



Understanding CiscoWorks Security

The CiscoWorks Server provides some of the security controls necessary for a web-based network management system. It also relies heavily on the end user's own security measures and controls to provide a secure computing environment for CiscoWorks applications.

The CiscoWorks Server provides and requires three levels of security to be implemented to ensure a secure environment:

- **General Security**—Partially implemented by the client components of CiscoWorks and by the system administrator.
- **Server Security**—Partially implemented by the server components of CiscoWorks and by the system administrator.
- **Application Security**—Implemented by the client and server components of the CiscoWorks applications.

For more information on security related features see [Setting up Security](#).

The following sections describe the general and server security levels.

General Security

The CiscoWorks Server provides an environment that allows the deployment of web-based network management applications.

Web access provides an easy-to-use and easy-to-access computing model that is more difficult to secure than the standard computing model that only requires a system login to execute applications.

The CiscoWorks Server also provides security mechanisms (authentication and authorization) used to prevent unauthenticated access to the CiscoWorks Server and unauthorized access to CiscoWorks applications and data.

However, CiscoWorks applications can change the behavior and security of your network devices. Therefore, it is critical to limit access to applications and servers as follows:

- Limit access to personnel who need access to applications or the data that the applications provide.
- Limit CiscoWorks Server logins to just the systems administrator.
- Limit connectivity access to the CiscoWorks Server by putting it behind a firewall.

Server Security

The CiscoWorks Server uses the basic security mechanisms of the operating system to protect the code and data files that reside on the server. The following CiscoWorks Server security control elements apply:

- [Server-Imposed Security](#)
- [System Administrator-Imposed Security](#)

Server-Imposed Security

The CiscoWorks Server has many dimensions, such as:

- [Files, File Ownership, and Permissions](#)
- [Runtime](#)
- [Remote Connectivity](#)
- [Access to Systems Other Than the CiscoWorks Server](#)
- [Access Control](#)

Files, File Ownership, and Permissions

The following describes the file ownership and permissions.

- UNIX Systems—CiscoWorks must be installed by a user with root privilege. It should be installed as the user, casuser with a casusers group. If the system administrator needs to work on casuser files, a user with a name chosen by the system administrator, must be created and added to the casusers group.

All files and directories are owned by casuser with group equal to casusers. Temporary files are created as the user casuser with permissions set to read-write for the user casuser and read for members of group casusers.

The only exception to this rule is the log files created by the CiscoWorks web server and diskwatcher. The CiscoWorks web server and diskwatcher must be started as *root*. Therefore, their log files are owned by the user *root* with “group=casusers.”

- Windows Systems—CiscoWorks must be installed by the administrator and must be installed as the user *casuser*.
 - If it is a new installation, the system displays a message prompting you to either create or to cancel the process. You can enter the password or can be generated.
 - If it is not a new installation, the system displays a message prompting you to either continue resetting the password or to retain the old password.

The CiscoWorks Server uses the password but the *casuser* user is never intended as a general user of the Windows system. No user is required to log on the Windows system as *casuser*.

All files and directories are owned by the user *casuser*. Read and write access are restricted to the user *casuser* and the administrator. Temporary files are created as the user *casuser* with permissions set to read-write for the user *casuser*.

The CiscoWorks Server relies on the security mechanisms of the NTFS filesystem to provide access control on Windows systems. If CiscoWorks is installed on a FAT filesystem, most security assumptions made about controlled access to files and network management data are not valid.

Runtime

This describes the runtime activities.

- UNIX Systems—Typically CiscoWorks back-end processes are executed with permissions set to the user ID of the binary file.

For example, if user “Joe” owns an executable file, it will be executed by the CiscoWorks daemon manager under the user ID of “Joe”).

The exception are files owned by the root user ID. To prevent a potentially harmful program from being executed by the daemon manager with root permissions, the daemon manager will execute only a limited set of CiscoWorks programs that need root privilege.

This list is not documented to preclude any user from trying to impersonate these programs.

All back-end processes are executed with a *umask* value of 027. This means that all files created by these programs are created with permissions equal to “rwxr-x,” with an owner and group of the user ID and group of the program that created it. Typically this will be “*casuser*” and “group=casusers.”

CiscoWorks foreground processes (typically *cgi-bin* programs or *servlets*) are executed under the control of the web server’s child processes or the *servlet engine*, which all run as the user *casuser*.

CiscoWorks uses standard UNIX *tftp* and *rcp* services. CiscoWorks also requires that user *casuser* have access to the directories that these services read and write to.

The CiscoWorks Server must allow the user *casuser* to run ***cron*** and ***at*** jobs to enable the Resource Manager Essentials Software Management application to run image download jobs.

- Windows—CiscoWorks back-end processes are executed with permissions set to the user *casuser*. Some of the special CiscoWorks Server processes are run as a service under the *localsystem* user ID.

These processes include:

- Daemon Manager
- Web server
- Servlet engine

- Rcp/rsh service
- Tftp service
- Corba service
- Database engine

CiscoWorks foreground processes (typically cgi-bin programs or servlets) are executed under the control of the web server and the servlet engine that run as the user localsystem.

The local system user has special permissions on the local system but does not have network permissions.

CiscoWorks provides several services for RCP, TFTP communication with devices. These services are targeted for use by CiscoWorks applications, but can be used for purposes other than network management.

The CiscoWorks Server uses the **at** command to run software update jobs for the Resource Manager Essentials Software Image Manager application. Jobs run by the **at** command, run with system level privileges.

Remote Connectivity

The remote connectivity details for Windows and Solaris are:

- UNIX Systems—The CiscoWorks daemon manager only responds to requests to start, stop, register, or show status for CiscoWorks back-end processes from the CiscoWorks Server.
- Windows Systems—The CiscoWorks daemon manager only responds to requests to start, stop, register, or show status for CiscoWorks back-end processes from the CiscoWorks Server.

Access to Systems Other Than the CiscoWorks Server

The access details for UNIX and Windows are:

- UNIX Systems—Systems used by the CiscoWorks Server as remote sources of device information for importing into the RME Inventory Manager application must allow the user casuser to perform remote shell operations on the user who owns the device information.
- Windows Systems—Systems used by the CiscoWorks Server as remote sources of device information for importing into the RME Inventory Manager application must allow the user casuser to perform remote shell operations on the user who owns the device information.

Access Control

The access control details are:

- UNIX Systems—The UNIX user casuser is a user ID that is not typically enabled for login. Using this user ID as the user ID under which to install the CiscoWorks Server software simplifies the installation process and ensures limited access to the CiscoWorks Server. This is because casuser is not a valid login ID as there is no password assigned to it. However, the casuser user on UNIX systems can perform system and possibly network-wide operations that could be harmful to the system or the network.

- Windows Systems—The user casuser, created as part of the install process, has no special permissions or considerations on a system so it is a “safe” user ID under which to execute the CiscoWorks Server and application code. The localsystem user can perform harmful system operations.

Therefore, consider that by using the localsystem user ID to run some of the backend processes, the localsystem user ID cannot perform network operations.

**Note**

The system administrator should review and adopt the security recommendations in “[System Administrator-Imposed Security](#)” section on page A-5.

System Administrator-Imposed Security

To maximize CiscoWorks Server security, follow these security guidelines:

- Do not allow users other than the systems administrator to have a login on the CiscoWorks Server.
- Do not allow the CiscoWorks Server file systems to be mounted remotely with NFS or any other file-sharing protocol.
- Limit remote access (for example, FTP, RCP, RSH) to the CiscoWorks Server to those users who are permitted to log in to the CiscoWorks Server.
- Place your network management servers behind firewalls to prevent access to the systems from outside of your organization.
- Change the database password after installation and periodically based on your company’s security policies.
- Back up the security certificates in a safe location, if you are using SSL in CiscoWorks Server.

Connection Security

The CiscoWorks server uses Secure Socket Layer (SSL) encryption to provide secure connection between the client browser and management server, and Secure Shell (SSH) to provide secure access between the management server and devices.

Security Certificates

Security certificates are similar to digital ID cards. They prove the identity of the server to clients. Certificates are issued by Certificate Authorities (CAs) such as VeriSign® or Thawte.

A certificate vouches for the identity and key ownership of an individual, a computer system (or a specific server running on that system), or an organization. It is a general term for a signed document.

Typically, certificates contain the following information:

- Subject public key value.
- Subject identifier information (such as the name and e-mail address).
- Validity period (the length of time that the certificate is considered valid).
- Issuer identifier information.
- The digital signature of the issuer. This attests to the validity of the binding between the subject public key and the subject identifier information.

A certificate is valid only for the period of time specified within it. Every certificate contains Valid From and Valid To dates, which are the boundaries of the validity period.

For example, a user's certificate verifies that the user owns a particular public key. The server certificate for the server named myserver.cisco.com verifies that a specific public key belongs to this server.

Certificates can be issued for a variety of functions such as web user authentication, web server authentication, secure e-mail (S/MIME), IP Security, Transaction Layer Security (TLS), and code signing.

CiscoWorks Server supports security certificates for authenticating secure access between client browser and management server.

CiscoWorks supports Self signed certificates and provides an option to create self-signed certificates. For more information, see [Creating Self Signed Certificates](#).

Terms and Definitions

The following explains the terms and corresponding definitions in CiscoWorks:

- [Secure Socket Layer \(SSL\)](#)
- [Public Key, Private Key](#)
- [Secure Shell \(SSH\)](#)
- [PKCS#8](#)
- [Base64- Encoded X.509 Certificate Format](#)
- [Certificate Authority](#)
- [CiscoWorks TrustStore or KeyStore](#)

Secure Socket Layer (SSL)

Secure Socket Layer (SSL) is an application-level protocol that enables secure transactions of data through privacy, authentication, and data integrity. It relies upon certificates, public keys, and private keys.

Public Key, Private Key

Public and private keys are the ciphers used to encrypt and decrypt information. While the public key is shared quite freely, the private key is never given out. Each public-private key pair works together. Data encrypted with the public key can only be decrypted with the private key.

Secure Shell (SSH)

Secure Shell (SSH) is an application and a protocol that provide a secure replacement to the Berkeley r-tools. The protocol secures the sessions using standard cryptographic mechanisms, and the application can be used similarly to the Berkeley rexec and rsh tools.

Two versions of SSH are currently available: SSH Version 1 and SSH Version 2. Common Services 3.0.3 supports SSH Version 1.

PKCS#8

Public-Key Cryptography Standards (PKCS) are a set of standards for public-key cryptography, developed by RSA Laboratories in cooperation with an informal consortium, originally including Apple, Microsoft, DEC, Lotus, Sun and MIT.

The PKCS have been cited by the OIW (OSI Implementers' Workshop) as a method for implementation of OSI standards.

The PKCS are designed for binary and ASCII data; PKCS are also compatible with the ITU-T X.509 standard. The published standards are PKCS #1, #3, #5, #7, #8, #9, #10, #11, #12, and #15; PKCS #13 and #14 are currently being developed.

PKCS #8 describes a format for private key information. This information includes a private key for some public-key algorithm, and optionally a set of attributes.

Base64- Encoded X.509 Certificate Format

X.509 certificate format is an emerging certificate standard. It is part of the OSI group of standards. X.509 certificates are very clearly defined using a notation called ASN.1 (Abstract Syntax Notation 1) which specifies the precise kinds of binary data that make up the certificate.

ASN.1 can be encoded in many ways, but the emerging standard is an encoding called DER (Distinguished Encoding Rules), which results in a compact binary certificate.

For e-mail exchange purposes the binary certificate is often Base64 encoded, resulting in an ASCII text document that looks like the following:

```
-----BEGIN CERTIFICATE-----
MIIC4jCCAkugAwIBAgIEA0E1UDANBgkqhkiG9w0BAQBhMCM
VVMxCzAJBgNVBAGTAKNBMRERwDwYDVQQHEwhTYNQ21z
Y28gU31zdGVtczENMAAGA1UECXMERU1CVTEqMCgG0ZXXN0
MiBDZXJ0aWZpY2F0ZSBNYW5hZ2VyMB4XDTAyMDas3DA4
NTgwOVowgYIxCzAJBgNVBAYTAklOMQswCQYDVQQIQ2h1
bm5haTEEMMAoGA1UEChMDSSENMQ0wCwYDVQQLEtZGlu
YWthci1wYzEhMB8GCSqGSIb3DQEJARYSc2RpbmFrYXfMA0G
CSqGSIb3DQEBAQUAA4GNADCBiQKBgQDV1o9PyO7txr5vme
FU/f9tp5To/HaLIWHVx9zpihPnVuKaep8kceX08Sed8crXeU8BP
9qHoIswGn1oJEGFXm9gs5uupJyAgeDd609eCuQbiSKgE1sFGFSL
xNGQJZbCrQIDAQABo2UwYzARBglghkgBhvhCAQEEB/BAQD
-----END CERTIFICATE-----
```

CiscoWorks requires the Certificates to be uploaded in this format.

**Note**

Other certificate formats such as PKCS#7 also have similar formats. Hence it is important that you confirm with the CA the format of the certificate, and request specifically for Base64 Encoded X.509Certificates formats.

Certificate Authority

A certificate authority (CA) is an authority in a network that issues and manages security credentials and public keys for message encryption.

As part of a public key infrastructure (PKI), a CA checks with a registration authority (RA) to verify information provided by the requestor of a digital certificate. If the RA verifies the requestor's information, the CA then issues a certificate.

CiscoWorks TrustStore or KeyStore

CiscoWorks TrustStore or KeyStore is the location where CiscoWorks maintains the list of Certificates that it trusts.

On Windows: *NMSROOT\MDC\Apache\conf\ssl*

On Solaris: *NMSROOT/MDC/Apache/conf/ssl*