Genesys Knowledge Center Deployment Guide

Transport Layer Security (TLS) with Genesys Servers

5/7/2025

Transport Layer Security (TLS) with Genesys Servers

Genesys Knowledge Center supports the Transport Layer Security (TLS) protocol to secure data exchanged with other Genesys components. For details about TLS, see the Genesys 8.1 Security Deployment Guide. You can configure TLS for Knowledge Center by completing the procedures on this page.

Configuring TLS for Genesys Servers

To configure the TLS parameters for Genesys servers, see Introduction to Genesys Transport Layer Security.

Configuring TLS for Genesys Knowledge Center Server

To enable TLS support for the Genesys Knowledge Center Server, you must do the following:

1. Have properly installed a trusted certificates for the Genesys server. For more information, please see Certificate Generation and Installation.
2. Configure TLS options for the Genesys Knowledge Center Server application.
3. Configure the appropriate connections between the Genesys Knowledge Center Server application and the necessary Genesys servers through secure ports. For example, by setting a secure Config Server port in the *Server Installation Folder*/server/setenv.bat file in the **PRIMARY_CFGSERVER_PORT** variable.

Configuring Secure Connections to Configuration Server

To configure a secured connection from Genesys Knowledge Center Server to Configuration Server use the following TLS-related configuration options in the **setenv.bat/sh** configuration:

Parameter Name	Acceptable Values	Purpose
PRIMARY_CFGSERVER_CONNECTION_MODE	TLS_UPGRADE or UNSECURED by default	Set this option to enable secured connection **Important** Incorrect setting of this parameter can lead to inability to establish a connection with the server
PROVIDER	PEM, JKS, MSCAPI, PKCS11	Type of used security provider

Parameter Name	Acceptable Values	Purpose
TRUSTED_CA	valid file name (including path)	Path to trusted CA PEM file or JKS truststore file or SHA-1 Thumbprint for MSCAPI storage. Specifies the name of the trusted store file which holds the public certificate to verify the server. Applicable for PEM and JKS trusted storage types only.
TRUSTSTORE_PASSWORD	n/a	Password for the JKS trusted storage. Provide password only if trusted CA is in the JKS format.
In case of enabled mutual TLS, configure the following options:		
CERTIFICATE	n/a	Client certificate file in PEM format or JKS keystore file or SHA-1 Thumbprint for MSCAPI storage.
PRIVATE_KEY	n/a	Unencrypted private key in PEM format or Certificate SHA-1 Thumbprint for MSCAPI storage. Ignored for JKS storage.
KEYSTORE_PASSWORD	n/a	Provide password if key storage is in the JKS format.
KEYENTRY_PASSWORD	n/a	Provide password if private key encrypted by its own password.

Configuring TLS Options

For connections with other Genesys servers, configure **Connections** of the Knowledge Center Cluster (8.5.1+) application through secure ports. The Genesys Knowledge Center Server Node includes the following TLS-related configuration options in its security section.

Parameter Name	Acceptable Values	Purpose
tls	Boolean value. Possible values are "1"/"0", "yes"/"no", "on"/"off", "true"/"false". Example: • "tls=1"	Client: 1 - perform TLS handshake immediately after connecting to server. 0 – do not turn on TLS immediately but autodetect can still work.
provider	"PEM", "MSCAPI", "PKCS11" Not case-sensitive. Example: • "provider=MSCAPI"	Explicit selection of security provider to be used. For example, MSCAPI and PKCS11 providers can contain all other parameters in their internal database. This parameter allow configuration of TLS through security provider tools.
certificate	PEM provider: path to a X.509 certificate file in PEM format.	Specifies location of X.509 certificate to be used by

Parameter Name	Acceptable Values	Purpose
	Path can use both forward and backward slash characters. MSCAPI provider: thumbprint of a certificate – string with hexadecimal SHA-1 hash code of the certificate. Whitespace characters are allowed anywhere within the string. PKCS11 provider: this parameter is ignored. Examples: • "certificate= C:\certs\client-cert-3-cert.pem" • "certificate=A4 7E A6 E4 7D 45 6A A6 2F 15 BE 89 FD 46 F0 EE 82 1A 58 B9"	application. MSCAPI provider keeps certificates in internal database and can identify them by hash code; so called thumbprint. In Java, PKCS#11 provider does not allow selection of the certificate; it must be configured using provider tools. **Note:** When using autodetect (upgrade) TLS connection, this option MUST be specified in application configuration, otherwise Configuration Server would return empty TLS parameters even if other options are set.
certificate-key	PEM provider: path to a PKCS#8 private key file without password protection in PEM format. Path can use both forward and backward slash characters. • MSCAPI provider: this parameter is ignored; key is taken from the entry identified by "certificate" field. • PKCS11 provider: this parameter is ignored. Examples: • "certificate-key= C:\certs\client-cert-3-key.pem"	Specifies location of PKCS#8 private key to be used in pair with the certificate by application. MSCAPI provider keeps private keys paired with certificates in internal database. In Java, PKCS#11 provider does not allow selection of the private key; it must be configured using provider tools.
trusted-ca	PEM provider: path to a X.509 certificate file in PEM format. Path can use both forward and backward slash characters. MSCAPI provider: thumbprint of a certificate – string with hexadecimal SHA-1 hash code of the certificate. Whitespace characters are allowed anywhere within the string. PKCS11 provider: this parameter is ignored. Examples: • "trusted-ca= C:\certs\ca.pem" • "trusted-ca=A4 7E A6 E4 7D 45 6A A6 2F 15 BE 89 FD 46 F0 EE 82 1A 58 B9"	Specifies location of a X.509 certificate to be used by application to validate remote party certificates. The certificate is designated as Trusted Certification Authority certificate and application will only trust remote party certificates signed with the CA certificate. MSCAPI provider keeps CA certificates in internal database and can identify them by hash code; so called thumbprint. In Java, PKCS#11 provider does not allow selection of the CA certificate; it must be configured using provider tools.

Parameter Name	Acceptable Values	Purpose
tls-mutual	Boolean value. Possible values are "1"/"0", "yes"/"no", "on"/"off", "true"/"false". Example: • "tls-mutual=1"	Has meaning only for server application. Client applications ignore this value. When turned on, server will require connecting clients to present their certificates and validate the certificates the same way as client applications do.
tls-crl	All providers: path to a Certificate Revocation List file in PEM format. Path can use both forward and backward slash characters. Example: • "tls-crl= C:\certs\crl.pem"	Applications will use CRL during certificate validation process to check if the (seemingly valid) certificate was revoked by CA. This option is useful to stop usage of leaked certificates by unauthorized parties.
tls-target-name-check	"host" or none. Not case-sensitive. Example: • "tls-target-name-check=host"	When set to "host", enables matching of certificate's Alternative Subject Name or Subject fields against expected host name. PSDK supports DNS names and IP addresses as expected host names.
cipher-list	String consisting of space-separated cipher suit names. Information on cipher names can be found online. Example: • "cipher-list= TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA"	Used to calculate enabled cipher suites. Only ciphers present in both the cipher suites supported by security provider and the cipher-list parameter will be valid.
fips140-enabled	Boolean value. Possible values are "1"/"0", "yes"/"no", "on"/"off", "true"/"false". Example: • "fips140-enabled=1"	PSDK Java: when set to true, effectively is the same as setting "provider=PKCS11" since only PKCS11 provider can support FIPS-140. If set to true while using other provider type, PSDK will throw exception.
sec-protocol	String value. Possible values are "SSLv23", "SSLv3", "TLSv1", "TLSv11", "TLSv12". Example: • "sec-protocol=TLSv1"	Starting with PSDK release 8.5.1, an application can specify the exact protocol to send and accept secure connection requests on one or more of its connections.

See Configuring Trusted Stores below for details about configuration for a specific type of store (PEM,

JKS, MSCAPI).

Configuring Trusted Stores

PEM Trusted Store

PEM stands for "Privacy Enhanced Mail", a 1993 IETF proposal for securing email using public-key cryptography. That proposal defined the PEM file format for certificates as one containing a Base64-encoded X.509 certificate in specific binary representation with additional metadata headers.

PEM certificate trusted store works with CA certificate from an X.509 PEM file. It is a recommended trusted store to work on Linux systems.

Complete the steps below to work with the PEM certificate trusted store:

Start

1. Configure TLS for Genesys servers to use certificates signed by CA certificate **certificateCA.crt**.
2. Place the trusted CA certificate in PEM format on the Genesys Knowledge Center Server application host. To convert a certificate of another format to .pem format you can use the OpenSSL tool. For example:
   - Convert a DER file (.crt .cer .der) to PEM:
     ```
     openssl x509 -inform der -in certificateCA.crt -out certificateCA.pem
     ```
   - Convert a PKCS#12 file (.pfx .p12) containing a private key and certificates to PEM:
     ```
     openssl pkcs12 -in certificateCA.pfx -out certificateCA.pem -nodes
     ```
     You can add **-nocerts** to only output the private key or add **-nokeys** to only output the certificates.
3. In Genesys Administrator, navigate to **Provisioning > Environment > Applications** and open your Knowledge Center Server application.
4. Click the **Options** tab and navigate to the security section.
5. Set the **trusted-ca-type** option to PEM.
6. Set the **trusted-ca** option to the path and file name for your trusted CA in PEM format on the Genesys Knowledge Center Server application host.
7. Click **Save & Close**.

End

JKS Trusted Store

A Java KeyStore (JKS) is a repository of security certificates used, for instance, in SSL/TLS encryption. The Java Development Kit provides a tool named keytool to manipulate the keystore.

Complete the steps below to work with the JKS certificate trusted store:

Start

1. Configure TLS for Genesys servers to use certificates signed by CA certificate **certificateCA.crt**.

2. Import the CA certificate to an existing Java keystore using keytool:
   - Run the `keytool` command with option `-alias` set to `root`:
     ```
     keytool -import -trustcacerts -alias root -file certificateCa.crt -keystore
     /path/to/keysore/keystore.jks
     ```
   - Enter the keystore password in command line prompt - for example:
     ```
     Enter keystore password: somepassword
     ```
3. In Genesys Administrator, navigate to **Provisioning > Environment > Applications** and open your Knowledge Center Server application.
4. Click the **Options** tab and navigate to the security section.
5. Set the **trusted-ca-type** option to JKS.
6. Set the **trusted-ca** option to the path and file name for your JKS trusted storage type on the Genesys Knowledge Center Server application host.
7. Set the **trusted-pwd** option to the password defined for your keystore in Step 2.
8. Click **Save & Close**.

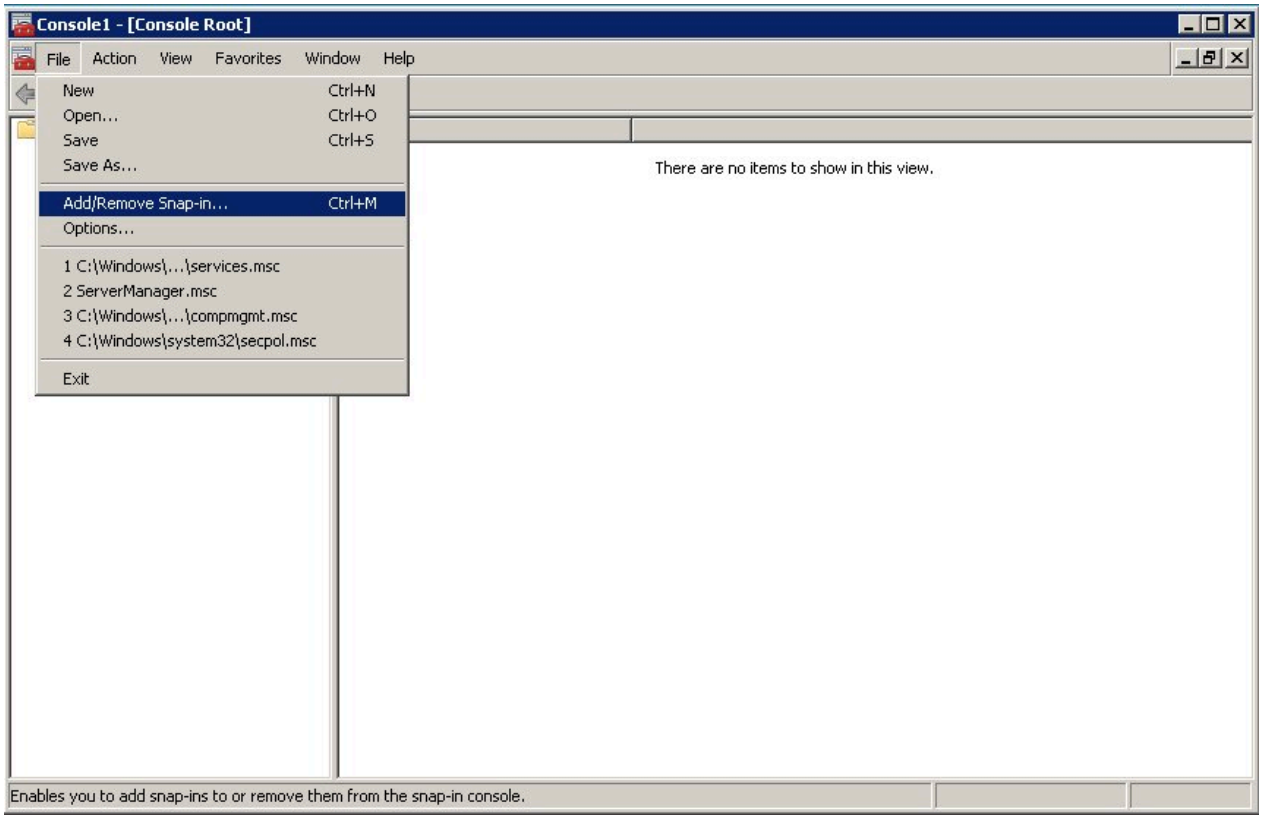
End

MSCAPI Trusted Store

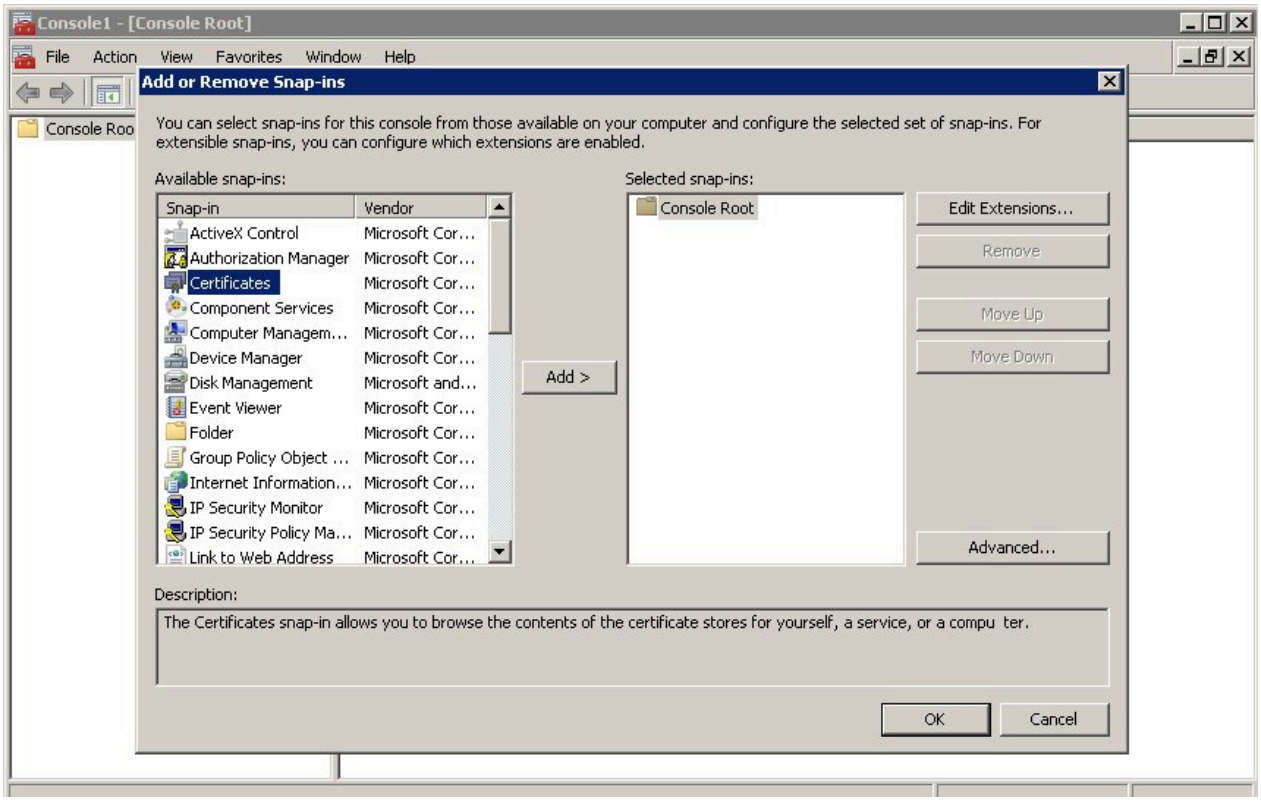
Complete the steps below to work with the MSCAPI certificate trusted store:

Start

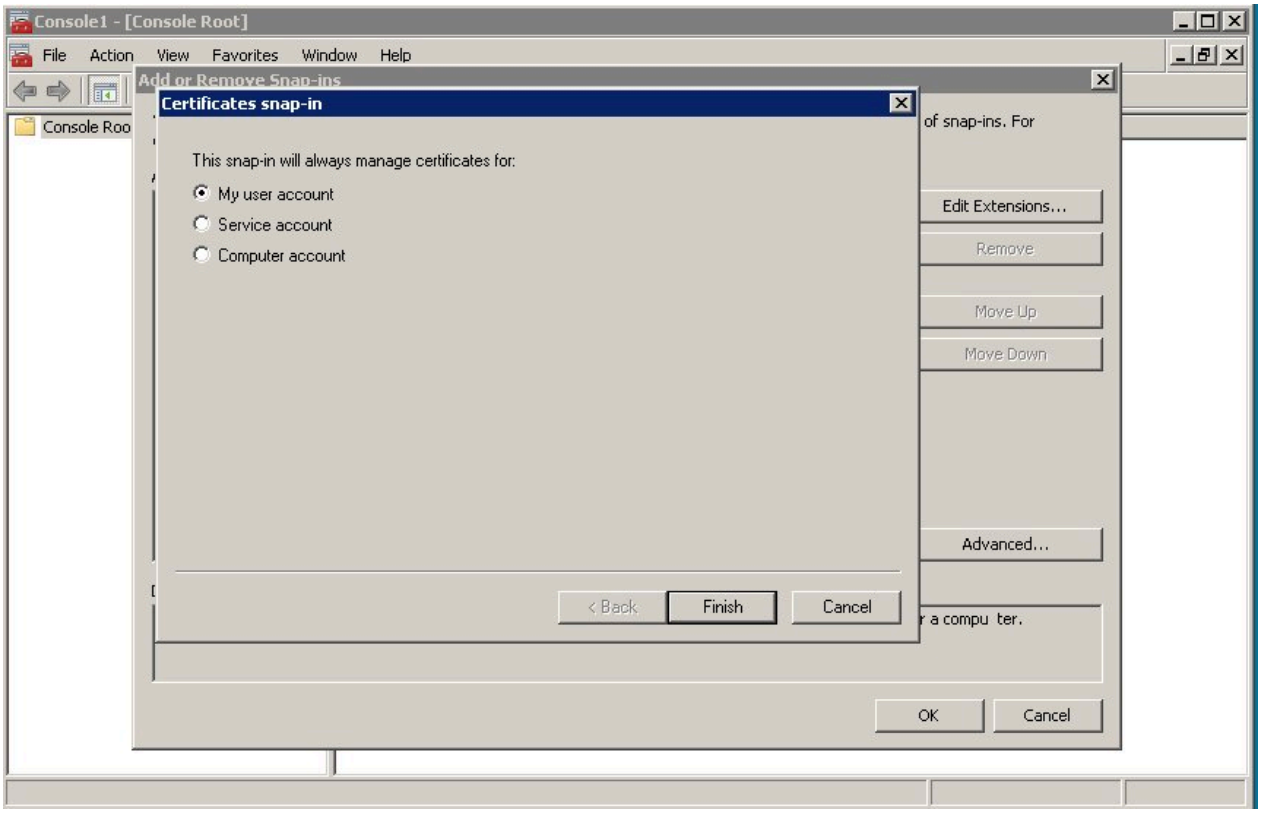
1. Configure and tune TLS for Genesys servers to use certificates signed by the same CA.
2. If the Knowledge Center Server is running on a different host, copy the trusted CA certificate to this host.
3. Import the CA certificate to WCS via Certificates Snap-in on the Knowledge Center Server host by launching the MMC console. Enter `mmc` at the command line.
4. Select **File > Add/Remove Snap-in...** from the main menu.



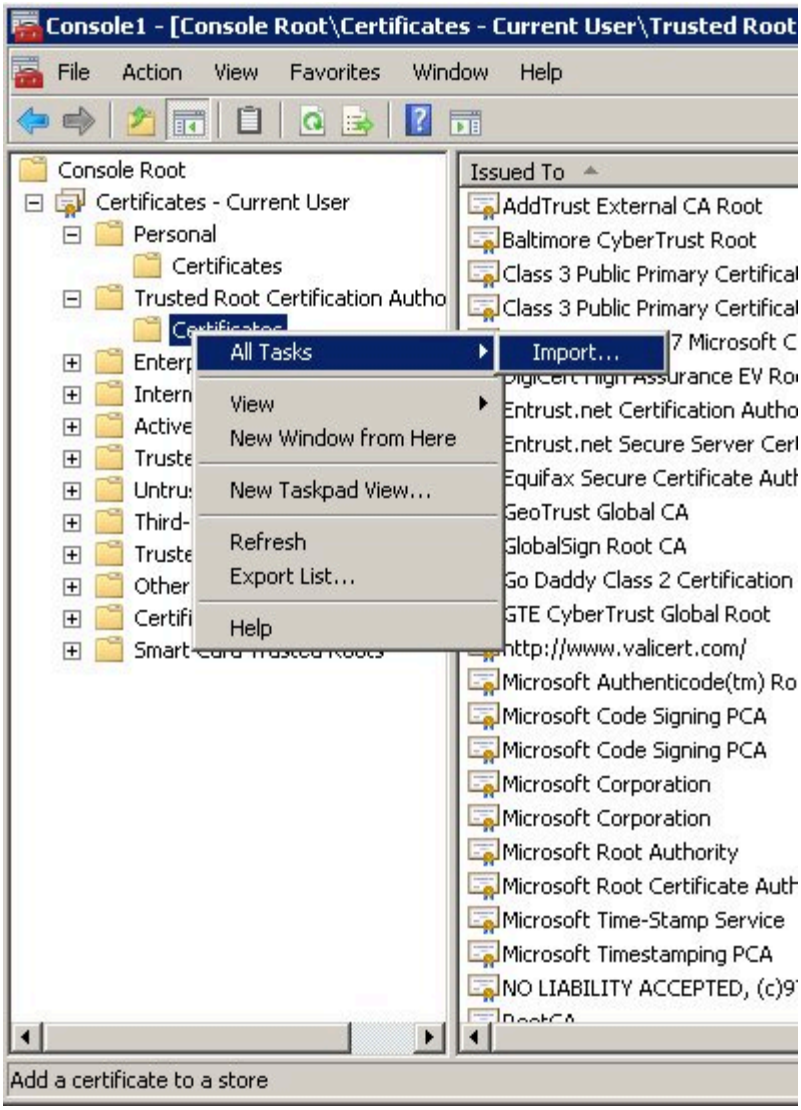
5. Select **Certificates** from the list of available snap-ins and click **Add**.



6. Select the account to manage certificates for and click **Finish**. It is important to place certificates under the correct Windows account. Some applications are run as services under the Local Service or System account, while others are run under user accounts. The account chosen in MMC must be the same as the account used by the application which certificates are configured for, otherwise the application will not be able to access this WCS storage.



7. Click **OK**.
8. Import a certificate. Right-click the "Trusted Root Certification Authorities/Certificates" folder and choose `All Tasks > Import...` from the context menu. Follow the steps presented by the Certificate Import Wizard. Oonce finished the imported certificate appears in the certificates list.



9. In Genesys Administrator, navigate to **Provisioning > Environment > Applications** and open your Knowledge Center Server application.
10. Click the **Options** tab and navigate to the security section.
11. Set the **trusted-ca-type** option to MSCAP.
12. Click **Save & Close**.

End

Configuring TLS for a server running Windows

By default, Genesys Knowledge Server as a Windows Service runs without a TLS connection. To configure a secure connection from Genesys Knowledge Center Server to a Configuration Server while running as a Windows Service you need to update the installed default service for Genesys

Knowledge Center Server.

In order to do this you will need to:

1. Remove Genesys Knowledge Center Server Windows Service, which was configured in the installation package.
    1. Run the Windows Command Prompt (cmd.exe)
    2. Go to *<Knowledge Center Server installation folder>/server*
    3. Run the next command: **server.bat** remove
2. Configure a secure connection settings in **setenv.bat** to Genesys Configuration Server, as described in Configuring Secure Connections to Configuration Server.
3. Re-install the Windows Service for Genesys Knowledge Center Server, now with the secure connection configured to Genesys Configuration Server.
    1. Run Windows Command Prompt (cmd.exe)
    2. Go to *<Knowledge Center Server installation folder>/server*
    3. Run the next command: **server.bat** install