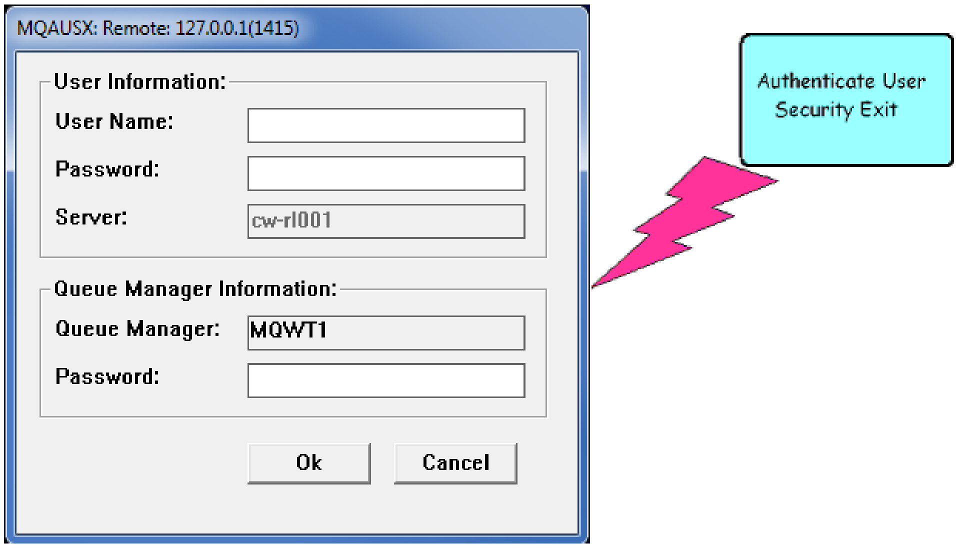
MQAUSX Cluster Configuration Manual


MQAUSX: Remote: 127.0.0.1(1415)

User Information:

User Name:

Password:

Server: cw-rl001

Queue Manager Information:

Queue Manager: MQWT1

Password:

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

1.1 Overview....................1

1.1.1 Client-Side Security Exit....................1

1.1.2 Server-Side Security Exit....................1

2 MQAUSX CLUSTERING OVERVIEW....................4

2.1 Cluster-Sender and Cluster-Receiver Channel Pair....................4

2.2 Channel Auto-Definition....................5

3 MQAUSX CLUSTERING PREREQUISITES....................6

3.1 Prerequisite # 1....................6

3.2 Prerequisite # 2....................6

3.3 Prerequisite # 3....................6

4 IMPLEMENTING MQAUSX WITH CLUSTER CHANNELS....................7

4.1 Stop The Cluster Channels....................7

4.2 Editing the Channel Auto-Definition IniFile....................7

4.2.1 Windows....................7

4.2.2 Unix and Linux 32-bit....................8

4.2.3 Unix and Linux 64-bit....................8

4.2.4 IBM i....................8

4.3 Configuring the CWCHAD Exit....................9

4.3.1 Windows....................9

4.3.2 Unix and Linux 32-bit....................10

4.3.3 Unix and Linux 64-bit....................10

4.3.4 IBM i....................10

4.4 Configuring a Cluster-Receiver Channel....................11

4.4.1 Windows....................11

4.4.2 Unix and Linux 32-bit....................12

4.4.3 Unix and Linux 64-bit....................12

4.4.4 IBM i....................13

4.5 Configuring a Cluster-Sender Channel....................14

5 CWCHAD CHANNEL AUTO-DEFINITION INIFILE....................15

5.1 Channel Auto-Definition IniFile Search Order....................15

5.2 Channel Auto-Definition IniFile Keywords....................15

5.2.1 ConName....................15

5.2.2 Partner....................15

5.2.3 MsgExit and MsgData....................16

5.2.4 RcvExit and RcvData....................16

5.2.5 ScyExit and ScyData....................16

5.2.6 SendExit and SendData....................17

6 CWCHAD INIFILE....................18

6.1 Logging....................18

6.2 Channel Auto-Definition IniFile Path....................18

6.3 SetMessageExit....................19

6.4 SETRECEIVEEXIT....................19
6.5 SETSECURITYEXIT....................19
6.6 SETSENDEXIT....................20

7 APPENDIX A – CHANNEL AUTO-DEFINITION INIFILE SUMMARY....................21

8 APPENDIX B – CWCHAD.INI SUMMARY....................24

9 APPENDIX C – ENCRYPTION....................26

10 APPENDIX D – LICENSE AGREEMENT....................27

11 APPENDIX E – NOTICES....................29

1 Introduction

1.1 Overview

MQ Authenticate User Security Exit (MQAUSX) is solution that allows a company to fully authenticate a user who is accessing a IBM MQ resource. It authenticates the user's UserId and Password (and possibly Domain Name) against the server's native OS system, LDAP server, Microsoft's Active Directory, Quest Authentication Services, Centrify's DirectControl, Unix/Linux PAM (Pluggable Authentication Module) or an encrypted MQAUSX FBA file.

The security exit will operate with IBM MQ v7.0, v7.1, v7.5, v8.0, v9.0, v9.1 and v9.2 in Windows, Unix and Linux environments. It works with Server Connection, Client Connection, Sender, Receiver, Server and Requestor channels of IBM MQ queue manager.

The MQ Authenticate User Security Exit solution is comprised of 2 components: client-side security exit and server-side security exit.

1.1.1 Client-Side Security Exit

The ***client-side security exit*** first checks if the server-side exit is defined for the particular channel. The client-side exit will receive a security token to be used in the encryption process of the user's password. It will prompt the user for his / her UserId and Password (and domain name for Windows), encrypt the data and send it to the server-side security exit.

For each connection attempt, the server-side security exit will verify that it is an acceptable client exit attempting the connection. If so, then the server-side will send a unique security token. When the server-side security exit receives the encrypted data, it will decrypt the incoming data and then perform UserId and Password (and domain) authentication against the native OS system, LDAP server, Microsoft's Active Directory, Quest Authentication Services, Centrify's DirectControl, Unix/Linux PAM (Pluggable Authentication Module) or an encrypted MQAUSX FBA file. If successful, the connection will be allowed.

1.1.2 Server-Side Security Exit

The ***server-side security exit*** supports the concept of 'Proxy IDs'. After a user has been successfully authenticated against the native OS system, LDAP server, Microsoft's Active Directory, Quest Authentication Services, Centrify's DirectControl, Unix/Linux PAM (Pluggable Authentication Module) or an encrypted MQAUSX FBA file and the 'Proxy Mode' flag is set, then the server-side 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.

An MQAdmin can define a password for a queue manager. Hence, when enabled, a back-end application and/or end-user would need to not only know their UserID and Password but also the queue manager’s Password to successfully log in. Defining and requiring a queue manager Password in MQAUSX is equivalent to adding perimeter security to your system.

The server-side security exit has the ability to allow or restrict 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, hostname, 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. If MQAUSX is authenticating against an LDAP server then the group querying can be against the LDAP server.

For those channels where authentication is not required, the server-side security exit can be set to not perform this function. This is controlled by the server-side security exit's property keyword 'NoAuth'.

The server-side security exit, when in non-authentication mode, 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, when in non-authentication mode, 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.

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.

MQAUSX is 4 products in 1

1. If the client application is configured with the client-side security exit then the user credentials are encrypted and sent to the remote queue manager. This is the best level of security.

2. If the client application is not configured with the client-side security exit and the client-side **AND** server-side are at MQ V8 then MQ V8 will encrypt the user credentials as they flow from the client application to the queue manager.

3. If the client application is not configured with the client-side security exit then the user credentials are sent in plain text to the remote queue manager. This feature is available for Java/JMS, Java and C# DotNet client applications. For native applications (i.e. C/C++), then the application must use and populate the MQCSP structure with the UserID and Password.
    - Using MQAUSX with No Client-side Security Exit - Part 1 (coding examples) http://www.capitalware.com/rl_blog/?p=638
    - Using MQAUSX with No Client-side Security Exit - Part 2 (configuring tools like MQ Explorer, SupportPac MO71, MQ Visual Edit, etc..) http://www.capitalware.com/rl_blog/?p=659

4. If the MQAdmin sets the MQAUSX IniFile parameter NoAuth to Y then it functions just like MQ Standard Security Exit (MQSSX). MQSSX does not authenticate. It filters the incoming connection based on UserID, IP address, hostname and/or SSL DN.

2 MQAUSX Clustering Overview

This section provides an overview of how MQAUSX can authenticate the UserId and Password of the connection request from one queue manager to any queue manager participating in the cluster.

2.1 Cluster-Sender and Cluster-Receiver Channel Pair

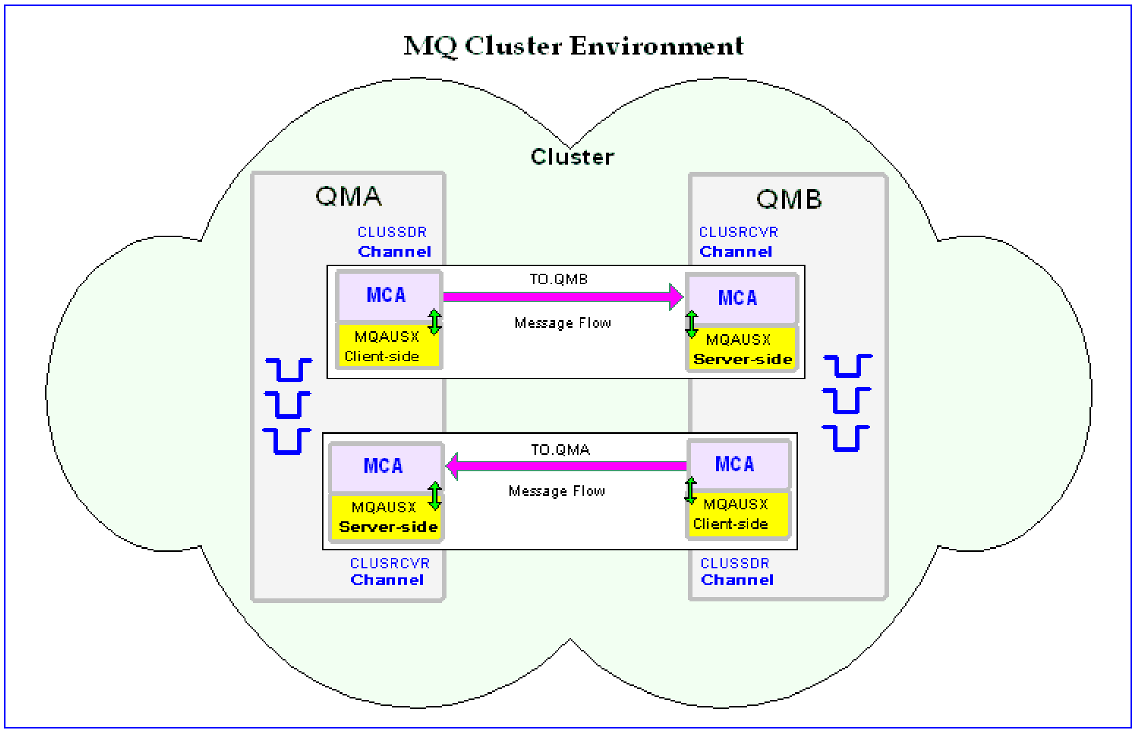
As mentioned in Chapter 1, MQAUSX is comprised of 2 MQ security exits: client-side security exit and server-side security exit. As noted below (in yellow) in the diagram, the MQAUSX client-side security exit works with the Cluster-Sender (CLUSSDR) channel and the MQAUSX server-side security exit works with the Cluster-Receiver (CLUSRCVR) channel.

There is a Message Channel Agent (MCA) at each end of the channel. The MCA is a component that handles the sending and receiving of messages between queue managers. Before the MCA can send and receive messages, the UserId and Password must be authenticated as detailed below:

- ➢ The MCA that is running the Cluster-Sender channel will call MQAUSX client-side security exit to send a security message that contains the UserId and encrypted Password across the channel to the Cluster-Receiver channel.
- ➢ The MCA that is running the Cluster-Receiver channel will call MQAUSX server-side security exit to authenticate the incoming UserId and encrypted Password.

After the UserId and Password has been successfully authenticated, the channel will go to a 'Running' state and the messages will flow along the channel.

The following diagram highlights security exits in a clustering environment:



2.2 Channel Auto-Definition

When a queue manager has joined a cluster, MCA creates, by auto-definition, the Cluster-Sender channel from the attributes of the Cluster-Receiver channel. If an MQAdmin manually defines a Cluster-Sender channel, the queue manager will automatically modify those values to match the Cluster-Receiver channel's attributes. In other words, the MQAdmin cannot modify an auto-created Cluster-Sender channel via MQSC commands.

To explicitly set the Cluster-Sender channel attributes, a Channel Auto-Definition exit must be used. The MQAUSX installation package includes a Channel Auto-Definition exit called ***CWCHAD***.

The CWCHAD exit has been designed to override 9 attributes of a Cluster-Sender channel as listed below:

- CONNAME - The connection information for the channel
- MSGEXIT - The message exit name for the channel
- MSGDATA - The message exit data for the channel
- RCVEXIT - The receive exit name for the channel
- RCVDATA - The receive exit data for the channel
- SCYEXIT - The security exit name for the channel
- SCYDATA - The security exit data for the channel
- SENDEXIT - The send exit name for the channel
- SENDDATA - The send exit data for the channel

The CWCHAD exit does not only work with auto-created Cluster-Sender channels but as an added bonus, it also works with these other channels: Cluster-Receiver, Receiver and Server-Connection channels.

3 MQAUSX Clustering Prerequisites

This section details the necessary steps that must be completed ***BEFORE*** implementing MQAUSX and the CWCHAD exit in a MQ clustering environment. The implementation of security exits in an MQ clustering environment is extremely complex and must be completed very carefully.

3.1 Prerequisite # 1

The MQAdmin must read and understand Chapters 1 through 4 of IBM's ***IBM MQ Queue Manager Clusters*** manual. The IBM IBM MQ manuals can be found at the following link: http://www.ibm.com/software/integration/wmq/library/

3.2 Prerequisite # 2

The MQAdmin must make sure that all of the cluster channels have been successfully started. i.e. all cluster channels must have a channel status of 'Running'.

3.3 Prerequisite # 3

MQAdmin must stop and restart all Cluster-Sender channels (since the Channel Auto-Definition does not kick-in until they have been restarted) at least twice (yes, twice). They must have a channel status of 'Running'. ***This step is absolutely crucial.***

4 Implementing MQAUSX with Cluster Channels

This section describes the necessary steps to enable MQAUSX and the CWCHAD exit with cluster channels. All steps must be followed exactly. i.e. no skipping of steps.

4.1 Stop The Cluster Channels

The MQAdmin must stop ***ALL*** CLUSSDR and CLUSRCVR channels (to which you are applying the MQAUSX security exits to). ***The channels must be in the 'Stopped' state before continuing any further.***

4.2 Editing the Channel Auto-Definition IniFile

This section describes the necessary entries to enable the CWCHAD exit. Please review Chapter 5 for more information on editing the Channel Auto-Definition IniFile.

Notes:

- The MQAdmin can simply use the **[default]** section if all of the cluster channels will be authenticated with the same UserId and Password. Otherwise, the MQAdmin will need to create a new section for each CLUSSDR channel in the Channel Auto-Definition IniFile.
- If the MQAdmin did not perform a default install of MQAUSX then they will need to update the value of ScyExit to reflect the new path information.
- If the MQAdmin wishes to have a different Path and/or filename for the Client IniFile then they will need to update the value of ScyData to reflect the new path information.

4.2.1 Windows

For Windows, a sample Channel Auto-Definition IniFile with the name: ***cwchad.clussdr.ini*** has been included with the MQAUSX installation package.

The following is the contents of the ***cwchad.clussdr.ini*** file:

```
[TO.QMGRNAME]
ScyExit=C:\Capitalware\MQAUSX\mqausxclnt(ClntExit)
ScyData=C:\Capitalware\MQAUSX\clnt.ini
[default]
ScyExit=C:\Capitalware\MQAUSX\mqausxclnt(ClntExit)
ScyData=C:\Capitalware\MQAUSX\clnt.ini
```

4.2.2 Unix and Linux 32-bit

For Unix and Linux, a sample Channel Auto-Definition IniFile with the name: ***cwchad.clussdr.ini*** has been included with the MQAUSX installation package.

The following is the contents of the ***cwchad.clussdr.ini*** file:

```
[TO.QMGRNAME]
ScyExit=/var/mqm/exits/mqausxclnt(ClntExit)
ScyData=/var/mqm/exits/clnt.ini
[default]
ScyExit=/var/mqm/exits/mqausxclnt(ClntExit)
ScyData=/var/mqm/exits/clnt.ini
```

4.2.3 Unix and Linux 64-bit

For Unix and Linux (excluding Linux x86), a sample Channel Auto-Definition IniFile with the name: ***cwchad.clussdr.ini*** has been included with the MQAUSX installation package.

The following is the contents of the ***cwchad.clussdr.ini*** file:

```
[TO.QMGRNAME]
ScyExit=/var/mqm/exits64/mqausxclnt(ClntExit)
ScyData=/var/mqm/exits64/clnt.ini
[default]
ScyExit=/var/mqm/exits64/mqausxclnt(ClntExit)
ScyData=/var/mqm/exits64/clnt.ini
```

4.2.4 IBM i

For IBM i, a sample Channel Auto-Definition IniFile with the name: ***cwchad.clussdr.ini*** has been included with the MQAUSX installation package.

The following is the contents of the ***cwchad.clussdr.ini*** file:

```
[TO.QMGRNAME]
ScyExit=MQAUSXCL    MQAUSX
ScyData=/QIBM/UserData/mqm/mqausx/clnt.ini
[default]
ScyExit=MQAUSXCL    MQAUSX
ScyData=/QIBM/UserData/mqm/mqausx/clnt.ini
```

4.3 Configuring the CWCHAD Exit

This section describes how to configure the CWCHAD exit for a queue manager.

Note: Do NOT alter the QMGR CHAD parameter, as it is not required for clustering.

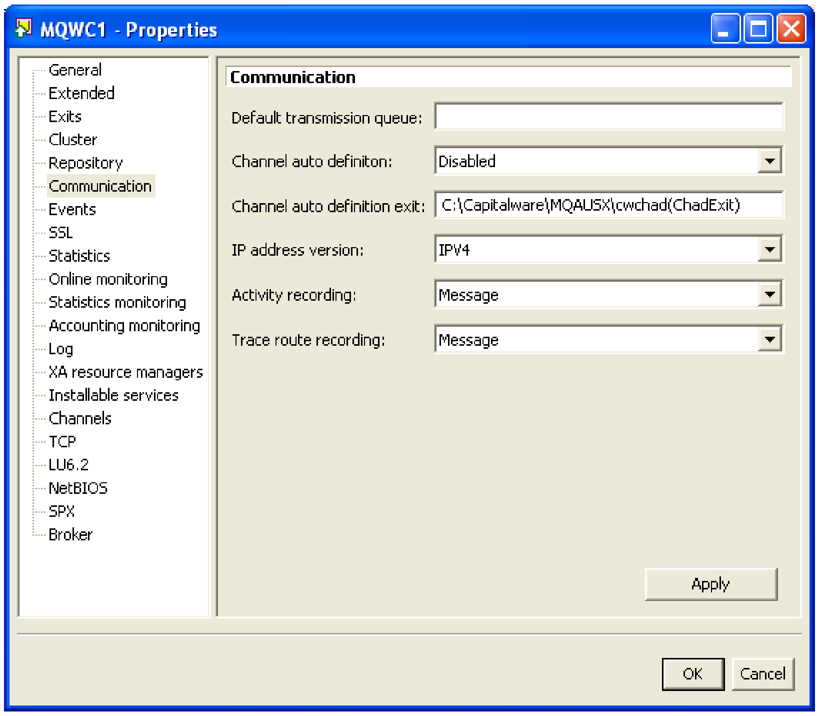
4.3.1 Windows

For Windows, CHADEXIT will contain the following values, assuming a default install:

- CHADEXIT
  `C:\Capitalware\MQAUSX\cwchad(ChadExit)`

The following is an example of an MQSC command:

```
ALTER QMGR CHADEXIT('C:\Capitalware\MQAUSX\cwchad(ChadExit)')
```



Communication

Default transmission queue:

Channel auto definiton: Disabled

Channel auto definition exit: C:\Capitalware\MQAUSX\cwchad(ChadExit)

IP address version: IPV4

Activity recording: Message

Trace route recording: Message

4.3.2 Unix and Linux 32-bit

For Unix and Linux, CHADEXIT will contain the following values, assuming a default install:

- CHADEXIT
  `/var/mqm/exits/cwchad(ChadExit)`

The following is an example of an MQSC command:

```
ALTER QMGR CHADEXIT('/var/mqm/exits/cwchad(ChadExit)')
```

4.3.3 Unix and Linux 64-bit

For Unix and Linux (excluding Linux x86), CHADEXIT will contain the following values, assuming a default install:

- CHADEXIT
  `/var/mqm/exits64/cwchad(ChadExit)`

The following is an example of an MQSC command:

```
ALTER QMGR CHADEXIT('/var/mqm/exits64/cwchad(ChadExit)')
```

4.3.4 IBM i

For IBM i, CHADEXIT will contain the following values, assuming a default install:

- CHADEXIT
  `CWCHAD    MQAUSX`

The following is an example of an MQSC command:

```
ALTER QMGR CHADEXIT('CWCHAD    MQAUSX    ')
```

4.4 Configuring a Cluster-Receiver Channel

This section describes the necessary entries to enable the server-side security exit on a Cluster-Receiver Channel. The server-side security exit and its data will be applied to 2 fields of the Cluster-Receiver Channel. The MQ Administrator will need to update these 2 fields of the Cluster-Receiver Channel.

For more information on server-side IniFile parameters, please review *Appendix A* of the ***MQAUSX Server-side Installation and Operation*** manual.

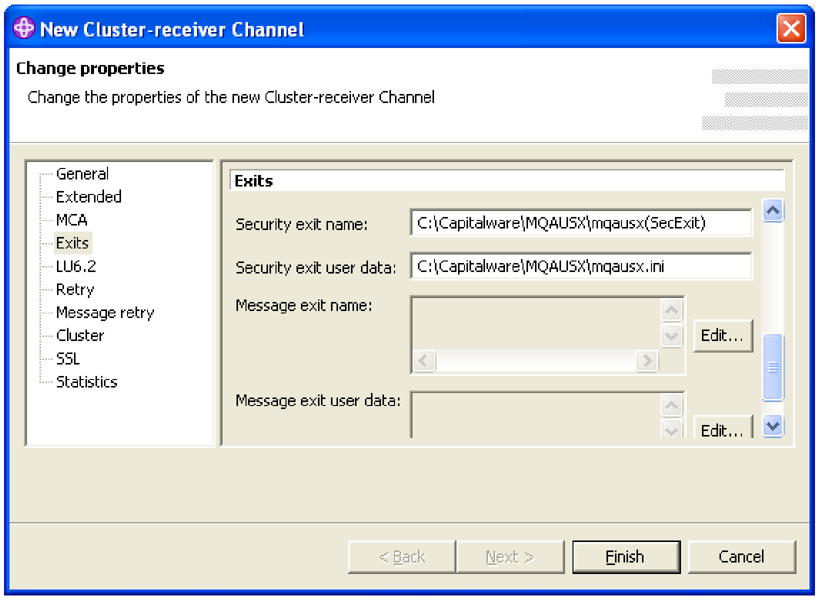
4.4.1 Windows

For Windows, SCYEXIT and SCYDATA will contain the following values, assuming a default install:

- SCYEXIT
  `C:\Capitalware\MQAUSX\mqausx(SecExit)`
- SCYDATA
  `C:\Capitalware\MQAUSX\mqausx.ini`

The following is an example of an MQSC command for creating a Cluster-Receiver Channel with the server-side security exit and its data:

```
DEFINE CHANNEL ('TO.QMA') CHLTYPE(CLUSRCVR) +
       TRPTYPE(TCP) +
       CONNAME(127.0.0.1(1414) +
       SCYEXIT('C:\Capitalware\MQAUSX\mqausx(SecExit)') +
       SCYDATA('C:\Capitalware\MQAUSX\mqausx.ini') +
       REPLACE
```



Change properties
Change the properties of the new Cluster-receiver Channel

Exits
Security exit name: C:\Capitalware\MQAUSX\mqausx(SecExit)
Security exit user data: C:\Capitalware\MQAUSX\mqausx.ini
Message exit name:
Message exit user data:

4.4.2 Unix and Linux 32-bit

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

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

The following is an example of an MQSC command for creating a Cluster-Receiver Channel with the server-side security exit and its data:

```
DEFINE CHANNEL ('TO.QMA') CHLTYPE(RECEIVER) +
       TRPTYPE(TCP) +
       CONNAME(127.0.0.1(1414) +
       SCYEXIT('/var/mqm/exits/mqausx(SecExit)') +
       SCYDATA('/var/mqm/exits/mqausx.ini') +
       REPLACE
```

4.4.3 Unix and Linux 64-bit

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

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

The following is an example of an MQSC command for creating a Cluster-Receiver Channel with the server-side security exit and its data:

```
DEFINE CHANNEL ('TO.QMA') CHLTYPE(RECEIVER) +
       TRPTYPE(TCP) +
       CONNAME(127.0.0.1(1414) +
       SCYEXIT('/var/mqm/exits64/mqausx(SecExit)') +
       SCYDATA('/var/mqm/exits64/mqausx.ini') +
       REPLACE
```

4.4.4 IBM i

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

- SCYEXIT
  `MQAUSX    MQAUSX`
- SCYDATA
  `mqausx.ini`

The following is an example of an MQSC command for creating a Cluster-Receiver Channel with the server-side security exit and its data:

```
DEFINE CHANNEL ('TO.QMA') CHLTYPE(RECEIVER) +
       TRPTYPE(TCP) +
       CONNAME(127.0.0.1(1414) +
       SCYEXIT('MQAUSX    MQAUSX    ') +
       SCYDATA('mqausx.ini') +
       REPLACE
```

4.5 Configuring a Cluster-Sender Channel

There is no MQAUSX configuration required for the Cluster-Sender channel, as the CWCHAD exit will handle the setting of the values of the SCYEXIT and SCYDATA fields.

5 CWCHAD Channel Auto-Definition IniFile

This section describes how to configure the CWCHAD Channel Auto-Definition IniFile.

5.1 Channel Auto-Definition IniFile Search Order

The CWCHAD exit supports individual or shared configuration of the ***.clussdr.ini*** file for multiple queue managers on the same server. The CWCHAD exit first searches for a Channel Auto-Definition IniFile with the prefix of the queue manager name. If successful, it will use that IniFile. Otherwise, the CWCHAD exit uses the default Channel Auto-Definition IniFile with the name: ***cwchad.clussdr.ini***.

e.g. If the queue manager name is QM1, the CWCHAD exit searches for the ***qm1.clusdr.ini*** file. If that IniFile is not found, the CWCHAD exit uses the Channel Auto-Definition IniFile with the name: ***cwchad.clussdr.ini***.

5.2 Channel Auto-Definition IniFile Keywords

IniFile keywords are grouped together in sections. A section name is the actual CLUSSDR channel name. A section name is surrounded by square brackets ('[' and ']').

```
[SECTION-NAME]
ScyExit=keyword-value
ScyData= keyword-value
```

To specify default values for any Channel Auto-Definition IniFile keyword, use the default section. The default section is optional.

```
[default]
ScyExit=C:\Capitalware\MQAUSX\mqausxclnt(ClntExit)
ScyData=C:\Capitalware\MQAUSX\clnt.ini
```

The IniFile supports the following keywords and their respective values:

5.2.1 ConName

The ConName keyword specifies a value to override the current CONNAME field in the CLUSSDR channel definition. ConName is optional. If the keyword is not specified or its value is blank then no override is performed.

5.2.2 Partner

The Partner keyword specifies a value to be verified against the incoming connection request's Partner name. Partner is optional. If the keyword is not specified or its value is blank then no check is performed.

5.2.3 MsgExit and MsgData

The MsgExit and MsgData keywords are only used if SetMessageExit is set to 'Y' in the cwchad.ini file.

5.2.3.1 MsgExit

The MsgExit keyword specifies a value in order to override the current MSGEXIT field in the CLUSSDR channel definition. MsgExit is optional. If the keyword is not specified or its value is blank then no override is performed.

5.2.3.2 MsgData

The MsgData keyword specifies a value in order to override the current MSGDATA field in the CLUSSDR channel definition. MsgData is optional. If the keyword is not specified or its value is blank then no override is performed.

Below is a sample Channel Auto-Definition IniFiles using MsgExit and MsgData entries:

```
[TO.QMGRNAME]
MsgExit=C:\temp\MyMsgExit(ME)
MsgData= C:\temp\MyData
```

5.2.4 RcvExit and RcvData

The RcvExit and RcvData keywords are only used if SetReceiveExit is set to 'Y' in the cwchad.ini file.

5.2.4.1 RcvExit Keyword

The RcvExit keyword specifies a value in order to override the current RCVEXIT field in the CLUSSDR channel definition. RcvExit is optional. If the keyword is not specified or its value is blank then no override is performed.

5.2.4.2 RcvData Keyword

The RcvData keyword specifies a value in order to override the current RCVDATA field in the CLUSSDR channel definition. RcvData is optional. If the keyword is not specified or its value is blank then no override is performed.

Below is a sample Channel Auto-Definition IniFiles using RcvExit and RcvData entries:

```
[TO.QMGRNAME]
RcvExit=C:\temp\MyRcvExit(RE)
RcvData= C:\temp\MyData
```

5.2.5 ScyExit and ScyData

The ScyExit and ScyData keywords are only used if SetSecurityExit is set to 'Y' in the cwchad.ini file.

5.2.5.1 ScyExit
The ScyExit keyword specifies a value in order to override the current SCYEXIT field in the CLUSSDR channel definition. ScyExit is optional. If the keyword is not specified or its value is blank then no override is performed.

5.2.5.2 ScyData
ScyData The ScyData keyword specifies a value in order to override the current SCYDATA field in the CLUSSDR channel definition. ScyData is optional. If the keyword is not specified or its value is blank then no override is performed.

Below is a sample Channel Auto-Definition IniFiles using ScyExit and ScyData entries:

```
[TO.QMGRNAME]
ScyExit=C:\Capitalware\MQAUSX\mqausxclnt(ClntExit)
ScyData=C:\Capitalware\MQAUSX\clnt.ini
```

5.2.6 SendExit and SendData
The SendExit and SendData keywords are only used if SetSendExit is set to 'Y' in the cwchad.ini file.

5.2.6.1 SendExit
The SendExit keyword specifies a value in order to override the current SENDEXIT field in the CLUSSDR channel definition. SendExit is optional. If the keyword is not specified or its value is blank then no override is performed.

5.2.6.2 SendData
The SendData keyword specifies a value in order to override the current SENDDATA field in the CLUSSDR channel definition. SendData is optional. If the keyword is not specified or its value is blank then no override is performed.

Below is a sample Channel Auto-Definition IniFiles using SendExit and SendData entries:

```
[TO.QMGRNAME]
SendExit=C:\temp\MySendExit(SE)
SendData= C:\temp\MyData
```

6 CWCHAD IniFile

This section describes how to configure the CWCHAD IniFile (cwchad.ini). The ***cwchad.ini*** file is optional.

6.1 Logging

This section describes the necessary entries to enable CWCHAD to record log information. To enable and control logging, you need 2 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 values are as follows:

   For Windows:
   ```
   LogFile=C:\Capitalware\MQAUSX\cwchad.log
   ```

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

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

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

```
LogMode=N
LogFile=C:\Capitalware\MQAUSX\cwchad.log
```

6.2 Channel Auto-Definition IniFile Path

This section describes the necessary steps to allow the MQAdmin to define a different path to the Channel Auto-Definition IniFile. By default, the path will be the CWCHAD install directory.

To specify a different Channel Auto-Definition IniFile Path, you need 2 keywords in the IniFile:

- **UseChadIniFilePath** set to Y if you are defining a different path to the Channel Auto-Definition IniFile
- **ChadIniFilePath** specifies the path (location) rather than the default path of the Channel Auto-Definition IniFile

```
UseChadIniFilePath=Y
ChadIniFilePath=c:\temp\chad\
```

6.3 SetMessageExit

This section describes the necessary steps to enable the overriding of the channel's MSGEXIT and MSGDATA fields. SetMessageExit specifies whether or not the CWCHAD exit will override the MSGEXIT and MSGDATA fields.

SetMessageExit can have a value of either Y or N. If you want the CWCHAD exit to override the MSGEXIT and MSGDATA fields, set the SetMessageExit value to Y as shown below.

```
SetMessageExit=Y
```

6.4 SetReceiveExit

This section describes the necessary steps to enable the overriding of the channel's RCVEXIT and RCVDATA fields. SetReceiveExit specifies whether or not the CWCHAD exit will override the RCVEXIT and RCVDATA fields.

SetReceiveExit can have a value of either Y or N. If you want the CWCHAD exit to override the RCVEXIT and RCVDATA fields, set the SetReceiveExit value to Y as shown below.

```
SetReceiveExit=Y
```

6.5 SetSecurityExit

This section describes the necessary steps to enable the overriding of the channel's SCYEXIT and SCYDATA fields. SetSecurityExit specifies whether or not the CWCHAD exit will override the SCYEXIT and SCYDATA fields.

SetSecurityExit can have a value of either Y or N. If you want the CWCHAD exit to override the SCYEXIT and SCYDATA fields, set the SetSecurityExit value to Y as shown below.

```
SetSecurityExit=Y
```

6.6 SetSendExit

This section describes the necessary steps to enable the overriding of the channel's SENDEXIT and SENDDATA fields. SetSendExit specifies whether or not the CWCHAD exit will override the SENDEXIT and SENDDATA fields.

SetSendExit can have a value of either Y or N. If you want the CWCHAD exit to override the SENDEXIT and SENDDATA fields, set the SetSendExit value to Y as shown below.

```
SetSendExit=Y
```

7 Appendix A – Channel Auto-Definition IniFile Summary

A sample Channel Auto-Definition IniFile below is the ***cwchad.clussdr.ini*** file supplied for Windows.

```
[TO.QMGRNAME]
ScyExit=C:\Capitalware\MQAUSX\mqausxclnt(ClntExit)
ScyData=C:\Capitalware\MQAUSX\clnt.ini
[default]
ScyExit=C:\Capitalware\MQAUSX\mqausxclnt(ClntExit)
ScyData=C:\Capitalware\MQAUSX\clnt.ini
```

IniFile keywords are grouped together in sections. A section name is the actual CLUSSDR channel name. A section name is surrounded by square brackets ('[' and ']'). To specify default values for any Channel Auto-Definition IniFile keyword, use the default section. The default section is optional. The IniFile supports the following keywords and their respective values:

Note: Keywords are case sensitive.

Keyword	Description of keywords
ConName	**ConName** specifies a value in order to override the current CONNAME field in the CLUSSDR channel definition. ConName is optional. If the keyword is not specified or its value is blank then no override is performed. e.g. ConName=127.0.0.1(1415)
Partner	**Partner** specifies a value to be verified against the incoming connection request's Partner name. Partner is optional. If the keyword is not specified or its value is blank then no check is performed. e.g. Partner=QM5
ScyData	**ScyData** specifies a value in order to override the current SCYDATA field in the CLUSSDR channel definition. ScyData is optional. If the keyword is not specified or its value is blank then no override is performed. For. Windows: ScyExit=C:\Capitalware\MQAUSX\clnt.ini For IBM MQ 32-bit on Unix and Linux: ScyExit=/var/mqm/exits/clnt.ini For IBM MQ 64-bit on Unix and Linux: ScyExit=/var/mqm/exits64/clnt.ini For IBM MQ on IBM i: ScyExit=/QIBM/UserData/mqm/mqausx/clnt.ini

Keyword	Description of keywords
ScyExit	**ScyExit** specifies a value in order to override the current SCYEXIT field in the CLUSSDR channel definition. ScyExit is optional. If the keyword is not specified or its value is blank then no override is performed. For. Windows: ScyExit=C:\Capitalware\MQAUSX\mqausxclnt(ClntExit) For IBM MQ 32-bit on Unix and Linux: ScyExit=/var/mqm/exits/mqausxclnt(ClntExit) For IBM MQ 64-bit on Unix and Linux: ScyExit=/var/mqm/exits64/mqausxclnt(ClntExit) For IBM MQ on IBM i: ScyExit=MQAUSXCL MQAUSX
MsgData	**MsgData** specifies a value in order to override the current MSGDATA field in the CLUSSDR channel definition. MsgData is optional. If the keyword is not specified or its value is blank then no override is performed. e.g. MsgData =SampleMessageData Note: Only used if SetMessageExit is set to 'Y' in the ***cwchad.ini*** file.
MsgExit	**MsgExit** specifies a value in order to override the current MSGEXIT field in the CLUSSDR channel definition. MsgExit is optional. If the keyword is not specified or its value is blank then no override is performed. e.g. MsgExit=SampleExit Note: Only used if SetMessageExit is set to 'Y' in the ***cwchad.ini*** file.
RcvData	**RcvData** specifies a value in order to override the current RCVDATA field in the CLUSSDR channel definition. RcvData is optional. If the keyword is not specified or its value is blank then no override is performed. e.g. RcvData=SampleMessageData Note: Only used if SetReceiveExit is set to 'Y' in the ***cwchad.ini*** file.

Keyword	Description of keywords
RcvExit	**RcvExit** specifies a value in order to override the current RCVEXIT field in the CLUSSDR channel definition. RcvExit is optional. If the keyword is not specified or its value is blank then no override is performed. e.g. RcvExit =SampleExit Note: Only used if SetReceiveExit is set to 'Y' in the ***cwchad.ini*** file.
SendData	**SendData** specifies a value in order to override the current SENDDATA field in the CLUSSDR channel definition. SendDatais optional. If the keyword is not specified or its value is blank then no override is performed. e.g. SendData=SampleMessageData Note: Only used if SetSendExit is set to 'Y' in the ***cwchad.ini*** file.
SendExit	**SendExit** specifies a value in order to override the current SENDEXIT field in the CLUSSDR channel definition. SendExit is optional. If the keyword is not specified or its value is blank then no override is performed. e.g. SendExit =SampleExit Note: Only used if SetSendExit is set to 'Y' in the ***cwchad.ini*** file.

8 Appendix B – cwchad.ini Summary

The cwchad.ini file is optional. The sample CWCHAD IniFile below is the Windows cwchad.ini file.

```
LogMode=Q
LogFile=C:\Capitalware\MQAUSX\cwchad.log
ChadIniFilePath=C:\Capitalware\MQAUSX\
```

The CWCHAD IniFile supports the following keywords and their respective values:

Note: Keywords are case sensitive.

Keyword	Description of keywords
ChadIniFilePath	**ChadIniFilePath** specifies the location of the Channel Auto-Definition IniFile. The default is as follows: For Windows: ChadIniFilePath=C:\Capitalware\MQAUSX\ For IBM MQ 32-bit on Unix and Linux: ChadIniFilePath=/var/mqm/exits/ For IBM MQ 64-bit on Unix and Linux: ChadIniFilePath=/var/mqm/exits64/ For IBM MQ on IBM i: ChadIniFilePath=/QIBM/UserData/mqm/mqausx/ Note: Only used if UseChadIniFilePath is set to 'Y'.
SetMessageExit	**SetMessageExit** specifies whether or not the CWCHAD exit will override the MSGEXIT and MSGDATA fields. SetMessageExit supports 2 values [Y / N] where the default value is N. e.g. SetMessageExit=Y
SetReceiveExit	**SetReceiveExit** specifies whether or not the CWCHAD exit will override the RCVEXIT and RCVDATA fields. SetMessageExit supports 2 values [Y / N] where the default value is N. e.g. SetReceiveExit=Y
SetSecurityExit	**SetSecurityExit** specifies whether or not the CWCHAD exit will override the SCYEXIT and SCYDATA fields. SetMessageExit supports 2 values [Y / N] where the default value is Y.

Keyword	Description of keywords
	e.g. SetSecurityExit=Y
SetSendExit	**SetSendExit** specifies whether or not the CWCHAD exit will override the SENDEXIT and SENDDATA fields. SetMessageExit supports 2 values [Y / N] where the default value is N. e.g. SetSendExit=Y
LogFile	**LogFile** specifies the location of the log file. The default is as follows: For Windows: LogFile=C:\Capitalware\MQAUSX\cwchad.log For IBM MQ 32-bit on Unix and Linux: LogFile=/var/mqm/exits/cwchad.log For IBM MQ 64-bit on Unix and Linux: LogFile=/var/mqm/exits64/cwchad.log
LogMode	**LogMode** specifies what type of logging the user wishes to have. LogMode supports 3 values [Q / N / V] where Q is Quiet, N is Normal and V is Verbose where the default value is N. e.g. LogMode=N
UseChadIniFilePath	**UseChadIniFilePath** specifies a user supplied Path to Channel Auto-Definition IniFile will be set via the ChadIniFilePath keyword. UseChadIniFilePath supports 2 values [Y / N] where the default value is N. e.g. UseChadIniFilePath=Y

9 Appendix C – Encryption

MQ Authenticate User Security Exit Solution uses the Advanced Encryption Standard (AES) for encryption and decryption of the user's password between the client-side security exit and the server-side security exit.

Wikipedia

> *the Advanced Encryption Standard (AES) is an encryption standard adopted by the U.S. government. The standard comprises three block ciphers, AES-128, AES-192 and AES-256, adopted from a larger collection originally published as Rijndael. Each AES cipher has a 128-bit block size, with key sizes of 128, 192 and 256 bits, respectively. The AES ciphers have been analyzed extensively and are now used worldwide, as was the case with its predecessor,[3] the Data Encryption Standard (DES).*
>
> *AES was announced by National Institute of Standards and Technology (NIST) as U.S. FIPS PUB 197 (FIPS 197) on November 26, 2001 after a 5-year standardization process in which fifteen competing designs were presented and evaluated before Rijndael was selected as the most suitable (see Advanced Encryption Standard process for more details). It became effective as a Federal government standard on May 26, 2002 after approval by the Secretary of Commerce. It is available in many different encryption packages. AES is the first publicly accessible and open cipher approved by the NSA for top secret information*

10 Appendix D – 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 user, or group of users for Muliple User Licenses and Site Licenses. This Agreement requires that each user of the SOFTWARE be Licensed, either individually, or as part of a group. A Multi-User License provides for a specified number of users to use this SOFTWARE at any time. This does not provide for concurrent user Licensing. Each user of this SOFTWARE must be covered either individually, or as part of a group Multi-User 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 users only.

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 mqausx.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.

11 Appendix E – Notices

<u>Trademarks:</u>