International Journal of Engineering Research & Management Technology ISSN: 2348-4039
March-2016 Volume 3, Issue-2
Email: editor@ijermt.org www.ijermt.org

KERBEROS USING PUBLIC KEY CRYPTOGRAPHY IN AUTHENTICATION PROTOCOL

Ms.Dhowmya Bhatt[1]
Madan Kumar[2]
Mradula Sharma[3]
Assistant Professor
Department of Information Technology
SRM University NCR Campus

ABSTRACT: *Kerberos is a trusted third party authentication protocol based on symmetric key cryptography. Adding Kerberos to the network can increase the overall security available to users .This paper analyzes how Kerberos authentication standard can be extended to support public key cryptography (PKC). The paper also addresses three protocols for PKC enhanced Kerberos and their performance using reference qualitative results from other research paper. It also addresses some security issues associated with PKC in traditional Kerberos framework.*

KEYWORDS: Kerberos protocol, performance analysis, Public Key Cryptography (PKC).

1. INTRODUCTION

Kerberos is an authentication service designed to allow clients to access servers in a secure manner over a network. One of the most important achievements of Kerberos is secure authentication. Windows 2000, Windows XP and Windows Vista use Kerberos as their default authentication method. The main limitation of Kerberos is its scalability. As the number of new
Applications and Internet is growing explosively, authentication schemes are needed which can be scaled to easily handle millions of principals within a single realm of trust. The papers we analyzed address this concern through the integration of public key cryptography with traditional Kerberos authentication. Part II of the paper presents the History and the basic principle of Kerberos. Part III describes motivation and different protocols for public-Key enhanced Kerberos. The performance of different protocols is analyzed in part IV. Part V and VI describe related security issues and conclusion. To analyze the quantitative performance differences between different protocols we have taken reference data from [1].

2. PRINCIPLE AND AUTHENTICATION PROCESS OF KERBEROS

Kerberos was originally developed at the Massachusetts Institute of Technology by Steve Miller and Clifford Newman. It was basically developed to protect Network Services provided by MIT's Project Athena.

2.1 Authentication Process of Kerberos

Kerberos divides the whole world into multiple realms. A Kerberos realm is a set of managed nodes that share the same Kerberos database. A realm represents a networked collection of client workstations, application servers and a single master key distribution centre (KDC). The KDC maintains a database of secret keys for each entity on the network. Knowledge of this key serves to prove an entity's identity. For communication between two entities, the KDC generates a session key which they can use to secure their interactions. Kerberos is a mechanism, which operates on a ticket granting methodology using a ticket-granting server (TGS). The client must first call the local KDC and request for a Ticket Granting Ticket

(TGT). The TGT, which consists of the Client ID, the Network Address of the client, the Ticket Validity period and the client/TGS session key, is encrypted using the Secret Key of the TGS. The client uses the TGT to seek a Service Ticket from the Ticket Granting server. The Service Ticket comprises of the Client ID, Client Network Address, Validity period and the Client/Server session key. The client now uses the Service Ticket to access the server. The Server decrypts the Service Ticket using its own secret key and sends the message to confirm its identity to the client. The client then decrypts the message using the client/server session key and checks the Time Stamp update. If the Time Stamp is properly updated, then the client can trust the authenticity of the server. Thus it can start requesting services from the server. This process of Ticket Granting prevents the usage of GMU-ECE 646 Fall use to secure their interactions. Kerberos is a mechanism, which operates on a ticket granting methodology using a ticket-granting server (TGS). The client must first call the local KDC and request for a Ticket Granting Ticket (TGT). The TGT, which consists of the Client ID, the Network Address of the client, the Ticket Validity period and the client/TGS session key, is encrypted using the Secret Key of the TGS. The client uses the TGT to seek a Service Ticket from the Ticket Granting server. The Service Ticket comprises of the Client ID, Client Network Address, Validity period and the Client/Server session key. The client now uses the Service Ticket to access the server. The Server decrypts the Service Ticket using its own secret key and sends the message to confirm its identity to the client. The client then decrypts the message using the client/server session key and checks the Time Stamp update. If the Time Stamp is properly updated, then the client can trust the authenticity of the server. Thus it can start requesting services from the server. This process of Ticket Granting prevents the usage of GMU-ECE 646 Fall 2007 password mechanism, which can be easily hacked by an intruder

2.2 Adding Public Key Cryptography to Kerberos

Kerberos has been using secret key Cryptography ever since its inception. Although public key crypto-system requires calculations that are computationally expensive, it can eliminate some of Kerberos protocol limitations First of all there is a need to maintain the secret key between every user and the KDC, between every KDC and KDC and between the Application server and the KDC [1]. The Ticket given by the TGS is encrypted by the user's Secret key. So, the KDC needs to remember every user's secret key, which is a problem and has an impact on Kerberos scalability. Adding Public Key cryptography will add a completely new dimension to Kerberos scalability as it eliminates the need to establish a large number of shared secrets.

Secondly, if there is an unauthorized entry into the KDC's database, even if it is a read-only access, the privacy of the user's secret and the security of the KDC will be compromised. But with public key cryptography to violate the security one has to obtain the write access to the database. Secondly, if there is an unauthorized entry into the KDC's database, even if it is a read-only access, the privacy of the user's secret and the

2.3 Different Public Key Extensions to Kerberos

There are 3 major proposals to add Public Key Cryptography to Kerberos [1]:

- Public key Cryptography for Initial Authentication in Kerberos (PKINIT).
- Public Key Cryptography for Cross Realm Authentication in Kerberos (PKCROSS).
- Public Key Utilizing Tickets for Application Servers (PKTAPP).

2.4 ***PKINIT***

PKINIT describes how Public Key Cryptography can be added to Kerberos in the Initial Authentication stages. Microsoft, Cyber safe and Heimdal adopted it in their implementations of Kerberos [5]. Figure 1 illustrates the flow of the PKINIT protocol.PKINIT requires the verification of the initial request message sent by the client to the local KDC. Any authentication requests made by intruders masquerading as legitimate users will be denied. PKINIT also requires that the TGT and the session key be encrypted so that the user's credentials will remain confidential. Also, PKINIT suggests the storage of the user's secret key in the KDC so as to allow the user to generate its own session key, which decreases the load on the KDC considerably.

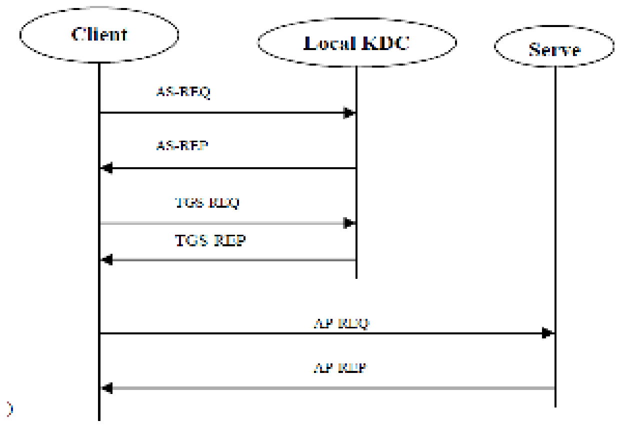

Figure1: The PKINIT transaction flow [1]

Authentication Process

- Initial TGT request to KDC. Public keys are used to authenticate client and KDC
- Request a ticket to application server using secret key encryption
- Authenticate to the remote application server using secret key encryption

2.5 PKCROSS

PKCROSS is a logical extension of PKINIT. If a client wants certain services form a server, which is located remotely, then there has to be an authentication procedure, which relates the KDC of the client (Local KDC) with the KDC of the KDC of the server (remote KDC). It simplifies the multiple realm authentications. Figure 2 illustrates the flow of PKCROSS authentication. The user needs to request a cross realm TGT request from its Local KDC. The messages exchanged between local and remote KDC is similar to PKINIT. Here the local KDC acts as a client. The local KDC sends a request comprising of the PKCROSS flag set to the remote KDC. The remote KDC replies with a PKCROSS ticket and trusts the local KDC to issue the remote realm TGT to its client on behalf of the remote KDC.

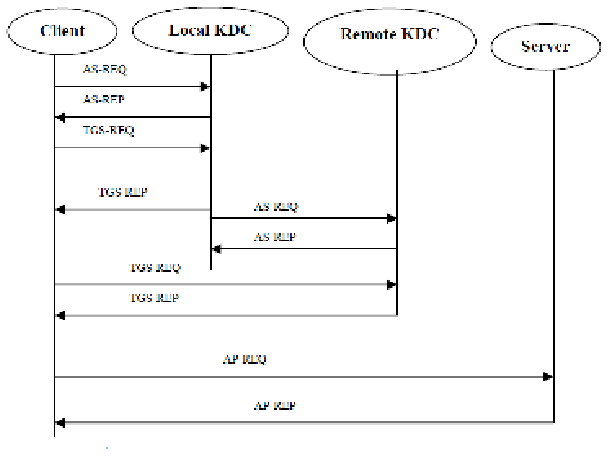

Figure 2: The PKCROSS transaction flow[1]

Authentication Process

- Initial secret key-based TGT request to KDC
- Request a ticket to the remote TGS
- Public key authentication between KDCS to establish a session key
- Request a ticket to the remote application server (all secret key encryption)
- Authenticate to the remote
  Application server (all secret key encryption)

2.6 *PKTAPP*

The PKTAPP scheme is the one that looks into the performance aspects. The main aspect that can be observed from the previous two schemes is the role played by the KDC. The KDC issues all TGT and server tickets to the local KDC and also to the remote KDC. Most of the authentication transactions have to transit the KDC [1]. This creates a performance bottleneck at the KDC. PKTAPP provides a scheme where there is only one path of communication i.e. between the client and the server. The client sends the request message that consists of the client's certificate chain and the identity of the service ticket. The server immediately responds to the request with the response signal that consists of the server's chain and the session key that is encrypted with the server's private key. The PKTAPP authentication scheme mainly reduces the number of authentication steps. Figure 3 illustrates the PKTAPP authentication message exchange.

Authentication Process

- Service request to the application server. Public keys are used to authenticate client and server.
- Authenticate to the application service

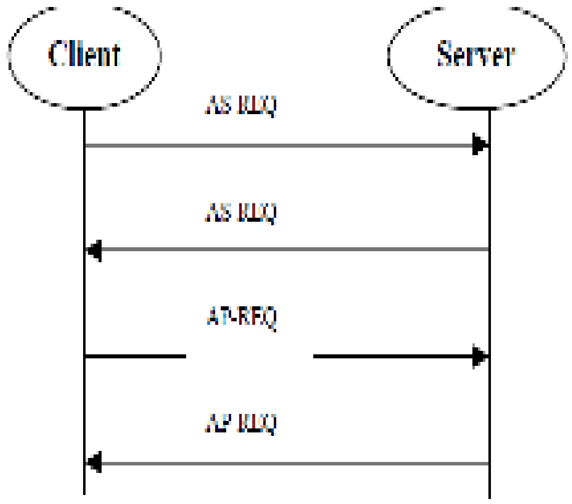

Figure 3: The PKTAPP transaction flow [1]

3. PERFORMANCE ANALYSIS OF DIFFERENT PROTOCOL

The main performance comparison is observed between PKCROSS and PKTAPP. Closed, class-switching queuing models are used to demonstrate the quantitative performance which compares the response time performance. Close queuing networks represent each system resource with a queuing discipline and a stochastic service distribution A multi-realm Kerberos environment is modeled to observed how the following parameters affected the throughput and response times of public key based authentication transactions:

- Number of realms in the Kerberos environment,
- Number of application servers per realm,
  Loads on application servers and key distribution centers, and
- Network delay.

Two assumptions were taken before all operations. Firstly, the client and KDC must present certificates for authentication. Secondly, the remote server must validate the certificate signed by the local CA and certificate signed by the remote CA. All encryption operations are configured with 1024-bit RSA keys or standard DES. Table 1 shows the summary of encryption operations performed for PKCROSS and PKTAPP authentication transactions

Table 1: Encryption Operations [1]

		Number of Operations		
	Transactions	**Private Key**	**Public Key**	**Secret Key**
PKCROSS	Client	0	0	7
	Local KDC	2	3	5
	Remote KDC	1	4	4
	Application Server	0	0	3
	Totals	*3*	*7*	*19*
PKTAPP	Client	2	3	3
	Application Server	1	4	4
	Totals	*3*	*7*	*7*

From the table it can be summarized that for authentication to a single application server, the no of private and public key operation for PKTAPP and PKCROSS are same. But PKCROSS requires more secret key operations. The calibrated model performance was investigated with an increased number of application servers.

Figure 4 shows the comparative performance- transaction response time plotted as a function of throughput for PKTAPP and PKCROSS protocols as the number of application servers are increased.

The transaction rates for both protocols were increased until the overall response time became unstable and grew rapidly Although for client authentication with remote application servers, PKTAPP requires fewer messages than PKCROSS, the cross-realm performance is better with PKCROSS than PKTAPP for networks with two or more application servers in a remote realm. Under PKCROSS, the KDC remains in the saturation point regardless of the number of remote realm application servers participating in the authentication. PKTAPP suffers from sending long public-key messages and performing expensive public-key calculations for authentication with multiple application servers. PKCROSS is more efficient as it uses public key cryptography only once between local and remote key distribution centers and then it uses more efficient secret key cryptography for subsequent communication with application servers . Thus from the analysis we can conclude that although PKTAPP is a significantly better performance choice for a single remote server, PKCROSS is significantly more stable for anything greater than two servers in the remote realm .

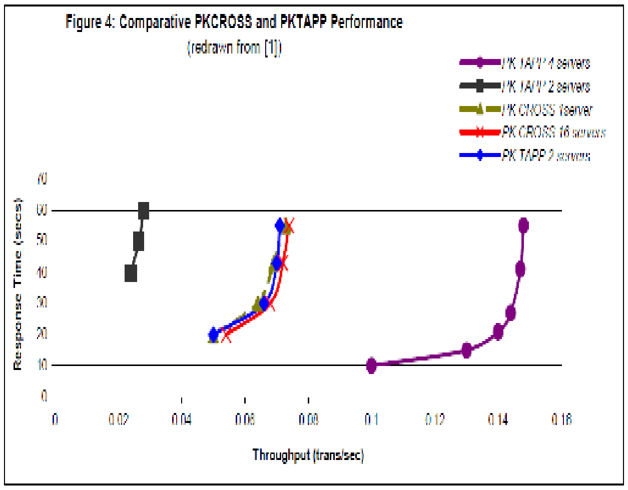

Figure 4: Comparative PKCROSS and PKTAPP Performance

4. RELATED SECURITY ISSUES

The security of private keys and the wide spread availability of their corresponding public keys are the main security issues in public key cryptography. The largest security issue associated with incorporating public-key cryptography in Kerberos involves expanding the scope of trusted entities to include the entire public key infrastructure (PKI) . Well secured public key infrastructure is required, otherwise man-in-the-middle attack could happen. If this is the case, an attacker capable of modifying the traffic between two communicating users can achieve unauthorized decryption of their public-key encrypted messages. Fortunately, using the X.509 Certificate Authorities supported by PKINIT, PKCROSS and PKDA can provide a trustworthy public-key infrastructure without introducing a liability into the Kerberos trust model.

5. RESULTS

We studied three public-key enabled Kerberos protocols-PKINIT, PKCROSS, and PKTAPP. All of them support public key cryptography at different stages of the Kerberos framework. They all improve Kerberos security and scalability by utilizing trustworthy public key infrastructures and simplifying key management system.

PKINIT specification defines how public key cryptography can be used to secure the initial authentication procedure. PKCROSS improves the scalability of Kerberos in large multi-realm networks where many application servers may be participating in the authentication process. PKTAPP enhances Kerberos scalability by distributing the authentication workload from the centralized KDC to the individual principals on the network. According to the research by PKCROSS is more favorable than PKTAPP when there is more than one application server in the remote realm. The complexity of public key computations and length of its messages makes PKTAPP generally less efficient than the simpler PKCROSS protocol for cross realm authentication. These findings can be used as a guideline for high-level protocol that combines both PKTAPP and PKCROSS to improve performance.

6. CONCLUSION

Thus this paper is an attempt to throw light on Kerberos authentication protocol and its significance in cryptography. This paper analyzes how Kerberos authentication standard can be extended to support public key cryptography (PKC). The risks and troubles related to Kerberos are discussed. We hope that readers will be able to get overview of Kerberos using Public Key Cryptography.

7. REFERENCES

1. Performance of public-key-enabled Kerberos authentication in large networks Harbitter, A.H.; Menasce, D.A. *Proceeding 2001 IEEE Symposium on Security and Privacy. S&P 2001, 2001,p 170-83*
2. [Cryptography and Network Security: Principles and Practices' By William Stallings, Prentice Hall; Fourth Edition, 2005.
3. Misplaced trust: Kerberos 4 session keys Dole, B. (Sun Microsyst., Mountain View, CA, USA); Lodin, S.; Spafford, E. Source**:** *Proceedings. 1997 Symposium on Network and Distributed System Security (Cat. No.97TB100100), 1997, p 60-70 GMU-ECE 646 Fall 2007 6*
4. Cryptography' By Bruce Schneier, john Wiley & Sons, Inc. Second Edition.
5. Public-key cryptography extensions into Kerberos Downnard, Ian (Naval Research Laboratory) Source*: IEEE Potentials, v 21, n 5, DEC/JAN, 2002, p 30-34.*
6. Authentication for distributed systems Woo, Thomas Y. C. (Univ of Texas, Austin, TX, USA); Lam, Simon S. Source: *Computer*, v 25, n 1, Jan, 1992, p 39-52
7. Research and realization of authentication technique based on OTP and Kerberos Cheng, Xiao-Rong (School of Computer Science and Technology, North China Electric Power University); Feng, Qi-Yuan; Dong, Chao; Zhang, Ming-Quan
8. Massachusetts Institute of Technology, *Kerberos: The Network Authentication Protocol.* <http://web.mit.edu/kerberos/www/>, 2000
9. Performance analysis of the Kerberos protocol in a distributed environment El-Hadidi, M.T.; Hegazi, N.H.; Aslan, H.K.; Computers and Communications, 1997. Proceedings., Second IEEE Symposium on 1-3 July 1997 Page(s):235 - 239 Digital Object Identifier 10.1109/ISCC.1997.616004
10. 'Applied Cryptography' By Bruce Schneier, john Wiley & Sons, Inc. Second Edition.
11. Kerberos Manuals on the HP documentation website: http://www.docs.hp.com/hpux/internet/index.html#Kerberos
12. The ITRC Website at http://itrc.hp.com where there is a wealth of information available for all HPUX products, regarding maintenance, support, training and education.
13. Harkins, D., and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, IETF
14. Network Working Group, November 1998, http://www.ietf.org/rfc/rfc2409.txt Frankel, Sheila, An Introduction to IPSec, NIST Information Technology Bulletin, March 2001, http://www.itl.nist.gov/lab/bulletns/bltnmar01.htm
15. Hernandez, Rich, Internet Protocol Security Revealed, Dell Power Solutions Magazine, Issue 2, 2000, http://www.dell.com/us/en/esg/topics/power_ps2q00-ipsec.ht