



Shepherd.Chat Service

J&J Computer Consulting Shepherd Server Publications

Overview

The Shepherd.Chat Service implements J&J Computer Consulting's j-Chat protocol within the Shepherd framework. Shepherd.Chat expands on j-Chat's offerings using Shepherd's rich security system to provide enhanced security for each chat room.

Installation

Unzip the Shepherd.Chat distribution to the Shepherd installation directory. This will install a sample chat configuration for the directory in `setup\jchat.ldif`, the Service initialization file in `services\jchat.svc`, and the dynamic link library for the Shepherd.Chat Service in `services\jchat.dll`. Use the Service initialization file `jchat.svc` to setup Shepherd.Chat in the Directory. Refer to the "Shepherd Installation and Configuration" for further information on configuring a ShepherdService object.

Configuration

Outside the ShepherdService object, Shepherd.Chat requires the appropriate configuration of three additional kinds of objects to work properly. These are the `internetDomain`, `jChatRoom`, and `ServiceLog`.

internetDomain

Shepherd.Chat uses the `internetDomain` objects to load support for each virtual host on the system. Using the `hostServer` attribute required of all `internetDomain` objects, Shepherd.Chat finds each domain listed for the server. Shepherd.Chat can work with hosts aliased to the same IP address or with hosts assigned to individual IP addresses.

The location of the `internetDomain` object is also important. Whatever container the `internetDomain` object is located in is the container that will be used by default to authenticate users to the chat rooms.

jChatRoom

The `jChatRoom` object represents an individual room in the Shepherd.Chat Service. Additional attributes have been included with Shepherd.Chat to facilitate the requirements of the j-Chat protocol. These attributes include:

- **chatLogonArea** - Identifies if users are unique to the room or the host. Possible values

- include "room" or "host". Current j-Chat clients only allow host authenticated logons.
- **chatRoomSecurity** - Identifies the security level of the room. If this attribute contains the value "public", unrestricted access to the room will be allowed. If the value is "private write", anyone attempting to post a message to the chat room will be authenticated and checked for Write access to the jChatRoom object. If the value is "private" all access will be authenticated through the directory. Both Read and Write access to the jChatRoom object will be verified.
- **moderator** - Moderator contains a list of distinguished names of Shepherd accounts that can act as chat room moderators. While only one moderator can be active at a time, multiple moderators can be listed to simplify administration of rooms that use different moderators at different times.

In addition to these attributes, the jChatRoom object requires the hostDomain attribute. The hostDomain attribute should be set to the distinguished name of the internetDomain under which it should run. Also, the common name (cn) attribute is used as the name of the room as seen by chat clients.

ServiceLog

If provided, Shepherd.Chat will log all logons, logoffs, and messages to a file. Using Shepherd, this requires creating a ServiceLog object referenced in the serviceLog attribute of the Shepherd.Chat Service object. The ServiceLog object requires only a common name, objectclass, and file name. The file can be put anywhere on the system. In most cases, it can be stored in the log sub-directory of the Shepherd installation. Typically, log/jchat.log will work fine.

A sample configuration is provided in setup\jchat.ldif. This configuration sets up two domains: chathost1.yourdomain.com and chathost2.yourdomain.com both directed to 127.0.0.1. (Note: For this to work, you will need to make the appropriate changes to your DNS records.) It then goes on to create two rooms under chathost1.yourdomain.com and one room under chathost2.yourdomain.com.

A private room, Sample Room 1, is created along with two users, user1 and user2. Both users are given read/write access to Sample Room 1 through the acl attribute. Be careful to identify the last component of the acl, the Service, so as not to open a security hole in the system. If "*" were given as a wildcard, user1 and user2 would have the appropriate access to modify the Sample Room 1 object through Shepherd.Admin (if it is running).

In this example, user1 and user2 were given access to the chat room with individual acls. While this is fine for two users, it may be too much work for 100 users. Therefore, you can create a Group object, assign user1, user2, and others to the member attribute, and use the group component of the acl to provide access to the group. In this way, if you need to add access to a second room for all 100 users, you will make one entry in the acl as opposed to 100.