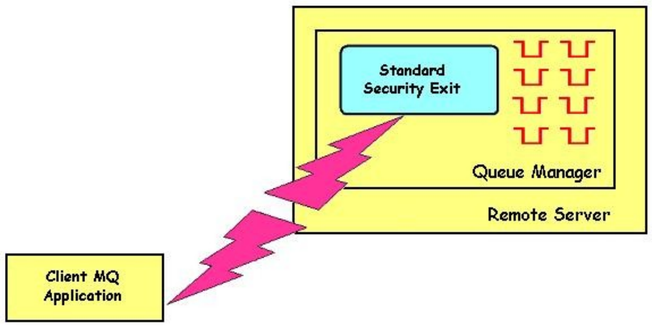
MQSSX Installation and Operation Manual



Capitalware Inc.
Unit 11, 1673 Richmond Street, PMB524
London, Ontario N6G2N3
Canada
sales@capitalware.com
https://www.capitalware.com

Table of Contents

1 Introduction

1.1 Overview

MQ Standard Security Exit (MQSSX) is a new solution that allows a company to control and restrict who is accessing a IBM MQ resource. The security exit will operate with IBM MQ v7.1, v7.5, v8.0, v9.0, v9.1 and v9.2 in Windows, Unix, IBM i and Linux environments. It works with Server Connection, Receiver, Requester and Cluster-Receiver channels of IBM MQ queue manager.

The MQ Standard Security Exit solution is comprised of a server-side security exit.

The server-side security exit has the ability to allow or restrict the incoming UserID. The server-side security exit uses a regular expression parser to parse the incoming client UserID against a predefined regular expression pattern.

The server-side security exit supports the concept of 'Proxy IDs'. After a user has been successfully validated against the native OS or file based validation data and the 'Proxy Mode' flag is set, then the security exit will look up the user's UserID in the Proxy file for their Proxy ID. The Proxy ID will be used for all MQ interactions.

The server-side security exit has the ability to allow or restrict users from connecting with a blank UserID value. This is controlled by the server-side security exit's property keyword 'AllowBlankUserID'.

The server-side security exit has the ability to block users from logging in with the 'mqm' or 'MUSR_MQADMIN' or 'QMQM' UserIDs. This is controlled by the server-side security exit's property keyword 'Allowmqm'.

The server-side security exit has the capability to allow or limit the incoming channel connections according to the name of the associated Server Connection channel (SVRCONN). Each Server Connection channel can be allocated a maximum number of connections and the server-side security exit will ensure that this maximum is not exceeded.

Client connections to a queue manager are limited by either channel name or the 'DefaultMCC' property keyword in the initialization file. In today's use of J2EE applications, it is a possibility that one J2EE application could overwhelm the queue manager with client connections, thus preventing any connections being made from other applications.

The MQAdmin can enable Excessive Client Connections alerting system that counts the number of connections over a period of time (i.e. Day / Hour / Minute) and writes a message to the log when the count exceeds a particular value. If the keyword WriteToEventQueue is set to ‘Y’ then an event message is also written to an event queue. ECC feature is designed to catch applications that are poorly written, for example, applications that continuously connect and disconnect from the queue manager for every message sent or received.

The server-side security exit has the ability to allow or restrict the incoming IP address, hostname and/or SSL DN. The server-side security exit uses a regular expression parser to parse the incoming client IP address and/or SSL DN against a predefined regular expression pattern.

The server-side security exit has the ability to allow or restrict the incoming UserID against a group. A list of groups can be queried for the incoming UserID. The groups can be in the local OS or a group file.

On AIX, HP-UX, Linux, Solaris and Windows, MQSSX can be configured and used with a non-default installation of MQ in a multi-install MQ environment.

Note: Raspberry Pi is a Linux ARM 32-bit OS (Operating System). Hence, simply follow the Linux 32-bit instructions for installing and using the solution on a Raspberry Pi.

1.2 Executive Summary

The MQ Standard Security Exit solution contains a server-side security exit.

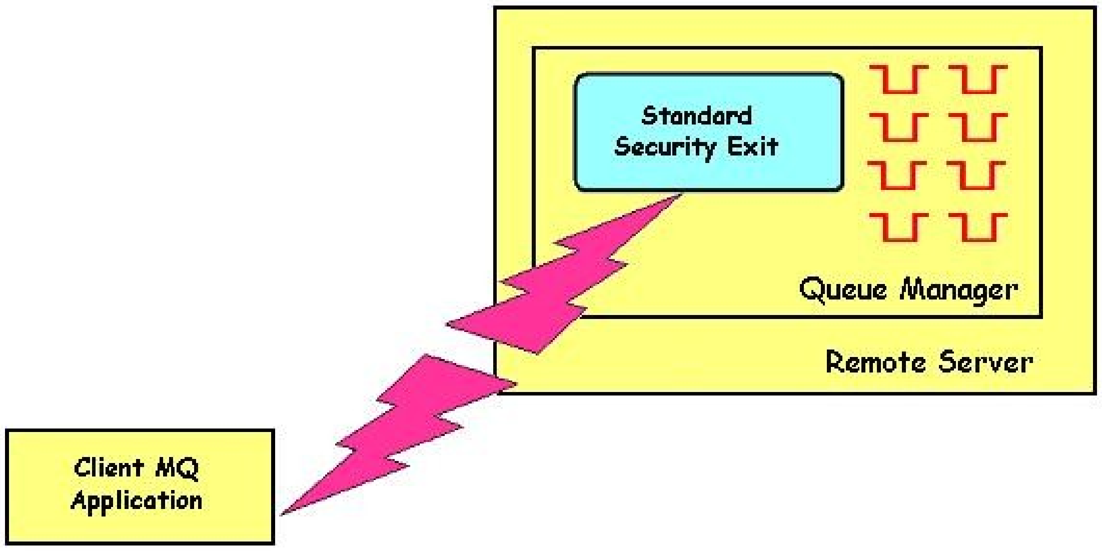
The server-side security exit is available in 3 forms:

- Windows DLL
- Shared library for AIX, HP-UX, Linux, and Solaris.
- IBM i exit module

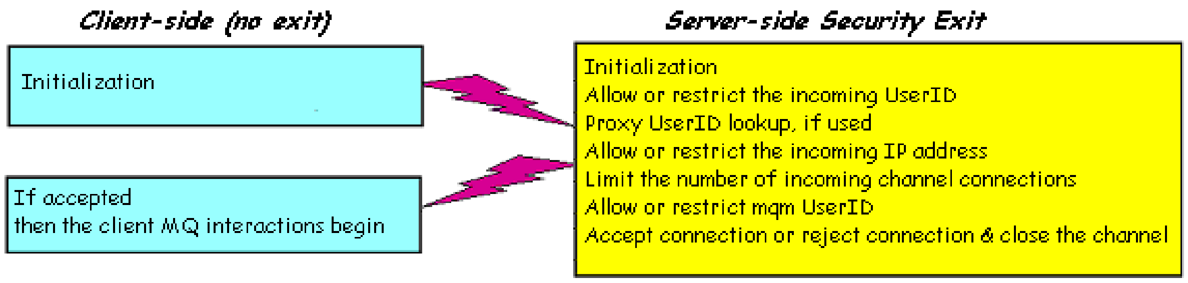
The major features of MQ Standard Security Exit are as follows:

- Allows or restricts the incoming UserID against a regular expression pattern
- Allows or restricts the incoming UserID against a Group
- Provides support for Proxy UserIDs
- Allows or restricts the incoming IP address against a regular expression pattern
- Allows or restricts the incoming hostname against a regular expression pattern
- Allows or restricts the incoming SSL DN against a regular expression pattern
- Limit the number of incoming channel connections on a SVRCONN channel.
- Allows or restricts the use of 'mqm', 'MUSER_MQADMIN' or 'QMQM' UserIDs
- Ability to set the maximum number of allowable connections per a given channel (MCC)
- Ability to monitor for excessive client connections (ECC) and then generate an alert
- Provides monitoring tool tie-in by using custom MQ event messages
- Provides logging capability for all connecting client applications regardless if they were successful or not.

1.3 Context Diagram (Logical View)



1.4 Security Message Flow (Logical View)


Client-side (no exit)
Server-side Security Exit
Initialization
If accepted
then the client MQ interactions begin
Initialization
Allow or restrict the incoming UserID
Proxy UserID lookup, if used
Allow or restrict the incoming IP address
Limit the number of incoming channel connections
Allow or restrict mqm UserID
Accept connection or reject connection & close the channel

1.5 Prerequisites

This section provides the minimum supported software levels. These prerequisites apply to server-side installations of MQ Standard Security Exit.

1.5.1 Operating System

MQ Standard Security Exit can be installed on any of the following supported servers:

1.5.1.1 IBM AIX

- IBM AIX 6L version 6.1 or higher

1.5.1.2 HP-UX IA64

- HP-UX v11.23 or higher

1.5.1.3 IBM i (OS/400)

- IBM i V6R1 or higher

1.5.1.4 Linux x86

- Red Hat Enterprise Linux v5, v6, v7, v8
- SUSE Linux Enterprise Server v11, v12, v15

1.5.1.5 Linux x86_64 (64-bit)

- Red Hat Enterprise Linux v5, v6, v7, v8
- SUSE Linux Enterprise Server v11, v12, v15

1.5.1.6 Linux on POWER

- Red Hat Enterprise Linux v5, v6, v7, v8
- SUSE Linux Enterprise Server v11, v12, v15

1.5.1.7 Linux on zSeries (64-bit)

- Red Hat Enterprise Linux v5, v6, v7, v8
- SUSE Linux Enterprise Server v11, v12, v15

1.5.1.8 Raspberry Pi (Linux ARM 32-bit)

- Raspberry Pi OS v9 or higher

1.5.1.9 Sun Solaris

- Solaris SPARC v10 or higher
- Solaris x86_64 v10 or higher

1.5.1.10 Windows

- Windows 2008, 2012 or 2016 Server (32-bit & 64-bit)
- Windows 7, 8, 8.1 & 10 (32-bit & 64-bit)

1.5.2 IBM MQ

- IBM MQ v7.1, v7.5, v8.0, v9.0, v9.1 and v9.2 (both 32-bit and 64-bit)

Operating System	MQ v7.1, v7.5, v8.0, v9.0, v9.1 and v9.2
AIX v6.1 or higher	64-bit
HP-UX IA64 v11.23 or higher	64-bit
IBM i (OS/400)	64-bit
Linux x86	32-bit
Linux x86_64	64-bit
Linux on POWER	64-bit
Linux on zSeries	64-bit
Raspberry Pi ARM	32-bit
Solaris SPARC v10 & v11	64-bit
Solaris x86_64 v10 & v11	64-bit
Windows 2008, 2012, 2016, 7, 8, 8.1 & 10	32-bit & 64-bit

1.5.3 Windows 64-bit

The following is the software prerequisite for Windows 64-bit:

- Microsoft Visual C++ 2010 Redistributable Package (x64)
  https://download.microsoft.com/download/1/6/5/165255E7-1014-4D0A-B094-B6A430A6BFFC/vcredist_x64.exe

2 Installing MQ Standard Security Exit

This section describes how to install Capitalware's MQ Standard Security Exit.

2.1 Server-side Security Exit

2.1.1 Windows 32-bit Installation

To install the security exit on Windows, first unzip the **mqssx.zip** and then run the **mqssx-setup-32bit.exe** file. Follow the on-screen instructions and the security exit will be installed in the **C:\Capitalware\MQSSX\** directory (default installation).

The user may copy or ftp the mqssx.dll, mqssx.ini, AddRegistryEntries.bat and mqssx.reg files from one Windows server to another Windows server.

2.1.2 Windows 64-bit Installation (MQ v8.0 or higher)

To install the security exit on Windows, first unzip the **mqssx.zip** and then run the **mqssx-setup-64bit.exe** file. Follow the on-screen instructions and the security exit will be installed in the **C:\Capitalware\MQSSX\** directory (default installation).

The user may copy or ftp the mqssx.dll, mqssx.ini, AddRegistryEntries.bat and mqssx.reg files from one Windows server to another Windows server.

2.1.3 Linux 32-bit Installation

To install the 32-bit version of MQSSX on Linux, first unzip the **mqssx.zip** and then select the appropriate TAR file for the target platform. You will find 2 TAR files in the original ZIP file:

- **Linux_x86/mqssx_linux.tar**
- **RaspberryPi_ARM/mqssx_raspberrypi_arm.tar**

Steps to install the server-side security exit:

1. ftp or copy the selected TAR file to the target platform to the ***/var/mqm/exits/*** directory.
2. Un-tar the mqssx_xxx.tar file into the ***/var/mqm/exits/*** sub-directory (xxx is either aix, hpux, solaris or linux)

   ```
   cd /var/mqm/exits/
   tar -xvf  mqssx_xxx.tar
   ```

3. Change directory to ***/var/mqm/exits/***
4. Next, do the following commands against ***mqssx***:

   ```
   chown mqm:mqm mqssx
   chmod 550 mqssx
   ```

2.1.4 Unix and Linux 64-bit Installation

To install the 64-bit version of MQSSX on Unix or Linux, first unzip the **mqssx.zip** and then select the appropriate TAR file for the target platform. You will find 10 TAR files in the original ZIP file:

- **AIX/64-bit/mqssx_aix53_64.tar**
- **AIX/64-bit/mqssx_aix61_64.tar** for AIX v6.1 or higher
- **HPUX_IA64 /mqssx_hpux64_ia64.tar**
- **Linux_x86_64/mqssx_linux_x86_64.tar**
- **Linux_POWER/mqssx_linux_power64.tar**
- **Linux_zSeries/64-bit/mqssx_linux_zseries64.tar**
- **Solaris_SPARC/64-bit/mqssx_solaris64.tar** for Solaris SPARC v8 and v9
- **Solaris_SPARC/64-bit/mqssx_solaris10_64.tar** for Solaris SPARC v10 or higher
- **Solaris_x86_64/mqssx_solaris_x86_64.tar**

Steps to install the server-side security exit:

1. ftp or copy the selected TAR file to the target platform to the ***/var/mqm/exits64/*** directory.
2. Un-tar the mqssx_xxx.tar file into the ***/var/mqm/exits64/*** sub-directory (xxx is either aix, hpux, solaris or linux)

   ```
   cd /var/mqm/exits64/
   tar -xvf  mqssx_xxx64.tar
   ```

3. Change directory to ***/var/mqm/exits64/***
4. Next, do the following commands against ***mqssx***:

   ```
   chown mqm:mqm mqssx
   chmod 550 mqssx
   ```

2.1.5 IBM i Installation

To install the MQSSX on IBM i, first unzip the **mqssx.zip** and then select the files in the IBM i (IBM i) directory.

- **mqssx.savf** is the IBM i 'Save File' that contains the library with the security exit.
- **mqssx_iseries.tar** is the IBM i IFS TAR file that contains a sample initialization file for the server-side security exit and sample MQSC script to define MQ channels with the security exits.

Steps to install the server-side security exit:

1. Log onto the target IBM i server and do the following command:

   ```
   CRTSAVF FILE(QGPL/MQSSX)
   ```

2. ftp the IBM i files to the IBM i server as follows:

   ```
   ftp -s:mqssx_iseries.ftp  iseries_hostname
   ```

   ```
   your-IBM i-userid
   your-IBM i-password

   binary
   cd QGPL
   put mqssx.savf

   quote SITE NAMEFMT 1

   cd /QIBM/UserData/mqm/
   put mqssx_iseries.tar
   quit
   ```

3. Log onto the target IBM i server and do the following commands:

   ```
   RSTLIB SAVLIB(MQSSX) DEV(*SAVF) SAVF(QGPL/MQSSX)
   CLRSAVF FILE(QGPL/MQSSX)
   CHGOBJOWN  OBJ(MQSSX) OBJTYPE(*LIB)  NEWOWN(QMQM)
   qsh
   cd /QIBM/UserData/mqm/
   tar -xvf mqssx_iseries.tar
   chown -R QMQM mqssx
   rm mqssx_iseries.tar
   ```

2.1.6 MQSSX-GUI Installation

This section will describe how to install the MQSSX-GUI. The user will find 2 files in the software package listed as follows:

- **MQSSX-GUI/mqssxgui-wthjre.exe** (for Windows)
- **MQSSX-GUI/mqssxgui.zip** (for Unix, Linux or macOS)

2.1.6.1 MQSSX-GUI Installation on Windows

To install MQSSX-GUI on Windows, run the **mqssxgui-withjre.exe** file located in the MQSSX-GUI directory. Follow the on-screen instructions and the program will be installed in the **C:\Capitalware\MQSSX-GUI\** directory (default installation).

2.1.6.2 MQSSX-GUI Installation on Unix, Linux or macOS

To install MQSSX-GUI on Unix or Linux, you will need to ftp or copy the selected TAR file to the target platform to the /home/mqm/ directory. Next, one must telnet to the Unix, Linux or macOS server and 'cd' (change directory) to the /home/mqm/ directory and unzip the archive file.

i.e. Do the following command:

```
unzip mqssxgui.zip
```

3 Configuring Server-side Security Exit

This section describes how to configure the server-side security exit.

3.1 MQSSX Filtering

Starting with IBM v7.1, IBM has included a feature called channel authentication record. Channel authentication record feature allows for the filtering of incoming client connections. Exiting queue managers that are migrated to MQ v7.1 or higher will have this feature disabled but when the MQAdmin creates a new queue manager, this feature will be enabled. Hence, to use MQSSX's filtering feature, issue the following MQSC command to disable channel authentication record mechanism:

```
ALTER QMGR CHLAUTH(DISABLED)
```

3.2 Security User Data (SCYDATA)

Security User Data (SCYDATA) field must NOT exceed 32 characters. In order to work with this limitation, MQSSX supports 3 ways to specify an IniFile path: absolute path, relative path and environment variable.

Note: The IniFile path that is determined by MQSSX server-side security exit will also be used for the following IniFile keywords (if no pathing is specified for these keywords): LicenseFile, LogFile and ProxyFile.

3.2.1 Absolute Path

Absolute pathing (specifying the complete path) for the SCYDATA works on Linux, Unix and Windows platforms.

E.g. Windows

```
DEFINE CHANNEL ('SYSTEM.ADMIN.SVRCONN') CHLTYPE(SVRCONN) +
       TRPTYPE(TCP) +
       SCYEXIT('C:\Capitalware\MQSSX\mqssx(SecExit)') +
       SCYDATA('C:\Capitalware\MQSSX\mqssx.ini') +
       REPLACE
```

Hence, MQSSX will use the following path as the IniFile path:
`C:\Capitalware\MQSSX\`

3.2.2 Relative Path

Relative pathing for the SCYDATA is supported on Linux, IBM i, Unix and Windows platforms. MQSSX will extract the path from SCYEXIT field and prefix it to the IniFile specified in the SCYDATA field in order to locate the IniFile.

E.g. Unix

```
DEFINE CHANNEL ('SYSTEM.ADMIN.SVRCONN') CHLTYPE(SVRCONN) +
       TRPTYPE(TCP) +
       SCYEXIT('/var/mqm/exits/mqssx(SecExit)') +
       SCYDATA('mqssx.ini') +
       REPLACE
```

Hence, MQSSX will use the following path as the IniFile path:
`/var/mqm/exits/`

3.2.3 Environment Variables

3.2.3.1 Global Environment Variable

MQSSX supports the use of the MQSSX_HOME environment variable which holds the directory path information. MQSSX_HOME environment variable is supported on Linux, IBM i, Unix and Windows platforms.

e.g. Unix

```
export MQSSX_HOME=/really/long/path/MQHA/QMgrName/data/
```

```
DEFINE CHANNEL ('SYSTEM.ADMIN.SVRCONN') CHLTYPE(SVRCONN) +
       TRPTYPE(TCP) +
       SCYEXIT('/var/mqm/exits64/mqssx(SecExit)') +
       SCYDATA('mqssx.ini') +
       REPLACE
```

Hence, MQSSX will use the following path as the IniFile path:
`/really/long/path/MQHA/QMgrName/data/`

3.2.3.2 Queue Manager Specific Environment Variable

MQSSX supports the use of the MQSSX_HOME environment variable post-fixed with the queue manager name which holds the directory path information. MQSSX_HOME environment variable post-fixed with the queue manager name is supported on Linux, IBM i, Unix and Windows platforms.

e.g. Unix with a queue manager name of MQA1

```
export MQSSX_HOME_MQA1=/really/long/path/MQHA/QMgrName/data2/
```

```
DEFINE CHANNEL ('SYSTEM.ADMIN.SVRCONN') CHLTYPE(SVRCONN) +
       TRPTYPE(TCP) +
       SCYEXIT('/var/mqm/exits64/mqssx(SecExit)') +
       SCYDATA('mqssx.ini') +
       REPLACE
```

Hence, MQSSX will use the following path as the IniFile path:
`/really/long/path/MQHA/QMgrName/data2/`

Note: If both environment variables are specified then the queue manager specific environment variable will be used.

3.3 SVRCONN Channel

This section describes the necessary entries to enable the server-side security exit. The MQ Administrator will need to update 2 fields of the SVRCONN Channel that the server-side security exit will be applied to.

Platform		Directory	Exit Module Name
Windows	32-bit	C:\Capitalware\MQSSX\	mqssx.dll
Linux/Unix	32-bit	/var/mqm/exits/	mqssx
Linux/Unix	64-bit	/var/mqm/exits64/	mqssx

MQSSX supports MQ's multi-install in a non-default directory.

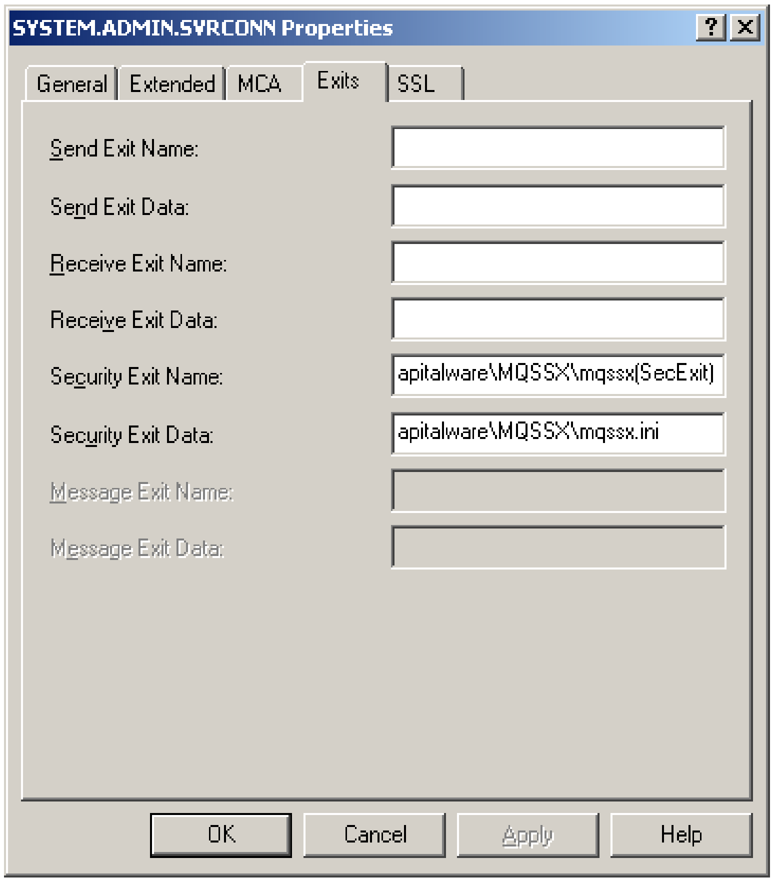
Note: The Security Exit Data (SCYDATA) field must NOT exceed 32 characters.

3.3.1 Windows

On Windows, SCYEXIT and SCYDATA will contain the following values assuming a default install.

- SCYEXIT
  `C:\Capitalware\MQSSX\mqssx(SecExit)`
- SCYDATA
  `C:\Capitalware\MQSSX\mqssx.ini`

```
DEFINE CHANNEL ('SYSTEM.ADMIN.SVRCONN') CHLTYPE(SVRCONN) +
       TRPTYPE(TCP) +
       SCYEXIT('C:\Capitalware\MQSSX\mqssx(SecExit)') +
       SCYDATA('C:\Capitalware\MQSSX\mqssx.ini') +
       REPLACE
```



Send Exit Name:	
Send Exit Data:	
Receive Exit Name:	
Receive Exit Data:	
Security Exit Name:	apitalware\MQSSX\mqssx(SecExit)
Security Exit Data:	apitalware\MQSSX\mqssx.ini
Message Exit Name:	
Message Exit Data:	

3.3.2 Linux 32-bit

On Linux, SCYEXIT and SCYDATA will contain the following values assuming a default install:

- SCYEXIT
  `/var/mqm/exits/mqssx(SecExit)`
- SCYDATA
  `/var/mqm/exits/mqssx.ini`

```
DEFINE CHANNEL ('SYSTEM.ADMIN.SVRCONN') CHLTYPE(SVRCONN) +
       TRPTYPE(TCP) +
       SCYEXIT('/var/mqm/exits/mqssx(SecExit)') +
       SCYDATA('/var/mqm/exits/mqssx.ini') +
       REPLACE
```

3.3.3 Unix and Linux 64-bit

On Unix and Linux (excluding Linux x86), SCYEXIT and SCYDATA will contain the following values assuming a default install:

- SCYEXIT
  `/var/mqm/exits64/mqssx(SecExit)`
- SCYDATA
  `/var/mqm/exits64/mqssx.ini`

```
DEFINE CHANNEL ('SYSTEM.ADMIN.SVRCONN') CHLTYPE(SVRCONN) +
       TRPTYPE(TCP) +
       SCYEXIT('/var/mqm/exits64/mqssx(SecExit)') +
       SCYDATA('/var/mqm/exits64/mqssx.ini') +
       REPLACE
```

3.3.4 IBM i

On IBM i, SCYEXIT and SCYDATA will contain the following values assuming a default install:

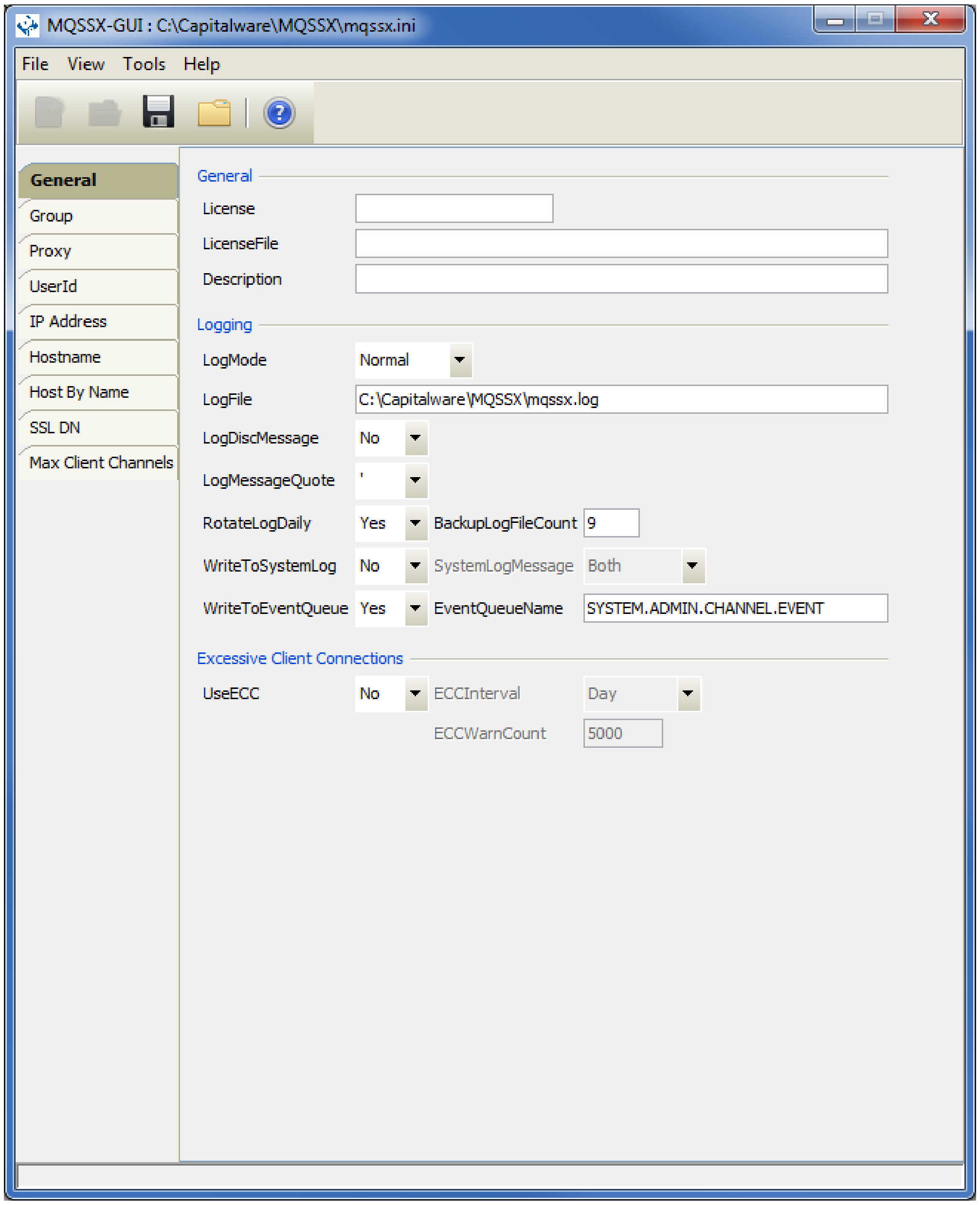
- SCYEXIT is made up of 10 characters for program name (padded with blanks) followed by 10 characters for the LIBRARY name (padded with blanks).
  `MQSSX     MQSSX`

- SCYDATA
  `mqssx.ini`

```
DEFINE CHANNEL ('SYSTEM.ADMIN.SVRCONN') CHLTYPE(SVRCONN) +
       TRPTYPE(TCP) +
       SCYEXIT('MQSSX     MQSSX     ') +
       SCYDATA('mqssx.ini') +
       REPLACE
```

3.4 MQSSX-GUI

This section briefly describes the new graphical program called MQSSX-GUI. MQSSX-GUI assists the user in creating and managing their MQSSX IniFiles. For more information, please see the ***MQSSX-GUI User Guide*** manual.



General

License

LicenseFile

Description

Logging

LogMode: Normal

LogFile: C:\Capitalware\MQSSX\mqssx.log

LogDiscMessage: No

LogMessageQuote: '

RotateLogDaily: Yes BackupLogFileCount: 9

WriteToSystemLog: No SystemLogMessage: Both

WriteToEventQueue: Yes EventQueueName: SYSTEM.ADMIN.CHANNEL.EVENT

Excessive Client Connections

UseECC: No ECCInterval: Day

ECCWarnCount: 5000

4 IniFile Keywords (Server-side)

This section describes IniFile keywords.

4.1 Logging

This section describes the necessary entries to enable MQSSX to record log information. To enable and control logging, there are 10 keywords in the IniFile:

1. **LogMode** specifies what type of logging the user wishes to have. LogMode supports 4 values [Q / N / V / D] where Q is Quiet, N is Normal, V is Verbose and D is Debug. The default value is N.

2. **LogFile** specifies the location of the log file. The default is as follows:

   For Windows:
   ```
   LogFile=C:\Capitalware\MQSSX\mqssx.log
   ```

   For IBM MQ 32-bit on Linux:
   ```
   LogFile=/var/mqm/exits/mqssx.log
   ```

   For IBM MQ 64-bit on Unix and Linux:
   ```
   LogFile=/var/mqm/exits64/mqssx.log
   ```

   For IBM MQ on IBM i:
   ```
   LogFile=/QIBM/UserData/mqm/mqssx/mqssx.log
   ```

   Token Replacement for LogFile keyword:
   - **%QM%** - Substitutes the name of the queue manager
   - **%CHL%** - Substitutes the name of the channel
   - **%UID%** - Substitutes the UserID
   - **%PID%** - Substitutes the Process ID
   - **%TID%** - Substitutes the Thread ID

3. **LogDiscMessage** specifies whether or not MQSSX write a disconnect message when the client application closes the channel. The default value is No.

4. **LogMessageQuote** specifies the type of quote (single or double) to be used on the log message. The default value is ' (single quote).

5. **RotateLogDaily** specifies whether or not the log files will be rotated on a daily basis. Setting ' RotateLogDaily' to 'Y' (Yes) will activate this feature; otherwise, the log files will left as is. The default value is Y.

   The RotateLogDaily feature will keep up to 9 backup log files. The first connection request after midnight (and not at midnight) will cause it to roll / rotate the log files. If there are already 9 backup log files then the ninth backup log file will be delete and 8 becomes 9, 7 becomes 8, etc...

6. **BackupLogFileCount** specifies the number of backup log files that should be kept by the security exit. The default value is 9. This keyword is only used if RotateLogDaily is set to 'Y'.

7. **WriteToSystemLog** specifies whether or not MQSSX will write a log entry to the server's 'logging system'. On Windows, the server's 'logging system' is the Event Log and on Unix / Linux, it is the syslog. Setting ' WriteToSystemLog' to 'Y' (Yes) will activate this feature; otherwise, there will be no logging to the server's logging system. The default value is N.

   The location of the Unix / Linux syslog output can be found for each operating system as follows:

➢ AIX:	/var/log/messages
➢ HP-UX:	/var/adm/syslog/syslog.log
➢ Linux:	/var/log/messages
➢ Solaris:	/var/adm/messages

   The Windows Event log can be viewed using the Windows Event Viewer.

   Note: If the user did not install MQSSX server-side security exit using the windows installer then the user will need to run the AddRegistryEntries.bat file to add the registry entries in order to view the MQSSX log messages in the Event Viewer.

8. **SystemLogMessage** specifies what messages will be written to the system log.. SystemLogMessage supports 3 values [B / A / R] where B is Both, A is Accepted Only, and R is Rejected Only messages. The default value is B.

9. **WriteToEventQueue** specifies whether or not MQSSX will write an event message containing the log entry information to the event queue. The default value is N.

   WriteToEventQueue provides the ability to write custom MQ Events to System Channel Event Queue to allow MQSSX to be tied into an MQ Monitoring tool.
   - 9101 for Connection rejected event message
   - 9201 for MCC Warning event message
   - 9202 for MCC Exceeded event message

10. **EventQueueName** specifies the event queue name. The default value is 'SYSTEM.ADMIN.CHANNEL.EVENT'.

```
LogMode=N
LogFile=C:\Capitalware\MQSSX\mqssx.log
RotateLogDaily=Y
WriteToSystemLog=Y
WriteToEventQueue=Y
EventQueueName=SYSTEM.ADMIN.CHANNEL.EVENT
```

4.2 Allow or Restrict the Incoming UserID against a Group

This section describes the necessary entries to enable the feature that allows or restricts the incoming UserID against an OS group or a group file. This feature uses the following four keywords:

- **UseGroups** keyword controls the use of Groups. Set to 'Y' to allow authorization by either OS or a group file.
- **Groups** keyword specifies the authorized groups that can connect to the queue manager. Each group is separated from the next by a semi-colon (';').
- **UseGroupFile** keyword controls the use of GroupFile. Set to Y to activate feature.
- **GroupFile** keyword specifies the location of the group file (i.e. groups.ini)

4.2.1 Authorization against Local OS

When UseGroups is set to 'Y' and UseGroupFile is set to 'N' then MQSSX will query the local OS for the groups listed in the Groups keyword.

Example:

```
UseGroups=Y
Groups=grpX;grpZ
UseGroupFile=N
```

4.2.2 Authorization against a Group File

When UseGroups is set to 'Y' and UseGroupFile is set to 'Y' then MQSSX will query the specified group file given in GroupFile keyword. This section describes how to implement groups files. The group files are implemented in a similar manner to the way they are implemented in Unix and Linux (i.e. **/etc/group** file).

Below is an example group file:

```
unique_group_name = UserID1;UserID2;UserID3
```

Example:

```
grp1=fred;wilma;pebbles
grp2=barney;betty;bammbamm
grpA=arnold;rockhead;slate;gazoo
grpB=dino;puss;doozy;hoppy
```

The following are the default values for GroupFile:

For Windows:
`GroupFile=C:\Capitalware\MQSSX\groups.ini`

For IBM MQ 32-bit on Unix and Linux:
`GroupFile=/var/mqm/exits/groups.ini`

For IBM MQ 64-bit on Unix and Linux:
`GroupFile=/var/mqm/exits64/groups.ini`

For IBM MQ on IBM i:
`GroupFile=/QIBM/UserData/mqm/mqme/groups.ini`

MQSSX will check, in order, each group listed in the Groups keyword for a particular UserID. The UserID must exist in one of the groups or else MQSSX will not allow the connection.

Example:

```
UseGroups=Y
Groups=grp1;grp2;grpB
UseGroupFile=Y
GroupFile=C:\Capitalware\MQSSX\groups.ini
```

4.3 Allow or Restrict the Incoming IP Address

This section describes the necessary entries to enable the feature that allows or restricts the incoming IP addresses through the use of regular expression patterns. This feature uses the following two keywords:

- **UseAllowIP** controls the use of AllowIP. Set to Y to activate feature.

- **AllowIP** specifies the regular expression patterns that limit the allowable incoming IP addresses

The server-side security exit will look up the regular expression patterns from the **AllowIP** keyword in order to determine if the entire incoming IP Address matches any of the specified expression patterns. Each regular expression pattern is separated from the next pattern by a semi-colon (';').

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any character in the specified set,
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

Note: AllowIP must NOT exceed 2048 characters.

```
UseAllowIP=Y
AllowIP=192.168.*.*;10.15[0-9].2[0-5][0-9];127.0.0.?
```

4.4 Allow or Restrict the Incoming Hostname

This section describes the necessary entries to enable the feature that allows or restricts the incoming Hostnames through the use of regular expression patterns. This feature uses the following two keywords:

- **UseAllowHostname** controls the use of AllowHostname. Set to Y to activate feature.
- **AllowHostname** specifies the regular expression patterns that limit the allowable incoming Hostnames

The server-side security exit will look up the regular expression patterns from the **AllowHostname** keyword in order to determine if the entire incoming Hostname matches any of the specified expression patterns. Each regular expression pattern is separated from the next pattern by a semi-colon (';').

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any character in the specified set,
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

Note: AllowHostname must NOT exceed 2048 characters.

Separate each Hostname pattern with a ';' semi-colon.

```
UseAllowHostname=Y
AllowHostname=abc01.acme.com;abc02.acme.com
```

4.5 Allow or Restrict the Incoming IP against IP of Hostname

This section describes the necessary entries to enable the feature that allows or restricts the incoming IP against IP of Hostnames that MQSSX will perform a gethostbyaddr() call against to compare the returned IP address against the incoming IP address. This feature uses the following two keywords:

- **UseAllowHostByName** controls the use of AllowHostByName. Set to Y to activate feature.
- **AllowHostByName** specifies the Hostnames that MQSSX will perform a gethostbyaddr() call against to compare the returned IP address against the incoming IP address to allow the incoming connection.

The server-side security exit will perform a gethostbyaddr() call against hostnames from the **AllowHostByName** keyword and use the returned IP address and compare the returned IP address.

Note: AllowHostByName must NOT exceed 2048 characters.

Separate each Hostname pattern with a ';' semi-colon.

```
UseAllowHostByName=Y
AllowHostByName=abc01.acme.com;abc02.acme.com
```

4.6 Allow or Restrict the Incoming SSL DN

This section describes the necessary entries to enable the feature that allows or restricts the incoming SSL DN through the use of regular expression patterns. This feature uses the following two keywords:

- **UseAllowSSLDN** controls the use of AllowSSLDN. Set to Y to activate feature.
- **AllowSSLDN** specifies the regular expression patterns that limit the allowable incoming SSL DN

The server-side security exit will look up the regular expression patterns from the **AllowSSLDN** keyword in order to determine if the entire incoming SSL DN matches any of the specified expression patterns. Each regular expression pattern is separated from the next pattern by a semi-colon (';').

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any character in the specified set,
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

Note: AllowSSLDN must NOT exceed 2048 characters.

```
UseAllowSSLDN=Y
AllowSSLDN=O=Capitalware,DC=net;CN=roger;O=acme
```

4.7 Allow or Restrict the Incoming UserID

This section describes the necessary entries to enable the feature that allows or restricts the incoming UserIDs through the use of regular expression patterns. This feature uses the following one keyword:

- **AllowUserID** specifies the regular expression patterns that limit the allowable incoming UserIds

The server-side security exit will look up the regular expression patterns from the **AllowUserID** keyword in order to determine if the entire incoming UserID matches any of the specified expression patterns. Each regular expression pattern is separated from the next pattern by a semi-colon (';').

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any character in the specified set,
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

Note: AllowUserID must NOT exceed 2048 characters.

```
AllowUserID=mq*;hr[0-9][a-f];abc??01
```

4.8 Reject the Incoming IP Address

This section describes the necessary entries to enable the feature that rejects the incoming IP addresses through the use of regular expression patterns. This feature uses the following two keywords:

- **UseRejectIP** controls the use of RejectIP. Set to Y to activate feature.
- **RejectIP** specifies the regular expression patterns that explicitly reject incoming IP Address

The server-side security exit will look up the regular expression patterns from the **RejectIP** keyword in order to determine if the entire incoming IP address matches any of the specified expression patterns. Each regular expression pattern is separated from the next pattern by a semi-colon (';').

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any character in the specified set,
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

Note: RejectIP must NOT exceed 2048 characters.

```
UseRejectIP=Y
RejectIP=192.161.*.*;10.13[0-9].2[0-5][0-9];10.10.1.15
```

4.9 Reject by Hostname

This section describes the necessary entries to enable the feature that rejects by the Hostnames through the use of regular expression patterns. This feature uses the following two keywords:

- **UseRejectHostname** controls the use of RejectHostname. Set to Y to activate feature.

- **RejectHostname** specifies the regular expression patterns that explicitly reject by hostname

The server-side security exit will look up the regular expression patterns from the **RejectHostname** keyword in order to determine if the entire incoming Hostname matches any of the specified expression patterns. Each regular expression pattern is separated from the next pattern by a semi-colon (';').

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any character in the specified set,
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

Note: RejectHostname must NOT exceed 2048 characters.

Separate each Hostname pattern with a ';' semi-colon.

```
UseReject=Y
RejectHostname=xyz01.acme.com;xyz02.acme.com
```

4.10 Reject by Incoming IP against IP of Hostname

This section describes the necessary entries to enable the feature that rejects the incoming IP against IP of Hostnames that MQSSX will perform a gethostbyaddr() call against to compare the returned IP address against the incoming IP address. This feature uses the following two keywords:

- **UseRejectHostByName** controls the use of RejectHostByName. Set to Y to activate feature.

- **RejectHostByName** specifies the Hostnames that MQSSX will perform a gethostbyaddr() call against to compare the returned IP address against the incoming IP address.

The server-side security exit will perform a gethostbyaddr() call against hostnames from the **RejectHostByName** keyword and used the returned IP address and compare the returned IP address to reject the incoming connection.

Note: RejectHostByName must NOT exceed 2048 characters.

Separate each Hostname pattern with a ';' semi-colon.

```
UseReject=Y
RejectHostByName=xyz01.acme.com;xyz02.acme.com
```

4.11 Reject the Incoming SSL DN

This section describes the necessary entries to enable the feature that rejects the incoming SSL DN through the use of regular expression patterns. This feature uses the following two keywords:

- **UseRejectSSLDN** controls the use of RejectSSLDN. Set to Y to activate feature.

- **RejectSSLDN** specifies the regular expression patterns that reject incoming SSL DN

The server-side security exit will look up the regular expression patterns from the **RejectSSLDN** keyword in order to determine if the entire incoming SSL DN matches any of the specified expression patterns. Each regular expression pattern is separated from the next pattern by a semi-colon (';').

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any character in the specified set,
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

Note: RejectSSLDN must NOT exceed 2048 characters.

```
UseRejectSSLDN=Y
RejectSSLDN=O=xyz*;O=abc*;
```

4.12 Reject the Incoming UserID

This section describes the necessary entries to enable the feature that rejects the incoming UserIDs through the use of regular expression patterns. This feature uses the following two keywords:

- **UseRejectUserID** controls the use of RejectUserID. Set to Y to activate feature.
- **RejectUserID** specifies the regular expression patterns that reject incoming UserId

The server-side security exit will look up the regular expression patterns from the **RejectUserID** keyword in order to determine if the entire incoming UserID matches any of the specified expression patterns. Each regular expression pattern is separated from the next pattern by a semi-colon (';').

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any character in the specified set,
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

Note: RejectUserID must NOT exceed 2048 characters.

```
UseRejectUserID=Y
RejectUserID=abc*;x[0-9][a-f]
```

4.13 Excessive Client Connections

This section describes the necessary entries to configure Excessive Client Connections (ECC) alert system in MQSSX. This is controlled by the IniFile's property keyword 'UseECC'.

ECC is an alert system that counts the number of connections over a period of time (i.e. Day / Hour / Minute) and writes a message to the log when the count exceeds a particular value. If the keyword WriteToEventQueue is set to ‘Y’, an event message is also written to an event queue. ECC feature is designed to catch applications that are poorly written, such as, applications that continuously connect and disconnect from the queue manager for every message sent or received.

To enable the alerting of excessive client connections, you need 3 keywords in the IniFile:

- **UseECC** enables excessive client connections feature
- **ECCWarnCount** specifies a count which, when exceeded, will cause an alert to be generated. The default value is 5000.
- **ECCInterval** specifies a time interval to monitor the incoming number of connections. Valid values are D/H/M (Day, Hour and Minute) The default value is 'D'.

```
UseECC=Y
ECCWarnCount=200
ECCInterval=H
```

4.14 Set Maximum Number of Incoming Connections per Channel

This section describes the necessary entries to set a maximum number of allowable connections per a given channel. This is controlled by the IniFile's property keyword 'UseMCC'. Setting 'UseMCC' to 'Y' (Yes) will cause the server-side security exit to look up channel's name as a property keyword in the IniFile.

To enable the restricting of allowable connections per a given channel, you need 7 keywords in the IniFile:

1. **UseMCC** enables restricting of allowable connections per a given channel
2. **DefaultMCC** specifies the default maximum allowable connections for a particular channel.
3. **MCCEventWarnLevel** specifies the percentage of incoming channels to the maximum allowable number of channels that will cause MQSSX to write a warning message to the event queue. The default value is 80.
4. **UseMCCRedo** keyword specifies whether or not the PCF 'display channel status' command should be issued. The default value is 'Y'.
5. **MCCRedoMinutes** specifies a time internal to issue the 'display channel status' command. The default value is 720 minutes.
6. **MCCRedoCount** specifies how often the 'display channel status' command should be issued. The default value is 5000.
7. **MCCGetTimeOut** specifies how long the security exit will wait for the reply from the queue manager's command server. The default value is 3 seconds.

For example, if 'UseMCC' is set to 'Y' and the incoming connection is on 'SYSTEM.ADMIN.SVRCONN', the server-side security exit will look up in the IniFile the keyword of 'SYSTEM.ADMIN.SVRCONN'. If the 'SYSTEM.ADMIN.SVRCONN' keyword is not found, then the server-side security exit will look up 'DefaultMCC' keyword in the IniFile.

If the 'DefaultMCC' keyword is not found, the 'UseMCC' keyword is then switched to 'N' (No).

The server-side security exit uses shared memory to keep track of the channel connection and disconnection. MQSSX was designed to periodically refresh the shared memory counter by issuing a PCF command to get the current channel status. This information is written to the shared memory.

There are 2 IniFile keywords to control how often the PCF Inquire channel status command is issued: 'MCCRedoMinutes' and 'MCCRedoCount'. 'MCCRedoMinutes' keyword states that the server-side security exit should issue PCF command if more than 'x' minutes have passed since the last PCF command was issued. The default value for 'MCCRedoMinutes' is 720 minutes. 'MCCRedoCount' keyword states that the server-side security exit should issue PCF command if more than 'x' connection attempts passed since the last PCF command was issued. The default value for 'MCCRedoCount' is 5000.

MCCEventWarnLevel keyword states that the server-side security exit should write a warning message to the event queue when the number of connections exceeds the percentage level. The default value for 'MCCEventWarnLevel' is 80. Note: Only used if both UseMCC and WriteToEventQueue are each set to 'Y'.

MCCGetTimeOut keyword specifies how long the security exit should wait for a reply from the queue manager's command server. The default value is 3 seconds.

```
UseMCC=Y
SYSTEM.ADMIN.SVRCONN=5
ABC.CH01=50
DEF.CH01=40
SYSTEM.DEF.SVRCONN=5
#
DefaultMCC=25
#
MCCRedoMinutes=900
MCCRedoCount=2000
MCCGetTimeOut=5
```

Note: For queue managers with thousands for active connections, the user may wish to increase the values for 'MCCRedoMinutes' and 'MCCRedoCount' to a higher value. This will keep the overhead to a minimum.

```
MCCRedoMinutes=1440
MCCRedoCount=8000
```

4.15 Proxy ID

This section describes the necessary steps to enable the use of 'Proxy IDs'. Proxy ID allows an authorized User to use a different UserID for MQ interactions.

- **UseProxy** allows an authorized User to use a different UserID for MQ interactions.
- **ProxyFile** specifies the location of the file to do alternate UserID look up.

```
UseProxy=Y
ProxyFile=PROXY
```

The format of the Proxy file is similar to an IniFile or properties file where each keyword has an associated value. Each keyword and its value is on a separate line. The format is as follows:

```
Validated_UserID = ProxyID
```

Example:

```
Roger=app1
Fred=app2
Barney=app1
```

If the UserID is not found in the Proxy file then the incoming connection is rejected. To have a default Proxy UserID in the Proxy file use the "DefaultProxyID" value.

Example:

```
DefaultProxyID=readonly
```

4.16 Allow Users to Login as 'mqm'

This section describes the necessary entries to enable users to login with the mqm or MUSR_MQADMIN or QMQM system account. This is controlled by the IniFile's property keyword 'Allowmqm'. Setting 'Allowmqm' to 'Y' (Yes) will activate this feature; otherwise, it will be blocked.

```
Allowmqm=Y
```

4.17 UserIDFormatting

This section describes the necessary entries on how to handle the incoming UserID. 'UserIDFormatting' supports 3 values [A / U / L]. ('As Is, Uppercase and Lowercase). The default value is A.

```
UserIDFormatting=U
```

4.18 Allow connection to have a blank UserID

This section describes the necessary entries to enable connection to have a blank UserID. This is controlled by the IniFile's property keyword 'AllowBlankUserID'. Setting 'AllowBlankUserID' to 'Y' (Yes) will allow connections to have a blank UserID.

```
AllowBlankUserID=Y
```

4.19 MCAUSER Field

This section describes the necessary steps to enable the use of the channel's MCAUSER field. If this IniFile parameter is set to 'Y' (Yes) then after the authentication process is complete, the connection will use the UserID value specified in the MCAUSER field.

- **UseMCAUser** enables the connection to use the UserID value specified in the channel's MCAUSER field

```
UseMCAUser=Y
```

4.20 CheckFinalUserID

This section describes the necessary entries to enable z/MQSSX to process the final UserID against UseAllowUserID, AllowUserID, UseRejectUserID, RejectUserID and Allowmqm keywords.

```
CheckFinalUserID=Y
```

4.21 SSL Self-Signed Certificate

This section describes the necessary steps to allow or reject SSL Self-Signed Certificates. If the AllowSSLSSCert IniFile parameter is set to 'Y' (Yes) then the SSL Self-Signed Certificate are allowed. If AllowSSLSSCert is set to 'N' (No), the SSL Self-Signed Certificate is disallowed (i.e. the incoming connection is closed).

- **AllowSSLSSCert** specifies whether or not to allow the Self-Signed Certificate on the channel.

```
AllowSSLSSCert=Y
```

4.22 Set UserID from SSL DN

MQSSX supports the retrieval of the UserID from the channel's SSL DN field. To enable the retrieval of the UserId from the channel's SSL DN field, you may use the following 4 keywords in the IniFile:

- **UseSSLUserIDFromDN** specifies that the UserID is to be retrieved from a SSL DN entry.
- **SSLDNAttrName** specifies the SSL DN attribute field name
- **SSLDNAttrStartPos** specifies the start position of the retrieval
- **SSLDNAttrLength** specifies the length of the field to be extracted (* means all)

```
UseSSLUserIDFromDN = Y
SSLDNAttrName = CN
SSLDNAttrStartPos = 1
SSLDNAttrLength = *
```

4.23 LicenseFile

This section will describe how to have a file that contains all of the user's MQSSX license keys.

The format of the LicenseFile is similar to an IniFile or properties file where each keyword has an associated value. Each keyword and its value are on a separate line. The format is as follows:

```
    QMgrName = License_Key
```

Example:

```
MQA1 = 10S0-AAAA-BBBBBBBB
MQB1 = 10S0-XXXX-CCCCCCCC
```

If the queue manager name is not found in the LicenseFile then the License keyword will be used to retrieve the license key value.

The following are the default values for LicenseFile:

For Windows:

```
    LicenseFile=C:\Capitalware\MQSSX\mqssx_licenses.ini
```

For IBM MQ 32-bit on Unix and Linux:

```
    LicenseFile=/var/mqm/exits/mqssx_licenses.ini
```

For IBM MQ 64-bit on Unix and Linux:

```
    LicenseFile=/var/mqm/exits64/mqssx_licenses.ini
```

For IBM MQ on IBM i:

```
    LicenseFile=/QIBM/UserData/mqm/mqssx/mqssx_licenses.ini
```

4.24 License Key

This section will describe how to license MQ Standard Security Exit to a particular queue manager.

Note: The License keyword is not required if the user has implemented the LicenseFile keyword or the License file actually exists in the default location.

Your license will look something like: 10S0-AAAA-BBBBBBBB (Note: This is a sample license only and will NOT work).

```
License=10S0-AAAA-BBBBBBBB
```

5 Miscellaneous

This section describes the extra files that were included to help the user get MQSSX up and running in a very quick manner.

5.1 Windows

Sample IniFile

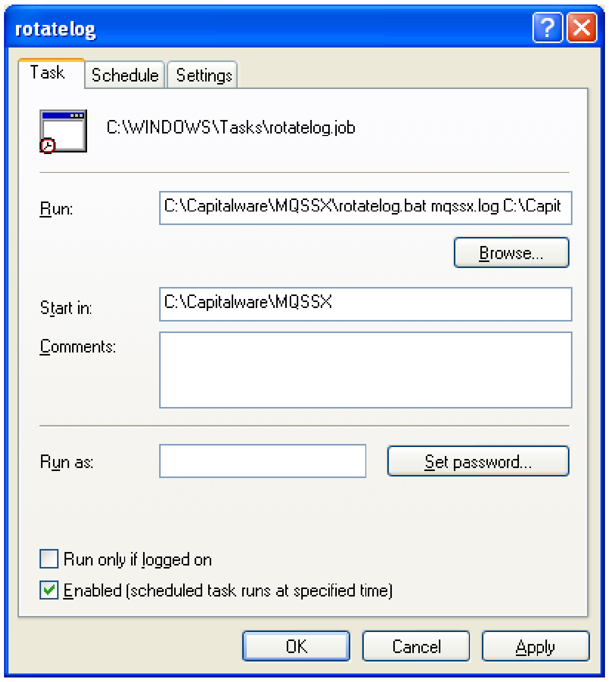
The '***mqssx.ini***' file is a basic MQSSX IniFile. It has the standard IniFile parameters that the user may need to use or update. The '***mqssx.ini.readme***' file is a plain text help file with a description of the parameters.

Sample MQSC scripts

The '***mqssx.sample.mqsc***' file is a sample MQSC script to update the 2 system defined channels with the MQSSX security exit information.

Rotate log script

The '***rotatelog.bat***' file is a Windows batch script to rotate (backup) the mqssx.log file. Actually, it is generic in implementation; hence, it can be used to rotate any log file that the user wishes to be rotated. The batch script requires 2 parameters: log file name and the directory of log file.



rotatelog

Task | Schedule | Settings

C:\WINDOWS\Tasks\rotatelog.job

Run: C:\Capitalware\MQSSX\rotatelog.bat mqssx.log C:\Capit

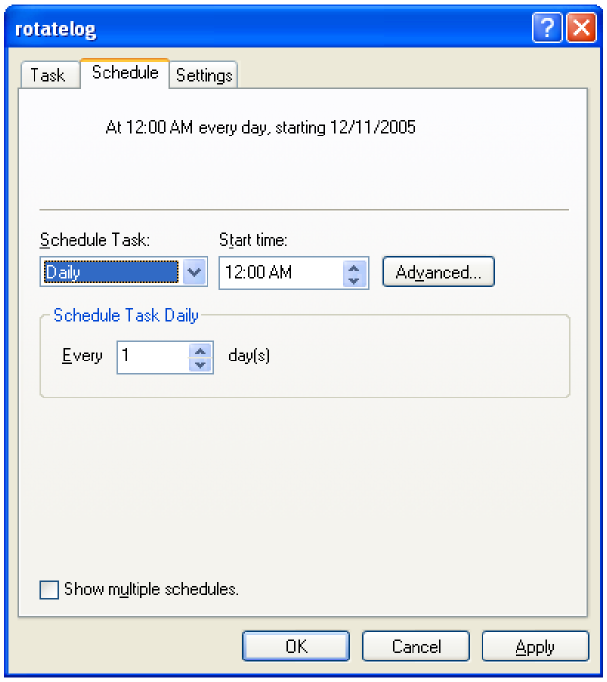
Start in: C:\Capitalware\MQSSX

Comments:

Run as:

☐ Run only if logged on

☑ Enabled (scheduled task runs at specified time)



rotatelog

Task | Schedule | Settings

At 12:00 AM every day, starting 12/11/2005

Schedule Task: Daily

Start time: 12:00 AM

Schedule Task Daily

Every 1 day(s)

☐ Show multiple schedules.

5.2 Unix and Linux

Sample IniFile
The '***mqssx.ini***' file is a basic MQSSX IniFile. It has the standard IniFile parameters that the user may need to use or update. The '***mqssx.ini.readme***' file is a plain text help file with a description of the parameters.

Sample MQSC scripts
The '***mqssx.sample.mqsc***' file is a sample MQSC script to update the 2 system defined channels with the MQSSX security exit information.

Rotate log script
The '***rotatelog.sh***' file is a Unix / Linux shell script to rotate (backup) the mqssx.log file. Actually, it is generic in implementation; hence, it can be used to rotate any log file that the user wishes to be rotated. The shell script requires 2 parameters: log file name and the directory of log file.

Sample daily CRON entry for IBM MQ 32-bit on Unix and Linux:

```
0 0 * * * /var/mqm/exits/rotatelog.sh mqaux.log /var/mqm/exits/  > /tmp/mqssx.log.run 2 > &1
```

Sample daily CRON entry for IBM MQ 64-bit on Unix and Linux:

```
0 0 * * * /var/mqm/exits64/rotatelog.sh mqaux.log /var/mqm/exits64/  > /tmp/mqssx.log.run 2 > &1
```

5.3 IBM i

Sample IniFile
The '***mqssx.ini***' file is a basic MQSSX IniFile. It has the standard IniFile parameters that the user may need to use or update. The '***mqssx.ini.readme***' file is a plain text help file with a description of the parameters.

Sample MQSC scripts
The '***mqssx.sample.mqsc***' file is a sample MQSC script to update the 2 system defined channels with the MQSSX security exit information.

5.4 Server-side Log File

To verify that the process flow was successful, you can view the log file for the events that are generated.

```
2006/03/23 19:47:11 MQSSX #01767 I: Connection accepted for QMgr='MQS1' ChlName='MY.TEST.EXIT' ConName='192.168.10.101' R
2006/03/23 19:47:31 MQSSX #01767 I: Connection accepted for QMgr='MQS1' ChlName='MY.TEST.EXIT' ConName='192.168.10.101' R
```

6 Appendix A – Summary of IniFile (Server-side)

The sample IniFile below is the mqssx.ini file supplied for Windows. The IniFile supports the following keywords and their values:

```
LogMode=N
LogFile=C:\Capitalware\MQSSX\mqssx.log
AllowUserID=mq*;abc[0-9]??;HR???
Allowmqm=N
AllowBlankUserID=N
UseMCC=N
UseAllowIP=N
UseProxy=N
```

Note: Keywords are case sensitive.

Keyword	Description of Server-side keywords
AllowBlankUserID	**AllowBlankUserID** specifies whether or not to allow the incoming connection to have a blank UserID value. AllowBlankUserID supports 2 values [Y / N]. The default value is N. e.g. AllowBlankUserID=Y
AllowHostByName	**AllowHostByName** specifies the Hostnames that MQSSX will perform a gethostbyaddr() call against to compare the returned IP address against the incoming IP address to allow the incoming connection. The default is '*'. You must separate the hostname regular expression patterns with a ';' semi-colon. e.g. AllowHostByName=abc01.acme.com;abc02.acme.com Note: Only used if UseAllowHostByName is set to 'Y'.
AllowHostname	**AllowHostname** specifies a set of regular expression patterns that the hostname will be compared against. The default is '*'. You must separate the hostname regular expression patterns with a ';' semi-colon. e.g. AllowHostname=abc01.acme.com;abc02.acme.com Note: Only used if UseAllowHostname is set to 'Y'.

Keyword	Description of Server-side keywords
AllowIP	**AllowIP** specifies a set of regular expression patterns that the incoming channel's IP address will be compared against. The default is '*'. You must separate the IP regular expression patterns with a ';' semi-colon. e.g. AllowIP=192.168.*.1[0-5][0-9];127.0.0.?;10.*.*.[0-9] Note: Only used if UseAllowIP is set to 'Y'.
Allowmqm	**Allowmqm** specifies whether or not to allow a user to be able to login using 'mqm' (Unix), 'MUSR_MQADMIN' (Windows) or 'QMQM' (OS/400) system account. Allowmqm supports 2 values [Y / N]. The default value is N. e.g. Allowmqm=Y
AllowSSLDN	**AllowSSLDN** specifies a set of regular expression patterns that the incoming channel's SSL DN will be compared against. You must separate the SSL DN regular expression patterns with a ';' semi-colon. e.g. AllowSSLDN=O=Capitalware,C=CA;O=IBM,DC=com Note: Only used if UseAllowSSLDN is set to 'Y'.
AllowSSLSSCert	**AllowSSLSSCert** specifies whether or not to allow Self-Signed Certificate on the channel. AllowSSLSSCert supports 2 values [Y / N]. The default value is Y. e.g. AllowSSLSSCert=Y
AllowUserID	**AllowUserID** specifies a set of regular expression patterns that the incoming connection's UserID will be parsed against. The default is '*'. You must separate each IP regular expression pattern with a ';' semi-colon. e.g. AllowUserID=mq*;abc??;xyz[0-9][a-f];hr[0-9][0-9]
BackupLogFileCount	**BackupLogFileCount** specifies the number of backup logfiles that the security exit will be keeping. The default value is 9. e.g. BackupLogFileCount=9

Keyword	Description of Server-side keywords
CheckFinalUserID	**CheckFinalUserID** specifies whether or not the final UserID will be checked against the UseAllowUserID, AllowUserID, UseRejectUserID, RejectUserID and Allowmqm keywords. CheckFinalUserID supports 2 values [Y / N]. The default value is N. e.g. CheckFinalUserID=Y
DefaultMCC	**DefaultMCC** specifies a default maximum number of incoming connections that a particular channel will allow. There is no default value. e.g. DefaultMCC=25
ECCInterval	**ECCInterval** specifies a time interval to monitor the incoming number of connections. Valid values are D/H/M (Day, Hour and Minute) The default value is 'D'. e.g. ECCInterval =H Note: Only used if UseECC is each set to 'Y'.
ECCWarnCount	**ECCWarnCount** specifies a count which, when exceeded, will cause an alert to be generated. The default value is 5000. e.g. ECCWarnCount =4000 Note: Only used if UseECC is each set to 'Y'.
EventQueueName	**EventQueueName** specifies the name of the event queue. The default value is 'SYSTEM.ADMIN.CHANNEL.EVENT'. e.g. EventQueueName= SYSTEM.ADMIN.CHANNEL.EVENT Note: Only used if WriteToEventQueue is set to 'Y'.
Groups	**Groups** specifies the list of groups that the authorizations will be performed against. e.g. Groups=grpA;grpF;grpM

Keyword	Description of Server-side keywords
GroupFile	**GroupFile** specifies the location of the group file. The default value is 'groups.ini'. e.g. GroupFile=C:\Capitalware\MQSSX\groups.ini Note: Only used if UseGroupFile is set to 'Y'.
License	**License** specifies the queue manager's license key. Your license will look something like: 10S0-AAAA-BBBBBBBB (Note: This is a sample license only and will NOT work). e.g. License=10S0-AAAA-BBBBBBB
LicenseFile	**LicenseFile** specifies the location of License file that contains all of the customer's license keys. The following are the default values for LicenseFile: For Windows: LicenseFile=C:\Capitalware\MQSSX\mqssx_licenses.ini For IBM MQ 32-bit on Unix and Linux: LicenseFile=/var/mqm/exits/mqssx_licenses.ini For IBM MQ 64-bit on Unix and Linux: LicenseFile=/var/mqm/exits64/mqssx_licenses.ini For IBM MQ on IBM i: LicenseFile=/QIBM/UserData/mqm/mqssx/mqssx_licenses.ini e.g. LicenseFile=/var/mqm/exits64/mqssx_licenses.ini
LogDiscMessage	**LogDiscMessage** specifies whether or not a disconnect message is outputted to the logfile. LogDiscMessage supports 2 values [Y / N]. The default value is N. e.g. LogDiscMessage=Y

Keyword	Description of Server-side keywords
LogFile	**LogFile** specifies the location of the log file. The default is as follows: For Windows: LogFile=C:\Capitalware\MQSSX\mqssx.log For IBM MQ 32-bit on Linux: LogFile=/var/mqm/exits/mqssx.log For IBM MQ 64-bit on Unix and Linux: LogFile=/var/mqm/exits64/mqssx.log For IBM MQ on IBM i: LogFile=/QIBM/UserData/mqm/mqssx/mqssx.log
LogMessageQuote	**LogMessageQuote** specifies what type of quote (single or double) is to be used with the log message. LogMessageQuote supports 2 values [' / "] (single or double quote). The default value is ' (single quote). e.g. LogMessageQuote="
LogMode	**LogMode** specifies what type of logging the user wishes to have. LogMode supports 4 values [Q / N / V / D] where Q is Quiet, N is Normal, V is Verbose and D is Debug. The default value is N. e.g. LogMode=N
MCCEventWarnLevel	**MCCEventWarnLevel** specifies the percentage of incoming channels to the maximum allowable number of channels that will cause MQSSX to write a warning message to the event queue. The default value is 80. e.g. MCCEventWarnLevel =80 Note: Only used if both UseMCC and WriteToEventQueue are each set to 'Y'.
MCCGetTimeOut	**MCCGetTimeOut** specifies the number of seconds that the security exit will wait for a reply form the queue manager's command server. The default value is 3. e.g. MCCGetTimeOut=3 Note: Only used if UseMCC is set to 'Y'.

Keyword	Description of Server-side keywords
MCCRedoCount	**MCCRedoCount** specifies the number of connection attempts to occur before the next PCF 'display current channel status' is issued. The default value is 5000. e.g. MCCRedoCount=5000 Note: Only used if UseMCC is set to 'Y'.
MCCRedoMinutes	**MCCRedoMinutes** specifies the period of time in minutes to wait before the next PCF 'display current channel status' is issued. The default value is 720 minutes. e.g. MCCRedoMinutes=720 Note: Only used if UseMCC is set to 'Y'.
ProxyFile	**ProxyFile** specifies the location of the file to do alternate UserID look-up. The default value is proxy.lst e.g. ProxyFile=C:\proxy.lst Note: Only used if UseProxy is set to 'Y'.
RejectHostByName	**RejectHostByName** specifies a list of hostnames that MQSSX will perform a gethostbyaddr() call against the hostname to compare the returned IP address against the incoming IP address to reject the incoming connection. You must separate the hostnames with a ';' semi-colon. e.g. RejectHostByName=xyz01.acme.com;xyz02.acme.com Note: Only used if UseRejectHostByName is set to 'Y'.
RejectHostname	**RejectHostname** specifies a set of regular expression patterns that the hostname will be compared against. You must separate the hostname regular expression patterns with a ';' semi-colon. e.g. RejectHostname=xyz01.acme.com;xyz02.acme.com Note: Only used if UseRejectHostname is set to 'Y'.

Keyword	Description of Server-side keywords
RejectIP	**RejectIP** specifies a set of regular expression patterns that the incoming channel's IP address will be compared against. You must separate the IP regular expression patterns with a ';' semi-colon. e.g. RejectIP=192.168.*.1[0-5][0-9];127.0.0.?;10.*.*.[0-9] Note: Only used if UseAllowIP is set to 'Y'.
RejectSSLDN	**RejectSSLDN** specifies a set of regular expression patterns that the incoming channel's SSL DN will be compared against. You must separate the SSL DN expression patterns with a ';' semi-colon. e.g. RejectSSLDN=O=xyz*,C=CA;O=abc*,DC=net Note: Only used if UseRejectSSLDN is set to 'Y'.
RejectUserID	**RejectUserID** specifies a set of regular expression patterns that the incoming connection's UserID will be compared against. You must separate each IP regular expression pattern with a ';' semi-colon. e.g. RejectUserID=mq*;abc??;xyz[0-9][a-f];hr[0-9][0-9] Note: Only used if UseRejectUserID is set to 'Y'.
RotateLogDaily	**RotateLogDaily** specifies whether or not daily log file rotation should take place. RotateLogDaily supports 2 values [Y / N]. The default value is Y. e.g. RotateLogDaily=Y
SSLDNAttrLength	**SSLDNAttrLength** specifies the length of the extraction of the UserId from the SSL DN attribute. The default value is '*' (* means all). e.g. SSLDNAttrLength=* Note: Only used if UseSSLUserIDFromDN is set to 'Y'.

Keyword	Description of Server-side keywords
SSLDNAttrName	**SSLDNAttrName** specifies the name of SSL DN attribute to be used to extract UserId from. The default value is 'CN'. e.g. SSLDNAttrName=CN Note: Only used if UseSSLUserIDFromDN is set to 'Y'.
SSLDNAttrStartPos	**SSLDNAttrStartPos** specifies the start position for the extraction of the UserId from the SSL DN attribute. The default value is '1'. e.g. SSLDNAttrStartPos=1 Note: Only used if UseSSLUserIDFromDN is set to 'Y'.
SystemLogMessage	**SystemLogMessage** specifies what messages will be written to the system log.. SystemLogMessage supports 3 values [B / A / R] where B is Both, A is Accepted Only, and R is Rejected Only messages. The default value is B. e.g. SystemLogMessage=B Note: Only used if WriteToSystemLog is set to 'Y'.
UseAllowHostByName	**UseAllowHostByName** allows MQ Admin to allow or restrict by performing a gethostbyaddr() call against the hostname to compare the returned IP address against the incoming IP address to allow the incoming connection. UseAllowHostByName supports 2 values [Y / N]. The default value is N. e.g. UseAllowHostByName=Y
UseAllowHostname	**UseAllowHostname** allows MQ Admin to allow or restrict by hostname by comparing it against a regular expression pattern. UseAllowHostname supports 2 values [Y / N]. The default value is N. e.g. UseAllowHostname=Y
UseAllowIP	**UseAllowIP** allows MQ Admin to allow or restrict incoming channel IP address by comparing it against a regular expression pattern. UseAllowIP supports 2 values [Y / N]. The default value is N. e.g. UseAllowIP=Y

Keyword	Description of Server-side keywords
UseAllowSSLDN	**UseAllowSSLDN** allows MQ Admin to allow or restrict incoming channel's SSL DN by comparing it against a regular expression pattern. UseAllowSSLDN supports 2 values [Y / N]. The default value is N. e.g. UseAllowSSLDN=Y
UseECC	**UseECC** allows MQ Admin to have MQSSX generate an alert when the ECCWarnCount is exceeded. UseECC supports 2 values [Y / N]. The default value is N. e.g. UseECC=Y
UseGroups	**UseGroups** allows or restricts the incoming UserID against an OS group or a group file. UseGroups supports 2 values [Y / N]. The default value is N. e.g. UseGroups=Y
UseGroupFile	**UseGroupFile** specifies whether or not a group file. UseGroupFile supports 2 values [Y / N]. The default value is N. e.g. UseGroupFile=Y
UseMCAUser	**UseMCAUser** allows the connection to use the UserID value specified in the channel's MCAUSER field. UseMCAUser supports 2 values [Y / N]. The default value is N. e.g. UseMCAUser=Y
UseMCC	**UseMCC** allows MQ Admin to set a limit on the maximum number of connections to a given channel. UseMCC supports 2 values [Y / N]. The default value is N. e.g. UseMCC=Y
UseMCCRedo	**UseMCCRedo** keyword specifies whether or not the server-side security exit should issue PCF command. UseMCCRedo supports 2 values [Y / N]. The default value is Y. e.g. UseMCCRedo=Y

Keyword	Description of Server-side keywords
UseProxy	**UseProxy** allows an authorized User to use a different UserID for MQ interactions. UseProxy supports 2 values [Y / N]. The default value is N. e.g. UseProxy=N
UseRejectHostByName	**UseRejectIHostByName** allows MQ Admin to perform a gethostbyaddr() call against the hostname to compare the returned IP address against the incoming IP address to reject the incoming connection. UseRejectHostByName supports 2 values [Y / N]. The default value is N. e.g. UseRejectHostByName=Y
UseRejectHostname	**UseRejectIHostname** allows MQ Admin to reject a hostname by comparing it against a regular expression pattern. UseRejectHostname supports 2 values [Y / N]. The default value is N. e.g. UseRejectHostname=Y
UseRejectIP	**UseRejectIP** allows MQ Admin to reject incoming channel IP address by comparing it against a regular expression pattern. UseRejectIP supports 2 values [Y / N]. The default value is N. e.g. UseRejectIP=Y
UseRejectSSLDN	**UseRejectSSLDN** allows MQ Admin to reject incoming channel's SSL DN by comparing it against a regular expression pattern. UseRejectSSLDN supports 2 values [Y / N]. The default value is N. e.g. UseRejectSSLDN=Y
UseRejectUserID	**UseRejectUserID** allows MQ Admin to reject incoming UserID by comparing it against a regular expression pattern. UseRejectUserID supports 2 values [Y / N]. The default value is N. e.g. UseRejectUserID=Y

Keyword	Description of Server-side keywords
UserIDFormatting	**UserIDFormatting** specifies how MQSSX will handle the incoming UserID. UserIDFormatting supports 3 values [A / U / L]. ('As Is, Uppercase and Lowercase). The default value is A. UserIDFormatting=U
UseSSLUserIDFromDN	**UseSSLUserIDFromDN** specifies to set the channel's UserId to be the value extracted from the SSL DN. UseSSLUserIDFromDN supports 2 values [Y / N]. The default value is N. e.g. UseSSLUserIDFromDN=Y
WriteToEventQueue	**WriteToEventQueue** specifies if MQSSX will write an event message containing the log entry information to an event queue. WriteToEventQueue supports 2 values [Y / N]. The default value is N. e.g. WriteToSystemLog =Y
WriteToSystemLog	**WriteToSystemLog** specifies if MQSSX write a log entry to the server's 'logging system'. On Windows, the server's 'logging system' is the Event Log and on Unix/Linux it is the syslog. WriteToSystemLog supports 2 values [Y / N]. The default value is N. The Unix / Linux syslog output can be found for each operating system as follows: ➢ AIX: /var/log/messages ➢ HP-UX: /var/adm/syslog/syslog.log ➢ Linux: /var/log/messages ➢ Solaris: /var/adm/messages e.g. WriteToSystemLog =Y

7 Appendix B – MQSSX Upgrade Procedures

To upgrade an existing installation of MQSSX from an older version to a newer version, do please do the following in the appropriate section below.

7.1.1 Windows Upgrade

- Stop all of the channels using the MQSSX server-side security exit or completely stop the queue manager.
- Backup all MQSSX IniFiles in the MQSSX install directory
- If MQSSX was installed using the Windows Installer then
  - Click the ***Start*** -> ***All Programs*** -> ***Control Panel*** -> ***Add or Remove Programs***, select MQSSX from the list and click the ***Remove*** button then follow the prompts to remove it
  - Run the **mqssx-setup.exe** file from the **Windows** directory to install the new version
- Otherwise copy the following files (latest version) to the MQSSX install directory:
  - mqssx.dll
  - cwchad.dll
  - AddRegistryEntries.bat
  - mqssx.reg
  - rotatelog.bat
- Run AddRegistryEntries.bat batch file
- Restore the MQSSX IniFiles if they were altered / deleted.
- Start all of the channels using the MQSSX server-side security exit or restart the queue manager if it was previously stopped.

7.1.2 Linux 32-bit Upgrade

- Login under the mqm account
- Stop all of the channels using the MQSSX server-side security exit or completely stop the queue manager.
- Backup all MQSSX IniFiles in the MQSSX install directory
- Copy the appropriate tar file to the ***/var/mqm/exits/*** directory
- Un-tar the contents of the tar file.
  i.e. For AIX, do the following command:
  ```
  tar -xvf  mqssx_aix.tar
  ```
- Run the script as follows:
  ```
  ./setssx.sh
  ```
- Restore the MQSSX IniFiles if they were altered / deleted.
- Delete the MQSSX tar file
- Start all of the channels using the MQSSX server-side security exit or restart the queue manager if it was previously stopped.

7.1.3 Unix and Linux 64-bit Upgrade

- Stop all of the channels using the MQSSX server-side security exit or completely stop the queue manager.
- Backup all MQSSX IniFiles in the MQSSX install directory
- Copy the appropriate tar file to the ***/var/mqm/exits64/*** directory
- Un-tar the contents of the tar file.
  i.e. For AIX, do the following command:
  ```
  tar -xvf  mqssx_aix.tar
  ```
- Run the script as follows:
  ```
  ./setssx.sh
  ```
- Restore the MQSSX IniFiles if they were altered / deleted.
- Delete the MQSSX tar file
- Start all of the channels using the MQSSX server-side security exit or restart the queue manager if it was previously stopped.

7.1.4 IBM i Upgrade

- Stop all of the channels using the MQSSX server-side security exit or completely stop the queue manager.
- Backup all MQSSX IniFiles in the MQSSX install directory
- ftp the IBM i files to the IBM i server as follows:

```
ftp -s:mqssx_iseries.ftp  iseries_hostname
```

```
your-IBM i-userid
your-IBM i-password

binary
cd QGPL
put mqssx.savf
quote SITE NAMEFMT 1
cd /QIBM/UserData/mqm/
put mqssx_iseries.tar
quit
```

- Log onto the target IBM i server and do the following commands:

```
RSTLIB SAVLIB(MQSSX) DEV(*SAVF) SAVF(QGPL/MQSSX)
CLRSAVF FILE(QGPL/MQSSX)
CHGOBJOWN  OBJ(MQSSX) OBJTYPE(*LIB)  NEWOWN(QMQM)
qsh
cd /QIBM/UserData/mqm/
tar -xvf mqssx_iseries.tar
chown -R QMQM mqssx
rm mqssx_iseries.tar
```

- Restore the MQSSX IniFiles if they were altered / deleted.
- Start all of the channels using the MQSSX server-side security exit or restart the queue manager if it was previously stopped.

8 Appendix F – Capitalware Product Display Version

MQSSX includes a program to display the product version number. The command to display the product version number is:

`cwdspver`

8.1 Examples

8.1.1 Windows

To use the cwdspver program on Windows, open a Command prompt and change the directory to `C:\Capitalware\MQSSX\` and type the following:

```
cwdspver.exe
```

8.1.2 Linux 32-bit

To use the cwdspver program on Linux for MQ 32-bit, open a shell prompt and change directory to `/var/mqm/exits/` and type the following:

```
./cwdspver
```

8.1.3 Unix and Linux 64-bit

To use the cwdspver program on Unix/Linux for MQ 64-bit, open a shell prompt and change directory to `/var/mqm/exits64/` and type the following:

```
./cwdspver
```

8.1.4 IBM i

To use the cwdspver program on IBM i for MQ, issue the following command on the prompt:

```
CALL MQSSX/CWDSPVER
```

9 Appendix D – Support

The support for MQ Standard Security Exit can be found at the following location:

By email at:
support@capitalware.com

By regular mail at:

> Capitalware Inc.
> Attn: MQSSX Support
> Unit 11, 1673 Richmond Street, PMB524
> London, Ontario N6G2N3
> Canada

10 Appendix E – Summary of Changes

- MQ Standard Security Exit v2.6.0
  - Enhanced the code for dumping the pointers passed into exit.
  - Fixed an issue in the subroutine that removes trailing blanks
  - Fixed issue when an invalid or expired license key is used
  - Fixed an issue with default exit path

- MQ Standard Security Exit v2.5.0
  - Tuned the code that is called on entry
  - Tuned the logging code

- MQ Standard Security Exit v2.4.0
  - Fixed an issue in the logging framework where a constant was being modified.

- MQ Standard Security Exit v2.3.0
  - Added support for log disconnect message (new keyword: LogDiscMessage)
  - Added support for single or double quotes for log message (new keyword: LogMessageQuote)
  - Enhanced logging - the LogFile keyword now supports the following tokens: %QM%, %CHL%, %UID%, %PID% & %TID%
  - Fixed an issue with ECC shared memory
  - Fixed an issue with MCC shared memory

- MQ Standard Security Exit v2.2.1
  - Fixed an issue on Windows with freeing environment variable memory (error with FreeEnvironmentStrings Windows API call)
  - Fixed an issue with using "size_t" variable type when it should have been "int"

- MQ Standard Security Exit v2.2.0
  - Ability to monitor for excessive client connections (ECC) and then generate an alert (new keywords: UseECC, ECCWarnCount & ECCInterval)
  - Fixed an issue with the installer script on Windows missing the mqssx_MI.dll file

- MQ Standard Security Exit v2.1.0
  - Added new CheckFinalUserID keyword. It will take the final UserID and reprocess it against UseAllowUserID, AllowUserID, UseRejectUserID, RejectUserID and Allowmqm keywords.
  - Improved the IniFile processing speed.
  - Fixed an issue with Enterprise License key not being loaded from a License file.
  - Fixed an issue with MQSSX not recognizing the filename specified for FBAFile keyword.
  - Tested with MQ v8.0
  - Tested with Windows 8/8.1

- MQ Standard Security Exit v2.0.1
    - Added UseMCCRedo flag to control MCCRedoCount, MCCRedoMinutes and MCCGetTimeOut
    - Added UserIDFormatting flag to force lowercase/uppercase/as_is UserID formatting on all platforms

- MQ Standard Security Exit v2.0.0
    - Added keyword UseAllowHostname and AllowHostname to only allow hosts by name (reverse lookup of incoming IP address)
    - Added keyword UseRejectHostname and RejectHostname to explicitly reject a hostname (reverse lookup of incoming IP address)
    - Added keyword UseAllowHostByName and AllowHostByName to only allow hosts by name
    - Added keyword UseRejectHostByName and RejectHostByName to explicitly reject a hostname
    - Added keyword SystemLogMessage to control what type of messages ('accepted' and/or 'rejected') are written to system log
    - Added keywords UseGroups, Groups, UseGroupFile & GroupFile
    - Added program cwdspver to display the product version number
    - Added code in the Ini parser to distinguish between 'ABC' and 'ABCDEF' keywords
    - Increased the accepted IniFile parameter length from 1024 to 2048 characters
    - Added support non-default install for MQ v7.1 & higher multi-install feature on IBM i, Linux, Unix and Windows
    - Updated the "Connection accepted" log record to include the UserID set for the connection.
    - Updated MCC logic so that a command server failure does not affect the exit.
    - Changed MCCRedoCount default value from 1000 to 5000
    - Fixed a bug with ConnectionName when both IPv4 and IPv6 stacks are used
    - Fixed a bug in the in-memory Ini parser
    - Fixed a bug with Proxy file processing
    - Fixed a bug in the AllowSSLDN processing
    - Fixed a bug in AllowHostname on Linux
    - Fixed an issue with BackupLogFileCount
    - Fixed a bug with SSLPeerNamePtr field.
    - Tested with MQ v7.5

- MQ Standard Security Exit v1.3.0
    - Added AllowSSLSSCert IniFile keyword to enable the check for Self-signed Certificate
    - Added UseSSLUserIDFromDN, SSLDNAttrName, SSLDNAttrStartPos and SSLDNAttrLength IniFile keywords to extract the UserID from the channel's SSL DN field
    - Added LicenseFile to support multiple license keys in a single file
    - Added support for MQSSX_HOME environment variable to explicitly define an IniFile location

    - Fixed a bug with Proxy file processing
    - Fully tested and supported for Windows 7 Professional

- MQ Standard Security Exit v1.2.0
    - New supported platform: IBM i (OS/400)
    - New supported platform: AIX 6.1
    - Major performance and tuning to many modules - a 7% - 12% improvement in speed depending on features used
    - Added the ability to explicitly reject an incoming IP address based on a pattern-matching (UseRejectIP and RejectIP).
    - Added the ability to explicitly reject an incoming UserId based on a pattern-matching (UseRejectUserID and RejectUserID).
    - Added the code to disable Event Warning messages when WriteToEventQueue is being used.
    - Added code to limit the number of messages written to the event queue when WriteToEventQueue is being used.
    - Added MCCGetTimeOut keyword to allow the user to define how long to wait on the "DIS CHL(<ChannelName>)" command when UseMCC is being used.
    - Added BackupLogFileCount which is used to control the number of backup log files (Default value is 9)
    - Fixed a shared memory issue on Windows when UseMCC is being used.

- MQ Standard Security Exit v1.1.4
    - Added the ability to write custom MQ Events to System Channel Event Queue to allow MQSSX to be tied into an MQ Monitoring tool.
        - 9101 for Connection rejected event message
        - 9201 for MCC Warning event message
        - 9202 for MCC Exceeded event message
    - Incorporated the rotatelog.bat and rotatelog.sh scripts into server-side security exit.
    - New supported platform: HP-UX IA64
    - Successfully tested with MQ v7.0
    - Fixed a bug related to a memory leak when using the MCC feature under extreme load.
    - Created a MQSSX manual: MQSSX Queue Manager To Queue Manager Configuration.

- MQ Standard Security Exit v1.1.3
    - Fixed an issue with Max Channel Connections on Solaris 8

- MQ Standard Security Exit v1.1.2
    - Added more debug information related to memory pointers.
    - Changed MCCRedoSeconds keyword to MCCRedoMinutes.
    - Changed default values for MCCRedoMinutes and MCCRedoCount

- MQ Standard Security Exit v1.1.1

   - Updated IniFile keyword handling.

- MQ Standard Security Exit v1.1.0
   - Provided a new graphical program, MQSSX-GUI, to assist the user in creating and managing their MQSSX IniFiles.
   - Added the ability to write log message to system log facility. For Unix and Linux that means syslog and for Windows it means Event Log
   - Added support for relative file access for IniFile, LogFile and ProxyFile. Ideal for cross-platform clustered channels.
   - Created a new memory manager - now uses 35% less memory per connection
   - Improved the speed of the IniFile handle.
   - Fixed a bug: when using UseAllowIP and normal logging, the rejected messages were not being written to the log file.
   - Fixed a bug: when using UseProxy that the DefaultProxyID was not being found.
   - Fixed a bug: in the IniFile when the last line does not have a trailing CRLF for Windows or LF for Unix/Linux, it did not make use of key word on the last line.

- MQ Standard Security Exit v1.0.7
   - Added support for MQSSX on the following 3 new platforms:
      - 'Linux on zSeries both 32-bit and 64-bit'
      - 'Linux on x86_64' (64-bit)
      - 'Solaris 10 on x86_64' (64-bit)

- MQ Standard Security Exit v1.0.6
   - Added tighter controls for allowing the mqm UserId.

- MQ Standard Security Exit v1.0.5
   - Added support for IBM MQ v6.0
   - Added a new supported platform 'Linux on POWER' for MQSSX.

- MQ Standard Security Exit v1.0.2
   - Added error message if specified IniFile does not exist.
   - Added support for older Linux servers with glibc 2.2.5

- MQ Standard Exit v1.0.1
   - Added support for using the channel's MCAUSER value.
   - Enhanced the speed for reading IniFile parameters.

- MQ Standard Security Exit v1.0.0
   - Initial release.

11 Appendix F – License Agreement

This is a legal agreement between you (either an individual or an entity) and Capitalware Inc. By opening the sealed software packages (if appropriate) and/or by using the SOFTWARE, you agree to be bound by the terms of this Agreement. If you do not agree to the terms of this Agreement, promptly return the disk package and accompanying items for a full refund.
SOFTWARE LICENSE

1. GRANT OF LICENSE. This License Agreement (License) permits you to use one copy of the software product identified above, which may include user documentation provided in on-line or electronic form (SOFTWARE). The SOFTWARE is licensed as a single product, to an individual queue manager, or group of queue managers for an Enterprise License. This Agreement requires that each queue manager of the SOFTWARE be Licensed, either individually, or as part of a group. Each queue manager's use of this SOFTWARE must be covered either individually, or as part of an Enterprise License. The SOFTWARE is in use on a computer when it is loaded into the temporary memory (i.e. RAM) or installed into the permanent memory (e.g. hard disk) of that computer. This software may be installed on a network provided that appropriate restrictions are in place limiting the use to registered queue managers only. Each licensed queue manager will be provided with a perpetual license key and the licensee may continue to use the SOFTWARE, so long as the licensee is current on the Yearly Maintenance Fee. If the licensee stops paying the Yearly Maintenance Fee, then the SOFTWARE must be removed from all systems at the end of the current maintenance period.

2. COPYRIGHT. The SOFTWARE is owned by Capitalware Inc. and is protected by United States Of America and Canada copyright laws and international treaty provisions. You may not copy the printed materials accompanying the SOFTWARE (if any), nor print copies of any user documentation provided in on-line or electronic form. You must not redistribute the registration codes provided, either on paper, electronically, or as stored in the files mqssx.ini, mqssx_licenses.ini or any other form.

3. OTHER RESTRICTIONS. The registration notification provided, showing your authorization code and this License is your proof of license to exercise the rights granted herein and must be retained by you. You may not rent or lease the SOFTWARE, but you may transfer your rights under this License on a permanent basis, provided you transfer this License, the SOFTWARE and all accompanying printed materials, retain no copies, and the recipient agrees to the terms of this License. You may not reverse engineer, decompile, or disassemble the SOFTWARE, except to the extent the foregoing restriction is expressly prohibited by applicable law.

LIMITED WARRANTY

LIMITED WARRANTY. Capitalware Inc. warrants that the SOFTWARE will perform substantially in accordance with the accompanying printed material (if any) and on-line documentation for a period of 365 days from the date of receipt.

CUSTOMER REMEDIES. Capitalware Inc. entire liability and your exclusive remedy shall be, at Capitalware Inc. option, either (a) return of the price paid or (b) repair or replacement of the SOFTWARE that does not meet this Limited Warranty and that is returned to Capitalware Inc.

with a copy of your receipt. This Limited Warranty is void if failure of the SOFTWARE has resulted from accident, abuse, or misapplication. Any replacement SOFTWARE will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer.

NO OTHER WARRANTIES. To the maximum extent permitted by applicable law, Capitalware Inc. disclaims all other warranties, either express or implied, including but not limited to implied warranties of merchantability and fitness for a particular purpose, with respect to the SOFTWARE and any accompanying written materials.

NO LIABILITY FOR CONSEQUENTIAL DAMAGES. To the maximum extent permitted by applicable law, in no event shall Capitalware Inc. be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or other pecuniary loss) arising out of the use or inability to use the SOFTWARE, even if Capitalware Inc. has been advised of the possibility of such damages.

12 Appendix G – Notices

<u>Trademarks:</u>