# SIMPOL Server

## About the SIMPOL PPCS Server Programs

In this chapter we will discuss the multi-user database server programs currently shipped with SIMPOL, `simpolserver.exe/simpolserver.smp` and `guisimpolserver.exe`. The first is designed to run without a user-interface, from a command line prompt or as a service (more later). The second is designed to run on a logged in server as a window program with buttons for sharing the tables, stopping sharing, reorganizing the tables, backing them up and restoring from the back up. Both take a single parameter on the command line to tell them which configuration file to load. Both do the job fine. Using these or variations of these we have been deploying systems for years on both Windows and Linux. In the next sections we will go into each of them in more detail. Before we do that though, since both make use of the same configuration file format, let's examine that.

## The Configuration File

The configuration file used by both servers is based on the ini file format. There are two sections in this file. Below is an example of the file format. The file format has changed since earlier versions, but the old format is still supported. Just don't include a `cfgversion` parameter and don't make use of any of the new parameters.

### Example 8.1. A Sample SIMPOL Server Configuration File

```
[Server]
cfgversion=2
port1=4000
txfactor1=0
#port2=4001
#txfactor2=6
tcpport=4002
logfilename=samplelog.txt
bz2libdll=c:\simpol\bin\bzip2.dll
archiveroot=SIMPOL Data Backup
backupdir=C:\Users\Public\Documents\SIMPOL\backup
title=SIMPOL Server
deflocktimeout=10000000
```

```
[Files]
1=c:\simpol\utilities\simpolserver\adrb.sbm,locktimeout=120000000
```

# The [Server] Section of the Config File

In the Server section, the port entry can be repeated multiple times, but each must end in a different sequential value starting with 1. The server will listen on each of the ports listed. The `txfactor` value is used to reduce the transmission speed of the server. This is important especially when working with Superbase clients, where the client cannot process the data fast enough to keep up, resulting in lost packets. Also, when debugging it may be necessary to slow down for SIMPOL to 6 or even 9 or 10. The `tcpport` is used so that the `serverclose.smp` (or `.exe`) can request the `simpolserver` to shutdown.
As can be seen above, the entries for `port2` and `txfactor2` are commented out. You don't need port2 and txfactor2 (but you always need both together), unless you are coping with systems with differing network access speeds, like a LAN and a WAN. You might set the `LAN txfactor` to 0 and the `WAN txfactor` to 6, for example. Tables with larger numbers of fields can take longer per record, so you may need to increase the `txfactor` to support the time it takes for the record or file definition to be processed.

The `tcpport` is used to host a TCP/IP server that can be contacted using the `serverclose.exe` (or `.smp`) file. As a seurity precaution, it must be run on the same physical machine or it will be ignored. The command it sends is simply a QUIT message, but this ensures that the server will correctly shut down and flush all changes to disk.
If the `logfilename` parameter is assigned, then a log will be produced and written to a file of the same name.
When using the `guisimpolserver.exe` or the `simpolserver.exe` with the `simpolserverclient.exe`, the following three entries are required for doing back up and re-store of data:

- `bz2libdll`
- `archiveroot`
- `backupdir`

The first provides the name and location of the BZip2 DLL that is used for compressing and decompressing the data files. The `archiveroot` defines the root file name to which the date and time will be appended. The last item identifies the location where the back up files will be stored. The `deflocktimeout` entry allows the lock time out value to be set to a standard value, which will be inherited by all of the entries in the [Files] section, unless expressly overridden.

# The [Files] Section of the Config File

The format of the Files entries is as follows: `<index value>=<path and filename>, locktimeout=.inf, hidden=f, reccount=f, codepage=850, r=, rc=, rl=, rlc=, rlm=, rld=, rlcd=, rlcm=, rlmd=, rlcmd=.`

Here they are listed out:

- `<index value>`
- `<path and filename>`
- `locktimeout`
- `hidden`
- `reccount`
- `codepage`
- `r`
- `rc`
- `rl`
- `rlc`
- `rlm`
- `rld`
- `rlcd`
- `rlcm`

The index value must be in sequential order starting with 1 and not contain duplicates. As soon as a value is missed the program stops reading files.

The path and file name for each `*.sbm` file adds that container file and all of its tables (except for the system tables) to the server.
The `locktimeout` value defaults to `.inf` (never unlocks from the server side), unless it is over-ridden with a value assigned to the `deflocktimeout` entry in the [Server] section. This value should be set to be appropriate for the individual table and its use. The value is in microseconds, so to automatically unlock in 12 seconds, use: 12000000. It applies to all tables located in the same container file.

The default hidden value is `f` (false). To enable it, set it to `t` (true). Tables can be hidden so that someone connecting to the PPCS server cannot list them. If they know the table name, then they can still open the it, unless it is password protected.
The reccount default value is t. To suppress the determination of the record count for a table, set it to `f` (false). This will assign the maximum record count to the table and not try to calculate it. On very large tables calculating the record count can take some time, making server starts slow.

PPCS operates in the DOS code page normally (for historical reasons having to do with compatibility with the PPCS protocol of its predecessor (Superbase). The default code page for PPCS is 850 (Latin 1). This can be changed to another code page by setting this parameter. This is the list of supported code pages: 437, 720, 737, 775, 850, 852, 855, 857, 862, 866, 874, 1258.
The r=,rc=,rl=,rlc=,rlm=,rld=,rlcd=,rlcm=,rlmd=,rlcmd= parameters are all passwords. To password protect the database table purely for access, it is only necessary to apply a read password. More fine-grained control can be had using the other password combinations, if desired. The letters stand for the following capabilities with respect to records:

- `r` – read
- `c` – create
- `l` – lock
- `m` – modify
- `d` – delete

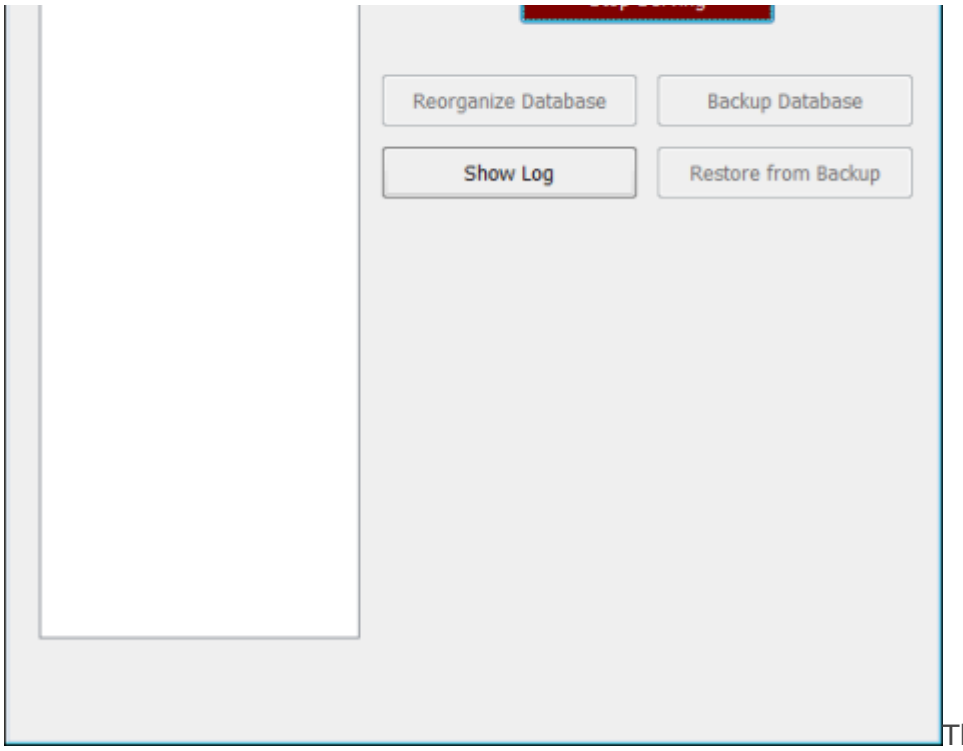
Please note that if you decide to use multiple passwords to access these tables from different users with different access rights, then the program code needs to be able to cope with any errors that may occur.

# Working with simpolserver.exe

The `simpolserver.exe` program is designed to run without any user interface. It takes a single argument on the command line, which is the name of the configuration file from which it should read the information it needs to run: which port number(s) to listen on, the name of the log file to create, the name(s) of the database files that should be shared and the parameters for each of them. If it is started without any parameters it will search for the file `simpolserverinfo.cfg` in the same directory from where the exe file was started. If it hits an error starting up it will output that information as a return value to the console and if it got far enough there may be some information in the log file.

It can also be run using the `simpolserver.smp` file, in which case it should definitely be run by passing the configuration file name to it as a parameter. This is the program to use when running on Linux.
When running this program, there is a companion program that can provide the user-interface for the server. That program is delivered as `simpolserverclient.exe`. It is currently only available for Windows. It will present a user-interface that looks more or less identical to that of the `guisimpolserver.exe` program, but it acts only as a front-end to the server version. It can also start and stop the sharing of tables, as well as perform a reorganize, back up, restore and can view the log. The `simpolserverclient.exe` takes a single parameter, which is the TCP/IP port where the server is listening. If no parameter is provided, it will search the directory where the exe file was located when it started for a file called `simpolserverinfo.cfg` and if it finds it will read the TCP/IP port value from there.

The `guisimpolserver.exe` program was designed as a desktop program to be run on a logged-in server. It gives much more fine-grained control than our original `simpolserver.smp` program. With the release of the new version of the `simpolserver.exe` and its ability to be controlled using `simpolserverclient.exe`, there are fewer reasons to choose this version, though it is a much simpler design and may have less issues in a working environment. For one thing, it is all self-contained. On the other hand, if left running for a very long time, it might be less stable than a version that does not have the user-interface components included and running. For now, pick the one that suits you best. Since they both support the same configuration file format, there is no difficulty moving from one version to another. Below is an image showing the user-interface. This image is from the simpolserverclient.exe program, but there is no obvious difference between them.



The user interface for guisimpolserver and simpolserverclient

# Running simpolserver.exe as a Service

In a production environment, you would ideally want the database server to start when the server starts, and to shut down gracefully when the server shuts down. Using Linux, this is quite easy depending on your distribution. It is fairly trivial to add `/usr/bin/smprun /home/ me/simpolserver.smp /home/me/simpolserverinfo.cfg` to the start up of the server and `/usr/bin/smprun /home/me/serverclose.smp 12345` to the shut down of the server (typically something like `local.start` and `local.stop` in `/etc/conf.d/ or baselayout1.start` and `baselayout1.stop in /etc/local.d/`). Linux makes a lot of things easier, which is why we only have one loader program for both console and windowing programs. On Windows, this is all much more complicated.

We created a special program that can run a specified program as a service, with a second program that can be run to stop the service. The program is called `svcrunnr.exe`. To install the service running program, from the command line run this: `svcrunnr.exe -install`. To remove the service, use this command line: `svcrunnr.exe -remove`. Before you start the service, it is useful to check that everything is correctly configured.
Once the service is installed, it is important to check its configuration file. The name of that file is: `svcrunnr.exe.ini`. This is a very simple file and it needs to be located in the same directory as the `svcrunnr.exe program`. It contains two sections, each with only one entry:

## Example 8.2. A Sample svcrunnr.exe Configuration File

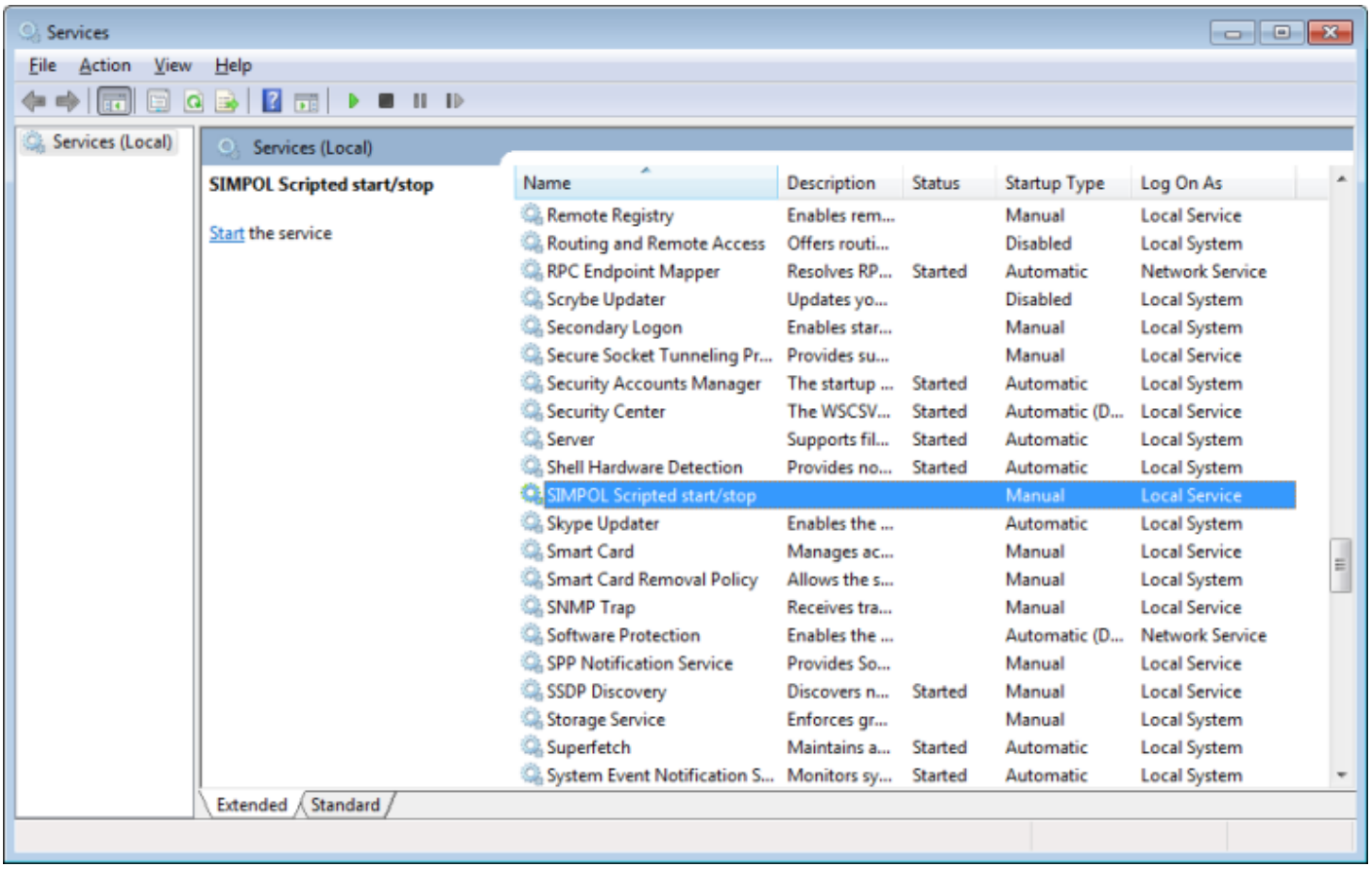
```
[Startup]
Command=C:\SIMPOL\bin\simpolserver.exe

[Shutdown]
Command=C:\SIMPOL\bin\serverclose.exe
```

Normally, there should be no reason to change this configuration file unless the locations are incorrect. It is specific to the installation of the `simpolserver.exe` suite of programs.

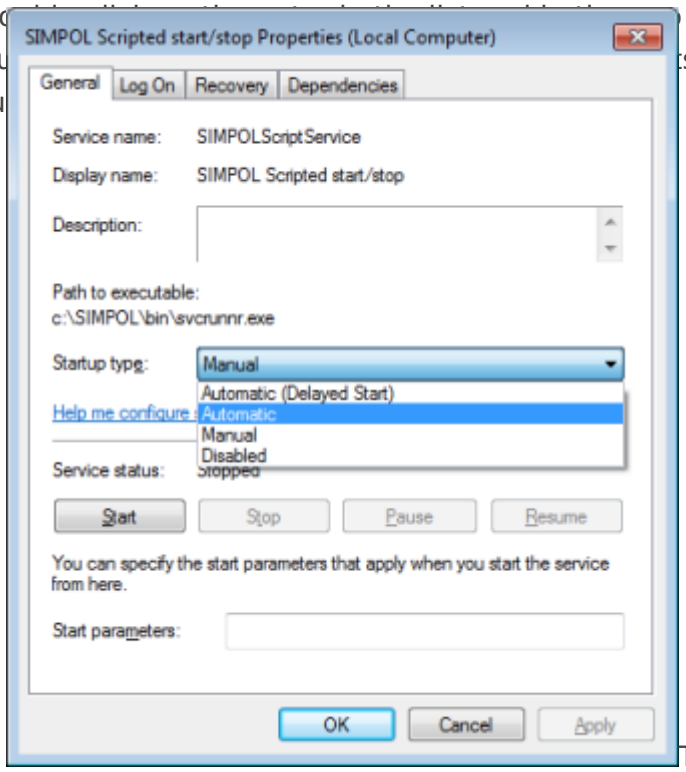
> Note
> Only one instance of this service can be running and/or installed concurrently, so there is no easy use that can be made of the service runner for other purposes. At some point we will release a more powerful service loader for SIMPOL.

After checking the configuration file and after making sure that the `simpolserverinfo.cfg` is correctly set up (because this method will always use this file name), if the server should run automatically, then it needs to be set to do so. Upon installation it is set up to run manually only. To change this, in a console window, or in the Start menu enter `services.msc` in the search window and press **Enter**. The services program will start. Navigate in the list to the `SIMPOL Scripted start/ stop entry` as shown below:




The Services window showing the svcrunnr entry

Do up window change the startup type to Au ts, then click on the Start button to start it ru



The SIMPOL Scripted start/stop window showing the set up

That's it! The database engine will now run automatically when the server starts, and will automatically shutdown when the server is shutdown. To see what is happening and to control it, the `simpolserverclient.exe` program can be run.

# SIMPOL Server Summary

In this chapter we have learned about the multi-user database server programs supplied with the SIMPOL product. We have learned how to configure them, start them, stop them, and use them for doing data back up and restore, as well as table maintenance using the reorganize command. Finally, we have learned how to use the `svcrunnr.exe` program to run the `simpolserver.exe` as a service so that it will always be available, even when the computer has just started and has not been logged in.

---

Revision #7
Created 15 November 2021 19:07:03 by Eva Crawford-McKee
Updated 27 December 2021 21:39:31 by Nikolaus Zolnhofer