Secure Remote Desktop Access Using SSHX and ECC on a PKI

Dr.T.Pandikumar[1], Abedella Mussema[2]

[1]*Ph.D. Department of Computer & IT, College of Engineering, Defence University, Ethiopia*
[2]*M.Tech. Department of Computer & IT, College of Engineering, Defence University, Ethiopia*

Abstract - *Remote Desktop Connection is a technology that allows you to sit at a computer and connect to a remote computer in a different location. There has been a greater implementation of remote access technologies in recent years. Many organizations are adapting remote technologies such as Virtual Network Computing (VNC) and Remote Desktop (RDP) applications as customer support application. They use these applications to remotely configure computers and solve computer and network issues of the client on spot. Therefore, the system administrator or the desktop technician does not have to sit on the client computer physically to solve a computer issue.*

Keywords: *SSH, PKI, Elliptic Curve Cryptography, ECDH, ECDSA, Ccertificate, Key, Authentication, RSA.*

1. INTRODUCTION

1.1 Remote Desktop Overview

With Remote Desktop, you can have access to a Windows session that is running on your computer when you are at another computer. This means, for example, that you can connect to your work computer from home and have access to all of your programs, files, and network resources as though you were sitting at your computer at work. You can leave programs running at work and when you get home, you can see your work desktop displayed on your home computer, with the same programs running.

With Fast User Switching, you can easily switch from one user to another on the same computer. For example, suppose you are working at home and have logged on to the computer at your office to update an expense report. While you are working, a family member needs to use your home computer to check for an important e-mail message. You can disconnect Remote Desktop, allow the other user to log on and check e-mail, and then reconnect to the computer at your office, where you will see the expense report exactly as you left it. Fast User Switching works on standalone computers and computers that are members of workgroups.

Always the first time authentication is vulnerable to the Man-in-the-Middle attack. Using a public key certificate as a host key will eliminate the above vulnerability. And it requires a PKI, Public Key Infrastructure to support the certificate approach. PKI may potentially impact the performance of the security protocol. And PKI path validation techniques (certificate revocation status checking) need more storage capacity, more communication cost and more processing time. This seems to have a problem to scale with large communicating nodes. the Elliptic Curve Cryptosystem (ECC) offers the highest strength per bit of any known public-key cryptosystem today.

1.2 Motivation

Today many organizations are adapting remote technologies such as Virtual Network Computing (VNC) and Remote Desktop (RDP) applications as customer support application. They use these applications to remotely configure computers and solve computer and network issues of the client on spot.

SSH2 uses a public key cryptosystems to derive symmetric keys and then use faster symmetric-key algorithms to ensure confidentiality, integrity and source authentication of bulk-data. It employs RSA and DSA public-key cryptosystems for that. Because of plain public-key usage for server authentication the protocol is vulnerable to the Man-in-the-Middle Attack. So certificate authentication is needed to avoid that problem.

The Processing time significantly increases as the larger key sizes are used. Table 1 [18] shows this expected key-size growth for various symmetric and public-key Cryptosystems.

Table 1: Computationally equivalent key sizes

Symmetric	ECC	RSA/DH/DSA
80	163	1024
128	283	3072
192	409	7680
256	571	15360

As shown in Table 1, the Elliptic Curve Cryptosystem (ECC) offers the highest strength per bit of any known public-key cryptosystem today. ECC not only uses smaller keys for equivalent strength compared to traditional public-key cryptosystems like RSA, the key size disparity grows as security needs increase. This makes it especially attractive for constrained wireless devices because smaller keys result in power, bandwidth and computational savings. Considering the low processing power of most internet hosts like handheld devices, it may be reasonable to restrict the key sizes. Therefore, evaluating the performance of the Protocol for different key exchange cipher suites and proposing a better solution is Valuable. This thesis tries to integrate ECC in to the Secure Shell Protocol and evaluate the performance impact. In addition thesis also tries to integrate certificate authentication and analyze the different PKI revocation mechanisms and use a scalable revocation scheme.

2. BACKGROUND OF REMOTEDESKTOP ACCESS AND SSH2

Remote Desktop Connection is a technology that allows you to sit at a computer (sometimes called the client computer) and connect to a remote computer (sometimes called the host computer) in a different location. For example, you can connect to your work computer from your home computer and have access to all of your programs, files, and network resources as though you were in front of your computer at work. You can leave programs running at work and then, when you get home, you can see your work computer's desktop displayed on your home computer, with the same programs running. Remote Desktop sessions operate over an encrypted channel, preventing anyone from viewing your session by listening on the network. However, there is vulnerability in the method used to encrypt sessions in earlier versions of RDP. This vulnerability can allow unauthorized access to your session using a man-in-the-middle attack. Remote Desktop can be secured using SSL/TLS in Windows Vista, Windows 7, and Windows Server 2003/2008.While Remote Desktop is more secure than remote administration tools such as VNC(virtual network computing) that do not encrypt the entire session, any time Administrator access to a system is granted remotely there are risks. There has been a greater implementation of remote access technologies in recent years. Many organizations are adapting remote technologies such as Virtual Network Computing (VNC) and remote desktop (RDP) applications as customer support application. They use these applications to remotely configure computers and solve computer and network issues of the client on spot. Therefore, the system administrator or the desktop technician does not have to sit on the client computer physically to solve a computer issue. This is because illegal activities can be performed over the connection.

SSH is a protocol for secure remote login and other secure network services over an insecure network [1]. SSH is intended to run over a reliable transport protocol, such as TCP. There are two versions of SSH, imaginatively called SSH1 and SSH2. Use of SSH1 is deprecated because of some security problems. SSH2 has been separated into modules and consists of three protocols working together:

- SSH Transport Layer Protocol (SSH-TRANS)
- SSH Authentication Protocol (SSH-AUTH)
- SSH Connection Protocol (SSH-CONN)

SSH is designed to be modular and extensible. All of the core protocols define abstract services they provide and requirements they must meet, but allow multiple mechanisms for doing so, as well as a way of easily adding new mechanisms. All the critical parameters of an SSH connections are negotiable, including the methods and algorithms used in:

- Session key exchange
- Server authentication
- Data privacy and integrity
- User authentication
- Data compression

SSH-TRANS is the fundamental building block, providing the initial connection, record protocol, server authentication, and basic encryption and integrity services. After establishing an SSH-TRANS connection, the client has a single, secure, full-duplex byte stream to an authenticated peer.

Next, the client can use SSH-AUTH over the SSH-TRANS connection to authenticate itself to the server. SSH-AUTH defines a framework within which multiple authentication mechanisms may be used, fixing such things as the format and order of authentication requests, conditions for success or failure, and how a client learns the available methods.

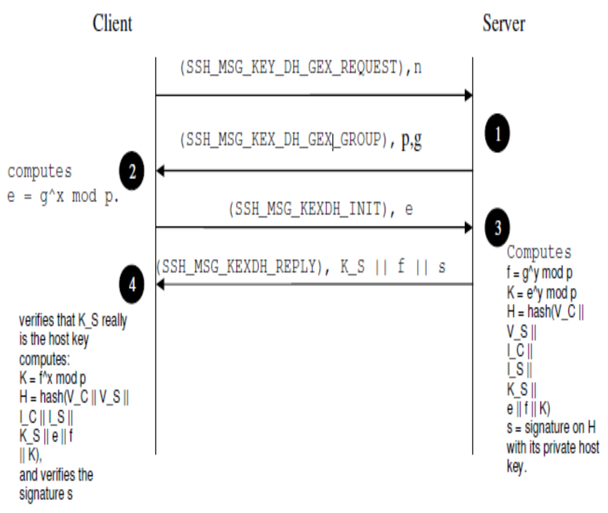
Client
Server
(SSH_MSG_KEY_DH_GEX_REQUEST),n
1
(SSH_MSG_KEX_DH_GEX_GROUP), p,g
computes
e = g^x mod p.
2
(SSH_MSG_KEXDH_INIT), e
3
Computes
f = g^y mod p
K = e^y mod p
H = hash(V_C || V_S || I_C || I_S || K_S || e || f || K)
s = signature on H with its private host key.
(SSH_MSG_KEXDH_REPLY), K_S || f || s
4
verifies that K_S really is the host key
computes:
K = f^x mod p
H = hash(V_C || V_S || I_C || I_S || K_S || e || f || K),
and verifies the signature s

Figure 1: Transport layer key exchange

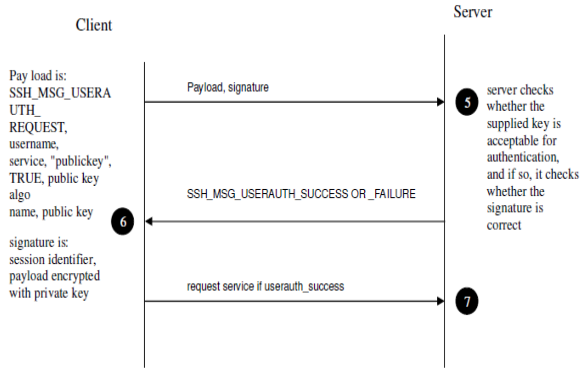
Client
Server
Pay load is: SSH_MSG_USERAUTH_REQUEST, username, service, "publickey", TRUE, public key algo name, public key
signature is: session identifier, payload encrypted with private key
Payload, signature
5
server checks whether the supplied key is acceptable for authentication, and if so, it checks whether the signature is correct
SSH_MSG_USERAUTH_SUCCESS OR _FAILURE
6
request service if userauth_success
7

Figure 2: User Authentication Layer (using public key)

2.1 Trust

The server host key is used during key exchange to verify that the client is really talking to the correct server. For this to be possible, the client must have a priori knowledge of the server's public host key. Two different trust models can be used:
The protocol provides the option that the server name - host key association is not checked when connecting to the host for the first time. This allows communication without prior communication of host keys or certification. The connection still provides protection against passive listening; however, it becomes vulnerable to active man-in-the middle attacks.

2.2 Public key cryptography in SSH

The public-key cryptographic operations performed by a client and server in different Modes of the public key algorithm are summarized as follows.

Table 2 : P**ublic key cryptographic operations**

	DSA	RSA
Client	DSAverify+DHoperation	RSAverify+DHoperation
Server	DSAsign+DHoperation	RSAsign+DHoperation

3. ECC AND PKI BASED KEY EXCHANGE

3.1 Possible solutions to avoid Man-in-the-Middle Attack

3.1.1 Key distribution center (KDC)

One way to check the binding of the server's public key to identity is to use a trusted node known as a Key Distribution Center [28]. The KDC knows keys for all the nodes. If a new node is installed in the network, only that new node and the KDC need to be configured with a key for that node. The KDC database essentially consists of a list of principals and their keys. For user principals, the key is derived from a password. Principals may also correspond to software services, such as an SSH server, etc.; their keys are randomly generated and stored in protected files where the services can access them.

When principal A wants to communicate with another—say, B—principal A first tells the KDC that it wants to talk to B. Principal A needs to do two things: prove its identity to B, and establish a shared secret with B for secure communication, called a *session key*. The KDC provides these things in a message called a *ticket*, which it sends back to A. The ticket is sealed with A's secret key, known only to the KDC and A—hence A trusts that it is genuine, and it is protected from network snooping. Unsealing the ticket, A finds the needed session key—and yet another ticket! This one, however, is sealed with B's secret key (known only to the KDC and B). A can't read this at all, but that doesn't matter; all A needs to do is send this ticket as-is to B. When B unseals its ticket, it finds A's name and another copy of the session key. Just as before, since B's ticket is sealed with B's key, B trusts that the ticket is genuine. The meaning of each ticket is that the KDC has shared the session key with A and B. The two principals then execute a protocol which proves to each that the other does in fact hold that key—at which point, mutual authentication is accomplished. Further, the session key can be used for subsequent security functions, such as encrypting a conversation between them.

3.1.2 Distributed Key Management

Distributed key management, solves the key management problem with introducers. Introducers are other users of the system who sign their friends' public keys.

3.2 Proposed method of authentication and key agreement Proposed solution

In the previous sections we have stated different alternatives but none of them can be employed because of their own limitations. Hence this thesis proposes to use a Public Key Infrastructure to make the SSH authentication strong.

3.2.1 PKI

Public key cryptography supports security mechanisms such as confidentiality, integrity, authentication, and non-repudiation. However, to successfully implement these security mechanisms, you must carefully plan an infrastructure to manage them. A public key infrastructure (PKI) is a foundation on which other applications, system, and network security components are built. A PKI is an essential component of an overall security strategy that must work in concert with other security mechanisms, business practices, and risk management efforts. PKI supports the various business and technical needs by allowing for the identification of entities. The distribution and management of public keys and associated certificates normally occur through the use of Certification Authorities (CAs), Registration Authorities (RAs), and directory services, which can be used to establish a hierarchy or chain of trust. CA, RA, and directory services allow for the implementation of digital certificates that can be used to identify different entities. One of the primary principles of a PKI is the establishment of a trust hierarchy. In Internet-based communications like e-commerce, formal trust mechanisms must exist to provide risk management controls. The concept of trust, relative to a PKI, can be explained by the role of the CA. The implementation of a PKI using a CA provides this trust.

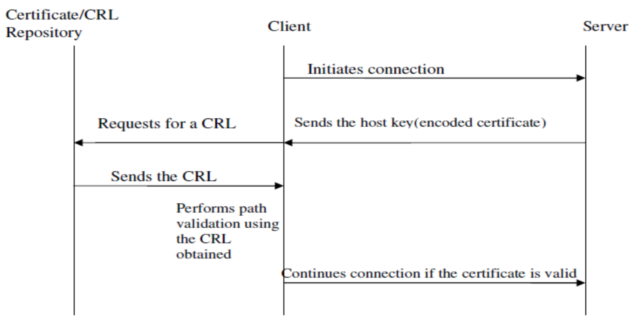
Certificate/CRL Repository
Client
Server
Initiates connection
Requests for a CRL
Sends the host key(encoded certificate)
Sends the CRL
Performs path validation using the CRL obtained
Continues connection if the certificate is valid

Figure 3: Authentication using public-key certificates

3.2.1.2 Public key cryptographic operations

The public-key cryptographic operations performed by a client and server in different modes of the public key algorithm including the online OCSP responder public key operation are summarized as follows.

Table 3: Public-key operations

	OCSP	CRL
Client	Server_Certverify+OCSP resp.verify + DSAverify/RSAverify +DHoperation	Server_Certverify+CRLverify + DSA/RSAverify+DHoperation
Server	2(DSAsign/RSAsign) +DHoperation	RSA/DSAsign+DHoperation
OCSP responder	DSAverify/RSAverify +OCSPresp.sign	

SSH allows both client- and server-side authentication. However, due to the difficulty of managing user certificates across multiple client devices, the former is rarely used. User authentication, in such cases, happens at the application layer, *e.g.* through passwords sent over an SSH protected channel. Authentication of the SSH client is not discussed further in this paper.

3.2.2 Proposed key-exchange scheme

3.2.2.1 ECC Basics

At the foundation of every public key cryptosystem is a hard mathematical problem that is computationally infeasible to solve. For instance, RSA and Diffie-Hellman rely on the hardness of integer factorization and the discrete logarithm problem respectively. Unlike these cryptosystems which operate over integer fields, the Elliptic Curve Cryptosystems (ECC) operates over points on an elliptic curve. The fundamental mathematical operation in RSA and Diffie-Hellman is modular integer exponentiation. However, the core of elliptic curve arithmetic is an operation called *scalar point multiplication*, which computes $Q = kP$ (a point P multiplied k times resulting in another point Q on the curve). Scalar multiplication is performed through a combination of point-additions (which add two distinct points together) and point doublings (which add two copies of a point together). For example, $11P$ can be expressed as $11P = (2 * ((2* (2* P)) + P)) + P$. The security of ECC relies on the hardness of solving the Elliptic Curve Discrete Logarithm Problem (ECDLP), which states that given P and $Q = kP$, it is hard to find k. While a brute-force approach is to compute all multiples of P until Q is found, k would be so large in a real crypto-graphic application that it would be infeasible to determine k in this way. Besides the curve equation, an important elliptic curve parameter is the *base point*, G, which is fixed for each curve. In the Elliptic Curve Cryptosystem, the large random integer k is kept private and forms the secret key, while the result Q of multiplying the private key k with the curve's base point G serves as the corresponding public key. Not every elliptic curve offers strong security properties and for some curves the ECDLP may be solved efficiently. Since a poor choice of the curve can compromise security, standards organizations like NIST and SECG have published a set of recommended curves with well understood security properties.

3.2.2.2 The modified key-exchange method

In this study, our goal is to maintain strong scalable security (protocol) scheme that is applicable for all types of internet hosts or scheme that ease practical deployment on the Internet. SSH uses public key to identify an entity by using digital signatures and deffie-hellman key exchange algorithm to establish a shared secret key. In here, we can see that two public-key cryptographic operations will be performed for every key exchange.

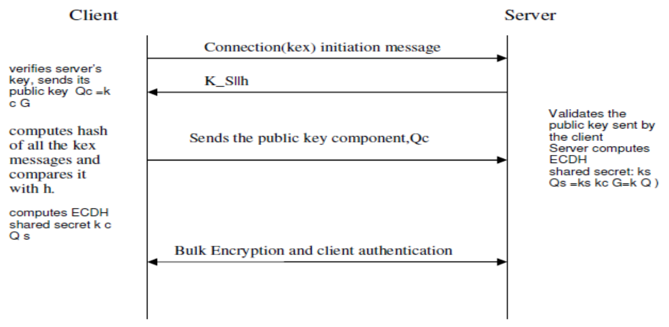
Client
Server
Connection(kex) initiation message
verifies server's key, sends its public key Qc =k c G
K_SIlh
computes hash of all the kex messages and compares it with h.
Sends the public key component,Qc
Validates the public key sent by the client Server computes ECDH shared secret: ks Qs =ks kc G=k Q)
computes ECDH shared secret k c Q s
Bulk Encryption and client authentication

Figure 5: The proposed key exchange handshake

The Server Certificate contains the server's ECDH public key signed by a certificate authority using ECDSA. After validating the ECDSA signature, the client conveys its ECDH public key to the server in the Client Key Exchange message. Next, each entity uses its own ECDH private key and the other's public key to perform an ECDH operation and arrive at a shared secret key. The derivation of the session symmetric keys is unchanged.

3.2.2.3 PKI proposed revocation method

Certificates are valid until they expire, unless they are revoked beforehand. Many things can happen that require the revocation of a certificate. For example, the secret key may be lost or irreversibly destroyed, the certificate holder may cease operation, the certificate holder's identifier may need to be updated due to a name change, one of the (other) Client Server attributes in the certificate may have become invalid, or the secret key may have been compromised. While revocation is an exceptional circumstance, the task of verifiers to check the revocation status of unexpired certificates unfortunately is not. They must either have the certificate status validated online (at the time of the communication or transaction) or regularly download a digitally signed update of a blacklist called the Certificate Revocation List (CRL). In both cases the status of certificates must be maintained by the CA (or by a special Revocation Authority).

3.2.2.4 A PKI authentication mechanism with Authenticated Dictionaries

By using Authenticated Dictionaries as revocation status responder we can eliminate the response delays caused by OCSP responders and long sized CRLs given by CRL repositories.

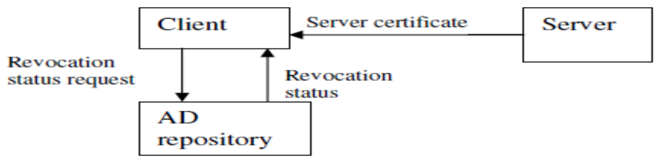

Figure 6: Overview of authentication using AD

3.2.2.3.2 Public-key cryptography in the modified scheme

Table 3: Proposed KEX with PKI

	OCSP	CRL
Client	ECDSAverify+OCSPverify +ECDHoperation	2*ECDSAverify +ECDHoperation
Server	OCSPreq.sign+ECDHoperation	ECDHoperation
OCSP server	OCSPreq.verify+OCSPresp.sign	

The client performs an ECDSA verification to verify the server's certificate and RSA verification to verify the root hash value of the revocation response and then an ECDH operation using its private ECDH key and the server's public ECDH key to compute the shared secret key. All the server needs to do is perform an ECDH operation to arrive at the same secret.

Table 4: Proposed KEX with proposed PKI

	AD
Client	2*ECDSAverify |+ECDHoperation
Server	ECDHoperation

4. RESULT AND PERFORMANCE ANALYSIS

4.1 Performance Metrics

For the performance analysis in a remote login considering End-to-end delay, Traffic sent, Traffic received (packets/second), load and Packet Delay Variation were taken as our means for evaluation.

Running simulation for a number of times with different seed values also helps to see the effects of random values of the simulation output. For each of the scenarios the simulation was run for an hour to get its steady state.

4.2 Experiments performed and Results

We start building the networking model by creating a project with Model Family LAN included, and work on the model at the network layer. A subnet is created to represent the office network.

Also need to define applications and profiles by adding a node for each, and we can associate the workstation with the profiles in order to use the applications.

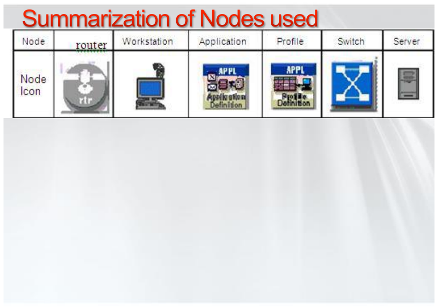

Figure 7 : Summarization of nodes used

Configurations are as following:

1. Applications: Remote login, Telnet, Ssh, ssh RSA, ssh ECC (all with high load).
2. Applications applied gradually with 30mins interval.
3. Each application runs for two hours of duration.
4. Simulation Duration: 1.0 hours
5. Simulation Running Time: ~1min
    a. From the following simulation result graph, we can see all the throughput starts at initial, remote login ends after 10m, Ssh ends at time 30m, and finally Telnet ends at time one hour.

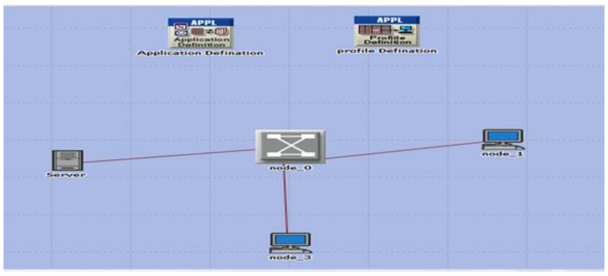

Figure 8: Structure of 1 server and 2 Workstation Networking

From the following simulation result graph, we can see all the throughput starts at initial, remote login ends after 10m, Ssh ends at time 30m, and finally Telnet ends at time one hour.

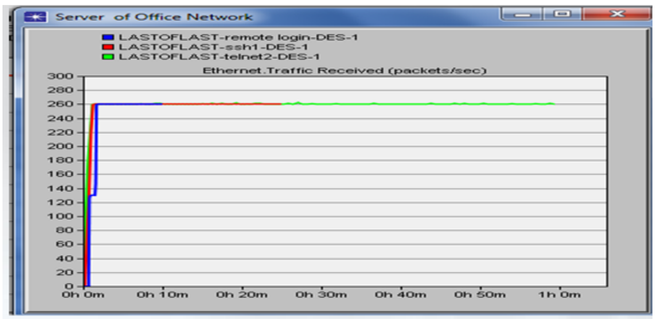

Figure 9: Throughput for Each Application

From the following simulation result graph, we can see the throughput and delay starts after half an hour at initial, sshecc ends after 1hour, Ssh RSA ends at time 3 hour.

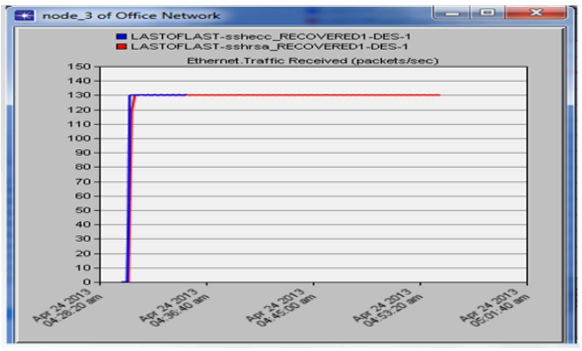

Figure 10 : Throughput for Each Application

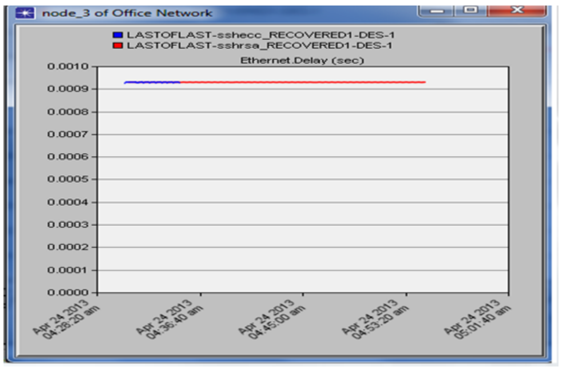

Figure 11: Delay for Each Application

Table 5: Cryptographic strength of RSA and ECC

Strength Level	ECC	RSA
1	160P,163K,163R	1024
2	224P,233K,233R	2048
3	256P,283K,283R	3072

5. ANALYSIS OF RESULTS

For the performance analysis of the network considering remote login, End-to-end delay, Traffic sent, Traffic received and Packet Delay Variation was taken as our means for evaluation.

The results shown above show the performance advantage of ECC over RSA for different security levels. Note that the performance advantage of ECC gets even better than its key-size advantage as security needs increase. Elliptic curve systems are increasingly seen as an alternative to RSA, rather than a replacement.

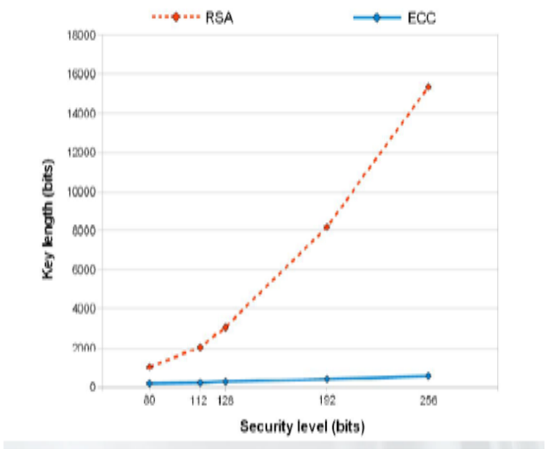

Figure 12: Key length comparisons RSA and ECC

At the 163-bit ECC/1024-bit RSA security level, an elliptic curve exponentiation for general curves over arbitrary prime fields is roughly 5 to 15 times as fast as an RSA private key operation, depending on the platform and optimizations.

9. REFERENCES

[1] Daniel J. Barrett & Richard E. Silverman," SSH The Secure Shell, TheDefinitive Guide", O'REILLY & Associates, 2005

[2] Carlisle Adams, Steve Lloyd,"Understanding PKI: Concepts, Standards, and Deployment Considerations", November 2002.

[3] Bruce Schneier, "Applied Cryptography: Protocols, Algorithms, and Source Code in C", John Wiley & Sons, 1996.

[4] A. Menezes, P. Van Oorschot, S. Vanstone," Handbook of Applie Cryptography", CRC Press, 1997.

[5] William Stallings, "Cryptography and Network Security: Principles and Practice", Prentice Hall, 2003.

[6] D. Hankerson, A. Menezes, S. Vanstone, "Guide to Elliptic Curve Cryptography", Springer, 2004.

[7] Markus Friedl, et al., "Diffie-Hellman Group Exchange for the SSH Transport Layer Protocol", Internet Draft, January 2002.

[8] Saarenmaa & Galbraith "X.509 authentication in SSH ", Internet Draft, February 28, 2006

[9] T. Ylonen, T. Kivinen, M. Saarinen, T. Rinne, and S. Lehtinen, "SSH Protocol Architecture", Network Working group, Internet Draft, January 31, 2002.

[10] T. Ylonen, T. Kivinen, M. Saarinen, T. Rinne, and S. Lehtinen, "SSH Transport Layer Protocol", Network Working group, Internet Draft, January 31, 2002.

[11] T. Ylonen, T. Kivinen, M. Saarinen, T. Rinne, and S. Lehtinen, "Authentication Protocol", Network Working group, Internet Draft, February 28, 2002.

[12] T. Ylonen, T. Kivinen, M. Saarinen, T. Rinne, and S. Lehtinen, "SSH Connection Protocol", Network Working group, Internet Draft, January 31, 2002.

[13] W. Ford ,R. Housley , W. Polk , D. Solo ,"Internet X.509 Public Key Infrastructure certificate and Certificate Revocation List (CRL) Profile",RFC 3280, April 2002 91

[14] Yasir Ali and Sean W. Smith, "Flexible and Scalable Public Key Security for SSH ", submitted for the Second Annual PKI Workshop, Feb 2003

[15] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. "X.509 Internet Public Key Infrastructure: Onlice Certificate Status Protocol – OCSP" Technical Report RFC2560, IETF, June 1999.

[16] M. Naor and K. Nissim. "Certificate Revocation and Certificate Update" IEEE Journal on Selected Areas in Communications, 18(4):561–570, 2000.

[17] Sean Mullan "JavaTM Certification Path API Programmer's Guide", February 2003

[18] S. Blake-Wilson and T. Dierks, "ECC Cipher Suites for TLS", Internet draft <draft- ietf-tls-ecc-01.txt>, work in progress, Mar. 2001.

[19] Vipul Gupta, Douglas Stebila, Stephen Fung, Sheueling Chang Shantz, Nils Gura, Hans Eberle," Speeding up Secure Web Transactions Using Elliptic Curve Cryptography"

[20] R. Rivest, A. Shamir, L. Adleman, "A Method for Obtaining Digital Signatures and Public-key Cryptosystems", http://theory.lcs.mit.edu/~rivest/rsapaper.pdf, 1977.

[21] OpenSSL Project, see http://www.openssl.org/.

[22] Robert Zuccherato, "Elliptic Curve Cryptography Support in Entrust", May, 2000.

[23] Certicom Research, "SEC 2: Recommended Elliptic Curve Domain Parameters", Standards for Efficient Cryptography, Version 1.0, Sep. 2000.

[24] NIST, "Special Publication 800-57: Recommendation for Key Management. Part 1: General Guideline", Draft Jan. 2003

[25] W. Diffie, M. Hellman, "New Directions in Cryptography", http:// crypto.csail.mit.edu/classes/6.857/papers/diffie-hellman.pdf, 1976.

[26] Man Young Rhee ,"Internet Security Cryptographic Principles, Algorithms and Protocols", 2003.

[27] Joel Weise, "Public Key Infrastructure Overview", August 2001.

[28] C.kaufman,R Perlman and M. Speciner, "Network Security-Private communication in a public world",2006.

[29] MeftunGoktepe, "WINDOWSXPOPERATING SYSTEM SECURITY ANALYSIS", September 2002.

[30] PareshKerai," Remote Access Forensics for VNC and RDP on Windows Platform", September 2010.