Oracle Utilities Network Management System

Security Guide

Release 2.5.0.2.0

F54386-01

March 2022

Oracle Utilities Network Management System Quick Install Guide, Release 2.5.0.2.0

F54386-01

Content

Content

Preface .... **i-i**
Audience .... i-i
Related Documents .... i-ii
Conventions .... i-ii

Chapter 1
Security **1-1**
NMS Security 1-1
Oracle Network Management System Data Sensitivity 1-2
Oracle Network Management System Security Philosophy 1-3
Oracle NMS Technology Components By Tier 1-5
Tier 1 - UNIX Files and Oracle RDBMS - Data/Config Tier 1-6
Tier 2 - Oracle Network Management System Services - Business Tier 1-7
Tier 3 - Oracle WebLogic Java Application Server - Data Access Tier 1-10
Tier 4 - Oracle NMS Clients - Presentation Tier 1-13
Tier 5 - Oracle NMS Mobile Clients - Mobile Presentation Tier 1-15
Additional Security Options for Key Technology Components 1-16
Oracle Relational Database Management System 1-16
Network Management System Services 1-20
Oracle WebLogic Server 1-21
Common Object Request Broker Architecture - Object Request Broker 1-22
HTTP Server 1-23
LDAP Server 1-24
Network Management System Java Client Applications 1-24
Security Certificates in Oracle Utilities Network Management System 1-25
SSL Certificate 1-25
Defining Encryption Parameters 1-26
Validating Encryption Suite 1-26
Trusting Certificates 1-26
Signing Certificates 1-26
Using a Signing Keystore From Another Source 1-27
Creating the Client Keystore 1-27
Creating the Certificate for SwService 1-28
Open Ports 1-28
Client/Server Security 1-29
Attachment Security 1-29
General Data Protection Regulation 1-30

Preface

Welcome to Oracle Utilities Network Management System Security Guide. This guide describes how you can configure security for Oracle Utilities by using the default features.

This preface contains these topics:

- Audience
- Related Documents
- Conventions

Audience

Oracle Utilities Network Management System Security Guide is intended for product administrators, security administrators, application developers, and others tasked with performing the following operations:

- Designing and implementing security policies to protect the data of an organization, users, and applications from accidental, inappropriate, or unauthorized actions
- Creating and enforcing policies and practices of auditing and accountability for inappropriate or unauthorized actions
- Creating, maintaining, and terminating user accounts, passwords, roles, and privileges
- Developing interfaces that provide desired services securely in a variety of computational models, leveraging product and directory services to maximize both efficiency and ease of use

To use this document, you need a basic understanding of how the product works, and basic familiarity with the security aspects of Oracle WebLogic and Database security.

Related Documents

For more information, see the following documents in the Oracle Utilities Network Management System Release 2.5.0.2.0 documentation set:

- *Oracle Utilities Network Management System Adapters Guide*
- *Oracle Utilities Network Management System Advanced Distribution Management System Implementation Guide*
- *Oracle Utilities Network Management System Configuration Guide*
- *Oracle Utilities Network Management System for Water User's Guide*
- *Oracle Utilities Network Management System Installation Guide*
- *Oracle Utilities Network Management System Licensing Information User Manual*
- *Oracle Utilities Network Management System Operations Mobile Application Installation and Deployment Guide*
- *Oracle Utilities Network Management System Quick Install Guide*
- *Oracle Utilities Network Management System Release Notes*
- *Oracle Utilities Network Management System User's Guide*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Chapter 1

Security

This chapter provides an overview of a standard implementation of Oracle Utilities Network Management System, including:

- NMS Security
- Oracle Network Management System Data Sensitivity
- Oracle Network Management System Security Philosophy
- Oracle NMS Technology Components By Tier
- Additional Security Options for Key Technology Components
- Security Certificates in Oracle Utilities Network Management System
- Open Ports
- Client/Server Security
- Attachment Security

NMS Security

The Oracle Utilities Network Management System (NMS) is comprised of several technology components that necessarily interact to provide a fully functional system. This document is intended to provide an overview of those components along with some of the options available to either enhance or ease security for each of the key components. By default Oracle attempts to configure each component as secure as practically possible by default. In some cases additional security can be provided beyond the default configuration. In other cases (maybe an internal development or test environment for example) it may be desirable to reduce security for certain components - to simplify support and/or interaction with the system.

Oracle Network Management System Data Sensitivity

The Oracle Utilities Network Management System minimally combines data from Geographic Information Systems (GIS) and Customer Information Systems (CIS) to form the operational electrical network model. Beyond GIS and CIS, NMS production deployments often include at least some data from one or more of the following:

- Supervisory Control and Data Acquisition (SCADA) systems
- Advanced Metering Infrastructure (AMI) systems
- Mobile Workforce Management systems
    - Oracle internal. For example, the NMS Operations Mobile Application (OMA), Oracle Field Service Cloud (OFSC), and so forth.
    - Oracle external. For example, CGI, Ventyx, and so forth.
- Work and Asset Management systems (WAM)
- Engineering Planning systems

Other more utility specific enterprise systems are also often integrated to Oracle NMS to help maintain the real-time operational NMS Electrical Network Data model (NMS model).

A typical Network Management System implementation has very little truly sensitive personal (utility customer data) involved in the construction or maintenance of the NMS model. The required personal information that is periodically extracted from the appropriate Customer Information System is generally used to identify or categorize utility customers. The required information typically includes customer name, address, phone and utility account number. Optional fields like "life support equipment," critical customer classification information and power usage data can also be used to help categorize customers for routine analysis. The majority of NMS Electrical Network Data model information is publicly available and as a whole is generally not considered highly sensitive.

Because the persistent data behind the Oracle Network Management System is not generally considered highly sensitive and because NMS systems typically reside behind multiple firewalls, NMS customers do not typically encrypt "data at rest" (on disk) or "on-the-wire/in-flight" (between core NMS components like the RDBMS and WebLogic). Specifically the vast majority of relevant Network Management System data is stored in an Oracle RDBMS. The NMS data that resides in the RDBMS is typically not encrypted on disk nor in-transit between the RDBMS and WebLogic by default - but can be.

Oracle Network Management System Security Philosophy

A major mission of the NMS electrical model is to allow efficient capture and sharing of static and real-time field updates that allow for more informed and coordinated operational decisions. As such primary security measures typically involve minimizing the possibility of unauthorized (end-user) access or manipulation of the production Network Management System model. The most secure data cannot be accessed by anyone. A system that anyone can openly access and update may perform well but is certainly not secure. We have to find an acceptable balance of accessibility and security while providing a sufficiently performant system. The right balance generally depends on available infrastructure, cost as well as the needs and priorities of your organization.

At a high-level, a typical NMS installation can typically be thought of as two major components. A front-end user interface and back-end (black box) services to support the user interface. These two primary components are expected to be separated by an internal firewall for a production NMS deployment.

1. The NMS front end provides end-user clients access to the NMS electrical model. NMS supports Java (Swing) user client and limited user access with the browser based NMS Flex client.
    - NMS end users are typically all within the corporate intranet behind a corporate firewall.
    - Internet access (without VPN) is generally not allowed within an NMS production environment.
        - The optional NMS Operations Mobile Application (OMA) can be an exception to this rule. OMA can be configured to support foreign crews and (if configured to do so) must (typically) allow some form of Internet access. OMA has its own security section later in this document.
    - A large NMS instance may have over 1000 authorized users (sometimes over 500 at any given time). This user volume alone qualifies NMS end-users as a legitimate security threat. End user tools/access needs to be closely monitored/managed to help ensure NMS model integrity.
2. The NMS back end manages the operational NMS electrical network model.
    - NMS back end technology components are expected to be deployed behind an additional internal firewall. The internal firewall is expected to be deployed in front of WebLogic - limiting NMS end user access to NMS back-end "services" via a single WebLogic port.
    - Only internal authorized (admin) users should have access granted to the Oracle NMS back-end technology/service components (RDBMS, WebLogic, NMS services, GIS, CIS, and so on).
        - Care must be taken to ensure that ONLY authorized administrative users have access to the Oracle NMS admin account.

The NMS high-level architecture and major product interfaces are shown in the illustration below:

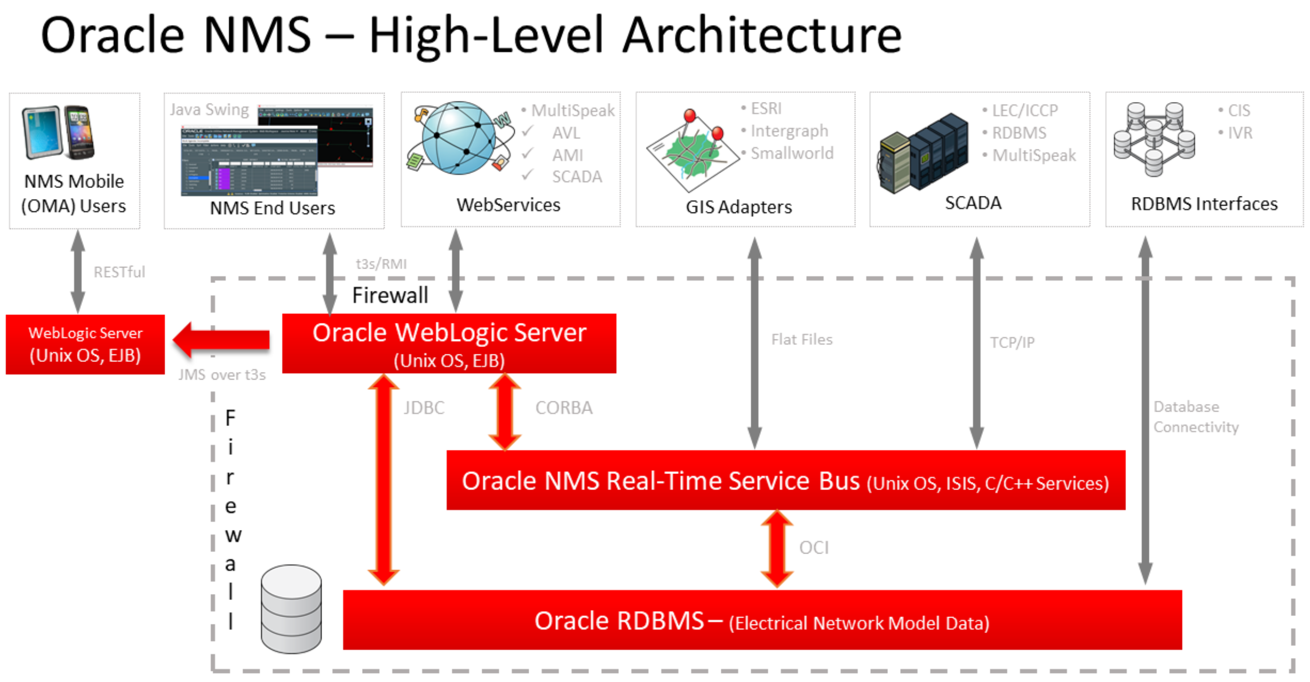

Figure 1: Network Management System High Level Architecture

The following are generally relevant factors when considering what level of security/ encryption is appropriate for core technology components within a given NMS deployment.

1. NMS generally does not typically contain extremely sensitive data.
2. NMS requires digital signatures to validate NMS Java clients connect to a trusted server.
3. NMS supports encryption of traffic between NMS Java clients and WebLogic.
4. NMS Java clients only have read access to a subset of (mostly) project configurable NMS RDBMS tables. Oracle NMS RDBMS read-only tables used for NMS end-user access should be screened for sensitive information before being configured for direct Oracle NMS client access.
5. NMS supports installing a firewall between NMS clients and the NMS back end (WebLogic).

The focus of NMS security revolves around properly authenticating and authorizing external integration adapters and NMS clients to minimize the possibility of unauthorized or inappropriate updates.

Data that travels between NMS end user clients and WebLogic can be restricted (via a firewall) to help prevent unauthorized access to NMS internal (back-end) technology components. Traffic between WebLogic and NMS clients is also typically encrypted (via t3s for WebLogic) to minimize the possibility of it being reverse engineered and/or tampered with and the NMS model being inappropriately updated.

Data that travels between NMS technology components on the back end is often not encrypted (because it is behind the corporate firewall AND often an additional internal firewall). There are also key exceptions to this encryption philosophy on the back-end. For example, communication of sensitive user authentication credentials to/from an LDAP accessible Directory Services is almost always via a secure protocol (LDAPS) - to help minimize the possibility of identity theft.

Together these measures are often adequate to reasonably secure a typical NMS installation - though additional measures can be taken if deemed appropriate. The remainder of this document provides a high-level overview of some of the available NMS security related configuration options.

Oracle NMS Technology Components By Tier

Figure 2, below, shows a high-level overview of the primary technology components, tiers, and ports used in a typical NMS installation.

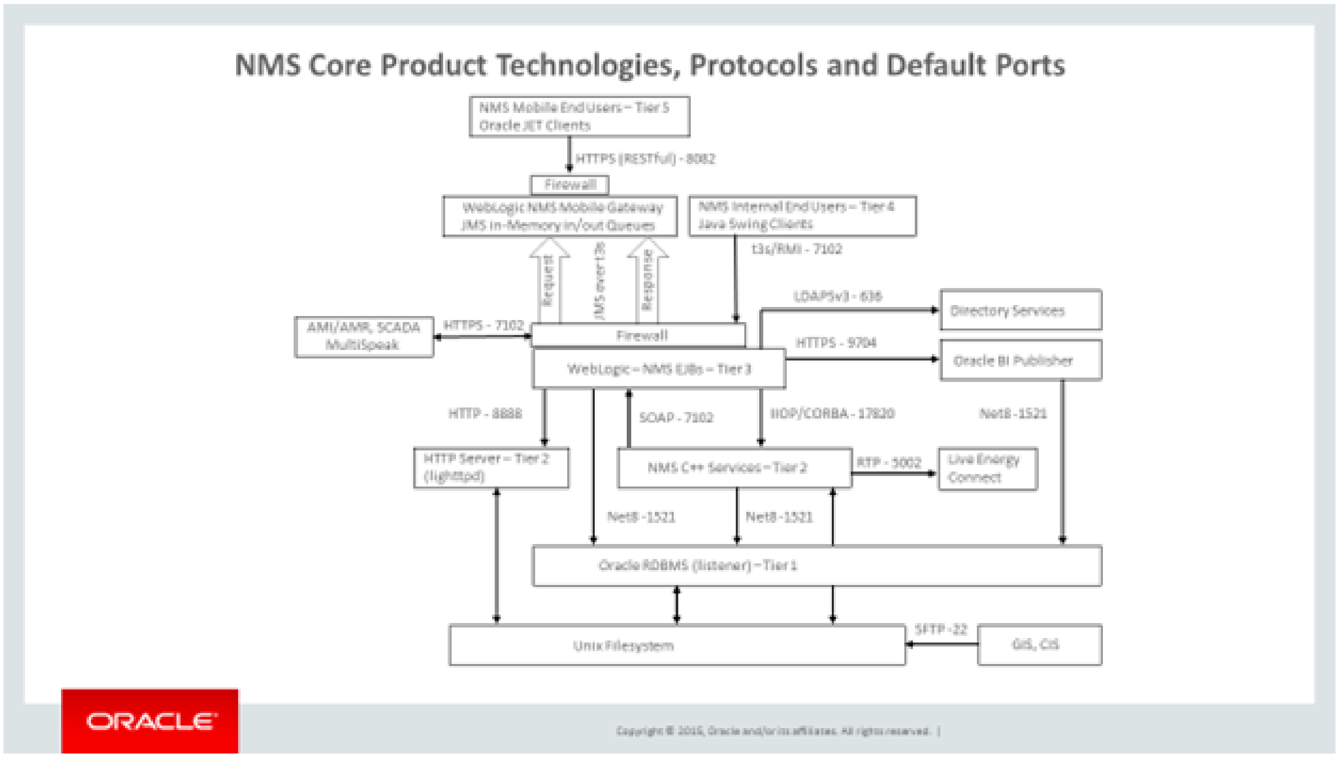

Figure 2: Typical Network Management System: Core Technology, Protocols, and Port Overview

Tier 1 - UNIX Files and Oracle RDBMS - Data/Config Tier

1. **Description/Purpose:**
    - The RDBMS is used to persist the NMS electrical network data model along with outage information, switch plans and analytical information.
    - The UNIX file system is used to hold executables, RDBMS data files, certain electrical and customer model initialization files (generally extracted from enterprise GIS and CIS respectively) and assorted configuration information - for multiple tiers.
2. **Common Inter-Tier Communication and Security Considerations:**
    - The Oracle RDBMS listener process is used to service requests from NMS services, NMS Perl, Python, and SQL*Plus scripts, WebLogic JDBC connections, and (potentially) other project specific applications.
    - NMS installations do not typically encrypt Oracle NMS RDBMS data "at rest" or "in flight."
    - Encrypting all in-transit Oracle RDBMS data has an NMS performance impact (roughly 5% in our internal NMS tests). Because of the internal nature of the typical infrastructure supporting NMS environments (NMS is typically deployed behind multiple firewalls) - in-transit Oracle RDBMS data is not encrypted by default. Still the performance penalty for encrypting all in-transit data should not have a major impact on overall NMS performance. If the NMS RDBMS is exposed to a difficult to monitor/manage set of users - such as other business owners/users outside the core NMS team/ users- encrypting in-transit data should be considered as a viable option to help protect NMS model data from unauthorized access.
    - Oracle RDBMS data "in flight" can be encrypted via the Oracle Enterprise Edition RDBMS supported Native Network Encryption option. This includes any data exchanged between Oracle RDMS instances via Data Guard.
    - Either a subset (specific columns within specific tables) of Oracle RDBMS data "at rest" or entire application tablespaces can also be encrypted via the Oracle Enterprise Edition RDBMS Transparent Data Encryption (TDE) option. If the TDE option is deployed it is recommended to be used for entire tablespaces rather than individual tables and columns - for efficiency. With modern hardware (with on-chip encryption/decryption) the overhead is typically very low (low single digits). Once configured this option is transparent to RDBMS users. The Oracle TDE option is included with Oracle EE RDBMS.
    - By using the Oracle Enterprise Edition RDBMS Advanced Security option all data "at rest" or "on-the-wire" can be encrypted and sensitive data can also be redacted. The Advanced Security option is a separate Oracle EE RDBMS license option.
    - Oracle RDBMS "in flight" data includes data handled by Oracle Data Guard if a backup Oracle NMS RDBMS is in play. The Oracle Data Guard Redo Transport Authentication mechanism leverages the Oracle RDBMS listener as a redo log data transport mechanism.
    - Oracle NMS executable files installed under the nmsadmin Unix user name should never have more than "-rwxr-x---" file permission. The nmsadmin Unix user should have a umask of at least 0027. The default Unix group id

for the nmsadmin username should only be used for Unix accounts that support NMS. For example the Unix group for nmsadmin should NOT be the generic Unix "users" group name (or similar). Suggest an NMS specific Unix group name of oranms (or similar) be used for the default Unix group for any nmsadmin type accounts - and only for those accounts.

- If other enterprise systems require access to the production NMS RDBMS recommend the principle of least privilege be applied - via an alternate RDBMS schema with limited privilege synonyms to the production schema. For example, Data Manipulation Language (DML) access and not Data Definition Language (DDL) access. If only read-only access is required - than configure accordingly. Using this approach should minimize the damage any outside account could potentially have on the production NMS RDBMS. Note this principle can apply to any external integration - including CIS integration.

3. **Common Deployment Options:**
   - Single server supporting a single Oracle RDBMS instance.
   - Two servers in a failover cluster supporting a single Oracle RDBMS instance.
   - Two servers in a Real Application Cluster supporting a single Oracle RDBMS.
   - Disaster Recovery and/or Business Continuity requires at least one additional RDBMS instance - in any of the above 3 configurations. Typically replicated using Oracle Data Guard or Oracle Active Data Guard.

Tier 2 - Oracle Network Management System Services - Business Tier

1. **Description/Purpose:**
   - NMS services (written in C/C++) are used to manage the NMS real-time operational electrical network data model.
   - A significant percentage of NMS project specific configuration is managed within NMS services via a range of configuration options. The vast majority of these configuration options are dependent on the NMS Oracle RDBMS model and configuration tables. These options tend to dictate how the NMS model looks and how it reacts to specific input stimuli. These options are mostly determined during the project configuration phase of an NMS deployment but can be modified by NMS admins after initial go-live.
   - NMS services are tied together by the internal ISIS messaging bus. ISIS is a real-time synchronous/asynchronous high-performance publication/ subscription in-memory messaging bus. NMS services use ISIS to publish relevant updates to each other and to the NMS CORBA gateway to get the messages to WebLogic. WebLogic provides an independent mechanism to cache messages from the NMS CORBA gateway and provides routine updates to NMS clients.
2. **Common Inter-Tier Communication and Security Considerations:**
   - ISIS only runs on the node where NMS services run. By default, ISIS runs on the loopback (localhost) network. In this mode, ISIS is only used to

coordinate messages between NMS service processes and no ISIS ports are accessible from any external nodes. NMS ISIS messages are not encrypted.

- Recommend that a minimal number of (non NMS production account related) Unix accounts be configured on the Unix host (physical or virtual) where production NMS services run. Ideally only required Unix accounts and production NMS admin accounts should have access to these hosts. Minimizing the possibility of someone other than a proper NMS admin accessing production NMS services hosts minimizes the potential attack surface.

- NMS services talk to the Oracle NMS RDBMS listener via Oracle Call Interface APIs. NMS run-time Perl, Python, and SQL*Plus scripts also communicate to the RDBMS via the Oracle NMS RDBMS listener process.
  - The Oracle NMS admin account has access to the Oracle RDBMS (for all of the above mechanisms) via an Oracle wallet installed on the Oracle NMS nmsadmin Unix account. The Oracle wallet stores encrypted RDBMS credentials for NMS daemon process access.
- If NMS services and WebLogic are running on distinct nodes (or under distinct UNIX usernames) there must be a mechanism in place to "serve" the NMS model map (*.mad and *mac) files from where NMS services are running to where WebLogic is running. These map files contain quasi-static coordinate and graphic symbology information for individual NMS electrical model partitions.
  - Using the Network File System (NFS) is not an acceptable option for serving the NMS model map files to WebLogic. NFS suffers from race conditions with the NMS message bus that can yield inconsistent data for the Java clients.
  - The preferred method to serve the NMS model map files is to use an HTTP file server. NMS comes with lighttpd embedded and auto-configured. The HTTP server is generally configured to serve a single directory of unencrypted NMS model map files to specific IP addresses where the WebLogic instances that support this NMS instance run. The NMS system release includes lighttpd in the 3rdparty products package and supporting scripts (nms-lighttpd and nms-lighttpd-oem) to automatically configure, start and stop each lighttpd instance.
    - Lighttpd can be configured to leverage HTTPS.
    - Lighttpd can be configured to limit access to specific file types.
    - Lighttpd can be configured to only allow access from specific hosts.
- NMS services communicate to Enterprise Java Beans within WebLogic via an NMS CORBA adapter. CORBA uses the Internet Inter-Orb Protocol (IIOP) to move data from NMS CORBA adapter (corbagateway) to WebLogic. IIOP traffic is not encrypted.
- NMS services use Simple Object Access Protocol (SOAP) over HTTP to send synchronous request messages to WebLogic that require a distinct response. These messages are authenticated and can also be encrypted via SSL. Oracle Wallet functionality is used to hold authentication credentials for the NMS services that send these SOAP messages to WebLogic.

    - Oracle Wallet functionality is used to hold authentication credentials for the NMS services that send SOAP messages to WebLogic. The nms-env-config script is typically used to configure all Oracle Wallet credentials required to support NMS Services..

- The node that hosts NMS services is typically supplied with a daily or weekly set of routine GIS (map) and CIS (customer model) updates. These updates are commonly in the form of flat files and are often transported via SFTP (Secure File Transfer Protocol) from the appropriate project specific servers to the NMS services server.
    - Oracle recommends the NMS admin account pull routine external updates (GIS, CIS, and so on) into NMS - rather than those accounts push updates into NMS. Minimizing external account access to NMS admin accounts further reduces the potential attack surface.
- Not pictured in Fig. 2 above is a common connection from one of the NMS corbagateway processes to a Simple Mail Transport Protocol (SMTP) server. This is a not always but often used configuration to send utility customer interruption notices and texts to utility internal account reps or other interested parties (via the NMS Service Alert product component option). Outgoing SMTP traffic from CORBA Publisher (corbagateway in publisher mode) is not encrypted.

3. **Common Deployment Options:**

- Typically NMS services are deployed on hardware vendor supported failover (active/passive) clustered infrastructure.
- NMS services can also be deployed on a standalone server - possibly using one or more Disaster Recover/Business Continuity sites for backup.
- For development, test, training, or model validation environments, NMS is often deployed on a server supporting other (similar category) NMS instances. Each NMS instance can be configured to run on a (mostly) distinct set of ports. This generally includes NMS instance specific ports for ISIS, the CORBA Object Request Broker (ORB), SOAP over HTTP and the NMS to WebLogic HTTP server. For a node supporting multiple NMS instances, each NMS instance would be deployed under a unique Unix user name (for example, nmsdev, nmstrain, nmstest). Multiple (non-production) NMS instances are often hosted on a single Oracle RDBMS (under separate schemas).
- In a dual-environment NMS configuration, two Unix user names (for example, nmsadmin1 and nmsadmin2) will each have an independent set of NMS executable files for a given NMS instance.

Tier 3 - Oracle WebLogic Java Application Server - Data Access Tier

1. **Description/Purpose:**

   - WebLogic Managed Servers are used to host NMS specific Enterprise Java Beans (EJBs) that generally cache updates from NMS services or the RDBMS. Caching model updates to improve access for many (hundreds) of NMS clients is a primary goal for this tier.

   Note: A small but significant percentage of NMS configuration options are also managed within these WebLogic EJBs.

   - Separate WebLogic Managed Servers (within the same WebLogic domain) are also used to support Web Service based integration to external systems - typically via the NMS MultiSpeak Adapter (AMI, AVL, SCADA).
   - Separate WebLogic Managed Servers (in different WebLogic domains) can be used to support NMS Operations Mobile Adapter (mobile) clients or NMS Flex (browser) clients. These NMS OMA/Flex specific WebLogic Managed Servers communicate to a matching internal WebLogic Managed Server - often behind a firewall. All communication between WebLogic Servers is initiated from the internal WebLogic server via a pair of JMS queues to the "external" WebLogic Managed Server. This allows communication from the inside out without opening a hole in the firewall from the outside in. In either case, if different WebLogic domains are used, domain trust must be established between the WebLogic Managed Servers involved.

2. **Common Inter-Tier Communication and Security Considerations:**

   - WebLogic uses CORBA to communicate to the NMS services CORBA adapter (corbagateway). CORBA IIOP traffic to/from the NMS CORBA adapter (corbagateway) is not encrypted.
   - WebLogic uses the Java Database Connectivity (JDBC) Thin Client driver (for a non-RAC RDBMS) or Active Gridlink driver (for RAC RDBMS) to communicate to the NMS Oracle RDBMS. These JDBC connections are generally not encrypted but can be via the Oracle Enterprise Edition RDBMS Native Network Encryption option.

   Note: The Oracle EE RDBMS "Advanced Security" option (separate license) can also be configured to encrypt Oracle Net8 traffic.

   - WebLogic uses an NMS instance specific port to communicate to an HTTP server on the NMS services node to pull in NMS map files to serve to NMS clients (lighttpd is typically used as the HTTP server).
   - One WebLogic managed server (or WebLogic cluster) instance is typically used to support NMS Java clients. If necessary it is possible to configure multiple WebLogic managed servers (on different nodes - typically behind some type of load balancing switch) to support a larger number or different categories of NMS Java clients.
     - If you have distinct categories of NMS end users different WebLogic Managed Servers can also be configured to support those specific categories of users.
     - For example, you may have one set of NMS end users that are allowed to send outbound SCADA control (open/close) SCADA requests. A specific WebLogic server (or WebLogic cluster) can be configured to

specifically allow outbound controls. Other (less privileged) NMS end users can be forced to login via an alternate WebLogic server (or WebLogic cluster) that does not allow outbound controls.

- This type of configuration typically requires some form of network segmentation. This is to ensure that ONLY qualified/privileged/ appropriate NMS end users with physical access to a given control room (for example) – are allowed to initiate outbound controls. This also requires very NMS specific corba-gw and WebLogic configuration – to ensure that non-privileged NMS users cannot initiate outbound controls.
- NMS end users can also use this scheme for other reasons. One WebLogic server to support primary control center users and another to support “non-control-center” users – for example. This can be a mitigation strategy to provide more consistent (dedicated) resources for core (control room) NMS end user access. It also provides an additional access option for at least some NMS end-users. Even if one WebLogic server is not available (being maintained or reset), at least some NMS end users can continue to access NMS via the other – for example.

- NMS clients are authenticated and authorized via WebLogic. WebLogic typically maintains a connection to an enterprise (or operations specific) Lightweight Directory Access Protocol (LDAP) accessible set of Directory Services - for NMS end user authentication. LDAPv3 traffic should be configured to be encrypted.
- A typical production NMS installation serves a broad spectrum of utility users - often too many to put them all behind an internal firewall. For a more secure NMS implementation it is recommended that WebLogic managed servers (along with the RDBMS and NMS services that support their NMS Java clients) be separated from their NMS Java clients by an additional internal firewall. The firewall ensures the WebLogic Managed Servers supporting NMS Java clients are minimally accessible outside of the internal firewall. The only port that is required to be opened is the port configured for SSL access in WebLogic.
- A separate (additional) WebLogic Managed Server instance (or cluster) can also be used to communicate to other (external) systems. For external integration, NMS specific EJBs generally communicate to external systems via a Web Service layer on top of the NMS EJBs.
  - This additional WebLogic managed server generally runs on the same node and WebLogic domain but on a different port than the WebLogic managed server that supports NMS Java clients. Running the WebLogic Managed Server used for external NMS integration on a separate port also hides this port from the general user population (if the recommended firewall between NMS end users and WebLogic is in place).
  - Generally only one WebLogic managed server at a time can be used to integrate to a given external system - generally via some type of Web Service interface. The default Oracle NMS WebLogic managed server (supporting a MultiSpeak-based integration, for example) can be configured to utilize a WebLogic cluster.

3. **Common Deployment Options:**

    - WebLogic is typically deployed on hardware vendor supported failover (active/passive) Unix cluster infrastructure. Using a Unix cluster is not necessary but provides protection against local hardware failure as well as a mechanism to support "rolling" infrastructure upgrades (upgrade one node of the Unix cluster at a time - within reason). Keep in mind that Unix clusters and WebLogic clusters are independent and have no direct dependency on each other.
        - Unix clusters are also generally convenient if/when a WebLogic managed server is configured to integrate to other (project specific) EJB servers. Since only one WebLogic managed server at a time can be configured to integrate to external EJB servers - a Unix failover cluster provides a useful platform to host the "active" WebLogic managed server (floating IP addresses managed by the cluster - for example).
            - If multiple concurrent WebLogic Servers are desired to support integration with an external system (typically a MultiSpeak integration), then the WebLogic Servers must be configured in a WebLogic cluster since the adapter uses an EJB singleton service to support external integration. A WebLogic singleton service requires either a single WebLogic instance or a WebLogic cluster.
        - If the Oracle NMS Web Switching module is in play along with multiple concurrent WebLogic Managed Servers (for whatever reason), WebLogic Clustering must be used. This is because Oracle NMS also requires an EJB Singleton to support the Web Switching module.
    - The WebLogic managed server(s) used to support NMS clients can be hosted on one or more (non-Unix-clustered) nodes. This is true even if/when WebLogic Enterprise Edition (or SOA Suite) is used in a WebLogic clustered configuration. WebLogic clusters do not have a dependency on hardware clusters - they do not need to share a common set of configuration or data files.
    - If two or more WebLogic managed servers are used to support NMS clients some form of load balancer is generally required to route incoming traffic to the available WebLogic managed servers.
    - In dual-environment configuration (for reduced downtime for NMS patching), each environment will use a different WebLogic managed server.

Tier 4 - Oracle NMS Clients - Presentation Tier

1. **Description/Purpose:**

    - NMS clients connect to the NMS WebLogic managed server and provide end users NMS electrical network model access and update options.
    - The NMS Java clients support a very wide range of user interface configuration options to accommodate project specific business practices. Configuration options include Java files, RDBMS tables, NMS specific XML (jbot) files and property files.
    - NMS Flex (browser) clients are a lighter weight NMS end user access option. They offer a subset of the functionality and configuration options offered by the NMS Java client.

2. **Common Inter-Tier Communication and Security Considerations:**

    - NMS clients only communicate to the NMS WebLogic Managed Server. Any subsequent communication to the RDBMS or NMS services is managed within WebLogic.
    - The RDBMS access that an NMS client can configure is accomplished within WebLogic using a dedicated JDBC connection pool using a specific read-only RDBMS user/schema name. The read-only RDBMS user/schema has project defined private synonyms that reference a configurable subset of the primary NMS RDBMS schema.
    - NMS Java clients receive updates from WebLogic by periodically polling the WebLogic managed server. The NMS Java clients use Remote Method Invocation (RMI) calls to facilitate this communication. The RMI calls between NMS Java clients and WebLogic should be configured as encrypted for a production NMS deployment.
    - NMS optional Operations Mobile Adapter (OMA) or Flex clients also receive updates from WebLogic by periodically polling a respective WebLogic Managed Server. Both NMS OMA and Flex clients make periodic RESTful calls to WebLogic to facility requests and/or updates.
    - NMS Flex clients utilize very similar infrastructure to NMS OMA clients - including a dedicated WebLogic Flex Gateway WLFG. The WLFG can utilize WebLogic clustering. The WLFG can be deployed within the same domain as the "internal" WebLogic Managed Server that supports it or on a separate WebLogic domain (typically outside some internal firewall). If the WLFG is deployed on a separate WebLogic domain then domain trust must be configured between the WLFG domain and the internal WL Managed Server domain that supports it. The use of domain trust in this manner is entirely similar to OMA.
    - NMS Java clients are authenticated via WebLogic. Typically, WebLogic uses a Secure LDAP connection to some form of Enterprise Directory Server (like Active Directory) for authentication, but any authentication mechanism supported by WebLogic can be used.
    - NMS Java clients can also be configured to support two-factor authentication. Two-factor authentication requires project specific configuration.
    - It is recommended that a firewall be placed in front of the WebLogic server that provides access for NMS Java, OMA, or Flex clients. Only a single port

needs to be kept open in the firewall to allow clients access to WebLogic. Note that this firewall should also block all direct end-user access to the RDBMS and NMS services tiers that NMS WebLogic instance leverages.

- For a production NMS instance only HTTPS access for the NMS clients should be allowed. Note that HTTPS equates to the t3s protocol on WebLogic.

3. **Common Deployment Options:**

- NMS Java clients are typically accessed via Java Web Start. This scheme requires a web page (hosted by the NMS WebLogic managed server) that lists project configured NMS end user environments. Each link references a Java Network Launch Protocol (JNLP) file. Each JNLP file contains commands to download, cache and execute the appropriate Java application. Each time Java Web Start initiates a Java client application it automatically validates that the most current version of the Java client application is being used - making this a convenient option for administration. To avoid man-in-the-middle attacks customers are asked to use JNLP over an HTTPS connection to WebLogic when they download the NMS Java client.
- If a project wants more control over which workstations have NMS access the NMS Configuration Assistant application can be used to generate project specific NMS Java client installers. It is then up to local NMS administrators to distribute and manage these installers for the appropriate NMS end user workstations. This includes ensuring the appropriate version of the various NMS Java clients are installed. Changes to the project configuration will automatically be picked up the next time a client starts (NMS Java client configuration is handled separately from client versions). However, if there is a new product version of NMS installed, new installers will need to be generated, distributed, and installed.
- If NMS Java clients need to run over high-latency, low-bandwidth or even an external connection (typically via a WAN or VPN) NMS Java clients can also be deployed using Windows Remote Desktop Services (or similar) technology.
  - This type of configuration allows the CPU, memory and network resources needed to run NMS Java clients to be more centralized - and possibly more performant.
  - This type of configuration also provides a more secure deployment option as no NMS Java client software executes on the end-user hardware. Only the technology specific remote viewer client executes on the NMS end user workstation - generally over an encrypted protocol. For this type of deployment no NMS specific software need ever be installed on the end user workstation.

Tier 5 - Oracle NMS Mobile Clients - Mobile Presentation Tier

1. **Description/Purpose:**
    - The Oracle NMS Operations Mobile Application (OMA) is an (optional) NMS product that provides mobile (tablet) access to the real-time NMS electrical network model. OMA includes a Javacript based template that a given NMS customer can update/configure to match business processes. The Oracle NMS provided OMA template is also updated on a regular basis in an effort to minimize any project specific effort required to deploy and maintain for a given customer.
    - The Oracle NMS OMA application has access to an (expanding) subset of NMS model data and functionality. OMA access typically includes the ability to integrate with the NMS Switching application to allow control room operators to more reliably coordinate field activity. This coordination allows an OMA user to update at least certain aspects of the NMS model directly (confirm a field switching operation as part of an existing NMS switch plan, for example). OMA can also be configured to directly manipulate the NMS model (open/close fuses, for example).
2. **Common Inter-Tier Communication and Security Considerations:**
    - OMA often uses standard cellular data technology to route mobile requests/ responses via the Internet through an Oracle NMS WebLogic "Mobile Gateway".
    - Since OMA users are typically "in-the-wild" (not on any kind of internal network behind a corporate firewall) special care must be taken to ensure only properly authorized OMA users get meaningful access to the NMS Mobile Gateway. OMA users communicate to the NMS Mobile Gateway using an encrypted RESTful protocol (over HTTPS).
    - Once OMA client requests have landed in the NMS WebLogic Mobile Gateway they are placed on a JMS request queue. A (separate) properly configured Oracle NMS WebLogic Managed Server will monitor the JMS request queue and download validated OMA requests. If the request is valid it will be processed and a response will be placed on a matching JMS response queue back on the WebLogic Mobile Gateway and back to the OMA client.
    - Both the OMA request and response JMS queues are managed on the NMS WebLogic Mobile Gateway and are always accessed from the (separate) internal NMS WebLogic Managed Server. This scheme eliminates the need to open up ports from the outside in. The only ports that are open are from the inside out. This is typically a more secure deployment model and also easier to shutdown if/when necessary.
        - To shutdown OMA entirely an NMS admin can simply stop the internal WebLogic server from accessing the external JMS queues in the NMS WebLogic Mobile Gateway. No firewall changes (or network admin privileges) are required.
3. **Common Deployment Options:**
    - The outward facing Oracle NMS WebLogic Mobile Gateway should be deployed on a separate (physical) server in a separate security zone from the (internal) Oracle NMS WebLogic Managed Server - with at least one firewall between them.

- The firewall in front of the internal NMS WebLogic Managed Server should have NO ports open from the outside in - only from the inside out.
- It is recommended to have some form of reverse proxy server and firewall between the NMS WebLogic Mobile Gateway and the OMA end users coming in from the Internet - to further isolate the WL Mobile Gateway.

Additional Security Options for Key Technology Components

Oracle Relational Database Management System

The vast majority of Oracle Network Management System static and dynamic model data is stored in an Oracle Relational Database Management System (RDBMS). All updates to a single NMS RDBMS instance are managed through at least one pair of Oracle RDBMS username/schema. This Oracle RDBMS user name is used by NMS C++ services, WebLogic JDBC connections, the Oracle SQL Plus utility along with Perl and Python scripts to update/maintain the NMS instance. Actual NMS end users are generally authenticated outside of the Oracle RDBMS (via LDAP accessible Directory Services). For security purposes - it is recommended that access to the production NMS RDBMS be managed appropriately.

For a typical installation NMS updates the RDBMS using the following access mechanisms:

- **Required:** NMS C++ Service daemons - via Oracle Call Interface.
- **Required:** WebLogic - via Java Database Connectivity (JDBC) connections.
- **Required:** Oracle SQL Plus - used for Oracle NMS schema creation/ configuration updates.
- **Required:** Perl (from the Oracle RDBMS installation) - used by NMS scripts to update/access RDBMS.
- **Required:** Python - used by NMS scripts to update/access RDBMS.
- **Optional:** BI Publisher via JDBC - used by Oracle NMS Switching to print switch plans.
- **Optional:** Project specific Oracle RDBMS query/update/monitor tools. SQL access/update tools like Oracle SQL Developer, OEM Grid/Cloud Control and/or similar may also be used to help monitor the NMS RDBMS but are not required to install or run NMS.

The NMS C++ Service daemons manage the vast majority of updates to the NMS RDBMS schema. NMS services perform RDBMS updates via calls to the various NMS *DBService daemon processes. The *DBService daemon processes use the Oracle Call Interface (OCI) to update/query the Oracle RDBMS. The Oracle Call Interface is an Application Programming Interface (API) that uses the Oracle Net8 (SQL*Net V2) protocol to communicate to the Oracle RDBMS "listener" process. The Oracle listener runs on the same node as the Oracle RDBMS and allows remote network nodes to access the Oracle RDBMS.

General RDBMS Security Options:

1. **Ports:** The Oracle RDBMS listener process uses port 1521 by default for Net8 (SQL*Net V2) communication. The Oracle RDBMS listener port can be configured via the $ORACLE_HOME/network/admin/listener.ora configuration file. Any change to the Oracle RDBMS listener port must be matched by the relevant $TNS_ADMIN/tnsnames.ora configuration file - which is used for Oracle RDBMS client access by Oracle NMS services.
2. **Standard:** Use independent Oracle RDBMS instances to manage production NMS data versus non-production NMS data.
    - Depending on RDBMS capability/capacity a single RDBMS instance can often support multiple production (or non-production) Oracle NMS instances concurrently. It is recommended that each NMS instance (Oracle RDBMS schema/user for a given NMS instance) have a dedicated Oracle RDBMS tablespace to hold NMS instance specific data - but is not required.
3. **Standard:** Secure NMS Oracle username credentials.
    - Use Oracle Wallet to secure credentials for NMS daemon processes.
        - NMS *DBService processes are "daemon" components that generally operate 24x7. As such these processes must be able to stop/start without human intervention. NMS uses Oracle Wallet technology to store and better secure user credentials needed to access/update the Oracle NMS RDBMS. This allows the NMS Unix admin account to stop/start the NMS *DBService processes without direct access to actual Oracle RDBMS schema/username credentials.
        - The NMS Unix admin account uses the Oracle SQL Plus utility to help create and manage the Oracle NMS RDBMS schema. The Oracle SQL Plus utility uses the same Oracle Wallet as NMS services to more securely access the Oracle NMS RDBMS.
    - The NMS nms-env-config script is generally used to setup the Oracle Wallet to hold Oracle NMS RDBMS username and password credentials for a given Oracle NMS admin (Unix) user account. In this manner access to the NMS admin Unix user account provides access to the Oracle NMS RDBMS username. These credentials are used to allow NMS OCI based processes to access the appropriate Oracle RDBMS instance and schema.
        - NMS *DBService processes, Perl, Python, and SQL Plus scripts that run under the NMS admin account can all leverage the same Oracle wallet username/password credentials for access to the Oracle NMS RDBMS user/schema.
4. **Standard:** Minimize the Oracle RDBMS privileges granted to the Oracle username used to support the Oracle RDBMS instance (table read/write privilege but not schema change privileges). An Oracle NMS instance requires at least two Oracle RDBMS schemas.
    - The Oracle NMS RDBMS admin schema/user has Data Definition Language (DDL) and Data Control Language (DCL) access and can update the schema as required. The Oracle NMS RDBMS admin user owns the schema (all NMS specific tables and views). It is used for initial NMS installs and for applying routine NMS patches (so NMS schema can be modified).

- The Oracle NMS RDBMS operational schema/user contains only synonyms with restricted Data Manipulation Language (DML) access to the Oracle RDBMS admin schema - no tables or views. This (restricted) Oracle RDBMS schema/user is for normal/daily NMS operations.
  - Note in a dual NMS environment there will be two separate Oracle NMS RDBMS user/schemas for production use. Support for dual NMS environments is an NMS 2.5.0.0 option that supports two separate (blue/green type) NMS admin accounts to support reduced downtime for incremental NMS patching.
  - See the $NMS_BASE/templates/nms_role.sql.template for example Oracle Data Control Language (DCL) statements to create the required Oracle RDBMS roles and privileges necessary to support an NMS installation.
  - See the $NMS_BASE/templates/nms.sql.template for example Oracle Data Definition Language (DDL) statements to create required Oracle RDBMS tablespaces and user names - based on the roles and privileges defined in the nms_role.sql.template file.

5. **Relaxed:** For a test/train RDBMS environment you might grant the ability for a given Oracle user to import/export RDBMS data. With Oracle 11gR2 or higher, you must use the Oracle datapump utility to export/import RDBMS data. The older imp/exp mechanism is not adequate and will not work. Example statements to grant an Oracle user named nms_admin datapump export/import privileges (from the nms.sql.template file noted above).
   - grant DATAPUMP_EXP_FULL_DATABASE TO nms_admin;
   - grant DATAPUMP_IMP_FULL_DATABASE TO nms_admin;
6. **Enhanced:** If a given NMS customer/installation wants or needs to encrypt Oracle NMS RDBMS data "on-the-wire" the Oracle EE RDBMS Native Network Encryption (NNE) option can be configured. Some high-level information on how to setup NNE for NMS is noted below. Reference Oracle RDBMS documentation on Native Network Encryption for more details.

   Native Network Encryption(NNE)

   The NNE option is built into the database itself and does not require SSL or a separate listener port and has relatively low overhead. The data is encrypted, but transported over regular tcp instead of SSL.

   In the NMS RDBMS server-side sqlnet.ora file, set the options below:

```
# SQLNET.ENCRYPTION_SERVER =
required|requested|accepted|rejected
SQLNET.ENCRYPTION_SERVER = requested
SQLNET.ENCRYPTION_TYPES_SERVER = (AES256)
```

   In the NMS services RDBMS client side sqlnet.ora file, set the options below:

```
# SQLNET.ENCRYPTION_CLIENT =
required|requested|accepted|rejected
SQLNET.ENCRYPTION_CLIENT = required
```

   You can verify connections are encrypted via the Oracle NMS ISQL utility (a wrapper around sqlplus) as follows:

```
$ ISQL
```

```
   SQL> select network_service_banner from v$session_connect_info
   where sid in (select distinct sid from v$mystat);

   NETWORK_SERVICE_BANNER
   ------------------------------------------------------------
   TCP/IP NT Protocol Adapter for Linux: Version 19.0.0.0.0 -
   Production
   Encryption service for Linux: Version 19.0.0.0.0 - Production
   AES256 Encryption service adapter for Linux: Version 19.0.0.0.0
   - Production
Crypto-checksumming service for Linux: Version 19.0.0.0.0 -
Production
```

In WebLogic data source, in the properties box:

```
   oracle.net.encryption_client=REQUIRED
oracle.net.encryption_types_client=(AES256)
```

7. **Enhanced:** If a given NMS customer/installation wants or needs more security for Oracle RDBMS "data-at-rest" encryption they can consider the Oracle "Advanced Security" option to the Oracle Enterprise Edition RDBMS. The Oracle Advanced Security option allows virtually all RDBMS data to be encrypted both on-disk and/or on-the-wire as it moves between the required technology components - such as between the Oracle RDBMS and WebLogic.
   - The Oracle "Advanced Security" option is a separately licensed Oracle EE RDBMS option. Configuration for the Oracle "Advanced Security" option should be generally transparent to the Oracle Network Management System application. Configuration for Oracle Advanced Security is not covered in this document.
8. **Enhanced:** Run the production Oracle NMS RDBMS instance on a dedicated server - not on the same server as might be used for a non-production NMS (or other) RDBMS dependent applications. This approach reduces potential attack vectors for the critical RDBMS component.
9. **Relaxed:** Support more than one NMS instance on a single RDBMS instance.
   - Note it is generally acceptable (and common) to host multiple NMS instances on a single RDBMS instance - for production and non-production. It is generally recommended to keep all production NMS instances on separate RDBMS instances from non-production NMS-instances.
   - Each NMS instance must have its own set of dedicated Oracle RDBMS schema/user names. For easier management it is recommended that each NMS instance RDBMS user/schema have a dedicated RDBMS tablespace.

Network Management System Services

Oracle NMS services are a set of C++ daemon processes that manage the majority of the processing and coordination required to manage the operational NMS Electrical Network model. These NMS C++ service processes coordinate with each other using the ISIS message bus.

ISIS runs on the node where NMS services run. By default, ISIS runs on the loopback (localhost) network. In this mode ISIS is only used to coordinate messages between NMS service processes and no ISIS ports are accessible from any external nodes. NMS ISIS messages are not encrypted. ISIS uses the configuration file pointed to by the $ISIS_PARAMETERS environment variable. In general, $ISIS_PARAMETERS should not be changed unless specifically instructed by Oracle NMS support.

The NMS SwService (Switching Service) uses Simple Object Access Protocol (SOAP) over HTTPS to send synchronous request messages to WebLogic that require a distinct response. These messages are authenticated and can also be encrypted via SSL. Oracle Wallet functionality is used to hold authentication credentials for the NMS services that send these SOAP messages to WebLogic. In order to use a secure connection, SwService needs to connect to the SSL port of the WebLogic server and provide an SSL certificate file, which is used to confirm identity of the WebLogic server. See **Creating the Certificate for SwService** on page 1-28 for information on creating an appropriate certificate file to use with the commands below.

SwService Command-line Options for Secure Connection to WebLogic Server

```
-outgoingURL https://
${NMS_APPSERVER_HOST}:${NMS_APPSERVER_PORT}${NMS_SWSERVICE_EJB_STRING}
/ExternalSwmanServiceImpl -sslCert ${NMS_SSL_CERT}
```

Oracle WebLogic Server

Oracle WebLogic provides NMS with a Java Enterprise Edition (EE) Application Server and general purpose integration platform. Oracle NMS utilizes a collection of Enterprise Java Beans (EJBs) that execute within WebLogic to:

- Authenticate and authorize NMS end users and adapters.
- Efficiently cache and provide updates for NMS Java end user client applications.
- Integrate to internal NMS applications (Switching)
- Integrate to external systems (AMI, SCADA, Mobile, etc).

A given WebLogic Managed Server communicates via a Java Database Connectivity (JDBC) pool to the Oracle listener process on the Oracle RDBMS host. WebLogic must be configured to use the JDBC Thin Drivers (not the OCI based JDBC drivers) for connection to a non-RAC RDBMS and Active GridLink drivers for a RAC RDBMS. WebLogic manages Oracle RDBMS user name and password credentials inside WebLogic. The user and password credentials must be specified when you configure the JDBC Data Source for given WebLogic Managed Server instance.

WebLogic can be configured to open up both a standard and secure port to support in-bound communication for each WebLogic Managed Server. The specific ports are configured within WebLogic. The WebLogic standard port supports HTTP, SOAP, EJB RMI calls and anything else that uses JNDI. In general WebLogic supports:

Standard: T3, HTTP, SOAP over HTTP, RMI-IIOP.

Secure: T3S, HTTPS, SOAP over HTTPS, RMI-IIOP over HTTPS

For a more secure implementation WebLogic should be placed behind an additional internal firewall (in addition to the corporate firewall - see Figure 1). In this case the only port that would need to be opened to NMS end users is the SSL (secure) port, and optionally the standard port. Specifically, the database and the NMS services including the corba gateway and publisher do not need to be directly accessible to the NMS Java client.

Note it is highly recommended that NMS Java clients be configured to run on the latest supported Java Runtime Environment (JRE). To help ensure the latest JRE versions is being used older JREs should also normally be uninstalled. In a similar fashion it is also generally recommended that the latest Java Development Kit (JDK) - which includes a JRE - be installed on the WebLogic host. Note WebLogic must also be properly configured to ensure the WebLogic JVM is actually using the latest JDK.

Common Object Request Broker Architecture - Object Request Broker

CORBA is used to allow back end NMS C++ services to support front end NMS clients. The NMS corbagateway (service process) speaks ISIS on the NMS service side and CORBA IIOP on the WebLogic side. The NMS corbagateway uses "The Ace Orb" (TAO) ORB and WebLogic uses the WebLogic ORB to coordinate communication via the TAO Naming Service (tao_cosnaming) process.

The port and host that WebLogic uses to find the tao_cosnaming process are defined in the NMS RDBMS, in the nms_parameters_view table column where nms_parameters_view.attrib='WEB_corbaInitRef'.

For example:

```
nms_parameters_view.attrib='NameService=corbaloc:iiop:1:2@myhost:17820
/NameService'
```

The port and host that the NMS corbagateway process uses to find the tao_cosnaming (ORB) process is defined in the project specific NMS system.dat configuration file. A typical entry for a given corbagateway process might start something like the following:

```
program corbagateway corbagateway -ORBInitRef NameService=iioploc://
myhost:17820/NameService
```

The port used to communicate to the naming service can be modified but must be coordinated. CORBA message traffic between NMS C++ services and WebLogic is not encrypted.

HTTP Server

The HTTP server, lighttpd, is a lightweight third party HTTP server shipped with Oracle NMS. It is used by Oracle NMS to serve NMS map files to WebLogic. The lighttpd daemon is auto-configured and started using the nms-lighttpd script:

```
   Usage: nms-lighttpd [start [port] [host] | stop | status]
   host - host the httpd server will serve files on.
   If not specified, will use value from
   nms_parameters_view where attrib=WEB_mapHttpdHost
   Default if not specified or in RDBMS: 0.0.0.0
   port - port the httpd server will serve files on.
   If not specified, will use value from
   nms_parameters_view where attrib=WEB_mapHttpdPort
   Default if not specified or in RDBMS: 8888
   start - will start the httpd server on the given port
   stop - will stop the httpd server
   status - will echo 0 if not running, 1 if running
   exit value is 0 if running, 1 if not running.

   Notes: nms_parameters_view attib=WEB_mapHttpdAllowedIPs
restricts access by IP
```

The lighttpd HTTP server is configured and monitored by SMService by default in the template system.dat files.

The HTTP server is generally configured to serve a single directory of unencrypted NMS model map files to specific IP addresses (where the WebLogic instances that support this NMS instance run). The HTTP server is typically not configured to serve encrypted NMS map files to WebLogic, however HTTPS is supported if desired. More information on configuration options for the nms-lighttpd script can also be found under [project]_parameters.sql in the Oracle Utilities Network Management System Installation Guide.

The lighttpd HTTP server is also used to serve up log files to the Oracle Utilities Network Management System Application Management Pack (AMP) for Oracle Enterprise Manager (OEM). The server logs can be downloaded from the NMS AMP plug-in and/or viewed without having to log into the server where the NMS services are running.

The nms-lighttpd-oem script is used to start this auto-configured instance of lighttpd.

```
   Usage: nms-lighttpd-oem [start [port] [host] | stop | status]
   host - host the httpd server will serve files on.
   If not specified, will use value from
   nms_parameters_view where attrib=WEB_oemHttpdHost
   Default if not specified or in RDBMS: 0.0.0.0
   port - port the httpd server will serve files on.
   If not specified, will use value from
   nms_parameters_view where attrib=WEB_oemHttpdPort
   Default if not specified or in RDBMS: 8888
   start - will start the httpd server on the given port
   stop - will stop the httpd server
   status - will echo 0 if not running, 1 if running
   exit value is 0 if running, 1 if not running.

Notes: nms_parameters_view attib=WEB_oemHttpdAllowedIPs restricts
access by IP
```

This nms-lighttpd-oem instance has the same monitoring and configurable options as what was described for the nms-lighttpd instance.

LDAP Server

Oracle NMS generally uses LDAP accessible enterprise Directory Services to authenticate NMS end users. This is accomplished by routing authentication requests from NMS Java clients through WebLogic - which must in turn be configured to connect to the appropriate LDAP accessible Directory Service. In practice (especially production) LDAPS (LDAP over TLS) should be used to encrypt LDAP traffic. See the *Oracle Utilities Network Management System Installation Guide* for details on configuring LDAP within WebLogic.

Network Management System Java Client Applications

Oracle NMS uses Java (Swing) clients to allow NMS end users access to the operational NMS Electrical Network data model. Once initiated these clients are in near-constant communication with WebLogic in an effort to remain synchronized with the real-time operational model.

The NMS Java clients generally use two forms of digital signing certificates:

1. **SSL certificate:** This is used to authenticate the WebLogic server to the client, as well as to encrypt communication to and from WebLogic.
2. **Signing certificate:** This is used to sign the NMS Java client jar files so that the client machine recognizes the application is from a trusted source and will allow the application to run on the client.

For both of these certificates, Oracle recommends using a certificate generated by a trusted Certificate Authority (CA). However for testing purposes, self-signed certificates are supported. Please see **Security Certificates in Oracle Utilities Network Management System** on page 1-25 for details on using security certificates.

Security Certificates in Oracle Utilities Network Management System

Oracle Utilities Network Management System ships with a dummy SSL and signing certificates as part of the OPAL NMS demonstration model. However, for production use these certificates should be replaced by third party certificates. It is also possible to use an internal certificate authority if one is available.

SSL Certificate

Oracle NMS uses an SSL certificate to secure network traffic between the Java client and the WebLogic server. To create this certificate, run the following commands from the NMS server:

```
cd $NMS_CONFIG/jconfig
keytool -genkeypair -alias nms-key -keyalg RSA -keystore nms-
ssl.keystore -validity 365
```

Choose a keystore password.

Then fill out the information about your server.

For "first and last name" choose the hostname of the server you wish to deploy. It should match the url that the end user will be using. For example, if the user accesses the site by http://nms.company.com:7010/nms, you would choose "nms.company.com". It is not critical what you put in the other fields. If a field doesn't apply (such as organizational unit) it may be left blank. State should be the full name and not an abbreviation.

When you are prompted to enter the key password, press **Enter**.

This certificate can be used as-is, as a self-signed certificate. That means this certificate can be used to help encrypt traffic but for the browser to trust the certificate it will be necessary for the end user to accept the certificate before continuing. Note that self-signed certificates generally need to be accepted each time the application is launched.

To avoid the issue of having to accept self signed certificates each time an application is launched Oracle recommends using a third party certificate. To request a third party certificate, follow these steps:

```
cd $NMS_CONFIG/jconfig
keytool -certreq -alias nms-key -keyalg RSA -keystore nms-
ssl.keystore -file nms-ssl.csr
```

This creates the nms-ssl.csr which should be sent to the Certificate Authority (CA) you are using. They will respond back with the certificate

To import the certificate do the following (replacing nms-ssl.pem with the name of the certificate you received):

```
cd $NMS_CONFIG/jconfig
keytool -importcert -keystore nms-ssl.keystore -alias nms-key -file
nms-ssl.pem
```

After importing the certificates into nms-ssl.keystore, you may need to copy the file to your WebLogic server. (See also the "Configure the Identity and Trust Keystore" section of the *Oracle Utilities Network Management System Installation Guide*.)

Defining Encryption Parameters

NMS will use the most secure cipher suite that is available to both the client and server. If there are specific requirements that certain ciphers be allowed or disabled, it may be done by adding the appropriate configuration options to the WebLogic managed server. See https://docs.oracle.com/javase/8/docs/technotes/guides/security/jsse/JSSERefGuide.html#DisabledAlgorithms for more information.

Validating Encryption Suite

The encryption used to secure the application is negotiated by the server and client. If you wish to see which cipher suite was chosen, add this parameter to either the client or server: -javax.net.debug=ssl:handshake

Trusting Certificates

If it is desired to automatically start the application without the validation screen when it is first installed, it is possible to configure a deployment rule to indicate that a certain application or location should always be considered trusted and run without further security prompts.

For more details, see:

https://docs.oracle.com/javase/8/docs/technotes/guides/deploy/deployment_rules.html

Signing Certificates

Signing certificates are used to help generate digital signatures. Digital signatures can in turn be used to validate that an application is legitimate and can be trusted for download.

Because of the security restrictions that Oracle places on Java Web Start applications, it is recommended that applications be signed with a third party certificate. Future versions of Java will not permit the running of self-signed applications by default.

The steps to acquire a 3rd party certificate are similar to creating an SSL certificate:

```
   cd $NMS_CONFIG/jconfig
keytool -genkeypair -alias nms-key -keyalg RSA -keystore nms-
signing.keystore -validity 365
```

For the "first and last name", enter the name of the application: Oracle NMS

The remainder is filled out the same as for SSL certificates (above0)

Next, request a 3rd party certificate:

```
   cd $NMS_CONFIG/jconfig
keytool -certreq -alias nms-key -keyalg RSA -keystore nms-
signing.keystore -file nms-signing.csr
```

Send the nms-signing.csr to the desired third party Certificate Authority (CA), and they will respond back with the certificate.

Next, import the certificate (replacing nms-signing.pem with the name of the certificate you received)

```
   cd $NMS_CONFIG/jconfig
```

```
keytool -importcert -keystore nms-signing.keystore -alias nms-key -
file nms-signing.pem
```

Your CA might also provide an intermediate certificate along with your SSL (server) certificate. You must import that certificate before you import the SSL certificate.

To import the certificate do the following (replacing IntermediateCA.cer with the name of the certificate you received):

```
keytool -importcert -keystore nms-ssl.keystore -alias intermediateCA -
file IntermediateCA.cer
```

After importing the certificates into nms-ssl.keystore, you may need to copy the file to your WebLogic server. (See also the "Configure the Identity and Trust Keystore" section of the *Oracle Utilities Network Management System Installation Guide*.)

Next run this command to update build.properties with an obfuscated copy of the pass phrase:

```
nms-keystore-password $NMS_CONFIG/jconfig/build.properties
key.server.pass
```

Using a Signing Keystore From Another Source

If you receive a Java keystore to use (instead of using the standard process for requesting and processing a signing keystore), follow these steps:

Rename the keystore to `nms-signing.keystore`, if necessary.

Change that alias of the signing entry to `nms-key`. You can determine the names of the entries by using:

```
keytool -keystore nms-signing.keystore -list
```

To change an alias name, use:

```
keytool -changealias <old alias> -destalias nms-key -keystore nms-
signing.keystore
```

The alias password needs to match the keystore password. If it doesn't, do the following:

```
keytool -keypasswd -keystore nms-signing.keystore -alias nms-key
```

Creating the Client Keystore

After creating the server certificate, regardless if it is a 3rd party or self-signed, the next step is to create the client keystore. It is recommended to use a different password than was used for the server keystore:

```
cd $NMS_CONFIG/jconfig
keytool -export -keystore nms-ssl.keystore -alias nms-key -file
nms_public.pem
keytool -importcert -keystore global/nms-client.keystore -alias
nms-key -file nms_public.pem
```

Creating the Certificate for SwService

SwService needs a certificate in order to communicate with WebLogic Server securely. This certificate can be created using the following command:

```
keytool -exportcert -keystore $NMS_CONFIG/jconfig/nms-ssl.keystore -
alias nms-key | openssl x509 -inform der -out $NMS_SSL_CERT
```

Open Ports

The following table lists the default ports that might be open for a typical NMS installation - by supporting node. See Figure 1 above for how typical NMS technology components tie together and communicate - including default ports.

Node	**Port**	**Protocol**	**Service**	**Open by Default?**	**Configurable**
RDBMS	1521	TCP	Net8	Yes	Yes
NMS Services	17820	TCP	IIOP	Yes	Yes
NMS Services	8888	TCP	HTTP	Yes	Yes
NMS Services	22	TCP	SSH	Yes	Yes
WebLogic	7001	TCP	T3	Yes	Yes

Client/Server Security

NMS works with a rich client that allows the best user experience. That rich client in turn calls various APIs on the server to accomplish the user request. JBot configuration will allow various different applications or user types to configure different options for different users. However, that configuration does not address securing the underlying security of the API calls. A technically savvy user could bypass the configuration by changing the tool configuration on their own copy of NMS or by using a debugger to modify the existing Java runtime. Therefore, it is important to use WebLogic roles and groups to limit the security exposure; this is used to protect the actual API calls made to the server.

It is strongly recommended to use different groups for each type of user, as using the same group will allow a user to do API calls for any of the groups. For example, if a standard user is in the service_users group, (or has access to the NMSService role, that would give them the ability to make a model change under someone else's name). Likewise, if a view only user has standard access, then they would be able to call any of the APIs that a standard user has access to. If an application is not used, such as Call Entry or Service Alert, then the group does not have to be assigned.

Please see the **Configuring NMS Security Roles** section in the *Oracle Utilities Network Management System Installation Guide* for information on configuring security roles and groups.

Another option, especially for an occasional user is to use a remote desktop technology such as Oracle Secure Global Desktop or Citrix. Users that only have access to NMS through Citrix only can access the options that are configured for their environment, and would not have any direct API access.

Attachment Security

NMS supports performing various validations on attachments (such as for switch plans and damage assessments). The validations are controlled by the following setting in CentricityServer.properites:

```
   # This is the validator that should be used to validate
   attachments. Multiple validators can be configured.
   # Custom validators can be used to integrate with an external
   malware scanner.
   # All scanners must implement
   com.splwg.oms.common.ejbaccess.AttachmentValidator
   # To disable the whitelist, comment or remove the following line:
attachment_validator =
com.splwg.oms.server.util.ExtensionAttachmentValidator
```

The default validator will compare the attachments against a list of valid extensions. The extensions are defined in CentricityTool.properties:

```
   # This is a list of file types that are allowed to be uploaded to
   NMS
allowed_extensions = jpg, jpeg, gif, png, pdf, txt, doc, docx, xls,
xlsx, mp4, avi, odt, ott, oth, odm, ods, ots, odp, odg, otp
```

Project specific validators can also be configured, such as for virus scanning. They should implement com.splwg.oms.common.ejbaccess.AttachmentValidator.

General Data Protection Regulation

The Oracle NMS obfuscate program is a command line utility that allows an NMS administrator to obscure specific sensitive customer and/or crew member data for inactive records in the Oracle NMS RDBMS. Records that may be obfuscated include names, addresses, phone numbers, customer comments and messages.

The obfuscate utility uses the following options:

```
obfuscate -days <number of days>
   -crew <crew key>
   -customer <account number>
[-replaceString <replacement word(s)>]
```

At least one of `[-crew, -customer, -days]` must be specified.

Option	Description
-replaceString	Optional. The replacement word(s). The string should be short enough to fit in all fields. Default: Obscured
-crew	The crews.crew_key to obfuscate
-customer	The ces_customers.account_number to obscure
-days	Obscure all crew and customer records that are at least this number of days old
-debug	Print the SQL statements that will ultimately be executed.

The AuditLog application works with the scripts and applications defined above to keep a persistent record in the database of the data manipulation activities that have been going on when a customer uses any of these scripts or applications. The information is stored in the MODEL_AUDIT_LOG database table and can be useful when trying to help support a customer with corrupted data by helping to provide a better scenario of the activities that might reproduce the problem.