DEVELOPMENT OF A CRYPTOGRAPHIC SOLUTION BASED ON KERBEROS FOR DATABASE SECURITY

Nikola Pavlović*,
Saša Adamović

Singidunum University,
Belgrade, Serbia

Abstract:

In this paper, we deal with the development and implementation of a security protocol for the cryptographic protection of a database. The proposed scheme is based on the protocol Kerberos whose functionality allows the use of the symmetric cryptography. On the other hand, allows easy control, keys generation and distribution in databases with multiple users.

Keywords:

Kerberos, symmetric cryptography, asymmetric cryptography.

1. INTRODUCTION

Modern Internet technologies lead to an enormous amount of new data on the Internet nowadays. We are aware of the fact that the Internet has become an essential part of our life. The reason for that is increased efficiency, effectiveness and omnipresence but it opens new security questions about confidential information secrecy which led to creating complex cryptographic [2] systems made for securing authentication, secrecy, integrity and undeniable services.

Modern business applications could not be working without databases which are also subject of our research in this project. The proper solution is suggested as a security protocol, which allows using symmetric ciphers in databases. There are all problems and challenges about the domain of cryptographic solution usages included in this solution.

The suggested protocol is primarily based on an authentication protocol – Kerberos [7]. It combines symmetric keys, timestamps and the third trusted party. The authentication process can be easily acknowledged by something we know (i.e. username and password), by something we have (i.e. smart card) and something we are (biometry). The authentication we used in our protocol is based on something we know, the most used model of authentication.

To secure user's credentials, username, and password, the Kerberos protocol that is currently used is upgraded by using asymmetric cryptography. In that way it provides better level of security of a user's credentials. Furthermore, the generic scheme of the suggested solution and its implementation will be explained.

Correspondence:
Nikola Pavlović

e-mail:
nikola.pavlovic.141@singimail.rs

DOI: 10.15308/Sinteza-2018-3-8

2. KERBEROS PROTOCOL

Kerberos [1] is an authentication protocol developed at MIT (Massachusetts Institute of Technology). It is based on Needham's and Schroeder's work, and it is projected to provide a strong authentication for client/server application by using cryptography based on the secret key. This allows privacy integration as well as the data integration. The main idea of this protocol is that the user's passwords are never to be sent as a plain text.

For this protocol to work, it should be based on the trusted third party and it uses symmetric cryptography which allows a communication between the users encrypted. The central part is KDC (Key Distribution Center) which shares symmetric keys with each user. Its role is to provide a safe communication between the users using a Kerberos symmetric master key.

In practice, it is often the case that Kerberos uses DES (Data Encryption Standard).

While working, Kerberos issues tickets which contain keys and other information needed for accessing network resources. KDC issues TGT (ticket-granting-ticket) to each user which is generated when the first user appeared on the system with their credentials. On the basis of TGT, the normal ticket is later issued which allows access to individual network resources.

TGT contains key's session, unique user's ID and a ticket's timestamp.

The principle of the Kerberos protocol is the following:

If a user signed up on a system, he would have to use their credentials. The user's computer performs a derivation of a key using a certain hash function and this key is shared with a KDC. Based on this key, the user receives the main Ticket (TGT) from the KDC based on which he can access the network resources. The idea is to use hash of a user's password as a symmetric key when requesting the ticket and then the obtained ticked to be used for networking resources.

3. STATE OF THE ART

Author [3] compares how different protection, software or hardware implementations of the database affect the relationship between the level of security and performance. In their research they had discovered that AES algorithm's performances are better than RSA's or Blowfish's. In addition to software protection, they had studied hardware protection through the use of various devices from well-known manufacturers such as IBM, Eracom, nCipher and Chrysalis. The author has shown that ciphering the database in a software realisation directly can lead to low performances in the sole work and suggests using the hybrid model which is the combination of software's and hardware's implementation.

Authors [4] suggest using the Kerberos protocol as a need for implementation service of authenticating over the Internet. The reason for this is that Kerberos provides the high level of reliability during the authentication. Kerberos protocol is implemented as an authentication protocol during login on a working station by checking the user credentials from the trusted third party.

In the paper [5], the authors suggest using the Kerberos protocol for an authentication in cloud computing. The reason for this is the need for the data protection which a client access via the Internet and their distribution on a reliable and a safe way, as well as the protection from unauthorized access.

4. MODERN CRYPTOGRAPHY

For strong data encryption we need to generate a secret key (symmetric or asymmetric). Depending on the cipher of using the key, there are two types of algorithms: which are based on symmetric and asymmetric cryptography. The main difference between algorithms is that the symmetric cryptography uses the same key for encryption, while asymmetric cryptography uses a pair of keys – public and private key.

Symmetric algorithms are divided into two groups: stream and block ciphers. Some of the stream ciphers are A5/1 and RC4 (arcfour), which work by encrypting the message bit by bit. Block ciphers (DES, AES), use different size for input block of data. The size of these blocks depend on chosen algorithm (64 – 256 bits).

In the case of block algorithms, it is necessary to bind these blocks to specific modes of operation or encrypt mode. These modes are: Electronic Codebook mode -ECB, Cipher Block Chaining - CBC, Counter mode - CTR and many others. These modes allow to chain the encrypted blocks in a specific way. The standard implementation (and in our case) of modern block cipher consider to use CBC encryption mode, it means that all generated output blocks are chained. Except encryption modes, we use the standard padding mode for input data blocks.

Asymmetric cryptography is not only used for encryption, but is also used to create a digital signature.

The main idea is that there are a pair of keys: public and a private key. The public key is used for encrypting the data, while the private key is used for decrypting. Default key size 1024 bits, but it can be increased to 2048 or 4096 bits. Private and public key are mathematically complimentary, it means that public key doesn't reveale information about private key.

5. PROPOSED SOLUTION

In this chapter, a developed solution will be presented from theory to practical implementation. Our solution consists of three parts: client, authentication server and a database server. The client side is in the role of communicating between the client and the authentication server, or the database server. The client generates the parameters for accessing to network resources. In order to communicate with the authentication server, the client first generates a pair of keys. The main idea is that every client has their own pair of keys, and the authentication server has a pair of keys for each client. Client and server in the duplex communication mode exchange their public keys. This scenario excludes man in the middle attack.

All parameters (key pair, timestamp, id client) are stored on the authentication server. The key part of the protocol is issuing tickets. There are two types of tickets: temporary and approved ticket. A ticket is crucial because it contains secret keys and other information that is needed to access network resources or databases.

The first type of ticket is a temporary ticket. It is generated for the first time when a user requests connection to a database, i.e. when the authentication process is required. This ticket allows the user and the authentication server to exchange public keys.

A temporary ticket is the basis for forwarding user credentials from users to the authentication server in encrypted form using asymmetric cryptography. After successful authentication, when the authentication server decrypts the private key for the given user and checks the parameters, an approved ticket is generated. The approved ticket contains the parameters for accessing the server to work with the database. These parameters are symmetric key and a unique identifier, which will be presented to the database server.

After that, ticket is forwarded to the client, encrypted with the client's public key. This principle of communication and exchange of keys secretly ensures the integrity and secrecy. The next chapter will present the implementation of the proposed solution. The protocol itself is developed in the Java programming language. We used Java cryptography extension.

6. REALISATION OF THE GIVEN SOLUTION

In this chapter, the realisation of the given idea will be shown step by step, as it is shown in the figure 1, below.

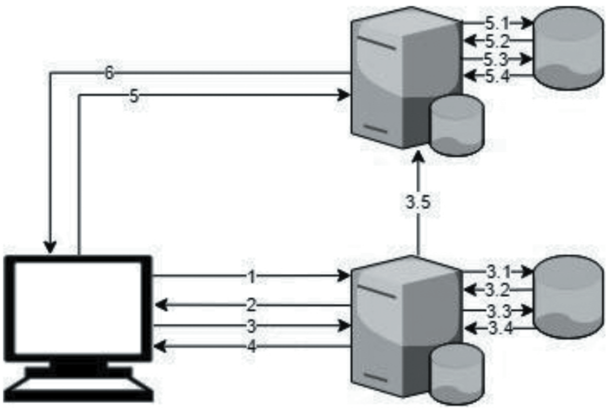

Figure 1. Proposed security protocol

- Step 1:

Represents the beginning communication between the client and the authentication server. The client initiates the communication by sending the following message:

```
{
"step":0,
"authToken":"",
"encryptedData":"",
"publicKey":""
}
```

In this message, all variables in the request are empty.

This step represents an indicator of what kind of request is required. Initialize requires generating of the temporary ticket or request verification of a user's credentials and creation of the approved ticket.

- Step 2:

This is the authentication server response. The server generates the unique id and a key pair for this client. Below, we show content of temporary ticket, forwarded to the client.

```
{
“authToken”:”3043b214-9fa0-49af-a5a5-00df64e
0275c”,
“publicKey”:”MIIBIjANBgkqhkiG9w0BAQEFAAO
CAQ8AMIIBCgKCAQEAzMDA7sVPsZeVRUFsJN
9H2RA81xtt0r78QCY2FibPLojovijsTlCJ+4rYSlz8cb
3IJjCtkrJoAJNLkevLoqDJ1s2TJYDulmyPDDOERq
oRF1Y7DDUG+a2R8+rdPvcnuaxxZRKmvahY8EY
smm8huEZVWviZyzzEiXqw3J6NmVl9N+Rw3Nmr
GjSfo2QlPjgdam7Xk99Ffu6qaI+WF/9No/qJgA66r-
yNmId5ip7pGwDaOPVd5D2aK34O6Rii6esa109tc-
839plYrsVUE6ei0N1I8QSK/kokap8eF/ksZ3AdiJ4+
DrjiqEwTA1+1QxCUwHlaMWNvxjFoxbTlObaVCl
PtaOYQIDAQAB”
}
```

- Step 3:

Step 3 has two parts. The first part is the user credential preparation. The password is hashed (SHA 512) and the salted.

```
{
“username”:”Admin”,
“password”:”bIOOk04/7vrmz6U68RN11JVPhcb17Y
iMAs14BqcWltHLRJ8r546ebqMBqVyB8orYdm865Y
L5vqrDPH3Ct7qRhw\u003d\u003d”
}
```

The next part is a packet prepared for the authentication server. Data makrs with “encryptedData” are encrypted data of previous step using server public key.

```
{
“step”:1,
“authToken”:”a1f355df-aa94-402b-b8aa-20ef00466
879”,
“encryptedData”:”k+Pg4YZy+cPySPL5Np8pP1nir
u3ssWeh3fYcjy8tPHMmGIw6/b1C3+UXD1h6ZyC/
Q7pwbrFzD7xbPfQrhqb83Hxz6lBxwBZd2PRyRo-
QCFZNyqbDqZQtYhT0TF7Vxo7eMZ0qt5Wa5Ks-
dao01lqpInNRIzFAfjHfhM6riBZy0x5zSE4XMSOB/
FAksw1gTXTOnbhQlLXTBaDevXFrueqhPtNulXb-
3czvsP3Dlyg1LUTIDjFh6zw3g6vEy3dZXM1O/Fym
0hVS+UtjBttJBQZjOqvLpYsDiSWtgFtKp1nQDu6X
3l+zlaA1KsXeU+qgoqGoxjZ02Za9pmtQl2uW5nR+
fKp/w\u003d\u003d”,
“publicKey”:”MIIBIjANBgkqhkiG9w0BAQEFAA
OCAQ8AMIIBCgKCAQEAqB1edQ4IL0oYYRP+
d5OuM5evyzHtSGCfC5VWgUMEFTFZiwBCuI
k943MmRzo3EPYy00fdVa3K0ZaFf3cYB7PXK9C
2aUZUKpsn/3pgQNwT0K+EF7E0wlnVbYDcgB
+CSaAx9vgqz8Ht0BxlAicf1CiYCdHfmVzhTaHw-
6zoCLYXh8q0SdmE4wMvgp/vRp/qhfmuHwvn-
1WiDq4B8Uc9pQQ36jzR3n8XxEnInkl7CnoF1VW-
MKzCj3SJOudVgiJj9OJdgU69PMI0jmHYN0ftdEi5D/
YYzJ948JkxhOnWbwzg4K0HQPwE9GoSXzez2lP-
1m7zLI0pnbHgKNFQoAz2wbdgHSmxpwIDAQAB”
}
```

- Step 4:

After receiving the previous packet, the authentication server checks the timestamp, and then take the private key which uses to decrypt the package for the unique ID of the temporary ticket. As decryption has been successful, the server checks all parameters in the package (if the username and password match with the saved ones in the database).

- Step 5:

The key which will be generated between client and database server, is use for AES 256 cipher in a CBC mode. In order to verify the validity of the ticket, a timestamp and a unique identifier of the approved ticket is also generated. Storing the data in the local database [6] summarizes part of generating the approval ticket.

The essential data which are sent to the database server are shown below:

```
{
 “token”:”a4715b64-624e-424b-b10a-70ddc768048f”,
 “userDateInfo”:{
   “ creationTime “:1519855280851,
   “ expirationTime “:1519898480851
 },
 “userDataInfo”:{
   “aesKey”:”51913B27B604DA0A”,
   “initVector”:”7374694CF770E038”,
 }
}
```

The database server is provided with the unique identifier of the approval ticket, the timestamp and the key with an essential initial vector. The whole packet is encrypted with a symmetric key which is exchanged in the same way the user had exchanged the keys in the previous steps. database server decrypts this ciphertext and stores it in a local database.

After reporting to the database server, the authentication server can send the following packet to the client, which is encrypted with the user's public key:

```
{
"token":"34a5d1af-7929-4458-8d4f-91ce2606ff7e",
"aesKey":"D5893D2B51CD8C8C",
"initVector":"5EA87175AA42C9DF",
}
```

Finally, the user can request executing any query from the database server . The look of the package which is sent:

```
{
"step":2,
"token":"978c951b-330d-4228-b596-421f4a2577c0",
"data":"q396FMbPV8czu8gIfshRgt4r/e20TpCNUNj
CPktVEtTfE40OhDyDnPwQzc+LpKUk"
}
```

Data represents encrypted query using AES cipher. Now, the database server returns encrypted data with the same symmetric key and an initial vector as a result of the query itself.

It is important to emphasize that this ticket has its own time duration and the database server checks it through a unique identifier of the approved ticket. It also finds which key to use for decryption.

7. THE SUGGESTED SECURITY PROTOCOL ANALYSIS

Since it is about the information of great importance, the protocol itself must be safe so that the attacker cannot reach them in real time, or that the resources the attacker uses are more valuable than the information they want to reach.

The attacker can reach the information by observing communication between the client and the authentication or the database server.

When the communication between the client and the authentication server is recorded, it can be possible reaching for the initial message which everyone can initiate and does not have any special importance, because it is created for each client independently and does not have any data which would reveal anything that would violate the security of the system. Regarding the server response that contains a unique identifier of the temporary ticket and a public key associated with that identifier. If an attacker wanted to exploit it in any way, he wouldn't be allowed to do so because the time of the temporary ticket for the authentication is set up to 1 second. After all, if an attacker wanted to decrypt the client's response to a server which is encrypted with a public key of the server, they would have to have a private key. Due to the inability of the private key to be reconstructed from the public key, the attacker only remains waiting for a public key to be repeated.

When the authentication was successful and all the communication paths between the client and the database server decrypted with symmetric cipher, the only thing the attacker can do at that moment is recorded as many messages as they could and then tries to decrypt it with the complete search or searching the keywords. Given that the AES key and the IV initial vector change after each successful authentication, the amount of encrypts the attacker encounters is very small for successful decoding in real time.

8. CONCLUSION

The goal of this paper is to develop own security protocol for the cryptographic protection of the database based on Kerberos. The need to achieve secure communication and encryption that will not affect the performance of the protocol is one of the main problems. In addition, the use of the username and password is the most appropriate authentication method. We proposed new protocol for key distribution and secure communication between client and database server. On this way, our database support multiuser encryption with different symmetric key and give solution for fast replacement used key for encrypting sensitive data in the database.

The implemented protocol must be based on the trusted third party, i.e. on the authentication server. The benefit of this authentication method is to achieve high level security of user credentials. The disadvantages is if the authentication server is not active, the entire system is inaccessible, due to the inability of the user to successfully authenticate and assign a ticket to work with the database server.

REFERENCES

[1] M. Milosavljević and S. Adamović, *Kriptologija 2*, vol. 1. Belgrade: Singidunum University, 2014.

[2] M. Veinović and S. Adamović, *Kriptologija 1*, vol. 1. Belgrade: Singidunum University, 2013.

[3] U. T. Mattsson, “Database Encryption - How to Balance Security with Performance,” *SSRN Electronic Journal*, 2005.

[4] J. Steiner, C. Neuman and J. Schiller, “Kerberos: An Authentication Service for Open Network Systems, ” , *Usenix Conference Proceedings*, 1988.

[5] S. Prema S. Rajeshwari, G. Kavitha “Input based encryption techinique in entrustment system,” *International Research Journal of Engineering and Technology (IRJET)*, 2018.

[6] S. Adamović and M. Veinović, “Analiza zaštite podataka u bazama podataka na lokalnom i serverskom nivou,” *TELFOR*, 2008.

[7] “MIT Kerberos Consortium - Protocol Tutorial”, *Kerberos.org*, 2018. [Online]. Available: https://www.kerberos.org/software/tutorial.html.