KMS User Guide

On-Ramp Wireless Incorporated
10920 Via Frontera, Suite 200
San Diego, CA 92127
U.S.A.

KMS User Guide

010-0062-00 Rev. C

October 5, 2012

Contents

Figures

Tables

Revision History

Revision	Release Date	Change Description
A	May 17, 2011	Initial release.
B	July 26, 2012	Updated the following: n CentOS/RHEL from version 5 to version 6 n Illustrations n Software installation instructions n Software configuration instructions Added the following: n Software compatibility matrix n Appendix for creating RSA keys
C	October 5, 2012	Corrected path and location for the Master Key File (keyring.csv) in Appendix B.

1 Introduction

This document describes the setup, configuration, and use of the Key Management Server (KMS) which is the main key server for the ULP network. The KMS maintains the Ultra-Link Processing™ (ULP) network's gateway key and code download key, as well as the root keys for all nodes provisioned and authorized to access the ULP network. This document also provides instructions on generating and exporting the gateway and code download (CDLD) keys to an encrypted file which are used to set up one or more Local Key Servers (LKS), as well as instructions on importing node root keys delivered from one or more LKS servers.

NOTE: This guide is intended for system administrator level users with root user privileges. General familiarity with security concepts is required.

The following table indicates software compatibility between software applications.

Table 1. Software Compatibility Matrix

	CommSys 1.2	CommSys 1.4
Provisioning 1.5 (includes LKS, NPT, and KMS utilities)	ü	ü
CIMA 1.2	ü	ü
CIMA 2.0	**No**	ü

2 ULP Key Management Overview

This chapter provides a brief overview of how ULP network keys are generated and managed. This involves the following basic steps:

1. The Key Management Server (KMS) generates the Gateway keys and creates the Master Key File 'keyring.csv' using the Generate Gateway Keys Utility (generate_gw_keys.py). This utility also creates an encrypted and signed output file that contains the Gateway keys for delivery to one or more Local Key Server (LKS) sites.
2. The LKS decrypts and imports the Gateway keys using the Import Gateway Keys Utility (import_gw_keys.py), which also creates the node key database.
3. After the LKS has provisioned one or more nodes, the node keys are exported to an encrypted and signed batch key file using the Export Keys Utility (export_keys.py) for delivery to the KMS.
4. The KMS imports the LKS batch key files merging the node keys into the single Master Key File 'keyring.csv' using the Import Keys Utility (import_keys.py).

The KMS maintains a Master Key File in CSV format for the ULP network's Gateway key, code download (CDLD) key, and the node root keys for all authorized nodes on the system. As nodes are manufactured, they are provisioned by the LKS with security keys. These security keys are exported from the LKS to a file (referred to as the batch key file) at user-defined intervals, along with a signature file and a manifest file. All three of these files must be delivered from the LKS to the KMS.

The batch key file is typically:

- Encrypted using the KMS RSA public key
- Digitally signed using a symmetric key shared between the KMS and LKS
- Transferred to the KMS

Usually, only the keys for the most recent batch of nodes are exported from the LKS to the KMS.

When received by the KMS, the Import Keys Utility (import_keys.py) does the following:

- Verifies the signature
- Decrypts the batch key file
- Verifies that the node IDs in the batch key file match the node IDs in the manifest file
- Merges the keys into the Master Key File

The KMS Master Key File is created using the Generate Gateway Keys Utility (generate_gw_keys.py) which also generates the Gateway key and the code download (CDLD) key. These two keys are exported to a file (the Gateway key file) which is encrypted using the symmetric key shared between the KMS and LKS and signed using the KMS private key.

The Gateway key file is then sent to all LKS machines that provision nodes for the system. The shared symmetric key is generated on the KMS side from a user-entered passphrase when the

Master Key File is created by the Generate Gateway Keys Utility (generate_gw_keys.py). The shared symmetric key is generated on the LKS side from a user-entered passphrase when the node key database is created using the Import Gateway Keys Utility (import_gw_keys.py). The passphrase used by KMS must be the same as the passphrase used by the LKS.

NOTE: The Gateway key and the code download (CDLD) key are unique to the network operator. The network operator sends these keys to all of its supplier factories.

The following figure provides an overview of network key generation and export.

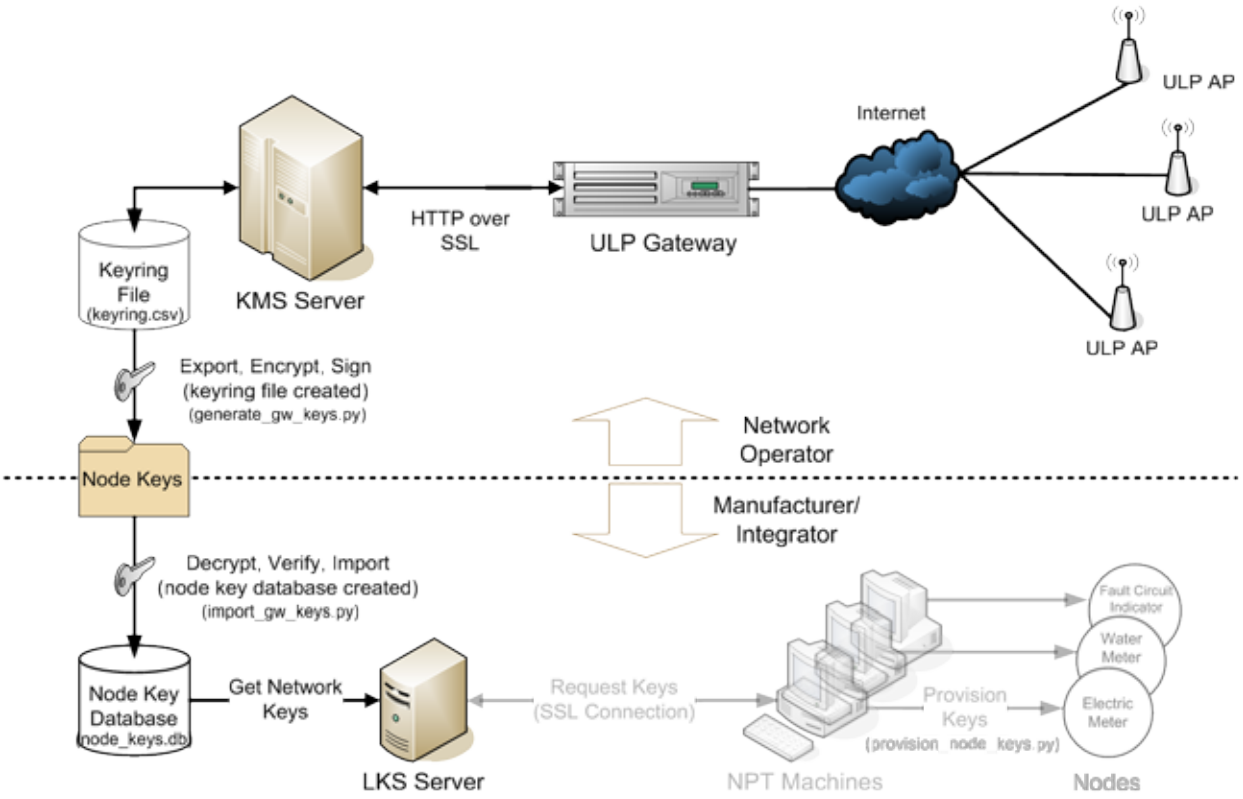

Figure 1. ULP Network Key Generation and Export

3 Key Management Server

The Key Management Server (KMS) is a physically secured server that uses the standard Diameter protocol (RFC3588) to communicate with one or more Gateway servers. The Gateway server requests keys from the KMS and the KMS implements the Diameter protocol which services Gateway node key requests. The KMS server maintains a Master Key File (keyring.csv) in comma separated values (CSV) format that contains the following keys for all nodes it serves:

- Gateway key
- Code download (CDLD) key
- Node-specific root key

The Local Key Server (LKS) generates the node keys upon request by the Node Provisioning Tool (NPT). The LKS stores all of the keys listed above in the local database (for example, node_keys.db). The NPT provisions the Nodes with the Gateway, code download (CDLD) key, and the node root key which are retrieved from the LKS. The LKS can export a subset or all of the node keys to an encrypted batch key file (for example, exported_keys.csv.rsa).

Upon user request, the LKS exports batch key files which should be securely delivered to the KMS. The Master Key File (keyring.csv) used by the KMS is built from one or more of these exported batch key files by merging them one by one into the existing Master Key File. The KMS reads the Master Key File and serves these keys to the Gateway using the Diameter protocol. When the Gateway starts up, it searches its local cache for Gateway and code download (CDLD) keys. If these keys are not available, then it requests them from the KMS and stores the keys in its local cache. Node keys are only requested from the KMS as new nodes join the network; the node keys are then cached locally on the Gateway.

The following figure illustrates how the KMS interfaces with other components in the system.

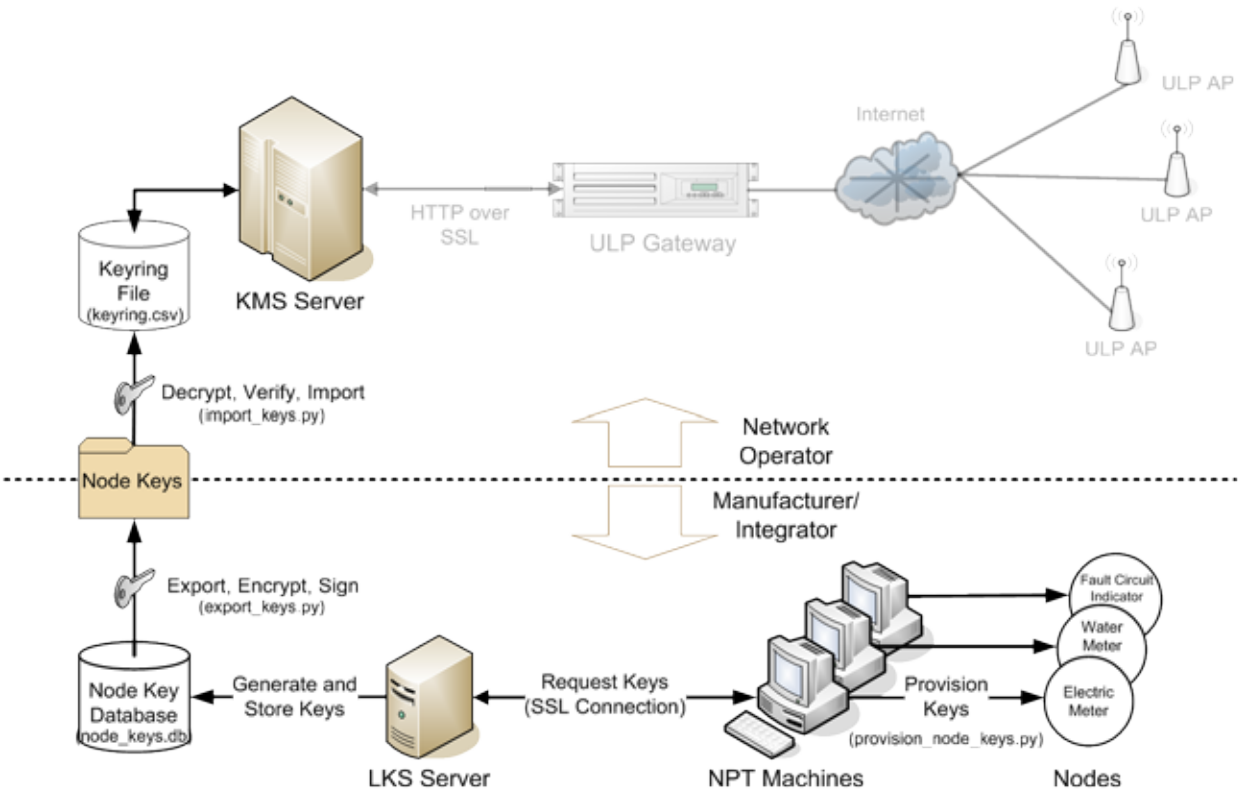

Figure 2. ULP Node Key Provisioning (Operational Mode)

4 Installing the Software

4.1 System Requirements

The system requirements for the KMS/KMC are as follows:

- An enterprise-level server running the 64-bit version of CentOS 6 or Red Hat® Enterprise Linux® (RHEL) 6 operating system (OS)

  NOTE: The KMS has been tested by On-Ramp Wireless on a system running CentOS 6/RHEL 6 operating system.

- Python 2.6 including the module for PyCrypto (version 2.0.1 or 2.1.0). For PyCrypto software installation instructions, refer to section 4.3.
- If using Diameter protocol, a license from F5 Networks (formerly Traffix Systems) must be acquired in order to configure this protocol on the KMS and KMC (which is located on the Gateway server). A license key is issued from F5 Networks for every site deployment and this license is machine dependent. Contact support@onrampwireless.com to obtain this license.

4.2 Security Requirements

As with any enterprise-level server, a number of basic and commonly accepted security precautions should be taken to protect the KMS server and its contents from unauthorized access. The following security precautions are recommended for the KMS server. Additional precautions can be taken as deemed appropriate.

- The KMS server should be placed in a physically secured environment (for example, a locked server room).
- The KMS server should use an operating system that has been configured to restrict access to only authorized and authenticated users using well-established access control mechanisms (for example, username/password with appropriate group permissions, etc.).
- Services on the KMS server that do not use a secure protocol should be disabled (for example, Telnet, FTP).
- Unneeded and unused services on the KMS server should be disabled.

For additional security measures, refer to the following NSA documentation which contains hardening tips and security configuration recommendations for RHEL 5, which are also applicable to CentOS 5 systems. These documents reference RHEL 5, however, the security recommendations are generally applicable for RHEL 6.

- www.nsa.gov/ia/_files/factsheets/rhel5-pamphlet-i731.pdf
- www.nsa.gov/ia/_files/os/redhat/rhel5-guide-i731.pdf

4.3 Installing PyCrypto

To install PyCrypto on a CentOS 6/RHEL 6 system for the Python 2.6 installation, follow the steps below.

1. Download PyCrypto 2.1.0 as follows:
   ```
   wget http://www.pycrypto.org/files/pycrypto-2.1.0.tar.gz
   ```
2. Extract PyCrypto and enter the extracted directory as shown below.
   ```
   tar -zxvf pycrypto-2.1.0.tar.gz
   cd pycrypto-2.1.0
   ```
3. Install PyCrypto directly by typing the following:
   ```
   python setup.py install
   ```
   The result should display the following information as the last two lines of error-free output.
   ```
   running install_egg_info
   Writing /usr/lib/python2.6/site-packages/pycrypto-2.1.0-py2.6.egg-info
   ```

4.4 Installing the KMS Software

The KMS software runs on a server with CentOS 6 or RHEL 6 operating system. KMS software has two versions: one that supports HTTP over SSL protocol and the other supports Diameter protocol to communicate between Gateway and KMS.

4.4.1 Installing HTTP over SSL Protocol Version of KMS

If you want to use HTTP over SSL protocol, install the KMS software using the following command as root:

```
rpm -Uvh ulp-kms-1.4.4-34770.x86_64.rpm
```

After installing the KMS software, the KMS binary files are located in the /opt/ulp/kms directory. The ulp_kms.conf file is located in the /etc directory.

4.4.2 Installing Diameter Protocol Version of KMS

If you want to use Diameter protocol, install the KMS software using the following command as root:

```
rpm -Uvh ulp-kms-1.4.4-34770_openblox.x86_64.rpm
```

After installing the KMS software, the KMS binary file is located in the /opt/ulp/kms directory. The diameter_server.conf file and the ulp_kms.conf file are both located in the /etc directory.

4.5 Installing the KMS Key Utilities

The KMS key utilities are scripts that generate and update the Master Key File (keyring.csv) and are released as a tar file (provisioning_kms.tar.gz). To install and configure the KMS key utilities, complete the following steps.

NOTE: The person performing this installation ***must*** have root privileges on the server.

1. Identify a parent directory location on the KMS where a subdirectory is to be created containing the KMS key utility files. Ideally, this parent directory would be the same location as the KMS Master Key File. The subdirectory is created when the KMS key utilities are extracted. If the parent directory (for example, /opt/ulp/kms) does not already exist, create it using the following command as root:

   ```
   mkdir -p /opt/ulp/kms
   ```

2. Go to the parent directory as follows:

   ```
   cd /opt/ulp/kms
   ```

3. Copy the 'provisioning_kms.tar.gz' file to this directory. The 'provisioning_kms.tar.gz' file is a GNU zipped tarball file that contains the necessary files for the KMS key utilities.
4. Unzip and untar the 'provisioning_kms.tar.gz' file with the following command:

   `tar xzf provisioning_kms.tar.gz`

   The KMS key utility files will be extracted into the directory /opt/ulp/kms/kms-utils. The 'kms-utils' subdirectory is created by the extraction process if it does not already exist.
5. After the key utility files have been extracted, copy the KMS server's private RSA key, in the unencrypted Privacy Enhanced Mail (PEM) file format, to the /opt/ulp/kms/kms-utils directory.

   NOTE: This is needed to decrypt the batch key file exported from the LKS and encrypted with the KMS server's public RSA key. The KMS private RSA key file is also needed to digitally sign the exported Gateway key file when the Master Key File is created. For instructions on how to generate RSA key pairs, refer to Appendix A: Creating RSA Keys.

4.6 Installing the KMC Software

The KMC software is part of the Gateway. For more information, see the *Gateway Software Installation Guide (010-0051-00)*.

5 Configuring the KMS Software

After changing any of the following configuration files, you must restart the KMS process with the following command as root:

```
/sbin/service ulp-kms restart
```

5.1 KMS Configuration File

The KMS configuration file (/etc/ulp_kms.conf) contains the location of the master key file (keyring.csv). The default configuration is shown below:

```
KEYRING=/etc/keyring.csv
```

The location for the master key file can be specified by updating KEYRING parameter in the KMS configuration file.

5.2 Master Key File

The Master Key File (keyring.csv) contains the Gateway and node keys that the KMS provides to serve the key requests that come from the Gateway. The installation includes the 'keyring.csv' file that runs on the KMS. The 'keyring.csv' file must be populated with node keys from the batch files generated by the LKS. The default location for the unencrypted 'keyring.csv' file is /etc directory. If you place this file in a different location, you must update /etc/ulp_kms.conf to reflect this change.

5.3 Diameter Configuration File

NOTE: The Diameter configuration file only needs to be configured when using the Diameter protocol version of the KMS software.

The Diameter configuration file (/etc/diameter_server.conf) contains the information to configure the Diameter protocol, such as the address of peers and clients to allow connections and/or Diameter protocol security options. Edit the '/etc/diameter_server.conf' file to match your environment.

5.3.1 Configuring Parameters for Diameter Protocol

Use the Diameter configuration file to configure the OpenBlox Diameter Stack. The Diameter configuration file contains multiple parameters, each specified in a separate line with the following format:

```
propertyName=propertyValue
```

The following table lists and describes the properties that are stack-specific and configure the stack to be either client or server depending on the values of the properties.

Table 2. Stack-Specific Properties for the Diameter Configuration File

Property Name	Description	Example
URI	Defines the Universal Resource Identifier (URI) of a Diameter node for both the client and the server. The URI must contain one of the following: n aaa://FQDN[:PORT] n aaas://FQDN[:PORT]	n URI=aaa://127.0.0.1:2045 n URI=aaa://10.50.4.40 n URI=aaas://kms.onramp.com
Realm	Defines the realm.	Realm=onramp.com
Vendor ID	Represents the vendor ID. The Internet Assigned Numbers Authority (IANA) assigns this number to vendors. Vendors use this number during the exchange of capabilities.	VendorId=27611
Product Name	Represents the product name. The product name is used during the exchange of capabilities.	n ProductName=KMS n ProductName=KMC
Supported Application IDs	Nested parameter that contains the supported application IDs. Each application ID entry contains: n Application ID n Vendor ID	n SupportedApplicationIds.0.ApplicationIdType=Auth n SupportedApplicationIds.0.ApplicationId=2 n SupportedApplicationIds.0.VendorId=0
In-band Security IDs	Nested parameters that contain supported security IDs.	n InbandSecurityIds.0.SecurityId=TLS n InbandSecurityIds.1.SecurityId=NO_INBAND_SECURITY
Peer Table	Contains statically configured information about the Diameter peer node (server or client). Each entry for the PeerTable parameter contains: n URI: Mandatory field that contains the URI of the peer. n Realm: Contains the realm of the peer. PeerConnecting: If this is set, an attempt to connect to this peer occurs when the Diameter stack starts up. Clients usually set this parameter to connect to the server at start up.	n PeerTable.0.URI=aaa://server.onramp.com n PeerTable.0.Realm=onramp.com n PeerTable.0.PeerConnecting=true
Stack Type	Defines the type of Diameter stack (server or client).	n StackType=SERVER n StackType=CLIENT

NOTE: Specify the following parameters for the KMS Diameter configuration file.

- **ApplicationID=2 for the SupportedApplicationIds property.**
  The KMS uses Mobile IPv4 (RFC4004) as the application interface over base Diameter protocol to exchange the Gateway and the node keys.
- **Fully Qualified Domain Name (FQDN) in the URI property.**
  The FQDN in the URI property must match your computer's Internet Protocol (IP) address or FQDN.

5.4 Running the KMS and the KMC Software on the Same Computer

The default configuration files work with a Gateway that runs on the same computer as the KMS, without any modifications required.

5.5 Running the KMS and the KMC Software on Different Computers

If the KMS and the Gateway are on different computers, follow the instructions below for the protocol that you are using.

5.5.1 HTTP over SSL Protocol

The KMS software uses Apache server (httpd) to serve the keys. The Apache configuration file used by KMS is located at /opt/ulp/kms/www/conf/httpd.conf. This configuration file contains the TCP port to which KMC connects. The default port is 8443. If the port is changed from the default, then the following files must be modified. For example, if the KMS runs on a computer with an IP address = <server-ip> and listens on TCP port = <port> then update the default configuration on KMS and Gateway as follows.

- On KMS (/opt/ulp/kms/www/conf/httpd.conf), edit the following parameter:

```
Listen <port>
```

- On Gateway (/etc/ulp_gateway.conf), edit the following parameters:

```
[kmc]
KmsHostname=<server-ip>
KmsPort=<port>
```

NOTE: If using IP tables, modify the KMS server (as root) by adding the following line to /etc/syscconfig/iptables using the <port> to which KMC connects. The following is an example using the default port (TCP:8443).

```
A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 8443 -j ACCEPT
```

Then restart the iptables process by typing the following command as root:

```
/sbin/service iptables restart
```

5.5.2 Diameter Protocol

Change the default Diameter configuration files to connect the KMS to the Gateway. For example, if the KMS runs on a computer with an IP address = <server-ip>, and the Gateway runs on a computer with an IP address = <client-ip>, then update the default configuration on KMS and Gateway as follows.

- On KMS (/etc/diameter_server.conf), edit the following parameters:

```
URI=aaa://<server-ip>
StackType=SERVER
Realm=<server-realm>
```

- Gateway (/etc/diameter_client.conf), edit the following parameters:

```
URI=aaa://<client-ip>
StackType=CLIENT
Realm=<client-realm>
PeerTable.0.URI=aaa://<server-ip>
PeerTable.0.Realm=<server-realm>
PeerTable.0.PeerConnecting=true
```

NOTE: If using IP tables, modify the KMS and Gateway servers (as root) by adding the following line to /etc/syscconfig/iptables (default port for Diameter protocol is TCP:3868):

```
A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 3868 -j ACCEPT
```

Then restart the iptables process by typing the following command as root:

```
/sbin/service iptables restart
```

5.6 Using Secure Communication between the KMS and the Gateway Servers

NOTE: This section is only required if using the Diameter protocol version of the KMS software.

The default installation and default Diameter Configuration does ***not*** set up a secure communication between the KMS and the Gateway servers, however, a secure communication can be set up using IPsec. IPsec is used for securing the server-to-server communication, in this case KMS to Gateway. Set up IPsec on the KMS and Gateway servers with a pre-shared key. To do this, follow the steps below.

1. Install the ipsec-tools RPM package.
2. Add the following lines to /etc/racoon/racoon.conf on the KMS and Gateway servers.

   For the KMS:
   `include "/etc/racoon/<gw-ipaddress>.conf";`

   For the Gateway:
   `include "/etc/racoon/<kms-ipaddress>.conf";`

3. Create the file /etc/sysconfig/network-scripts/ifcfg-ipsec0 on both the KMS and the Gateway and add the following entries as specified.

   For the KMS:
   `DST=<gw-ipaddress> TYPE=IPSEC ONBOOT=yes IKE_METHOD=PSK`

   For the Gateway:
   `DST=<kms-ipaddress> TYPE=IPSEC ONBOOT=yes IKE_METHOD=PSK`

4. Create the pre-shared key file /etc/sysconfig/network-scripts/keys-ipsec0 on ***both the KMS and the Gateway*** and add the following entry.

   ```
   IKE_PSK=<pre-shared-passphrase>
   ```

   NOTE: You can select your own secure pre-shared passphrase.

5. Start IPsec using the following command:
   ```
   /sbin/ifup ipsec0
   ```

6. To verify the IPsec (AH and ESP), run the following command on the KMS.
   ```
   /usr/sbin/tcpdump –i any host <gw-ipaddress>
   ```

5.7 Restarting the KMS and Gateway Servers after Configuration

Restart both the KMS and the Gateway servers as follows. The Master Key File 'keyring.csv' file should have already been generated by the 'generate_gw_keys.py' utility. Ideally, the node keys from the LKS should have been imported into the Master Key File.

For the KMS:
```
/sbin/service ulp-kms restart
```

For the Gateway:
```
/sbin/service ulp-gateway restart
```

5.8 Location of KMS Log Files

5.8.1 Log Files for HTTP over SSL Version of KMS Software

The log files for the KMS are:

- /var/log/httpd/kms_error_log
- /var/log/httpd/kms_ssl_error_log

The log file for the KMC (on the Gateway) is:

- /opt/ulp/gateway/logs/gw.log

5.8.2 Log Files for Diameter Protocol Version of KMS Software

The log files for the KMS are:

- /tmp/kms_error.log
- /opt/ulp/kms/logs/kms-<date timestamp>

The log files for the KMC (on the Gateway) are:

- /tmp/kmc_error.log
- /opt/ulp/gateway/logs/kmc-<date timestamp>

6 Uninstalling the KMS Software

To uninstall the KMS software on a CentOS 6/RHEL 6 computer, run the following command as root:

```
rpm -e ulp-kms
```

Appendix A Creating RSA Keys

These instructions describe the steps necessary to create an RSA public/private key pair. RSA key pairs must be generated for secure communication between entities such as a Local Key Server or a Node Provisioning Tool client. Note that RSA key generation does not need to be performed on the computer that will use the keys.

NOTE: Creation of RSA keys can be performed on any computer and copied to another computer. However, care must be taken when transferring a private key. Generally, private keys should not be transferred off of the target machine. However, it is acceptable to create a private key on a secure server for a target machine and then securely transfer the private key to the target machine.

1. Create a 2048-bit RSA private key using the following command. Note that the filename is user-defined.

   ```
   openssl genrsa –out <kms_key.priv.pem> 2048
   ```

   NOTE: This command does not contain the '-des3' option which creates an encrypted private key file. PyCrypto does not support encrypted private key files.

2. Create an RSA public key from the private key. Note that some filenames are user-defined.
   ```
   openssl rsa –in <kms_key.priv.pem> –pubout > <kms_key.pub.pem>
   ```
3. The 'kms_key.pub.pem' file must be copied to LKS server so that LKS can securely export node keys to the KMS server.

Be sure to follow these safeguards:

- ***Never distribute or disclose the private key. Only distribute the public key.***
- Encrypt messages using the public key of the intended recipient.
- Decrypt received messages using the private key. The message should be encrypted by the sender using the public key.
- Sign transmitted messages using the private key.
- Verify/authenticate signatures using the public key of the sender.

Appendix B ULP Network Key Generation, Provisioning, and Backup

This appendix provides instructions on how to generate and export gateway keys for distribution to one or more LKS sites as well as how to import and merge an incremental batch key file exported from the LKS into the Master Key File used by the KMS. The batch key files exported from the LKS contain the node root keys. The KMS Master Key File contains the keys for all nodes authorized to join the network. The KMS Master Key File must be named 'keyring.csv'.

B.1 Generating Gateway Keys

After the KMS key utilities have been installed, the KMS Master Key File must be created. This file is created using the Generate Gateway Keys Utility (generate_gw_keys.py). To create this file, type the following command, all on one line, from the /opt/ulp/kms/kms-utils directory. Note that some filenames are user-defined.

```
python generate_gw_keys.py -m /etc/keyring.csv -o <gw_keys.csv.aes> -k
<kms_key.priv.pem> -v
```

This command does the following:

- Creates the Master Key File 'keyring.csv' in the /etc directory
- Creates an AES encrypted output file 'gw_keys.csv.aes' (filename is user-defined)
- Creates the signature file 'gw_keys.csv.aes.sig'

The encrypted Gateway key file (gw_keys.csv.aes) and its signature file (gw_keys.csv.aes.sig) are then delivered to any LKS server that provisions nodes for the network managed by this KMS.

NOTE: The Generate Gateway Keys Utility (generate_gw_keys.py) allows any arbitrary name to be used for the Master Key File. However, the KMS software expects the file to be named 'keyring.csv' and to be located in the /etc directory.

When the Master Key File is created by the command indicated above, the user is prompted to enter a passphrase which is used to encrypt the output file containing the Gateway keys. This passphrase is also used to verify the signature of batch key files to be imported from the LKS. This passphrase must be identical to the passphrase used by the LKS when the node key database is created. It is recommended that commonly accepted security practices be observed regarding the selection of the shared passphrase, such as:

- Avoid words that are in the dictionary
- Avoid words, terms, phrases, names, and dates that are easy for others to guess
- Avoid using short passphrases
- Use both upper and lower case characters
- Use numbers and special characters (for example, #, $, ^, *, &, ~, !)

B.2 Node Key Provisioning

Each Node must be provisioned with three security keys:

- Gateway key
- Code download (CDLD) key
- Node-specific root key

The Gateway key and the code download (CDLD) key are used by all nodes on a network. However, each node has its own unique node root key. Node provisioning is performed on the NPT client using Node Provisioning Tools. For details on node provisioning, see the *NPT User Guide (010-0060-00)*.

B.3 Process for Nodes to Join the Network

1. In order for a deployed node to join the network, it must be authorized by the KMS.
2. The node's join request is first received by a ULP Access Point which then forwards the request to the Gateway.
3. The Gateway sends the request to the KMS which checks its Master Key File to determine if the node is in the list.
   a. If the node's ID is in the list, the node's root key is sent to the Gateway. The Gateway then allows the node to join the network.
   b. If the node's ID is not recognized by the KMS, the node is not allowed to join the network.
4. When a batch key file is received from the LKS, it is digitally signed using an AES-CMAC hashing algorithm and encrypted using the KMS server's public key.

   NOTE: The AES key used for the AES-CMAC hashing algorithm is generated by a secret passphrase shared by both the LKS and KMS. This passphrase was entered when the KMS Master Key File was created using the Generate Gateway Keys Utility (generate_gw_keys.py).
5. If the digital signature of the imported batch key file cannot be verified, an error is generated and the key file is not decrypted or imported.
6. As new batch key files are received from the LKS (along with their corresponding signature file and manifest file), they can be imported and merged into the existing Master Key File ('keyring.csv') using a command similar to that shown below (all on one line). Note that some filenames are user-defined.

   ```
   python import_keys.py -m /etc/keyring.csv -i <batch_keys7.csv.rsa> -k
   <kms_key.priv.pem> -s –v
   ```

NOTE: After merging the new batch keys into the existing 'keyring.csv' file, the KMS process must be restarted using the following command as root:

`/sbin/service ulp-kms restart`

If the KMS process is not restarted after updating the 'keyring.csv' file, the KMS will not be able to provide the new keys to the Gateway.

B.4 Master Key File

B.4.1 Backup

When importing a batch key file, a time stamped backup copy of the Master Key File can be optionally generated using the '-b' (or '--backup') command line option. The original Master Key File, prior to importing and merging, is renamed to a file containing a time stamp with the format 'YYMMDDhhmmss'. For example, the Master Key File 'keyring.csv' is renamed to 'keyring.110222172503.csv'.

B.4.2 File Format

The first four (4) lines of the Master Key File (keyring.csv) contain, in the following order:

1. 16-byte passphrase hash
2. 16-byte random base used by the LKS for its Key Generation Function (KGF)
3. 24-byte 3DES Gateway key
4. 16-byte code download (CDLD) key

All subsequent lines in the Master Key File contain the following four columns:

1. node ID
2. node root key
3. batch number
4. reprovisioned count

An example Master Key File is provided below:

```
pwdh,0xa39b3510d0ef8d3b5a2e701bf568c491
base,0xf08916b6f998ad78ee31079c9afdca0f
gateway,0x015723c7196862208f6bfb1a5219f725ae041a79a1673ebc
gateway_cdld,0x2cd4cea3efe06e6ce9541954000cc055
0x00010200,0x370ed34aef1835709b7eedf259d83fc5,1,0
0x00010201,0xce9d3fccf3a6a0767a83e2d5f5bd671c,1,0
0x00010202,0xf619bf254fb36e8cc3bd7ed5f8277104,1,0
0x00010203,0xf00a566d63bc69e9fc1d3eaa82b00d61,2,0
0x00010204,0x1104aa0791a2cd960a60762f9e7bebbe,2,0
```

Appendix C Abbreviations and Terms

Abbreviation/Term	Definition
3DES	Triple Data Encryption Standard
AES	Advanced Encryption Standard
AH	Authentication Header
AP	Access Point. The ULP network component geographically deployed over a territory.
CMAC	Cipher-based Message Authentication Code
CSV	Comma Separated Values
ESP	Encapsulating Security Payloads
FQDN	Fully Qualified Domain Name
FTP	File Transfer Protocol
IANA	Internet Assigned Numbers Authority
IP	Internet Protocol
IPsec	Internet Protocol Security
KGF	Key Generation Function
KMC	Key Management Client.
KMS	Key Management Server
KPA	Key Provisioning Agent
LKS	Local Key Server
NMS	Network Management System. The network component that provides a concise view of the ULP network for controls and alarms.
NPT	Node Provisioning Tool
Node	The ULP wireless module developed by On-Ramp Wireless that integrates with OEM sensors and communicates sensor data to an Access Point. Also, the generic term used interchangeably with end point device.
OS	Operating System
PEM	Privacy Enhanced Mail
ULP	Ultra-Link Processing™. The On-Ramp Wireless proprietary wireless communication technology.
URI	Universal Resource Identifier