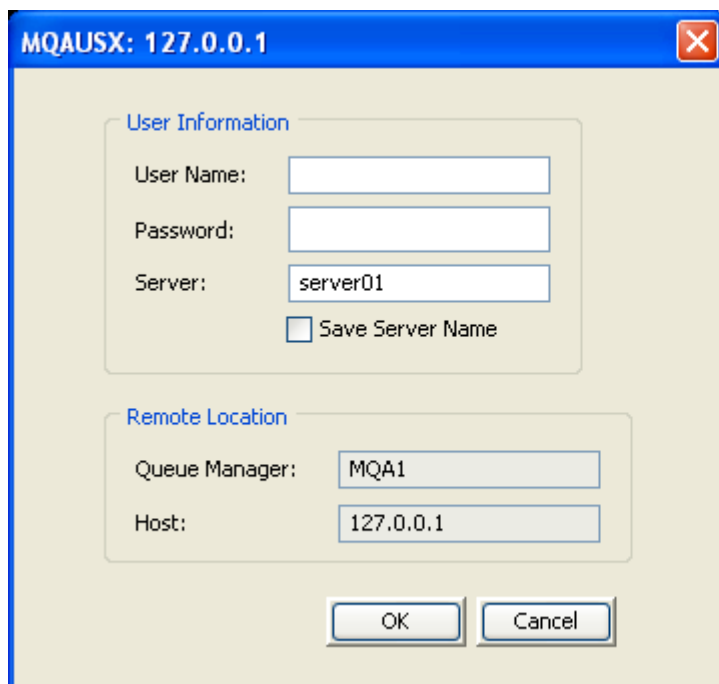


MQ Authenticate User Security Exit for z/OS Overview



A screenshot of a Windows-style dialog box titled "MQAUSX: 127.0.0.1". The dialog is divided into two sections: "User Information" and "Remote Location".

User Information:

- User Name: []
- Password: []
- Server: [server01]
- Save Server Name

Remote Location:

- Queue Manager: [MQA1]
- Host: [127.0.0.1]

Buttons: [OK] [Cancel]

Authenticate User
Security Exit



Capitalware Inc.
1673 Richmond Street, Suite 524
London, Ontario N6G2N3
Canada
sales@capitalware.biz
<http://www.capitalware.biz>

Table of Contents

1 INTRODUCTION.....	1
1.1 OVERVIEW.....	1
1.1.1 <i>Client-Side Security Exit</i>	1
1.1.2 <i>Server-Side Security Exit</i>	1
1.2 EXECUTIVE SUMMARY.....	3
1.2.1 <i>Server-Side Security Exit</i>	3
1.2.2 <i>Client-Side Security Exit</i>	3
1.3 CONTEXT DIAGRAM (LOGICAL VIEW).....	4
1.4 SECURITY MESSAGE FLOW (LOGICAL VIEW).....	4
1.5 PREREQUISITES.....	5
1.5.1 <i>Operating System</i>	5
1.5.2 <i>WebSphere MQ</i>	5
2 CLIENT-SIDE SECURITY EXIT TESTED APPLICATIONS.....	6
3 APPENDIX C – ENCRYPTION.....	7
3.1 TEA ENCRYPTION ALGORITHM.....	7

1 Introduction

1.1 Overview

MQ Authenticate User Security Exit for z/OS (z/MQAUSX) is a new solution that allows a company to fully authenticate a user who is accessing a WebSphere MQ resource. It verifies the User's UserId and Password against the server's native z/OS system.

The security exit will operate with WebSphere MQ v5.3.1, v6.0 and v7.0 in z/OS v1.4 or higher environments. It works with Server Connection, Client Connection, Sender, Receiver, Server, Requestor, Cluster-Sender and Cluster-Receiver channels of WebSphere MQ queue manager.

The MQ Authenticate User Security Exit for z/OS 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 128-bit 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 128-bit 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) verification against the native OS (or file - optional). If successful, the connection will be allowed.

If the company or MQ Administrator chooses not to use native OS UserId and Password checking, he or she can set up the server-side security exit to use a file for UserId and Password checking. The file is a plain text file where each row will contain 2 columns: UserId and Password. Any standard text editor can be used to modify the file.

1.1.2 Server-Side Security Exit

The *server-side security exit* supports the concept of 'Proxy IDs'. After a user has been successfully validated against the native z/OS or file based validation data 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.

The server-side security exit has the ability to allow or restrict users from logging in with the 'CHIN' or the CHIN's Started-task 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 server-side security exit has the ability to allow or restrict the incoming IP address. The server-side security exit uses a regular expression parser to parse the incoming client IP address against a predefined regular expression pattern.

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.

1.2 Executive Summary

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

1.2.1 Server-Side Security Exit

The server-side security exit is available in:

- z/OS load-module

The major features of the server-side security exit are as follows:

- Authenticate a user against the server's native z/OS or a z/MQAUSX file.
- Provides support for Proxy UserIDs
- Allows or restricts the incoming IP address against a regular expression pattern
- Limit the number of incoming channel connections on a SVRCONN channel.
- Allows or restricts the use of 'CHIN' or the CHIN's Started-task UserIds
- Ability to turn off server-side authentication
- Allows or restricts the incoming UserID against a regular expression pattern when authentication is off
- Provides logging capability for all connecting client applications regardless if they were successful or not.
- Provides logging capability via Write To Operator (WTO) facility.

1.2.2 Client-Side Security Exit

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

- Windows DLL,
- Java JAR and
- Non-GUI shared library for AIX, HP-UX, Linux, and Solaris.

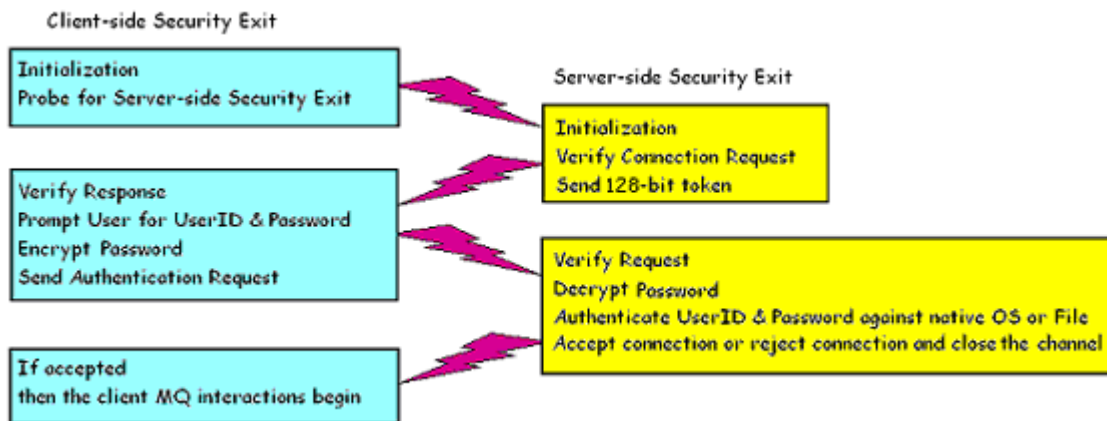
The client-side security exit has been tested against the following MQ client programs:

- IBM's MQ Explorer v5.2, v5.3 and v6.0
- SupportPac MO71 (MQMon)
- IBM's WBIMB Eclipse Tool Kit
- Mercury's SiteScope
- Capitalware's MQ Visual Edit
- Capitalware's MQ Visual Browse
- Capitalware's MQ Batch Toolkit
- Any program that uses Client Channel Tables (i.e. SupportPac MS03, WatchQ, etc.)
- J2EE web server (i.e. WebLogic, WebSphere, etc.)

1.3 Context Diagram (Logical View)



1.4 Security Message Flow (Logical View)



1.5 Prerequisites

This section provides the minimum supported software levels. These prerequisites apply to both client-side and server-side installations of MQ Authenticate User Security Exit for z/OS.

1.5.1 Operating System

MQ Authenticate User Security Exit for z/OS can be installed on any of the following supported servers:

1.5.1.1 IBM z/OS

- IBM z/OS v1.4 or higher

1.5.2 WebSphere MQ

- WebSphere MQ v5.3.1
- WebSphere MQ v6.0 and v7.0

2 Client-side Security Exit Tested Applications

This section describes on what applications has been tested with the client-side security exit.

The client-side security exit has been built and tested on the following operating systems:

- Java v1.3 or higher on platforms that support a JVM of v1.3 or higher
- Windows XP / 2000 / 2003 native GUI and non-GUI mode
- J2EE applications native non-GUI mode
- AIX v5.1 native non-GUI mode
- HP-UX v11 native non-GUI mode
- Solaris v8 native non-GUI mode
- Red Hat Linux v8 native non-GUI mode

The client-side security exit has been tested with the following applications:

- IBM's MQ Explorer v5.2, v5.3 and v6.0
- SupportPac MO71 (MQMon)
- IBM's WBIMB Eclipse Tool Kit
- Mercury's SiteScope
- Capitalware's MQ Visual Edit
- Capitalware's MQ Visual Browse
- Capitalware's MQ Batch Toolkit
- Any program that uses Client Channel Tables (i.e. SupportPac MS03, WatchQ, etc.)

3 Appendix C – Encryption

MQ Authenticate User Security Exit for z/OS Solution uses the ‘Tiny Encryption Algorithm Variant’ (called TEAV or XTEA) for encryption and decryption of the user’s password between the client-side security exit and the server-side security exit.

3.1 TEA Encryption Algorithm

This is relatively new, sufficiently strong and very compact and fast block cipher algorithm with a 128-bit key. It is not patented and is available in public domain.

Initially, the *Tiny Encryption Algorithm* (TEA) was developed by David Wheeler and Roger Needham of Cambridge University Computer Lab, UK, in 1994:
<http://www.ftp.cl.cam.ac.uk/ftp/papers/djw-rmn/djw-rmn-tea.html>

Later it was enhanced and renamed

- Block TEA, XTEA or TEAN, 1997:
<http://www.ftp.cl.cam.ac.uk/ftp/users/djw3/xtea.ps>

- And XXTEA, 1998:
<http://www.ftp.cl.cam.ac.uk/ftp/users/djw3/xxtea.ps>

The review, cryptanalysis, summary of attacks and discussion is presented by Matthew D. Russell in ‘An Overview of TEA and Related Ciphers’, 2004:
<http://www-users.cs.york.ac.uk/~matthew/TEA/TEA.html>

Also see the *Tiny Encryption Algorithm* website maintained by Simon Shepherd, Professor of Computational Mathematics, Director of the Cryptography and Computer Security Laboratory, Bradford University, England:
<http://www.simonshepherd.supanet.com/tea.htm>