International Journal of Security and Its Applications
Vol.9, No.11 (2015), pp.429-438
http://dx.doi.org/10.14257/ijsia.2015.9.11.39

Research on Hadoop Identity Authentication Based on Improved Kerberos Protocol

Daming Hu, Deyun Chen*, Yuanxu Zhang and Shujun Pei

The College of Computer Science and Technology, Harbin University of Science and Technology, Harbin 150080, China
chendeyun@hrbust.edu.cn

Abstract

This paper researches the authentication mechanism of Kerberos protocol under HDFS, and points out the problems that identity authentication mechanism of Kerberos protocol faced in HDFS cluster environment: time synchronization, KDC security, dictionary attacks and denial mechanism. Aiming at these security problems, firstly, this paper provides an overview of the authentication process of the current Kerberos protocol under HDFS cluster environment; secondly, it modifies Kerberos protocol by using public key encryption and data signature mechanism; lastly, it provides the authentication process of improved Kerberos protocol in HDFS environment. Comprehensive analysis shows that both safety and time efficiency of the improved Kerberos protocol are improved compared with the existing identity authentication mechanism. It provides a more reliable and efficient identity authentication solution for HDFS cluster.

Keywords*: HDFS; Identity Authentication; Kerberos; Public Key Encryption; Digital Signature*

1. Introduction

Hadoop developed by Apache Foundation is a kind of distributed system architecture with PT level data storage and analysis capabilities. It has received extensive attention and application since its inception [1]. Facebook, Amazon and other well-known enterprises have adopted Hadoop cluster as the private cloud infrastructure of internal business processing. However, with the recognition and applying in practice, the problem of weak security mechanism is more and more prominent, and has an effect on the popularization and promotion of Hadoop platform.

The Hadoop platform is mainly composed of HDFS (Hadoop Distributed File System) and Mapreduce (Distributed Programming Model). HDFS, the open source version of Google File System, provides basic storage services of distributed files for Hadoop platform. [1]

As a new generation of file system its advantages are as follows: First of all, HDFS which can build high performance clusters by making full use of cheap business machines solved the problems that machine downtime and online expansion of machine nodes that traditional distributed file system may face in cluster. Secondly, HDFS provided the ability to localize data processing for parallel computing of Hadoop Mapreduce, and solved the bottleneck problem that data shared network bandwidth in traditional grid computing.

Deyun Chen* is the corresponding author.

ISSN: 1738-9976 IJSIA

In order to ensure the safety of Hadoop platform, reliable safety mechanism needs to be provided for HDFS, the basic part of Hadoop platform. The research will be carried out on authentication mechanism under HDFS cluster environment.

2. Architecture and Principle of HDFS

2.1 Architecture

From the view of architecture, HDFS uses a master-slave architecture model (Figure 1). A HDFS cluster generally sets up a Master node and a number of Slave nodes. Master node and Slave node respectively provide services for Client as NameNode and DataNode. Master node manages all the Slave nodes in HDFS cluster by using centralized control strategy [3]. DataNode communicates with NameNode interactively by heartbeat mechanism. As shown in Figure 1, in order to guarantee the identity safety of nodes in HDFS cluster, KDC (Key Distribution Center) can be introduced to this system as an independent security authentication mechanism.

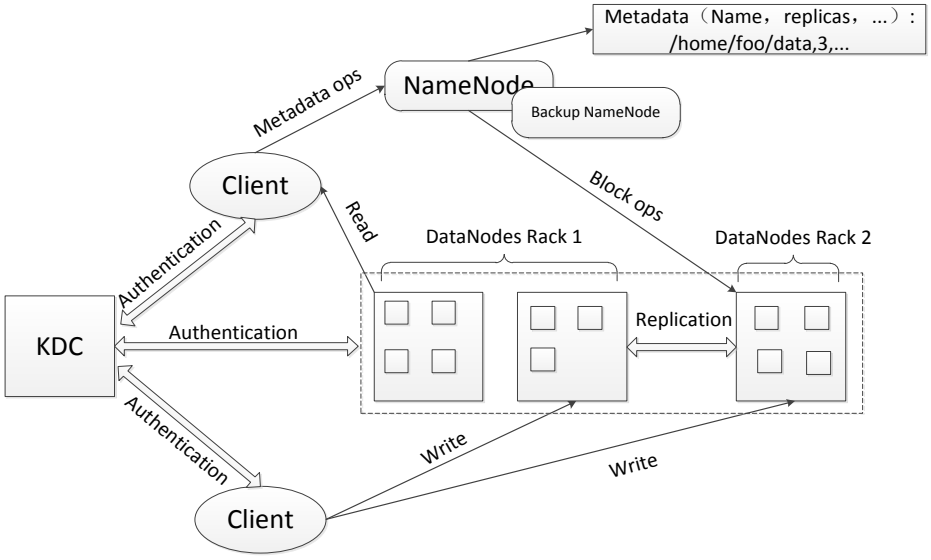

Figure 1. Architecture

Further discussions about NameNode and DataNode are as follows:

2.1.1 NameNode: NameNode is the management unit of HDFS, and is not used for storing data. On the one hand, NameNode manages and maintains the metadata (file directory tree, file index directory and so on) of the entire HDFS, and makes the information permanently stored in namespace images and edit logs that are on local disks in order to ensure the safety of the distributed file system; On the other hand, by receiving heartbeat information from DataNode, NameNode dynamically maintains a list of available DataNodes. It allocates available space on DataNode for blocks of file when the Client has storage requirements and controls blocks to store data according to established strategy in cluster.

2.1.2 DataNode: DataNode is in charge of the implementation of HDFS. The file blocks in DataNode are the basic storage units. DataNode has the following characteristics: First of all, in order to ensure the security of data, HDFS chooses reasonable amount of DataNodes and backups the blocks without distinction in accordance with the system configuration strategy. Secondly, if the files stored in a DataNode fail to reach the block size, they will not occupy the entire block space. Therefore, compared with traditional file system the usage of storage space in DataNode is improved obviously. Finally, each DataNode is in charge of the operation of creating, deleting, updating files, *e.g*. And it reports to NameNode about its node condition by heartbeat mechanism at a certain frequency.

2.2 Principle

HDFS reads and writes file data by using stream. It's especially suitable for the task whose data is only written once but read and analyzed more than once. The client needs to communicate with NameNode to get the information of the DataNode's position which it has file operations on. After that, the client can carry out operations of files [4].

The following part will discuss the client's operations of reading and writing files:

2.2.1 Reading file: Client's specific processes of reading files are shown in, Figure 2

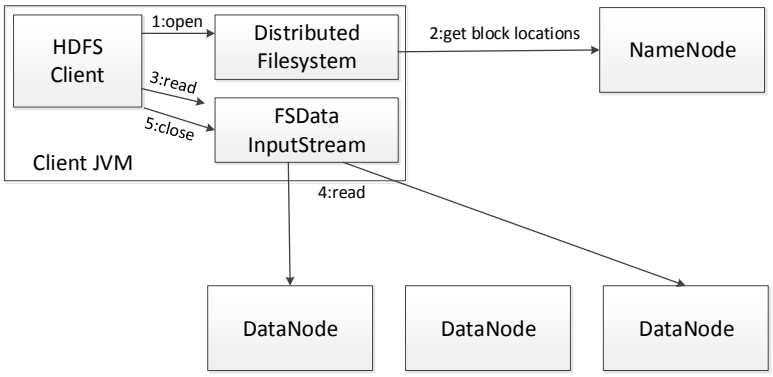

Figure 2. Reading File

(1) HDFSClient opens object DistributedFilesystem. The object can request file block information from progress in NameNode by RPC (Remote Procedure Call).
(2) HDFSClient sends reading operation to object FSDataInputStream, and prepares to receive data from FSDataInputStream.
(3) FSDataInputStream sends read operation to DataNode. Its function is to read specific data and transmit them to HDFSClient.
(4) After HDFSClient gets all the data, it sends close operation to FSDataInputStream .

2.2.2 Writing File: Client's specific processes of writing files are shown in Figure 3.

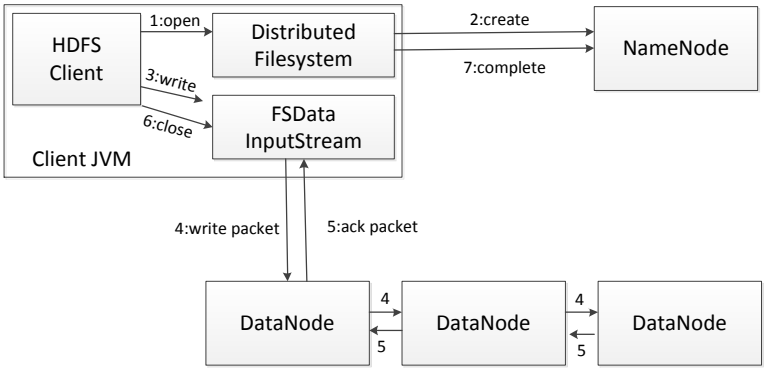

Figure 3. Writing File

(1) HDFSClient opens object DistributedFilesystem. The object can request the progress in NameNode to create file information that data will be written by RPC (Remote Procedure Call).
(2) HDFSClient sends writing operation to object FSDataInputStream, and prepares itself to send data to FSDataInputStream
(3) FSDataInputStream sends writing packet operation to DataNode and transmits data to it. At the same time, the DataNode transmits file information just written to the selected backup nodes in cluster that system sets according to the security backup strategy to complete the backup of files.
(4) DataNodes confirm that they have finished file writing successively and then send the information of ack packet to FSDataInputStream.
(5) HDFSClient sends close operation to FSDataInputStream. After that, DistributedFilesystem sends a complete operation to NameNode.

3. Analysis of Kerberos Protocol

At the beginning of the design, there was no authentication mechanism in Hadoop cluster assuming that the cluster was in a trusted domain. With the popularity of Hadoop application,security problems got more and more serious, and Hadoop designers had gradually realized that. Aiming at the security flaws in Hadoop, a third party security authentication mechanism based on Kerberos protocol was introduced in to make sure the credibility between communication nodes since Hadoop1.0.0 version [5].

3.1 HDFS Authentication Mechanism Based on Kerberos Protocol

Kerberos was originally developed by MIT. The latest version was Kerberos V5 protocol. The protocol can provide credible identity authentication mechanism for communicating nodes which are in unsafe network environments [6, 7]. The basic principles of Kerberos in HDFS environment: To set up an authentication center KDC (Key Distribution Centre) to keep usernames, passwords and other information of clients, NameNodes and DataNodes in cluster and to offer services of identity authentication and authorization. KDC is composed of two logically independent servers: Authentication Server (AS) and Ticket Grant Server (TGS). In the cluster, firstly, any user who wants to apply for service needs to communicate with AS to get Ticket Grant Ticket (TGT). Secondly, it gets Ticket for service by communicating with TGS using TGT. Finally, the user communicates with the node that provides services to get services by Ticket [8, 9].

The specific implementation processes of Kerberos protocol under HDFS are shown

in Figure 4.

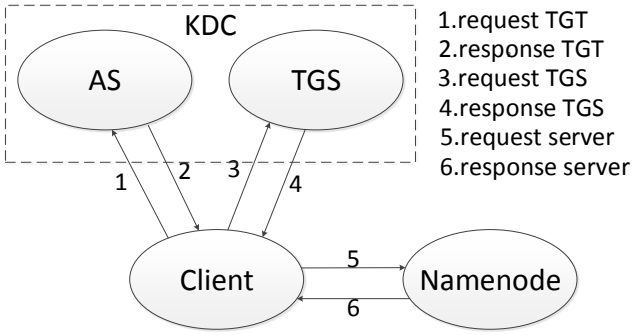

Figure 4. Kerberos Authentication Process

3.2 Security Analysis of Kerberos Protocol under HDFS

The introduction of Kerberos protocol solves the following security problems in original HDFS cluster.

(1) Because of the dynamic scalability of Hadoop cluster, illegal user can disguise as a DataNode server and join to the cluster to receive data information from NameNode.
(2) Illegal user disguises as authorized user by altering data package to request service sources
(3) In an unsafe network environment, illegal user can intercept the exchange project of datagram and disturb the normal operation of NameNode or DataNode by replay attacks.

The Kerberos protocol provides identity authentication mechanism for HDFS, but there's still limitation. The safety problems are as follows:

(1) The problem of time synchronization in Hadoop cluster: In the process of Kerberos identity authentication, it's necessary to contrast timestamp to judge the authenticity of user's identity which requires the internal network of Hadoop cluster to have high ability of clock synchronization. Obviously it's difficult to achieve in Hadoop cluster that is composed of cheap commercial computers.
(2) The security problem of KDC: Because of the Kerberos server stores all the passwords and other related information of clients, NameNodes and DataNodes. Once KDC is broken by a malicious user, it will cause a devastating blow to the entire Hadoop cluster.
(3) The Problem of Dictionary Attack: In the process of Kerberos certification, AS server doesn't verify user's identity directly, but does it via the packet included TGT and encrypted by client secret key Kc which is postback information. Only the user knows Kc can get TGT, and then conduct subsequent authentication steps. If a malicious user collects a number of TGT information, user's password Kc is possibly cracked[10].
(4) The problem of denial mechanism: Due to public key technology is not introduced in Kerberos protocol, so it does not provide digital signature for transmitting information, and cannot realize denial mechanism of information transmitting in authentication process.

In order to eliminate these limitations of Kerberos authentication mechanism, this paper will make some appropriate changes on Kerberos protocol in the framework of

Hadoop authentication mechanism. By bringing in asymmetric encryption for Kerberos protocol, it can make full use of the features of asymmetric key mechanism to solve the problems of Hadoop cluster authentication listed above.

4. The Improvement and Implementation of Kerberos

Asymmetrical encryption system is also called public key cryptosystem. It is composed of public key and private key that are generated by specific algorithm. The public key is public, while the private key which is the critical part of the asymmetrical encryption system is not open. In public key cryptosystem, data encrypted by public key can only be decrypted by private key. Similarly, data encrypted by private key must be decrypted by public key. Compared with symmetric key system, asymmetric key mechanism has longer key digits and separates public key and private key. As a result, it can provide more secure encryption service and is widely used in data encryption and data signature [11].

Data signature provides verification of fingerprint level by relevant processing on protected data. It usually includes generating summary of data and encrypting the summary. Digital signature technology is based on public key cryptosystem: Before data transmission the sender uses HASH function to get a summary, and then uses the private key to encrypt the summary. The summery together with the original data is sent to the receiver. The receiver decrypts the signature information by the sender's public key, generates a summary of the original data by corresponding HASH function and contrasts the summery with the decrypted one. If there is no different between two summaries, the data will be received. Data signature is widely used in ensuring the integrity of information transmission, identity authentication of the sender and non-repudiation of electronic trading [12].

4.1 Client Request NameNode

The specific processes of the request are shown in Figure 5, and the symbols used in certification process are as follows [13-16]:
Client: Service requester
KDC: Key Distribute Center
AS: Authentication Server
TGS: Ticket Grant Server
NameNode: The server in HDFS named NameNode
Kc,as: The symmetrical key used between Client and AS, the same as Kc,tgs and Kc,n
KcPr: Private key of client, the same as KasPr, KtgsPr and KnPr
KcPu: Public key of client, the same as KasPu, KtgsPu and KnPu
IDc: The identity of client, the same as IDn
Timestamp: Timestamp
Random: Random number
Ac,tgs: The identification between AS and TGS, the same as Ac,n
Ack: *Ack*nowledgement
TGT:Ticket Grant Ticket
Tn:The ticket can access NameNode
SKcPr{}:Sign the data by KcPr, the same as SKasPr{}, SKasPr{}, KtgsPr{}and SKnPr{}

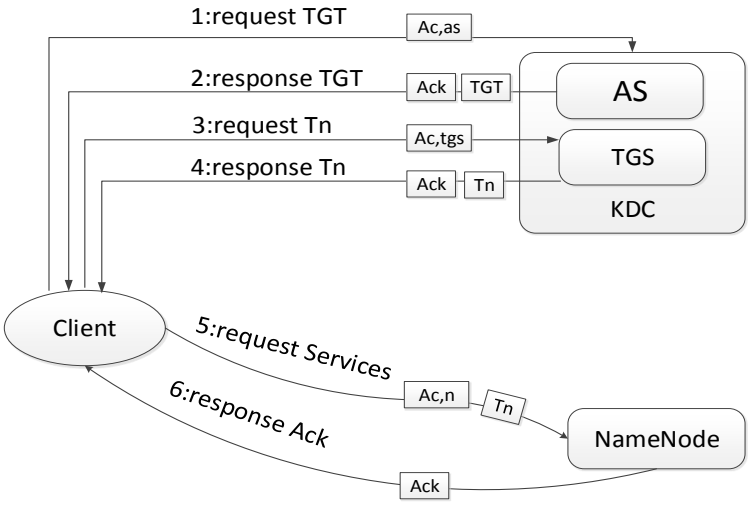

Figure 5. Improved Kerberos Authentication Process

(1) Client Request AS
{Ac,as}
Ac,as =KasPu { SKcPr {IDc, IDtgs, Kc,as, Random, Timestamp} }

Client signs IDc, IDtgs, Kc,as, Random and Timestamp by the private key KcPr,and then encrypts them by the public key of AS ,finally generates the packet of {Ac,as}. It is sent to AS as the credential for requesting TGT.

(2) AS Response Client
{Ack, TGT}
Ack= Kc,as {IDc, Random }
TGT=KtgsPu{SKasPr{IDc, Random, Lifetime }}

AS uses its private key KasPr to decrypt Ac,as, client's public key KcPu to verify the integrity of Ac,as and tests the validity of timestamp to prevent the replay attacks. Only if the above processes are successfully validated, AS generates TGT and Ack for Client. The processes of generating TGT and Ack are as follows: Firstly, to sign the information in TGT (including IDc, Random,Lifetime) by AS' private key KasPr and to encrypt them by TGS' public key. Afterwards, to encrypt IDc and Random included in Ack by Kc,as. Finally, to combine TGT and Ack as postback packet of {Ack,TGT}. Description: Kc,as, IDc and Random included in Ack are all from the initial decrypted message Ac,as.

(3) Client Request TGS
{Ac,tgs, TGT}
Ac,tgs= KtgsPu{SKcPr{IDc, IDn, Kc,tgs, Random, Timestamp}
Client uses symmetric key Kc,as saved locally to decrypt Ack and to compare the required IDc and Random with the native duplicate in order to prove the correctness of the message. After the above processes are successfully validated, client then generates symmetric key Kc,tgs and packet sent to TGS server. The packet is composed of Ac,tgs and TGT. Ac,tgs includes IDc,IDn,Kc,tgs,Random and Timestamp that are signed by client's private key KcPr and encrypted by TGS' public key KtgsPu.

(4) TGS Response Client
{Ack,Tn}
Ack= Kc,tgs {IDc,Random}

Tn=KnPu{SKtgsPr{ IDc ,Lifetime }}

Firstly, TGS decrypts the packet of Ac,tgs by its private key KtgsPr, verifies signing messages of Ac,tgs by client's public key KcPu and tests the effectiveness of timestamp to prevent replay attacks. Then TGS uses its private key KtgsPr to decrypt TGT, AS' public key KasPu to verify the signing information of TGT and checks whether the lifetime of TGT is in force.If the above processes are successfully validated, TGS then generates ACK and Tn for client to access NameNode. The steps of the generation of ACK and Tn are as follows: Firstly, to sign IDc and lifetime included in Tn by TGS' private key KnPr and encrypt them by NameNode's public key. Secondly, to encrypt IDc and Random included in Ack by Ac,tgs. Finally,to combine TGT and Ack as postback packet of { Ack,Tn}.

(5) Client Request NameNode
{Ac,n,Tn}
Ac,n= KnPr{SKcPr{IDc, Kc,n, Random, Timestamp}}

Client uses symmetric key Kc,tgs saved locally to decrypt Ack and to compare the required IDc and Random with the native duplicate in order to prove the correctness of the message. After the above processes are successfully validated, first of all, client generates symmetric key Kc,n to prepare for data communication with NameNode after connection establishment. Afterwards, client signs IDc, Random and Timestamp by its private key KcPr and encrypts them by the public key of NameNode KnPu to generate Ac,tgs from client to TGS. Finally, client packages Ac,tgs and Tn to form packet of {Ac,n, Tn} sent to NameNode.

(6) NameNode Response Client
Kc,n{Random}

NameNode decrypts Ac,tgs by its own private key KnPr, verifies signing messages by client's public key and tests the effectiveness of timestamp to prevent replay attacks. Then NameNode uses its private key KnPr to decrypt Tn, TGS' public key KtgsPu to verify the signing information of Tn and checks the valid identification of ticket lifetime in Tn. There's a need to compare IDc decrypted from Ac,tgs and Tn respectively. If the above processes are successfully validated, NameNode regards the client as the credible client that passes KDC authentication. Finally, NameNode encrypts Random by Kc,n and sends back to client. Client decrypts the message by Kc,n and compares it with the random preserved itself to verify the identity of the server.

These are the whole processes of identity authentication.

4.2 Client Request DataNode

This part is the same as Client Request NameNode, and the paper does not descript it again.

5. Analysis of Improved Kerberos Protocol

The paper will analyze the process of identity authentication of improved Kerberos protocol under HDFS from safety and time efficiency.

5.1 Analysis of Safety

It solves potential safety hazard of the original authentication mechanism by introducing public key encryption and data signature mechanism into Kerberos protocol. The security analysis is as follows:

(1) The improved Kerberos protocol based on public key encryption system uses distributed keys management strategy. Users keep the private keys themselves and can get the public keys from cluster. Therefore, KDC needn't to save the secret information like passwords intensively. Due to this change, it greatly improves the security of KDC. Even if the KDC is invaded, the attacker can't get the user's private key and impersonate the user to obtain service.
(2) Public key encryption uses longer key compared with the symmetric key encryption. Theoretically, if the key is more than 1024 bit, it will be safety. The improved protocol can defense dictionary attack effectively.
(3) Public key encryption and private key signature are used in the processes of sending and receiving authentication messages of the user, KDC server, NameNode and DataNode. On account of the privacy of private key, the identity of the sender can be verified.
(4) The requirements of time synchronization are reduced after the improvement of the authentication protocol. The data encrypted by private key can only be decrypted by private key. In addition, public key encryption system is difficult to broken. As a result, timestamp is used to judge the validity of ticket and prevent replay attacks as auxiliary in improved authentication protocol [17].

5.2 Analysis of Efficiency

Symmetric key encryption needs shorter time than public key encryption for the same data. However, KDC bottleneck caused by identity authentication needs to be considered in large clusters. Specific efficiency analysis is as follows:

(1) The improved Kerberos protocol removes the redundancy information such as IP address transmitted between client and KDC and retains IDc and Random merely. Less data makes encryption and decryption more efficient in authentication process.
(2) The improved Kerberos protocol does not abandon the symmetric encryption mechanism absolutely, but makes an improvement. Firstly, it makes KDC liberated from generating symmetric key that the clients are responsible for doing it. So even if a large number of clients need to request for KDC, it won't occupy KDC's limited computing capability to generate symmetric key. Secondly, the client sends symmetric key information encrypted by public key encryption to KDC so that the information can only be decrypted by KDC. KDC uses the symmetric key to encrypt Ack information and client decrypts Ack by its own symmetric key.

In a word, the design achieved the balance between security and efficiency. It provides a more reliable and efficient identity authentication solution for HDFS cluster.

6. Conclusion

The authentication mechanism of Hadoop varied from nothing to Kerberos protocol. According to the features of Hadoop cluster, this paper carries on research about identity authentication based on Kerberos Protocol. Firstly, it analyzes the security mechanism of Kerberos protocol and points out the problems such as time synchronization, KDC security, dictionary attacks and denial mechanism of symmetric key system of the original Kerberos protocol in Hadoop cluster. Secondly, aiming at the related question, it proposes an improved strategy of Kerberos protocol based on public key encryption system. Finally, it verifies the feasibility of the improved protocol by specific analysis. Of course, authentication is just one of the security problems of Hadoop cluster. Hadoop cluster faced a series of security problems including access

control, security of data storage, *etc*. These problems need research and settlement in the future.

Acknowledgments

This study is supported by high and new technology industry foundation of Harbin Grant by No. 20120056.

References

[1] L. Junzhou, J. Jiahui, S.Aibo, D. Fang, "Cloud computing: Architecture and key technologies. Journal of Communication", no. 7, **(2011),** pp 4-21.
[2] H. Shukui, HDFS Hadoop and MapReduce architecture analysis. Post Design Technology, (**2012**), pp. 37-42.
[3] X. Chunling, Z. Guangquan, " Comparison and analysis of distributed file system Hadoop HDFS and traditional Linux file system FS", Journal of Soochow University, no. 4, **, (2010)**, pp. 6-9.
[4] W. Feng, L.Baohua. "Analysis of Hadoop distributed file system model. Telecommunications Science", no. 8**, (2010)** , pp. 95-99.
[5] . Huangqi, S. Cheng. "The design of security mechanism based on HDFS",Computer Security, no. 12, **(2010)**, pp. 22-25.
[6] F. Butle, I Cervesato, A D Jaggard, *et al*. "A Formal Analysis of Some Properties of Kerberos 5 Using MSR ", the Fifteenth IEEE Computer Security Foundations Workshop. **(2002)**.
[7] The Kerberos Network Authentication Service (V5):InternetRPC4120, July **(2005)**.
[8] MA Yuan. "Study on the platform security mechanism of cloud computing based on Hadoop"., Information and Communication Security, no. 6, **(2012)**, pp. 89-92.
[9] Y. Xiaojie. "Security analysis of Kerberos protocol", Computer Security, **(2013)**, pp. 17-21.
[10] Z. jiqiang, T. yuanxin, H. lilong. "Network Vulnerability Assessment Using Network Service Correlation", Journal of Harbin University of Science and Technology, vol. 18, no. 4, **(2013)**, pp. 79-83.
[11] L. Chengkai. "*Computer cryptography"*, BeiJing: Tsinghua University Press, **(2003)**, pp. 363-370.
[12] Z. Xiang. "Overview of digital signature", "Computer Engineering and Design", no. 2,**(2006),** pp. 195 -197.
[13] Z. Xiao. "The improved PKI protocol based on Kerberos. Information Technology", no.10, (**2011),** pp. 211-213.
[14] L. Zhuang, G. Heqing. "Research on Kerberos distributed authentication method based on public key", Computer Engineering and Applications, **(2006)**, pp. 121-124.
[15] S. Yeqin, C. Jianping, G. Xiang, "Improved Kerberos single sign on Protocol"., Computer Engineering, no. 27, **(2011)**, pp. 109-110.
[16] R. Marin-Lopez, F Pereniguez-Garcia, Y. Ohba, *et al*. "A Kerberized Architecture for Fast Re-authentication in Heterogeneous Wireless Networks", Moblile Networks & Applications, vol. 15, no. 3, **(2010)**, pp. 392-412.
[17] S. Mingsong, Z. Xiuna, S. Xibei *et al*. "Design of Campus Network Live video System Based on Cloud Computing", Journal of Harbin University of Science and Technology, vol. 12, no.1, **(2012)**, pp. 58-67.

