Position Papers of the Federated Conference on Computer Science and Information Systems pp. 133–139
DOI: 10.15439/2015F59
ACSIS, Vol. 6

An Architecture for Secure Web Resource with Outsourced Database

Kirill Shatilov[1], Sergey Krendelev[1], Diana Anisutina[1], Artem Sumaneev[1], and Evgeny Ogurtsov[2]

[1]Department of Information Technology, Novosibirsk State University, Novosibirsk, Russia
Email: {shatilov, krendelev, anisyutina, sumaneev}@ccfit.nsu.ru
[2]Plesk, Parallels
Email: eogurtsov@parallels.com

***Abstract*—Security of outsourced data is crucial for businesses. To protect and secure outsourced database, as a part of dynamic web resource while maintaining site's work, we propose a solution featuring following key ideas. Firstly, we suggest to encrypt database content granularly with order preserving and homomorphic encryptions in order to conduct operation inside unmodified DBMS. Secondly, proposed solution implies presence of trusted intermediate component, responsible for SQL query precessing and site hosting tasks. Our approach has been validated through the implementation of complete web resource infrastructure with encrypted database. While web resource, using currently most popular content management system - WordPress, was functioning in normal mode, content of DMBS was secured. In this paper basic ideas of creating secure web resource will be discussed, as long as practical aspects. In addition proposed solution analysis using different metrics will be provided.**

I. Introduction

ENTERPRISES of all sizes are dependent on Internet for business. While some require simple brochure style web sites, others have need for a sophisticated dynamic web resources with vast amounts of data being processed. One of the most common types of dynamic web site is the database driven type. Outsourcing a database or whole hosting infrastructure brings many benefits - it lowers costs, increases reliability and accessibility, enables scalability. However delegating important data to third parties has some drawbacks; privacy issue is one of the most severe of them.

As long as insider threat exists, content of database may be stolen: malicious administrators or hosting provider's staff may capture or leak data[1], adversaries may obtain illegal access to sensitive information[2][3].

In this paper we present an architecture for secure web resource with remote DBMS to address listed security issues. Main effort is put towards securing database's content, while maintaining normal functioning of site's Content Management System (CMS). The key ideas of proposed secure web resource can be described as following:

1) *encrypted* contents of untrusted DBMS. Used encryptions are order preserving, homomorphic, so that selected operations over data can be preformed *inside* DBMS.
2) CMS and hosting software is set in *trusted* zone, where site administrators enforce their own security policies.
3) *interception* and *processing* of SQL queries from CMS and DBMS responses on trusted intermediate component. Transformed queries are executed on DBMS server.
4) *creating environment*, that would support queries transformation and maintain confidentiality, integrity and availability of resource's data.

This research was performed in Novosibirsk State University under support of the Ministry of education and science of Russia (contract no. 02.G25.310054)

The next section of paper discusses related work. After it, in Section 3, we describe the general ideas of proposed secure web resource. Section 4 gives some insight into encryptions, which are used in secure SQL queries processing. Following Section 5 presents practical aspects of the proposed solution. Analysis and evaluated results are listed in Section 7. Finally, the last section of this paper exposes future development options.

II. Related Work

While business processes are being tightly converged with information technology security procedures[4], multiple solutions for securing outsourced content were presented in Industry and Academy to fulfill demand of preserving data's privacy. Main accent is made to secure remote relation database by various encryptions.

Transparent Data Encryption (TDE) solution from DBMS developers, such as Microsoft[5] and Oracle[6], offers encryption at file level. TDE solves the problem of protecting data at rest, encrypting databases both on the hard drive and consequently on backup media, but does not protect contents of DBMS from access of illegal parties from database interface (e.g. theft of database user's authentication information).

In order to avoid limitation of database's full encryption several solutions for conducting queries over encrypted data were proposed. MIT CryptDB[7] presents a solution with support of multi-purpose encryptions based on, so called, "onion" database structure. Other solutions[8] offer fast execution of queries and search over encrypted data. However most of solutions lack fully homomorphic encryption and impose limitations on SQL queries.

III. Basic Principles

This section describes key ideas and distinctive features of the proposed solution.

Let us consider a web resource that is backed up by the database. The web resource is viewed as *secure* if its content is available only for legal users, or, in other words, confidentiality, integrity and availability of all resource's data are maintained constantly.

We propose an architecture for secure web resource with outsourced database. There are four main components:

- DBMS server
- crypto environment
- site engine (CMS) and web server
- client application

There is a clear division for architecture's components - they are either trusted or untrusted. The set of systems that can provide secure storage and processing facilities for confidential data is considered as *trusted* components. *Untrusted* environment is a set of systems, where security policy is setted up and enforced by third parties, therefore it can not be deputed to handle confidential information. As such, an untrusted system is one whose compromising will not lead to data leakage[9].

A. DBMS

The only untrusted component in proposed solution's architecture is outsourced DBMS. This DBMS is unmodified and plays a significant role in web resource functioning. All data stored in such DBMS in any particular moment of time must be secured. Additionally, all traffic to database and database's inner logs must not contain any vulnerable information, because it is supposed that malefactor can capture or monitor them.

B. Crypto environment

The key component of architecture is Crypto environment. Its main purpose to process SQL queries and DBMS responses while encrypting vulnerable data. Crypto environment is widely discussed in our first work[10]. Since first publication it has been improved in multiple aspects. Crypto environment consists of the following parts:

1) Encryptions subsystem: this component is an expandable set of encryption libraries with specified interfaces. Used encryptions are discussed in Section IV.

2) Metafile management subsystem: all keys and encryptions' meta data (types, names, count of output columns, other auxiliary information) are stored in meta file. Meta file subsystem's main purpose is to fulfil requests for keys and meta information of authorized components and to add new records for newly created tables and columns. Another subsystem's task is to encrypt/decrypt meta file when saving/loading from hard disk drive (meta file is always encrypted on Non-volatile Storage Devices for security reasons).

3) SQL queries processor: this component's objective is to parse SQL queries, reveal encrypted columns, encrypt data from this columns and to reconstruct resulting queries resting upon an encryption properties and encryption-specific math.

4) DBMS responses processor: decrypting selections from database.

As all encryption procedures are performed inside Crypto environment, DBMS never gets access to encryption keys.

C. Site engine

Site engine and web server software remains unmodified. The only subject to change is default SQL schema, that has to be encryption aware. All SQL- and response transitions are performed by Crypto environment. For site engine or any other SQL-backed application inside Crypto environment it is indistinguishable whether they work with usual DBMS or encrypted one.

D. Client application

End-users access secure web resource through ordinary browser. Access to resource is limited by login/password combination. Two factor authentication is the best option for hardening security.

E. Mechanism

Mechanism of web resource functioning is analogous to usual one - web server stores, processes and delivers web pages generated by site engine, that uses information from external database. But SQL queries from site engine are processed by Crypto environment, content of DBMS is encrypted, all queries are performed over encrypted data. During initialisation phase all "CREATE" queries are reconstructed and original data schema is mapped to encrypted one; all mapping parameters (meta information and encryption keys) are stored inside meta file. Analysis and reconstruction of further queries is based on this mapping parameters. All information requested by site engine for web page generation is presented in decrypted form by Crypto environment.

F. Configuration

There are two possible configurations of secure web resource - centralised and distributed.

1) Centralised: Schema of centralised secure web resource architecture is shown in Fig. 1. This concept implies that there is an additional trusted server, where site engine, web server and Crypto environment are set. All end users connect to this server using web browser. This proxy server has to be maintained in trusted environment by the administrators of secured web resource and can be a possible bottleneck for site access in case of high loads, but, on the other hand, proxy server allows exploitation of secure resource from mobile devices.

2) Distributed: Distributed configuration means that all trusted components are set on end user's device. Thus so called heavy client consists of http client, web server, site engine and Crypto environment. In case of distributed architecture additional meta file synchronisation mechanism is required in order to maintain consistency of meta information across different clients.

Distributed secure web resource architecture is illustrated in Fig. 2.

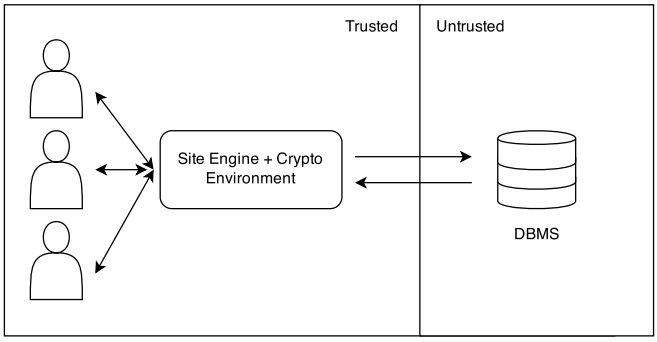

Fig. 1. Centralized Architecture

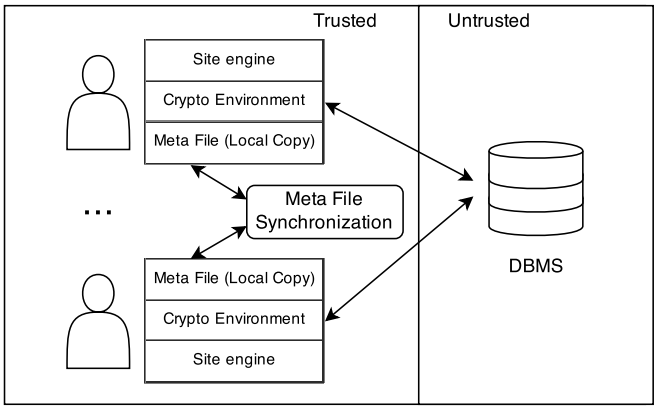

Fig. 2. Distributed Architecture

IV. Encryptions

A. Probabilistic

Probabilistic encryption is the use of randomness in an encryption algorithm, so that when encrypting the same message several times it will, in general, yield different ciphertexts. Data that are intended only for retrieval, but posses high value for users, can be encrypted in such way. Probable applications of such an encryption are comments' content or texts, where search operation is not essential. In the proposed solution, we use proprietary developed probabilistic block encryption. Implementation of Crypto environment is flexible, so that any other symmetric encryption can be used; for compatibility C++ interface-adaptor has to be implemented.

B. Deterministic

Deterministic encryption provides strong security, it leaks only which encrypted values correspond to the same data value. In site's data schema, deterministic encryption can be applied to user names and emails, search tags and password hashes. Usage of deterministic encryption for such data fields allows performing equality comparison and join operations, while it removes duplicate values from encrypted database (user names or emails must be unique, only one instance of each search tag exists).

C. Order Preserving

Order preserving encryption allows order relations between encrypted data items to be established, without revealing data itself. Such an encryption can be used for constructing secure indexes in database. With additional library functionality order preserving encryption can be applied to fixed-sized strings and dates with specified format.

Alternatively, order preserving encryptions can be probabilistic; encryptions with this property increase security, removing duplicate values from database, but also limiting operations held over data (equality comparison is not working properly). In proposed solution we use order preserving encryption discussed in [11].

D. Homomorphic

An encryption schema is called fully homomorphic, if it is able to evaluate an arbitrary function over ciphertexts. In this case decrypted value must match to a calculation result of the same function over plaintexts. The main feature of schema[12] that is used in the proposed solution is ability to define a strict upper bound of ciphertext size when performing calculations on it for both addition and multiplication. This fully homomorphic encryption is practically efficient and does not require huge computational or storage resources.

Homomorphic encryption can be applied to any integer data fields in site's SQL schema, that presents valuable information and supposed to be multiplied or added during web page generation (i.e. products' quantity and price in on-line shop).

V. Methods

This section features some examples of SQL queries handling to illustrate the challenge of correct processing of user queries with minimum limitations.

A. JOIN operation

In case, when some operations are intended to be performed over encrypted data across multiple columns and tables, following functionality can be used. In "CREATE" queries user must explicitly state that columns are members of one, so-called, "JOIN-group" in order to notify SQL queries processor to use one encryption key for all columns from group. The encryption must be deterministic for correct execution of further "SELECT" queries.

B. Probabilistic Order Preserving Encryption Handling

As it was previously stated, probabilistic order preserving encryption limits operation of equality comparison inside encrypted database. In order to avoid this limitation, following workaround was implemented.

Initial query example:

```
SELECT *
FROM table_name
WHERE column_name = value;
```

Output processed query:

```
SELECT *
FROM table_name
WHERE encrypte_column_name > OPE(value-1)
AND encrypted_column_name < OPE(value+1);
```

Here *OPE(value)* is considered as procedure of order preserving encryption.

C. Aggregate functions

If homomorphic encryption is used, some limitations are imposed. For example, aggregate function AVG (average value calculation) cannot be correctly performed inside DBMS. For correct execution of "SELECT" queries with average value calculation following transformation rules are applied.

Initial query example:

```
SELECT AVG(column_name)
FROM table_name;
```

Output processed query:

```
SELECT
SUM (encrypted_column_name),
COUNT (encrypted_column_name)
FROM table_name;
```

After response from DBMS is received, it consists of 2 values; first (sum) is decrypted and divided on second one(count), the result is passed to application.

In this section only a few examples of different SQL processing techniques were given to illustrate the mechanism of how Crypto environment work. However, some restrictions remain and it is a subject of future work to support all possible queries.

VI. Practical Approach

Practical implementation of the proposed concept in real application was essential for proving that such an approach for securing outsourced DBMS would work. In this section we discuss practical aspects of creating secure web resource with remote relational DBMS (MySQL[13]) using WordPress site engine[14].

WordPress is a free open-source blogging tool and content management system. Features include a plugin architecture and a template system. WordPress was used by more than 23.3% of the top 10 million websites as of January 2015[15]. WordPress is the most popular blogging system in use on the Web, at more than 60 million websites, according to official web site.

Proposed solution's practical implementation schema is illustrated in Fig. 3. Nginx[16] was used as a web server paired with a PHP FPM (FastCGI Process Manager); WordPress was configured to be used as a web page generation engine.

MySQL DBMS provides functionality of MYSQL Proxy, which is a software that intercepts traffic between database client (in case of web resource it is a php code of WordPress engine) and MySQL server(s) and can monitor, analyze or

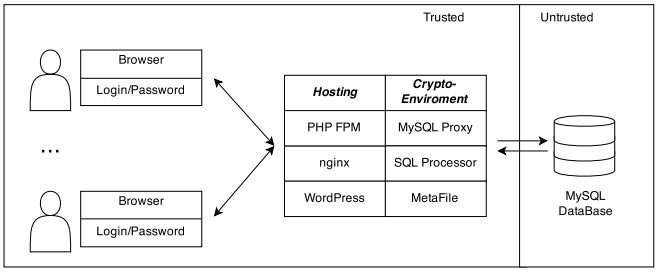

Fig. 3. Detailed schema of proposed solution

transform their communication. Its flexibility allows query analysis, query filtering and modification. Instance of MySQL Proxy was using .lua scripts, that called SQL queries and DBMS responses processors' procedures from prepared dynamic link library(.dll). All syntax analysis modules, encryption libraries were written on C++ programming language.

WordPress modifications were superficial and were made in three following directions:

A. Proxy

MySQl Proxy's IP address and TCP port were configured in WordPress as a default database, so that all queries generated by WordPress are processed by Proxy.

B. SQL schema modification

As it was stated in Section III, the application's SQL schema, which uses Crypto environment, must be encryption aware. Each column, which is intended to be encrypted, must be marked with encryption identification string in "CREATE" statement, as long as, JOIN-group name must be specified if needed:

```
CREATE TABLE wp_comments (
        --column name:
        comment_author
        --column type:
        tinytext
        --column constraints:
        NOT NULL
        --encryption id:
        encrypted_deterministic
        --join-group id:
        JOIN_GROUP wp_users_common,
... )
```

Encryptions applied to WordPress data schema are listed in Table I.

Homomorphic encryption was not applied to default WordPress configuration, but it is available for usage in various plugins with mathematical or financial functionality, e.g. for secure e-commerce resources.

Current way of encrypting WordPress SQL schema does not provide search over posts' content functionality, because probabilistic encryption is used for texts. But it is possible to add tags or keywords to post, so search will be performed over

TABLE I
ENCRYPTIONS, APPLIED TO WORDPRESS DATA SCHEMA

Field	Type	Encryption
tags, headers	text	deterministic
post, comments text	long text	probabilistic
post, comments, events date	date	OPE
user email, name	text	deterministic
user password	text	deterministic
ratings, order terms	integer	OPE

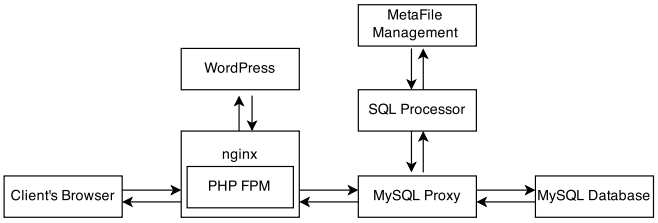

Fig. 4. Components Interaction

them. Mechanism of searching over encrypted data, proposed in [8], are similar - keywords are extracted from text and are indexed, search is held over set of keywords, not over the original text. So in this case, disabling search over texts is not a serious limitation.

Encrypting user names and passwords does not only protect database's login information's confidentiality, but also disables malicious database administrators from adding illegal users with right to access site's content.

C. Security Modifications

WordPress was configured so, that access to site was limited to users who are logged in or to users with an IP addresses from a specified set. In addition, search engines indexing was disabled.

To conclude, let us consider a typical components' interaction:

1) Client accesses web resource through browser using login/password.
2) Web server resolves http requests.
3) Web pages are generated by site engine using user's data from SQL database.
4) Site engine uses MySQL Proxy as a SQL database interface.
5) Scripts, used in Proxy, call SQL queries processor for syntax analysis and data encryption (or Selection processor to decrypt data selections).
6) Keys used for encryption or decryption are stored in meta file. Meta file management system resolves all requests by SQL queries processor and returns keys and meta information required for correct syntax processing.
7) After queries processed and all confidential data inside them are encrypted, proxy sends queries for execution inside DBMS.

Components interaction mechanism is shown in Fig. 4.

All web resource infrastructure was deployed on Microsoft Windows Server 2012 32bit OS. Current version of Crypto environment supports only Windows Operation System, while other components of secure web resource can be deployed across multiple platforms defined by MySQL and nginx vendors (e.g. CentOS, Ubuntu).

As a result of experiment, web resource was functioning, information uploading and retrieving was working correctly, while content of DBMS was encrypted. All queries, that was generated by WordPress, were compatible with Crypto environment and were executed correctly. However, slight time overhead was detected, and it will be discussed in following section.

VII. ANALYSIS

This section features analysis of the proposed solution and provides evaluated results.

A. Concept

There are 2 possible variants for configuring site infrastructure according to proposed solution - distributed and centralised. Heavy client, same as basic intermediate component, requires not more than 300 available hard disk space. Around 200 Mb of RAM is needed for normal functioning of heavy client. Centralised variant requires more RAM on machine with Web-server, site engine and Crypto environment in order to support sufficient quantity of simultaneous sessions, and optional high-availability component is recommended.

B. Performance

Developed solution uses encryptions and syntax processing, thus performance and data overhead exists.

1) Performance overhead: WordPress initialization with crypto environment took around 50% longer than unmodified WordPress. This significant increase is not critical, because initialization phase is performed very rarely. Content upload took ~20% more time, while content retrieval took less than 15% extra time.

2) Data overhead: Average database size increase is 60%. This high value of data overhead is caused by used encryptions.

C. Security

To prevent data leakage and strengthen security various mechanisms are available for the proposed solution. This part of a current section features overview of possible security breaches, evaluated damage to data confidentiality and suggested countermeasures.

Main purpose of suggested solution, as it was previously stated, is to protect vulnerable data in outsourced database. Adversary who gained access to database's content can not read private data, that are stored in database encrypted all of the time, but he can spoil or delete data. In addition he can calculate distribution of chosen cipher texts (applied for order

preserving encryptions) and perform evaluation of database size. Preventing successful cryptanalysis is possible by using strong encryptions.

Malefactor's access to Crypto environment is more serious security threat. Meta file is always stored encrypted on hard disk and is partially decrypted in RAM, thus crypto keys can be gained with low chances of success through RAM scanning. Accessing database proxy without database user login pass is ineffective.

Breaches on end users side are most severe. Malefactor can gain login and password of site users by exploiting various key loggers and relying on user's ignorance for security policies(password length, storing password in browser or in any other possibly insecure container). In this case full access to all sabotaged user's data is granted. After such a security breakthrough reencryption of meta file is the best option. This action will limit access of compromised user to none. To prevent this serious leakage scenario, several measures are recommended: responsible passwords management policy (constant passwords changing, reliable password length), regular reencryption of meta file, two-step authentication on client, increased security measures on users' workstations (anti viruses, proper firewall configuration, etc.).

To maintain integrity and availability of data in case of Meta file loss or corruption encrypted local meta file backups have to be done on user-defined schedule.

D. List of restrictions

Proposed solution solves problem of keeping confidentiality of outsourced data, while adding some constrains to normal functioning of web resource. In particular practical approach, using WordPress site engine following limitations are actual:

1) plugins usage: For each plugin that is working with confidential data special security patch is needed to secure plugin's data in DBMS.

2) backups usage: All backups must be performed in trusted zone, encrypted there and passed to DBMS. Otherwise an unencrypted backup in untrusted zone is an obvious threat to data confidentiality.

3) version dependency: WordPress data schema is changing through different versions, multiple tables and columns are being added or removed, thus any new confidential information, intended to stored in database, has to be encrypted; each version has to be revised and special security patch has to be applied.

VIII. Future Work

We presented our solution for secure site architecture, backed by database with minimal trust, that is immune to database theft and malicious insiders on DBMS side. Our research is aimed to delivery of a product, which will solve actual problems of information security, with minimal changes to site's hosting software and hardware. To achieve determined goal many changes must be done to the project in current state, including the following. First is automated or user defined selection of encryption for confidential data from any arbitrary SQL data schema. This improvement will discard some of described restrictions, while allowing us to present universal solution for multiple site engines and any custom plugins. Secondly, further security improvements to prevent any attack scenarios are going to be added to solution's features. Another direction for development is adopting client software for mobile platforms.

References

[1] W. R. Claycomb and A. Nicoll, "Insider threats to cloud computing: Directions for new research challenges," in *Proceedings of the 2012 IEEE 36th Annual Computer Software and Applications Conference*, ser. COMPSAC '12, Washington, DC, USA: IEEE Computer Society, 2012, pp. 387–394, ISBN: 978-0-7695-4736-7. DOI: 10.1109/COMPSAC.2012.113. [Online]. Available: http://dx.doi.org/10.1109/COMPSAC.2012.113.

[2] (2015). Data loss statistics, [Online]. Available: http://datalossdb.org/statistics (visited on 04/20/2015).

[3] (2015). Privacy rights clearinghouse. chronology of data breaches, [Online]. Available: http://www.privacyrights.org/data-breach/ (visited on 04/20/2015).

[4] G. Wangen and E. A. Snekkenes, "A comparison between business process management and information security management," in *Proceedings of the 2014 Federated Conference on Computer Science and Information Systems*, M. P. M. Ganzha L. Maciaszek, Ed., ser. Annals of Computer Science and Information Systems, vol. 2, IEEE, 2014, pages 901–910. DOI: 10.15439/2014F77. [Online]. Available: http://dx.doi.org/10.15439/2014F77.

[5] (2015). Microsoft. Transparent Data Encryption, [Online]. Available: https://msdn.microsoft.com/en-us/library/bb934049.aspx (visited on 04/20/2015).

[6] (2015). Oracle. Transparent Data Encryption, [Online]. Available: http://www.oracle.com/technetwork/database/options/advanced-security/index-099011.html (visited on 04/20/2015).

[7] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, "Cryptdb: Protecting confidentiality with encrypted query processing," in *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, ser. SOSP '11, Cascais, Portugal: ACM, 2011, pp. 85–100, ISBN: 978-1-4503-0977-6. DOI: 10.1145/2043556.2043566. [Online]. Available: http://doi.acm.org/10.1145/2043556.2043566.

[8] M. Sharma, A. Chaudhary, and S. Kumar, "Query processing performance and searching over encrypted data by using an efficient algorithm," *CoRR*, vol. abs/1308.4687, 2013. [Online]. Available: http://arxiv.org/abs/1308.4687.

[9] "The trusted systems problem: Security envelopes, statistical threat analysis, and the presumption of innocence," *IEEE Intelligent Systems*, vol. 20, no. 5, pp. 80–83, 2005.

[10] K. Shatilov, V. Boiko, S. Krendelev, D. Anisutina, and A. Sumaneev, "Solution for secure private data storage in a cloud," in *Proceedings of the 2014 Federated Conference on Computer Science and Information Systems*, M. P. M. Ganzha L. Maciaszek, Ed., ser. Annals of Computer Science and Information Systems, vol. 2, IEEE, 2014, pages 885–889. DOI: 10.15439/2014F43. [Online]. Available: http://dx.doi.org/10.15439/2014F43.

[11] M. Usoltseva, S. Krendelev, and M. Yakovlev, "Order-preserving encryption schemes based on arithmetic coding and matrices," in *Proceedings of the 2014 Federated Conference on Computer Science and Information Systems*, M. P. M. Ganzha L. Maciaszek, Ed., ser. Annals of Computer Science and Information Systems, vol. 2, IEEE, 2014, pages 891–899. DOI: 10.15439/2014F186. [Online]. Available: http://dx.doi.org/10.15439/2014F186.

[12] A. Zhirov, O. Zhirova, and S. F. Krendelev, "Practical fully homomorphic encryption over polynomial quotient rings," in *Internet Security (WorldCIS), 2013 World Congress on*, IEEE, 2013, pp. 70–75.

[13] (2015). MySQL official site, [Online]. Available: https://www.mysql.com/ (visited on 04/20/2015).

[14] (2015). WordPress official site, [Online]. Available: https://wordpress.org/ (visited on 04/20/2015).

[15] (2015). Usage statistics and market share of content management systems for websites, [Online]. Available: http://w3techs.com/technologies/overview/content_management/all/ (visited on 04/20/2015).

[16] (2015). nginx official site, [Online]. Available: http://nginx.org/ (visited on 04/20/2015).