Services

There are basically 2 types of services in UWIN
1) UMS   UWIN Master service.
2) UCS   UWIN Client service.

UWIN Master service

This is a Windows NT service. At startup this service is started automatically. The internal name for this service is UWIN_ms.

UWIN Client service

This is also a Windows NT service. The UMS activates this service. It cannot be started automatically or manually.
The internal name for this service is :
UWIN_CS<domain>#<account>
e.g. UWIN_CSAdministrator
Administrator here refers to the account since it's a local host.

Functionality of UMS

On system boot, it does the following:
1) Creates the /etc/passwd and /etc/group files
2) Starts "inetd"(internet super server) and "at" services by running the /etc/rc.sh script.
3) Adds the /etc/passwd.add or /etc/group.add entries if necessary(new accounts).
4) Enumerates the domain information if necessary. This is tunable. By default it is disabled. Generally when there are lots of users in the domain this creates lots of problems like it take a long time to start ,creates huge files to maintain this info etc.
5) Creates and sets the user home directories.
6) Provides the *setuid()* support using the UCS service.
7) Provides the setuid bit file support by starting the executable under proper account. ie the file can be run only in the owners account.
8) Obtains the user/group information on demand (e.g. getpwnam) and adds it to passwd or group file. This detail is appended to the end of the file.

Functionality of UCS

Can get installed for the user account by the following ways

1) Telnet to the account
2) Through the UWIN Control panel applet client installation wizard.
3) Through the "/etc/ucs install" command.

Is started by UMS on demand. It Obtains the security token for the account installed and passes it to UMS.

Setuid Implementation

When a setuid () is executed the application that is to be executed in that context first creates an event and sends the handle of the synchronizing event along with its process id and user id to the request pipe \\.\pipe\UWIN_setuid in which UMS is always waiting. This synchronization event is unnamed because only one process can request the UMS for the service at a time. This synchronizing event waits to be signaled with timeout. It basically waits for around 3 seconds after which it times out to prevent waiting infinitely.

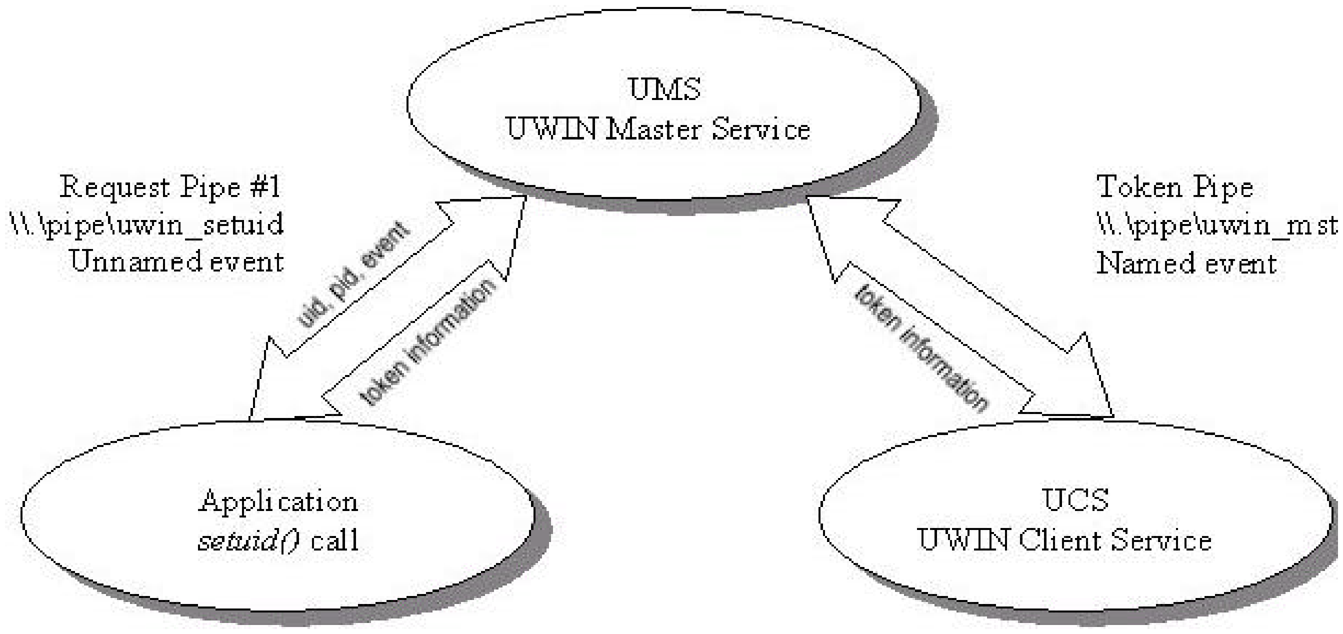
UMS that was waiting on this pipe reads the setuid data written by the application. On reading this data UMS checks if the necessary token is present in the cache.

Case 1:
If token is in the token cache it compares with the account name using the user id supplied, duplicate this token to the application process and write the token value into the request pipe. It finally signals the synchronization event (unnamed) for the application.

Case 2:
If the token is not in the cache, it creates another pipe \\.\pipe\UWIN_mstok and start the UCS service. It also sets the synchronization event (named), and waits for a connection on the pipe. The UCS then opens a security token for itself (OpenProcessToken API), duplicates the token as a primary token so that it can be passed onto to UMS and waits for the synchronization event (named) which will be signaled by UMS.
It then connects to the token pipe, writes the token information and ends its execution.
Now UMS reads the token written by UCS from the token pipe. Then it duplicates this token handle into UMS process space and store in the token cache if it is required in future. It then stop the UCS service.
It then Duplicates the token handle into the application process space and writes it into the request pipe and signals the synchronization event (unnamed) for the application.
Now the application reads the token information and impersonate onto the token using the ImpersonateLoggedOnUser() API.

Note
UMS – UCS synchronization event
Used to make sure that the UMS is doing a ConnectNamedPipe() before UCS does a CreateFile() on the token pipe.

UMS – Application synchronization event
Used to provide a timeout for the application so that if UMS does not respond within the time frame (3 seconds), the *setuid()* call will fail.


UMS
UWIN Master Service
Request Pipe #1
\\.\pipe\uwin_setuid
Unnamed event
uid, pid, event
token information
Token Pipe
\\.\pipe\uwin_mst
Named event
token information
Application
setuid() call
UCS
UWIN Client Service