

[16] Irakli Kardava, Nana Gulua, Beka Toklikishvili and Jemal Antidze - Training Process Automation for Computer Vision - Proceedings of the World Congress on Engineering 2018 Vol I WCE 2018, July 4-6, 2018, London, U.K.ISBN: 978-988-14047-9-4 ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)WCE 2018

[17] Irakli Kardava - Georgian Speech Recognizer in Famous Searching Systems and Management of the Software Package by Voice Commands in Georgian Language - Proc. of The Third Intl. Conf. on Advances in Computing, Electronics and Communication - ACEC 2015 Copyright © Institute of Research Engineers and Doctors, USA .All rights reserved. ISBN: 978-1-63248-064-4 doi: 10.15224/ 978-1-63248-064-4-02

[18] Irakli Kardava - Some Steps to Dialogue between Human Brain and Artificial Intelligence - Lambert Academic Publishing. ISBN: 978-3-659-76480-6 - 2015