SYBASE®
An SAP® Company

Security

Sybase Unwired Platform 2.1

DOCUMENT ID: DC01703-01-0210-02
LAST REVISED: October 2011
Copyright © 2011 by Sybase, Inc. All rights reserved.
This publication pertains to Sybase software and to any subsequent release until otherwise indicated in new editions or technical notes. Information in this document is subject to change without notice. The software described herein is furnished under a license agreement, and it may be used or copied only in accordance with the terms of that agreement.
To order additional documents, U.S. and Canadian customers should call Customer Fulfillment at (800) 685-8225, fax (617) 229-9845.
Customers in other countries with a U.S. license agreement may contact Customer Fulfillment via the above fax number. All other international customers should contact their Sybase subsidiary or local distributor. Upgrades are provided only at regularly scheduled software release dates. No part of this publication may be reproduced, transmitted, or translated in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without the prior written permission of Sybase, Inc.
Sybase trademarks can be viewed at the Sybase trademarks page at *http://www.sybase.com/detail?id=1011207*. Sybase and the marks listed are trademarks of Sybase, Inc. ® indicates registration in the United States of America.
SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world.
Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.
Unicode and the Unicode Logo are registered trademarks of Unicode, Inc.
All other company and product names mentioned may be trademarks of the respective companies with which they are associated.
Use, duplication, or disclosure by the government is subject to the restrictions set forth in subparagraph (c)(1)(ii) of DFARS 52.227-7013 for the DOD and as set forth in FAR 52.227-19(a)-(d) for civilian agencies.
Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Contents

Documentation Roadmap for Unwired Platform

Learn more about Sybase® Unwired Platform documentation.

See *Documentation Roadmap* in *Fundamentals* for document descriptions by user role. *Fundamentals* is available on Production Documentation.

Check the Sybase Product Documentation Web site regularly for updates: access *http://sybooks.sybase.com/nav/summary.do?prod=1289*, then navigate to the most current version.

Security Task Flows

Task flows identify important security setup activities in an Unwired Platform environment. Activities performed by the Unwired Platform administrator, may also require the collaboration or participation of mobile application developers, or Afaria, security, or database administrators, depending on the role distribution of your organization.

Follow the activities documented in the first task flow in this list. The remaining task flows are optional, and depend on the mobility context into which Unwired Platform is installed.

For an introduction to the security landscape of Unwired Platform, see Introduction to Security.

- *Setting Up Security After Installation*
  Once you have installed all components of your environment you can start securing the infrastructure components and enabling encrypted communications between them.
- *Enabling Single Sign-on for DOE-C Packages*
  Enable single sign-on (SSO) over secure paths for Data Orchestration Engine Connector (DOE-C) packages.
- *Enabling Single Sign-on for Mobile Business Object Packages*
  Enable single sign-on (SSO) over secure paths for mobile business object (MBO) packages.
- *Enabling Single Sign-on for OData Applications*
  Enable single sign-on (SSO) over secure paths for OData applications.

Setting Up Security After Installation

Once you have installed all components of your environment you can start securing the infrastructure components and enabling encrypted communications between them.

Many security activities vary depending on the context in which Unwired Platform is deployed. However, common initial setup activities typically include these tasks:

1. *Securing the Data Infrastructure*
   Secure data by first protecting the infrastructure on which it resides, then securing runtime databases.
2. *Securing Data Tier Databases*
   Secure all databases installed as the Unwired Platform data tier. You can change DBA passwords, grant DBA permissions to other users, and encrypt data and logs.
3. *Enabling Authentication and RBAC for Administrator Logins*
   Administrators are authenticated both by Sybase Control Center and by Unwired Server. To make first-time installs easier, the Unwired Platform installer collects a single

password that is used to configure and enable a security provider called PreconfiguredUser login module.

4. *Securing the Server Infrastructure*

   Before you can secure the runtime, you must first secure the underlying infrastructure. This activity prevents files from being tampered with on the host, also allows the Unwired Server to run with existing security mechanisms.

5. *Enabling Authentication and RBAC for User Logins*

   Enable authentication and role-based access control (RBAC) for device user logins by creating a new security configuration (that is, one that is distinct from the "admin" security configuration on the "default" domain), and mapping roles, then assigning it.

6. *Securing Data Change Notifications*

   If you use data change notifications (DCN) to notify Unwired Server of EIS changes, secure the communication stream to avoid packet sniffing or data tampering. Use Sybase Control Center to configure the DCN stream after you have created SSL certificates.

Enabling Single Sign-on for DOE-C Packages

Enable single sign-on (SSO) over secure paths for Data Orchestration Engine Connector (DOE-C) packages.

1. *Preparing Your SAP Environment for Single Sign-on*

   Verify that the SAP® enterprise information system (EIS) is configured correctly to accept SSO connections from Unwired Server.

2. *Configuring X.509 Certificates for SAP Single Sign-on*

   Import, export, and generate the X.509 certificates that secure communication paths between Unwired Server and the SAP enterprise information system (EIS), and for client authentication, including single sign-on (SSO) with X.509 or SSO2 tokens.

3. *Deploying and Configuring DOE-C Packages*

   Deploy the ESDMA (DOE-C package) to an Unwired Server domain using the DOE-C command line utility (CLU).

4. *Creating Security Profiles to Enable Mutual Authentication for SAP*

   Create security profiles and associate them with X.509 server certificates that can be used to establish secure connections between a client, Unwired Server, and the SAP EIS.

5. *Enabling the HTTPS Port and Assigning the SUPServer Security profile*

   Enable an HTTPS port for secure communication between Unwired Server and the SAP EIS.

6. *Enabling the DOE-C Connection and Assigning the TechnicalUser Security profile*

   Configure the SAP Data Orchestration Engine Connector (DOE-C) connection pool between Unwired Server and the SAP EIS. This is the port on which Unwired Server

communicates with the DOE, including forwarding subscriptions and allowing client operations to flow through to the DOE.

7. *Security Configurations That Implement Single Sign-on Authentication*

   Use the CertificateAuthenticationLoginModule authentication module to implement X.509 authentication or HttpAuthenticationLoginModule to implement SSO2.

8. *Seeding Messaging Devices With Unwired Server*

   If you are not using Afaria, you can install the client application then connect to corporate LAN using Wifi or any other method of your choosing in order to seed devices with required files.

See also

- *SAP Single Sign-on and DOE-C Package Overview* on page 42
- *Single Sign-on Authentication* on page 39

Enabling Single Sign-on for Mobile Business Object Packages

Enable single sign-on (SSO) over secure paths for mobile business object (MBO) packages.

1. *Single Sign-on for SAP MBO Package Prerequisites*

   Before implementing SSO for SAP MBO packages, configure the MBOs so client credentials can be propagated to the EIS and, if enabling SSO for a Workflow application, add the appropriate starting point.

2. *Preparing Your SAP Environment for Single Sign-on*

   Verify that the SAP® enterprise information system (EIS) is configured correctly to accept SSO connections from Unwired Server.

3. *Installing the SAPCAR Utility*

   Unzip and install the latest SAPCAR utility on your Unwired Server or Unwired WorkSpace host, which you can use to extract the contents of compressed SAP files, for example RFC and cryptographic library files.

4. *Installing the SAP Cryptographic Libraries on Unwired Platform*

   Installation and configuration is required if you want to configure Secure Network Communications (SNC) for Unwired Platform SAP JCo connections. SNC may be required by the SAP EIS in question, if SSO2 tokens or X.509 certificates are used for connection authentication.

5. *Configuring X.509 Certificates for SAP Single Sign-on*

   Import, export, and generate the X.509 certificates that secure communication paths between Unwired Server and the SAP enterprise information system (EIS), and for client authentication, including single sign-on (SSO) with X.509 or SSO2 tokens.

6. *Creating Connections and Connection Templates*

   Create a new connection or connection template that defines the properties needed to connect to a new data source.

7. *Security Configurations That Implement Single Sign-on Authentication*

   Use the CertificateAuthenticationLoginModule authentication module to implement X.509 authentication or HttpAuthenticationLoginModule to implement SSO2.

8. *Provisioning Security Artifacts*

   Typically, you must preprovision Unwired Platform security artifacts before applications can be used. The manner by which an artifact is provisioned depends on the artifact.

9. *Single Sign-on for SAP MBO Package Postrequisites*

   After configuring SSO for SAP MBO packages on Unwired Server, install certificates on the mobile device and test them.

See also

- *SAP Single Sign-on and Mobile Business Object Package Overview* on page 44
- *Single Sign-on Authentication* on page 39

Enabling Single Sign-on for OData Applications

Enable single sign-on (SSO) over secure paths for OData applications.

1. *Preparing the SAP Gateway*

   Configure the SAP Gateway to push OData application data to Unwired Server, including configuring the RFC destination for HTTPS on the Gateway.

2. *Preparing Your SAP Environment for Single Sign-on*

   Verify that the SAP® enterprise information system (EIS) is configured correctly to accept SSO connections from Unwired Server.

3. *Using Keytool to Generate Self-Signed Certificates and Keys*

   Whenever possible, use a PKI system and a trusted CA to generate production-ready certificates and keys that encrypt communication among different Unwired Platform components. You can then use keytool to import and export certificate to the platform's keystores and truststores. Otherwise, you can also use keytool to generate self-signed certificates and keys.

4. *Configuring X.509 Certificates for SAP Single Sign-on*

   Import, export, and generate the X.509 certificates that secure communication paths between Unwired Server and the SAP enterprise information system (EIS), and for client authentication, including single sign-on (SSO) with X.509 or SSO2 tokens.

5. *Creating Security Profiles to Enable Mutual Authentication for SAP*

Create security profiles and associate them with X.509 server certificates that can be used to establish secure connections between a client, Unwired Server, and the SAP EIS.

6. *Enabling the HTTPS Port and Assigning the SUPServer Security profile*

   Enable an HTTPS port for secure communication between Unwired Server and the SAP EIS.

7. *Security Configurations That Implement Single Sign-on Authentication*

   Use the CertificateAuthenticationLoginModule authentication module to implement X.509 authentication or HttpAuthenticationLoginModule to implement SSO2.

8. *Provisioning Security Artifacts*

   Typically, you must preprovision Unwired Platform security artifacts before applications can be used. The manner by which an artifact is provisioned depends on the artifact.

See also

- *SAP Single Sign-on and Online Data Proxy Overview* on page 55
- *Single Sign-on Authentication* on page 39

Introduction to Security

Mobility has changed the computing and network environments of today. Before mobility, enterprise security primarily focused on the firewall and limited access to digital assets to only those users authenticated within the enterprise information system (EIS).

An Unwired Platform deployment introduces a multilayer approach to corporate security designed for mobility. This approach ensures that:

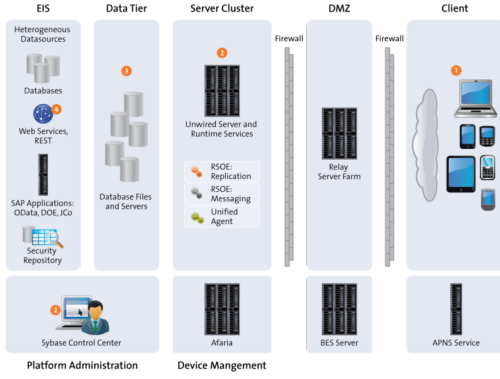
- Internal and external device users can securely connect to enterprise information systems.
- Every network link that transfers corporate information and every location that stores enterprise data guarantees confidentiality.

Before you can prepare for the scale and scope of activities required to secure Unwired Platform, learn which components you can secure, which communication streams you can protect, and how you can control access to mobile digital assets:

Component Security

Unwired Platform consists of multiple components that are installed on internal networks, primarily on the corporate LAN and the demilitarized zone (DMZ). Each component requires specific administration tasks to secure it.

Review this diagram to understand where platform components are installed, then review the table to understand how they are secured.



Numbers in this diagram identify various Sybase Unwired Platform security features. Some features are standards of the platform, while others are optional and up to the administrator or developer to implement.

Component	How Secured
1. Mobile application and local data	• Login screens • Encryption of local data • Initial provisioning • Remote administration and security features See Device Security.
2. Unwired Server and runtime data services	• Authentication of users and administrators • Enforcing device registration • Secure communication to subcomponents • Secure administration of server and services See Server Security.
3. Cache (CDB) and messaging database	• Encryption of data and logs • Changing default passwords See Data Tier Security.
4. Enterprise Information Servers (EIS)	• Secure connections • Secure data change notifications See EIS Security.

Communication Security

Secure Unwired Platform component communications to prevent packet sniffing or data tampering. Different combinations of components communicate with different protocols and different ports.

Device-to-Platform Communications

Depending on your environment, devices can connect to either a Relay Server deployed to the DMZ (recommended), or to Unwired Servers directly deployed to the DMZ. In the former case, the Relay Server is the first line of defense to the Platform by acting as a proxy for the device and facilitating interactions with Unwired Servers installed on the corporate LAN.

- Relay Server connections — The traffic content depends on the payload protocol of the application:
  - Messaging communication encrypts the entire communication stream with a proprietary protocol.
  - Replication use an HTTP or HTTPS protocol which can optionally be encrypted end-to-end via TLS.

For Relay Server, configure the Web server host (IIS or Apache) to use a secure port, and use Sybase Control Center to configure Relay Server to use secureprotocols, ports, and certificates in Sybase Control Center.

On the client (in the case of mutual authentication), install certificates, and configure profiles to connect to Relay Server.

- Unwired Server connections — The traffic content also depends on the payload protocol of the application:
  - Messaging communication uses an fully encrypted end-to-end with a proprietary protocol.
  - Replication use an HTTP or HTTPS protocol.

Unwired Server and Device Application Communications

Unwired Server communicates differently with replication, messaging, or Gateway applications.

- Replication applications – can use HTTP or HTTPS. By default, the data content in HTTP is unencrypted but compressed. HTTPS keeps the data confidential. For additional security, application developers can add end-to-end encryption (E2EE), in which all the data is encrypted between the device and Unwired Server. To do this, an administrator uses a MobiLink utility to generate a key pair. The public key is installed on the client device, and the ConnectionProfile is configured with the key location. Data is encrypted using an AES in cipher block chaining mode; RSA handles the key exchange.
- Messaging applications, hybrid workflow apps, and Online Data Proxy/OData applications – network traffic is HTTP (HTTPS is not supported). Each HTTP message contains an encrypted message. It follows this process:
  1. When the Messaging Server is installed, the server generates an RSA key pair.
  2. When a device first contacts the server, it retrieves the public key and uses it to secure all future communication. For performance reasons, only a small section of the data from device to server is encrypted with the public key. Other items of note:
     - Developers can pre-provision the RSA public key to the client application using Afaria.
     - Administrators can enable auto-registration by setting up an application connection template in Sybase Control Center. Automatic registration means that administrator white-listing and generation of a single-use passwords is not necessary. For details, see *Automatically Registering Applications* in Sybase Control Center online help.
  3. Registration adds the user name and authorization code to a white list. When the messaging client connects to the messaging server, it passes the user name, the activation code, and the DeviceID to the server. The DeviceID is derived from the hardware.
  4. The device identified by the DeviceID is permanently assigned to that user and added to the white list. For every future interaction, communication session is initialized

using the public key. For the remainder of the session, a rotating sequence of AES keys is used to yield better performance.

All data transferred between the device and the Messaging Server is encrypted in this manner.

- Gateway applications – can use HTTP or HTTPS. Authentication over the channel varies:
  - If BES, use a HTTP Basic authentication.
  - If APNS, use Certificate authentication.

See also

- *Encrypting Synchronization for Replication Payloads* on page 79

Unwired Server and Data Tier Communications

Unwired Server uses two synchronization databases for its data tier: one for replication and one for messaging. It also connects to a cluster, monitor, and domain log database.

All communication streams between Unwired Platform and databases comprising are unencrypted, because they are exchanged on the corporate LAN. Nontheless, ensure that the local subnet is protected from network sniffing and file access.

Unwired Server and Sybase Control Center Communications

There are two different communication streams used to communicate with the Sybase Control Center administration tool: one for communications with the Sybase Control Center web console, and one for the communications with the Sybase Control Center X.X Windows service.

Communications between Unwired Server and the Sybase Control Center X.X Windows service use IIOPS on port 2001 by default. While Unwired Platform installs a sample certificate to enable the use of IIOPS automatically, you should exchange the certificate with a production-ready one immediately following installation.

There are two self-signed certificates that need to be changed: one for Unwired Server, and one for Sybase Control Center.

Note: The certificate alias of the default certificate that is used by both the IIOPS listener, and HTTPS listener for DCN, is sample1. When replace the IIOPS default certificate with a production certificate and keep the same alias, then you change the certificate for both of these listeners at the same time.

See also

- *Changing Installed Certificates Used for Unwired Server and Sybase Control Center HTTPS Listeners* on page 31

Unwired Server and EIS Communications

Secure communication between Unwired Server and any supported backend depends on the direction of the interaction between these components.

- EIS to Unwired Server – can communicate only via the DCN feature. DCN uses HTTP or HTTPS and all requests are also authenticated via the DCN User role. If you enable HTTPS, you can enable mutual authentication between these components as required. To secure the channel:
  - The EIS developer uses HTTPS to construct and send DCN requests to the listener.
  - The administrator manages the certificates, then uses Sybase Control Center to configure an HTTPS listener for DCNs.
- Unwired Server to EIS – Unwired Server can perform operation replays. The manner in which those replays are communicated depends on the EIS and whether or not the administrator secures this channel in Sybase Control Center:
  - REST/SOAP uses BASIC authorized over HTTP or HTTPS.
  - REST/SOAP for SAP uses BASIC/SSO2/X.509 authentication over HTTP or HTTPS.
  - JCo for SAP uses one of username/password, SSO2 tokens, or X.509 over SNC.
  - JDBC uses driver specific mechanisms to encrypt traffic. Review your JDBC driver documentation to learn how to configure this.

See also

- *Securing Data Change Notifications* on page 97

Unwired Server Nodes in Production Cluster Communications

(Not applicable to Online Data Proxy) Unwired Servers communicate with other Unwired Servers in the same cluster differently, depending on the type of communication performed.

- If multiple servers are used to perform replication synchronization, use the secure replication ports to negotiate which server will act as the primary synchronization server.
- If multiple servers perform cluster and domain configuration synchronization, use HTTP listeners. If a shared disk is used, the nodes do not use any network protocol for the file transfers; instead, they just access the disk.
- If multiple servers need to exchange JMS messages, they communicate indirectly through the shared databases of the data tier over IIOPS.

Authentication and Access Security

Authentication and role-based access control (RBAC) are core security feature supported by all application types to control access to enterprise digital assets. Review key concepts of authentication and role-based access control in Unwired Platform.

Security Provider Plug-in Model

Implement authentication and access control with the Common Security Infrastructure (CSI) component. Use CSI to authenticate and authorize administrator, developer, and end-user operations. CSI has a service provider plug-in model that integrates with the customer's existing security infrastructure.

Unwired Platform does not provide its own security systems for storing and maintaining users and access control rules, but delegates these functions to the enterprise's existing security solutions. Security provider plug-ins for many common security solutions are included with Unwired Platform.

One of the service provider types, the login module, authenticates the user. The login module interface conforms to the Java Authentication and Authorization Service (JAAS). All of the login modules in the Unwired Platform authenticate with userID and password credentials. Multiple login modules — each of which links to a different security store — can be stacked. When the user logs in, each login module attempts authentication in the order specified in the CSI configuration definition. The authentication attempt stops iterating the sequence when authentication has been achieved or rejected.

Security Configurations

Sybase Unwired Platform does not provide proprietary security systems for storing and maintaining users and access control rules, but delegates these functions to the enterprise's existing security solutions.

A security configuration determines the scope of user identity, performs authentication and authorization checks, and can be assigned multiple levels (domain or package). Applications inherit a security configuration when the administrator assigns the application to a domain via a connection template.

Users can be authenticated differently, depending on which security configuration is used. For example, a user identified as "John" may be authenticated different ways, depending on the named security configuration protecting the resource he is accessing: it could be an MBO package, a DCN request, use of Sybase Control Center .

Security configurations aggregate various security mechanisms for protecting Unwired Platform resources under a specific name, which administrators can then assign. Each security configuration consists of:

- A set of configured security providers. Security provider plug-ins for many common security solutions are included with the Sybase Unwired Platform.
- Role mappings (which are set at the domain and package level) that map logical roles to back end physical roles.

A user entry must be stored in the security repository used by the configured security provider to access any resources (that is, either a Sybase Control Center administration feature or an application package that accesses data sets from a back-end data source). When a user

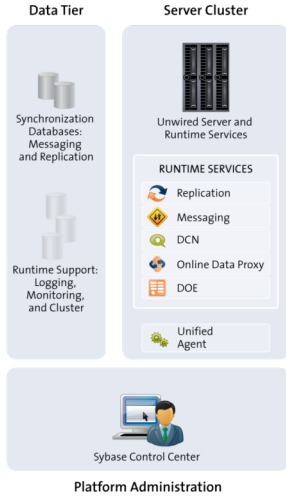
attempts to access a particular resource, Unwired Server tries to authenticate and authorize the user, by checking the security repository for:

- Security access policies on the requested resource
- Role memberships

Server Security

The Unwired Server provides data services to device clients by interacting with the data tier. The data tier is installed along with server tier components, to the internal corporate LAN.

Each runtime service uses its own communication port (secured and unsecured).



Secure the server runtime by performing activities that secure the infrastructure and administration of those components, in addition to enabling user authentication and secure communication.

1. *Securing the Server Infrastructure*

   Before you can secure the runtime, you must first secure the underlying infrastructure. This activity prevents files from being tampered with on the host, also allows the Unwired Server to run with existing security mechanisms.
2. *Securing Platform Administration*

   Use the Web-based console called Sybase Control Center to remotely and securely administer Unwired Platform.
3. *Enabling Authentication and RBAC for User Logins*

   Enable authentication and role-based access control (RBAC) for device user logins by creating a new security configuration (that is, one that is distinct from the "admin" security configuration on the "default" domain), and mapping roles, then assigning it.
4. *Encrypting Synchronization for Replication Payloads*

(Not applicable to Online Data Proxy) By default, replication is encrypted, which ensures that synchronization of enterprise dataoccurs safely.

Securing the Server Infrastructure

Before you can secure the runtime, you must first secure the underlying infrastructure. This activity prevents files from being tampered with on the host, also allows the Unwired Server to run with existing security mechanisms.

1. *Handling Intrusion Detection/Prevention Software*

   A personal firewall, or intrusion detection/prevention software (IPS or IDPS), can cause Unwired Platform components to malfunction or not function at all. Unwired Platform uses regular IP communication between components on the primary network interface of a computer, even when all components are installed on the same host.
2. *Setting File System Permissions*

   Unwired Platform runs as a collection of Windows services. By default, these services are installed to run as "Local System account".

See also

- *Securing Platform Administration* on page 20

Handling Intrusion Detection/Prevention Software

A personal firewall, or intrusion detection/prevention software (IPS or IDPS), can cause Unwired Platform components to malfunction or not function at all. Unwired Platform uses regular IP communication between components on the primary network interface of a computer, even when all components are installed on the same host.

If the local network interface is secured by intrusion detection/prevention software (IPS or IDPS, for example, McAfee Host Intrusion Prevention software or equivalent), you must configurethe security software to allow all network communication between Unwired Platform components.

For a single-node installation of all of the Sybase Unwired Platform components, try one of these options to work around the limitations imposed by the host intrusion prevention software and policy settings, without violating any security policy, until the settings of your security software are adjusted to the needs of Unwired Platform.

Choose an option:

- Removing the host machine from the network – this option ensures that all interconnections between Sybase Unwired Platform components are treated as local traffic and is not be flagged as incoming connections from external sources, thereby causing connection failures due to security policy setting. This option is suitable when you use your laptop in a network other than your corporate network, and want to demonstrate a

mobile solution using a simulator or emulator with all components running on the same machine. To use this option:

1. Stop the Sybase Unwired Platform services in the correct order. See *Starting and Stopping Unwired Platform Server Services* in *System Administration*.
2. Disconnect the host from all networks.
3. Restart Sybase Unwired Platform services in the correct order.
4. Change the Sybase Control Center URL link to use "localhost" or <*yourhostname*> as the host name, instead of the original fully qualified host name of the machine that included the domain name (for example: `https://localhost:8283/scc`, or `https://yourhostname:8283/scc`). Accept any security warnings to connect to Sybase Control Center.

- Connecting the host to the corporate network – this option ensures that all interconnections among Sybase Unwired Platform components are internal to your corporate network and validated against the corporate network security policy. The option of connecting to corporate network through VPN is especially suitable when you use your laptop in a network other than your corporate network, and want to demonstrate a mobile solution using your physical devices, and need outgoing connections to a backend Enterprise Information System (EIS) or Relay Server (Sybase Hosted Relay Server or otherwise).
  1. Stop the Sybase Unwired Platform services in the correct order. See the *Starting and Stopping Unwired Platform Server Services* topic in the *System Administration*.
  2. Reconnect the host to your corporate network directly or through corporate VPN, to ensure that the corporate network security policy applies.
  3. Restart Sybase Unwired Platform services in the correct order.
  4. Change the Sybase Control Center URL link to use "localhost" or <*yourhostname*> as the host name, instead of the original fully qualified host name of the machine that included the domain name (for example: `https://localhost:8283/scc`, or `https://yourhostname:8283/scc`). Accept any security warnings to connect to Sybase Control Center.
- Configuring the firewall software to allow connections to the ports the Unwired Platform uses. For a list of ports, see *Unwired Platform Ports* in *System Administration*.

Always check for the latest available patches and updates for your Unwired Server version on *http://downloads.sybase.com/swd/base.do?client=support*.

Setting File System Permissions

Unwired Platform runs as a collection of Windows services. By default, these services are installed to run as "Local System account".

You can restrict permissions by removing most users and groups from the Unwired Platform installation directory.

1. Open File Explorer.
2. Right-click <*UnwiredPlatform_InstallDir*>, and click **Properties**.

3. On the **Security** tab, click **Advanced**.
4. Unselect **Inherit from parent the permission entries that apply to child objects. Include these with entries explicitly defined here.**.
5. In the confirmation pop-up, choose **Copy**, then select **Replace permission entries on all child objects with entries shown here that apply to child objects.**.
6. In the table of Permission entries, remove all users except SYSTEM and Administrators entries.
7. Click **OK**.

Securing Platform Administration

Use the Web-based console called Sybase Control Center to remotely and securely administer Unwired Platform.

Sybase Control Center relies on a Windows service called Sybase Control Center X.X that runs on each Unwired Server on the cluster. The service handles communication between Sybase Control Center and Unwired Server runtime components.

1. *Enabling Authentication and RBAC for Administrator Logins*

   Administrators are authenticated both by Sybase Control Center and by Unwired Server. To make first-time installs easier, the Unwired Platform installer collects a single password that is used to configure and enable a security provider called PreconfiguredUser login module.
2. *Changing Installed Certificates Used for Unwired Server and Sybase Control Center HTTPS Listeners*

   Both Unwired Server and Sybase Control Center include default certificates that are used for these components' HTTPS listeners. Since all installations use the same certificates by default, you must change these certificates with production-ready ones after you install Unwired Platform.
3. *Securing Multiple Domains*

   To prevent role mapping leaks between multiple tenant domains, configure domains and assign shared security configurations.

See also

- *Securing the Server Infrastructure* on page 18
- *Enabling Authentication and RBAC for User Logins* on page 37

Enabling Authentication and RBAC for Administrator Logins

Administrators are authenticated both by Sybase Control Center and by Unwired Server. To make first-time installs easier, the Unwired Platform installer collects a single password that is used to configure and enable a security provider called PreconfiguredUser login module.

This module is not considered production-ready and must be substituted with one suitable for a production environment. This substitution must be coordinated between the configuration mechanisms used by Unwired Server and Sybase Control Center. If the configuration is not coordinated, future logins may fail.

Note: In most cases, production system can use any provider. An LDAP directory is a common choice, and has therefore been used as the example in this task sequence.

1. *Gathering Information on the LDAP Directory*

   Production environments rely on an LDAP directory to authenticate administrators. Consider which users need to be in the SUP Administrator or SUP Domain Administrator role, then identify or create LDAP groups corresponding to these roles. You must also allocate a group for the DCN User role.

2. *Logging Into Sybase Control Center With Installer Defined Password*

   The person acting as Platform administrator logs into Sybase Control Center for the first time after all Unwired Platform components are installed. During installation, an administrator user and password was defined. The Platform administrator must use these values to log into Sybase Control Center.

3. *Setting Up Login Security for Unwired Server*

   Extend installation security defaults with new production-ready providers. Because you are also configuring server logs to capture detailed security events, you will be able to validate these changes later in this task flow before finalizing the server security setup.

4. *Replicating Unwired Server Setup in Sybase Control Center*

   To prevent unexpected login issues, the changes you implement for the "admin" security configuration need to be repeated for Sybase Contorl Center. This task effects changes against two files: <UnwiredPlatform_InstallDir>\SCC-XX\conf\csi.properties and roles-map.xml.

5. *Validating the LDAP Provider*

   Once LDAP has been added as a provider to both the "admin" security configuration for Unwired Server and the CSI property file for Sybase Control Center you can test to see that the login works successfully before removing the PreconfiguredUser login module from both components' configurations.

See also

- *Changing Installed Certificates Used for Unwired Server and Sybase Control Center HTTPS Listeners* on page 31
- *SUP Administrator and SUP Domain Administrator* on page 77

- *Authentication in Unwired Platform* on page 38
- *Why Configuration of Sybase Control Center Must Be Coordinated With Unwired Server* on page 26

Gathering Information on the LDAP Directory

Production environments rely on an LDAP directory to authenticate administrators. Consider which users need to be in the SUP Administrator or SUP Domain Administrator role, then identify or create LDAP groups corresponding to these roles. You must also allocate a group for the DCN User role.

Note: If you have installed a previous version of Unwired Platform as part of a development deployment, you may have an OpenDS LDAP server running in your environment and both Unwired Platform and Sybase Control Center may be using this directory. Sybase no longer uses this directory and strongly encourages that you use a different LDAP directory and use the new one when configuring administration authentication for the current version of Unwired Platform.

1. Use an LDAP browser to evaluate existing groups.

   If there are existing groups that seem to already contain the right subjects that correspond to SUP DCN User, SUP Administrator, and SUP Domain Administrator platform roles, you can use those groups. The names need not be exact, as you can map them in Sybase Control Center to address those differences.
2. If no sufficient group exists, use an LDAP browser (or other native adminisration tool) to add them.
3. Add subjects to these groups to assign Unwired Platform corresponding permissions.
4. Determine what values are needed for LDAP login module properties in Unwired Platform.
   For example, providerURL, serverType, bind user, bind password, search base are values frequently used when configuring the login module for LDAP providers.

Logging Into Sybase Control Center With Installer Defined Password

The person acting as Platform administrator logs into Sybase Control Center for the first time after all Unwired Platform components are installed. During installation, an administrator user and password was defined. The Platform administrator must use these values to log into Sybase Control Center.

This password is not intended to be a permanent authentication solution. The login is authenticated by a PreconfiguredUser login module, which is enabled by default upon installation of the Platform. Later in this task flow you will be replacing this module with an LDAP authentication module used by both Unwired Platform and Sybase Control Center components.

1. Launch Sybase Control Center.
2. Enter the installer-defined name and password.

3. Click **Login.**
4. Open the Unwired Platform perspective and authenticate with Unwired Server using the same credentials used to log into Sybase Control Center.

Setting Up Login Security for Unwired Server

Extend installation security defaults with new production-ready providers. Because you are also configuring server logs to capture detailed security events, you will be able to validate these changes later in this task flow before finalizing the server security setup.

1. *Adding an LDAP Provider to the Admin Security Configuration*

   Modify the "admin" security configuration to add an LDAPLoginModule on the Authentication tab. Order this new provider and set the control flag attribute according to instructions, so this provider can be tested before the production-ready configuration are finalized.
2. *Mapping Unwired Platform Logical Roles to Physical Roles*

   Unwired Platform requires that you map these default platform roles: SUP Administrator and SUP Domain Administrator. In a development environment, these mappings are automatic if the provider roles use the exact logical role names. If the development environment roles do not match, then you must manually map them.
3. *Setting the Log Level for Security*

   When you validate the "admin" security configuration changes, you need to have the log level for the security component set to a sufficient level.

See also

- *Replicating Unwired Server Setup in Sybase Control Center* on page 25
- *Why Configuration of Sybase Control Center Must Be Coordinated With Unwired Server* on page 26

Adding an LDAP Provider to the Admin Security Configuration

Modify the "admin" security configuration to add an LDAPLoginModule on the **Authentication** tab. Order this new provider and set the control flag attribute according to instructions, so this provider can be tested before the production-ready configuration are finalized.

Configure the "admin" security configuration on the "default" domain to authenticate only platform and domain administrators. Do not use this security configuration ("admin" on "default") for MBO packages and mobile users; you must create a custom security configuration for this purpose.

1. In the navigation pane of Sybase Control Center, expand the **Security** folder, then click the security configuration named **admin**.
2. In the administration pane, click the **Authentication** tab.

3. Add an LDAPLoginModule, configuring the providerURL, serverType, bind user, bind password, search base, and other properties determined by you and the LDAP administrator.
4. Add a `ControlFlag` attribute for the configured LDAP login module, and set the value to `sufficient`.
5. Make the LDAP module the first module in the list.

   Note: Do not remove the PreConfiguredUser login module from the list of login modules used by the "admin" security configuration until the LDAP login module has been tested.
6. Select the **General** tab, select **Validate** then **Apply**.
7. Click **OK**.

See also

- *Mapping Unwired Platform Logical Roles to Physical Roles* on page 24
- *Preconfigured User Login Security Provider* on page 71
- *Preconfigured User Authentication Properties* on page 115
- *LDAP Security Provider* on page 67
- *LDAP Configuration Properties* on page 101

Preconfigured User Login Security Provider

If you are accessing Sybase Control Center for the first time you are authenticated as the Unwired Platform administrator with the PreconfiguredLoginModule. You must supply the password defined during the installation of Unwired Platform.

During installation,

Note: Do not forget this installation password. The installer hashes the password with a SHA-256 algorithm before it is saved as part of the PreconfiguredLoginModule configuration. This value cannot be returned to cleartext once the installer hashes it.

Once logged in, the Unwired Platform administrator immediately reconfigures the "admin" security configuration to replace this provider with an production-grade security provider like LDAP. Once complete, the corresponding changes then need to be implemented in security configuration files used by Sybase Control Center. Failure to coordinate these changes can result in unpredictable behavior in Sybase Control Center. If you configure a new provider in Unwired Platform and Sybase Control Center and login fails, review possible login failure solutions *Troubleshooting* guide.

Mapping Unwired Platform Logical Roles to Physical Roles

Unwired Platform requires that you map these default platform roles: SUP Administrator and SUP Domain Administrator. In a development environment, these mappings are automatic if the provider roles use the exact logical role names. If the development environment roles do not match, then you must manually map them.

Use Sybase Control Center to map these logical roles to the appropriate physical roles or groups in the underlying security provider. The mapping determines whether a platform or

domain administrator has access privileges. The administration logical to physical role mapping is done in "default" domain for the "admin" security configuration.

1. Open Sybase Control Center.
2. In the left navigation pane, expand **Domains**.
3. Expand the **Default** domain.
4. Open the Security folder and click the **Admin** security configuration.
5. Map roles to the security provider groups or roles:
   - If default roles exactly match the names in the security provider repository, select **AUTO**.
   - If the default role differ from those manually added to the repository, click the list adjacent to the logical role and choose **Map Role**. The Role Mappings dialog allows you to manually set logical and physical role mappings.Once saved, the state automatically changes to MAPPED.
6. Assign domain administration access to users :
   a) Register the user by clicking the **Security** node and selecting the **Domain Administrators** tab, then clicking **New**.
   b) Assign the required domain administrator physical role to the user in the underlying security provider repository for the **admin** security configuration.

See also

- *Adding an LDAP Provider to the Admin Security Configuration* on page 23

Setting the Log Level for Security

When you validate the "admin" security configuration changes, you need to have the log level for the security component set to a sufficient level.

You can only configure log settings on the primary server.

1. In the left navigation pane, expand the **Servers** folder and select the server to configure.
2. Select **Log**.
3. In the right administration pane, click the **Settings** tab.
4. For the security log component, set the log level to DEBUG.
   This gives you detailed system information, warnings, and all errors.
5. Click **Save**.
   The log file is located in: `<UnwiredPlatform_InstallDir>\<UnwiredPlatform>\Servers\UnwiredServer\logs\<hostname>-server.log`.

Replicating Unwired Server Setup in Sybase Control Center

To prevent unexpected login issues, the changes you implement for the "admin" security configuration need to be repeated for Sybase Contorl Center. This task effects changes against

two files: <*UnwiredPlatform_InstallDir*>\SCC-XX\conf\csi.properties and roles-map.xml.

1. *Configuring a Provider to Authenticate Sybase Control Center Logins*

   In a production environment, Unwired Server uses an LDAP provider to authenticate administrator login requests. Use the same values you used to configure the LDAP provider for Unwired Server.
2. *Mapping Unwired Platform Roles to Sybase Control Center Roles*

   Map Sybase Control Center roles to Unwired Platform roles, so a single userID controls which privileges the administrator has upon authentication and authorization. Use roles-map.xml to customize access privileges for your administrators.

See also

- *Setting Up Login Security for Unwired Server* on page 23
- *Validating the LDAP Provider* on page 30
- *Why Configuration of Sybase Control Center Must Be Coordinated With Unwired Server* on page 26

Why Configuration of Sybase Control Center Must Be Coordinated With Unwired Server

Learn about configuration defaults of Sybase Control Center after installation.

When Sybase Control Center is installed, it includes a file called csi.properties, which is used to define its security providers. For production installations, this file includes a single provider definition:

```
## SUP PreConfiguredUser Login Module
CSI.loginModule.2.options.moduleName=SUP PreConfiguredUser
Delegation Login Module
CSI.loginModule.
2.provider=com.sybase.ua.services.security.sup.SUPPreConfiguredUser
DelegationLoginModule
CSI.loginModule.2.controlFlag=sufficient
```

A separate file defines how roles are mapped (roles-map.xml), and contains these default mappings:

```
<module name="SUP PreConfiguredUser Login Module">
    <role-mapping modRole="SUP Administrator"
uafRole="uaAnonymous,uaAgentAdmin,uaPluginAdmin,sccAdminRole,sccUse
rRole,sccOperRole,sccGuestRole,jmxDirectAccess"/>
    <role-mapping modRole="SUP Domain Administrator"
uafRole="uaAnonymous,uaAgentAdmin,uaPluginAdmin,sccUserRole,sccOper
Role,sccGuestRole,jmxDirectAccess"/>
  </module
```

Together, these security and role mapping definitions tell Sybase Control Center to use the Unwired Server's configuration to authenticate Sybase Control Center logins, as well as to retrieve roles.

The `roles-map.xml` settings are looking for our built-in SUP Administrator and SUP Domain Administrator logical roles and it granting all necessary Sybase Control Center roles to the user. Whenever you are using Sybase Control Center, the Unwired Server is always going to authenticate the user against this configuration and determine if they have the "SUP Administrator" and/or "SUP Domain Administrator" logical roles. Therefore, Sybase Control Center must share the same configuration details used by the configuration used for Unwired Server administration (that is, the providers and properties defined by "admin" security configuration).

See also

- *Replicating Unwired Server Setup in Sybase Control Center* on page 25
- *Setting Up Login Security for Unwired Server* on page 23
- *Enabling Authentication and RBAC for Administrator Logins* on page 21

Configuring a Provider to Authenticate Sybase Control Center Logins

In a production environment, Unwired Server uses an LDAP provider to authenticate administrator login requests. Use the same values you used to configure the LDAP provider for Unwired Server.

Prerequisites

Always back up the `csi.properties` before changing it. That way, if you experience login issues, you can revert to an earlier instance of the file as required.

Task

You can make these changes while Sybase Control Center is running.

1. Use a text editor to open `<UnwiredPlatform_InstallDir>\SCC-XX\conf\csi.properties`.
2. Define a module in this file, similar to the LDAP sample below.

   Each line of the LDAP server module of the properties file must begin with "CSI.loginModule." followed by a module number. The module number in this sample is 8, however you should use an index value that places the LDAP login module before the PreconfiguredUser login module. The module number must be unique in the properties file, and you must use the same number in every line of the module.

   For example, this module configures an LDAP provider module using Active Directory, so that administrators can log in to Sybase Control Center with their Windows user name.

   Notice that the controlFlag attribute is also set to sufficient, and the debug attribute is set to true.

```
=====================================================

CSI.loginModule.
8.options.AuthenticationSearchBase=ou=sup,dc=mycompany,dc=com
CSI.loginModule.
```

```
8.options.BindDN=CN=suppad,ou=sup,dc=mycompany,dc=com
CSI.loginModule.8.options.BindPassword=mybindpassword
CSI.loginModule.
8.options.DefaultSearchBase=ou=sup,dc=mycompany,dc=com
CSI.loginModule.
8.options.AuthenticationFilter=(&(sAMAccountName={uid})
(objectclass=user))
CSI.loginModule.8.options.RoleFilter=(&(objectclass=groupofnames)
(objectclass=group))
CSI.loginModule.8.options.RoleScope=subtree
CSI.loginModule.8.options.AuthenticationScope=subtree
CSI.loginModule.8.options.ProviderURL=ldap://msadserver:389
CSI.loginModule.
8.options.RoleSearchBase=ou=sup,dc=mycompany,dc=com
CSI.loginModule.8.options.ServerType=msad2k
CSI.loginModule.8.options.moduleName=SUP LDAP Login Module
CSI.loginModule.8.controlFlag=sufficient
CSI.loginModule.
8.provider=com.sybase.ua.services.security.ldap.LDAPWithRoleLogin
Module
CSI.loginModule.8.debug=true
=====================================================
```

3. For some internal communication, you must include the Anonymous Login Module:

   ```
   # Anonymous Login Module
   CSI.loginModule.
   0.provider=com.sybase.ua.services.security.anonymous.AnonymousLog
   inModule
   CSI.loginModule.0.controlFlag=sufficient
   CSI.loginModule.0.options.moduleName=Anonymous Login Module
   CSI.loginModule.0.options.roles=uaAnonymous
   ```

   Adding this anonymous login module does not relax or allow anonymous access to the Sybase Control Center. Authentication and authorization checks are still enforced.

4. Save the file.
5. If your LDAP server uses a secure connection, and its SSL certificate is signed by a nonstandard certificate authority (for example, if it is self-signed), use the **keytool** utility to import it into the truststore. Execute a command similar to:

   ```
   keytool -import -keystore <UnwiredPlatform_InstallDir>\SCC_X-X
   \services\Messaging\lib\eas\lib\Repository\Security
   \truststore.jks -file
   <your cert file and path> -alias ldapcert -storepass changeit
   ```

   Note: The `\security\truststore.jks` file does not exist. You must create the folder and the file before saving the certificate to this location.

6. Restart the Sybase Control Center *X.X* Windows service.
7. Open Sybase Control Center and log in.

See also

- *Preconfigured User Login Security Provider* on page 71

• *Mapping Unwired Platform Roles to Sybase Control Center Roles* on page 29

Encrypting a Password

Use the **passencrypt** utility to encrypt passwords and other values that must be kept secure while stored in text files.

You can safely store an encrypted password in a properties file. Enter the password in clear text (unencrypted) when you execute **passencrypt** and when you use the password to log in.

passencrypt, which is located in the Sybase Control Center `bin` directory, uses the DES encryption algorithm.

1. Open a command window and change to the `bin` directory:
2. To encrypt a password, enter **passencrypt**. Enter your new password at the resulting prompt.
   passencrypt encrypts the password you enter (which does not appear on the screen) and displays the password in encrypted form.
3. Copy the encrypted password.
4. Paste the encrypted password where needed.

Mapping Unwired Platform Roles to Sybase Control Center Roles

Map Sybase Control Center roles to Unwired Platform roles, so a single userID controls which privileges the administrator has upon authentication and authorization. Use `roles-map.xml` to customize access privileges for your administrators.

An administrator other than the Unwired Platform administrator can secure and manage resources from Sybase Control Center. However, if they are the same individual, you can map roles so a single login allows administration privileges to both Sybase Control Center and Unwired Platform.

Note: `roles-map.xml` is a file used by multiple Sybase products; therefore, you see roles in this file you do not need. Edit the file as described here to avoid issues that may arise with unrecognized roles being defined in this file.

1. Add these lines:

```
<role-mapping modRole="MySUPplatformAdminRole"
uafRole="uaAnonymous,uaAgentAdmin,uaPluginAdmin,sccAdminRole,sccU
serRole" />
<role-mapping modRole="MySUPDomainAdminRole"
uafRole="uaAnonymous,uaAgentAdmin,uaPluginAdmin,sccUserRole" />
<role-mapping modRole="MyNon-SUPDomainAdminRole"
uafRole="uaAnonymous,uaAgentAdmin,uaPluginAdmin,sccUserRole" />
```

   The last two entries give both domain administrators access as a SCC User (which is mapped to sccUserRole). This means that the user can use the Unwired Platform plugin to administer Unwired Server.

2. Ensure that the roles defined in the LDAP repository map to the UAF* roles in this file.

   By default, the role mapping file contains these roles in the LDAP Login Module definition:

   ```
    <module name="SUP LDAP Login Module">
         <role-mapping modRole="MySUPplatformAdminRole"
   uafRole="uaAnonymous,uaAgentAdmin,uaPluginAdmin,sccAdminRole,sccU
   serRole" />
         <role-mapping modRole="MySUPDomainAdminRole"
   uafRole="uaAnonymous,uaAgentAdmin,uaPluginAdmin,sccAdminRole,sccU
   serRole" />
       </module>
   ```

   These lines map logical Sybase Control Center roles to the provider's physical roles. For example, replace *MySUPplatformAdminRole* and *MySUPDomainAdminRole* with the name of the LDAP group for you created for "SUP Administrator" and "SUP Domain Administrator" users.

   Change *MySUPplatformAdminRole* and *MySUPDomainAdminRole* to the role names used by your provider.

   As a Sybase Control Center administrator, a user who is granted this role can perform administration and configuration tasks from the Unwired Platform management console after a successful login.

3. To add role mapping for the Anonymous Login Module to the uaAnonymous role:

   a) Add the uaAnonymous role to the `<uaf-roles>` section of the `roles-map.xml` file:

   ```
   <role name="uaAnonymous" description="Anonymous role" />
   ```

   b) Add the role mapping in the `<security-modules>` section of the `roles-map.xml` file:

   ```
   <module name=''Anonymous Login Module''>
                       <role-mapping modRole=''uaAnonymous''
   uafRole=''uaAnonymous'' />
       </module>
   ```

See also

- *Configuring a Provider to Authenticate Sybase Control Center Logins* on page 27

Validating the LDAP Provider

Once LDAP has been added as a provider to both the "admin" security configuration for Unwired Server and the CSI property file for Sybase Control Center you can test to see that the login works successfully before removing the PreconfiguredUser login module from both components' configurations.

1. Log into Sybase Control Center using the login values of an LDAP user that is in an LDAP group mapped to the "SUP Administrator" logical role.

2. If the login succeeds:
   a) Remove PreconfiguredUser login module from Unwired Server and Sybase Control Center setup locations.
   b) Reduce the logging levels for both Unwired Server and Sybase Control Center to Info or Warn.
3. If the login fails:
   a) Check Sybase Control Center's log in `<UnwiredPlatform_InstallDir>\SCC-X_X\log\agent.log` to see if authentication failed with this component.
   b) If no issues are identified, continue checking with the Unwired Server log in `<UnwiredPlatform_InstallDir>\<UnwiredPlatform>\Servers\UnwiredServer\logs\<hostname>-server.log`.

      Look for hints about may be occurring with the Unwired Server login modules.

See also

- *Replicating Unwired Server Setup in Sybase Control Center* on page 25

Changing Installed Certificates Used for Unwired Server and Sybase Control Center HTTPS Listeners

Both Unwired Server and Sybase Control Center include default certificates that are used for these components' HTTPS listeners. Since all installations use the same certificates by default, you must change these certificates with production-ready ones after you install Unwired Platform.

Prerequisites

By default, Unwired Server includes two security profiles, which is used by secure management and Data Change Notification (DCN) listeners: default and default_mutual. Therefore, you need to determine what type of authentication is required. The security profile you use determines which certificate file you need, and where they need to be deployed. The most secure profile is default_mutual, whereby components are mutually authenticated.

- The default security profile uses domestic authentication and uses the alias of "sample1". With this authentication type, Unwired Server sends its certificate to the client (that is, either Sybase Control Center or DCNs). However, it does not require a certificate in return from the client. Instead, you must configure the client to trust the Unwired Server certificate.
- The default_mutual security profile uses domestic_mutual authentication and uses the alias of "sample2". This authentication type requires that Sybase Control Center and Unwired Server truststores each contain a copy of the other component's certificate.

For details about what cipher suites are supported for domestic and domestic_mutual authentication, see *Creating an SSL Security Profile in Sybase Control Center* in the Sybase Control Center online help.

Task

Note: Because secure DCN has automatically been configured to use these same profiles by default, you are updating certificates used for secure DCN communication. If you want DCN to use a unique profile and certificates, see *EIS Tier Security*.

1. Generate new production-ready certificates:
   a) For Unwired Server: if you are using default , create new server certificates for Unwired Server and keep the current alias of "sample1"; if you are using default_mutual also generate new server certificates for Sybase Control Center and keep the current alias of "sample2". This replaces the sample certificates in this keystore.
      - If you use a PKI system, ensure that the generated certificates and key pairs are signed by the Certificate Authority (CA) certificate that is widely trusted in your organization. Unwired Platform is compliant with certificates and key pairs generated from most well known PKI systems. Sybase recommends that you use this option.
      - If you do not use a PKI system, use the **keytool** utility to generate new self-signed certificates by following these steps. For an example of a **keytool** command, see *Preparing Certificates and Key Pairs*.

      Note: For a clustered environment, set the CN of the certificate to `*.domain`. The truststore and keystore files, as well as the definitions for default and default_mutual profiles are then synchronized across the cluster. As a result, there will only ever be a single certificate shared by all nodes that are members of the same cluster.
   b) For Sybase Control Center: generate a new certificate for this keystore with a "jetty" alias. This replaces the default self-signed certificate installed in that keystore.
2. Import production-ready certificates, then update the security profile to associate these files with the Unwired Server encrypted port.
   a) Use **keytool** to import the new production certificates into the primary Unwired Server keystore.
   b) In the left navigation pane, expand the **Servers** folder and select the primary Unwired Server.
   c) Select **Server Configuration**.
   d) In the right administration pane, click **General** then **SSL Configuration**.
   e) Optional. If you have used a different alias, rather than keep the alias of "sample1", locate the profile name row and modify the alias name to match the one used by your certificate.
   f) Optional. If you are using a PKI system that includes OCSP, configure an OCSP responder. See *Enabling OCSP*.
3. Update Sybase Control Center keystores and configure it to also use these production-ready certificates.

a) Use **keytool** to import the new production Unwired Server certificate into the Sybase Control Center keystore at `<UnwiredPlatform_InstallDir>\SCC-XX\plugins\com.sybase.supadminplugin_X.X.X\security\truststore.jks`.

b) Open `<UnwiredPlatform_InstallDir>\SCC-XX\services\Messaging\lib\eas\lib\Repository\Server\EmbeddedJMS\Instance\com\sybase\djc\server\ApplicationServer\EmbeddedJMS.properties`, and revise the *filePath*,*keyStoreName*, *trustStoreName* and *password* properties, so that Sybase Control Center can locate and access these stores.

4. Optional. If you are using default_mutual authentication, use **keytool** to import the new server certificate for Sybase Control Center into the primary Unwired Server truststore.
5. Replace the default certificate for Sybase Control Center's HTTPS listener. Use **keytool** to import the new Sybase Control Center certificate with the "jetty" alias to the `<UnwiredPlatform_InstallDir>\SCC-X_X\keystore` keystore.

See also

- *Enabling Authentication and RBAC for Administrator Logins* on page 21
- *Securing Multiple Domains* on page 36

Enabling OCSP

(Optional) Enable OCSP (Online Certificate Status Protocol) to determine the status of a certificate used to authenticate a subject: current, expired, or unknown. OCSP configuration is enabled as part of server level SSL configuration. OCSP checking must be enabled if you are using the CertificateAuthenticationLoginModule and have set Enable revocation checking to true.

Enable OCSP for an Unwired Server when configuring SSL.

1. To enable OCSP when doing certificate revocation checking, check **Enable OCSP**.
2. Configure the responder properties (location and certificate information):

Responder Property	Details
URL	A URL to responder, including its port. For example, `https://ocsp.example.net:80`.

Responder Property	Details
Certificate subject name	The subject name of the responder's certificate. By default, the certificate of the OCSP responder is that of the issuer of the certificate being validated. Its value is a string distinguished name (defined in RFC 2253), which identifies a certificate in the set of certificates supplied during cert path validation. If the subject name alone is not sufficient to uniquely identify the certificate, the subject value and serial number properties must be used instead. When the certificate subject name is set, the certificate issuer name and certificate serial number are ignored. For example, `CN=MyEnterprise, O=XYZCorp`.
Certificate issuer name	The issuer name of the responder certificate. For example, `CN=OCSP Responder, O=XYZCorp`.
Certificate serial number	The serial number of the responder certificate.

Using Keytool to Generate Self-Signed Certificates and Keys

Whenever possible, use a PKI system and a trusted CA to generate production-ready certificates and keys that encrypt communication among different Unwired Platform components. You can then use **keytool** to import and export certificate to the platform's keystores and truststores. Otherwise, you can also use **keytool** to generate self-signed certificates and keys.

Review sample commands, to see how to use **keytool** to import, export, and generate certificates and keys:

1. If you have the root certificate of the certificate authority (CA) or if you have a self-signed certificate, import the CA certificate into the keystore and truststore.
   For example, if you have a CA certificate in a PKCS#10 file named `cust-ca.crt`, run this command from the *<UnwiredPlatform_InstallDir>*`\UnwiredPlatform\Servers\UnwiredServer\Repository\Security` directory:

   ```
   keytool -importcert -alias customerCA -file cust-ca.crt -storepass
   changeit -keystore truststore.jks -trustcacerts
   ```

The truststore is used when Unwired Platform makes an out-bound connection over SSL to another server with a server certificate. Unwired Server checks that the server certificate is in the truststore, or is signed by a CA certificate in the truststore.

2. Generate a key pair in the Unwired Platform keystore.

   The command you use depends on the environment for which you are generating the keystore. For most Unwired Platform deployments, this command may be sufficient:

   ```
   keytool -genkeypair -alias supServer -keystore keystore.jks -
   keyalg RSA -keysize 2048
   -validity 365 -keypass mySecret -storepass changeit
   ```

   However, if you are generating a key pair to secure an HTTPS communication port between the SAP Gateway and Unwired Server for OData push notifications, you might use a command like:

   ```
   keytool –genkeypair –alias SAPpush –keyalg RSA –keysize 1024 –
   sigalg SHA1withRSA
   –keypass mySecret –keystore keystore.jks
   ```

3. Supply values for each of the resulting prompts.

   The first prompt is the most critical. If you are running multiple Unwired Server in a cluster, type an asterisk followed by the domain name where the Unwired Servers are running.

   ```
   What is your first and last name?
   [Unknown]: *.mydomain.com
   What is the name of your organizational unit?
   [Unknown]: myOU
   What is the name of your organization?
   [Unknown]: mycompany
   What is the name of your City or Locality?
   [Unknown]: place
   What is the name of your State or Province?
   [Unknown]: state
   What is the two-letter country code for this unit?
   [Unknown]: AB
   Is CN=*.mySUPdomain.com, OU=myOU, O=mycompany,
   L=place, ST=state, C=AB correct?
   [no]: y
   ```

   Note: The asterisk before the domain name allows this same certificate to be used by multiple Unwired Servers deployed as members of a common cluster. The CN value must be the domain name of the host on which Unwired Server is installed.

4. Generate a certificate signing request, send it to the certificate authority, and install the issued certificate in the Unwired Server keystore:

   a) Generate a certificate signing request (CSR). For example:

   ```
   keytool -certreq -alias supServer -keystore keystore.jks -
   storepass changeit
   -keypass mySecret -file supServer.csr
   ```

b) Send the CSR to the CA for signing.
   For example, for SAP, may perform steps similar to:
   1. Launch the URL for your SAP CA.
   2. Change the option to **Certify the cert req** in the **select cmd** option.
   3. Paste the content of the `.csr` file generated in the previous step.
   4. Copy the content between (and including) `"-----BEGIN CERTIFICATE-----" "-----END CERTIFICATE-----"` of the response, to a text file named `<name of the cert>.cer`.
   5. View and verify the status of the certificate.
c) Use `keytool` to import the CA.

Note: The -alias/-keypass values are the same as those used to generate the key pair and CSR. By sharing these values, you pair the signed certificate with the keypair:

```
keytool -importcert -alias supServer -file supServer.crt -
keypass mySecret -storepass changeit
-keystore keystore.jks -trustcacerts
Certificate reply was installed in keystore
```

Securing Multiple Domains

To prevent role mapping leaks between multiple tenant domains, configure domains and assign shared security configurations.

Sybase recommends that the Platform administrator:

1. Create at least one new tenant domain in Sybase Control Center. You may require more, depending on your mobility strategy.
2. Restrict the use of the "admin" security configuration on the "default" domain to administration authentication only.
3. Assign at least one domain administrator. Depending on the maintenance issues of large-scale deployments, the administrator may want to use at least one Domain administrator per domain.
4. Create and assign at least one new security configuration. The administrator may create and assign security configurations, if security requirements (stringency, uniqueness) differ between tenant domains.

For more information, search for *Domains* in *Sybase Control Center* online help.

For example, a company named "Acme" has two separate divisions, HR and sales. The employees in each division use different mobile applications. In this case, Sybase recommends using two domains in Sybase Control Center to simplify the management of packages, users, applications and related artifacts.

Acme implements separate domain administrators for each domain, but is using a single "acme" security configuration due to the way the corporate LDAP directory is configured. This configuration includes an LDAPLoginModule provider that uses this URL:

```
ldap://ldap.acme.com
```

As a result, all employees of all domains are authenticated by the same LDAP server, and authorized by the same set of groups and roles.

Note: Because domain administrators are authenticated from the same acme LDAP repository via the admin security configuration on the default domain, those role mappings can "leak" between domains. Consequently, a domain administrator assigned to one domain gets granted access to another. This side-effect is undesirable and should be avoided.

See also

- *Changing Installed Certificates Used for Unwired Server and Sybase Control Center HTTPS Listeners* on page 31

Benefits and Drawbacks of a Shared Security Configuration

Determine whether or not you should use a shared security configuration across multiple domains.

Sybase recommends using different named security configuration for each domain, unless you are willing to accept the risks and domain administrators collaborate before implementing changes.

Benefit	Drawback
• Set up the modules you required in a named security configuration once.	• A domain administrator from one domain can make changes to role mapping at the default level, potentially with adverse effects to packages deployed to a different domain.

Enabling Authentication and RBAC for User Logins

Enable authentication and role-based access control (RBAC) for device user logins by creating a new security configuration (that is, one that is distinct from the "admin" security configuration on the "default" domain), and mapping roles, then assigning it.

Of all the default roles included with Unwired Platform, only the "SUP DCN User" role must be mapped, and only if DCNs are used for MBO packages associated with the security configuration you create for device user logins.

1. *Creating a Security Configuration for Device Users*

   Create and name a set of security providers and physical security roles to protect Unwired Platform resources. For device user authentication, create at least one provider that is not the "admin" security configuration on the "default" domain, which is used exclusively for administrator authentication in Sybase Control Center.
2. *Assigning Providers to a Security Configuration*

Assign providers after you have created a security configuration.

3. *Assigning Security Configurations to Domains, Packages, or Applications*

   Once the platform administrator creates a security configuration, it can be assigned to a domain. Domain administrators can then select the security configuration when deploying synchronization packages or creating application templates.

4. *Mapping Roles for Domains, Packages, or Applications*

   Role mappings can occur at two levels. Package level mappings override the mappings set at the default level. A default level is a level from which the role mapping is inherited.

See also

- *Securing Platform Administration* on page 20
- *Encrypting Synchronization for Replication Payloads* on page 79
- *Authentication in Unwired Platform* on page 38

Authentication in Unwired Platform

A security provider verifies the identities of application users and administrators who request access via one or more configured login modules.

Device user authentication and administrator authentication are configured differently:

- device users are authenticated with custom Unwired Server security configurations created by the platform administrator in Sybase Control Center. For SAP EIS backends, SSO authentication can be configured.
- Administrators are authenticated with the "admin" security configuration on the "default" domain. For first-time logins, administrators are authenticated with the PreconfiguredLoginModule. Once logged in, administrators for production systems should immediately reconfigure security to use the enterprise security backend and delete this login module from Sybase Control Center. .

See also

- *Enabling Authentication and RBAC for User Logins* on page 37
- *Enabling Authentication and RBAC for Administrator Logins* on page 21

Single Sign-on for SAP

Unwired Platform supports single sign-on (SSO) authentication for mobile clients that access data from an SAP enterprise information system (EIS) using either X.509 certificates or SSO logon tickets (SSO2).

Single sign-on credential support for SAP includes:

- X.509 certificates – use the CertificateAuthenticationLoginModule provider to implement X.509 authentication. At runtime, the mobile client selects the certificate signed by a trusted CA, which is authenticated by the SAP EIS.

- SAP single sign-on (SSO2) tokens – use the HttpAuthenticationLoginModule provider for both basic HTTP authentication and to implement SSO2. At runtime, the client enters a user name/password combination that maps to a user name/password in the SAP EIS. For SSO2, a token is obtained from the configured SAP server using the client-supplied user name/password and is forwarded to other SAP servers configured in the endpoints to authenticate the client, instead of using client-supplied user name/password credentials.

Single Sign-on Authentication

Understand the role of user credentials and X.509 certificates in single sign-on authentication.

Encrypt the communication channel between Unwired Server and the SAP EIS for security reasons:

- For Web services, DOE, and Gateway interactions this requires an HTTPS communication path with mutual certificate authentication. Use Sybase Control Center to navigate to the corresponding connection pool, edit the properties and add the properties "Certificate Alias" (give the name of a certificate alias in the keystore.jks), and "Password" (provide the key password).
- For JCo connections you must configure the SNC properties.

During mutual certificate authentication, the client presents a certificate to Unwired Server. In order for authentication to succeed, the client's certificate, or more typically the CA certificate that signed the client certificate must be present in the Unwired Server truststore. The Unwired Server truststore also contains a server-certificate (CN=host.domain) which is issued by the server (SAP for example), and which other SAP servers are configured to trust, meaning that once the server-certificate is authenticated during the HTTPS mutual certificate authentication, the SAP server further trusts that the credentials (SSO2 or X.509 values) given to identify the end user are correct, and the SAP server executes its EIS operations as that asserted end-user.

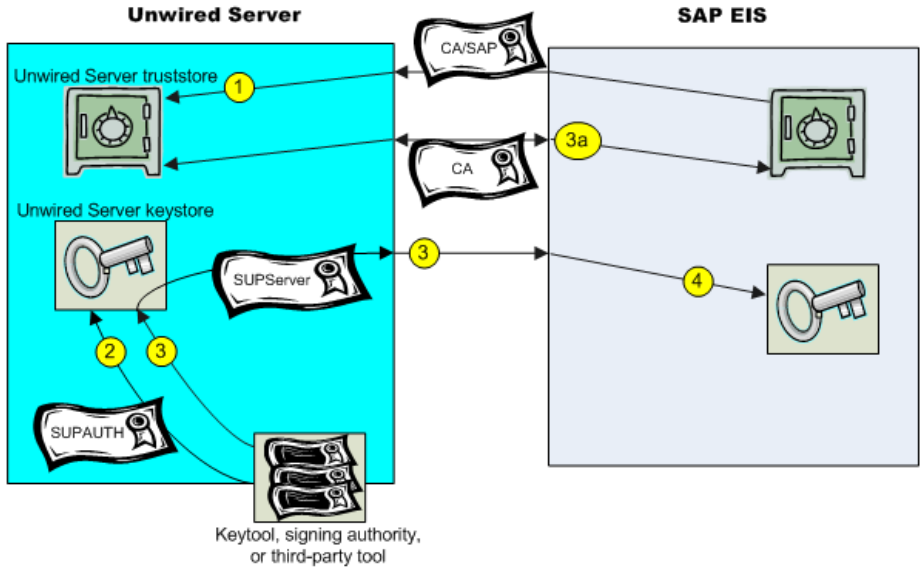
There is a separate notion of a "technical user" (CN=someTechUserName), which is different than the (CN=host.domain) server-certificate used for SSO. In a "normal" pooled JCo connection, the username is a technical user, and all RFCs are executed in the SAP EIS as that user. The technical user is granted all rights and roles within SAP to allow it to execute the range of RFCs behind the MBOs, which is the opposite of SSO.

See also

- *Enabling Single Sign-on for DOE-C Packages* on page 4
- *SAP Single Sign-on and DOE-C Package Overview* on page 42
- *SAP Single Sign-on and Online Data Proxy Overview* on page 55
- *Enabling Single Sign-on for OData Applications* on page 6
- *SAP Single Sign-on and Mobile Business Object Package Overview* on page 44
- *Enabling Single Sign-on for Mobile Business Object Packages* on page 5

Configuring X.509 Certificates for SAP Single Sign-on
Import, export, and generate the X.509 certificates that secure communication paths between Unwired Server and the SAP enterprise information system (EIS), and for client authentication, including single sign-on (SSO) with X.509 or SSO2 tokens.

Figure 1: Creating, Importing, and Exporting Certificates



Use Java **keytool** commands to import these certificates into the Unwired Server truststore and keystore.

1. Import SAP CA certificates into the Unwired Server truststore, including:
   - The standard SAP/DOE server root certificate (.crt or .cer) required to establish a trusted relationship between Unwired Server and the SAP EIS.
   - Any CA certificate used to sign .pse certificates used for JCo/SNC communications.
   - For Gateway deployments where Unwired Server is the Online Data Proxy (ODP), import the Gateway server's CA into the truststore of Unwired Platform.

     The ODP requires two certificate files: one that contains the certificate and private key for use by the server, and another that contains only the certificate for use by clients. The certificates should be in the form of a PKCS#10 file using an RSA key pair (key lengths in the range of 512–16384 are supported), in PEM or DER format. The key usage should be set to Key Encipherment, Data Encipherment, Key Agreement (38).
   - Any other required SAP CA certificate. For example, any CA certificate used to sign a client certificate that is to be authenticated by Unwired Server must be imported if you are implementing SSO with X.509.

Note: If Unwired Server is communicating with a server that is hosting a Web service that is bound to SAP function modules, import that server's CA certificate into the Unwired Server truststore.

For example:

```
keytool -import -keystore <UnwiredPlatform_InstallDir>/
UnwiredPlatform/Servers/UnwiredServer/Repository/Security/
truststore.jks -file <CertificateFile>
```

```
Enter keystore password:  changeit
Trust this certificate? [no]:  yes
```

2. Create a keystore on the Unwired Server host into which you can import the certificate and private key (PKCS #12) issued by the SAP system administrator, then import the certificate into the Unwired Server keystore. This certificate secures communications for packages and is used when a user uses an X.509 certificate rather than a user name and password. For example:

   ```
   keytool -importkeystore -srckeystore SUPAUTH.p12 -
   srcstoretype pkcs12 -srcstorepass <techuserpass> -srcalias
   CERTALIAS -destkeystore <UnwiredPlatform_InstallDir>/
   Servers/UnwiredServer/Repository/Security/keystore.jks -
   deststoretype jks -deststorepass changeit -destkeypass
   changeit
   ```

   Even if the EIS administrator is using the native SAP public-key infrastructure (PKI) to generate certificates, you must still import them into the Unwired Server keystore. The certificate name, *SUPAUTH* and alias, *CERTALIAS* represent the type of package/client to be authenticated, for example:

   - TechnicalUser certificate with doectech alias – a DOE-C package client.
   - SAPUser certificate with SAPClient alias – a SAP or Web service MBO package client.

3. Create and import the SUPServer certificate into the Unwired Server keystore. For example:

   ```
   keytool -importkeystore -srckeystore SUPServer.p12 -
   srcstoretype pkcs12 -srcstorepass <supserverpass> -srcalias
   SUP -destkeystore <UnwiredPlatform_InstallDir>/Servers/
   UnwiredServer/Repository/Security/keystore.jks -
   deststoretype jks -deststorepass changeit -destkeypass
   changeit
   ```

Note: (3a) You can create the SUPServer certificate using Java keytool commands, a third-party tool such as OPENSSL, or the signing authority used to create all SAP server certificates, in which case you need not import any other CA signing authority certificate

into the Unwired Server truststore. However, if you create the SUPServer certificate with another CA signing authority, you must import that CA certificate into both the Unwired Server truststore, and into the SAP Server using the STRUST transaction.

4. Import the SUPServer certificate into SAP/DOE server using the STRUST transaction.

You can now configure your environment for mutual authentication and SSO, in which any client connecting to Unwired Server presents credentials, and a server certificate (SUPAUTH) is selected for Unwired Server to present to clients.

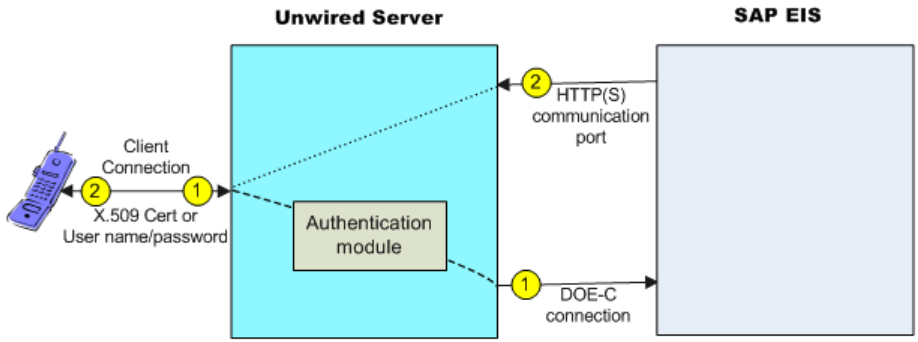
SAP Single Sign-on and DOE-C Package Overview

Understand how DOE-C packages fit in the Unwired Platform landscape, including how to secure communication paths and enable single sign-on (SSO) for these packages.

DOE-C is the connector from Unwired Server to SAP NetWeaver Mobile, which contains the DOE. Unwired WorkSpace is not used to create MBOs, generate code, create applications, or for deployment. Instead, in DOE-based mobile applications that run in the Unwired Platform environment:

- NetWeaver Mobile handles the data modeling for DOE-C connections.
- Field mappings, connection information, and other application- and package-specific information is defined in the ESDMA, for example the SAP CRM ESDMA, which is deployed to Unwired Server, and automatically converted into an Unwired Platform package by the ESDMA converter.
- DOE-C packages are message-based – NetWeaver Mobile is message-based, and performs queue handling, data caching, and is push-enabled to push data changes out to mobile devices through Unwired Server.

Unwired Server works as a pass-through gateway in the DOE/DOE-C configuration:



1. A DOE-C client application registers with Unwired Server and subscribes to message channels. Unwired Server remembers the push notification information/deviceID/applicationID from the client, but forwards the subscription to DOE through the DOE-C

connection (HTTP(S)) to the DOE. When the client performs an operation, that operation flows through Unwired Server via this same connection to the DOE.

In an SSO configuration, the client provides credentials to Unwired Server (user name and password or X.509 user certificate) that are authenticated by the security configuration's authentication module (CertificateAuthenticationLoginModule for X.509 or HttpAuthenticationLoginModule for SSO2). Once authenticated by Unwired Server, and assuming that Unwired Server and the SAP Server have a secure communication path, SSO is enabled.

2. When application data changes in the SAP EIS and the DOE determines that a particular client has a subscription to that change, DOE connects to the Unwired Server HTTP(S) port and sends a message identifying the client, along with the message payload. Unwired Server looks up the client and queues a message. If the client is connected, the message is delivered immediately. If the client is offline, then Unwired Server attempts to send a push notification to the client (BES HTTP Push for Blackberry, APNS notification for iOS) to attempt to wake up the client and have it retrieve the messages.

   WindowsMobile does not have a separate push notification protocol, so Unwired Server waits for those clients to connect and retrieve their messages.

See also

- *Enabling Single Sign-on for DOE-C Packages* on page 4
- *Single Sign-on Authentication* on page 39

Enabling the DOE-C Connection and Assigning the TechnicalUser Security profile

Configure the SAP Data Orchestration Engine Connector (DOE-C) connection pool between Unwired Server and the SAP EIS. This is the port on which Unwired Server communicates with the DOE, including forwarding subscriptions and allowing client operations to flow through to the DOE.

This type of connection is available in the list of connection templates only after a DOE-C package has been deployed to Unwired Server.

1. From Sybase Control Center, expand **Domains** > **<DomainName>**, and select **Connections**.

   DomainName is the domain that contains the DOE-C package.

2. Select an existing connection pool or select **New**, and enter a new connection pool name, for example, `SAPDOEConnection`, select **SAP** as the connection pool type and **Data Orchestration Engine template**.

3. Enter the property values required to enable an authenticated HTTPS connection to the DOE.

   If defining a security profile to implement mutual authentication with basic authentication, add the `certificateAlias` property, which overrides the technical user name and password fields. The technical user name and password fields can be empty, but only if

`certificateAlias` is set. The specified certificate is extracted from the Unwired Server keystore and supplied to the DOE.

4. Select **Save**.

Deploying and Configuring DOE-C Packages

Deploy the ESDMA (DOE-C package) to an Unwired Server domain using the DOE-C command line utility (CLU).

1. Start the command line utility console.
2. Deploy the ESDMA. During deployment, you can set the domain and security configuration using the **setPackageSecurityConfiguration** command, or perform this task later from Sybase Control Center.
   See the *Sybase SAP DOE Connector Command Line Utility* appendix in the *SAP DOE Connector Installation Guide*.

Next

Verify or set the security configuration for the domain or package.

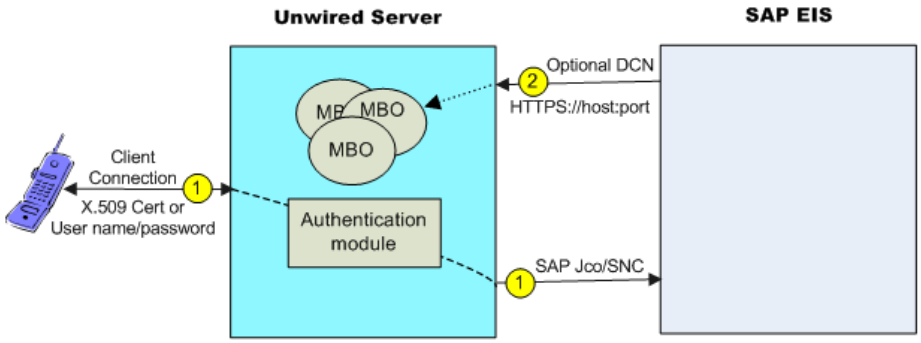
SAP Single Sign-on and Mobile Business Object Package Overview

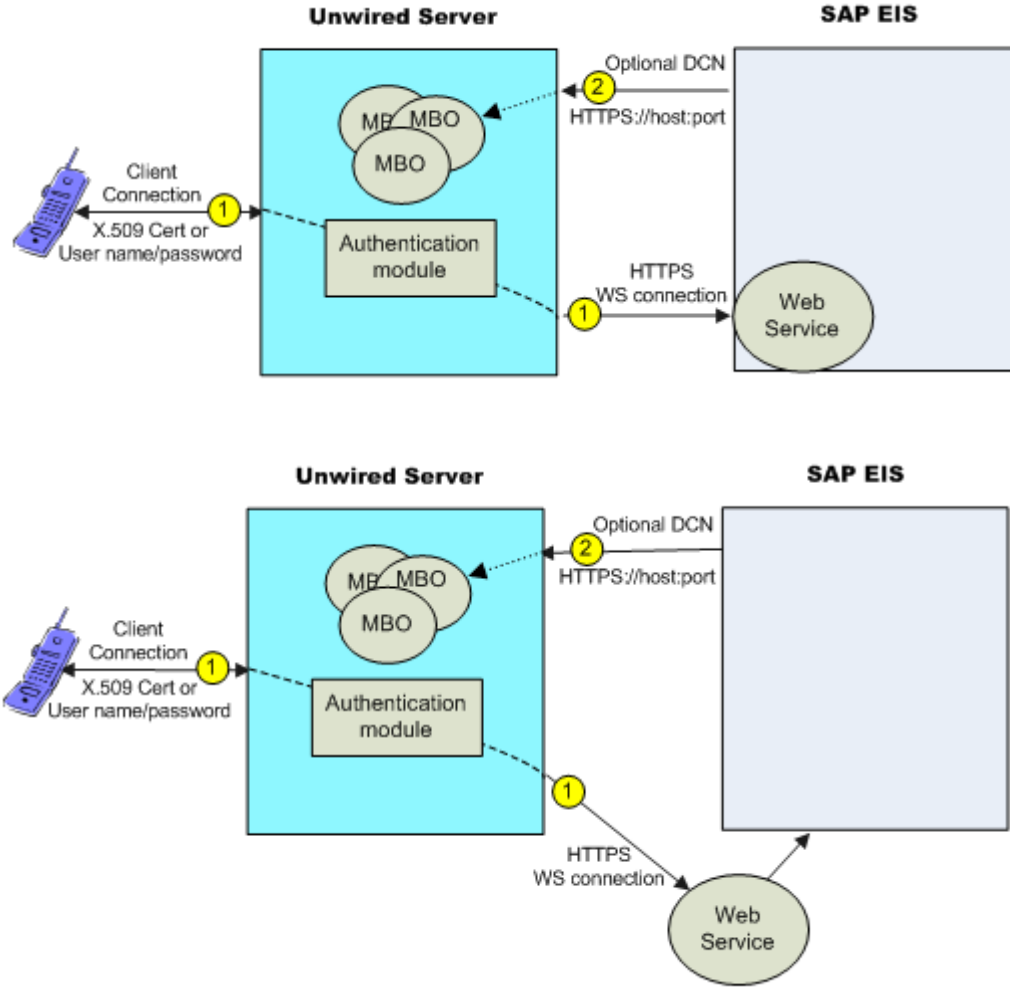
Understand how to secure communication ports and enable single sign-on (SSO) for packages that contain mobile business objects (MBOs) bound to an SAP enterprise information system (EIS).

SAP MBOs bound directly to SAP BAPIs and RFCs, as well as SAP BAPIs exposed as Web services. Once deployed, Unwired Platform supports Java connector (JCo) connections and Secure Network Communications (SNC) for SAP MBOs, and HTTP(S) connections to Web services.

Once deployed, connection information, and other application- and package-specific information is maintained by Unwired Server. Unwired Server packages that contain SAP MBOs support message-based and replication-based applications and perform queue handling, data caching, and synchronization services.

Typical data flow for SAP MBO packages that use data change notification (DCN) as a refresh mechanism:





1. Data flows from Unwired Server to the EIS through a configured connection pool. For secure connections:
   - Jco – communicates with the SAP EIS using the SAP JCo proprietary communication protocol. Optionally use SNC if required for your installation.
   - Web service – communicates to the Web service host using HTTPS, whether the Web service is on the same server that hosts the SAP BAPIS/RFCs to which the Web service is bound, or a different server.

     In an SSO configuration, the client provides credentials to Unwired Server (username and password or X.509 user certificate) that are authenticated by the security configuration's authentication module (CertificateAuthenticationLoginModule for X.509 or HttpAuthenticationLoginModule for SSO2). Once authenticated by Unwired

Server, and assuming that Unwired Server and the EIS have a secure communication path, SSO is enabled.

2. (Optional) Configure a data change notification (DCN) port if this is the data refresh policy for any of the MBOs within the package.

See also

- *Enabling Single Sign-on for Mobile Business Object Packages* on page 5
- *Single Sign-on Authentication* on page 39

Single Sign-on for SAP MBO Package Prerequisites

Before implementing SSO for SAP MBO packages, configure the MBOs so client credentials can be propagated to the EIS and, if enabling SSO for a Workflow application, add the appropriate starting point.

Configure the MBO. See these topics in *Sybase Unwired WorkSpace - Mobile Business Object Development*:

- *Propagating a Client's Credentials to the Back-end Data Source*
- *Configuring an MBO to Use an SAP Java Connector* – for SAP MBOs
- *Configuring an SAP Exposed Web Service MBO to Use Credentials*– for SAP function modules exposed as Web services

Configure the Workflow application to use SSO2 or X.509 credentials by adding and configuring a credential starting point for the workflow application. See *Configuring the Workflow Application to Use Credentials* in the *Developer Guide: Mobile Workflow Packages* for details.

Single Sign-on for SAP MBO Package Postrequisites

After configuring SSO for SAP MBO packages on Unwired Server, install certificates on the mobile device and test them.

- Workflow applications – see *Installing and Testing X.509 Certificates on Simulators and Mobile Devices* in the *Developer Guide for Mobile Workflow Packages*.
- Native applications – install and import X.509 certificates and use the Object API to select them for client connections. Refer to your platform's *Developer Guide* for details:
  - *Installing and Testing X.509 Certificates on Simulators and Mobile Devices*
  - *Single Sign-On With X.509 Certificate Related Object API*

Creating Connections and Connection Templates

Create a new connection or connection template that defines the properties needed to connect to a new data source.

1. In the left navigation pane, expand the **Domains** folder, and select the domain for which you want to create a new connection.

2. Select **Connections**.
3. In the right administration pane:
   - To create a new connection – select the **Connections** tab, and click **New**.
   - To create a new connection template – select the **Templates** tab, and click **New**.
4. Enter a unique **Connection pool name** or template name.
5. Select the **Connection pool type** or template type:
   - JDBC – choose this for most database connections.
   - Proxy - choose this if you care connecting to the Online Data Proxy.
   - WS – choose this if you are connecting to a Web Services (SOAP or REST) data source.
   - SAP – choose this if you are connecting to an SAP (JCO) datasource.
6. Select the appropriate template for the data source target from the **Use template** menu. By default, several templates are installed with Unwired Platform; however, a production version of Unwired Server may have a different default template list.
7. Template default properties appear, along with any predefined values. You can customize the template, if required, by performing one of:
   - Editing existing property values – click the corresponding cell and change the value that appears.
   - Adding new properties – click the **<ADD NEW PROPERTY>** cell in the Property column and select the required property name. You can then set values for any new properties you add.

   Note: In a remote server environment, if you edit the sambledb Server Name property, you must specify the remote IP number or server name. Using the value "localhost" causes cluster synchronization to fail.
8. Test the values you have configured by clicking **Test Connection**. If the test fails, either values you have configured are incorrect, or the data source target is unavailable. Evaluate both possibilities and try again.
9. Click **OK** to register the connection pool.
   The name appears in the available connection pools table on the Connections tab. Administrators can now use the connection pool to deploy packages.

Configuring an SAP Java Connector Connection Using SNC to an SAP Server

Create a Java Connection (JCo) to an SAP Server in Unwired Server from Sybase Control Center where SNC is required.

Prerequisites

Start Unwired Server services and log in to Sybase Control Center as the administrator, and download and install the SAP cryptographic libraries.

Task

The SAP JCo connection provides access for various client types, including those that use SSO2 tokens and X.509 certificates.

1. Expand the cluster, expand the **Domains** folder, expand the domain to which the package is to be deployed, and select **Connections**.
2. Select the **Connections** tab and click **New**. Name the connection pool `SAP Server`, select **SAP** as the Connection pool type, select the **SAP SNC template**, and enter appropriate properties for the SAP enterprise information system (EIS) to which you are connecting. For example:
   - Language (jco.client.lang) = `EN`
   - Host name (jco.client.ashost) = `sap-doe-vm1.sybase.com`
   - System number (jco.client.sysnr) = `00`
   - SNC mode (jco.client.snc_mode) = `1`
   - SNC name (jco.snc_myname) = `p:CN=SNCTEST, O=Sybase, L=Dublin, SP=California, C=US`
   - SNC service library path (jco.client.snc_lib) = `C:/sapcryptolib/sapcrypto.dll` (the location of the cryptographic library)
   - Client number (jco.client.client) = `100`
   - SNC partner (jco.client.snc_partnername) = `p:CN=sap-doe-vm1, OU=SUP, O=Sybase, C=US`
   - SNC level (jco.client.snc_qop) = `1`
3. Click **Test Connection** to verify access to the SAP server, and click **OK**.

Generating and Installing a PSE Certificate on Unwired Server

Generate a PSE certificate on Unwired Server to use in testing connections with SAP Systems when using the SAP Cryptographic Library to secure the connection using Secure Network Communications (SNC).

Prerequisites

Download and install the SAP Cryptographic Library.

Task

These instructions describe how to generate an X.509 certificate for testing SAP JCo and single sign-on with SNC only. In a production environment, a different entity controls certificate management. For example, an SAP system administrator controls certificate generation and management for his or her particular environment, including maintaining the certificate list in a Personal Security Environment (PSE) with trust manager.

Note: When the CertificateAuthenticationLoginModule gets a certificate from a client, it can optionally validate that it is a trusted certificate. The easiest way to support validation is to import the CA certificate into the `<UnwiredPlatform_InstallDir>/Servers/`

`UnwiredServer/Repository/Security/truststore.jks` file, which is the default Unwired Server truststore.

Use the SAPGENPSE utility to create a PSE certificate to use for testing. See *http://help.sap.com/saphelp_nw04s/helpdata/en/a6/f19a3dc0d82453e10000000a114084/content.htm*. The basic steps are:

1. Generate the certificate from the SAP Cryptograhpic Library directory. For example,`C:\sapcryptolib`:

   ```
   sapgenpse get_pse <additional_options> -p <PSE_Name> –r
   <cert_req_file_name> -x <PIN> <Distinguished_Name>
   ```

2. Copy the PSE certificate (for example, SNCTEST.pse) to the location of your installed SAP Cryptographic Library. For example, `C:\sapcryptolib`.
3. Generate a credential file (cred_v2) from the `C:\sapcryptolib` directory:

   ```
   sapgenpse seclogin -p SNCTEST.pse -O DOMAIN\your_name_here
   -x password
   ```

   Note: The user generating the certificate must have the same user name as the process (mlserv32.dll or eclipse.exe) under which the Unwired Platform service runs.

SAP Java Connector Properties

Configure SAP Java Connector (JCo) connection properties.

For a comprehensive list of SAP JCo properties you can use to create an instance of a client connection to a remote SAP system, see *http://help.sap.com/javadocs/NW04/current/jc/com/sap/mw/jco/JCO.html#createClient(java.util.Properties)*.

Table 1. General connection parameters

Name	Description	Supported values
Client Number	Specifies the SAP client.	Three-digit client number; preserve leading zeros if they appear in the number
Logon User	Specifies the login user ID.	User name for logging in to the SAP system If using X.509 certificate authentication, remove the JCo properties `jco.client.passwd` and `jco.client.user` defined for the SAP connection profile in Sybase Control Center (SCC).

Name	Description	Supported values
Password	Specifies the login password.	Password for logging in to the SAP system
Language	Specifies a login language.	ISO two-character language code (for example, EN, DE, FR), or SAP-specific single-character language code. As a result, only the first two characters are ever used, even if a longer string is entered. The default is EN.
System Number	Indicates the SAP system number.	SAP system number
Host Name	Identifies the SAP application server.	Host name of a specific SAP application server
Message Server	Identifies the SAP message server.	Host name of the message server
Gateway Host	Identifies the SAP gateway host.	Host name of the SAP gateway Example: `GWHOST=hs0311`
Gateway Service	Identifies the SAP gateway service.	Service name of the SAP gateway Example: `GWSERV=sapgw53`
R/3 Name	Specifies R/3 name.	Name of the SAP system
Server Group	Identifies the group of SAP application servers.	Group name of the application servers
External Server Program	Identifies the program ID of the external server program.	Path and name of the external RFC server program, or program ID of a registered RFC server program Example: `TPNAME=/sap/srfcserv`

Name	Description	Supported values
External Server Program Host	Identifies the host of the external server program. This information determines whether the RFC client connects to an RFC server started by the SAP gateway or to an already registered RFC server. **Note:** If the gateway host and external server program host are different, make sure that the SAP gateway has access to start the server program through a remote shell.	Host name of the external RFC server program Example: `TPHOST=hs0311`
Remote Host Type	Identifies the type of remote host.	2: R/2 3: R/3 E: external
RFC Trace	Specifies whether or not to enable RFC trace.	0: disable 1: enable
Initial Codepage	Identifies the initial code page in SAP notation. A code page is used whenever character data is processed on the application server, appears on the front end, or is rendered by a printer.	Four-digit SAP code page number
Enable ABAP Debugging	Enables or disables ABAP debugging. If enabled, the connection is opened in debug mode and the invoked function module can be stepped through in the debugger. For debugging, an SAP graphical user interface (SAPGUI) must be installed on the same machine the client program is running on. This can be either a normal Windows SAPGUI or a Java GUI on Linux/UNIX systems.	0: no debugging 1: attach a visible SAPGUI and break at the first ABAP statement of the invoked function module

Name	Description	Supported values
Remote GUI	Specifies whether a remote SAP graphical user interface (SAPGUI) should be attached to the connection. Some older BAPIs need an SAPGUI because they try to send screen output to the client while executing.	0: no SAPGUI 1: attach an "invisible" SAPGUI, which receives and ignores the screen output 2: attach a visible SAPGUI For values other than 0 a SAPGUI needs to be installed on the machine, where the client program is running. This can be either a Windows SAPGUI or a Java GUI on Linux/Unix systems.
Get SSO Ticket	Generates an SSO2 ticket for the user after login to allow single sign-on. If RfcOpenConnection() succeeds, you can retrieve the ticket with RfcGetPartnerSSOTicket() and use it for additional logins to systems supporting the same user base.	0: do not generate SSO2 ticket 1: generate SSO2 ticket
Use Cookie Version 2	Indicates whether or not to use the specified SAP Cookie Version 2 (SSO2) as the login ticket instead of user ID and password.	User: $MYSAPSSO2$ Password: Base64-encoded ticket Login with single sign-on is based on secure network connection (SNC) encryption and can only be used in combination with an SNC.
Use X509	Indicates whether or not to use the specified X509 certificate as the login certificate instead of user ID and password.	User: $X509CERT$ Password: Base64-encoded ticket Login with X509 is based on secure network connection (SNC) encryption and can only be used in combination with an SNC.

Name	Description	Supported values
Logon Check	Enables or disables login check at open time.	0: disable 1: enable If you set this to 0, RfcOpenConnection() opens a network connection, but does not perform the login procedure. Therefore, no user session is created inside the back-end system. This parameter is intended only for executing the function module RFC_PING.
Additional GUI Data	Provides additional data for graphical user interface (GUI) to specify the SAProuter connection data for the SAPGUI when it is used with RFC.	/H/ *`router string`* : the entire router string for the SAPGUI /P/ *`password`* : specify this value if the password for the SAPGUI connection is not the same as the password for the RFC connection.
GUI Redirect Host	Identifies which host to redirect the remote graphical user interface to.	Host name
GUI Redirect Service	Identifies which service to redirect the remote graphical user interface to.	Name of the service
Remote GUI Start Program	Indicates the program ID of the server that starts the remote graphical user interface.	Program ID of the server
SNC Mode	Enables or disables secure network connection mode.	0: off 1: on
SNC Partner	Identifies the secure network connection partner.	Secure network connection name of the application server (for example, p:CN=R3, O=XYZ-INC, C=EN)

Name	Description	Supported values
SNC Level	Specifies the secure network connection security level.	1: digital signature 2: digital signature and encryption 3: digital signature, encryption, and user authentication 8: default value defined by backend system 9: maximum value that the current security product supports
SNC Name	Indicates the secure network connection name. This property overrides the default secure network connection partner.	Token or identifier representing the external RFC program
SNC Service Lib Path	Identifies the path to the SAP crytpographic library that provides secure network connection service.	Full path and name of third-party security library. You must download and install the library from the SAP Service Marketplace.
R/2 Destination	Identifies a configured R/2 system defined in the sideinfo configuration.	
Logon ID	Defines the string for SAPLOGON on 32-bit Windows.	String key to read parameters from the saplogon.ini file created by the SAPLogon GUI program on Windows
External Authentication Data	Provides data for external authentication (PAS). This is an old login mechanism similar to SSO; Sybase recommends that you do not use this approach.	
External Authentication	Specifies type of external authentication (PAS). See External Authentication Data property.	

Web Services Properties
Configure connection properties for the Simple Object Access Protocol (SOAP) and Representational State Transfer (REST) architectures.

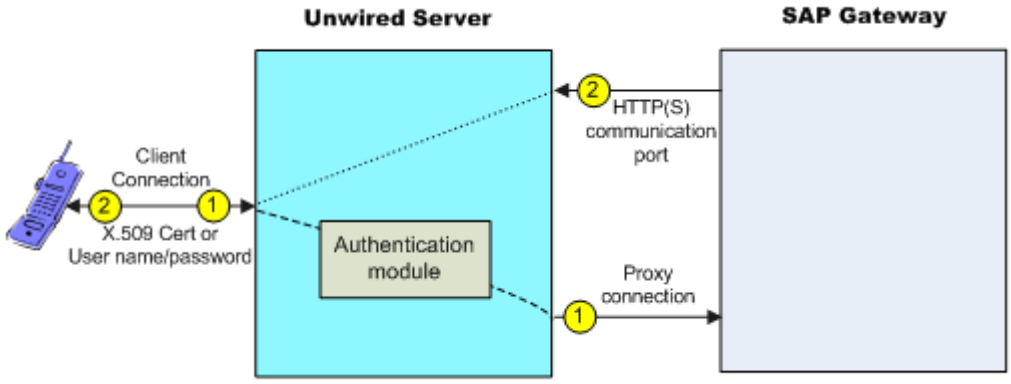
Name	Description	Supported values
Password	Specifies the password for HTTP basic authentication, if applicable.	Password
Address	Specifies a different URL than the port address indicated in the WSDL document at design time.	HTTP URL address of the Web service
User	Specifies the user name for HTTP basic authentication, if applicable.	User name
Certificate Alias	Sets the alias for the Unwired Platform keystore entry that contains the X.509 certificate for Unwired Server's SSL peer identity. If you do not set a value, mutual authentication for SSL is not used when connecting to the Web service.	Use the alias of a certificate stored in the Unwired Server certificate keystore.

SAP Single Sign-on and Online Data Proxy Overview
Understand how OData applications fit in the Unwired Platform landscape and learn how to secure communication paths and enable single sign-on (SSO) for these applications.

The proxy connector is the online data proxy (ODP) connector between OData applications and the SAP Gateway, and uses an HTTP(S) connection from Unwired Server to the SAP Gateway. A separate HTTP(S) port is used by the SAP Gateway to push changes through Unwired Server to the OData application. Unwired WorkSpace is not used to create MBOs, generate code, create applications, or for deployment. Instead, in OData-based mobile applications that run in Unwired Server:

- Applications are developed using the OData SDK.
- The SAP Gateway/enterprise information system (EIS) is responsible for data federation and content management.
- OData applications are message based – the SAP Gateway performs queue handling, data caching, and is push-enabled to push data changes out to Unwired Server, which in turn pushes these changes to the physical devices.

Unwired Server acts as a pass-through server for OData-based applications:



1. An OData client application registers with Unwired Server and subscribes to push notifications from the SAP Gateway. Unwired Server forwards the subscription request to the SAP Gateway. The SAP Gateway stores the subscription request for the collection with the push delivery address (HTTP(S) SSL Port).

   In an SSO configuration, the client provides credentials to Unwired Server (user name and password or X.509 user certificate) that are authenticated by the security configuration's authentication module (CertificateAuthenticationLoginModule for X.509 or HttpAuthenticationLoginModule for SSO2). Once authenticated by Unwired Server, and assuming that Unwired Server and the SAP Gateway have a secure communication path, SSO is enabled.
2. When application data changes in SAP and determines that a particular client has a subscription to that change, the Gateway connects to the Unwired Server HTTP(S) port and sends a message identifying the client, along with the message payload. Unwired Server looks up the client and queues the message. If the client is connected, the message is delivered immediately. If the client is offline, then Unwired Server attempts to send a push notification to the client (BES HTTP Push for Blackberry, APNS notification for iOS) to attempt to wake up the client and have it retrieve the messages.

See also

- *Single Sign-on Authentication* on page 39
- *Enabling Single Sign-on for OData Applications* on page 6

Preparing the SAP Gateway

Configure the SAP Gateway to push OData application data to Unwired Server, including configuring the RFC destination for HTTPS on the Gateway.

1. Log on to a Gateway system and go to transaction **sm59**.
2. Create a RFC connection of type **G**.

3. Enter the domain name of Unwired Server (CN of the Unwired Server certificate) in the **Target Host** field .
4. Enter `/GWC/SUPNotification` in the **Path Prefix** field.
5. Enter 8004 in the **Service No.** field.
6. Select the **Logon & Security** tab.
7. Under **Security Options**, click the SSL **Active** option.
8. Select **Default SSL Client (Standard)** from the **SSL Certificate** drop-down list.
9. Click **Save**, then **Connection Test**.

   The test should be successful, a HTTP OK success message displays.

Note: Change the push endpoint in the proxy property of the application templates to https://<domain name of the SUP server>:<SSL Port>/GWC/SUPNotification/. In this example, 8004 is the SSL port to which the Gateway pushes data changes:

```
https://inln50089324a.dhcp.blrl.sap.corp:8004/GWC/
SUPNotification/
```

Preparing Your SAP Environment for Single Sign-on

Verify that the SAP® enterprise information system (EIS) is configured correctly to accept SSO connections from Unwired Server.

The general steps for enabling SAP systems to communicate with Unwired Server over secure communication paths are:

1. Set all parameters for the type of credentials accepted by the server:
   - SSO2 token – verify everything is set properly with the SSO2 transaction.
   - X.509 certificate – set up, import, and verify certificates using the Trust Manager (transaction STRUST).
2. Use the ICM configuration utility to enable the ICM HTTPS port.
3. Set the type of authentication to enable over HTTPS:
   - Server authentication only – the server expects the client to authenticate itself using basic authentication, not SSL
   - Client authentication only – the server requires the client to send authentication information using SSL certificates. The ABAP stack supports both options. Configure the server to use SSL with client authentication by setting the ICM/HTTPS/verify_client parameter:
     - 0 – do not use certificates.
     - 1 – allow certificates (default).
     - 2 – require certificates.
4. Use the Trust Manager (transaction STRUST) for each PSE (SSL server PSE and SSL client PSE) to make the server's digitally signed public key certificates available. Use a public key infrastructure (PKI) to get the certificates signed and in the SAP system.

   There are no SSO access restrictions for MBO data that span multiple SAP servers.

See *https://cw.sdn.sap.com/cw/docs/DOC-10618* for information about the SAP Trust Manager.

5. To enable secure communication, Unwired Server and the SAP server it communicates with must exchange valid CA X.509 certificates. Deploy these certificates, which are used during the SSL handshake with the SAP server into the Unwired Server truststore.
6. The user identification (distinguished name), specified in the certificate must map to a valid user ID in the AS ABAP, which is maintained by the SM30 view (VUSREXTID).

See *Configuring the AS ABAP for Supporting SSL* at *http://help.sap.com/saphelp_aii710/helpdata/en/49/23501ebf5a1902e10000000a42189c/frameset.htm*

Security Configurations That Implement Single Sign-on Authentication

Use the CertificateAuthenticationLoginModule authentication module to implement X.509 authentication or HttpAuthenticationLoginModule to implement SSO2.

Creating and Assigning a Security Configuration That Uses SSO2 Tokens

Create a new security configuration, assign the HttpAuthenticationLoginModule authentication provider to it, and assign the security configuration to an Unwired Server domain or package.

The HttpAuthenticationLoginModule authentication provider supports SSO2 token logins to SAP systems through JCo and Web service connections, DOE-C packages, and other packages that require token authentication.

1. Create the new security configuration:
   a) From Sybase Control Center, select **Security**.
   b) Select the **General** tab, click **New**, and enter a name for the new security configuration, for example, `SAPSSOSECADMIN`. Click **OK**.
2. Configure the new security configuration:
   a) Select the **SAPSSOSECADMIN** security configuration.
   b) Select the **Authentication** tab.
   c) Click **New** and select **HttpAuthenticationLoginModule** as the Authentication provider. Set the SAP server URL, the SSO cookie name (typically set to `MYSAPSSO2`), and other properties as appropriate for the connection.
3. Select the **General** tab, and click **Validate** to confirm that Unwired Server accepts the new security configuration.
   A message indicating the success of the validation appears above the menu bar.
4. Click **Apply** to save changes to the security configuration, and apply them across Unwired Server.
   A message indicating the success of the application appears above the menu bar.
5. Assign the SAPSSOSECADMIN security configuration to an Unwired Server domain. This example uses the default domain, but you can specify any domain to which the package is deployed:

a) Expand the **Domains** folder and select **default**.
b) Select the **Security Configurations** tab and click **Assign**.
c) Select **SAPSSOSECADMIN** and click **OK**.

6. Select the **Security Configurations** tab, and remove any other security configurations for the domain.

Creating and Assigning a Security Configuration That Uses X.509 Credentials

Create a new security configuration, assign the CertificateAuthenticationLoginModule authentication provider to it, and assign the security configuration to an Unwired Server domain or package.

The CertificateAuthenticationLoginModule authentication provider supports X.509 certificate logins to SAP systems through JCo, DOE-C, Online Data Proxy, and Web service connections. You can assign security configurations to domains,packages, or applications.

1. Create the new security configuration:
   a) From Sybase Control Center, select **Security**.
   b) Select the **General** tab, click **New**, and enter a name for the new security configuration, for example, `X509SECADMINCERT`. Click **OK**.
2. Configure the new security configuration:
   a) Expand the Security folder.
   b) Select the **X509SECADMINCERT** security configuration.
   c) Select **Authentication**.
   d) Select **New**.
   e) Select **com.sybase.security.core.CertificateAuthenticationLoginModule** as the Authentication provider.
   f) Click **OK** to accept the default settings, or modify any of these settings as required:
      - Click **<Add New Property>**, select **Validate Certificate Path** and set the value to **true**.
      - If more than one truststore is defined in Unwired Server, click **<Add New Property>**, select **Trusted Certificate Store** and set the value to the location of the Java truststore that contains the Unwired Server trusted CA certificates. Otherwise, the default Unwired Server truststore is used.
      - If you change the default password for the truststore, click **<Add New Property>**, select **Trusted Certificate Store Password** and set the value of the truststore password.
   g) Click **OK**.
3. Select the **General** tab, select **Validate**, then **Apply**.
4. Assign the X509SECADMINCERT security configuration to an Unwired Server domain. This example uses the default domain, but you can specify any domain to which the package is deployed:

a) Expand the **Domains** folder and select **default**.
b) Select the **Security Configurations** tab and click **Assign**.
c) Select **X509SECADMINCERT** and click **OK**.

5. Select the **Security Configurations** tab, and remove any other security configurations for the domain.

Creating Security Profiles to Enable Mutual Authentication for SAP

Create security profiles and associate them with X.509 server certificates that can be used to establish secure connections between a client, Unwired Server, and the SAP EIS.

Prerequisites

- Your SAP system must be configured for HTTPS mutual authentication
- Import the third party's private-key certificate used by Unwired Server to mutually authenticate the client into the Unwired Server keystore:
  - SUPServer certificate – represents the certificate used to secure an HTTPS connection between Unwired Server and SAP Server or other enterprise information system (EIS), where data and information flow from Unwired Server to the EIS, which could be a DOE-C, Web Service, or Proxy connection.
  - SAPServer certificate – represents the certificate used to secure the communication path between the SAP Server or EIS and Unwired Server, where data and information flow from the EIS to Unwired Server on an HTTPS port (8001, 8002, and so on), which are made available to the EIS for pushing data to Unwired Server. For SAP Servers, this could be NetWeaver/DOE (TechnicalUser), or the SAP Gateway.

Task

1. Create a SAPServer security profile :
   a) From Sybase Control Center, expand **Servers** > **<UnwiredServerName>**, and select **Server Configuration**.
   b) Select **General** > **SSL Configuration**.
   c) Select **<ADD NEW SECURITY PROFILE>**.
   d) Enter these values:
      - Security profile name – for example, `TechnicalUser` for NetWeaver/DOE connections or `Proxy` for SAP Gateway connections.
      - Certificate Alias – the case sensitive certificate alias you defined when you imported the certificate into the keystore. For example, `doetech`, `proxy` (or whatever value you set the alias to using the **keytool -alias** option)
      - Authentication: `strong_mutual`
   e) If you imported the user and CA certificates into keystore or truststore locations other than the default, make sure the paths and passwords reflect them.
   f) Select **Save**.

2. Create a SUPServer security profile:
   a) From Sybase Control Center, expand **Servers** > **<UnwiredServerName>**, and select **Server Configuration**.
   b) Select **General** > **SSL Configuration**.
   c) Select **<ADD NEW SECURITY PROFILE>**.
   d) Enter these values:
      - Security profile name: `SUPServer`
      - Certificate Alias: `SUP` (or whatever value you set the alias to using the **keytool -alias** option)
      - Authentication: `strong_mutual`
   e) If you imported the user and CA certificates into keystore or truststore locations other than the default, make sure the paths and passwords reflect them.
   f) Select **Save**.
3. Restart Unwired Server.

Enabling the HTTPS Port and Assigning the SUPServer Security profile

Enable an HTTPS port for secure communication between Unwired Server and the SAP EIS.

For DOE-C and SAP Gateway, this is the port to which the DOE or Gateway connects to Unwired Server and sends a message identifying the client, along with the message payload.

1. From Sybase Control Center, expand **Servers** > **<UnwiredServerName>**, and select **Server Configuration**.
2. Select **General** > **Communication Ports**.
3. Expand **Hide secure data change notification ports**.
4. Select a port number and enter:
   - Status: `enabled`
   - SSL Security Profile: `SUPServer` or whatever you named the security profile associated with the SUP Server user certificate.

   Note: You can add a new HTTPS port if you do not want to use 8001 or 8002. For Gateway connections, the port must match that of the port you defined in the Gateway. For example 8004. See *Preparing the SAP Gateway*.
5. Select **Save**.
6. Restart Unwired Server.

Distributing Single Sign-on Related Files in an Unwired Server Cluster

(Not applicable to Online Data Proxy) Place required files in the appropriate primary Unwired Server subdirectory so they are distributed to all Unwired Servers within the cluster during cluster synchronization.

Any changes to a named security configuration affect the cluster and trigger a cluster synchronization, which automatically zips the files in the primary Unwired Server `CSI`

subdirectory and distributes them to the other servers in the cluster. Copy all certificate and other security-related files to the `CSI` subdirectory.

The provider configuration information, which includes the server certificate file name and location, must be the same on all cluster nodes. The same is true for the cryptographic DLLs and certificate files for SSO using X509.

1. On the primary server in the cluster, put any SAP certificate files or truststores into the `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer \Repository\CSI\conf` directory.

   Use system properties to specify the full path and location of the file in the configuration so they can be accessed from different servers within the cluster if installation directories are different from that of the primary server. For example:

   ```
   ${djc.home}/Repository/CSI/conf/
   SNCTEST.pse
   ```

   For X.509 CertificateAuthenticationLoginModule, if the *ValidateCertificatePath* is set to true, the default, the CA certificate (or one of its parents) must be installed in the truststore for each server.

   Note: Unwired Server truststore and keystore files:
   - `<UnwiredPlatform_InstallDir\Servers\UnwiredServer \Repository\Security\truststore.jks` – is the Unwired Server trust store that contains CA (or parent) certificates. Unwired Server trusts all CA or parent certificates in `truststore.jks`.
   - `<UnwiredPlatform_InstallDir\Servers\UnwiredServer \Repository\Security\keystore.jks` – contains client certificates only.

   The CertificateAuthenticationLoginModule also has `Trusted Certificate Store*` and `Store Password` properties which you can to keep the module out of the default Unwired Server trust store. You must first:

   a) Use **keytool** to put the CA certificate into a new keystore.
   b) Put the keystore into the `Repository\CSI\conf` subdirectory.
   c) Include the path in the `Trusted Certificate Store` property.
2. From Sybase Control Center, add the login module.
3. Restart all Unwired Servers within the cluster.

SAP External Libraries Overview

Understand the purpose of the external files you can optionally download from SAP and install into Unwired Platform to enable communication with an SAP EIS.

- **SAP Cryptographic Libraries –** required by Unwired Platform to enable Secure Network Communications (SNC) between Unwired Server or Unwired WorkSpace and the SAP EIS.

- **SAPCAR utility –** required to extract files from the SAP cryptographic library.

Installing the SAP Cryptographic Libraries on Unwired Platform

Installation and configuration is required if you want to configure Secure Network Communications (SNC) for Unwired Platform SAP JCo connections. SNC may be required by the SAP EIS in question, if SSO2 tokens or X.509 certificates are used for connection authentication.

Prerequisites

Download and install the SAPCAR utility, which is required to extract the contents of the cryptographic library.

Task

Unzip and install the contents of the latest SAP Cryptographic archive on your Unwired Server host. There are different distribution packages for various hardware processors.

Make sure you are installing the correct libraries for your environment, and into folders based on the particular architecture of your machine.

1. Go to the SAP Web site at *http://service.sap.com/swdc* and download the latest SAP cryptographic library suitable for your platform.
   a) Navigate to **Installations and Upgrades** > **Browse our Download Catalog** > **SAP Cryptographic Software** > **SAP Cryptographic Software**.
   b) Select and download the platform specific file.
2. Create a directory in which you unzip the Cryptographic zip file. For example: `C:\sapcryptolib`.
3. Copy the appropriate Windows cryptographic library for your machine (for example, 90000101.SAR) to the `C:\sapcryptolib` directory.
4. Open a command prompt and navigate to `C:\sapcryptolib`.
5. Extract the SAR file. For example:

   SAPCAR_4-20002092.EXE -xvf C:\90000101.SAR -R C:\sapcryptolib
6. Depending on your platform:
   - 64-bit – delete the `nt-x86_64` directory from the `C:\sapcryptolib` directory. Copy the contents of the `nt-x86_64` subdirectory to `C:\sapcryptolib`.
   - 32-bit – delete the `ntia64` and `nt-x86_64` directories from the `C:\sapcryptolib` directory. Copy the contents of the `ntintel` subdirectory to `C:\sapcryptolib`. Delete the `ntintel` subdirectory.
7. Add the SECUDIR environment variable based on machine type and Unwired Platform component:
   - 64-bit Unwired Server – requires access to the 64-bit crypto libraries. Set SECUDIR in `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers`

`\UnwiredServer\bin\usersetenv.bat` to point to the location of the 64-bit crypto libraries. For example:
`set SECUDIR=C:\sapcryptolib`

- 32-bit Unwired Server – requires access to the 32-bit crypto libraries. Set SECUDIR in `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers \UnwiredServer\bin\usersetenv.bat` to point to the location of the 32-bit crypto libraries. For example:
  `set SECUDIR=C:\sapcryptolib`
- Unwired WorkSpace – runs in 32-bit mode regardless of machine type and requires access to the 32-bit crypto libraries. Set SECUDIR in `C:\Sybase \UnwiredPlatform\Eclipse\UnwiredWorkSpace.bat` above the line SUP_ROOT=<SUP_HOME>. For example:
  `set SECUDIR=C:\sapcryptolib`

Installing the SAPCAR Utility

Unzip and install the latest SAPCAR utility on your Unwired Server or Unwired WorkSpace host, which you can use to extract the contents of compressed SAP files, for example RFC and cryptographic library files.

The installation package is available to authorized customers on the SAP Service Marketplace. There are different distribution packages for various hardware processors. Select the package appropriate for your platform.

1. Go to the SAP Web site at *http://service.sap.com/swdc*.
2. From the SAP Download Center, navigate to **Support Packages and Patches > Browse our Download Catalog > Additional Components**
3. Select **SAPCAR**.
4. Select the current version. For example, **SAPCAR 7.10**, then select and download the SAPCAR appropriate for your platform.

Creating a Security Configuration for Device Users

Create and name a set of security providers and physical security roles to protect Unwired Platform resources. For device user authentication, create at least one provider that is not the "admin" security configuration on the "default" domain, which is used exclusively for administrator authentication in Sybase Control Center.

Only platform administrators can create security configurations. Domain administrators can view configurations only after the platform administrator creates and assigns them to a domain.

1. In the left navigation pane, expand the **Security** folder.
2. In the right administration pane, click **New**.
3. Enter a name for the security configuration and click **OK**.

Assigning Providers to a Security Configuration

Assign providers after you have created a security configuration.

1. In the left navigation pane, expand the **Security** folder.
2. Select the security configuration you want to assign a provider to.
3. In the right administration pane, select the **Settings** tab to set an authentication cache timeout value.

   The timeout determines how long authentication results should be cached before a user is required to reauthenticate. For details, see *Authentication Cache Timeouts*. To configure this value:

   a) Set the cache timeout value in seconds. The default is 3600.
   b) Click **Save**.
4. Select the tab corresponding to the type of security provider you want to configure: Authentication, Authorization, Attribution, or Audit.
5. To edit the properties of a preexisting security provider in the configuration:
   a) Select the provider, and click **Properties**.
   b) Configure the properties associated with the provider by setting values according to your security requirements. Add properties as documented in the individual reference topics for each provider.
   c) Click **Save**.
6. To add a new security provider to the configuration:
   a) Click **New**.
   b) Select the provider you want to add.
   c) Configure the properties associated with the provider by setting values according to your security requirements. Add properties as documented in the individual reference topics for each provider.
   d) Click **OK**.
   The configuration is saved locally, but not yet committed to the server.
7. Select the **General** tab, and click **Validate** to confirm that Unwired Server accepts the new security configuration.
8. Click **Apply** to save changes to the security configuration, and apply them across Unwired Server.

Next

If you have multiple providers, understand how to stack/sequence them and know what the implication of provider order means.

See also

- *Assigning Security Configurations to Domains, Packages, or Applications* on page 75

Authentication Cache Timeouts

Set a cache timeout value to cache user authentication credentials, which improves runtime performance.

Set timeout properties to avoid repeatedly reauthenticating users— a benefit of particular interest for device clients that receive separately authenticated messages. For example, if a user logs in successfully, he or she can reauthenticate with the same credentials without validating them against a security repository. However, if the user provides a user name or password that is different from the ones cached, Unwired Server delegates the authentication request to the security repository.

This property affects only authentication results:

- The authentication status pass or fail is cached.
- If authentication passes, user roles are assigned.

Authorization results are not cached.

For example, if an MBO is protected by "LogicalRoleA", and the security configuration that MBO is deployed in has a role mapping to "PhysicalRoleA", each time a user tries to access this MBO, the provider checks to see if they are in PhysicalRoleA based on cached role membership from the original authentication. It does not check the security repository each time thereafter.

By default, the cache timeout value is 3600 (seconds). This value is enabled whether the property exists or not. You can change this value by configuring a new value for the property in Sybase Control Center for the appropriate security configuration.

Enabling CRLs

Identify the certificate revocation lists (CRLs) that define a list of digital certificates which have been revoked. Revoked certificates should not give the Unwired Platform device user access to the Unwired Server runtime.

Administrators can configure certificate revocation lists (CRLs) to check if any of the certificates in the path are revoked. A series of URI's define the CRL location.

1. Using Sybase Control Center, open the CertificateAuthenticationLoginModule use by your security configuration.
2. For the CRL property, define one or more URIs. If using multiple URIs, each must be indexed.

   The index number used determines the order in which CLRs are checked. This example uses two URI, each indexed accordingly so that the Verisign CRL comes first.

   ```
   crl.1.uri=http://crl.verisign.com/
   ThawtePersonalFreemailIssuingCA.crl
   crl.2.uri=http://crl-server/
   ```

Built-in Security Providers

Unwired Server supports a variety of built-in security providers. Administrators define one or more security providers when they create security configurations using Sybase Control Center.

You can configure a provider of a given type only if that provider is available on the enterprise network.

If you are using Unwired Platform in an Online Data Proxy deployment, not all providers are applicable in this environment.

No Security Provider

A NoSec provider offers pass-through security for Unwired Server, and is intended for use in development environments or for deployments that require no security control. Do not use this provider in production environments— either for administration, or device user authentication.

If you use NoSec providers, all login attempts succeed, no matter what values are used for the user name and password. Additionally, all role and control checks based on attributes also succeed.

Sybase provides these classes to implement the NoSec provider:

- **NoSecLoginModule** – provides pass-through authentication services.
- **NoSecAttributer** – provides pass-through attribution services.
- **NoSecAuthorizer** – provides pass-through authorization services.

LDAP Security Provider

(Not applicable to Online Data Proxy) The LDAP security provider includes authentication, attribution, and authorization providers.

You can configure these providers:

- **LDAPLoginModule**– provides authentication services. Through appropriate configuration, you can enable certificate authentication in **LDAPLoginModule**.
- (Optional)**LDAPAuthorizer** or **RoleCheckAuthorizer** – provide authorization services for **LDAPLoginModule**. **LDAPLoginModule** works with either authorizer. In most production deployments, you must always configure your own authorizer. However, if you are authenticating against a service other than LDAP, but want to perform authorization against LDAP, you can use the **LDAPAuthorizer**.

  The RoleCheckAuthorizer is used with every security configuration but does not appear in Sybase Control Center.

  Use **LDAPAuthorizer** only when **LDAPLoginModule** is not used to perform authentication, but roles are still required to perform authorization checks against the LDAP data store. If you use **LDAPAuthorizer**, always explicitly configure properties; for it cannot share the configuration options specified for the **LDAPLoginModule**.

You need not enable all LDAP providers. You can also implement some LDAP providers with providers of other types. If you are stacking multiple LDAP providers, be aware of, and understand the configuration implications.

See also

- *LDAP Configuration Properties* on page 101
- *Adding an LDAP Provider to the Admin Security Configuration* on page 23
- *Stacking Providers and Combining Authentication Results* on page 72

Configuration Best Practices for Multiple LDAP Trees

Use the Unwired Platform administration perspective to configure LDAP authentication and authorization security providers, which are used to locate LDAP user information when organizational user groups exist within multiple LDAP trees.

To accommodate an LDAP tree structure that cannot be directly accessed using one search base:

- Create an LDAP authentication module for each level in the hierarchy – during the authentication process, Unwired Platform tries to authenticate against every login module in the ordered list until authentication succeeds or until it reaches the end of the list. Depending on the number of login modules you configure, this approach may have some performance issues.
- Use different AuthenticationScopes for performing user searches – specify the root node of a particular LDAP tree, by entering `AuthenticationSearchBase="dc=sybase, dc=com"` and `set Scope=subtree`. Unwired Platform performs an LDAP query against the entire subtree for authentication and authorization information. Depending on the number of AuthenticationScope within the LDAP tree structure, this approach can have performance implications.
- If multiple servers are clustered together to form a large logical directory tree, configure the LDAPLoginModule by setting the `Referral` property to `follow`.
- If subjects have been made members of too many LDAP groups and the search for physical roles results in too many results, the maximum result limit may be reached and authentication fails. To avoid this, narrow the `RoleSearchBase` to LDAP groups that are relevant only to Unwired Platform.

LDAP Role Computation

Role checks are the primary means of performing access control when using LDAP authentication. Authentication and attribution capabilities both utilize role computation techniques to enumerate roles that both authenticated users have.

There are three distinct types of role constructs supported by LDAP providers; each may be used independently, or all three may be configured to be used at the same time.

- User-level role attributes, specified by the UserRoleMembershipAttributes configuration property, are the most efficient role definition format. A user's roles are enumerated by a

read-only directory server-managed attribute on the user's LDAP record. The advantage to this technique is the efficiency with which role memberships can be queried, and the ease of management using the native LDAP server's management tools. These constructs are supported directly by ActiveDirectory products, and use these configuration options:

- UserRoleMembershipAttributes – the multivalued attribute on the user's LDAP record that lists the role DNs that the user is a member of. An example value for this property is "memberOf" on ActiveDirectory.
- RoleSearchBase – the search base under which all user roles are found, for example, "ou=Roles,dc=sybase,dc=com". This value may also be the root search base of the directory server.
- RoleFilter – the search filter that, coupled with the search base, retrieves all roles on the server.
- (Optional) RoleScope – enables role retrieval from subcontexts under the search base.
- (Optional) RoleNameAttribute –choose an attribute other than "cn" to define the name of roles.

These properties retrieve correct values automatically based on the type of server you configure.

- LDAP group role definitions may be used as role definitions. This is a common construct among older LDAP servers, but is supported by nearly all LDAP servers. You may select this approach to use the same LDAP schema across multiple LDAP server types. Unlike the user-level role attributes, LDAP group memberships are stored and checked on a group-by-group basis. Each defined group, typically of objectclass "groupofnames" or "groupofuniquenames," has an attribute listing all of the members of the group. The configuration settings used are the same as for user-level role attributes, except for the RoleMemberAttributes property, which replaces the UserRoleMembershipAttributes property. This property defines a comma-delimited list of attributes that contain the members of the group. An example value for this property is "uniquemember,member," which represents the membership attributes in the above-mentioned LDAP objectclasses.
- Freeform role definitions are unique in that the role itself does not have an actual entry in the LDAP database. A freeform role starts with the definition of one or more user-level attributes. When roles are calculated for a user, the collective values of the attributes (each of which may can be multivalued) are added as roles to which the user belongs. This technique is particularly useful when the overhead of managing roles uses are administration-heavy. For example, assign a freeform role definition that is equivalent to the department number of the user. A role check performed on a specific department number is satisfied only by users who have the appropriate department number attribute value. The only property that is required or used for this role mapping technique is the comma-delimited UserFreeformRoleMembershipAttributes property.

LDAP Login and Authorization Modules

LDAP login and authorization modules can sometimes share a common configuration. However, authorizers do not inherit configuration from login modules you configure. Configurations must be explicit.

In the case where both LDAPLoginModule and LDAPAuthorizer are configured:

- Matching configuration – LDAPAuthorizer simply skips the role retrieval.
- Differing configuration – LDAPAuthorizer proceeds with the role retrieval from the configured backend, and performs the authorization checks using the complete list of roles (from both the login module and itself). Even in the case where multiple LDAPLoginModules are configured, only one LDAPAuthorizer is required as it compares its configuration with the configuration used for the successful authentication of the user.

NTProxy Security Provider

(Not applicable to Online Data Proxy) NTProxy (sometimes known as native Windows login) integrates with existing Windows login security. Users can authenticate with their native Windows user name and password, which gives them access to roles that are based on their existing Windows memberships.

The NTProxy provider fulfills authentication services only with classes in `csi-nativeos.jar`; role-based access control and attribution are not directly supported. Groups are also not supported in NTProxy. Instead, group memberships are transformed into a role of the same name and can be mapped in Sybase Control Center.

SAP SSO Token Security Provider

The SAPSSOTokenLoginModule has been deprecated and will be removed in a future release. Use HttpAuthenticationLoginModule for SAP SSO2 token authentication.

Use HttpAuthenticationLoginModule for both JCo and DOE-C connections to the SAP system. Unwired Server does not provide authorization control or role mappings for user authorization; enforce any access control policies in the SAP system.

See also

- *SAP SSO Token Authentication Properties* on page 112
- *Certificate Authentication Properties* on page 110
- *HTTP Basic Authentication Properties* on page 116

Certificate Security Provider

Use the Unwired Server CertificateAuthenticationLoginModule authentication provider to implement SSO with an SAP enterprise information system (EIS) with X.509 certificates.

Unwired Server does not provide authorization control or role mappings for user authorization; enforce any access control policies in the EIS.

See also

- *SAP SSO Token Authentication Properties* on page 112
- *Certificate Authentication Properties* on page 110
- *HTTP Basic Authentication Properties* on page 116

HTTP Basic Security Provider

Use an HTTP Basic provider to enable automatic application registration. This provider is required when registration is set to automatic, and will be used for integrating with third-party security providers like SiteMinder.

A LoginModule validates standard username/password style credentials by passing them to a Web server. Configure the URL property to point to a Web server that challenges for Basic authentication.

Best practice guidelines include:

- Using an HTTPS URL to avoid exposing credentials.
- If the Web server's certificate is not signed by a well known CA, import the CA certificate used to sign the Web server's certificate into the Unwired Server `truststore.jks`. The truststore is prepopulated with CA certificates from reputable CAs.
- If this Web server returns a cookie as part of successful authentication, set the `SSO Cookie Name` configuration property to the name of this cookie. Upon successful authentication, this login module places the cookie value into an `HttpSSOTokenCredential` object and attaches it to the `java.security.Subject` as a public credential.

  Note: The HTTP Basic login module is the module that can either be used for SSO tokens or HTTP basic without SSO. The sole condition being that the backend support HTTP Basic authentication.
- When using this module in lieu of the deprecated SAPSSOTokenLoginModule, the cookie name is typically "MYSAPSSO2".

For example, SiteMinder is often used in mobile deployments to protect existing Web-based applications. Existing users point their browser at a URL, and SiteMinder intercepts an unauthenticated session to challenge for credentials (Basic). When the authentication succeeds, it returns a SMSESSION cookie with a Base64-encoded value that can be used for SSO into other SiteMinder enabled systems.

See also

- *SAP SSO Token Authentication Properties* on page 112
- *Certificate Authentication Properties* on page 110
- *HTTP Basic Authentication Properties* on page 116

Preconfigured User Login Security Provider

If you are accessing Sybase Control Center for the first time you are authenticated as the Unwired Platform administrator with the PreconfiguredLoginModule. You must supply the password defined during the installation of Unwired Platform.

During installation,

Note: Do not forget this installation password. The installer hashes the password with a SHA-256 algorithm before it is saved as part of the PreconfiguredLoginModule configuration. This value cannot be returned to cleartext once the installer hashes it.

Once logged in, the Unwired Platform administrator immediately reconfigures the "admin" security configuration to replace this provider with an production-grade security provider like LDAP. Once complete, the corresponding changes then need to be implemented in security configuration files used by Sybase Control Center. Failure to coordinate these changes can result in unpredictable behavior in Sybase Control Center. If you configure a new provider in Unwired Platform and Sybase Control Center and login fails, review possible login failure solutions *Troubleshooting* guide.

See also

- *Preconfigured User Authentication Properties* on page 115
- *Adding an LDAP Provider to the Admin Security Configuration* on page 23
- *Configuring a Provider to Authenticate Sybase Control Center Logins* on page 27

Stacking Providers and Combining Authentication Results

(Not applicable to Online Data Proxy) Implement multiple LoginModule to provide a security solution that meets complex security requirements.

Each provider has a controlFlag attribute that controls overall behavior when you enable multiple providers. Set the controlFlag to refine how results are processed.

1. Use Sybase Control Center to create a security configuration and add multiple providers as required for authentication.
2. Order multiple providers by selecting a LoginModule and using the up or down arrows at to place the provider correctly in the list.

   The order of the list determines the order in which authentication results are evaluated.
3. For each provider:
   a) Select the provider name.
   b) Click **Properties**.
   c) Configure the controlFlag property with one of the available values: required, requisite, sufficient, optional.

      See *controlFlag Attribute Values* for descriptions of each available value.
   d) Configure any other common security properties as required.
4. Click **Save**.
5. Select the **General** tab, and click **Apply**.

For example, say you have sorted these login modules in this order and used these controlFlag values:

- LDAP (required)
- NT Login (sufficient)
- SSO Token (requisite)
- Certificate (optional)

The results are processed as indicated in this table:

Provider	Authentication Status							
LDAP	pass	pass	pass	pass	fail	fail	fail	fail
NT Login	pass	fail	fail	fail	pass	fail	fail	fail
SSO Token	*	pass	pass	fail	*	pass	pass	fail
Certificate	*	pass	fail	*	*	pass	fail	*
Overall result	pass	pass	pass	fail	fail	fail	fail	fail

See also

- *LDAP Security Provider* on page 67
- *LDAP Configuration Properties* on page 101

controlFlag Attribute Values

(Not applicable to Online Data Proxy) The Sybase implementation uses the same controlFlag values and definitions as those defined in the JAAS specification.

If you stack multiple providers, you must set the controlFlag attribute for each enabled provider.

Control Flag Value	Description
(Default) required	The LoginModule is required. Authentication proceeds down the LoginModule list.
requisite	The LoginModule is required. Subsequent behavior depends on the authentication result: • If authentication succeeds, authentication continues down the LoginModule list. • If authentication fails, control returns immediately to the application (authentication does not proceed down the LoginModule list).

Control Flag Value	Description
sufficient	The LoginModule is not required. Subsequent behavior depends on the authentication result: • If authentication succeeds, control returns immediately to the application (authentication does not proceed down the LoginModule list). • If authentication fails, authentication continues down the LoginModule list.
optional	The LoginModule is not required. Irrespective of success or failure, authentication proceeds down the LoginModule list.

Example

Providers are listed in this order and with these controlFlag:

1. CertificateAuthenticationLoginModule (sufficient)
2. LDAP (optional)
3. NativeOS (sufficient)

A client doing certificate authentication (for example, X.509 SSO to SAP) can authenticate immediately. Subsequent modules are not called, because they are not required. If there are regular username/password credentials, they go to LDAP, which may authenticate them, and set them up with roles from the LDAP groups they belong to. Then NativeOS is invoked, and if that also succeeds, Unwired Platform picks up roles based on the Windows groups they are in.

Stacking LoginModules in SSO Configurations

(Not applicable to Online Data Proxy) Use LoginModule stacking to enable role-based authorization for MBOs and data change notifications (DCNs).

Retrieving Roles for Subjects Authenticating to Single Sign-on Enabled Login Modules

The CertificateAuthenticationLoginModule does not extract role information. If MBOs and MBO operations have roles assigned, stack login modules to get roles for the user.

1. HttpAuthenticationLoginModule – username and password credentials are supplied by the user. If these credentials go to an LDAP/AD EIS, add an LDAPAuthorizer with appropriate properties to look up the LDAP subject and retrieve LDAP groups as roles. You can also use the csi-userrole authorizer; but role-mapping maintenance is onerous with a large user base.
2. CertificateAuthenticationLoginModule – use the csi-userrole provider to map logical roles to physical roles named user:*subject* where *subject* matches the common name (CN=*xxx*) from the X.509 certificate.

See *Configuring an LDAP Authentication Module* in *Sybase Control Center online help*.

Stacking Login Modules to Allow for DCN in Packages Using SSO Login Modules
The DCN user must authenticate as a user in the "SUP DCN User" logical role.

All DCN operations require the "SUP DCN User" logical role in the named security configuration (role mapping applies). An additional LoginModule with authorization, that can assign a physical role, is required. Module stacking authenticates DCN users, and grants them the DCN role. Ordering of modules and control flag settings in the security configuration can vary. For example:

1. HttpAuthenticationLoginModule – the CertificateAuthenticationLoginModule is first in the list with the controlFlag set to "sufficient". If authentication succeeds, no other Login Modules are called unless their controlFlags are set to "required".
2. If CertificateAuthenticationLoginModule is used to authenticate mobile users, stack another login module for the DCN user because the DCN user does not have a "certblob" credential with the X.509 certificate.
   a) Set the CertificateAuthenticationLoginModule's controlFlag to "sufficient", and first in the stack, which allows normal device users to authenticate quickly.
   b) Choose any other username/password based login module to stack with controlFlag set to "optional" (or sufficient). If this login module does not include an authorizer that can retrieve roles, use the csi-userrole provider.

      For example, if a DCN includes one DCN technical user, it requires only one role mapping, from "SUP DCN User" to user:dcnTechnicalUser.

Assigning Security Configurations to Domains, Packages, or Applications

Once the platform administrator creates a security configuration, it can be assigned to a domain. Domain administrators can then select the security configuration when deploying synchronization packages or creating application templates.

By selecting a security configuration at the package or application connection template level, you can choose the granularity required for user authentication. For details on how to assign and select a security configuration at the domain, package or application level, search for Security Configuration in the *Sybase Control Center* online help.

See also

- *Assigning Providers to a Security Configuration* on page 65

Mapping Roles for Domains, Packages, or Applications

Role mappings can occur at two levels. Package level mappings override the mappings set at the default level. A default level is a level from which the role mapping is inherited.

Unwired Platform uses a role mapper to map logical and physical roles during an access control check. This allows developers to create applications that incorporate a logical access control policy. When the application is deployed, a security administrator can work with the developer to understand what the logical roles in the application were intended to do and map these logical roles to physical roles that exist in the real security system.

Default SUP Roles

Default Unwired Platform roles are logical roles that are built into the system.

These are the default roles: SUP Administrator, SUP Domain Administrator, and SUP DCN User.

All default roles must be mapped:

- In a development environment, mappings are automatically created.
- In a production environment, physical roles must be added manually to the directory used, and then manually mapped in Sybase Control Center.

SUP DCN User Role

The SUP DCN User is a logical role that Unwired Platform uses to authorize any DCN event: updating data in the cache, executing an operation, or triggering a workflow package.

Before any DCN event is submitted, the person or group mapped to this role must be authenticated and authorized by the security configuration. By default, SUP DCN User is automatically available to all new security configurations. However, the underlying default varies depending on the environment in use.

Before this logical role can be used, SUP DCN User must be mapped to a physical role in the enterprise security repository, and the user who performs DCN must be in that physical role.

- To map the SUP DCN User to a user in the underlying security repository, the user name must be first defined in Sybase Control Center as a physical role that is mappable. Then, SUP DCN User role can be mapped to a physical user or to a physical role from Sybase Control Center. For example, to map SUP DCN User to a user that is not in the security repository, use the format `user:`*`User`*.

  If you are supporting multiple domains, the user name must also include the named security configuration for the package the DCN is targeted for, by appending *@DomainSecurityConfigName* as a suffix to that name. Suppose you have two packages (PKG_A, PKG_B) deployed to two domains (Domain_A, Domain_B) respectively. Further, assume that PKG_A in Domain_A has been assigned to the "admin" security configuration, whereas PKG_B in Domain_B has been assigned to the "alternateSecurityConfig" security configuration.

- A user doing DCN to PKG_A should identify him or herself as *User@admin*.
- A user doing DCN to PKG_B should identify him or herself as *User@alternateSecurityConfig*.

If you are using ActiveDirectory, and are using e-mail addresses for user names, definitions appear as *username@myaddress@DomainSecurityConfigName*.

The implementation varies, depending on the DCN service used:

- For workflows, because the resource the user is pushing data toward is a group of named users (users authenticated previously successfully against a certain security configuration), he or she must have the authorization to push to that particular security configuration. The user must be mapped to SUP DCN User in the security configuration for the workflow target.
- A user having SUP DCN User logical role in security configuration "mySecConfig1"' must not have the right to push workflow DCN or regular DCN to a user or package associated with "mySecConfig2".

See also

- *SUP Administrator and SUP Domain Administrator* on page 77

SUP Administrator and SUP Domain Administrator

The SUP Administrator logical role conveys all administrative rights (that is, anybody in this role is a superuser of Sybase Control Center and can therefore perform all administrative operations). The SUP Domain Administrator logical role conveys only subset of administrative rights in Sybase Control Center (that is, pertaining only to those SUP domains to which a given subject has been assigned as domain administrator).

Note: The terms Unwired Platform (platform) administrator and domain administrator are used in this document to refer to the user with "SUP Administrator" role and "SUP Domain Administrator" role, respectively.

The SUP Administrator and SUP Domain Administrator logical roles are mapped to specific physical roles in the security repository. These roles may or may not be used for protecting mobile business objects as well.

The domain-level role mappings for these logical roles in the "admin" security configuration in the "default" domain enable the platform administrator or domain administrator access control to Unwired Platform. Both types of administrators are authenticated and authorized by the security provider of the 'admin' security configuration. All administrative and application users and their passwords are managed in the enterprise security repository. Unwired Platform delegates security checks by passing login and password information to the security provider of the "admin" security configuration.

Note: Always create a new security configuration separate from "admin", and always create a different domain besides "default". This action separates the authentication of administration users from device users to unique domains, which is a practice that Sybase strongly recommends.

Sybase Control Center limits feature visibility depending on the role an administrator logs in with. The platform administrator login has access to the full cluster, whereas the domain administrator login can view information pertaining to only the assigned domains.

Only "supAdmin" is the default login for cluster-wide administration. This login is initially assigned both "SUP Administrator" and "SUP Domain Administrator" roles.

See also

- *SUP DCN User Role* on page 76
- *Enabling Authentication and RBAC for Administrator Logins* on page 21

Mapping State Reference

The mapping state determines the authorization behavior for a logical name instance.

State	Description
AUTO	Map the logical role to a physical role of the same name. The logical role and the physical role must match, otherwise, authorization fails.
NONE	Disable the logical role, which means that the logical role is not authorized. This mapping state prohibits anyone from accessing the resource (MBO or Operation). Carefully consider potential consequences before using this option.
MAPPED	A state that is applied after you have actively mapped the logical role to one or more physical roles. Click the cell adjacent to the logical role name and scroll to the bottom of the list to see the list of mapped physical roles.

Dynamically Mapping Physical Roles to Logical Roles

Map roles at either a domain or package level, depending on the scope requirements of a particular binding. If you use a particular role mapping for a package and a different role mapping at the domain level, the package mapping overrides the domain-level mapping.

In Sybase Control Center for Unwired Platform, determine where the role mapping needs to be applied:

- For domain-level mappings, configure role mappings as part of the security configuration for a domain. For details, see *Configuring Domain Security* in *Sybase Control Center* online help.
- For package-level mappings, configure role mappings when you deploy a package to Unwired Server, or at the package-level after deployment. For details, see *Assigning Package-Level Security* in *Sybase Control Center* online help.

Security Provider Issues

If you experience problems with security configurations or the authentication or authorization providers in these configurations, check the Unwired Server logs for issues.

If no errors are being reported, despite failures that may occur while authenticating or authorizing users, you may need to increase the severity level of your logs. Search for *Logs* in the *Sybase Control Center online help*. If you are still experiencing issues, look for problems and solutions in the *Troubleshooting* guide.

Encrypting Synchronization for Replication Payloads

(Not applicable to Online Data Proxy) By default, replication is encrypted, which ensures that synchronization of enterprise dataoccurs safely.

Default certificates and keys used for replication encryption are generated during installation. These artifacts should be adequately secure for their intended purpose. If you need to replace these certificates, you can use the **createcert** and **createkey** utilities. However, in most cases, you only need to configure the secure listener for replicationse using Sybase Control Center.

See also

- *Enabling Authentication and RBAC for User Logins* on page 37
- *Unwired Server and Device Application Communications* on page 11
- *Provisioning Security Artifacts* on page 93
- *Certificate Creation (createcert) Utility* on page 122
- *Key Creation (createkey) Utility* on page 125

Validating Default Synchronization Listener Properties for Unwired Server

By default, SSL encryption is used to secure HTTP and HTTPS communications between device clients and Unwired Server.

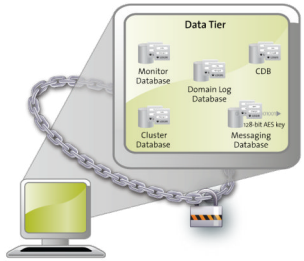
Review the default values set for Unwired Server at install time to ensure they are appropriate for your production environment.

1. Open Sybase Control Center.
2. In the left navigation pane, expand the **Servers** folder and select the server you want to configure.
3. Select **Server Configuration**.
4. In the right administration pane, click the **Replication** tab.
5. Select **Listener** from the menu bar.
6. View listener properties and optional properties, and make modifications as required.

For details on all end-to-end encryption (E2EE) properties to review, or to change the default configuration for your deployment, search for *Configuring a Synchronization Listener* in *Sybase Control Center* online help.

Data Tier Security

The data tier consists of multiple databases, each of which plays a unique role in Unwired Platform, and contains various types of sensitive data that must be secured.



Securing the Data Infrastructure

Secure data by first protecting the infrastructure on which it resides, then securing runtime databases.

1. *Setting File System Permissions*

   Unwired Platform runs as a collection of Windows services. By default, these services are installed to run as "Local System account".
2. *Securing Backup Artifacts*

   If you perform backups of Unwired Platform, you should also secure these artifacts.

Setting File System Permissions

Unwired Platform runs as a collection of Windows services. By default, these services are installed to run as "Local System account".

You can restrict permissions by removing most users and groups from the Unwired Platform installation directory.

1. Open File Explorer.
2. Right-click `<UnwiredPlatform_InstallDir>`, and click **Properties**.
3. On the **Security** tab, click **Advanced**.
4. Unselect **Inherit from parent the permission entries that apply to child objects. Include these with entries explicitly defined here.**.

5. In the confirmation pop-up, choose **Copy**, then select **Replace permission entries on all child objects with entries shown here that apply to child objects.**.
6. In the table of Permission entries, remove all users except SYSTEM and Administrators entries.
7. Click **OK**.

Securing Backup Artifacts

If you perform backups of Unwired Platform, you should also secure these artifacts.

1. Follow the instructions in *System Administration* for using **dbvalid** and **dbbackup** to back up the platform runtime data.

   Sybase recommends that you encrypt your database and all backups.
2. Protect these backups with Administrator and SYSTEM permissions.
3. Perform any additional enterprise security requirements.

Securing Data Tier Databases

Secure all databases installed as the Unwired Platform data tier. You can change DBA passwords, grant DBA permissions to other users, and encrypt data and logs.

Changing DBA Passwords for SQLAnywhere Databases

By default, Unwired Platform uses multiple SQLAnywhere databases to support the server runtime environment and transactions. Unwired Server accesses these databases with the DBA user identity.

During installation, enter a custom password for the DBA user on each of the SQLAnywhere databases used by the runtime: the cache database (CDB), the cluster database, the log database, and monitor database.

Depending on whether or not you have deployed Unwired Platform as a single node (as in the case of Online Data Proxy environments), or as a cluster, the process varies slightly:

- In single-node deployment, a single database server named CacheDB supports all installed databases.
- In a cluster deployment, these servers are used: the monitor and domain log databases use a server called LogDataDB, the default database used for the cache data uses the CacheDB server, and the cluster database uses the ClusterDB server.

1. Stop all instances of Unwired Server, as well as all database services.

   For a list of database services, search for *Unwired Platform Windows Services* in *System Administration*.
2. On data tier host:

a) Set the location of the BIN directory for your operating system (32-bit or 64-bit):

```
set SQLANY12_BIN=<UnwiredPlatform_InstallDir>\Servers
\SQLAnywhere12\bin<32|64>
```

b) Set the path to the data:

If you are using a single node, run:

```
set DATA_PATH= <UnwiredPlatform_InstallDir>\Servers
\UnwiredServer\data
```

If you are using a cluster deployment, run:

```
set DATA_PATH= <UnwiredPlatform_InstallDir>\data\CDB
```

3. Use **dbisql** to change passwords for each database as required:

For the cache database, use:

```
"%SQLANY12_BIN%\dbisqlc" -q -c "Server=default;DBF=%DATA_PATH%
\default.db;
UID=DBA;PWD=ExistingPwd" grant connect to dba identified by NewPwd
```

For the cluster database, use:

```
"%SQLANY12_BIN%\dbisqlc" -q -c "Server=clusterdb;DBF=%DATA_PATH%
\clusterdb.db;
UID=DBA;PWD=ExistingPwd" grant connect to dba identified by NewPwd
```

For the monitor database, use:

```
"%SQLANY12_BIN%\dbisqlc" -q -c "Server=monitordb;DBF=%DATA_PATH%
\monitordb.db;
UID=DBA;PWD=ExistingPwd" grant connect to dba identified by NewPwd
```

For the domain log database, use:

```
"%SQLANY12_BIN%\dbisqlc" -q -c "Server=domainlogdb;DBF=%DATA_PATH
%\domainlogdb.db;
UID=DBA;PWD=ExistingPwd" grant connect to dba identified by NewPwd
```

4. Run this command on each Unwired Server node host:

```
Register-dsn.bat cdb.install_type cdb.serverhost cdb.serverport
cdb.username
 %CDB_PASSWORD% cdb.servername cldb.dsnname %CLDB_PASSWORD%
%MONITORDB_PASSWORD% %DOMAINLOGDB_PASSWORD%
```

To see the values used in the properties of this command, open the `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\Instance\com\sybase\sup\server\SUPServer\sup.properties` file and search for the corresponding property.

5. If you receive an `Invalid user ID` or `Invalid password` error, you may have already the password from "sql" to different one). In this case:

a) Backup the **register-dsn.bat**.

b) Open this file in a text editor and locate this section of the file:

```
IF "default" == "%CDB_INSTALLTYPE%" (
    echo Changing DBA password for databases ...
    "%SQLANY12_BIN%\dbisqlc" -q -c "Server=default;DBF=
```

```
%DJC_HOME%\data\default.db;UID=DBA;PWD=sql" grant connect to
dba identified by %CDB_PASSWORD%
    "%SQLANY12_BIN%\dbisqlc" -q -c "Server=clusterdb;DBF=
%DJC_HOME%\data\clusterdb.db;UID=DBA;PWD=sql" grant connect to
dba identified by %CLDB_PASSWORD%
    "%SQLANY12_BIN%\dbisqlc" -q -c "Server=monitordb;DBF=
%DJC_HOME%\data\monitordb.db;UID=DBA;PWD=sql" grant connect to
dba identified by %MONITORDB_PASSWORD%
    "%SQLANY12_BIN%\dbisqlc" -q -c "Server=domainlogdb;DBF=
%DJC_HOME%\data\domainlogdb.db;UID=DBA;PWD=sql" grant connect
to dba identified by %DOMAINLOGDB_PASSWORD%
)
```

c) Replace `PWD=sql` with the `PWD=PreviousPassword`.

6. Restart all database services, then all Unwired Servers.

Encrypting Data and Log Outputs

Sybase SQL Anywhere database files and log files that are used as part of the Unwired Platform data tier can be encrypted. The databases that use this database type are the CDB, the monitoring database, and the domain log database.

1. Stop all Sybase Unwired Platform services.
2. From **dbisql**, issue:

   ```
   CREATE ENCRYPTED DATABASE ‘newdbfile’ FROM ‘existingdbfile’ KEY
   ‘someKey’ ALGORITH ‘algorithm’
   ```

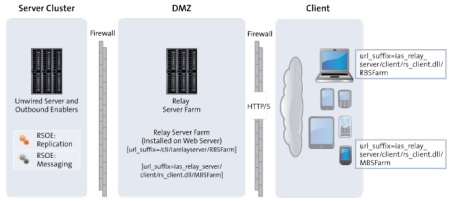
   Supported algorithms include:
   - SIMPLE
   - AES
   - AES256
   - AES_FIPS
   - AES256_FIPS

   FIPS options areavailable only as a separately licensed option for SQLAnywhere.
3. Once the database files and log files are encrypted:
   a) Shut down the database server and modify its start-up to use the encrypted copy.
   b) Change the database command line options to include the -ek <encryption key> database option.
4. Start the database server then restart all stopped services.

DMZ Security

DMZ security involves controlling Internet traffic to private networks by installing a Relay Server between your inner and outer firewalls.

The outer firewall has HTTP and HTTPS ports open to allow client Internet traffic to reach the Relay Server.



Relay Server as Firewall Protection

The Relay Server is a pair of Web server plug-ins, which you can install in an Internet Information Service (IIS) server on Windows or the Apache Web server on Linux.

The Relay Server is intended to run between a company's inner and outer firewalls. The outer firewall has HTTP and HTTPS ports open to allow client Internet traffic to reach the Relay Server. The client's URL includes the address of the client-side plug-in of the Relay Server and the name of the backend Sybase Unwired Platform "farm" the client is trying to reach. A farm includes multiple Relay Servers for load balancing and fault tolerance. The network administrator needs to install a load balancer in front of the Relay Servers. The load balancer is not included with Sybase Unwired Platform. To make the interaction secure, the clients should use end-to-end encryption.

The server-side plug-in accepts connections from various Relay Server Outbound Enabler (RSOE) processes, which indicate to the Relay Server what back-end farm each process represents. The Relay Server matches the farm name in the client's request to a server-side plug-in connection and routes the client's request contents to that connection. Other than the farm name in the request URL, the Relay Server knows nothing about the content of these messages. The clients are not authenticated or authorized in any way. The client information is in memory and not susceptible to interception or modification. (If the administrator turns certain tracing options up very high, data may get copied to log files.) If end-to-end encryption is used, the data is undecipherable.

Security administrators secure the Relay Server as they would with any other Web server or proxy server they run between firewalls, so the same security precautions should be taken of setting up a proxy server.

See also

- *RSOE as the Unwired Server Protection* on page 86
- *Relay Server and RSOE Communication Security* on page 86

RSOE as the Unwired Server Protection

One RSOE process is installed in each Unwired Server cluster member, in front of each synchronization subcomponent that communicates with a client. Replication services components and messaging services components both use RSOEs attached to their communication ports.

The RSOE configuration enables the Relay Server to identify the RSOE and connect to it. The RSOE configuration also has a single port number that enables the RSOE to make a `http://localhost:port` connection whenever a client request comes to it from the Relay Server.

See also

- *Relay Server as Firewall Protection* on page 85
- *Relay Server and RSOE Communication Security* on page 86

Relay Server and RSOE Communication Security

The RSOE runs on the same computer as an Unwired Server and is configured with the address of a Relay Server (the inner firewall is open to outgoing traffic, but not incoming traffic).

The RSOE connects to the Relay Server via HTTP or HTTPS and identifies itself through the Media Access Control (MAC) address, security token, and the backend Sybase Unwired Platform farm it services. The Relay Server identifies the RSOE's authenticity. If the RSOE's identity is accepted, the Relay Server sends it a list of all the other Relay Servers in the Relay Server farm. The RSOE establishes a blocking GET HTTP request to each of the Relay Servers in the farm. When a Relay Server receives a client request for a given Sybase Unwired Platform farm, it picks one of the available RSOE connections to that farm and sends the client request there.

In this way, the network administrator does not need to open ports on the inner firewall to allow connection requests between the firewalls into the intranet. All connection requests come from within the intranet. Avoiding firewall portholes protects the intranet from hackers who breach the outer firewall.

This network traffic contains exactly the same content, and thus the same security concerns as network communication between the device application or database and the Relay Server.

See also

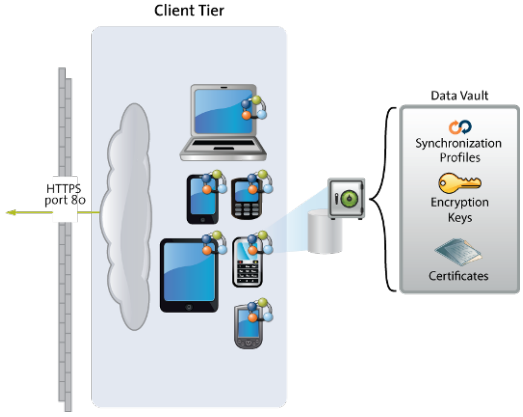
- *Relay Server as Firewall Protection* on page 85

- *RSOE as the Unwired Server Protection* on page 86

Device Security

You can combine multiple mechanisms to fully secure devices. In addition to using the built-in security features of both the device or Unwired Platform, Sybase recommends that you also use Afaria so you can remotely initiate security features as required.

Unwired Platform security features for devices include data encryption, login screens, and data vaults for storing sensitive data.



Limiting Application Access

Application access to Unwired Platform runtime is tightly controlled: before a user can access a mobile application, he or she must provide a passcode; before the application can access the runtime, the application must be registered and provisioned with required connections and security configurations.

1. *Encrypting Data*

   Encrypting all data on the device client requires multiple techniques.
2. *DataVaults on Client Devices*

   All applications, except Workflow, use a DataVault to store secrets, including those used to authenticate user (for example, certificates). Developers include a login screen to unlock the DataVault. Unlocking the vault enables access to application data off-line or on-line. In contrast, Workflow applications can require user credentials that must be checked against Unwired Server on-line before granting access to Workflow content.
3. *Registering Applications, Devices, and Users*

   Before messaging, workflow, or OData applications can access the runtime, the user, device, and application must be identified by first registering with Unwired Server and pairing them with a device and user entry. Only when all three entities are known, can

subscriptions can be made or data synchronized. In contrast, replication users must know only their authentication credentials (passed on to backend security stores via the security provider).

4. *Locking and Unlocking a Subscription*

   (Not applicable to Online Data Proxy) Create a subscription template to lock or unlock a subscription. A subscription determines what data set the device user receives upon synchronization and how frequently the synchronization can occur. Lock the subscription to prevent modification to the template and control the synchronization frequency

5. *Locking and Unlocking Application Connections*

   Lock or unlock connections to control which users are allowed to synchronize data. Locking an application connection is an effective way to disable a specific user without making changes to the security profile configuration to which he or she belongs.

Encrypting Data

Encrypting all data on the device client requires multiple techniques.

Some Unwired Platform components do not support encryption. Review this table to see which components can enable this security feature.

Component	Implementation notes
Device data	Sybase recommend full device encryption with Afaria. See the Afaria documentation for details.
Device client database	(Not applicable to Online Data Proxy) A `<package>DB.generateEncryptionKey()` method in the Object API for MBO packages should always be used during application initialization. It computes a random AES-128 bit encryption key used to encrypt the client database. The encryption key is stored in the data vault.
Data vault	The DataVault APIs provide a secure way to persist and encrypt data on the device. The data vault uses AES-128 symmetric encryption of all its contents. The AES key is computed as a hash of the passcode provided and a "salt" value that is usually set by the SUP device application developer.

DataVaults on Client Devices

All applications, except Workflow, use a DataVault to store secrets, including those used to authenticate user (for example, certificates). Developers include a login screen to unlock the DataVault. Unlocking the vault enables access to application data off-line or on-line. In contrast, Workflow applications can require user credentials that must be checked against Unwired Server on-line before granting access to Workflow content.

See also

- *Registering Applications, Devices, and Users* on page 91
- *Implementing Login Screens and Data Vaults* on page 91

Implementing Login Screens and Data Vaults

An application that implements a login screen is considered to be secure. Mobile application developers are responsible for creating login screens for the applications they create. A login screen allows the device user to enter a passcode to unlock the data vault. The data vault acts as the local repository of sensitive data.

The data vault holds sensitive artifacts securely, because all data or artifacts in the data vault is strongly encrypted. Contents can include encryption keys, user and application login credentials, sync profile settings, certificates (as BLOBS).

A secure application that uses a login screen:

1. Prompts the user to enter the datavault passcode to open the application and get access to the local client database. If the wrong passcode is used, the application is rendered useless: the key that encrypts and decrypts data in the vault cannot be used to access data until this code is accurately entered.

   After a certain amount of time passes, the login in screen can be redeployed to prompt the user to re-enter the passcode.
2. Can be locked out after a configured number of retries.
3. Can self-destruct after a set number of incorrect passcode attempts.

   When this occurs, the device user must uninstall, reinstall, then perform an initial sync to recover from a destroyed data vault.

For more information on the data vault, see the *Developer Guide* for your device type.

See also

- *Data Vaults on Client Devices* on page 90

Registering Applications, Devices, and Users

Before messaging, workflow, or OData applications can access the runtime, the user, device, and application must be identified by first registering with Unwired Server and pairing them with a device and user entry. Only when all three entities are known, can subscriptions can be made or data synchronized. In contrast, replication users must know only their authentication credentials (passed on to backend security stores via the security provider).

In Sybase Control Center, Platform administrators set up an application connection template for a messaging, workflow, or OData application, wherein they can specify whether or not to Enable Automatic Registration.

- When automatic registration is enabled, a device user need only provide valid Unwired Platform credentials that are defined as part of the security configuration.
- When automatic registration is disabled, the platform administrator must provide the user a username/passcode out-of-band. This is the passcode initially required by login screens to access the application for the first time, and expires within a predetermined time period (72 hours, by default).

See also

- *Data Vaults on Client Devices* on page 90
- *Locking and Unlocking a Subscription* on page 92

Registering Application Connections

Devices can be registered using either a one-time passcode during the registration of the device or application, or to allow automatic registration.

When a package is deployed, an Application Connection Template is automatically created for it. As long as the user is able to authenticate to the security configuration associated with the application, they are registered. If automatic registration is disabled then the administrator must generate a passcode. For details, see *Sybase Control Center* online help and search for *Registration and Onboarding*.

1. Select the application template, and click the **Properties** button.
2. Navigate to the **Application Settings** tab and set the **Automatic Registration Enabled** property to True or False. If True, no one-time passcode is required.

Manually Creating Applications

Create an application manually by assigning a unique application ID and other key application properties, such as domain, MBO package, security configuration, among others. At this time, the manual process is only needed for Online Data Proxy applications or when using a Hybrid Web Container built using the iOS sample, where developers can use their own application IDs for workflow applications.

Locking and Unlocking a Subscription

(Not applicable to Online Data Proxy) Create a subscription template to lock or unlock a subscription. A subscription determines what data set the device user receives upon synchronization and how frequently the synchronization can occur. Lock the subscription to prevent modification to the template and control the synchronization frequency

1. In the left navigation pane, expand the **Packages** folder and select the replication-based package to configure.
2. In the right administration pane, click the **Subscriptions** tab.
3. From the menu bar, select **Templates**, then click **New**.

4. Select **Admin Lock** to prevent device users from modifying the push synchronization state or sync interval value configured in the subscription. If the admin lock is disabled, the device client user can change these properties, and these changes take effect the next time the client user synchronizes the package to which the subscription applies.

See also

- *Registering Applications, Devices, and Users* on page 91

Locking and Unlocking Application Connections

Lock or unlock connections to control which users are allowed to synchronize data. Locking an application connection is an effective way to disable a specific user without making changes to the security profile configuration to which he or she belongs.

1. In the left navigation pane, select the**Applications** node.
2. In the right administration pane, select the **Application Connections** tab.
3. Select the application connection you want to manage, and:
   - If the connection is currently unlocked and you want to disable synchronization, click **Lock**.
   - If the connection is currently locked and you want to enable synchronization, click **Unlock**.
4. In the confirmation dialog, click **OK**.

Provisioning Security Artifacts

Typically, you must preprovision Unwired Platform security artifacts before applications can be used. The manner by which an artifact is provisioned depends on the artifact.

See also

- *Encrypting Synchronization for Replication Payloads* on page 79

Seeding Applications Using Afaria

(Applies only to Online Data Proxy) Applications must be provisioned with name=value pair parameters to connect to Unwired Server. Those parameters can include connection parameters for Unwired Server including the server's public key, and an X.509 certificate for authentication through SSO. Administrators can automate the process of provisioning applications by enabling a group of applications to be intially provisioned with a set of parameters from a configuration file using Afaria. Developers include API calls in the application to retrieve the provisioning data and certificate.

This task assumes that an Afaria client is co-installed on the device, that the user has enrolled in Afaria management, and that a Certificate Authority server is available to be configured for use with an Afaria portal package.

1. Ensure that required configuration files, keys (for E2EE), or other artifacts are available.

   The administrator creates an initial configuration file for applications which are to be deployed and provisioned through Afaria. See *System Administration* for the format of this file, and instructions on how to get the public key of the Unwired Server, for inclusion in the file.).

   Note: Without Afaria, the initial configuration file can be manually placed on a device, and consumed by the application. iOS devices do not support this capability because the application sandbox cannot be manually accessed. See *System Administration* to learn what the name of the file must be, and where it needs to be placed (it varies by platform).

2. Ensure the mobile application developer includes code in the application to allow the application client to make request for a certificate and provisioning data to Afaria: validate that this key exists on the device and is properly provisioned.
   a) Include code to check whether the application is provisioned, and if not, retrieve a provisioning file using Afaria. The provisioning file includes the public key of the Unwired Server.
   b) If an application wishes to use certificate-based authentication, the application developer includes code in the application to retrieve an X.509 certificate using Afaria and a Certificate Authority configured for the Afaria portal package. The application consumes, uses, or deletes the certificate as required. The application developer sets up a synchronization profile, and presents the user's certificate for authentication.

      The application developer also includes in the synchronization profile any other required application configuration parameters, such as encryption keys used for synchronization.

      The developer also includes code to prompt the application user to enter the Common Name and Challenge Code.

   For details on client APIs, see the *Developer Guide* for your device type.
3. The Afaria administrator performs the following tasks to creates a portal package to serve deployments for a group of application of a specified device platform on an Unwired Server:
   a) Defines the Certificate Authority server for the portal package.
   b) Includes the portal package into a Group Profile.
   c) Imports the configuration file to provision applications in that portal package.
   d) Enables Simple Certificate Enrollment Protocol (SCEP) in Afaria Server.

      Use SCEP to:
      - Create or obtain a user certificate on behalf of a mobile client.
      - Send this certificate back to the device.

   For details, see the Afaria server configuration documentation.

4. The IT administrator generates and distributes to end users the Common Name and Challenge Code required by the Certificate Authority configured for the Afaria portal package.

See also

- *Seeding Messaging Devices With Unwired Server* on page 95

Configuring Application Code Properties for HTTPS Connections

Application client properties can be set by the mobile application developer to connect to the secure Unwired Server port.

Ensure the application code uses the HTTPS protocol, port, and stream parameters.

- Unwired Server port – 2481.
- Protocol – HTTPS (equivalent to the MobiLink Stream Type).
- Stream Parameter – `trusted_certificates=mypublic_cert.crt`

Seeding Messaging Devices With Unwired Server

If you are not using Afaria, you can install the client application then connect to corporate LAN using Wifi or any other method of your choosing in order to seed devices with required files.

Sybase recommends you follow this method when you are not using Afaria. This method mitigates man-in-the-middle attacks that might otherwise occur.

1. Install the device client application to the device.
2. Connect to the corporate LAN upon which Unwired Platform is installed.
3. Use a device connection that connects directly to Unwired Server. Alternatively, you can also connect using the Relay Server settings, but only if it is accessible from the corporate LAN.
4. The messaging service on Unwired Server seeds the client with the public key.

   The client uses this public key for all subsequent connections.
5. Provide the user with instructions to re-configure the connection properties on the device to useRelay Server from the Internet for subsequent connections.

See also

- *Seeding Applications Using Afaria* on page 93

Establishing Encrypted Application Connections

Synchronization and messaging connection are encrypted by default. However, for replication connections that use E2EE, the client must be configured correctly to establish connections to the correct HTTP or HTTPS port.

Connecting to the SSL Relay Server Port

BlackBerry and Windows Mobile replication clients should connect to Unwired Server through a Relay Server on the DMZ.

Prerequisites

When using E2EE over SSL, certificates need to be created and distributed to both Unwired Server and clients.

Task

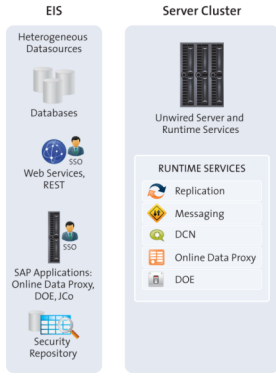
Only BlackBerry and Windows Mobile can enable E2EE over SSL because they use an UltraLite client database. To enable E2EE:

1. Configure application connection code properties to connect to Relay Server with the correct combination of properties.
2. Generate and deploy files that reference this configuration.

For details see *Enable End-to-End Encryption (E2EE) Using SSL* in the corresponding guides: either *Developer Guide: BlackBerry Native Applications* or *Developer Guide: Windows Mobile Native Applications*.

EIS Security

Secure interactions with EIS data sources. DCNs can be securely communicated to Unwired Server. Otherwise, appropriately secure the connections Unwired Server makes to each heterogeneous data source.



Securing Data Change Notifications

If you use data change notifications (DCN) to notify Unwired Server of EIS changes, secure the communication stream to avoid packet sniffing or data tampering. Use Sybase Control Center to configure the DCN stream after you have created SSL certificates.

1. *Preparing SSL Certificates for DCNs*

   DCN, which uses HTTP Basic authentication, uses a Base64-encoded username:password field that can be intercepted by network sniffers. Encrypt DCN communications and always use HTTPS to send DCNs. HTTPS requires the DCN sender import the Unwired Server certificate (or that of its CA signer) into its local equivalent of a truststore. If you are also configuring the HTTPS listener for mutual certificate authentication, then corresponding files need to be copied to the Unwired Server truststore.
2. *Creating and Enabling a DCN Security Profile*

   An administrator must enable and configure the HTTPS port for DCN connections as part of a security profile so that developers can construct callouts from the EIS backend to send Unwired Server data change notification (DCN) requests.
3. *Enabling Authentication for Data Change Notifications*

   (Not applicable to Online Data Proxy) Data change notifications (DCN) are always authorized by the "SUP DCN User" logical role. The DCN servlets through which the

HTTP DCN requests are sent are configured to require HTTP Basic authentication and then check this role.

Preparing SSL Certificates for DCNs

DCN, which uses HTTP Basic authentication, uses a Base64-encoded username:password field that can be intercepted by network sniffers. Encrypt DCN communications and always use HTTPS to send DCNs. HTTPS requires the DCN sender import the Unwired Server certificate (or that of its CA signer) into its local equivalent of a truststore. If you are also configuring the HTTPS listener for mutual certificate authentication, then corresponding files need to be copied to the Unwired Server truststore.

In Sybase Control Center, configure DCN to use HTTPS by creating a security profile and enabling the HTTPS listener for DCN to use the security profile configured.

See also

- *Creating and Enabling a DCN Security Profile* on page 99

Creating and Importing Self-Signed Certificates in Development Environments

Use a self-signed certificate generated from the Java SDK **keytool** utility, andimports it into the Unwired Platform for DCN encryption testing.

Prerequisites

To use the **keytool** utility, set the `JAVA_HOME` environment variable to the JDK directory used by Unwired Platform, in addition to defining `%JAVA_HOME%\bin` as a the path variable. to the path variable, because keytool utility needs this setting.

Task

1. Change directory to `<UnwiredPlatform_InstallDir>\Repository\Security`, and run this command:

   ```
   keytool -genkey -alias dcn -keypass changeit -keyalg RSA -keysize
   1024 -validity 3650 -keystore dcn.jks
   -storepass chageit
   ```

   Follow the prompts to generate a public-private keypair for the Acme organization.

Prompt	Value Entered
Name	`Admin's name`
Organization	`Your company name`
Organizational Unit	`Development`
City	`Your city`

Prompt	Value Entered
State/Province	`Your location`
Country Code	`Your country`

2. Use **keytool** to import the keystore to the destination keystore by using this command:

```
keytool -importkeystore -destkeystore keystore.jks -deststorepass
changeit -srckeystore dcn.jks
-srckeypass changeit -alias DCN
```

3. Use **keytool** to export just the public key to the local disk.

```
keytool -keystore keystore.jks -storepass changeit -alias DCN -
export -file C:\temp\dcn.crt
```

Creating and Enabling a DCN Security Profile

An administrator must enable and configure the HTTPS port for DCN connections as part of a security profile so that developers can construct callouts from the EIS backend to send Unwired Server data change notification (DCN) requests.

1. In Sybase Control Center for Unwired Platform, expand the server node in the corresponding cluster.
2. In the left navigation pane, click **Server Configuration**.
3. In the right administration pane, click **General**.
4. Select **SSL Configuration**.
5. Create a new security profile by entering these values in an empty row of the table.

   For self-signed certificates, the alias value is the same value set with the **keytool** -alias property.

Column Name	Value
Security Profile	*`MyDCNprofileName`*
Certificate Alias	*`aliasUsed`*
Authentication	`intl`

6. On the General tab, click the **Communication Ports**.
7. Enable the listener for DCN:
   - If you are using Unwired Platform with MBOs, in the **SSL Profile** column for the port number 8001, select **Enabled**, then choose **dcn_profile**.
   - If you are using Unwire Platform with OData and Gateway, enter a new port of 8004, select **Enabled** then choose *MyDCNprofileName*
8. Restart the Unwired Server.

9. Deploy a package that contains an application that uses DCN and test the DCN settings with it.

Next

Share connection properties with the development team.

See also

- *Preparing SSL Certificates for DCNs* on page 98

Enabling Authentication for Data Change Notifications

(Not applicable to Online Data Proxy) Data change notifications (DCN) are always authorized by the "SUP DCN User" logical role. The DCN servlets through which the HTTP DCN requests are sent are configured to require HTTP Basic authentication and then check this role.

1. In Sybase Control Center, ensure that the MBO package that is the target of DCN uses the "admin" security configuration, and that an HTTP Basic authentication provider is configured for DCN.
2. Ensure that the user name specified in the HTTPS Basic authentication response needs is formatted as:

   ```
   dcnSubject@security_configuration
   ```

   This format ensures that authentication/authorization happens in the correct security context.

   Note: In some ActiveDirectory configurations, the username may have the form user@activeDirectoryDomain. If a DCN user is fred@acme.com, and the security configuration is named "myCustomConfig", then the username specified for DCN request must be "fred@acme.com@myCustomConfig".

See also

- *HTTP Basic Security Provider* on page 71
- *HTTP Basic Authentication Properties* on page 116

Security Reference

Security Provider Configuration Properties

Security providers implement different properties, depending on whether or not they support authentication, authorization, audit, or attribution.

Platform administrators can configure application security properties in the Sybase Control Center for Unwired Platform console. These properties are then transcribed to an XML file in the `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\CSI\` directory. A new section is created for each provider you add.

LDAP Configuration Properties

(Not applicable to Online Data Proxy) Use these properties to configure the LDAP provider used to authenticate SCC administration logins or to configure the LDAP provider used to authenticate device application logins. If you are creating a provider for device application logins, then Unwired Platform administrators use Sybase Control Center to write these properties to the *<UnwiredPlatform_InstallDir>*`\Servers\UnwiredServer\Repository\CSI\default.xml` file.Use these properties to configure the LDAP provider used to authenticate SCC administration logins or to configure the LDAP provider used to authenticate device application logins. If you are creating a provider for device application logins, then Unwired Platform administrators use Sybase Control Center to write these properties to the *<UnwiredPlatform_InstallDir>*`\Servers\UnwiredServer\Repository\CSI\default.xml` file.

Unwired Server implements a Java LDAP provider through a common security interface used by other Sybase products like Sybase Control Center.

The Java LDAP provider consists of three provider modules, each of which is in the `com.sybase.security.ldap` Java package. This is why the syntax used between Sybase Control Center provider and Unwired Server varies.

- **LDAPLoginModule**– provides authentication services. Through appropriate configuration, you can enable certificate authentication in **LDAPLoginModule**.
- (Optional)**LDAPAuthorizer** or **RoleCheckAuthorizer** – provide authorization services for **LDAPLoginModule**. **LDAPLoginModule** works with either authorizer. In most production deployments, you must always configure your own authorizer. However, if you are authenticating against a service other than LDAP, but want to perform authorization against LDAP, you can use the **LDAPAuthorizer**.
  The RoleCheckAuthorizer is used with every security configuration but does not appear in Sybase Control Center.

Use **LDAPAuthorizer** only when **LDAPLoginModule** is not used to perform authentication, but roles are still required to perform authorization checks against the LDAP data store. If you use **LDAPAuthorizer**, always explicitly configure properties; for it cannot share the configuration options specified for the **LDAPLoginModule**.

Use this table to help you configure properties for one or more of the supported LDAP providers. When configuring modules or general server properties in Sybase Control Center, note that properties and values can vary, depending on which module or server type you configure.

Property	Default Value	Description
ServerType	None	Optional. The type of LDAP server you are connecting to: • `sunone5` -- SunOne 5.x OR iPlanet 5.x • `msad2k` -- Microsoft ActiveDirectory, Windows 2000 • `nsds4` -- Netscape Directory Server 4.x • `openldap` -- OpenLDAP Directory Server 2.x The value you choose establishes default values for these other authentication properties: • RoleFilter • UserRoleMembership • RoleMemberAttributes • AuthenticationFilter • DigestMD5Authentication • UseUserAccountControl
ProviderURL	`ldap://localhost:389`	The URL used to connect to the LDAP server. Without this URL configured, Unwired Server cannot contact your server. Use the default value if the server is: • Located on the same machine as your product that is enabled with the common security infrastructure. • Configured to use the default port (389). Otherwise, use this syntax for setting the value: `ldap://<hostname>:<port>`

Property	Default Value	Description
DefaultSearchBase	None	The LDAP search base that is used if no other search base is specified for authentication, roles, attribution and self registration: 1. `dc=<domainname>,dc=<tld>` For example, a machine in sybase.com domain would have a search base of dc=sybase,dc=com. 2. `o=<company name>,c=<country code>` For example, this might be o=Sybase,c=us for a machine within the Sybase organization.
SecurityProtocol	None	The protocol to be used when connecting to the LDAP server. To use an encrypted protocol, use "ssl" instead "ldaps" in the url. **Note:** ActiveDirectory requires the SSL protocol when setting the value for the password attribute. This occurs when creating a user or updating the password of an existing user.
AuthenticationMethod	simple	The authentication method to use for all authentication requests into LDAP. Legal values are generally the same as those of the java.naming.security.authentication JNDI property. Choose one of: • simple — For clear-text password authentication. • DIGEST-MD5 — For more secure hashed password authentication. This method requires that the server use plain text password storage and only works with JRE 1.4 or later.

Property	Default Value	Description
AuthenticationFilter	For most LDAP servers: `(&(uid={uid})(objectclass=person))` or For Active Directory email lookups: `(&(userPrincipalName={uid})(objectclass=user))[ActiveDirectory]` For Active Directory Windows username lookups: `(&(sAMAccountName={uid})(objectclass=user))`	The filter to use when looking up the user. When performing a username based lookup, this filter is used to determine the LDAP entry that matches the supplied username. The string "{uid}" in the filter is replaced with the supplied username.
AuthenticationScope	onelevel	The authentication search scope. The supported values for this are: • `onelevel` • `subtree` If you do not specify a value or if you specify an invalid value, the default value is used.
AuthenticationSearchBase	none	The search base used to authenticate users. If this value is not specified, the LDAP DefaultSearchBase is used.

Property	Default Value	Description
BindDN	none	The user DN to bind against when building the initial LDAP connection. In many cases, this user may need read permissions on all user records. If you do not set a value, anonymous binding is used. Anonymous binding works on most servers without additional configuration. However, the LDAP attributer may also use this DN to create the users in the LDAP server. When the self-registration feature is used, this user may also need the requisite permissions to create a user record. This behavior can occur if you do not set useUserCredentialsToBind to `true`. In this case, the LDAP attributer uses this DN to update the user attributes.
BindPassword	none	BindPassword is the password for BindDN, which is used to authenticate any user. BindDN and BindPassword are used to separate the LDAP connection into units. The AuthenticationMethod property determines the bind method used for this initial connection. If you use an encrypted the password using the CSI encryption utility, append .e to the property name. For example: `CSI.loginModule.7.options.BindPassword.e=1-AAAAEgQQOLL+LpXJO8fO9T4SrQYRC9lRT1w5ePfdczQTDsP8iACk9mDAbm3F3p5a1wXWKK8+NdJuknc7w2nw5aGJlyG3xQ==`
RoleSearchBase	none	The search base used to retrieve lists of roles. If this value is not specified, the LDAP DefaultSearchBase is used.

Property	Default Value	Description
RoleFilter	For SunONE/iPlanet: `(&(objectclass=ldapsubentry) (objectclass=nsroledefinition))` For Netscape Directory Server: (`objectclass=groupofnames) (objectclass=groupofuniquenames`s)) For ActiveDirectory: `(objectclass=groupofnames) (objectclass=group))`	The role search filter. This filter should, when combined with the role search base and role scope, return a complete list of roles within the LDAP server. There are several default values depending on the chosen server type. If the server type is not chosen or this property is not initialized, no roles are available.
RoleMemberAttributes	For Netscape Directory Server: member,uniquemember	The role's member attributes defines a comma-delimited list of attributes that roles may have that define a list of DN's of people who are in the role. These values are cross referenced with the active user to determine the user's role list. One example of the use of this property is when using LDAP groups as placeholders for roles. This property only has a default value when the Netscape server type is chosen.
RoleNameAttribute	cn	The attribute for retrieved roles that is the common name of the role. If this value is "dn" it is interpreted specially as the entire dn of the role as the role name.
RoleScope	onelevel	The role search scope. The supported values for this are: • `onelevel` • `subtree` If you do not specify a value or if you specify an invalid value, the default value is used.

Property	Default Value	Description
UserRoleMembershipAttributes	For iPlanet/SunONE: nsRoleDN For ActiveDirectory: memberOf For all others: none	The user's role membership attributes property is used to define an attribute that a user has that contains the DN's of all of the roles as user is a member of. These comma-delimited values are then cross-referenced with the roles retrieved in the role search base and search filter to come up with a list of user's roles.
UserFreeformRoleMembershipAttributes	None	The "freeform" role membership attribute list. Users who have attributes in this comma-delimited list are automatically granted access to roles whose names are equal to the attribute value. For example, if the value of this property is "department" and user's LDAP record has the following values for the department attribute, { "sales", "consulting" }, then the user will be granted roles whose names are "sales" and "consulting".
Referral	ignore	The behavior when a referral is encountered. The valid values are those dictated by LdapContext, for example, "follow", "ignore", "throw".
DigestMD5Authentication-Format	DN For OpenLDAP: User-name	The DIGEST-MD5 bind authentication identity format.
UseUserAccountControlAttribute	For most LDAP servers: false For ActiveDirectory: true	The UserAccountControl attribute to be used for detecting disabled user accounts, account expirations, password expirations and so on. ActiveDirectory also uses this attribute to store the above information.
controlFlag	optional	Indicates whether authentication with this login module is sufficient to allow the user to log in, or whether the user must also be authenticated with another login module. Rarely set to anything other than "sufficient" for any login module. **Note:** controlFlag is a generic login module option rather than an LDAP configuration property.

See also

- *LDAP Security Provider* on page 67

- *Adding an LDAP Provider to the Admin Security Configuration* on page 23
- *Stacking Providers and Combining Authentication Results* on page 72

NTProxy Configuration Properties

(Not applicable to Online Data Proxy) Configure these properties to allow the operating system's security mechanisms to validate user credentials using NTProxy (Windows Native OS). Access these properties from the Authentication tab of the Security node in Sybase Control Center.

Table 2. Authentication properties

Properties	Default Value	Description
Extract Domain From Username	true	If set to true, the user name can contain the domain in the form of *<username>@<domain>*. If set to false, the default domain (described below) is always used, and the supplied user name is sent to through SSPI untouched.
Default Domain	The domain for the host computer of the Java Virtual Machine.	Specifies the default host name, if not overridden by the a specific user name domain.
Default Authentication Server	The authentication server for the host computer of the Java Virtual Machine.	The default authentication server from which group memberships are extracted. This can be automatically determined from the local machine environment, but this property to bypass the detection step.
useFirstPass	false	If set to true, the login module attempts to retrieve only the user name and password from the shared context. It never calls the callback handler.
tryFirstPass	false	If set to true, the login module first attempts to retrieve the user name and password from the shared context before attempting the callback handler.

Properties	Default Value	Description
clearPass	false	If set to true, the login module clears the user name and password in the shared context when calling either commit or abort.
storePass	false	If set to true, the login module stores the user name and password in the shared context after successfully authenticating.

NoSec Configuration Properties

A NoSec provider offers pass-through security for Unwired Server. In development environments, you can apply a NoSec security provider for authentication, authorization, and attribution modules. However, never use this provider in production environments — either for administration, or device user authentication.

The NoSecLoginModule class provides open authentication services, the NoSecAuthorizer class provides authorization services, and the NoSecAttributer provides attribution services. However, you need to configure only authentication properties for a NoSec provider.

Table 3. Authentication properties

Property	Default Value	Description
useUsernameAsIdentity	true	If this option is set to true, the user name supplied in the callback is set as the name of the principal added to the subject.
identity	nosec_identity	The value of this configuration option is used as the identity of the user if either of these conditions is met: • No credentials were supplied. • The useUsernameAsIdentity option is set to false.
useFirstPass	false	If set to true, the login module attempts to retrieve only the user name and password from the shared context. It never calls the callback handler.

Property	Default Value	Description
tryFirstPass	false	If set to true, the login module first attempts to retrieve the user name and password from the shared context before attempting the callback handler.
clearPass	false	If set to true, the login module clears the user name and password in the shared context when calling either commit or abort.
storePass	false	If set to true, the login module stores the user name and password in the shared context after successfully authenticating.

Certificate Authentication Properties

Add and configure authentication provider properties for CertificateAuthenticationLoginModule, or accept the default settings.

Note: This provider cannot be used for administrative security (in the "admin" security configuration).

Table 4. CertificateAuthenticationLoginModule properties

Property	Description
Implementation class	The fully qualified class that implements the login module. `com.sybase.security.core.CertificateAuthenticationLoginModule` is the default class.
Provider type	`LoginModule` is the only supported value.
Control flag	Determines how success or failure of this module affects the overall authentication decision. `optional` is the default value.
Clear password	(Optional) If true, the login module clears the user name and password from the shared context. The default is false.
Store password	(Optional) If true, the login module stores the user name and password in the shared context. The default is false.
Try first password	(Optional) If true, the login module attempts to retrieve user name and password information from the shared context, before using the callback handler. The default is false.

Property	Description
Use first password	(Optional) If true, the login module attempts to retrieve the user name and password only from the shared context. The default is false.
Enable revocation checking	(Optional) Enables online certificate status protocol (OCSP) certificate checking for user authentication. If you enable this option, you muse enable OCSP in Unwired Server. This provider uses the values defined as part of the SSL security profile. Revoked certificates result in authentication failure when both of these conditions are met: • revocation checking is enabled • OCSP properties are configured correctly
Regex for username certificate match	(Optional) By default, this value matches that of the certificates common name (CN) property used to identify the user. If a mobile application user supplies a user name that does not match this value, authentication fails.
Trusted certificate store	(Optional) The file containing the trusted CA certificates (import the issuer certificate into this certificate store). Use this property and `Store Password` property to keep the module out of the system trust store. The default Unwired Server system trust store is `<Unwired-Platform_InstallDir\Servers\Unwired-Server\Repository\Securitytruststore\truststore.jks`. **Note:** This property is required only if Validate certificate path is set to true.
Trusted certificate store password	(Optional) The password required to access the trusted certificate store. For example, import the issuer of the certificate you are trying to authenticate into the shared JDK cacerts file and specify the password using this property. **Note:** This property is required only if Validate certificate path is set to true.
Trusted certificate store provider	(Optional) The keystore provider. For example, "SunJCE." **Note:** This property is required only if Validate certificate path is set to true.

Property	Description
Trusted certificate store type	(Optional) The type of certificate store. For example, "JKS." **Note:** This property is required only if Validate certificate path is set to true.
Validate certificate path	If true (the default), performs certificate chain validation of the certificate being authenticated, starting with the certificate being validated. Verifies that the issuer of that certificate is valid and is issued by a trusted certificate authority (CA), if not, it looks up the issuer of that certificate in turn and verifies it is valid and is issued by a trusted CA. In other words, it builds up the path to a CA that is in the trusted certificate store. If the trusted store does not contain any of the issuers in the certificate chain, then path validation fails. For information about adding a certificate to the truststore, click *Preparing Certificates and Key Pairs*.

See also

- *Certificate Security Provider* on page 70
- *SAP SSO Token Security Provider* on page 70
- *HTTP Basic Security Provider* on page 71
- *Using Keytool to Generate Self-Signed Certificates and Keys* on page 34

SAP SSO Token Authentication Properties

The SAPSSOTokenLoginModule has been deprecated, Use the HttpAuthenticationLoginModule when SAP SSO2 token authentication is required. This authentication module will be removed in a future release.

Table 5. SAPSSOTokenLoginModule properties

Property	Description
Implementation class	(Required) – the fully qualified class that implements the login module. `com.sybase.security.sap.SAPSSOTokenLoginModule` is the default class.
Provider type	(Required and read-only) – `LoginModule` is the only supported value.
Control flag	(Required) – `optional` is the default value. Determines how success or failure of this module affects the overall authentication decision.

Property	Description
SAP server URL	(Required) – the SAP server URL that provides the SSO2 token. This may or may not be the same server that authenticates the user. If providing and authenticating servers are different, you must import the SAP Token provider server certificate or one of its CA signers into the Unwired Server truststore in addition to that of the authenticating server to enable HTTPS communication. In environments where the servers are different, the basic flow is: **1.** Unwired Server passes credentials over HTTPS to the token granting service. **2.** An SSO2Token cookie is returned to Unwired Server. **3.** The SSO2Token flows to the authenticating server, which could be an SAP EIS or a server that hosts a Web service bound to SAP function modules. **Note:** The SAP Server URL must be configured to require BASIC authentication, not just FORM based authentication.
Clear password	(Optional) – if set to `True`, the login module clears the username and password in the shared context.
Disable server certificate validation	(Optional) – the default is False. If set to `True`, disables certificate validation when establishing an HTTPS connection to the SAP server using the configured URL. Set to `True` only for configuration debugging.
SAP server certificate	(Optional) – name of the file containing the SAP certificate's public key in `.pse` format. This is required only when token caching is enabled by setting a SAP SSO token persistence data store value.
SAP server certificate password	(Optional) – password used to access the SAP server certificate.

Property	Description
SAP SSO token persistence data store	(Optional) – JNDI name used to look-up the data source to persist the retrieved SSO2 tokens. Set to "jdbc/default" to store tokens in the Unwired Server CDB. If unconfigured, some caching is still done based on the "Authentication cache timeout interval" property associated with the security configuration setting. If you use the default setting, you do not need to set SAP SSO token persistence data store, SAP server certificate, SAP server certificate password, or Token expiration interval properties. To enable token caching through the SAPSSOTokenLoginModule: **1.** Set the SAP SSO token persistence data store value to "jdbc/default." **2.** Download and install the SAP SSO2 token files. See *Installing the SAP SSO2Token Files on Unwired Server Hosts* in the *Security* guide. **3.** Specify the correct value for the SAP server certificate, SAP server certificate, SAP server certificate password and Token expiration interval properties.
Store password	(Optional) – if set to true, the login module stores the username/password in the shared context after successfully authenticating the user.
Token expiration interval	(Optional) – this property is ignored when the SAP SSO token persistence data store property is not configured. It specifies the token validity period, after which time a new token is retrieved from the SAP EIS. The default value is 120 seconds. Keep in mind that: • The "Token expiration interval" cannot exceed the "Token validity period", which is the amount of time defined in the back-end SAP server for which the token is valid. • The "Authentication cache timeout" property must be less than the "Token expiration interval" property value.

Property	Description
Try first password	(Optional) – if set to `True`, the login module attempts to retrieve the username/password from the shared context, before calling the callback handler.
Use first password	(Optional) – if set to `True`, the login module attempts to retrieve the username/password only from the shared context, and never calls the callback handler.
HTTP connection timeout interval	The value, in seconds, after which an HTTP(s) connection request to the EIS times out. If the HTTP connection made in this module (for either user authentication or configuration validation) does not have a time out set, and attempts to connect to an EIS that is unresponsive, the connection hangs, which could potentially cause Unwired Server to hang. Setting the timeout interval ensures authentication failure is reported without waiting for ever for the server to respond.

See also

- *Certificate Security Provider* on page 70
- *SAP SSO Token Security Provider* on page 70
- *HTTP Basic Security Provider* on page 71

Preconfigured User Authentication Properties

The PreConfiguredUserLoginModule authenticates the Unwired Platform Administrator user whose credentials are specified during installations.

This login module is recommended only to give the Platform administrator access to Sybase Control Center so it can be configured for production use. Administrators are expected to replace this login module immediately upon logging in for the first time.

The PreConfiguredUserLoginModule:

- Provides role based authorization by configuring the provider com.sybase.security.core.RoleCheckAuthorizer in conjunction with this authentication provider.
- Authenticates the user by comparing the specified username/password against the configured user. Upon successful authentication, the configured roles are added as Principals to the Subject.

Table 6. PreConfiguredUserLoginModule properties

Property	Description
User name	A valid user name.
Password	The encoded password hash value.
Roles	Comma separated list of roles granted to the authenticated user for role-based authorization. Platform roles include "SUP Administrator" and "SUP Domain Administrator". Roles are mandatory for "admin" security configuration. For example, if you define "SUP Administrator" to this property, the login id in the created login module has Platform administrator privileges. **Note:** If you use other values, ensure you map Unwired Platform roles to the one you define here.

See also

- *Preconfigured User Login Security Provider* on page 71
- *Adding an LDAP Provider to the Admin Security Configuration* on page 23
- *Mapping Unwired Platform Logical Roles to Physical Roles* on page 24

HTTP Basic Authentication Properties

The HttpAuthenticationLoginModule provider authenticates the user with given credentials (user name and password) against a HTTP(S) enterprise information system (EIS) using a GET against an URL that requires BASIC authentication, and can be configured to retrieve a cookie with the configured name and add it to the JAAS subject to facilitate single sign-on.

Note: If you are using this provider for the "admin" security configuration, ensure you make corresponding provider changes for Sybase Control Center logins. See *Enabling Authentication and RBAC for Administrators* in the *Security* guide.

Table 7. HttpAuthenticationLoginModule properties

Property	Description
URL	The HTTP(S) URL that authenticates the user. For SSO, this is the server URL from which Unwired Server acquires the SSO cookie/token.

Property	Description
Disable certificate validation	(Optional) – the default is False. If set to `True`, disables certificate validation when establishing an HTTPS connection to the EIS using the configured URL. Set to `True` only for configuration debugging.
SSO cookie name	(Optional) – name of the cookie that is set in the session and holds the SSO token for single sign-on. The authentication provider ignores the status code when a SSO cookie is found in the session. If the cookie is found, authentication succeeds regardless of the return status code.
Roles HTTP header	(Optional) – comma separated list of roles granted to the authenticated user for role-based authorization.
Successful connection status code	HTTP status code interpreted as success when connection is established to the EIS. The default is 200.
HTTP connection timeout interval	The value, in seconds, after which an HTTP(s) connection request to the EIS times out. If the HTTP connection made in this module (for either user authentication or configuration validation) does not have a time out set, and attempts to connect to an EIS that is unresponsive, the connection hangs, which could potentially cause Unwired Server to hang. Setting the timeout interval ensures authentication failure is reported without waiting for ever for the server to respond.

See also

- *Certificate Security Provider* on page 70
- *SAP SSO Token Security Provider* on page 70
- *HTTP Basic Security Provider* on page 71

Auditor Filter Properties Reference

(Not applicable to Online Data Proxy) Configure multiple resource classes when defining an auditor for a named security configuration.

Filter resource classes require a specific syntax. Based on that syntax, an audit token is supplied to the core CSI classes. This audit token identifies the source for core audit requests of operations, such as auditing the results for authorization decisions, authentication decisions, in addition to placing information such as active provider information into the audit trail. The audit records have their resource class prefixed by the prefix core. CSI core will able to audit a large number of items.

Syntax

Filter resource classes consist of one or more filter expressions that are delimited by parenthesis (()). Square brackets ([]) denote optional values. The syntax is:
`[key1=value [,key2=value...]].`

The allowed keys are: `ResourceClass`, `Action`, or `Decision`.

This table describes core auditable items:

Resource Class	Action	Description	Attributes
provider	activate	Called when a provider is activated by CSI. The Resource ID is the provider class name.	Generated unique provider identifier.
subject	authentication.provider	The result of a provider's specific authentication request. Depending on the other providers active, the actual CSI request for authentication may not reflect this same decision. **Note:** that this is not a provider-generated audit record. CSI core will generate this audit record automatically after receiving the provider's decision. The resource ID is not used.	• Provider identifier • Decision (yes, no) • Failure reason (if any) • Context ID
subject	authentication	The aggregate decision after considering each of the appropriate provider's authentication decisions. This record shares the same request identifier as the corresponding authentication.provider records. The resource ID is Subject identifier if authentication successful.	• Decision (yes or no) • Context ID

Resource Class	Action	Description	Attributes
subject	authorization.role.provider	The result of a provider's specific role authorization request. The resource ID is the subject ID.	• Provider identifier • Decision (yes, no or abstain) • Role name • Supplied subject identifier (if different from context subject) • Context ID
subject	authorization.role	The result of a resource-based authorization request. The resource ID is the subject ID.	• Resource name • The access requested Decision (yes, no or abstain) • Supplied subject identifier (if different from context subject) • Context ID
subject	authorization.resource	The aggregate decision authorization decision after considering each of the appropriate provider's authorization decision. The resource ID is the subject ID.	• Resource name • Access requested Decision (yes, no) • Supplied subject identifier (if different from context subject) • Context ID
subject	logout	Generated when an authenticated context is destroyed. The resource ID is the subject ID.	Context ID

Resource Class	Action	Description	Attributes
profile	access	Issued when profile is explicitly accessed. The resource ID is the profile name.	• Context ID • Decision (success or not) • Provider identifier that provided the profile attributes (if successful) • Failure reason (if available)
profile	create.cipher	Issued when profile is accessed to create a cipher. The resource ID is the profile name.	• Context ID • Decision (success or not) • Provider identifier that provided the profile attributes (if successful) • Failure reason (if available)
profile	create.digest	Issued when profile is accessed to create message digest. The resource ID is the profile name.	• Context ID • Decision (success or not) • Provider identifier that provided the profile attributes (if successful) • Failure reason (if available)
profile	create.signature	Issued when profile is accessed to create signature. The resource ID is the profile name.	• Context ID • Decision (success or not) • Provider identifier that provided the profile attributes (if successful) • Failure reason (if available)

Resource Class	Action	Description	Attributes
subject	modify.provider	The provider-level record issued for subject modification requests. The resource ID is the subject ID.	• Provider identifier • Decision • Modified attributes • Context ID
subject	modify	Aggregate, generated when a subject modification request is made. The resource ID is the subject ID.	• Decision • Modified attributes • Context ID
subject	create.provider	Provider-level record issued for anonymous self registration requests. The resource ID is the subject identifier.	• Provider identifier • Decision • Subject attributes
subject	create	Aggregate, generated when an anonymous self-registration request is made. The resource ID is the subject identifier.	• Decision • Subject attributes

Examples

- **Example 1 –** enable auditing of all of the CSI core resource classes that involve a deny decision:

  ```
  (ResourceClass=core.*,Decision=Deny)
  ```

- **Example 2 –** enable auditing for all core resource classes where the action is not the subject modification action:

  ```
  Resource=core.*,Action!=subject.modify.*)
  ```

Certificate and Key Management Utilities

Use the certificate management utilities to encrypt Unwired Server ports.

Launch these utilities from the command line; the certificate and key management utilities are not available from any other administration tool.

Use **createcert** and **createkey** for MobiLink and Ultralite server/client purposes (specific for the end-to-end encryption features of RBS clients). For all other purposes, use **keytool** or the PKI system deployed to your environment.

Certificate Creation (createcert) Utility

Generates X.509 certificates or signs pregenerated certificate requests. This utility is located in <*UnwiredPlatform_InstallDir*>\Servers\SQLAnywhere*XX*\BIN*XX*.

You may choose to purchase certificates from a third party. These certificate authorities (CAs) provide their own tools for creating certificates. You can use **createcert** to create certificates for development and testing; you can also use it for production certificates.

Syntax

```
createcert [options]
```

Parameters

- **[*options*] –** these options are available through the **createcert** utility:

Option	Description
`-r`	Creates a PKCS #10 certificate request. **createcert** does not prompt for a signer or any other information used to sign a certificate.
`-s <filename>`	Signs the PKCS #10 certificate request that is in the specified file. The request can be DER or PEM encoded. **createcert** does not prompt for key generation or subject information.

Note: To create a signed certificate, use **createcert** without options. To break the process into two separate steps, for example so one person creates a request and another person signs it, the first person can run **createcert** with `-r` to create a request and the second person can sign the request by running **createcert** with `-s`.

When you run **createcert**, you are asked for all or some of this information, depending on whether you specified `-r`, `-s`, or neither.

- `Choose encryption type` – select RSA.
- `Enter RSA key length (512-16384)` – this prompt appears only if you chose RSA encryption. Specify a length between 512 bits and 16384 bits.
- Subject information – enter this information, which identifies the entity:
  - `Country Code`
  - `State/Province`
  - `Locality`
  - `Organization`
  - `Organizational Unit`
  - `Common Name`
- (Optional) `Enter file path of signer's certificate` – supply a location and file name for the signer's certificate. If you supply this information, the

generated certificate is a signed certificate. If you do not supply this information, the generated certificate is a self-signed root certificate.

- `Enter file path of signer's private key` – supply a location and file name to save the private key associated with the certificate request. This prompt appears only if you supplied a file in the previous prompt.
- `Enter password for signer's private key` – supply the password that was used to encrypt the signer's private key. Supply this password only if the private key was encrypted.
- (Optional) `Serial number` – supply a serial number. The serial number must be a hexadecimal string of 40 digits or less. This number must be unique among all certificates signed by the current signer. If you do not supply a serial number, **createcert** generates a GUID as the serial number.
- `Certificate will be valid for how many years (1-100)` – specify the number of years for which the certificate is valid. After this period, the certificate expires, along with all certificates it signs.
- `Certificate Authority (y)es or (n)o` – indicate whether this certificate can be used to sign other certificates. The default value is no.
- `Key usage` – supply a comma-separated list of numbers that indicate the ways in which the certificate's private key can be used. The default, which depends on whether the certificate is a certificate authority, should be acceptable for most situations.
- `File path to save request` – this prompt appears only if you specify the -r option. Supply a location and file name for the PCKS #10 certificate request. Supply a location and file name in which to save the certificate. The certificate is not saved unless you specify a location and file name.
- `Enter file path to save private key` – supply a location and file name in which to save the private key. Enter a password to protect private key. Optionally, supply a password with which to encrypt the private key. If you do not supply a password, the private key is not encrypted. This prompt appears only if you supplied a file in the previous prompt.
- `Enter file path to save identity` – supply a location and file name in which to save the identity. The identity file is a concatenation of the certificate, signer, and private key. This is the file that you supply to the server at start-up. If the private key was not saved, **createcert** prompts for a password to save the private key. Otherwise, it uses the password provided earlier. The identity is not saved unless you provide a file name. If you do not save the identity file, you can manually concatenate the certificate, signer, and private key files into an identity file.

Examples

- **Example 1: –** creates a self-signed certificate. No file name is provided for the signer's certificate, which makes it a self-signed root certificate.

```
<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers
\SQLAnywhereXX\BINXX>createcert
SQL Anywhere X.509 Certificate Generator Version xx.xx.xx.xx
```

```
Enter RSA key length (512-16384): 1024
Generating key pair...
Country Code: US
State/Province: CA
Locality: Dublin
Organization: MyCompanyCA
Organizational Unit: PTO
Common Name: MyCompanyCA
Enter file path of signer's certificate:
Certificate will be a self-signed root
Serial number [generate GUID]:<enter>
Generated serial number: 3f52ee68c8604e48b8359e0c0128da5a
Certificate valid for how many years (1-100): 10
Certificate Authority (Y/N) [N]: Y
1.  Digital Signature
2.  Nonrepudiation
3.  Key Encipherment
4.  Data Encipherment
5.  Key Agreement
6.  Certificate Signing
7.  CRL Signing
8.  Encipher Only
9.  Decipher Only
Key Usage [6,7]: <enter>
Enter file path to save certificate: rsa_root.crt
Enter file path to save private key: rsa_key.key
Enter password to protect private key: <MyPwd>
Enter file path to save identity: id.pem
```

- **Example 2: Generating an enterprise root certificate –** to generate an enterprise root certificate (a certificate that signs other certificates), create a self-signed root certificate with a CA. The procedure is similar to Example 1. However, the response to the CA prompt should be `yes` and choice for roles should be option `6,7` (the default).

```
Certificate Authority (Y/N) [N]: y
1.  Digital Signature
2.  Nonrepudiation
3.  Key Encipherment
4.  Data Encipherment
5.  Key Agreement
6.  Certificate Signing
7.  CRL Signing
8.  Encipher Only
9.  Decipher Only
Key Usage [6,7]: 6,7
```

See also

- *Encrypting Synchronization for Replication Payloads* on page 79

Key Creation (createkey) Utility

Creates RSA and ECC key pairs for use with Unwired Server end-to-end encryption. This utility is located in `<UnwiredPlatform_InstallDir>\Servers\SQLAnywhereXX\BINXX`.

Syntax

```
createkey
```

When you run **createkey**, you are prompted for this information:

- `Choose encryption type` – choose RSA.
- `Enter RSA key length (512-16384)` – this prompt appears only if you chose RSA encryption. Choose a length between 512 bits and 16384 bits.
- `Enter file path to save public key` – specify a file name and location for the generated PEM-encoded public key. This file is specified on the MobiLink client by the e2ee_public_key protocol option.
- `Enter file path to save private key` – specify a file name and location for the generated PEM-encoded private key. This file is specified on the MobiLink server via the e2ee_private_key protocol option.
- `Enter password to protect private key` – optionally, supply a password with which to encrypt the private key. The private key is not encrypted if you do not supply a password. This password is specified on the MobiLink server via the e2ee_private_key_password protocol option.

Examples

- **Example –** creates an RSA key pair:

```
>createkey
SQL Anywhere Key Pair Generator Version 11.0.0.2376
Enter RSA key length (512-16384): 2048
Generating key pair...
Enter file path to save public key: rsa_key_public.key
Enter file path to save private key: rsa_key_private.key
Enter password to protect private key: pwd
```

See also

- *Encrypting Synchronization for Replication Payloads* on page 79

Truststore and Keystore Properties

The `<UnwiredPlatform_InstallDir>\SCC-XX\services\Messaging\lib\eas\lib\Repository\Server\EmbeddedJMS\Instance\com\sybase\djc\server\ApplicationServer\EmbeddedJMS.properties` file contains

properties for the truststore and keystore that you can configure. These are the keystore/truststore for the Sybase Control Center Windows service.

Change the default properties for:

Property	Default	Description
keyStore	*<UnwiredPlatform_InstallDir>*\SCC-*XX*\services\Messaging\lib\eas\lib\Repository\Security\keystore.jks	The default location of the keystore used by Sybase Control Center. The analogous location for Unwired Server is *<UnwiredPlatform_InstallDir>*/Servers/UnwiredServer/Repository/Security/keystore.jks
keyStorePassword	changeIt	The password used to unlock the keystore.
trustStore	*<UnwiredPlatform_InstallDir>*\SCC-*XX*\services\Messaging\lib\eas\lib\Repository\Security\truststore.jks	The default location of the truststore used by Sybase Control Center. The truststore is used when Sybase Control Center makes an out-bound connection over SSL to another server with a server certificate. Sybase Control Center checks that the server certificate is in the truststore, or is signed by a CA certificate in the truststore. The analogous location for Unwired Server is *<UnwiredPlatform_InstallDir>*/Servers/UnwiredServer/Repository/Security/truststore.jks
trustStorePassword	changeIt	The password used to unlock the truststore.

Index

M

P

R

S

T

U

X