Public Key Infrastructure in iDRAC

A Dell Technical White Paper

Dell Enterprise Team

Jeethendra Telagu

March 2011

Contents

Overview of Public Key Infrastructure

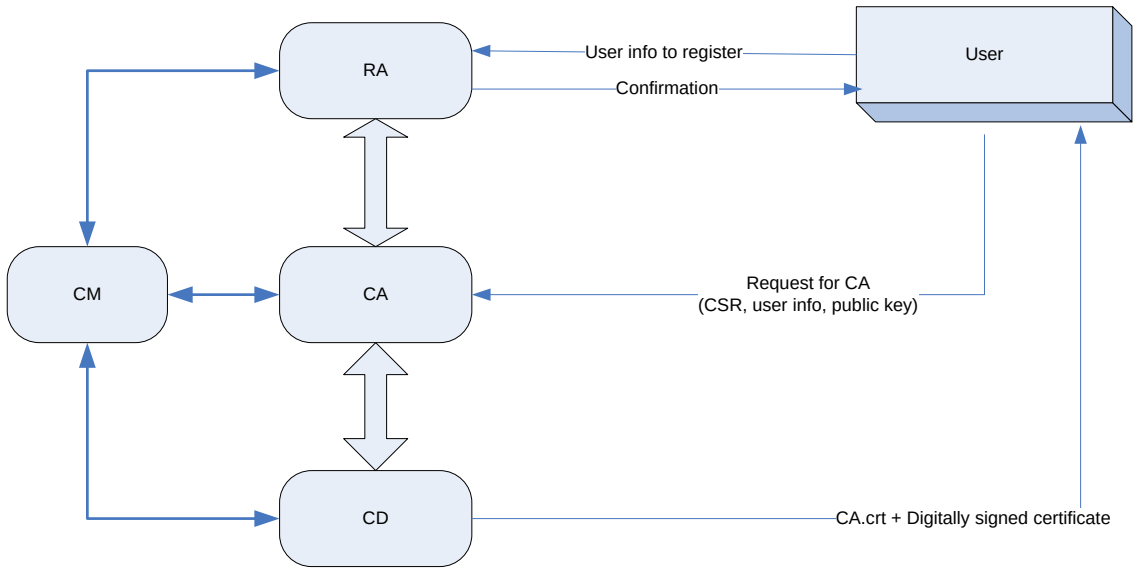
Public Key Infrastructure (PKI) is a set of policies or procedures defined by individuals or a standards body to create, manage and distribute digital certificates to support the secure exchange of data. iDRAC6 supports PKI over SSH. This security feature improves SSH scripting automation by removing the need to embed or prompt for a user ID or password.

The various components of PKI include:

- Registration Authority (RA)
- Certificate Authority (CA)
- Certificate Management (CM)
- Certificate Distributor (CD)

The central feature of the PKI framework is the Certificate Authority (CA), as shown in the following figure:

Figure 1. PKI Infrastructure



Registration Authority

Before a user can use the services of a CA, they must first enroll with a Registration Authority (RA). The Registration Authority registers the user after validating the user's identity. Thereafter, whenever the user requests a service from the CA, the Registration Authority will first check the user's credentials.

Certificate Authority

The Certificate Authority (CA) provides several key functions in the PKI framework.

First, the CA checks that that the user is who they claim to be by confirming their identity with the Registration Authority. The CA then generates a unique key pair (CA key) which corresponds to the registered user. The key pair consists of a *private key* and a *public key*. The CA public key will be generated in the form of a *public key digital certificate*. The CA distributes the CA public key certificate to the corresponding registered user.

Another main function of the CA is to certify the *user public key* (a public key generated by the user in their host machine) and generate a digital certificate. The CA binds the user's unique identity or the user information provided by the user with the public key by digital signature. The digital signature is an encryption of the user's unique identity, certificate information, and user's public key, using the CA private key which corresponds to that user. The signature then becomes the part of the public key digital certificate. The digital certificate is then distributed to the user. The x.509 v3 standard is the most commonly used digital certificate format.

Certificate Management

The Certificate Management manages the list of CA public key digital certificates corresponding to the registered users. Each digital certificate is generated with a validity period. The CM validates the certificate against its validity time and revokes any expired certificates. Note that any public key certificate signed by an expired or invalid root CA private key is also invalid.

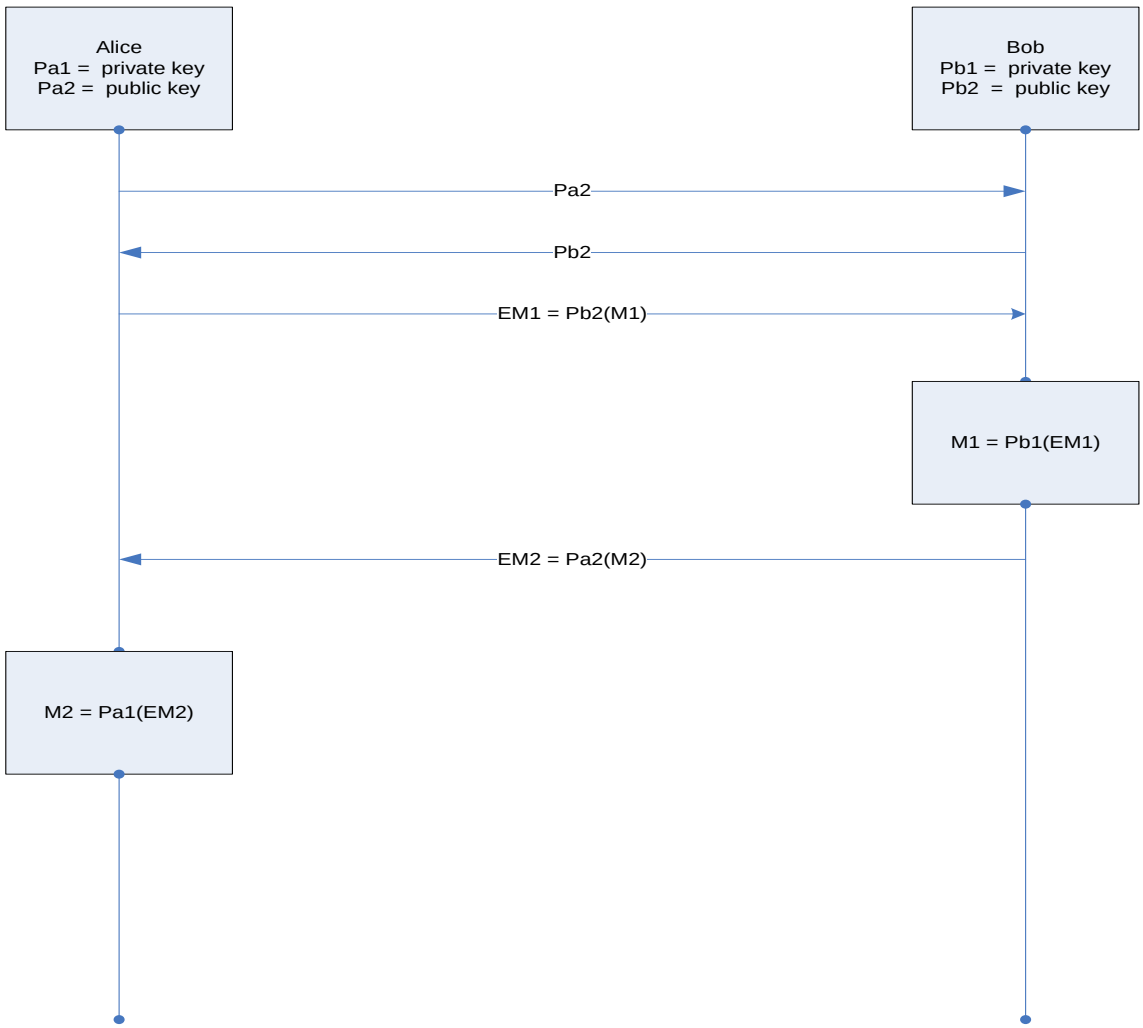
Certificate Distributor

The Certificate Distributor distributes the digital certificates to the corresponding users registered with the Registration Authority.

Asymmetric Keys

An asymmetric key cryptosystem uses a key pair consisting of a unique private key and a corresponding public key. As the name implies, the private key is held secret and the public key is made public. The asymmetric key cryptosystem uses the private key for encryption and the public key for decryption. The key pair is typically generated using the RSA algorithm.

Figure 2. Overview of an Asymmetric Key Exchange



Consider the example shown in Figure 2 to understand asymmetric key cryptology. Alice and Bob each have a unique key pair. The private key is held secret and the public key information is exchanged between the two users.

First, Alice uses Bob's public key Pb2 to encrypt message M1 before sending it to Bob.

Encrypted message EM1 = Pb2(M1)

Next, Bob extracts message M1 by decrypting EM1. Bob uses his private key Pb1 for decryption.

Decrypted message M1 = Pb1(EM1)

Bob then uses Alice's public key Pa2 to encrypt message M2 before sending it to Alice.

Encrypted message EM2 = Pa2(M2)

Finally, Alice uses her private key Pa1 to decrypt and extract the message M2.

In the case of a man in the middle attack, a hacker will have both the public key and the encrypted messages, but still cannot decrypt the encrypted message because of the lack of private keys. (Note that in asymmetric key cryptography the message encrypted by a private key can only be decrypted by the corresponding public key.)

Components of an Asymmetric Key

The typical RSA key consists of the following information:

- Exponent
- Modulus

Exponent

The exponent can have a maximum size of 65537, with a range of 3–65537. The greater the exponent, the more secure the key is. However data encrypted with a key which has a large exponent takes more time to decrypt. The private and public keys will have unique exponents.

Modulus

The size of the modulus defines the key size. The key size ranges from 786 bits to 4096 bytes. Both the private key and corresponding public key will have the same modulus.

Third Party PKI

Third Party PKI or Trusted Third Party PKI can be used for certificate authority services. The components of third party PKI are the same as PKI except the Certificate Authority function is handled and maintained by a trusted third party. Companies such as Verisign offer services for third party PKI.

Implementing PKI Using iDRAC

Obtaining an iDRAC Web Server Digital Certificate using Third Party PKI

The iDRAC has a web server that is configured to use the industry-standard SSL security protocol to transfer encrypted data over the network. SSL protocol is built on the asymmetric key cryptosystem technology previously discussed.

The SSL-enabled system can perform the following tasks:

- Authenticate itself to an SSL-enabled client
- Allow the client to authenticate itself to the server
- Allow both systems to establish an encrypted connection

The iDRAC web server has a Dell-signed SSL web certificate by default. However, to ensure high security, a SSL web server digital certificate signed by a third party PKI should be used. The process to obtain the certificate is as follows:

1. Register with the third party PKI
2. Generate a certificate signing request (CSR)
3. Submit a Request for Certificate to the Third Party PKI CA
4. Receive a digitally signed certificate from the CA
5. Upload the digital certificate to iDRAC

Registering With the Third Party PKI.

The user has to first register with the third party PKI by providing user information. Once the user's identity is confirmed, the third party PKI generates a CA key pair corresponding to that user. The CA private key is held secret by the third party PKI whereas the CA public key is distributed to the user as a CA digital certificate.

Generating a CSR

The user can then use the iDRAC GUI to generate a CSR or certificate signing request. The iDRAC internally generates a key pair and uses the user information as well as the public key from the CA to generate a CSR file. (The CSR file can also be encrypted with the user private key.)

The CSR consist of the user's information and the CA public key:

- Common name
- Organization name
- Organization unit
- Locality
- State
- CA public key

A sample CSR file is shown below:

-----BEGIN CERTIFICATE REQUEST-----

MIIBijCB9AIBADBLMQswCQYDVQQGEwJVUzELMAkGA1UECBMCVHgxCzAJBgNVBAcT

AlJSMQ0wCwYDVQQKEwREZWxsMRMwEQYDVQQDEwpqZWV0aGVuZHJhMIGfMA0GCSqG

SIb3DQEBAQUAA4GNADCBiQKBgQC++EOp37BPaAFWb3OiA85n3jU+U5fUsUxTRiqP

DbpwfiwBDlyImnrdKVbqOdeSFAZSDEf/Crdcza083qWjYoyw8N0sdeg/EB3irMzn

yv2FNG6YMO8e+bL5E/s3Dk2QYutFz1+Et5yo5NLibbzRqhlkTFFgJ1k6qQi3vX6s

QlNGVQIDAQABoAAwDQYJKoZIhvcNAQEFBQADgYEAIgu0TLALvJRwt6Ku36CiFmSa

IMpHMxCTlAh38PulG1QlAUtgru4lNuGAiQhHxFuH5fnKFazkda/7JVzBXWFK5rLF

UGVGikLNHvkCA+Td/mz6LQzHWKtBk+4pNHSoXvPudQ+GR3AwsV8/zAEibjVFNYCX

MCHSKklJi8n8gQG07Cw=

-----END CERTIFICATE REQUEST-----

Receiving a Digitally Signed Certificate from the Trusted Third Party PKI

The third party PKI CA verifies that the user is the registered user, and then creates a digitally signed certificate and issues it to the user:

1. The CA decrypts the CSR file using the user public key and extracts the certificate information.

2. The CA generates a public key certificate using the certificate information from the CSR file and the user public key. The certificate is generated in a Web certificate format such as X.509 v3.

3. The CA signs the certificate using the user CA private key. The signature is appended to the certificate.

4. The digitally signed certificate is issued to the user along with the CA public key in the form of a CA public key digital certificate.

A sample digitally signed certificate is shown below

Certificate:
Data:
Version: 1 (0x0)
Serial Number: 7829 (0x1e95)
Signature Algorithm: md5WithRSAEncryption
Issuer: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
OU=Certification Services Division,
CN=Thawte Server CA/Email=server-certs@thawte.com
Validity

Not Before: Jul 9 16:04:02 1998 GMT
Not After : Jul 9 16:04:02 1999 GMT
Subject: C=US, ST=Maryland, L=Pasadena, O=Brent Baccala,
OU=FreeSoft, CN=www.freesoft.org/Email=baccala@freesoft.org
Subject Public Key Info:
Public Key Algorithm: rsaEncryption
RSA Public Key: (1024 bit)
Modulus (1024 bit):
00:b4:31:98:0a:c4:bc:62:c1:88:aa:dc:b0:c8:bb:
33:35:19:d5:0c:64:b9:3d:41:b2:96:fc:f3:31:e1:
66:36:d0:8e:56:12:44:ba:75:eb:e8:1c:9c:5b:66:
70:33:52:14:c9:ec:4f:91:51:70:39:de:53:85:17:
16:94:6e:ee:f4:d5:6f:d5:ca:b3:47:5e:1b:0c:7b:
c5:cc:2b:6b:c1:90:c3:16:31:0d:bf:7a:c7:47:77:
8f:a0:21:c7:4c:d0:16:65:00:c1:0f:d7:b8:80:e3:
d2:75:6b:c1:ea:9e:5c:5c:ea:7d:c1:a1:10:bc:b8:
e8:35:1c:9e:27:52:7e:41:8f
Exponent: 65537 (0x10001)
Signature Algorithm: md5WithRSAEncryption
93:5f:8f:5f:c5:af:bf:0a:ab:a5:6d:fb:24:5f:b6:59:5d:9d:
92:2e:4a:1b:8b:ac:7d:99:17:5d:cd:19:f6:ad:ef:63:2f:92:
ab:2f:4b:cf:0a:13:90:ee:2c:0e:43:03:be:f6:ea:8e:9c:67:
d0:a2:40:03:f7:ef:6a:15:09:79:a9:46:ed:b7:16:1b:41:72:
0d:19:aa:ad:dd:9a:df:ab:97:50:65:f5:5e:85:a6:ef:19:d1:
5a:de:9d:ea:63:cd:cb:cc:6d:5d:01:85:b5:6d:c8:f3:d9:f7:
8f:0e:fc:ba:1f:34:e9:96:6e:6c:cf:f2:ef:9b:bf:de:b5:22:
68:9f

Uploading the Digital Certificate to iDRAC

The user uploads the digitally signed certificate from the third party PKI to the iDRAC web server.

At this point, the User host machine with SSL client should have the following information

- Copy of digitally signed certificate.
- User private key
- CA public key digital certificate

Authentication Using a Web Server Certificate

Figure 3. Authentication Using a Web Server Certificate

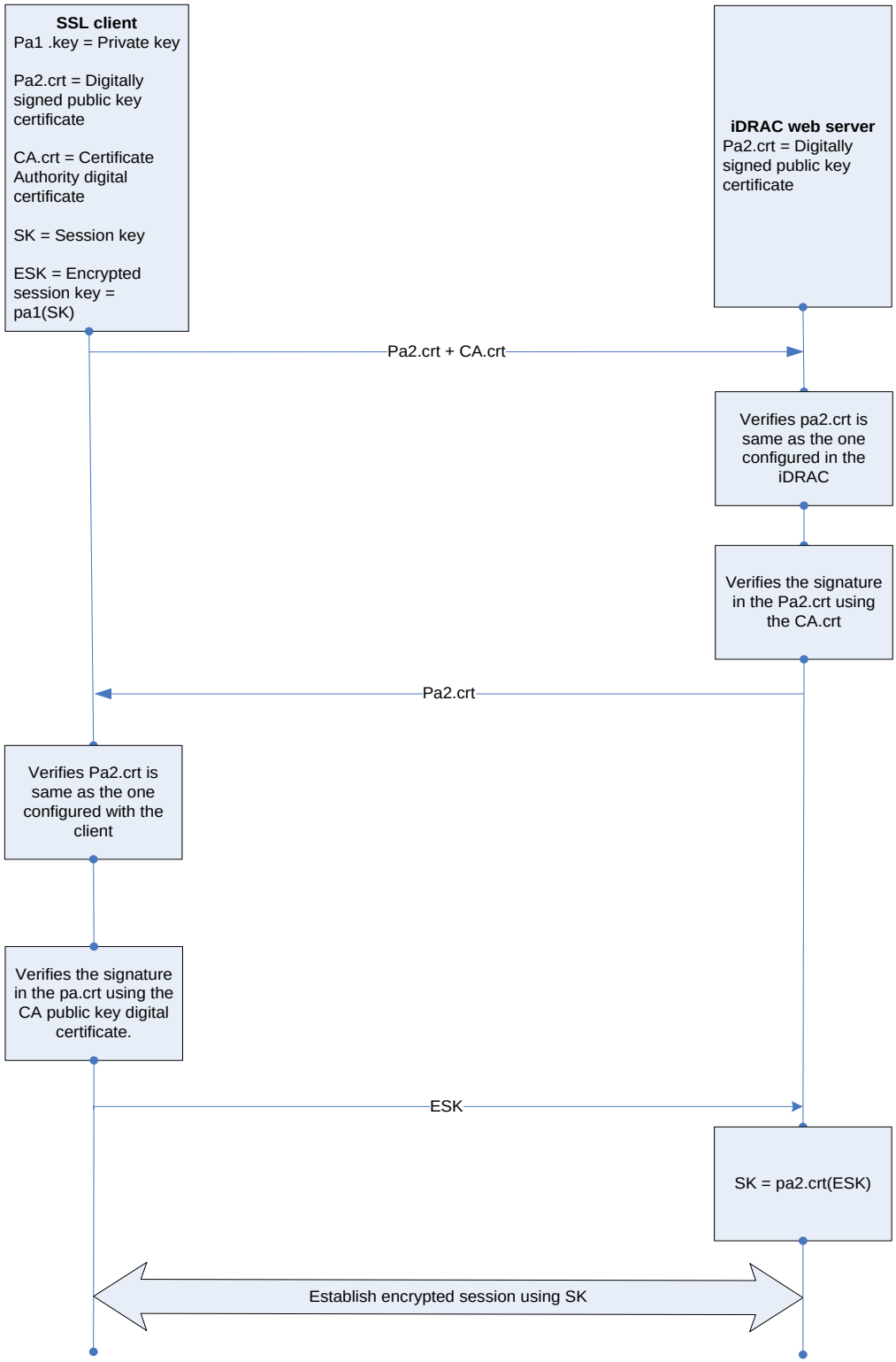


Table 1. Legend – Authentication Using a Web Server Certificate

Authentication components present with SSL client or user host machine	
Pa1.key	Private key
Pa2.crt	Public key certificate digitally signed by the trusted third party PKI
Ca.crt	Certificate Authority digital certificate
SK	Session key
ESK	Encrypted session key
Authentication components configured in iDRAC	
Pa2.crt	Public key certificate digitally signed by the trusted third party PKI

Figure 3 illustrates the process of authentication using a Web Server Certificate:

1. The Host machine sends the public key certificate (Pa2.crt) and CA digital certificate (CA.crt) to the iDRAC.
2. The iDRAC web server verifies the authenticity of the user by comparing the public key certificate with the one configured in the iDRAC. The iDRAC also confirms the user's identity by verifying the signature in the Pa2.crt using the CA.crt.

   If the user verification is successful, the iDRAC web server sends the copy of the public key certificate Pa2.crt to the client.

   If the user verification fails, the iDRAC web server terminates the SSL session.
3. The SSL client verifies the public key certificate to confirm that it is talking to the intended server.
4. The SSL client generates a session key SK, encrypts the session key using the private key Pa1.key, and sends the encrypted key to the server:

   Encrypted session key = ESK = Pa1(SK)
5. The iDRAC Web server extracts the session key SK using the public key web certificate Pa2.crt.
6. Henceforth all communication between client and server and vice versa will be encrypted by the session key SK.

Logging Into iDRAC Using PKI

A user can login to iDRAC using the following two options

- Authentication using a user ID and password
- Authentication using Public Key Infrastructure

Conventional User ID and Password Method

The user can SSH to iDRAC using the user id and password. The iDRAC identifies the user by comparing the user ID and password against the list of user IDs and corresponding passwords configured in the iDRAC and allows the user to login if they match. However, in this case the user password is exposed to hackers.

Public Key Infrastructure Method

In this method the user digital signature is used for authentication. The user can generate an RSA asymmetric key pair in the Host machine. The user can then encrypt the private key which can be held secret in the Host machine, and configure iDRAC with the user id and the public key.

The following steps outline this process. For details, see the iDRAC *User's Guide*.

1. Generate an RSA key pair in the Host machine. The key pair can be generated using the OpenSSH tool in a Linux machine, or the PuTTy KeyGen utility on a Windows-based machine.
2. The private key should be held secret in the Host machine.
3. Configure iDRAC with a user id and upload the public key. Each user can be configured with up to four public keys.
4. Open a SSH client in the Host machine and login to the iDRAC using the user id and the private key.

   If you have PuTTy, gen an SSH agent. The Putty gen can be configured for PKI authentication as follows:

   a. Open PuTTy.

   b. Select **session** to enter a Hostname or IP address.

   c. Select **SSH->Auth** to browse to and upload the private key.

   d. Enter the User ID in the SSH terminal to log into iDRAC.
5. The SSH client encrypts a message using the private key and sends the user id and the message to the iDRAC server.
6. The iDRAC server verifies the user's authenticity by decrypting the message using the public key associated with the user.

   If the user authenticity is verified successfully the iDRAC encrypts the session key with the public key and sends it back to the SSH client.

7. The SSH client extracts the session key by decrypting the encrypted key using the private key.

8. Henceforth all communication between the SSH client and server will be secured by encryption, using the session key.

(In case the user authentication fails in step 6, the SSH client will prompt for a user password.)

Advantages of Using Public Key Infrastructure

- There should be no security threat as long as the private key is held secret. The security of the Asymmetric keys depends on the key size. The greater the key size the more secure the PKI implementation.

- When the PKI over SSH is set up and used correctly, the user does not have to enter the username or password when logging into the iDRAC6. This can be very useful for setting up automated scripts to perform various functions. This feature can be managed with RACADM and also from the GUI. See the iDRAC *Users Guide* for more information.

Disadvantages of Using Public Key Infrastructure

- As the key size increases, the time taken for encryption and decryption is almost proportionally increased.

- While generating the key pair care should be taken to choose a high exponent value. The greater the exponent size the more secure the key is. If the exponent size is not specified during key generation most of the tools default to 3. The message encrypted with a key of exponent size 3 can be easily decrypted as below

    - Message = (Encrypted message)^1/3.