Universal Broker 7.7.x Reference Guide

Universal Agent 7.7.x

Table of Contents

1 Universal Broker

Universal Broker manages Universal Agent components.

This document provides operating system-specific detailed technical information for Universal Broker:

- Started task (z/OS)
- Configuration Options
- Component Definition options
- Universal Access Control List entries

1.1 Universal Broker Utilization

For information how Universal Broker is utilized, see the Universal Agent 7.7.x User Guide.

2 Detailed Information

The following pages provide detailed information for Universal Broker:

3 Universal Broker Examples

See Starting and Stopping Agent Components - Examples for examples of how to start and stop Universal Broker.

See Maintaining Universal Broker Definitions in UEC Database for examples of how to maintain Universal Broker definitions in the UEC Database.

4 Universal Broker Key Store

The Universal Broker Key Store feature lets you generate a random 32-byte encryption key, for use with AES encryption, and a Universal Broker key store in which to locate the key.

Every Universal Broker in your enterprise can have a key store; each key store has a single encryption key.

During installation, you can generate the encryption key and place it in a key store of the local Universal Broker. All command files encrypted by Universal Encrypt (UENCRYPT) will use this encryption key.

The Universal Broker KEYSTORE_PATH configuration option lets you select a remote Universal Broker key store. UENCRYPT then will use the encryption key in that key store for encrypting files.

If you do not generate an encryption key and key store during installation, UENCRYPT lets you do so.

Only Universal Broker has access to its key store. Any component that wants to encrypt/decrypt a file requests the encryption key from Universal Broker. If access to the encryption key is granted, and the encryption key exists, Universal Broker sends the encryption key to the component. Otherwise, the default hard-coded encryption key is used by the component.

All Universal Agent components that are not local Broker-aware - Universal Automation Center Agent (UAG), OMS Admin, Universal Certificate (UCERT), and Universal Products Install Merge (UPIMERGE) - have their own KEYSTORE_PATH configuration option for specifying the path to a local or remote Universal Broker service interface from which the encryption key can be obtained.

If an encryption key is not generated or specified, UENCRYPT uses a default, 8-byte key for encryption.

4.1 Key Store Configuration Options for Universal Encrypt

Universal Encrypt (UENCRYPT) contains the following configuration options for encryption keys and the Universal Broker Key Store:

ENCRYPTION_KEY	User-defined encryption key used to encrypt a command file. If you specify an encryption key with this option, and the Universal Broker Key Store contains an encryption key, UENCRYPT uses the encryption key specified with this option.
GENERATE_KEY	Specification for whether or not to generate an encryption key. GENERATE_KEY either writes a generated encryption key to the *local* Universal Broker key store specified by the KEYSTORE_PATH Universal Encrypt configuration option or, if the STORE_KEY Universal Encrypt configuration option is yes, to a *remote* key store location specified by the KEYSTORE_PATH Universal Broker configuration option.
KEYSTORE_PATH	Path to the local Universal Broker key store.
STORE_KEY	Specification for whether or not to store the encryption key (generated or specified explicitly) in a remote Universal Broker key store specified by the Universal Broker KEYSTORE_PATH configuration option.

4.2 Key Store Location

The local Universal Broker key store is placed at the following locations by default:

UNIX	/var/opt/universal/keystore

Windows	..\keystore
z/OS	dd:UNVKSTR

5 Universal Broker for z/OS

5.1 Started Task

The following figure illustrates the JCL procedure for the Universal Broker started task. **UBROKER** is the member name of this JCL procedure in the Universal Agent sample library (**SUNVSAMP**).

```
//UBROKER  PROC HLQ=#SHLQ.UNV,
//              DBHLQ=#PHLQ.UNV,
//              PHLQ=#PHLQ.UNV,
//              SAPRFC=USPRFC00,
//              RGN=50M,
//              UPARM=,
//              LEPARM=
//*
//S1       EXEC PGM=UBROKER,REGION=&RGN,TIME=NOLIMIT,
//              PARM='ENVAR(TZ=EST5EDT) &LEPARM/&UPARM'
//STEPLIB  DD  DSN=&HLQ..SUNVLOAD,
//              DISP=SHR
//UNVCONF  DD  DSN=&PHLQ..UNVCONF,
//              DISP=SHR
//UNVCOMP  DD  DSN=&PHLQ..UNVCOMP,
//              DISP=SHR
//UNVRFC   DD  DSN=&PHLQ..UNVCONF(&SAPRFC),
//              DISP=SHR
//UNVNLS   DD  DSN=&HLQ..SUNVNLS,
//              DISP=SHR
//UNVTMPL  DD  DSN=&HLQ..SUNVTMPL,
//              DISP=SHR
//UNVKSTR  DD  DSN=&PHLQ..UNVKSTR,
//              DISP=SHR
//UNVCREF  DD  DSN=&PHLQ..UNVCREF,
//              DISP=SHR
```

```
//UNVDB    DD  DSN=&DBHLQ..UNVDB,
//             DISP=SHR
//UNVSPOOL DD  DSN=&DBHLQ..UNVSPOOL,
//             DISP=SHR
//UNVTRACE DD  DSN=&PHLQ..UNVTRACE,
//             DISP=SHR
//UNVTRMDL DD  DSN=&PHLQ..MDL,
//             DISP=SHR
//UNVLOG   DD  SYSOUT=*,HOLD=YES
//SYSPRINT DD  SYSOUT=*,HOLD=YES    -- standard output
//SYSOUT   DD  SYSOUT=*,HOLD=YES    -- standard error
//CEEDUMP  DD  SYSOUT=*,HOLD=YES    -- LE dumps
//SYSUDUMP DD  SYSOUT=*,HOLD=YES    -- system dumps
//SYSIN    DD  DUMMY                -- standard input
```

5.1.1 DD Statements used in JCL Procedure

The following table describes the DD statements used in the Universal Broker for z/OS JCL procedure, above.

ddname	DCB Attributes	Mode	Description
STEPLIB	DSORG=PO, RECFM=U	input	Universal Agent load library containing the program being executed.
UNVCONF	DSORG=PO, RECFM=(F, FB, V, VB)	input	Configuration members for all Universal Agent components.
UNVCOMP	DSORG=PO, RECFM=(F, FB, V, VB)	input	Universal Broker component definition PDS.
UNVRFC	DSORG=PS, RECFM=(F, FB, V, VB)	input	SAP RFC file used by Universal Connector.
UNVNLS	DSORG=PO, RECFM=(F, FB, V, VB)	input	Universal Agent national language support library. Contains message catalogs and code page translation tables.
UNVTMPL	DSORG=PO, RECFM=(V, VB)	input	Universal Agent configuration template library.
UNVKSTR	DSORG=PO, RECFM=(VB), LRECL=2086 or above	input, output	Universal Broker Keystore data set.
UNVCREF	DSORG=PO, RECFM=(F, FB, V, VB)	input	Universal Command Server command reference PDS.
UNVDB	DSNTYPE=HFS	input, output	Universal Broker database. **Note** This ddname is not used if zFS data sets are used instead of HFS data sets.

ddname	DCB Attributes	Mode	Description
UNVSPOOL	DSNTYPE=HFS	input, output	Universal Agent spool database. **Note** This ddname is not used if zFS data sets are used instead of HFS data sets.
UNVTRACE	DSORG=PO, RECFM=(F, FB, V, VB), LRECL=256 or above.	output	Universal Agent trace PDS. This ddname is used only if UNVTRMDL is not defined.
UNVTRMDL	DSORG=PS, RECFM=(F,FB,V,VB), LRECL=256 or above.	output	Universal Agent trace model data set. The data set name is used as the high-level qualifier of the dynamically allocated trace data sets.
UNVLOG	DSORG=PS, RECFM=(F,FB,V,VB), LRECL=256 or above.	output	Universal Broker message destination ddname when option MESSAGE_DESTINATION value is LOGFILE.
SYSPRINT	DSORG=PS, RECFM=(F, FB, V, VB)	output	Standard output file for the UBROKER program.
SYSOUT	DSORG=PS, RECFM=(F, FB, V, VB)	output	Standard error file for the UBROKER program.
SYSIN	DSORG=PS, RECFM=(F, FB, V, VB)	input	Standard input file for the UBROKER program.

5.1.2 Started Task System Commands

The Universal Broker started task is managed using the following z/OS system commands. For a complete description of z/OS system commands, refer to the IBM *z/OS MVS System Commands* manual.

5.1.2.1 START Command

The START system command starts the Universal Broker started task.

```
S UBROKER
```

5.1.2.2 STOP Command

The STOP system command stops the Universal Broker started task.

```
P UBROKER
```

5.1.2.3 MODIFY Command

The MODIFY command sends the specified command option to the Universal Broker for processing.

The Universal Broker STC supports the following MODIFY commands.

```
F UBROKER,APPL=cmd
```

The following *cmd* values are supported.

`REFRESH`	Refresh the Universal Broker configuration data.
`UAG,STATUS`	Display the Universal Automation Center Agent (UAG) SMF exit status. Syntax: `APPL=UAG,STATUS [ ALL | SMF | HCOM ]` • `STATUS` and `STATUS,SMF` display the SMF exit data. • `STATUS,HCOM` displays the HCS High Common Storage information. • `STATUS,ALL` displays both.
`UAG,LOGSWITCH`	Close the active UAG agent log data set and opens a new one.

5.1.2.4 TRACE Command

The TRACE command turns tracing on (the default), off or closes the active trace dataset and opens a new one.

```
F <ubroker>,APPL=UAG,TRACE [ ,ON | , OFF | ,SWITCH }
```

5.2 z/OS Console Commands

```
F <ubroker>,APPL=UAG,PRIMARY
```

This command causes an agent that is running in Sysplex Secondary mode to become a Primary agent until it is restarted or otherwise caused to become a Secondary agent.

If the agent is not running in Secondary mode, or a Primary agent is already active with the same system ID, the command will fail.

```
F <ubroker>,APPL=UAG,SECONDARY
```

This command causes an agent that is running in Sysplex Primary mode to become a Secondary agent until it is restarted or otherwise caused to become a Primary agent.

If the agent is not running in Primary mode, the command will fail.

```
F <ubroker>,APPL=SHUTDOWN, [ FAILOVER [ ,<sysname> ] | NOFAILOVER ]
```

When issued against a Secondary agent	This command behaves like the z/OS STOP command (P <ubroker>).
When issued against a Primary agent	This command shuts down the Broker (and agent) while controlling the Sysplex failover behaviour:
When issued without the FAILOVER or NOFAILOVER parameter	Failover will behave as configured by the automatic_failover parameter in UAGCFG00.
When FAILOVER Is specified	An available Secondary agent will take over as Primary, regardless of how failover is configured. When the optional < `sysname` > is specified, the agent running on the designated z/OS system will take over as Primary agent regardless of how fail over is configured.
When NOFAILOVER Is specified	No Secondary agent will take over as Primary, regardless of how failover is configured.

Note

Behaviour of the z/OS STOP console command with failover is identical to the F <ubroker>,APPL=SHUTDOWN command with no other parameters.

5.3 Configuration

Universal Broker reads configuration options only from the Universal Broker configuration file, which is allocated to ddname **UNVCONF**.

5.3.1 Configuration Options

The following table identifies all of the Universal Broker for z/OS configuration options. Each **Option Name** is a link to detailed information about that option.

Option Name	Description
ACTIVITY_MONITORING	Specification for whether or not product activity monitoring events are generated.
CA_CERTIFICATES	Path to PEM formatted trusted CA X.509 certificates.
CERTIFICATE	Path to Broker's PEM formatted X.509 certificate.
CERTIFICATE_EXPIRATION_NOTICE	Number of days prior to certificate expiration to begin issuing informational messages about the expiration.

CERTIFICATE_REVOCATION_LIST	Path to PEM formatted CRL.
CODE_PAGE	Text translation code page.
COMPONENT_BACKLOG	Component interface backlog size for pending connection requests.
CTL_SSL_CIPHER_LIST	SSL/TLS cipher list for the control sessions.
DNS_CACHE_TIMEOUT	Time-out for DNS cache.
EVENT_GENERATION	Events to be generated as persistent event records.
MESSAGE_DESTINATION	Location where messages are written.
MESSAGE_LANGUAGE	Language of written messages.
MESSAGE_LEVEL	Level of messages written.
MIN_SSL_PROTOCOL	Minimum SSL/TLS protocol level that will be negotiated and used for communications channels.
MONITOR_EVENT_EXPIRATION	Duration of a monitoring event record in the Universal Broker local UES database.
MOUNT_POINT	HFS or zFS database mount directory.
MOUNT_POINT_MODE	HFS or zFS permission mode for MOUNT_POINT.
MSG_SUPPRESSION_LIST	List of message IDs representing Universal messages to be suppressed.
PERSISTENT_EVENT_EXPIRATION	Duration of a persistent event record in the Universal Broker local UES database.
PRIVATE_KEY	Path to Broker's PEM formatted RSA private key.
PRIVATE_KEY_PWD	Password for the Broker's PRIVATE_KEY.
REQ_USAP_CONN	Number of SAP connections that Universal Broker will request from a pool of connections permitted by your Universal Agent license.
REQUIRE_SSL	Specification whether or not Universal Broker will enforce the use of SSL/TLS connections by the clients (managers) of Universal Command Server and Universal Data Mover Server,
RETRY_MAX_COMP	Specification for whether or not the Start Component request is retryable when the maximum number of components are running.
RUNNING_MAX	Maximum number of simultaneous components.
SAF_KEY_RING	SAF certificate key ring name.
SAF_KEY_RING_LABEL	SAF certificate key ring label.
SERVICE_BACKLOG	Service interface backlog size for pending connection requests.
SERVICE_IP_ADDRESS	TCP/IP address on which the Broker listens.
SERVICE_PORT	TCP/IP port number on which the Broker listens.
SHARED_MOUNT_POINT	zFS database shared mount directory.
SHARED_MOUNT_POINT_MODE	zFS permission mode for SHARED_MOUNT_POINT.
SMF_EXIT_LOAD_LIBRARY	UNVACTRT SMF exit load library.

SSL_IMPLEMENTATION	SSL/TLS implementation.
SYSPLEX_ROLE	Role that Universal Broker will perform in a Sysplex configuration.
SYSTEM_ID	Broker running on a system (O/S image).
TMP_DIRECTORY	z/OS UNIX directory name for temporary files.
TRACE_FILE_LINES	Maximum number of lines written to the trace file.
TRACE_TABLE	Memory trace table specification.
UCMD_PATH	Absolute path to the UCMD external link that was created manually on the USS file system to support disabling the UID 0 requirement for the Universal Broker started task.
UCMD_STC_SUPPORT	Support for Universal Command started tasks.
UCTL_PATH	Absolute path to the UCTL external link that was created manually on the USS file system to support disabling the UID 0 requirement for the Universal Broker started task.
UNIX_DB_DATA_SET	HFS or zFS data set used for the Universal Broker's databases.
UNIX_SPOOL_DATA_SET	HFS or zFS data set used for the Universal Broker's spool.
USAP_PATH	Absolute path to the USAP external link that was created manually on the USS file system to support disabling the UID 0 requirement for the Universal Broker started task.

5.4 Component Management

Universal Broker is aware only of Universal Agent components that have been defined. It is the responsibility of Universal Broker to start, stop, and query these defined components.

One of the steps in the installation of a component is defining it to the local Universal Broker. These component definitions provide Universal Broker with the necessary information that it needs to manage the components.

5.4.1 Component Definitions

Component definitions are text files that define Universal Agent components to the Universal Broker. All z/OS component definition files are located in the Universal Broker component definition library **UNVCOMP** allocated to the **UNVCOMP** ddname.

The syntax of a component definition file is the same as the Universal Broker configuration file.

The following table identifies all of the options that comprise Universal Agent for z/OS component definitions. Each **Option Name** is a link to detailed information about that option.

Option Name	Description
AUTOMATICALLY_START	Specification for whether the component automatically starts by the Universal Broker at start-up time or only on demand.
COMPONENT_NAME	Name by which clients know the component.
COMPONENT_TYPE	Type of component.
CONFIGURATION_FILE *	Component's configuration file name.

Option Name	Description
RESTART	Specification for whether or not the component should be restarted if it ends.
RESTART_CONDITIONS	Exit conditions criteria for which the server is considered eligible for restart.
RESTART_DELAY	Number of seconds to wait before restarting.
RESTART_MAX_FREQUENCY	Maximum frequency a server can be restarted.
RUNNING_MAXIMUM	Maximum number of this component that can run simultaneously.
START_COMMAND *	Component program member name.
WORKING_DIRECTORY *	Path used as the working directory of the component.
* These options are required in the component definitions.	

5.5 Universal Access Control List

The Universal Broker uses the Universal Access Control List (UACL) file as an extra layer of security. The UACL file contains Universal Broker entries that contain Access Control List (ACL) rules that permit or deny access to the Universal Broker.

The Universal Broker reads in the UACL entries when the program is started. If the UACL file is changed, the new entries can be activated by recycling the Broker or by sending the Universal Broker a Universal Control REFRESH command that will instruct the Universal Broker to reread all its configuration files including the UACL file.

5.5.1 UACL Entries

The syntax of a UACL entry file is the same as the Universal Broker configuration file.

The following table identifies all UACL entries for Universal Broker for z/OS. Each **UACL Entry Name** is a link to detailed information about that option.

UACL Entry Name	Description
UBROKER_ACCESS	Allows or denies access to Universal Broker services.
CERT_MAP	Maps a client X.509 certificate to a certificate identifier.
EVENT_ACCESS	Controls which Universal Enterprise Controller has read and delete access to the Universal Event Subsystem event data maintained by the Universal Broker.
REMOTE_CONFIG_ACCESS	Authorizes update access to the product configuration files and setting of the configuration managed mode of the Broker.

6 Universal Broker for Windows

6.1 Configuration

Universal Broker reads its configuration options from the Universal Broker configuration file, `ubroker.conf`.

This file can be edited manually with any text editor.

For overriding Universal Agent component definition options and UAG configuration options - in a Windows console mode broker environment - command line options are provided.

Also, for overriding the default Universal Configuration in a Windows console mode broker environment, command line options are provided.

6.1.1 Configuration Options

The following table identifies all of the Universal Broker for Windows configuration options. Each **Option Name** is a link to detailed information about that option.

Option Name	Description
ACTIVITY_MONITORING	Specification for whether or not product activity monitoring events are generated.
BIF_DIRECTORY	Broker Interface Directory where Universal Broker will create its broker interface file.
CA_CERTIFICATES	Path to PEM formatted trusted CA X.509 certificates.
CERTIFICATE	Path to Broker's PEM formatted X.509 certificate.
CERTIFICATE_EXPIRATION_NOTICE	Number of days prior to certificate expiration to begin issuing informational messages about the expiration.
CERTIFICATE_REVOCATION_LIST	Path to PEM formatted CRL.
CODE_PAGE	Text translation code page.
COMPONENT_BACKLOG	Component interface backlog size for pending connection requests.
COMPONENT_PORT	TCP/IP port used for Broker-Component communications.

CTL_SSL_CIPHER_LIST	SSL/TLS cipher list for the control sessions.
DNS_CACHE_TIMEOUT	Time-out for DNS cache.
EVENT_GENERATION	Events to be generated as persistent event records.
INSTALLATION_DIRECTORY	Base directory where product is installed.
LOG_DIRECTORY	Directory where log files are created.
LOG_FILE_GENERATIONS	Total number of log files that will be saved within the log directory.
LOG_FILE_LINES	Total number of lines to be written to the log file before the log file is wrapped.
MESSAGE_DESTINATION	Location where messages are written.
MESSAGE_LANGUAGE	Language of written messages.
MESSAGE_LEVEL	Level of messages written.
MSG_SUPPRESSION_LIST	List of message IDs representing Universal messages to be suppressed.
MIN_SSL_PROTOCOL	Minimum SSL/TLS protocol level that will be negotiated and used for communications channels.
MONITOR_EVENT_EXPIRATION	Duration of a monitoring event record in the Universal Broker local UES database.
NLS_DIRECTORY	Location of UMC and UTT files.
PERSISTENT_EVENT_EXPIRATION	Duration of a persistent event record in the Universal Broker local UES database.
PID_FILE_DIRECTORY	PID file location.
PRIVATE_KEY	Path to Broker's PEM formatted RSA private key.
PRIVATE_KEY_PWD	Password for the Broker's PRIVATE_KEY.
REQ_UPPS_CONN	Number of PeopleSoft connections that Universal Broker will request from a pool of connections permitted by your Universal Agent license.
REQ_USAP_CONN	Number of SAP connections that Universal Broker will request from a pool of connections permitted by your Universal Agent license.
REQUIRE_SSL	Specification for whether or not Universal Broker will enforce the use of SSL/TLS connections by the clients (managers) of Universal Command Server and Universal Data Mover Server,
RUNNING_MAX	Maximum number of simultaneous components.
RETRY_MAX_COMP	Specification for whether or not the Start Component request is retryable when the maximum number of components are running.
SERVICE_BACKLOG	Service interface backlog size for pending connection requests.
SERVICE_IP_ADDRESS	TCP/IP address on which the Broker listens.
SERVICE_PORT	TCP/IP port number on which the Broker listens.
SPOOL_DIRECTORY	Spool file directory.
TMP_DIRECTORY	Temporary file directory.

TRACE_DIRECTORY	Trace file directory.
TRACE_FILE_LINES	Maximum number of lines written to the trace file.
TRACE_TABLE	Memory trace table specification.
WORKING_DIRECTORY	Broker's working directory.
Override Options	The following Universal Broker configuration options let you override, at Universal Broker start-up, UAG, OMS, and UEM automatically start component definition options and multiple UAG configuration options.
UAG_AUTOSTART	UAG AUTOMATICALLY_START component definition option override.
UEM_AUTOSTART	UEM AUTOMATICALLY_START component definition option override.
OMS_AUTOSTART	OMS AUTOMATICALLY_START component definition option override.
UAG_AGENT_CLUSTERS	UAG AGENT_CLUSTERS configuration option override.
UAG_EXTENSION_ACCEPT_LIST	UAG EXTENSION_ACCEPT_LIST configuration option override.
UAG_EXTENSION_CANCEL_TIMEOUT	UAG EXTENSION_CANCEL_TIMEOUT configuration option override.
UAG_EXTENSION_DEPLOY_ON_REGISTRATION	UAG EXTENSION_DEPLOY_ON_REGISTRATION configuration option override.
UAG_EXTENSION_PYTHON_LIST	UAG EXTENSION_PYTHON_LIST configuration option override.
UAG_NETNAME	UAG NETNAME configuration option override.
UAG_OMS_SERVERS	UAG OMS_SERVERS configuration option override.
UAG_TRANSIENT	UAG TRANSIENT configuration option override.

6.2 Component Management

Universal Broker is aware only of Universal Agent components that have been defined to it. It is the responsibility of Universal Broker to start, stop, and query these defined components.

One of the steps in the installation of a component is defining it to the local Universal Broker. These component definitions provide Universal Broker with the necessary information that it needs to manage the components.

6.2.1 Component Definitions

Component definitions are text files that define Universal Agent components to the Universal Broker.

Component definition files reside in `%ALLUSERSPROFILE%\Application Data\Universal\comp`, where `%ALLUSERSPROFILE%` is an environment variable that resolves by default to:

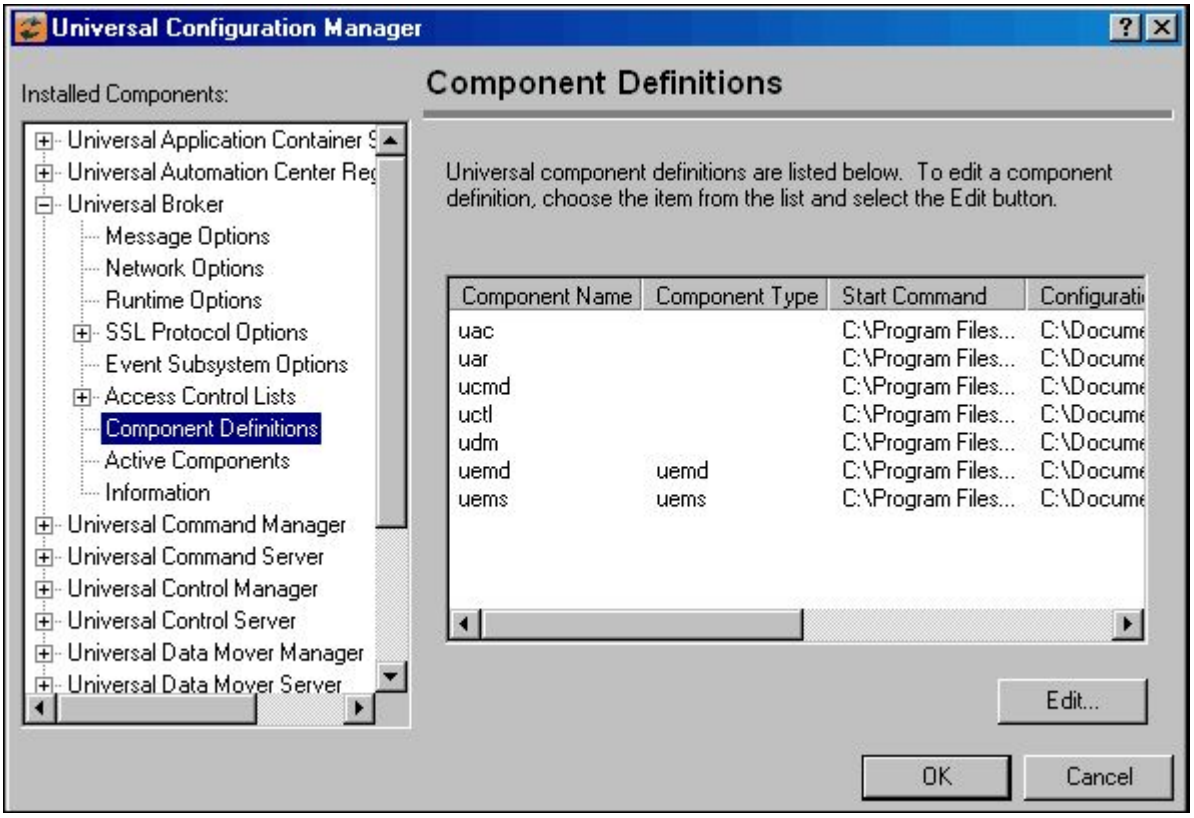
- `C:\Documents and Settings\All Users` on Windows 2000/XP/Server 2003
- `C:\ProgramData` on Windows Vista/Server 2008

The syntax of a component definition file is the same as the Universal Broker configuration file.

Although component definition files can be edited with any text editor (for example, Notepad), the Universal Configuration Manager application is the recommended way to edit component definitions for Windows.

Note

The component definitions for all Universal Agent are identified in the Component Definitions property page of the Universal Broker, as shown below.



The following identifies all of the options that comprise Universal Agent for Windows component definitions. Each **Option Name** is a link to detailed information about that option.

Option Name	Description
AUTOMATICALLY_START	Specification for whether the component automatically starts by the Universal Broker at start-up time or only on demand.
COMPONENT_NAME	Name by which clients know the component.
COMPONENT_TYPE	Type of component.
CONFIGURATION_FILE *	Component's configuration file name.
RESTART	Specification for whether or not the component should be restarted if it ends.
RESTART_CONDITIONS	Exit conditions criteria for which the server is considered eligible for restart.
RESTART_DELAY	Number of seconds to wait before restarting.
RESTART_MAX_FREQUENCY	Maximum frequency a server can be restarted.

Option Name	Description
RUNNING_MAXIMUM	Maximum number of this component that can run simultaneously.
START_COMMAND *	Command that starts the component.
WORKING_DIRECTORY *	Path used as the working directory of the component.
* These options are required in the component definitions.	

6.3 Universal Access Control List

Universal Broker uses the Universal Access Control List (UACL) as an extra layer of security. The UACL contains Broker entries that contain Access Control List (ACL) rules that permit or deny access to the Broker.

Universal Broker reads the UACL entries when the program is started. If the UACL file is changed, the new entries can be activated either by:

- Stopping and starting Universal Broker.
- Sending Universal Broker a Universal Control REFRESH command, which instructs Universal Broker to reread all of its configuration files, including the UACL file.

Note

Although the UACL file, like all configuration files, can be edited with any text editor (for example, Notepad), the Universal Configuration Manager application, accessible via the Control Panel, is the recommended way to change UACL entries.

Via this method, a REFRESH command is sent to Universal Broker, and any new entries take effect immediately. There is no need to stop and restart the Broker in order for the changes to be applied.

6.3.1 UACL Entries

The syntax of a UACL entry file is the same as the Universal Broker configuration file.

The following table identifies all Universal Broker for Windows UACL entries. Each **UACL Entry Name** is a link to detailed information about that option.

UACL Entry Name	Description
UBROKER_ACCESS	Allows or denies access to Universal Broker services
CERT_MAP	Maps a client X.509 certificate to a certificate identifier.
EVENT_ACCESS	Controls which Universal Enterprise Controller has read and delete access to the Universal Event Subsystem event data maintained by the Universal Broker.
REMOTE_CONFIG_ACCESS	Authorizes update access to the product configuration files and setting of the configuration managed mode of the Broker.

7 Universal Broker for UNIX

7.1 Configuration

Universal Broker reads its configuration options from the Universal Broker configuration file, `ubroker.conf`.

This file can be edited manually with any text editor.

For overriding Universal Agent component definition options and UAG configuration options, environment variable and command line options are provided.

Also, for overriding the default Universal Broker and UAG run-time configurations, environment variables and command line options are provided.

7.1.1 Configuration Options

The following table identifies all of the Universal Broker for UNIX configuration options. Each **Option Name** is a link to detailed information about that option.

Option Name	Description
ACTIVITY_MONITORING	Specification for generation of product activity monitoring events.
BIF_DIRECTORY	Broker Interface Directory where Universal Broker will create its broker interface file.
CA_CERTIFICATES	Path to PEM formatted trusted CA X.509 certificates.
CERTIFICATE	Path to Broker's PEM formatted X.509 certificate.
CERTIFICATE_EXPIRATION_NOTICE	Number of days prior to certificate expiration to begin issuing informational messages about the expiration.
CERTIFICATE_REVOCATION_LIST	Path to PEM formatted CRL.
CODE_PAGE	Text translation code page.
COMPONENT_BACKLOG	Component interface backlog size for pending connection requests.

COMPONENT_DIRECTORY	Component definition file directory.
CTL_SSL_CIPHER_LIST	SSL cipher list for the control sessions.
DNS_CACHE_TIMEOUT	Time-out for DNS cache.
EVENT_GENERATION	Events to be generated as persistent event records.
INSTALLATION_DIRECTORY	Base directory where product is installed.
LOG_DIRECTORY	Log file directory.
LOG_FILE_GENERATIONS	Total number of log files that will be saved within the log directory.
LOG_FILE_LINES	Total number of lines to be written to the log file before the log file is wrapped.
MESSAGE_DESTINATION	Location where messages are written.
MESSAGE_LANGUAGE	Language of written messages.
MESSAGE_LEVEL	Level of messages written.
MIN_SSL_PROTOCOL	Minimum SSL protocol level that will be negotiated and used for communications channels.
MONITOR_EVENT_EXPIRATION	Duration of a monitoring event record in the Universal Broker local UES database.
MSG_SUPPRESSION_LIST	List of message IDs representing Universal messages to be suppressed.
NLS_DIRECTORY	UMC and UTT file directory.
PERSISTENT_EVENT_EXPIRATION	Duration of a persistent event record in the Universal Broker local UES database.
PID_FILE_DIRECTORY	PID file location.
PRIVATE_KEY	Path to Broker's PEM formatted RSA private key.
PRIVATE_KEY_PWD	Password for the Broker's PRIVATE_KEY.
REQ_UPPS_CONN	Number of PeopleSoft connections that Universal Broker will request from a pool of connections permitted by your Universal Agent license.
REQ_USAP_CONN	Number of SAP connections that Universal Broker will request from a pool of connections permitted by your Universal Agent license.
REQUIRE_SSL	Specification for whether or not Universal Broker will enforce the use of SSL connections by the clients (managers) of Universal Command Server and Universal Data Mover Server,
RETRY_MAX_COMP	Specification for whether or not the Start Component request is retryable when the maximum number of components are running.
RUNNING_MAX	Maximum number of simultaneous components.
SERVICE_BACKLOG	Service interface backlog size for pending connection requests.
SERVICE_IP_ADDRESS	TCP/IP address on which the Broker listens.
SERVICE_PORT	TCP/IP port number on which the Broker listens.
SPOOL_DIRECTORY	Spool file directory.
TMP_DIRECTORY	Temporary file directory.

TRACE_DIRECTORY	Trace file directory.
TRACE_FILE_LINES	Maximum number of lines written to the trace file.
TRACE_TABLE	Memory trace table specification.
WORKING_DIRECTORY	Broker's working directory.
Override Options	The following Universal Broker configuration options let you override UAG, OMS, and UEM automatically start component definition options and multiple UAG configuration options at Universal Broker start-up.
UAG_AUTOSTART	UAG AUTOMATICALLY_START component definition option override.
UEM_AUTOSTART	UEM AUTOMATICALLY_START component definition option override.
OMS_AUTOSTART	OMS AUTOMATICALLY_START component definition option override.
UAG_AGENT_CLUSTERS	UAG AGENT_CLUSTERS configuration option override.
UAG_EXTENSION_ACCEPT_LIST	UAG EXTENSION_ACCEPT_LIST configuration option override.
UAG_EXTENSION_CANCEL_TIMEOUT	UAG EXTENSION_CANCEL_TIMEOUT configuration option override.
UAG_EXTENSION_DEPLOY_ON_REGISTRATION	UAG EXTENSION_DEPLOY_ON_REGISTRATION configuration option override.
UAG_EXTENSION_PYTHON_LIST	UAG EXTENSION_PYTHON_LIST configuration option override.
UAG_NETNAME	UAG NETNAME configuration option override.
UAG_OMS_SERVERS	UAG OMS_SERVERS configuration option override.
UAG_TRANSIENT	UAG TRANSIENT configuration option override.

7.2 Component Management

Universal Broker is aware only of Universal Agent components that have been defined. It is the responsibility of Universal Broker to start, stop, and query these defined components.

One of the steps in the installation of a component is defining it to the local Universal Broker. These component definitions provide Universal Broker with the necessary information that it needs to manage the components.

7.2.1 Component Definitions

Component definitions are text files that define Universal Agent components to the Universal Broker. All UNIX component definition files are located in the Universal Broker component definition directory (specified with the COMPONENT_DIRECTORY configuration option).

The syntax of a component definition file is the same as the Universal Broker configuration file.

The following table identifies all of the options that comprise Universal Agent for UNIX component definitions. Each **Option Name** is a link to detailed information about that option.

Option Name	Description
AUTOMATICALLY_START	Specification for whether the component automatically starts by the Universal Broker at start-up time or only on demand.
COMPONENT_NAME	Name by which clients know the component.
COMPONENT_TYPE	Type of component.
CONFIGURATION_FILE *	Component's configuration file name.
RESTART	Specification for whether or not the component should be restarted if it ends.
RESTART_CONDITIONS	Exit conditions criteria for which the server is considered eligible for restart.
RESTART_DELAY	Number of seconds to wait before restarting.
RESTART_MAX_FREQUENCY	Maximum frequency a server can be restarted.
RUNNING_MAXIMUM	Maximum number of this component that can run simultaneously.
START_COMMAND *	Command that starts the component.
WORKING_DIRECTORY *	Path used as the working directory of the component.
* These options are required in the component definitions.	

7.3 Universal Access Control List

Universal Broker uses the Universal Access Control List (UACL) file as an extra layer of security. The UACL file contains Universal Broker entries that contain Access Control List (ACL) rules that permit or deny access to Universal Broker.

Universal Broker reads in the UACL entries when the program is started. If the UACL file is changed, the new entries can be activated either by:

- Stopping and starting Universal Broker
- Sending Universal Broker a Universal Control REFRESH command, which instructs Universal Broker to reread all its configuration files, including the UACL file.

The UNIX REFRESH command is:

```
uctl -refresh -host BROKER-IPADDR
```

7.3.1 UACL Entries

The syntax of a UACL entry file is the same as the Universal Broker configuration file.

The following table identifies all Universal Broker for UNIX UACL entries. Each **UACL Entry Name** is a link to detailed information about that option.

UACL Entry Name	Description
UBROKER_ACCESS	Allows or denies access to Universal Broker services.
CERT_MAP	Maps a client X.509 certificate to a certificate identifier.
EVENT_ACCESS	Controls which Universal Enterprise Controller has read and delete access to the Universal Event Subsystem event data maintained by the Universal Broker.
REMOTE_CONFIG_ACCESS	Authorizes update access to the product configuration files and setting of the configuration managed mode of the Broker.

8 Universal Broker for IBM i

Currently, Universal Agent 7.7.0 for IBM i does not include Universal Command (UCMD) or Universal Data Mover (UDM). Customers who require those components should use an older version of Workload Automation for IBM i. The latest version containing those components is 5.1.1. Older versions of Workload Automation for IBM i can be installed alongside version 7.7.0.

8.1 Configuration

Universal Broker reads configuration options only from the Universal Broker configuration file.

The Universal Broker configuration file is named **UNVPRD770/UNVCONF(UBROKER)**. File **UNVCONF** is a physical source file located in the **UNVPRD770** library. File member **UBROKER** contains the configuration options for the Universal Broker. File **UNVCONF** contains configuration members for the Universal Agent for IBM i components. This file can be edited manually with any text editor.

8.1.1 Configuration Options

The following table identifies all of the Universal Broker for IBM i configuration options. Each **Option Name** is a link to detailed information about that option.

Option Name	Description
ACTIVITY_MONITORING	Specification for whether or not product activity monitoring events are generated.
CA_CERTIFICATES	Path to PEM formatted trusted CA X.509 certificates.
CERTIFICATE	Path to Broker's PEM formatted X.509 certificate.
CERTIFICATE_REVOCATION_LIST	Path to PEM formatted CRL.
CODE_PAGE	Text translation code page.
CTL_SSL_CIPHER_LIST	SSL/TLS cipher list for the control sessions.

DNS_CACHE_TIMEOUT	Time-out for DNS cache.
EVENT_GENERATION	Events to be generated as persistent events.
KEYSTORE_PATH	Specifies the location of a remote Universal Broker key store.
MESSAGE_DESTINATION	Location where messages are written.
MESSAGE_LANGUAGE	Language of written messages.
MESSAGE_LEVEL	Level of messages written.
MIN_SSL_PROTOCOL	Minimum SSL protocol level that will be negotiated and used for communications channels.
MONITOR_EVENT_EXPIRATION	Duration of a monitoring event record in the Universal Broker local UES database.
PERSISTENT_EVENT_EXPIRATION	Duration of a persistent event record in the Universal Broker local UES database.
PRIVATE_KEY	Path to Broker's PEM formatted RSA private key.
PRIVATE_KEY_PWD	Password for the Broker's PRIVATE_KEY.
REQUIRE_SSL	Specification for whether or not Universal Broker will enforce the use of SSL connections by the clients (managers) of Universal Command Server and Universal Data Mover Server,
RUNNING_MAX	Maximum number of simultaneous components.
SERVICE_BACKLOG	Service interface backlog size for pending connection requests.
SERVICE_IP_ADDRESS	TCP/IP address on which the Broker listens.
SERVICE_PORT	TCP/IP port number on which the Broker listens.
TRACE_FILE_LINES	Maximum number of lines written to the trace file.
TRACE_TABLE	Memory trace table specification.

8.2 Component Management

Universal Broker is aware only of Universal Agent components that have been defined. It is the responsibility of Universal Broker to start, stop, and query these defined components.

One of the steps in the installation of a component is defining it to the local Universal Broker. These component definitions provide Universal Broker with the necessary information that it needs to manage the components.

8.2.1 Component Definitions

Component definitions are text files that define Universal Agent components to the Universal Broker. All IBM i component definitions are located in the source physical file **UNVPRD770/UNVCOMP** as individual members.

The syntax of a component definition file is the same as the Universal Broker configuration file.

The following table identifies all of the options that comprise Universal Agent for IBM i component definitions. Each **Option Name** is a link to detailed information about that option.

Option Name	Description
AUTOMATICALLY_START	Specification for whether or not the component automatically starts by the Universal Broker at start-up time or only on demand.
COMPONENT_NAME	Name by which clients know the component.
COMPONENT_TYPE	Type of component.
CONFIGURATION_FILE *	Component's configuration file name.
RESTART	Specification for whether or not the component should be restarted if it ends.
RESTART_CONDITIONS	Exit conditions criteria for which the server is considered eligible for restart.
RESTART_DELAY	Number of seconds to wait before restarting.
RESTART_MAX_FREQUENCY	Maximum frequency a server can be restarted.
RUNNING_MAXIMUM	Maximum number of this component that can run simultaneously.
START_COMMAND *	Component program name.
WORKING_DIRECTORY *	Path used as the working directory of the component.
* These options are required in the component definitions.	

8.3 Universal Access Control List

Universal Broker uses the Universal Access Control List (UACL) file as an extra layer of security. The UACL file contains Universal Broker entries that contain Access Control List (ACL) rules that permit or deny access to the Broker.

Universal Broker reads in the UACL entries when the program is started. If the UACL file is changed, the new entries can be activated either by:

- Stopping and starting Universal Broker.
- Sending Universal Broker a Universal Control **REFRESH** command, which instructs Universal Broker to reread all its configuration files, including the UACL file.

The IBM i REFRESH command is:

```
STRUCT REFRESH(*YES) HOST(hostname)
```

8.3.1 UACL Entries

The syntax of a UACL entry file is the same as the Universal Broker configuration file.

The following table identifies all Universal Broker for IBM i UACL entries. Each **UACL Entry Name** is a link to detailed information about that option.

UACL Entry Name	Description
UBROKER_ACCESS	Allows or denies access to Universal Broker services.
CERT_MAP	Maps a client X.509 certificate to a certificate identifier.
EVENT_ACCESS	Controls which Universal Enterprise Controller has read and delete access to the Universal Event Subsystem event data maintained by the Universal Broker.
REMOTE_CONFIG_ACCESS	Authorizes update access to the product configuration files and setting of the configuration managed mode of the Universal Broker.

9 Universal Broker Configuration Options

9.1 Universal Broker Configuration Options

This page provides links to detailed information on the configuration options available for use with the Universal Broker. Information on how these options are used is documented in the operating system-specific pages of this document.

The options are listed alphabetically, without regard to any specific operating system.

9.2 Configuration Options Information

For each configuration option, the following information is provided.

9.2.1 Description

Describes the configuration option and how it is used.

9.2.2 Usage

Provides a table of the following information:

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line, Long Form	<Format / Value>				
Environment Variable	<Format / Value>				

Configuration File Keyword	<Format / Value>				

9.2.2.1 Method

Identifies the method used to specify Universal Broker configuration options:

- Command Line Option, Long Form
- Environment Variable
- Configuration File Keyword

> **Note**
>
> You can specify most configuration options only using the Configuration File Keyword method. The details for each option identify the available methods.

9.2.2.2 Syntax

Identifies the syntax of the method used to specify the option:

- Format: Specific characters that identify the option.
- Value: Type of value(s) to be supplied for this method.

9.2.2.3 (Operating System)

Identifies the operating systems for which each method of specifying the option is valid:

- IBM i
- UNIX
- Windows
- z/OS

9.2.3 Values

Identifies all possible values for the specified value type.

Defaults are identified in **bold type**.

9.2.4 <Additional Information>

Identifies any additional information specific to the option.

9.3 Configuration Options List

The following table identifies all Universal Broker configuration options.

Option	Description
ACTIVITY_MONITORING	Specification for whether or not product activity monitoring events are generated.
BIF_DIRECTORY	Broker Interface File directory that specifies where Universal Broker will create its interface file.
CA_CERTIFICATES	Path to PEM-formatted trusted CA X.509 certificates.
CERTIFICATE	Path to Broker's PEM-formatted X.509 certificate.
CERTIFICATE_EXPIRATION_NOTICE	Number of days prior to certificate expiration to begin issuing informational messages about the expiration.
CERTIFICATE_REVOCATION_LIST	Path to PEM-formatted CRL.
CODE_PAGE	Text translation code page.
COMPONENT_BACKLOG	Component interface backlog size for pending connection requests.
COMPONENT_DIRECTORY	Component definition file directory.
COMPONENT_PORT	TCP/IP port used for Broker-Component communications.
CTL_SSL_CIPHER_LIST	SSL cipher list for the control sessions.
CTL_SSL_CIPHER_SUITES	SSL/TLS 1.3 specific cipher suites that are acceptable to use for network communications on the control session, which is used for component internal communication.
DB_FILE_GENERATIONS	Number of archived UBroker databases that should be stored.
DNS_CACHE_TIMEOUT	Time-out for DNS cache.
ECDSA_CERTIFICATE	The ECDSA_CERTIFICATE option specifies the file / ddname name of the PEM-formatted ECDSA X.509 certificate that identifies the Universal Broker.
ECDSA_PRIVATE_KEY	The ECDSA_PRIVATE_KEY option specifies the location of the PEM-formatted ECDSA private key that corresponds to the X.509 certificate specified by the ECDSA_CERTIFICATE option.
ECDSA_PRIVATE_KEY_PWD	The ECDSA_PRIVATE_KEY_PWD option specifies the password or passphrase for the PEM-formatted ECDSA private key specified with the ECDSA_PRIVATE_KEY option.
EVENT_GENERATION	Events to be generated as persistent events.
INSTALLATION_DIRECTORY	Base directory where product is installed.
KEYSTORE_PATH	Location of a remote Universal Broker key store.
LOG_DIRECTORY	Log file directory.
LOG_FILE_GENERATIONS	Total number of log files that will be saved within the log directory.
LOG_FILE_LINES	Total number of lines to be written to the log file before the log file is wrapped.
MAX_SSL_PROTOCOL	Maximum SSL/TLS protocol level that will be negotiated and used for communications channels.
MESSAGE_DESTINATION	Location where messages are written.
MESSAGE_LANGUAGE	Language of messages written.

MESSAGE_LEVEL	Level of messages written.
MIN_SSL_PROTOCOL	Minimum SSL protocol level that will be negotiated and used for communications channels. **Note** This option was introduced to IBM i in version 5.1.1.0.
MONITOR_EVENT_EXPIRATION	Duration of a monitoring event record in the Universal Broker local UES database.
MOUNT_POINT	HFS or zFS database mount directory.
MOUNT_POINT_MODE	HFS or zFS permission mode for MOUNT_POINT.
MSG_SUPPRESSION_LIST	List of message IDs representing Universal messages to be suppressed.
NLS_DIRECTORY	UMC and UTT file directory.
PERSISTENT_EVENT_EXPIRATION	Duration of a persistent event record in the Universal Broker local UES database.
PID_FILE_DIRECTORY	PID file location.
PRIVATE_KEY	Path to Broker's PEM formatted RSA private key.
PRIVATE_KEY_PWD	Password for the Broker's PRIVATE_KEY.
REQ_UPPS_CONN	Number of PeopleSoft connections that Universal Broker will request from a pool of connections permitted by your Universal Agent license.
REQ_USAP_CONN	Number of SAP connections that Universal Broker will request from a pool of connections permitted by your Universal Agent license.
REQUIRE_SSL	Specification for whether or not Universal Broker will enforce the use of SSL connections by the clients (managers) of Universal Command Server and Universal Data Mover Server. **Note** This option was introduced to IBM i in version 5.1.1.0.
RETRY_MAX_COMP	Specification for whether or not the Start Component request is retryable when the maximum number of components are running.
RUNNING_MAX	Maximum number of simultaneous components.
SAF_KEY_RING	SAF certificate key ring name.
SAF_KEY_RING_LABEL	SAF certificate key ring label.
SERVICE_BACKLOG	Service interface backlog size for pending connection requests.
SERVICE_IP_ADDRESS	TCP/IP address on which the Broker listens.
SERVICE_PORT	TCP/IP port number on which the Broker listens.
SHARED_MOUNT_POINT	zFS database shared mount directory.

SHARED_MOUNT_POINT_MODE	zFS permission mode for SHARED_MOUNT_POINT .
SMF_EXIT_LOAD_LIBRARY	UNVACTRT SMF exit load library.
SPOOL_DIRECTORY	Spool file directory.
SSL_IMPLEMENTATION	SSL implementation to be used for network configuration.
SYSPLEX_ROLE	Role that Universal Broker will perform in a Sysplex configuration.
SYSTEM_ID	Universal Broker running on a system (O/S image).
TCP_RECV_BUFFER	TCP/IP receive buffer size.
TCP_SEND_BUFFER	TCP/IP send buffer size.
TMP_DIRECTORY	Directory for temporary files.
TRACE_DIRECTORY	Directory for trace files.
TRACE_FILE_LINES	Maximum number of lines written to the trace file.
TRACE_TABLE	Memory trace table specification.
UCMD_STC_SUPPORT	Support for Universal Command started tasks.
UCMD_PATH	Absolute path to the UCMD external link that was created manually on the USS file system to support disabling the UID 0 requirement for the Universal Broker started task.
UCTL_PATH	Absolute path to the UCTL external link that was created manually on the USS file system to support disabling the UID 0 requirement for the Universal Broker started task.
UNIX_DB_DATA_SET	HFS or zFS data set used for the Universal Broker's databases.
UNIX_SPOOL_DATA_SET	HFS or zFS data set used for the Universal Broker's spool.
USAP_PATH	Absolute path to the USAP external link that was created manually on the USS file system to support disabling the UID 0 requirement for the Universal Broker started task.
WORKING_DIRECTORY	Universal Broker's working directory.
Override Options	The following Universal Broker configuration options let you override UAG, OMS, and UEM automatically start component definition options and multiple UAG configuration options at Universal Broker start-up for: • Windows • UNIX
UAG_AUTOSTART	UAG AUTOMATICALLY_START component definition option override.
UEM_AUTOSTART	UEM AUTOMATICALLY_START component definition option override.
OMS_AUTOSTART	OMS AUTOMATICALLY_START component definition option override.
UAG_AGENT_CLUSTERS	UAG AGENT_CLUSTERS configuration option override.
UAG_BUSINESS_SERVICES	UAG BUSINESS_SERVICES configuration option override.
UAG_EXTENSION_ACCEPT_LIST	UAG EXTENSION_ACCEPT_LIST configuration option override.
UAG_EXTENSION_CANCEL_TIMEOUT	UAG EXTENSION_CANCEL_TIMEOUT configuration option override.

UAG_EXTENSION_DEPLOY_ON_REGISTRATION	UAG EXTENSION_DEPLOY_ON_REGISTRATION configuration option override.
UAG_EXTENSION_PYTHON_LIST	UAG EXTENSION_PYTHON_LIST configuration option override.
UAG_NETNAME	UAG NETNAME configuration option override.
UAG_OMS_SERVERS	UAG OMS_SERVERS configuration option override.
UAG_PROCESS_CANCEL_TIMEOUT	UAG PROCESS_CANCEL_TIMEOUT configuration option override.

9.4 ACTIVITY_MONITORING - UBROKER configuration option

9.4.1 Description

The ACTIVITY_MONITORING option specifies whether or not product activity monitoring events are generated.

9.4.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	activity_monitoring *option*	✓	✓	✓	✓

9.4.3 Values

option is the specification for whether or not product activity monitoring events are generated.

Valid values for *option* are:

- **yes**
  Activate product activity monitoring events
- **no**
  Deactivate product activity monitoring events

Default is yes.

9.5 BIF_DIRECTORY - UBROKER configuration option

9.5.1 Description

The BIF_DIRECTORY option specifies the Broker Interface File (BIF) directory where Universal Broker will create its interface file, **ubroker.bif**.

9.5.1.1 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	bif_directory *directory*		✓	✓	

9.5.2 Values

directory is the name of the BIF directory.

9.5.2.1 Default

UNIX	/var/opt/universal
Windows	• System installs: C:\ProgramData\Universal • User mode installs: BIF directory resides in the Install directory

9.6 CA_CERTIFICATES - UBROKER configuration option

9.6.1 Description

The CA_CERTIFICATES option specifies the location of the PEM-formatted trusted Certificate Authority (CA) X.509 certificates file.

Trust CA certificates are required if certificate authentication and verification is desired.

9.6.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	ca_certificates *ddname* or *file*	✓	✓	✓	✓

9.6.3 Values

z/OS	*ddname* is the ddname of the X.509 certificates. The value is used only when the SSL_IMPLEMENTATION option is set to **openssl**. Allocated to the ddname must be either a sequential data set or a member of a PDS that has a variable record format.

UNIX	*file* is the path name of the X.509 certificates file. Relative paths are relative to the current working directory.
Windows	*file* is the path name of the X.509 certificates file. Relative paths are relative to the current working directory.
IBM i	file_ is the qualified file name of the X.509 certificates file. The file name can be qualified by a library name. If not, the library list ***LIBL** is searched for the first occurrence of the file name.

9.7 CERTIFICATE - UBROKER configuration option

9.7.1 Description

The CERTIFICATE option specifies the file / ddname name of the PEM-formatted X.509 certificate that identifies the Universal Broker.

A UCMD Manager X.509 certificate is required if clients require Universal Broker authentication.

> **Note**
>
> If the CERTIFICATE option is used, the PRIVATE_KEY option also is required.

9.7.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	certificate *ddname* or *file*	✓	✓	✓	✓

9.7.3 Values

z/OS	*ddname* is the ddname of the X.509 certificate. The value is used only when the SSL_IMPLEMENTATION option is set to **openssl**. Allocated to the ddname must be either a sequential data set or a member of a PDS that has a variable record format.
UNIX	*file* is the path name of the X.509 certificate file. Relative paths are relative to the current working directory.
Windows	*file* is the path name of the X.509 certificate file. Relative paths are relative to the current working directory.
IBM i	*file* is the qualified file name of the X.509 certificate file. The file name can be qualified by a library name. If not, the library list ***LIBL** is searched for the first occurrence of the file name.

9.8 CERTIFICATE_EXPIRATION_NOTICE - UBROKER configuration option

9.8.1 Description

The CERTIFICATE_EXPIRATION_NOTICE option specifies the number of days prior to certificate expiration to begin issuing informational messages about the expiration.

9.8.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	days *number*		✓	✓	✓

z/OS

For OpenSSL only; not implemented for SystemSSL.

9.8.3 Values

number is the number of days prior to certificate expiration to begin issuing informational messages about the expiration.

Default is 15.

9.9 CERTIFICATE_REVOCATION_LIST - UBROKER configuration option

9.9.1 Description

The CERTIFICATE_REVOCATION_LIST option specifies the file / ddname of the PEM-formatted file containing the Certificate Revocation List (CRL) issued by the trusted Certificate Authority.

9.9.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	crl *file* or *ddname*	✓	✓	✓	✓

9.9.3 Values

z/OS	*ddname* is the ddname of the file containing the CRL. Allocated to the ddname must be either a sequential data set or a member of a PDS that has a variable record format. The value is used only when the SSL_IMPLEMENTATION option is set to **openssl**.
UNIX	*file* is the path name of the file containing the CRL. Relative paths are relative to the current working directory.
Windows	*file* is the path name of the file containing the CRL. Relative paths are relative to the current working directory.
IBM i	*file* is the qualified file name of the CRL file. The file name can be qualified by a library name. If not, the library list ***LIBL** is searched for the first occurrence of the file name.

9.10 CODE_PAGE - UBROKER configuration option

9.10.1 Description

The CODE_PAGE option specifies the character code page that is used to translate text data received and transmitted over the network.

The Universal Translate Table (UTT) files are used to translate between Unicode and the local single-byte code page.

9.10.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	codepage *codepage*	✓	✓	✓	✓

9.10.3 Value

codepage is the character code page that is used to translate data.

codepage references a Universal Translate Table (UTT) file provided with the product. UTT files are used to translate between Unicode and the local single-byte code page. (All UTT files end with an extension of **.utt**.)

9.10.3.1 Default

The default code page is different for different operating systems:

- ISO8859-1 (8-bit ASCII): ASCII-based operating systems*
- IBM1047 (EBCDIC): EBCDIC-based operating system*

See Character Code Pages for a complete list of character code pages provided by Stonebranch Inc. for use with Universal Agent components.

9.11 COMPONENT_BACKLOG - UBROKER configuration option

9.11.1 Description

The COMPONENT_BACKLOG option specifies the component interface backlog size for pending connection requests.

9.11.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	component_backlog *size*	✓	✓	✓	✓

9.11.3 Values

size is the component interface backlog size.

size must be greater than 0.

Default is 200.

z/OS

The system-wide default maximum backlog size for TCP/IP is **10**. The **TCPIP.PROFILE** parameter `SOMAXCONN` sets the maximum backlog size.

If you require a COMPONENT_BACKLOG *size* greater than **10**, the `SOMAXCONN` value must be increased.

9.12 COMPONENT_DIRECTORY - UBROKER configuration option

9.12.1 Description

The COMPONENT_DIRECTORY option specifies the name of the directory where component definitions are stored.

All files located in the component directory are read and processed as component definitions. The name of each file found represents the component name.

9.12.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	component_directory *directory*		✓		

9.12.3 Value

directory is the name of the directory.

Relative path names are relative to the installation directory.

Default is `/etc/universal/comp`.

9.13 COMPONENT_PORT - UBROKER configuration option

9.13.1 Description

The COMPONENT_PORT option specifies the IP port on which components communicate with the Universal Broker.

The content of messages exchanged over this connection includes, but is not necessarily limited to:

- Local configuration information
- Local component registration
- Information reported to and managed by the Universal Event Subsystem (UES)

These messages are exchanged on all supported Agent platforms; however, Windows is the only platform that relies on TCP/IP to implement the connection. That is why a separate, configurable IP port is provided for Windows. Other platforms rely on Unix domain sockets to implement the connection.

9.13.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	component_port *port*			✓	

9.13.3 Value

port is the IP port.

Valid values for *port* are:

- Numbers (for example, 7000)
- Service name (for example, **ubrokercmp**)

Default is 7987.

Note

By default, the Agent adds 100 to the value used for the Universal Broker service port (**default: 7887**). If another service already is listening on the calculated COMPONENT_PORT value, use another value to avoid collision with the existing service. Simply add component_port to the Universal Broker configuration file (if it is not already there) and set it to the desired value.

9.14 CTL_SSL_CIPHER_LIST - UBROKER configuration option

9.14.1 Description

The CTL_SSL_CIPHER_LIST option specifies one or more SSL/TLS cipher suites that are acceptable to use for network communications on the control session, which is used for component internal communication.

9.14.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	ctl_ssl_cipher_list *cipherlist*	✓	✓	✓	✓

9.14.3 Values

cipherlist is a comma-separated list of SSL/TLS cipher suites. The list should be ordered with the most preferred suite first and the least preferred suite last.

The following table identifies the list of SSL/TLS cipher suites supported for this option.

Cipher Suite	Description
AES256-GCM-SHA384	256-bit AES encryption in Galois Counter Mode, SHA-2 384-bit message digest.
AES256-SHA	256-bit AES encryption and SHA-1 message digest.
AES128-GCM-SHA256	128-bit AES encryption in Galois Counter Mode, SHA-2 256-bit message digest.
AES128-SHA	128-bit AES encryption and SHA-1 message digest.
ECDHE-RSA-AES256-GCM-SHA384	Ephemeral Elliptic Curve Diffie-Hellman Key Exchange, RSA authentication, 256-bit AES encryption in Galois Counter Mode, SHA-2 384-bit message digest.
ECDHE-ECDSA-AES256-GCM-SHA384	Ephemeral Elliptic Curve Diffie-Hellman Key Exchange, ECDSA authentication, 256-bit AES encryption in Galois Counter Mode, SHA-2 384-bit message digest.

Cipher Suite	Description
ECDHE-RSA-AES128-GCM-SHA256	Ephemeral Elliptic Curve Diffie-Hellman Key Exchange, RSA authentication, 128-bit AES encryption in Galois Counter Mode, SHA-2 256-bit message digest.
ECDHE-ECDSA-AES128-GCM-SHA256	Ephemeral Elliptic Curve Diffie-Hellman Key Exchange, ECDSA authentication, 128-bit AES encryption in Galois Counter Mode, SHA-2 256-bit message digest.
RC4-SHA	128-bit RC4 encryption and SHA-1 message digest.
RC4-MD5	128-bit RC4 encryption and MD5 message digest.
DES-CBC3-SHA	128-bit Triple-DES encryption and SHA-1 message digest.
DES-CBC-SHA	128-bit DES encryption with SHA-1 message digest. **Note** As of Universal Agent 6.7.0.0, DES-CBC-SHA is supported only on HP-UX. Additionally, any Agents on HP-UX that accept connections from, or attempt connections to, Agents on other platforms must be configured with at least one currently supported cipher suite besides DES-CBC-SHA. Therefore, those HP-UX Agents cannot be configured only with DES-CBC-SHA in their list of cipher suites.

Default is AES256-GCM-SHA384,AES256-SHA,AES128-GCM-SHA256,AES128-SHA,ECDHE-RSA-AES256-GCM-SHA384,ECDHE-ECDSA-AES256-GCM-SHA384,ECDHE-RSA-AES128-GCM-SHA256,ECDHE-ECDSA-AES128-GCM-SHA256,RC4-SHA,RC4-MD5,DES-CBC3-SHA.

9.15 CTL_SSL_CIPHER_SUITES - UBROKER configuration option

9.15.1 Description

The CTL_SSL_CIPHER_SUITES option specifies one or more **SSL/TLS 1.3 specific** cipher suites that are acceptable to use for network communications on the control session, which is used for component internal communication.

This option is specific to TLS 1.3. To configure ciphers for TLS 1.2 and earlier, see the `ctl_ssl_cipher_list` option.

9.15.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	ctl_ssl_cipher_suites *cipherlist*	✓	✓	✓	

The option is NOT currently supported on HP-UX

9.15.3 Values

cipherlist is a comma-separated list of SSL/TLS 1.3 specific cipher suites. The list should be ordered with the most preferred suite first and the least preferred suite last.

The list is in default order, with the most preferred suite first and the least preferred suite last.

Cipher Suite	Description
TLS_AES_256_GCM_SHA384	256-bit AES encryption in Galois Counter Mode, SHA-2 384-bit message digest
TLS_CHACHA20_POLY1305_SHA256	256-bit CHACHA encryption with POLY1305 message authentication, SHA-2 256-bit message digest
TLS_AES_128_GCM_SHA256	128-bit AES encryption in Galois Counter Mode, SHA-2 256-bit message digest

9.16 DB_FILE_GENERATIONS- UBroker configuration option

9.16.1 Description

The DB_FILE_GENERATIONS option enables archival of databases that fail the validation which may occur at Universal Broker startup.

The Broker initiates this validation when a previous instance of Broker failed to properly close the databases at shutdown.

Databases eligible for archival include:

- `bcomponent.db`
- `scomponent.db`
- `uems.db`
- `ues.db`

Archives are saved under the location specified by the SPOOL_DIRECTORY option, in an `.archive` folder that the Broker will create if it does not exist.

The number of archived copies for each of the eligible databases is managed by this option. When a new archival would exceed the configured value, the oldest copy is removed to accommodate the new archive.

The Broker will make every attempt to start successfully. Whether the archival is successful or not, the Broker will recreate the affected database so startup may proceed.

Only the databases listed above will be recreated. Other databases contain persistent data that will not be deleted automatically.

9.16.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line, Short Form	n/a				
Command Line, Long Form	-db_file_generations *generations*		✓	✓	
Environment Variable	UBRDBFILEGENERATIONS=*generations*		✓	✓	✓
Configuration File Keyword	db_file_generations *generations*		✓	✓	✓

9.16.3 Value

generations is the number of archive files that will be saved within the archive directory.

The maximum number of generations of log files that can be saved is 9999.

Default is 3.

Note

If the value is decreased, only the specified number of generations will be maintained. The "excess" archive files are not cleaned up immediately, but as the archive files rotate, this "excess" will be cleaned up and reused.

This change removes the `recreate_databases` option added in 7.7.0.0, to copy the functionality of the `recreate_databases` option, simply set the value of the new `db_file_generations` option to 0.

9.17 DNS_CACHE_TIMEOUT - UBROKER configuration option

9.17.1 Description

The DNS_CACHE_TIMEOUT option specifies the number of seconds that a DNS cached host entry remains valid.

When the host name resolver is asked to resolve a host name into an IP address, the host entry returned is saved in the DNS cache. The next call to resolve a host name will return the cached entry and not go back to the resolve. The cached entry is considered valid until the cache time-out period is reached.

The DNS cache provides a performance improvement as the resolution of a host name can take some time depending on the environment.

9.17.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	dns_cache_timeout *seconds*	✓	✓	✓	✓

9.17.3 Value

seconds is the number of seconds that a DNS cached host entry remains valid.

A value of **0** disables caching of host entries.

Default is 120.

9.18 ECDSA_CERTIFICATE - UBROKER configuration option

9.18.1 Description

The ECDSA_CERTIFICATE option specifies the file / ddname name of the PEM-formatted ECDSA X.509 certificate that identifies the Universal Broker.

An ECDSA X.509 certificate is required if clients require Universal Broker authentication.

> **Note**
>
> If the ECDSA_CERTIFICATE option is used, the ECDSA_PRIVATE_KEY option also is required.

9.18.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	ecdsa_certificate *ddname* or *file*	✓	✓	✓	✓

9.18.3 Values

z/OS	*ddname* is the ddname of the X.509 ECDSA certificate. The value is used only when the SSL_IMPLEMENTATION option is set to **openssl**. Allocated to the ddname must be either a sequential data set or a member of a PDS that has a variable record format.

UNIX	*file* is the path name of the X.509 ECDSA certificate file. Relative paths are relative to the current working directory.
Windows	*file* is the path name of the X.509 ECDSA certificate file. Relative paths are relative to the current working directory.
IBM i	*file* is the qualified file name of the X.509 ECDSA certificate file. The file name can be qualified by a library name. If not, the library list ***LIBL** is searched for the first occurrence of the file name.

9.19 ECDSA_PRIVATE_KEY - UBROKER configuration option

9.19.1 Description

The ECDSA_PRIVATE_KEY option specifies the location of the PEM-formatted ECDSA private key that corresponds to the X.509 certificate specified by the ECDSA_CERTIFICATE option.

Note

ECDSA_PRIVATE_KEY is required only if a certificate is specified by ECDSA_CERTIFICATE.

z/OS

ECDSA_PRIVATE_KEY is used only when the SSL_IMPLEMENTATION option is set to **openssl**.

9.19.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	ecdsa_private_key *ddname* or *file*	✓	✓	✓	✓

9.19.3 Values

z/OS	*ddname* is the ddname of the PEM-formatted ECDSA private key that corresponds to the X.509 certificates. Allocated to the ddname must be either a sequential data set or a member of a PDS that has a variable record format.
UNIX	*file* is the path of the PEM-formatted ECDSA private key file that corresponds to the X.509 certificates.
Windows	*file* is the path of the PEM-formatted ECDSA private key file that corresponds to the X.509 certificates.

IBM i	*file* is the qualified name of the PEM-formatted ECDSA private key file that corresponds to the X.509 certificates. The file name can be qualified by a library name. If not, the library list ***LIBL** is searched for the first occurrence of the file name.

9.20 ECDSA_PRIVATE_KEY_PWD - UBROKER configuration option

9.20.1 Description

The ECDSA_PRIVATE_KEY_PWD option specifies the password or passphrase for the PEM-formatted ECDSA private key specified with the ECDSA_PRIVATE_KEY option.

Note

Whether or not the password is required depends on whether or not it is required by the private key.

z/OS

ECDSA_PRIVATE_KEY_PWD is used only when the SSL_IMPLEMENTATION option is set to **openssl**.

9.20.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	ecdsa_private_key_password *password*	✓	✓	✓	✓

9.20.3 Values

password is the password for the private key.

9.21 EVENT_GENERATION - UBROKER configuration option

9.21.1 Description

The EVENT_GENERATION option specifies which types of events are to be generated and processed as persistent events by the Universal Event Subsystem (UES).

A persistent event record is saved in a Universal Enterprise Controller (UEC) database, the UES database (***uec.evm.db***), for long-term storage.

For a list of all event types for all Universal Agent components, see Event Definition Details.

9.21.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	event_generation *types*	✓	✓	✓	✓

9.21.3 Values

type specifies a comma-separated list of event types. It allows for all or a subset of all potential event message types to be selected.

Event type ranges can be specified by separating the lower and upper range values with a dash (**-**) character.

Event types can be selected for inclusion or exclusion:

- Inclusion operator is an asterisk (*****).
- Exclusion operator is (**X**) or (**x**).

9.21.4 Examples

100,101,102	Generate event types 100, 101, and 102.
100-102	Generate event types 100 through 102.
100-102,200	Generate event types 100 through 102 and 200.
*	Generate all event types.
*,X100	Generate all event types except for 100.
X*	Generate no event types.
*,X200-250,!300	Generate all event types except for 200 through 250 and 300.

Default is X* (no event types).

9.22 INSTALLATION_DIRECTORY - UBROKER configuration option

9.22.1 Description

The INSTALLATION_DIRECTORY option specifies the Universal Broker base installation directory.

Note

This is a required option.

9.22.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	installation_directory *directory*		✓	✓	

9.22.3 Value

directory is the name of the Universal Broker base installation directory.

A full path name is required.

UNIX	If Universal Broker is installed in `/opt/universal/ubroker`, specify that entire path name: `/opt/universal/ubroker`.
Windows	The default is set in the `ubroker.conf` file at installation time.

9.23 KEYSTORE_PATH - UBROKER configuration option

9.23.1 Description

The KEYSTORE_PATH option specifies the location of a remote Universal Broker key store.

if it specified path to a local keystore file, it establishes the broker as a keystore owner

9.23.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	keystore_path *path*		✓	✓	✓

9.23.3 Value

path is the remote location of the key store.

The format of *path* is: *[port@]ipaddr*

The default for *path* is 7887.

9.24 LOG_DIRECTORY - UBROKER configuration option

9.24.1 Description

The LOG_DIRECTORY option specifies the name of the directory where log files are created.

Log file creation is specified by the MESSAGE_DESTINATION option.

9.24.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	log_directory *directory*		✓	✓	

9.24.3 Value

directory is the name of the directory where log files are created.

Relative directory paths are relative to the Universal Broker installation directory. Fully qualified path names are recommended.

UNIX	**Default is** `/var/opt/universal/log`.
Windows	**Default is** `log`.

9.25 LOG_FILE_GENERATIONS - UBROKER configuration option

9.25.1 Description

The LOG_FILE_GENERATIONS option specifies the total number of log files that will be saved within the log directory.

Log file creation is specified by the MESSAGE_DESTINATION option (value = **logfile**).

9.25.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	log_file_generations *generations*		✓	✓	

9.25.3 Value

generations is the number of log files that will be saved within the log directory.

The maximum number of generations of log files that can be saved is 999.

Default is 5.

> **Note**
>
> If the value is decreased, only the specified number of generations will be maintained. The "excess" log files are not cleaned up immediately, but as the log files rotate, this "excess" will be cleaned up and reused.

9.26 LOG_FILE_LINES - UBROKER configuration option

9.26.1 Description

The LOG_FILE_LINES option specifies the total number of lines to be written to the Universal Broker log file (`unv.log`) before the log file is archived and a new log file is created.

Log file creation is specified by the MESSAGE_DESTINATION option.

9.26.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	log_file_lines *lines*		✓	✓	

9.26.3 Value

lines is the total number of lines to be written to the log file before the log file is wrapped. The average number of bytes per line is approximately 50.

The maximum number of lines that can be written is 2,147,483,647.

Default is 2000.

9.27 MAX_SSL_PROTOCOL - UBROKER configuration option

9.27.1 Description

The MAX_SSL_PROTOCOL option specifies the maximum SSL/TLS protocol level that will be negotiated and used for communications channels.

9.27.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	max_ssl_protocol *option*	✓	✓	✓	
Manager Override	n/a				

This option is NOT currently supported on HP-UX and z/OS

9.27.3 Values

option is the specification for the maximum SSL/TLS protocol level that will be supported.

- **TLS1_2**
  Maximum SSL/TLS protocol is TLS 1.2.
- **TLS1_3**
  Maximum SSL/TLS protocol is TLS 1.3.

Default is TLS1_2.

9.28 MESSAGE_DESTINATION - UBROKER configuration option

9.28.1 Description

The MESSAGE_DESTINATION option specifies the location where messages are written.

9.28.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	message_dest *destination*	✓	✓	✓	✓

9.28.3 Value

destination is the location where messages are written.

Valid values for *destination* are:

z/OS	• **logfile** Writes the messages to ddname `UNVLOG`. • **system** Writes the messages to the console as WTO messages. **Default for a console process is system.**
Windows	• **logfile** Writes the messages to a log file. The log file location is specified by the Log Directory option. The current log file name is `unv.log`. Past generation log files are named `unvNNNN.log`, where `NNNN` equals the generation number. By default, five generations are kept. • **stderr** Writes the messages to the console. **stderr** is a valid value only if Universal Broker is running as a console application. • **system** Writes the messages to the Windows Application Event Log. **Default depends on how Universal Broker is started:** • Default for a Windows Service is system. • Default for a console application is stderr.
UNIX	• **stderr** Writes the messages to the console. **stderr** is a valid value only if Universal Broker is running as a console application. • **logfile** Writes the messages to a log file. The log file location is specified by the Log Directory option. The current log file name is `unv.log`. Past generation log files are named `unvNNNN.log`, where `NNNN` equals the generation number. By default, five generations are kept. • **system** Writes the messages to the `syslog` daemon. **Default depends on how Universal Broker is started:** • Default for a console process is stderr. • Default for a daemon process is logfile.
IBM i	• **stderr** Writes the messages to the STDERR file. A batch job's STDERR file is allocated to the print file `QPRINT`. • **logfile** Writes the messages to the job's job log. • **system** Writes the messages to the system operator message queue `QSYSOPR`. The product is delivered with a value of **logfile**. If a value of **system** is preferred, you may want to reduce the number of messages written to the message queue by specifying a MESSAGE_LEVEL of **warn**. **Default is stderr.**

9.29 MESSAGE_LANGUAGE - UBROKER configuration option

9.29.1 Description

The MESSAGE_LANGUAGE option specifies the Universal Message Catalog (UMC) that is used to format messages.

There is a message catalog for each language. The first three characters of the language are used as a three-character suffix of the member name. All UMC files have a **.UMC** extension.

> **Note**
>
> Currently, the only message catalog provided is for English.

9.29.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	language *language*	✓	✓	✓	✓

9.29.3 Values

language is the name of the UMC file.

z/OS	*language* translates to a member name of the library allocated on the **UNVNLS** DD statement. Universal Broker message catalog member names start with characters **USSMC**.
UNIX	The location of the UMC file is specified by the NLS_DIRECTORY option.
IBM i	UMC file members are located in the physical source file **UNVPRD510/UNVNLS**.

Default is *ENGLISH* (UMC member USSMCENG is used).

9.30 MESSAGE_LEVEL - UBROKER configuration option

9.30.1 Description

The MESSAGE_LEVEL option specifies the level of messages to write.

9.30.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	message_level *level*	✓	✓	✓	✓

9.30.3 Values

level is the level of messages to write.

Valid values for *level* are:

- **trace**
  Writes trace messages used for diagnostic purposes (see Trace Files, below).

  > **Note**
  >
  > Use **trace** only as directed by Stonebranch, Inc. Customer Support.

- **audit**
  Writes audit, informational, warning, and error messages.
- **info**
  Writes informational, warning, and error messages.
- **warn**
  Writes warning and error messages.
- **error**
  Writes error messages only.

Default is info.

9.30.4 Trace Files

IBM i	The trace file name is `UNVTMP510/UNVTRCUBR)` .
UNIX	The trace file name depends on how it is started: • If running as a console application, the file name is `ubroker.trc` . • If running as a daemon, the file name is `ubrokerd.trc` . The trace file is created in the directory `/var/opt/universal/trace` .

Windows	• The trace file name depends on how it was started:
	  • If running as a console application, the file name is **`ubroker.trc`**.
	  • If running as a service, the file name is **`ubrsvc.trc`**.
	
	  The trace file is created in the installation directory of Universal Broker, which defaults to: **`C:\Program Files\Universal\Ubroker`**

z/OS	There are two possible destinations of the trace data:
	1. If ddname **UNVTRMDL** is defined in the UBROKER started task procedure, a sequential data set is created using the data set allocated to UNVTRMDL as a model.
	
	  The dynamically allocated trace data set name is **`#HLQ.UBR.Dyymmdd.Thhmmss`**, where:
	  • **#HLQ** is the data set name allocated on the UNVTRMDL ddname.
	  • **yymmdd** is the year, month, and day.
	  • **hhmmss** is the hour, minute, second the data set was allocated.
	
	  The amount of space allocated for trace data sets modeled after **UNVTRMDL** is based upon the TRACE_FILE_LINES configuration option and the record format of the model data set. If the model data set is fixed record format, the total amount of space measured in bytes is TRACE_FILE_LINES *LRECL. If the model data set is variable record format, the total amount of space measured in bytes is TRACE_FILE_LINES * 50 (50 is considered the average length of a trace file record).
	
	  The number of cylinders is calculated from the total amount of space in bytes. The total number of cylinders is calculated base on a total of 16 extents being allocated.
	
	  The formula is cylCount = (totalSize / 16) / 750000.
	
	  The allocation unit is set to cylinders and the primary and secondary space allocation is set to cylCount (that is, SPACE=(CYL,(cylCount,cylCount),RLSE)).
	
	2. If ddname **UNVTRMDL** is not defined in the UBROKER started task procedure, member name **UBROKER** is created in the PDS or PDS/E allocated to the UNVTRACE ddname.

Depending on the error condition being diagnosed, it is possible that the member name of the **UNVTRACE** PDS or PDS/E is not created. If this occurs, the **UNVTRMDL** ddname must be used to create a sequential data set name.

The records written to PDS and PDS/E members cannot be wrapped, so the TRACE_FILE_LINES limit has no effect on the maximum number of trace records written to the member.

9.31 MIN_SSL_PROTOCOL - UBROKER configuration option

9.31.1 Description

The MIN_SSL_PROTOCOL option specifies the minimum SSL/TLS protocol level that will be negotiated and used for communications channels.

9.31.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	min_ssl_protocol *option*	✓	✓	✓	✓

Manager Override	n/a				

Note

This option was introduced to IBM i in version 5.1.1.0.

9.31.3 Values

option is the specification for the minimum SSL/TLS protocol level that will be supported.

- **TLS1_0**
  Minimum SSL/TLS protocol is TLS 1.0.
- **TLS1_2**
  Minimum SSL/TLS protocol is TLS 1.2.
- **TLS1_3**

  TLS 1.3 is NOT currently supported on HP-UX and z/OS

  Minimum SSL/TLS protocol is TLS 1.3.

Default is TLS1_2.

9.32 MONITOR_EVENT_EXPIRATION - UBROKER configuration option

9.32.1 Description

The MONITOR_EVENT_EXPIRATION option specifies the duration of an event record, for an event used for product activity monitoring, in the Universal Broker local UES database.

If a monitoring event record is not delivered to UEC within this time period, Universal Broker will delete the record from the local UES database. (A monitoring event record is not saved in a UEC database for long-term storage.)

9.32.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	monitor_event_expiration *seconds*	✓	✓	✓	✓

9.32.3 Values

seconds is the amount of time (in seconds) that a monitoring event record will remain in the database.

Default is 600 (10 minutes).

9.33 MOUNT_POINT - UBROKER configuration option

9.33.1 Description

The MOUNT_POINT option specifies the z/OS UNIX directory in which the HFS or zFS data sets are mounted. The actual mount points will be subdirectories named after the HFS or zFS data set names being mounted.

HFS data sets are specified by either of the following:

- UNIX_DB_DATA_SET and UNIX_SPOOL_DATA_SET options.
- **UNVDB** and **UNVSPOOL** ddnames.

zFS data sets are specified only by the UNIX_DB_DATA_SET and UNIX_SPOOL_DATA_SET options. zFS data set names cannot be specified by ddname.

The mount points are created by Universal Broker if they do not exist. The z/OS UNIX permission mode is set to the value specified by the MOUNT_POINT_MODE option.

9.33.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	mount_point *directory*				✓

9.33.3 Values

directory is the z/OS UNIX directory in which the HFS or zFS data sets are mounted.

Default is /tmp.

9.34 MOUNT_POINT_MODE - UBROKER configuration option

9.34.1 Description

The MOUNT_POINT_MODE option specifies the z/OS UNIX access permission mode value with which the mounted database file system's root directory is set.

The z/OS UNIX database file system (HFS or zFS) is initialized only if the file **.inited** is not found in the root directory. When initialization is performed, **.inited** is created; initialization will not be performed again.

If you need to customize the directory ownership or permissions, define the file **.inited** in the file system's root directory; the Broker will not perform its initialization.

9.34.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	mount_point_mode *mode*				

9.34.3 Values

mode is the z/OS UNIX access permission mode value, which is a ***sum of the permission modes*** to be granted.

The following table describes each permission mode.

Mode	Description
100	User execute permission.
200	User write permission.
400	User read permission.
010	Group execute permission.
020	Group write permission.
040	Group read permission.
001	Other execute permission.
002	Other write permission.
004	Other read permission.

The format of *mode* is the same as the "change mode" USS command **chmod**. It is an octal number that specifies the permission mode value corresponding to the user, group, and other permission mode fields.

Refer to the IBM *UNIX System Services Command Reference* for complete details on the **chmod** command.

Default is 750, which specifies:

- Read-write-execute access for the user (permission modes 100, 200, and 400)
- Read-execute access for the group (permission modes 010 and 040)
- No access for other

9.35 MSG_SUPPRESSION_LIST - UBROKER configuration option

9.35.1 Description

The MSG_SUPPRESSION_LIST option specifies a list of message IDs for Universal messages to be suppressed.

The list consists of zero or more comma-separated Universal message ID numbers. For example:

- 193 - Suppress message UNV0193W only.
- 192,193 - Suppress message UNV0192W and UNV0193W.

Suppressed messages are not printed to logs or output, even if a condition arises that normally would produce the message(s).

9.35.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	msg_suppression_list *list*	✔	✔	✔	✔

9.35.3 Values

list is the list of message IDs for Universal messages to be suppressed.

9.36 NLS_DIRECTORY - UBROKER configuration option

9.36.1 Description

The NLS_DIRECTORY option specifies the directory name where the Universal Broker message catalog and code page tables are located.

9.36.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	nls_directory *directory*		✔	✔	

9.36.3 Values

directory is the name of the directory where the files are located.

Full path names are recommended.

Relative path names are relative to the **universal** installation directory.

9.36.3.1 Defaults

UNIX	Default is `/opt/universal/nls`.
Windows	Default is `\nls`.

9.37 PERSISTENT_EVENT_EXPIRATION - UBROKER configuration option

9.37.1 Description

The PERSISTENT_EVENT_EXPIRATION option specifies the duration of an event record, for an event identified as a persistent event, in the Universal Broker local UES database.

If a persistent event record is not delivered to Universal Enterprise Controller (UEC) within this time period, Universal Broker will delete the record from the local UES database. (A persistent event record is saved in a UEC database for long-term storage.)

> **Note**
>
> Events are identified as persistent events via the EVENT_GENERATION option.

9.37.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	persistent_event_expiration *seconds*	✓	✓	✓	✓

9.37.3 Values

seconds is the amount of time (in seconds) that a persistent event record will remain in the database.

Default is 172800 (2 days).

9.38 PID_FILE_DIRECTORY - UBROKER configuration option

9.38.1 Description

The PID_FILE_DIRECTORY option specifies the name of the directory that Universal Broker uses for its PID file.

The PID file is used by Universal Broker to ensure that only one instance is executing at any one time.

9.38.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	pid_file_directory *directory*		✓	✓	

9.38.3 Values

directory is the name of the directory for the PID file.

Relative path names are relative to the Universal Broker installation directory. Full path names are recommended.

9.38.3.1 Default

UNIX	/var/opt/universal
Windows	• System installs: .\Universal (relative to the install directory) • User mode installs: ubroker.pid file resides in the install directory.

Note

If the default value is changed, the PID file directory location in the Universal Broker start-up script requires the same value. See Starting Universal Broker - UNIX for details on the Broker start-up script.

9.39 PRIVATE_KEY - UBROKER configuration option

9.39.1 Description

The PRIVATE_KEY option specifies the location of the PEM-formatted RSA private key that corresponds to the X.509 certificate specified by the CERTIFICATE option.

Note

PRIVATE_KEY is required only if a certificate is specified by CERTIFICATE.

z/OS

PRIVATE_KEY is used only when the SSL_IMPLEMENTATION option is set to **openssl**.

9.39.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	private_key *ddname* or *file*	✓	✓	✓	✓

9.39.3 Values

z/OS	*ddname* is the ddname of the PEM-formatted RSA private key that corresponds to the X.509 certificates. Allocated to the ddname must be either a sequential data set or a member of a PDS that has a variable record format.
UNIX	*file* is the path of the PEM-formatted RSA private key file that corresponds to the X.509 certificates.
Windows	*file* is the path of the PEM-formatted RSA private key file that corresponds to the X.509 certificates.
IBM i	*file* is the qualified name of the PEM-formatted RSA private key file that corresponds to the X.509 certificates. The file name can be qualified by a library name. If not, the library list ***LIBL** is searched for the first occurrence of the file name.

9.40 PRIVATE_KEY_PWD - UBROKER configuration option

9.40.1 Description

The PRIVATE_KEY_PWD option specifies the password or pass phrase for the PEM-formatted RSA private key specified with the PRIVATE_KEY option.

Note

Whether or not the password is required depends on whether or not it is required by the private key.

z/OS

PRIVATE_KEY_PWD is used only when the SSL_IMPLEMENTATION option is set to **openssl**.

9.40.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	private_key_password *password*	✔	✔	✔	✔

9.40.3 Values

password is the password for the private key.

9.41 REQ_UPPS_CONN - UBROKER configuration option

9.41.1 Description

The REQ_UPPS_CONN option specifies a number of PeopleSoft connections that Universal Broker will request from a pool of connections permitted by your Universal Agent license.

For agents that connect to a Universal Controller, the Universal Broker receives the number of connections granted. This number may be less than the requested number of connections, depending on the number of connections already granted to other agents.

For purposes of license enforcement, a connection is a named object that represents a network connection to a PeopleSoft instance. While connections are unique to each agent, all agents cannot exceed the total number of licensed connections.

Note

Connections are consumed only when an agent connects to a 6.9.0.0 or later Universal Controller. Universal Connector for PeopleSoft execution will continue to be enforced by local licenses when either:

1. An agent does not connect to a Controller.
2. An agent connects to a Controller earlier than 6.9.0.0.

Local licensing is planned for deprecation; agent releases later than 6.9.0.0 may be expected to obtain license information via a Universal Controller connection.

9.41.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line, Long Form	-req_upps_conn *connections*		✓	✓	
Environment Variable	UBRREQUPPSCONN=*connections*		✓	✓	
Configuration File Keyword	req_upps_conn *connections*		✓	✓	

9.41.3 Values

connections is the number of PeopleSoft connections that Universal Broker will request.

Default is 0.

9.42 REQ_USAP_CONN - UBROKER configuration option

9.42.1 Description

The REQ_USAP_CONN option specifies a number of SAP connections that Universal Broker will request from a pool of connections permitted by your Universal Agent license.

For agents that connect to a Universal Controller, the Universal Broker receives the number of connections granted. This number may be less than the requested number of connections, depending on the number of connections already granted to other agents.

For purposes of license enforcement, a connection is a named object that represents a network connection to an SAP instance. While connections are unique to each agent, all agents cannot exceed the total number of licensed connections.

Note

Connections are consumed only when an agent connects to a 6.9.0.0 or later Universal Controller. Universal Connector for SAP execution will continue to be enforced by local licenses when either:

1. An agent does not connect to a Controller.
2. An agent connects to a Controller earlier than 6.9.0.0.

Local licensing is planned for deprecation; agent releases later than 6.9.0.0 may be expected to obtain license information via a Universal Controller connection.

9.42.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line, Long Form	-req_usap_conn *connections*		✓	✓	✓
Environment Variable	UBRREQUSAPCONN=*connections*		✓	✓	✓
Configuration File Keyword	req_usap_conn *connections*		✓	✓	✓

9.42.3 Values

connections is the number of SAP connections that Universal Broker will request.

Default is 0.

9.43 REQUIRE_SSL - UBROKER configuration option

9.43.1 Description

The REQUIRE_SSL option specifies whether or not Universal Broker will enforce the use of SSL/TLS connections by the clients (managers) of Universal Command Server and Universal Data Mover Server, from which Universal Broker inherits REQUIRE_SSL.

So, any client that tries to connect to a server using UNVv2 legacy protocol will be rejected if REQUIRE_SSL is enabled.

9.43.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	require_ssl *option*	✓	✓	✓	✓

Note

This option was introduced to IBM i in version 5.1.1.0.

9.43.3 Values

option is the specification for whether or not Universal Broker will enforce the use of SSL/TLS connections.

Valid values for *option* are:

- **yes**
  Enforce the use of SSL/TLS.
- **no**
  Do not enforce the use of SSL/TLS.

Default is no.

9.44 RETRY_MAX_COMP - UBROKER configuration option

9.44.1 Description

The RETRY_MAX_COMP option specifies whether or not the Start Component request is retryable when the maximum number of components are running.

When it is retryable, the manager component that sent the Start Component request will resend the request after waiting a number of seconds. When it is not retryable, the manager component will end with an error.

Refer to the Universal Broker RUNNING_MAX configuration option, and the RUNNING_MAXIMUM component definition option for selected server components, for specifying the maximum number of running components.

9.44.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	retry_max_comp *option*		✓	✓	✓

9.44.3 Values

option is the specification for whether or not the Start Component request is retryable.

Valid values for *option* are:

- **yes**
  Start Component request is retryable.
- **no**
  Start Component request is not retryable.

Default is yes.

9.45 RUNNING_MAX - UBROKER configuration option

9.45.1 Description

The RUNNING_MAX option specifies the maximum number of components that can run simultaneously.

If this maximum is reached, any command received to start a component is rejected.

9.45.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	running_max *maximum*	✓	✓	✓	✓

9.45.3 Values

maximum is the maximum number of components that can be run simultaneously.

> **Note**
>
> For z/OS systems, *maximum* must be less than 1,000; otherwise, the UBroker service cannot be started.

Default is 100.

9.46 SAF_KEY_RING - UBROKER configuration option

9.46.1 Description

The SAF_KEY_RING option specifies the SAF (RACF is a SAF implementation) certificate key ring name that the Universal Broker started task should used for its certificate.

The key ring must be associated with the user profile with which the Universal Broker started task executes.

> **Note**
>
> SAF_KEY_RING is required if the SSL_IMPLEMENTATION option is set to **system**.

9.46.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	saf_key_ring *name*				

9.46.3 Values

name is the name of the SAF certificate key ring.

9.47 SAF_KEY_RING_LABEL - UBROKER configuration option

9.47.1 Description

The SAF_KEY_RING_LABEL option specifies the label of the certificate in the SAF (RACF is a SAF implementation) certificate key ring that the Universal Broker started task should use for its certificate.

(The key ring is specified by the SAF_KEY_RING option.)

9.47.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS

Configuration File Keyword	saf_key_ring_label *label*				✓

9.47.3 Values

label is the label of the SAF certificate key ring.

Default is the default certificate in the key ring.

9.48 SERVICE_BACKLOG - UBROKER configuration option

9.48.1 Description

The SERVICE_BACKLOG option specifies the service interface backlog size for pending connection requests.

9.48.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	service_backlog *size*	✓	✓	✓	✓

9.48.3 Values

size is the service interface backlog size.

size must be greater than 0.

Default is 100.

z/OS

The system-wide default maximum backlog size for TCP/IP is **10**. The **TCPIP.PROFILE** parameter `SOMAXCONN` sets the maximum backlog size.

If you require a SERVICE_BACKLOG *size* greater than **10**, the `SOMAXCONN` value must be increased.

9.49 SERVICE_IP_ADDRESS - UBROKER configuration option

9.49.1 Description

The SERVICE_IP_ADDRESS option specifies the IP interface on which to accept network connection requests.

SERVICE_IP_ADDRESS is useful only if the system has multiple IP interfaces.

- If the system has multiple interfaces and SERVICE_IP_ADDRESS is not used, connection requests are accepted on all interfaces defined on the system.
- If the system has only one interface, do not use SERVICE_IP_ADDRESS.

9.49.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	service_ip_address *ipaddress*	✓	✓	✓	✓

9.49.3 Values

ipaddress is the IP address on which to accept network connection requests.

Valid values for *ipaddress* are:

- Dotted numeric format (for example, *20.30.40.50*)
- Domain name format (for example, *myinterface*).

Note

An asterisk (*) specifies all interfaces.

Default is *.

9.50 SERVICE_PORT - UBROKER configuration option

9.50.1 Description

The SERVICE_PORT option specifies IP port on which to accept network connection requests.

9.50.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	service_port *port*	✓	✓	✓	✓

9.50.3 Values

port is the IP port on which to accept network connection requests.

Valid values for *port* are:

- Numeric value (for example, 7000)
- Service name (for example, **ubroker**)

Default is 7887.

> **Note**
>
> It is recommended that you use the default port, 7887, if possible.

9.51 SHARED_MOUNT_POINT - UBROKER configuration option

9.51.1 Description

The SHARED_MOUNT_POINT option specifies the z/OS UNIX directory in which the zFS data sets are shared across members of a Sysplex.

- In a Sysplex configuration, the specified directory must be a directory shared across the Sysplex.
- In a non-Sysplex configuration, the directory should be the same as that specified for the MOUNT_POINT configuration option.

zFS data sets are specified only by the UNIX_DB_DATA_SET and UNIX_SPOOL_DATA_SET options. zFS data set names cannot be specified by ddname.

9.51.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	shared_mount_point *directory*				✓

9.51.3 Values

directory is the z/OS UNIX directory in which the zFS data sets are shared..

Default is the same as the directory specified for MOUNT_POINT**.**

9.52 SHARED_MOUNT_POINT_MODE - UBROKER configuration option

9.52.1 Description

The SHARED_MOUNT_POINT_MODE option specifies the z/OS UNIX access permission mode value with which the shared mounted database file system's root directory is set.

The zFS UNIX database file system is initialized only if the file **.inited** is not found in the root directory. When initialization is performed, **.inited** is created; initialization will not be performed again.

If you need to customize the directory ownership or permissions, define the file **.inited** in the file system's root directory; the Broker will not perform its initialization.

9.52.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	shared_mount_point_mode *mode*				

9.52.3 Values

mode is the z/OS UNIX access permission mode value, which is a ***sum of the permission modes*** to be granted.

The following table describes each permission mode.

Mode	Description
100	User execute permission.
200	User write permission.
400	User read permission.
010	Group execute permission.
020	Group write permission.
040	Group read permission.
001	Other execute permission.
002	Other write permission.

Mode	Description
004	Other read permission.

The format of *mode* is the same as the "change mode" USS command **chmod**. It is an octal number that specifies the permission mode value corresponding to the user, group, and other permission mode fields.

Refer to the IBM *UNIX System Services Command Reference* for complete details on the **chmod** command.

Default is the value specified for MOUNT_POINT_MODE**.**

9.53 SMF_EXIT_LOAD_LIBRARY - UBROKER configuration option

9.53.1 Description

The SMF_EXIT_LOAD_LIBRARY option specifies a cataloged data set from which the SMF exit routine **UNVACTRT** is loaded and dynamically installed at exit point `SYSSTC.IEFACTRT`.

> **Note**
>
> The UNVACTRT exit should reside in LPA, the LNKLST concatenation, or the nucleus. Do not use the DSNAME keyword when defining the exit in PROGxx; the system will not be able to load the exit when restarting SMF.

If SMF_EXIT_LOAD_LIBRARY is not specified, the exit routine is not dynamically installed. It then must be installed prior to the Universal Broker address space starting with an alternative method. (See the z/OS Configuration - SMF Exits for alternative methods.)

The exit routine is deleted when last the Universal Broker address space running is stopped. If multiple Universal Broker address spaces are running, the last Universal Broker to stop removes the exit routine.

SMF_EXIT_LOAD_LIBRARY is required if the UCMD_STC_SUPPORT option is set to **yes**.

9.53.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	smf_exit_load_library *dsn*				✓

9.53.3 Values

dsn is the cataloged data set from which the SMF exit routine is loaded and installed.

9.54 SPOOL_DIRECTORY - UBROKER configuration option

9.54.1 Description

The SPOOL_DIRECTORY option specifies the directory name that Universal Broker uses for its spool database files.
The Universal Broker spool files should not require a large amount of disk space; two or three MB should be sufficient.

9.54.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	spool_directory *directory*		✓	✓	

9.54.3 Values

directory is the name of the directory for spool database files.
Relative path names are relative to the Universal Broker installation directory. Full path names are recommended.

9.54.3.1 Defaults

Windows	Default is `C:\Program Files\Universal\spool`.
UNIX	Default is `/var/opt/universal/spool`.

9.55 SSL_IMPLEMENTATION - UBROKER configuration option

9.55.1 Description

The SSL_IMPLEMENTATION option specifies the SSL/TLS implementation to be used for network communications.

9.55.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	ssl_implementation *option*				✓

9.55.3 Values

option is the SSL/TLS implementation to be used.

Valid values for option are:

- **openssl**
  OpenSSL SSL library is used for the SSL/TLS protocol.
- **system**
  z/OS System SSL library is used for the SSL/TLS protocol. The z/OS System SSL library has installation and configuration prerequisites. (See the Configuration of zOS System SSL for a description of the prerequisites before using System SSL.)

Default is openssl.

9.56 SYSPLEX_ROLE - UBROKER configuration option

9.56.1 Description

The SYSPLEX_ROLE option specifies the role that Universal Broker will perform in a Sysplex configuration.

9.56.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	sysplex_role *role*				✓

9.56.3 Values

role is the role that Universal Broker will perform in a Sysplex configuration.

Valid values are:

- none
- primary
- secondary

Default is none.

9.57 SYSTEM_ID - UBROKER configuration option

9.57.1 Description

The SYSTEM_ID option uniquely identifies the Universal Broker.

- If SYSTEM_ID is not used to identify the Universal Broker, the default (a blank value) is used. If there are more than one Universal Brokers running on an O/S image, only one can use the default. SYSTEM_ID must be used to identify all of the other Universal Brokers.
- If SYSTEM_ID is used to identify the Universal Broker, all of its Manager jobs must include the SYSTEM_ID option to identify the Universal Broker.

9.57.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	system_id *ID*				✓

9.57.3 Values

ID is the system identifier of the local Universal Broker (1 to 8 characters in length).

Valid values for *ID* are:.

- First character must be alphabetic.
- All subsequent characters must be alphabetic or numeric.

Default is a blank value.

9.58 TCP_RECV_BUFFER - UBROKER configuration option

9.58.1 Description

The TCP_RECV_BUFFER option specifies the size of the TCP receive buffer used for socket connections.

TCP_RECV_BUFFER provides the ability to tune TCP data transfer performance between a manager component and a server component started by the Universal Broker. See Network Data Transfer Tuning for a description on using this option to tune data transfer performance.

Universal Broker will instruct TCP to set the socket receive buffer to the specified size. The actual TCP receive buffer size used is determined by TCP based on its configuration. The TCP configuration can limit the buffer size to a maximum value for example.

9.58.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	tcp_recv_buffer *size* [*unit*]		✓	✓	✓

9.58.3 Values

The *size* value specifies the requested size of the TCP receive buffer.

The *size* value is specified in units of *unit*. Possible *unit* values are

Unit	Description
B	Bytes (the default)
K	Kilobytes (1024 bytes)
M	Megabytes (1048576 bytes)
G	Gigabytes (1073741824 bytes)

The *unit* value is case insensitive.

The maximum supported buffer size is 1G. The default is 0B.

9.59 TCP_SEND_BUFFER - UBROKER configuration option

9.59.1 Description

The TCP_SEND_BUFFER option specifies the size of the TCP send buffer used for socket connections.

TCP_SEND_BUFFER provides the ability to tune TCP data transfer performance between a manager component and a server component started by the Universal Broker. See Network Data Transfer Tuning for a description on using this option to tune data transfer performance.

Universal Broker will instruct TCP to set the socket send buffer to the specified size. The actual TCP send buffer size used is determined by TCP based on its configuration. The TCP configuration can limit the buffer size to a maximum value for example.

9.59.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	tcp_send_buffer *size* [*unit*]		✓	✓	✓

9.59.3 Values

The *size* value specifies the requested size of the TCP send buffer.

The *size* value is specified in units of *unit*. Possible *unit* values are

Unit	Description
B	Bytes (the default)

Unit	Description
K	Kilobytes (1024 bytes)
M	Megabytes (1048576 bytes)
G	Gigabytes (1073741824 bytes)

The *unit* value is case insensitive.

The maximum supported buffer size is 1G. The default is 0B.

9.60 TMP_DIRECTORY - UBROKER configuration option

9.60.1 Description

The TMP_DIRECTORY option specifies the directory that the Universal Broker uses for temporary files.

z/OS

TMP_DIRECTORY specifies the name of a z/OS UNIX directory.

The amount of space required for the temporary directory is small. Most of the files are IPC pipes used for Broker and Server IPC.

9.60.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	tmp_directory *directory*		✔	✔	✔

9.60.3 Values

directory is the name of the directory.

A fully qualified path name is recommended.

9.60.3.1 Defaults

UNIX	Default is `/var/opt/universal/tmp`.
Windows	Default is `..\tmp`.

z/OS	Default is `/tmp`.

9.61 TRACE_DIRECTORY - UBROKER configuration option

9.61.1 Description

The TRACE_DIRECTORY option specifies the directory that the Universal Broker uses for trace files.

9.61.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	trace_directory *directory*		✓	✓	

9.61.3 Values

directory is the name of the directory for trace files.

Relative path names are relative to the Universal Broker installation directory. Full path names are recommended.

9.61.3.1 Defaults

Windows	Default is `C:\Program Files\Universal\UBroker`.
UNIX	Default is `/var/opt/universal/trace`.

9.62 TRACE_FILE_LINES - UBROKER configuration option

9.62.1 Description

The TRACE_FILE_LINES option specifies the maximum number of lines to write to the trace file.

A trace file is generated when the MESSAGE_LEVEL option is set to **trace**. The trace file will wrap around when the maximum number of lines has been reached and start writing trace entries after the trace header lines.

(The average size of a trace file line is 50 characters.)

IBM i

Trace file records are 366 bytes long.

9.62.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	trace_file_lines *lines*	✓	✓	✓	✓

9.62.3 Values

lines is the maximum number of lines to write to the trace file.

Default = 500,000.

Note

If space is limited in the trace file directory, set *lines* to a smaller value.

IBM i

If space is limited in the trace file ASP (ASP in which the **UNVTMP511** library is located), set the default to a smaller value. If a larger value is required, either create or change the maximum number of records allowed in the physical file **UNVTMP511/UNVTRCUBR** and increase this setting. The largest value allowed without increasing the number of records allowed is **509000**.

9.63 TRACE_TABLE - UBROKER configuration option

9.63.1 Description

The TRACE_TABLE option specifies the size of a wrap-around trace table maintained in memory.

The trace table is written to a file / data set when the program ends under the conditions specified in this option. Tracing is activated, and a trace file is generated, when the MESSAGE_LEVEL option is set to **trace**.

9.63.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS

Configuration File Keyword	trace_table *size, condition*	✓	✓	✓	✓

9.63.3 Values

size is the size (in bytes) of the trace table.

The size can be suffixed with either of the following characters:

- **M** indicates that the size is specified in megabytes
- **K** indicates that the size is specified in kilobytes

For example, **50M** indicates that 50 X 1,048,576 bytes of memory is allocated for the trace table.

> **Note**
>
> If *size* is **0**, the trace table is not used.

condition is the condition under which the trace table is written.

Possible values for *condition* are:

- **error**
  Write the trace table if the program ends with a non-zero exit code.
- **always**
  Write the trace table when the program ends regardless of the exit code.
- **never**
  Never write the trace table.

9.64 UCMD_PATH - UBROKER configuration option

9.64.1 Description

The UCMD_PATH option specifies the absolute path to an external link that resides on the z/OS UNIX file system.

This option is provided to support disabling the UID 0 requirement for the Universal Broker started task.

If a value is provided for this option, the link must be manually created prior to Universal Broker start-up, it must point to 'UCMD', and it must be owned by UID 0.

9.64.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	ucmd_path *directory*				✓

9.64.3 Values

directory is the absolute path to an external link that resides on the z/OS UNIX file system.

If this option is not specified, the Universal Broker started task will format the link's name and attempt to create it in the location specified by the TMP_DIRECTORY configuration option.

9.65 UCMD_STC_SUPPORT - UBROKER configuration option

9.65.1 Description

The UCMD_STC_SUPPORT option specifies whether or not the Universal Broker establishes the environment to support Universal Command start task requests.

9.65.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	ucmd_stc_support *option*				✓

9.65.3 Values

option is the specification for whether or not the Universal Broker establishes the environment.

Valid values for *option* are:

- **yes**
  Universal Broker establishes the environment.
- **no**
  Universal Broker does not establish the environment.

> **Note**
>
> If the value for *option* = **no**, Universal Command will not support the execution of started tasks.

The environment support for Universal Command started tasks consists of installing SMF exit routine **UNVACTRT** at exit point `SYSSTC.IEFACTRT` and a small amount of CSA storage for address space communication.

Default is yes.

9.66 UCTL_PATH - UBROKER configuration option

9.66.1 Description

The UCTL_PATH option specifies the absolute path to an external link that resides on the z/OS UNIX file system.

This option is provided to support disabling the UID 0 requirement for the Universal Broker started task.

If a value is provided for this option, the link must be manually created prior to Universal Broker start-up, it must point to 'UCTL', and it must be owned by UID 0.

9.66.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	uctl_path *directory*				✓

9.66.3 Values

directory is the absolute path to an external link that resides on the z/OS UNIX file system.

If this option is not specified, the Universal Broker started task will format the link's name and attempt to create it in the location specified by the TMP_DIRECTORY configuration option.

9.67 UNIX_DB_DATA_SET - UBROKER configuration option

9.67.1 Description

The UNIX_DB_DATA_SET option specifies the HFS or zFS data set used for the Universal Broker's databases. The data set can be mounted prior to starting the Broker. If not, the Broker will mount the data set at a specified mount point derived from the MOUNT_POINT option.

UNIX_DB_DATA_SET is the only way to specify a zFS data set. HFS data sets can be allocated in the Broker's started task procedure as ddname **UNVDB**. zFS data sets cannot be allocated on a ddname.

> **Note**
>
> When using a zFS data set, the **UNVDB** ddname statement in the Broker's started task procedure should be removed.

9.67.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	unix_db_data_set *DSN*				✓

9.67.3 Values

DSN is the HFS or zFS data set used for the databases.

9.68 UNIX_SPOOL_DATA_SET - UBROKER configuration option

9.68.1 Description

The UNIX_SPOOL_DATA_SET option specifies the HFS or zFS data set used for the Universal Broker's spool. The data set can be mounted prior to starting the Broker. If not, the Broker will mount the data set at a specified mount point derived from the MOUNT_POINT option.

UNIX_SPOOL_DATA_SET is the only way to specify a zFS data set. HFS data sets can be allocated in the Broker's started task procedure as ddname **UNVSPOOL**. zFS data sets cannot be allocated on a ddname.

> **Note**
>
> When using a zFS data set, the **UNVSPOOL** ddname statement in the Broker's started task procedure should be removed.

9.68.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	unix_spool_data_set *DSN*				✓

9.68.3 Values

DSN is the HFS or zFS data set used for the spool.

9.69 USAP_PATH - UBROKER configuration option

9.69.1 Description

The USAP_PATH option specifies the absolute path to an external link that resides on the z/OS UNIX file system.

This option is provided to support disabling the UID 0 requirement for the Universal Broker started task.

If a value is provided for this option, the link must be manually created prior to Universal Broker start-up, it must point to 'USAP', and it must be owned by UID 0.

9.69.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	usap_path *directory*				✓

9.69.3 Values

directory is the absolute path to an external link that resides on the z/OS UNIX file system.

If this option is not specified, the Universal Broker started task will format the link's name and attempt to create it in the location specified by the TMP_DIRECTORY configuration option.

9.70 WORKING_DIRECTORY - UBROKER configuration option

9.70.1 Description

The WORKING_DIRECTORY option specifies the directory name that the Universal Broker uses as its working directory.

WORKING_DIRECTORY may be of value if you want the Universal Broker daemon to use a working directory other than the default. Ideally, daemons should use the root directory as their working directory. This prevents the need to stop the daemon should a file system require unmounting.

9.70.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	working_directory *directory*		✓	✓	

9.70.3 Values

directory is the name of the working directory.

Relative path names are relative to the Universal Broker installation directory. Full path names are recommended.

9.70.3.1 Defaults

Windows	Default is Universal Broker installation directory.
UNIX	Default is startup directory.

9.71 UAG_AUTOSTART - UBROKER Configuration Option

9.71.1 Description

The UAG_AUTOSTART option specifies a value to override the AUTOMATICALLY_START option in the UAG Server component definition.

9.71.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line	-uag_autostart *option*		✓	✓	
Environment Variable	UAGAUTOSTART=*option*		✓		

9.71.3 Values

option is the specification for whether or not the UAG Server will start automatically at Universal Broker start-up, regardless of the value specified in the AUTOMATICALLY_START UAG Server component definition.

Valid values for *option* are:

- **yes**
  UAG Server will automatically start at Universal Broker start-up.
- **no**
  UAG Server will NOT start automatically at Universal Broker start-up.

9.71.4 Default

There is no default. If this option is not specified, the AUTOMATICALLY_START option in the UAG Server component definition controls start-up behavior.

9.71.5 Example

ubroker -uag_autostart no

9.72 UEM_AUTOSTART - UBROKER Configuration Option

9.72.1 Description

The UEM_AUTOSTART option specifies a value to override the AUTOMATICALLY_START option value in the UEM Server component definition.

9.72.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line	-uem_autostart *option*		✓	✓	
Environment Variable	UEMAUTOSTART=*option*		✓		

9.72.3 Values

option is the specification for whether or not the UEM Server will start automatically at Universal Broker start-up, regardless of the value specified in the UEM Server component definition.

Valid values for *option* are:

- **yes**
  UEM Server will automatically start at Universal Broker start-up.
- **no**
  UEM Server will NOT start automatically at Universal Broker start-up.

9.72.4 Default

There is no default.

If this option is not specified, the AUTOMATICALLY_START option in the UEM Server component definition controls start-up behavior.

9.72.5 Example

ubroker -uem_autostart no

9.73 OMS_AUTOSTART - UBROKER Configuration Option

9.73.1 Description

The OMS_AUTOSTART option specifies a value to override the AUTOMATICALLY_START option value in the OMS Server component definition.

9.73.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line	-oms_autostart *option*		✓	✓	
Environment Variable	OMSAUTOSTART=*option*		✓		

9.73.3 Values

option is the specification for whether or not the OMS Server will start automatically at Universal Broker start-up, regardless of the value specified in the OMS Server component definition.

Valid values for *option* are:

- **yes**
  OMS Server will automatically start at Universal Broker start-up.
- **no**
  OMS Server will NOT start automatically at Universal Broker start-up.

9.73.4 Default

There is no default.

If this option is not specified, the AUTOMATICALLY_START option in the OMS Server component definition controls start-up behavior.

9.73.5 Example

ubroker -oms_autostart no

9.74 UAG_AGENT_CLUSTERS - UBROKER Configuration Option

9.74.1 Description

The UAG_AGENT_CLUSTERS option overrides the list of Universal Agent clusters to join automatically as specified in the AGENT_CLUSTERS UAG configuration option.

9.74.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line	-uag_agent_clusters *list*		✓	✓	
Environment Variable	UAGAGENTCLUSTERS=*list*		✓		

9.74.3 Values

list is a comma-separated list of agent clusters.

Default is 'Opswise - Default Linux/Unix Cluster, Opswise - Default Windows Cluster'.

> **Note**
>
> For information on creating agent clusters and assigning agents to agent clusters, see Agent Clusters.

9.74.3.1 Example

ubroker -uag_agent_clusters 'Cluster 1, Cluster 2'

9.75 UAG_BUSINESS_SERVICES - UBROKER Configuration Option

9.75.1 Description

The UAG_BUSINESS_SERVICES option overrides the list of business services to join automatically as specified in the BUSINESS_SERVICES UAG configuration option

9.75.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line	-uag_business_services list		✓	✓	✓
Environment Variable	UAGBUSINESSSERVICES=list		✓	✓	✓

9.75.3 Values

list is a comma-separated list of agent clusters.

Default is 'Opswise - Default Linux/Unix Cluster, Opswise - Default Windows Cluster'.

> **Note**
>
> For information on creating business services and assigning records to business services, see Business Services.

9.75.3.1 Example

ubroker -uag_business_services ‘Collections, Fees'

9.76 UAG_EXTENSION_ACCEPT_LIST - UBROKER Configuration Option

9.76.1 Description

The UAG_EXTENSION_ACCEPT_LIST option specifies a value to override the EXTENSION_ACCEPT_LIST option of Universal Automation Center Agent (UAG).

It specifies one or more comma-separated names of Universal Extensions that the Agent will accept via auto-deployment from Universal Controller.

9.76.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line	-uag_extension_accept_list *list*		✓	✓	
Environment Variable	UAGEXTENSIONACCEPTLIST=*list*		✓		

9.76.3 Values

list is the list of Universal Extensions that the Agent will accept.

- A single value of ***** indicates that all extensions are accepted.
- A single value of **none** indicates that no extensions are accepted.

9.77 UAG_EXTENSION_CANCEL_TIMEOUT - UBROKER configuration option

9.77.1 Description

The UAG_EXTENSION__CANCEL_TIMEOUT option specifies a value to override the EXTENSION_CANCEL_TIMEOUT option of Universal Automation Center Agent (UAG).

It specifies the length of time that a Universal Extension task is given to complete its response to a CANCEL request received from Universal Controller.

If the task fails to finish its own termination processing within the specified timeout period, UAG Server will forcefully terminate the task.

9.77.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	extension_cancel_timeout *time*[**s**|**m**|**h**|**d**]		✓	✓	

9.77.3 Values

time must be numeric, but a one-letter suffix is accepted to specify a time unit of (**s**)econds, (**m**)inutes, (**h**)ours, or (**d**)ays. If no time unit is specified, the default is seconds.

The following maximums are enforced:

- 2147483647 or 2147483647s
- 35791394m
- 596523h
- 24855d

Minute, hour, and day maximums are set to ensure that their value, when represented as a number of seconds, does not exceed 2147483647.

Default is 10 seconds.

9.78 UAG_EXTENSION_DEPLOY_ON_REGISTRATION - UBROKER Configuration Option

9.78.1 Description

The UAG_EXTENSION_DEPLOY_ON_REGISTRATION option specifies a value to override the EXTENSION_DEPLOY_ON_REGISTRATION option of Universal Automation Center Agent (UAG).

It controls Extension deployment behavior from Universal Controller.

- If **yes**, the Controller will preemptively deploy all extensions acceptable by UAG.

- If **no**, the Controller will only send Extension modules as needed (on demand).

9.78.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line	-uag_extension_deploy_on_registration *option*		✓	✓	
Environment Variable	UAGEXTENSIONDEPLOYONREGISTRATION=*option*		✓		

9.78.3 Values

option specifies when the Controller will send Extensions to Universal Agent..

Valid values for *option* are:

- **yes**
  Universal Controller will preemptively deploy all extensions acceptable by UAG
- **no**
  Universal Controller will only send Extension modules as needed (on demand).

9.79 UAG_EXTENSION_PYTHON_LIST - UBROKER Configuration Option

9.79.1 Description

The UAG_EXTENSION_PYTHON_LIST option specifies a value to override the EXTENSION_PYTHON_LIST option of Universal Automation Center Agent (UAG).

It specifies a comma-separated list of zero or more Python locations. Each item in the list is expected to contain a complete path to a Python executable.

9.79.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line	-uag_extension_python_list *list*		✓	✓	
Environment Variable	UAGEXTENSIONPYTHONLIST=*list*		✓		

9.79.3 Values

list is the list of Python locations.

9.80 UAG_NETNAME - UBROKER Configuration Option

9.80.1 Description

The UAG_NETNAME option specifies a value to override the network ID of Universal Automation Center Agent (UAG) in the NETNAME UAG configuration option.

9.80.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line	-uag_netname *name*		✓	✓	
Environment Variable	UAGNETNAME=*name*		✓		

9.80.3 Values

name is the network ID of UAG. It is a case sensitive value.

9.80.4 Default

The default is **OPSAUTOCONF**.

9.80.5 Notes

If *name* is **OPSAUTOCONF** (the default), ID is requested from the Universal Controller after the first successful connection attempt.

To ensure that a UAG Server uses a Universal Controller-assigned network ID, specify **OPSAUTOCONF** for this option.

If the UAG Server is configured to use a static UAG_NETNAME value, this option will override that value and cause the Agent to receive a Universal Controller-assigned network ID.

If the UAG Server already is configured to receive a network ID from the Controller (the default), and the Agent already has connected to the Controller, the value previously assigned to Agent is used.

If *name* is **OPSAUTOCONF** (the default) and the UAG Server already has connected to a Universal Controller, the qname value that holds the UAG Server's assigned netname must be deleted before it can be overridden with this option.

9.80.6 Examples

ubroker -uag_netname OPSAUTOCONF

ubroker -uag_netname AGNT099

9.81 UAG_OMS_SERVERS - UBROKER Configuration Option

9.81.1 Description

The UAG_OMS_SERVERS option specifies values to override the Universal Agent's port and network address of the Universal Message Service (OMS) server(s) used for network communication, as specified in the OMS_SERVERS UAG configuration option.

9.81.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line	-uag_oms_servers *port1@host1*[,*port2@host2*...]		✓	✓	
Environment Variable	UAGOMSSERVERS= *port1@host1*[,*port2@host2*...]		✓		

9.81.3 Values

port is the TCP port on which the OMS server is listening. The default is **7878**.

host is the host name or IP address of the OMS server.

9.81.3.1 Example

ubroker -uag_oms_servers 7878@hal90000.com;7878@sal90000.com

9.81.3.2 OMS Failover Configuration

OMS servers may be deployed in a High Availability (HA) cluster, in which there are two or more OMS servers. An HA cluster has one active OMS server and one or more inactive OMS servers. When the active OMS server fails, one of the inactive OMS servers becomes the new active OMS server. UAG will automatically failover to the new active member of the HA cluster.

An OMS server HA cluster is specified as a comma-separated list of OMS servers, where each OMS server specified in the list is a member of the same HA cluster. UAG will connect to the first OMS server in the list. If that OMS server connection fails, UAG will attempt to connect to the next OMS server in the list, and so on, until it has successfully connected. The first OMS server in the list is should be considered the primary OMS server and subsequent OMS servers in the list are backup OMS servers.

Do not specify OMS servers in the comma-separated list that are not part of the same HA cluster, otherwise OMS messages may be lost in the case of fail over.

9.82 UAG_PROCESS_CANCEL_TIMEOUT - UBROKER configuration option

9.82.1 Description

The UAG_PROCESS__CANCEL_TIMEOUT option specifies a value to override the PROCESS_CANCEL_TIMEOUT option of Universal Automation Center Agent (UAG).

It specifies the length of time that an OS task is given to complete its response to a CANCEL request received from Universal Controller.

If the task fails to finish its own termination processing within the specified timeout period, UAG Server will forcefully terminate the task.

9.82.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	process_cancel_timeout *time*[**s**|**m**|**h**|**d**]				
Environment Variable	UAGPRCCANCELTIMEOUT=*time*[**s**|**m**|**h**|**d**]				

9.82.3 Values

time must be numeric, but a one-letter suffix is accepted to specify a time unit of (**s**)econds, (**m**)inutes, (**h**)ours, or (**d**)ays. If no time unit is specified, the default is seconds.

The following maximums are enforced:

- 2147483647 or 2147483647s
- 35791394m
- 596523h
- 24855d

Minute, hour, and day maximums are set to ensure that their value, when represented as a number of seconds, does not exceed 2147483647.

Default is 10 seconds.

9.83 UAG_TRANSIENT - UBROKER Configuration Option

9.83.1 Description

The UAG_TRANSIENT option specifies a value to override the value of the TRANSIENT option in the UAG configuration file.

9.83.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Command Line	-uag_transient *option*		✓	✓	
Environment Variable	UAGTRANSIENT=*option*		✓		

9.83.3 Values

option indicates whether or not the OMS Server will start automatically at Universal Broker start-up, regardless of the value specified in...

Valid values are:

- yes
  OMS Server will automatically start at Universal Broker startup, regardless of the value specified in the component definition file.
- no
  OMS Server will not start when the Universal Broker starts, regardless of the value specified in the component definition file.

9.83.3.1 Default

If this option is omitted, the value specified for the auto_start option in the OMS Server component definition file controls startup behavior.

9.83.3.2 Example

ubroker -uag_transient no

10 Universal Broker Component Definition Options

10.1 Universal Broker Component Definition Options

This page provides links, in the Component Definition Options table, below, to detailed information about the options that comprise the Universal Agent component definitions provided to the Universal Broker for all server components.

The options are listed alphabetically, without regard to any specific operating system.

Information on how component definitions are used is documented in the operating system-specific pages of this Universal Broker Reference Guide.

To see which options comprise the component definition for each server component, see:

- Universal Automation Center Agent
- Universal Message Service
- Universal Command server
- Universal Data Mover server
- Universal Event Monitor server
- Universal Control server
- Universal Application Container

10.2 Component Definition Options Information

For each component definition option, these pages provide the following information.

10.2.1 Description

Describes the option and how it is used.

10.2.2 Usage

Provides a table of the following information:

Method	Syntax	IBM i	UNIX	Windows	z/OS
Component Definition Keyword	<Format / Value>				

10.2.2.1 Method

Identifies the method used for specifying a Universal Agent component definition option:

- Component Definition Keyword

10.2.2.2 Syntax

Identifies the syntax of the method used to specify the option:

- Format: Specific characters that identify the option.
- Value: Type of value(s) to be supplied for this method.

10.2.2.3 (Operating System)

Identifies the operating systems for which the method of specifying the option is valid:

- IBM i
- UNIX
- Windows
- z/OS

10.2.3 Values

Identifies all possible values for the specified value type.

Defaults are identified in **bold type**.

10.3 Component Definition Options

The following table identifies all of the options that can comprise a component definition provided to Universal Broker.

Option	Description
AUTOMATICALLY_START	Specification for whether the component automatically starts by the Universal Broker at start-up time or only on demand.
COMPONENT_NAME	Name by which clients know the component.
COMPONENT_TYPE	Type of component.
CONFIGURATION_FILE *	Component's configuration file name.
RESTART	Specification for whether or not the component should be restarted if it ends.
RESTART_CONDITIONS	Exit conditions criteria for which the server is considered eligible for restart.
RESTART_DELAY	Number of seconds to wait before restarting.
RESTART_MAX_FREQUENCY	Maximum frequency a server can be restarted.
RUNNING_MAXIMUM	Maximum number of this component that can run simultaneously.

Option	Description
START_COMMAND *	Component program name.
WORKING_DIRECTORY *	Path used as the working directory of the component.
* These options are required in all component definitions.	

10.4 AUTOMATICALLY_START - UBROKER Component Definition option

10.4.1 Description

The AUTOMATICALLY_START option specifies whether the component automatically starts by the Universal Broker at start-up time or only on demand.

> **Note**
>
> AUTOMATICALLY_START is optional in a component definition.

10.4.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Component Definition Keyword	auto_start *option*	✓	✓	✓	✓

10.4.3 Values

10.4.4 Values

option is the specification for how the component is started.

Valid values for *option* are:

- **yes**
  Component is started automatically by Universal Broker.
- **no**
  Component is started only on demand.

Default is no.

10.5 COMPONENT_NAME - UBROKER Component Definition option

10.5.1 Description

The COMPONENT_NAME option specifies the name by which the clients know the component.

> **Note**
>
> COMPONENT_NAME is optional in a component definition. If it is not specified, the file name is used as the component name.

10.5.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Component Definition Keyword	component_name *name*	✓	✓	✓	✓

10.5.3 Values

name is the name by which the clients know the component.

10.6 COMPONENT_TYPE - UBROKER Component Definition option

10.6.1 Description

The COMPONENT_TYPE option specifies the type of component.

Some components can execute multiple instances simultaneously with different component names. The COMPONENT_TYPE specifies the common type of component that applies to this component definition.

> **Note**
>
> COMPONENT_TYPE is optional in a component definition. If it is not specified, the component name is used.

10.6.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Component Definition Keyword	component_type *type*	✓	✓	✓	✓

10.6.3 Values

type is the type of component.

10.7 CONFIGURATION_FILE - UBROKER Component Definition option

10.7.1 Description

The CONFIGURATION_FILE option specifies the component's configuration file name (member name in z/OS).

> **Note**
>
> CONFIGURATION_FILE is required in a component definition.

IBM i	Non-qualified file names are located in the library list ***LIBL**.
UNIX	Relative paths are relative to the component's working directory.
z/OS	Member names are located in the **UNVCONF** library allocated to the **UNVCONF** ddname.

10.7.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Component Definition Keyword	configuration_file *member* or configuration_file *filename*	✓	✓	✓	✓

10.7.3 Values

member / *filename* is the name of the configuration member / file.

10.8 RESTART - UBROKER Component Definition option

10.8.1 Description

The RESTART option specifies whether or not the component should be restarted if it ends.

The component restart facility is only applicable to auto-started components.

A component is restarted when the following conditions are met:

1. Universal Broker is not in shutdown mode.
2. Component has not been stopped by the Broker, Universal Control, or Universal Enterprise Controller. This is considered a controlled shutdown.
3. RESTART option value = **no**.
4. Component's exit conditions must meet one of the values specified by the RESTART_CONDITIONS option.
5. Component must not have been restarted more than specified by the RESTART_MAX_FREQUENCY option.

10.8.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	restart *options*	✓	✓	✓	✓

10.8.3 Values

options is the specification for whether or not the component should be restarted.

Valid values for *options* are:

- **yes**
  Component should be restarted if it meets the restart criteria. (This is applicable only for auto-started components.)
- **no**
  Component should not be restarted.

Default is no.

10.9 RESTART_CONDITIONS - UBROKER Component Definition option

10.9.1 Description

The RESTART_CONDITIONS option specifies the exit conditions of the component for which the component should be considered eligible for restart.

If the exit conditions of the component do not meet the criteria, it will not be restarted.

10.9.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	restart_conditions *conditions*	✓	✓	✓	✓

10.9.3 Values

conditions is a comma-separated list of exit (return) conditions.

Exit condition names are based on the Universal Agent return codes. Category names are used instead of numeric values, as the exit code numeric value may not be consistent across all platforms.

The exit conditions are:

ABNORMAL	Component ended abnormally due to a UNIX signal, Windows Exception, z/OS ABEND, etc.
SUCCESS	Component ended normally with exit code 0.
WARN	Component ended normally with a warning exit code.
ERROR	Component ended normally with an error exit code.
FATAL	Component ended normally with a fatal exit code.
CONFIG	Component ended normally with a configuration error exit code.
SECURITY	Component ended normally with a security related exit code.
NETWORK	Component ended normally with a network related exit code.
SHUTDOWN	Component ended normally with a shutdown related exit code.
LICENSE	Component ended normally with a license violation related exit code.
ALL	All of the above.

Default is ABNORMAL.

10.10 RESTART_DELAY - UBROKER Component Definition option

10.10.1 Description

The RESTART_DELAY option specifies the number of seconds to wait from the time the Universal Broker detects the component has ended until the Broker restarts the component.

10.10.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	restart_delay *seconds*	✓	✓	✓	✓

10.10.3 Values

seconds is the number of seconds to wait.

Default is 5.

10.11 RESTART_MAX_FREQUENCY - UBROKER Component Definition option

10.11.1 Description

The RESTART_MAX_FREQUENCY option specifies the maximum frequency in which a component can be restarted in a specific time interval.

If a component becomes eligible for restart but exceeds the maximum restart frequency, it will not be restarted.

10.11.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Configuration File Keyword	restart_max_frequency *number*/*interval*	✓	✓	✓	✓

10.11.3 Values

number is the maximum number of restarts.

interval is the time interval in which the specified maximum number of restarts (*number*) is allowed.

Valid values for interval are **week**, **day**, **hour**, and **minute**.

Default is 2 / day.

10.12 RUNNING_MAXIMUM - UBROKER Component Definition option

10.12.1 Description

The RUNNING_MAXIMUM option specifies the maximum number of this component that can run simultaneously.

If this maximum number is reached, any command received to start the component is rejected.

Note

RUNNING_MAXIMUM is optional in a component definition.

10.12.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Component Definition Keyword	running_max *maximum*	✓	✓	✓	✓

10.12.3 Values

maximum is the maximum number of this component that can run simultaneously.

Default is 100.

Note

If you specify 0 for *maximum*, the default (100) will be used. To use 0 for the maximum number of the component, specify -1 or less for *maximum*.

10.13 START_COMMAND - UBROKER Component Definition option

10.13.1 Description

The START_COMMAND option specifies the full path name (member name for z/OS) of the program.

Optionally, START_COMMAND also can specify command line options.

Note

START_COMMAND is required in a component definition.

z/OS	Member names are located in the **SUNVLOAD** library.
IBM i	Non-qualified program names are located in the library list ***LIBL**.

10.13.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Component Definition Keyword	start_command *member* **or** start_command *name[options]*	✓	✓	✓	✓

10.13.3 Values

member / *name* is the program name of the component.

options is the optional list of command line options.

z/OS

options is not a valid value for START_COMMAND.

10.14 WORKING_DIRECTORY - UBROKER Component Definition option

10.14.1 Description

The WORKING_DIRECTORY option specifies the full path name of the directory used as the working directory of the component.

Note

WORKING_DIRECTORY is required in a component definition.

10.14.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
Component Definition Keyword	working_directory *directory*	✓	✓	✓	✓

10.14.3 Values

directory is the full path name of the working directory.

Default is (.).

IBM i	working_directory serves as a required placeholder only. Do not change its value.
UNIX	Relative path names are relative to the Universal Broker working directory. Full path names are recommended.
Windows	Relative path names are relative to the Universal Broker working directory. Full path names are recommended.
z/OS	The path is the z/OS UNIX path used as the working directory of the component.

11 Universal Broker UACL Entries

11.1 Universal Broker UACL Entries

This page provides links to detailed information on the Universal Access Control List (UACL) entries available for use with Universal Broker.

The UACL entries are listed alphabetically, without regard to any specific operating system.

Information on how these UACL entries are used is documented in the operating system-specific pages of this document.

11.2 UACL Entries Information

For each UACL entry, these pages provide the following information.

11.2.1 Description

Describes the UACL entry and how it is used.

11.2.2 Usage

Provides a table of the following information:

Method	Syntax	IBM i	UNIX	Windows	z/OS
UACL File Keyword	<Type / Rule>				

11.2.2.1 Method

Identifies the method used for specifying a UACL entry:

- UACL FIle Keyword

11.2.2.2 Syntax

Identifies the syntax of the method used for a UACL entry:

- Type: Universal Agent component to which the rule applies.
- Rule: Client's identity, request to which the entry pertains, and security attributes that the entry enforces.

11.2.2.3 (Operating System)

Identifies the operating systems for which the method of specifying the UACL entry is valid:

- IBM i
- UNIX
- Windows
- z/OS

11.2.3 Values

Identifies all possible values for the fields in a UACL entry rule.

Defaults are identified in **bold type**.

11.3 UACL Entries List

The following table identifies all Universal Broker UACL Entries.

UACL Entry	Description
UBROKER_ACCESS	Allows or denies a Universal Agent component access to Universal Broker services.
CERT_MAP	Maps a client X.509 certificate to certificate identifier.
EVENT_ACCESS	Controls which Universal Enterprise Controller has read and delete access to the Universal Event Subsystem event data maintained by the Universal Broker.
REMOTE_CONFIG_ACCESS	Authorizes update access to the product configuration files and setting of the configuration managed mode of the Broker. There are two forms to this entry: • **remote_config_access** • **remote_config_cert_access**

11.4 UBROKER_ACCESS - UBROKER UACL entry

11.4.1 Description

A UBROKER_ACCESS UACL entry specifies whether to allow or deny a Universal Agent component access to Universal Broker services.

If a request from a component comes from an IP address identified in this UBROKER_ACCESS entry, the rule is considered a match. The first matching rule is used to control access.

11.4.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
UACL File Keyword	ubroker_access *host,access*	✓	✓	✓	✓

11.4.3 Values

host specifies an IP address of a Universal Agent component.

(See UACL Entries for details on *host* specification syntax.)

access specifies whether the connection is allowed or denied.

Valid values for *access* are:

- **deny**
  IP connection is denied. No message is returned to the remote end. The connection is immediately closed.
- **allow**
  IP connection is accepted and processed.

Default is allow.

11.5 CERT_MAP - UBROKER UACL entry

11.5.1 Description

A CERT_MAP UACL entry maps a client X.509 certificate to certificate identifier.

CERT_MAP defines one or more certificate fields and values that are used to match against the client's certificate. All of the fields defined by CERT_MAP must match the client certificate in order for the rule to be considered a match.

11.5.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
UACL File Keyword	cert_map *id=certid,cert-field(s)*	✓	✓	✓	✓

11.5.3 Values

id is the certificate identifier.

cert-fields is a comma-separated list of one or more certificate fields. Values in the certificate fields support generic specification.

11.5.4 CERT_MAP Examples:

Example	Description
cert_map id=myhost,hostname=myhost.com	Validates certificate subject alternate name dns.
cert_map id=myhost,hostname=myhost.com,serialnumber=025678B34	Validates certificate subject alternate name dns, and certificate serial number.
cert_map id=myhost,subject="/CN=myhost.com/"	Validates certificate subject common name.
cert_map id=myuser,email=myuser@myhost.com	Validates certificate subject alternate name email.
cert_map id=myuser,ipaddress=127.0.0.1	Validates certificate subject alternate name IP address.

(See X.509 Certificates for a detail discussion on the *cert-fields* values.)

11.6 EVENT_ACCESS - UBROKER UACL entry

11.6.1 Description

A EVENT_ACCESS entry controls which Universal Enterprise Controller has read and delete access to the Universal Event Subsystem event data maintained by the Universal Broker.

There are two forms of the EVENT_ACCESS entry:

- **event_access** is based on the host name and user ID of the client.
- **event_cert_access** is based on a certificate map of the client.

11.6.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
UACL File Keyword	event_access *host,remote_user,read_access,delete_access* event_cert_access *certid,read_access,delete_access*	✓	✓	✓	✓

11.6.3 Values

host specifies an IP address of a Universal Agent component.

remote_user is the user identifier with which Universal Enterprise Controller is executing on the remote system.

(See Client Identification for details on *host* and *remote_user* specification syntax.)

read_access specifies whether or not reading event data is allowed.

Valid values for *read_access* are:

- **deny**
  Access is denied to the read request.
- **allow**
  Access is allowed to the read request.

delete_access specifies whether or not deleting event data is allowed.

Valid values for *delete_access* are:

- **deny**
  Access is denied to the delete request.
- **allow**
  Access is allowed to the delete request.

11.6.3.1 Defaults

event_access ALL,*,allow,deny

event_cert_access *,allow,deny

11.6.4 Examples

event_access 10.20.30.40,uecprod,allow,allow

event_access ALL,*,deny,deny

event_cert_access uecprod,allow,allow

event_cert_access *,deny,deny

11.7 REMOTE_CONFIG_ACCESS - UBROKER UACL entry

11.7.1 Description

A REMOTE_CONFIG_ACCESS entry authorizes update access to the product configuration files and setting of the configuration managed mode of the Broker.

Universal Enterprise Controller requests this access when it needs to configure a product using its remote configuration capabilities.

There are two forms of the REMOTE_CONFIG_ACCESS entry:

- **remote_config_access** is based on the host name and user ID of the client.
- **remote_config_cert_access** is based on a certificate map of the client.

11.7.2 Usage

Method	Syntax	IBM i	UNIX	Windows	z/OS
UACL File Keyword	remote_config_access *host,remote_user,update_access,control_access* remote_config_cert_access *certid,update_access,control_access*	✓	✓	✓	✓

11.7.3 Values

host specifies an IP address of a Universal Agent component.

remote_user is the user identifier with which Universal Enterprise Controller is executing on the remote system.

(See Client Identification for details on *host* and *remote_user* specification syntax.)

update_access specifies whether or not configuration file updates are allowed.

Valid values for *update_access* are:

- **deny**
  Access is denied to the update request.
- **allow**
  Access is allowed to the update request.

control_access specifies whether or not the Broker can be placed into managed mode or taken out of managed mode.

Valid values for *control_access* are:

- **deny**
  Access is denied to the managed mode request.
- **allow**
  Access is allowed to the managed mode request.

11.7.3.1 Defaults

remote_config_access ALL,*,deny,deny

remote_config_cert_access *,deny,deny

11.7.4 Examples

remote_config_access 10.20.30.40,uecprod,allow,allow

remote_config_access ALL,*,deny,deny

```
remote_config_cert_access uecprod,allow,allow
remote_config_cert_access *,deny,deny
```

12 Universal Broker Configuration Options Refresh

12.1 Universal Broker Configuration Options Refresh

As with all Universal Agent components, all Universal Broker configuration options can be modified by editing the configuration file directly.

However, unlike other components, not all Universal Broker options can be modified via I-Management Console. (In I-Management Console, these Universal Broker options are read-only.)

Some options can be modified only by editing the Universal Broker configuration file, `unbroker.conf`. For these modifications to be updated in Universal Broker memory and take immediate effect, Universal Broker must be recycled; they do not take effect when Universal Broker is simply refreshed. (See Options - Configuration File Editable Only, Recycle Required.)

All other Universal Broker options can be modified either:

- By editing `ubroker.conf`.
- Via I-Management Console.
- Via the Universal Configuration Manager.

Depending on the option, for a modification to be updated in Universal Broker memory and take immediate effect:

- Universal Broker must be recycled (see Options - I-Management Console and Configuration File Editable, Recycle Required).
- Universal Broker must be refreshed (see Options - I-Management Console and Configuration File Editable, Refresh Required).
  This is done either:
  - By issuing a Universal Control configuration refresh request (via the REFRESH_CMD configuration option), if the modifications are made in the configuration file.
  - Automatically, if the modifications are made via I-Management Console or the Universal Configuration Manager.

12.2 Options - Configuration File Editable Only, Recycle Required

12.2.1 Configuration File Editable Only, Recycle Required

The following table identifies Universal Broker options that you can modify only by editing the Universal Broker configuration file.

Universal Broker must be recycled in order for the modified values to be used. These options are not updated when Universal Broker simply is refreshed.

(In I-Management Console, these options are Read-Only.)

Option	Description
BIF_DIRECTORY	Broker Interface File directory that specifies where Universal Broker will create its interface file.

Option	Description
COMPONENT_DIRECTORY	Component definition file directory.
INSTALLATION_DIRECTORY	Base directory where product is installed.
MOUNT_POINT	HFS or zFS database mount directory.
MOUNT_POINT_MODE	HFS or zFS permission mode for MOUNT_POINT.
NLS_DIRECTORY	UMC and UTT file directory.
PID_FILE_DIRECTORY	PID file location.
SMF_EXIT_LOAD_LIBRARY	UNVACTRT SMF exit load library.
SPOOL_DIRECTORY	Spool file directory.
SYSTEM_ID	Universal Broker running on a system (O/S image).
UCMD_STC_SUPPORT	Support for Universal Command started tasks.
UNIX_DB_DATA_SET	HFS or zFS data set used for the Universal Broker's databases.
UNIX_SPOOL_DATA_SET	HFS or zFS data set used for the Universal Broker's spool.

A Stonebranch tip:

If the PID_FILE_DIRECTORY value is modified, the UNIX script that starts/stops/restarts the Universal Broker, **ubrokerd**, also must be modified to indicate the location of the Broker's PID file.

If **ubrokerd** is not modified, it will not know the Process ID of the executing Universal Broker. Thus, it will not be able to return status information of the executing Universal Broker successfully.

12.3 Options - I-Management Console and Configuration File Editable, Recycle Required

12.3.1 I-Management Console and Configuration File Editable, Recycle Required

The following table identifies Universal Broker options that you can modify either by editing the Universal Broker configuration file or via I-Management Console, and for which Universal Broker must be recycled in order for the modifications to take effect. These options are not updated when Universal Broker simply is refreshed.

Windows

If the options are modified via the Universal Configuration Manager, Universal Broker must be recycled.

Option	Description
CA_CERTIFICATES	Path to PEM-formatted trusted CA X.509 certificates.
CERTIFICATE	Path to Broker's PEM-formatted X.509 certificate.
CERTIFICATE_REVOCATION_LIST	Path to PEM-formatted CRL.
COMPONENT_PORT	TCP/IP port used for Broker-Component communications.
PRIVATE_KEY	Path to Broker's PEM formatted RSA private key.
PRIVATE_KEY_PWD	Password for the Broker's PRIVATE_KEY.
SAF_KEY_RING	SAF certificate key ring name.
SAF_KEY_RING_LABEL	SAF certificate key ring label.
SERVICE_BACKLOG	Service interface backlog size for pending connection requests.
SERVICE_IP_ADDRESS	TCP/IP address on which the Broker listens.
SERVICE_PORT	TCP/IP port number on which the Broker listens.
SSL_IMPLEMENTATION	SSL/TLS implementation to be used for network configuration.

12.4 Options - I-Management Console and Configuration File Editable, Refresh Required

12.4.1 I-Management Console and Configuration File Editable, Refresh Required

The following table identifies Universal Broker options that you can modify by editing the Universal Broker configuration file or via I-Management Console, and for which Universal Broker only needs to be refreshed in order for the modifications to take effect.

- If the options are modified by editing the Universal Broker configuration file, a Universal Control REFRESH command must be issued.
- If the options are modified via I-Management Console, Universal Broker is refreshed automatically.

Windows

If the options are modified via the Universal Configuration Manager, Universal Broker is refreshed automatically.

Option	Description
ACTIVITY_MONITORING	Specification for whether or not product activity monitoring events are generated.
CODE_PAGE	Text translation code page.
CTL_SSL_CIPHER_LIST	SSL/TLS cipher list for the control sessions.

DNS_CACHE_TIMEOUT	Time-out for DNS cache.
EVENT_GENERATION	Events to be generated as persistent events.
LOG_DIRECTORY	Log file directory.
LOG_FILE_GENERATIONS	Total number of log files that will be saved within the log directory.
LOG_FILE_LINES	Total number of lines to be written to the log file before the log file is wrapped.
MESSAGE_DESTINATION	Location where messages are written.
MESSAGE_LANGUAGE	Language of messages written.
MESSAGE_LEVEL	Level of messages written.
MONITOR_EVENT_EXPIRATION	Duration of a monitoring event record in the Universal Broker local UES database.
PERSISTENT_EVENT_EXPIRATION	Duration of a persistent event record in the Universal Broker local UES database.
REQ_UPPS_CONN	Number of PeopleSoft connections that Universal Broker will request from a pool of connections permitted by your Universal Agent license. **Note** Cannot be modified via Universal Configuration Manager.
REQ_USAP_CONN	Number of SAP connections that Universal Broker will request from a pool of connections permitted by your Universal Agent license. **Note** Cannot be modified via Universal Configuration Manager.
RUNNING_MAX	Maximum number of simultaneous components.
TMP_DIRECTORY	Directory for temporary files.
TRACE_DIRECTORY	Directory for trace files.
TRACE_FILE_LINES	Maximum number of lines written to the trace file.
TRACE_TABLE	Memory trace table specification.
WORKING_DIRECTORY	Broker's working directory.

13 Universal Broker Additional Information

13.1 Universal Broker Additional Information

The following table identifies and provides links to additional information used by or specific to Universal Broker.

Information	Description
UACL Generics	Generics allow you to specify a string pattern to match a value, thus providing a convenient way of specifying one or more values.
Character Code Pages	Character code pages provided by Stonebranch Inc. for use with Universal Agent components on each supported operating system.
SSL/TLS Cipher Suites	SSL/TLS cipher suites for use with Universal Broker.
UTT Files	Universal Translate Table (UTT) files are used to translate between Unicode and the local single-byte code page.

13.2 Character Code Pages - UBROKER

The following table identifies the character code pages provided by Stonebranch Inc. for use with Universal Agent on each supported operating system.

Code Page	CCSID	z/OS	UNIX	Windows	IBM i / HFS	IBM i / LIB	HP NonStop
IBM037	037	✓			✓	✓	
IBM273	273	✓			✓	✓	
IBM277	277	✓			✓	✓	
IBM278	278	✓			✓	✓	
IBM280	280	✓			✓	✓	
IBM284	284	✓			✓	✓	
IBM500	500	✓			✓	✓	
IBM875	875	✓					
IBM1025		✓					

Code Page	CCSID	z/OS	UNIX	Windows	IBM i / HFS	IBM i / LIB	HP NonStop
IBM1047		✓			✓	✓	
IBM1140	1140	✓			✓	✓	
IBM1141	1141	✓			✓	✓	
IBM1142	1142	✓			✓	✓	
IBM1143	1143	✓			✓	✓	
IBM1144	1144	✓			✓	✓	
IBM1145	1145	✓			✓	✓	
IBM1146	1146	✓			✓	✓	
IBM1147	1147	✓			✓	✓	
IBM1148	1148	✓			✓	✓	
IBM4971	4971	✓					
ISO8859-1	819		✓	✓	✓		✓
ISO8859-2	912		✓	✓	✓		✓
ISO8859-3	913		✓	✓	✓		✓
ISO8859-4	914		✓	✓	✓		✓
ISO8859-5	915		✓	✓	✓		✓
ISO8859-6	1089		✓	✓	✓		✓
ISO8859-7	813		✓	✓	✓		✓
ISO8859-8	916		✓	✓	✓		✓
ISO8859-9	920		✓	✓	✓		✓
ISO8859-10			✓	✓	✓		✓
ISO8859-13	921		✓	✓	✓		✓
ISO8859-14			✓	✓	✓		✓
ISO8859-15	923		✓	✓	✓		✓
PC437	437			✓	✓		
PC737	737			✓	✓		

Code Page	CCSID	z/OS	UNIX	Windows	IBM i / HFS	IBM i / LIB	HP NonStop
PC775	775			✔	✔		
PC850	850			✔	✔		
PC852	852			✔	✔		
PC855	855			✔	✔		
PC857	857			✔	✔		
PC860	860			✔	✔		
PC861	861			✔	✔		
PC862	862			✔	✔		
PC863	863			✔	✔		
PC864	864			✔	✔		
PC865	865			✔	✔		
PC866	866			✔	✔		
PC869	869			✔	✔		
PC874	874			✔	✔		
WIN1250	1250			✔	✔		
WIN1251	1251			✔	✔		
WIN1252	1252			✔	✔		
WIN1253	1253			✔	✔		
WIN1254	1254			✔	✔		
WIN1255	1255			✔	✔		
WIN1256	1256			✔	✔		
WIN1257	1257			✔	✔		
WIN1258	1258			✔	✔		

13.3 SSL/TLS Cipher Suites - UBROKER

13.3.1 SSL/TLS Cipher Suites

The following table identifies all SSL/TLS 1.2 (and prior) cipher suites provided by Stonebranch, Inc. for use with Universal Broker.

The list is in default order, with the most preferred suite first and the least preferred suite last.

Cipher Suite	Description
AES256-GCM-SHA384	256-bit AES encryption in Galois Counter Mode, SHA-2 384-bit message digest.
AES256-SHA	256-bit AES encryption and SHA-1 message digest.
AES128-GCM-SHA256	128-bit AES encryption in Galois Counter Mode, SHA-2 256-bit message digest.
AES128-SHA	128-bit AES encryption and SHA-1 message digest.
ECDHE-RSA-AES256-GCM-SHA384	Ephemeral Elliptic Curve Diffie-Hellman Key Exchange, RSA authentication, 256-bit AES encryption in Galois Counter Mode, SHA-2 384-bit message digest.
ECDHE-ECDSA-AES256-GCM-SHA384	Ephemeral Elliptic Curve Diffie-Hellman Key Exchange, ECDSA authentication, 256-bit AES encryption in Galois Counter Mode, SHA-2 384-bit message digest.
ECDHE-RSA-AES128-GCM-SHA256	Ephemeral Elliptic Curve Diffie-Hellman Key Exchange, RSA authentication, 128-bit AES encryption in Galois Counter Mode, SHA-2 256-bit message digest.
ECDHE-ECDSA-AES128-GCM-SHA256	Ephemeral Elliptic Curve Diffie-Hellman Key Exchange, ECDSA authentication, 128-bit AES encryption in Galois Counter Mode, SHA-2 256-bit message digest.
RC4-SHA	128-bit RC4 encryption and SHA-1 message digest.
RC4-MD5	128-bit RC4 encryption and MD5 message digest.
DES-CBC3-SHA	128-bit Triple-DES encryption and SHA-1 message digest.
DES-CBC-SHA	128-bit DES encryption with SHA-1 message digest. **Note** As of Universal Agent 6.7.0.0, DES-CBC-SHA is supported only on HP-UX. Additionally, any Agents on HP-UX that accept connections from, or attempt connections to, Agents on other platforms must be configured with at least one currently supported cipher suite besides DES-CBC-SHA. Therefore, those HP-UX Agents cannot be configured only with DES-CBC-SHA in their list of cipher suites.

13.3.2 SSL/TLS 1.3 Cipher Suites

The following table identifies all SSL/TLS 1.3 cipher suites provided by Stonebranch, Inc. for use with Universal Broker.

The list is in default order, with the most preferred suite first and the least preferred suite last.

Cipher Suite	Description
TLS_AES_256_GCM_SHA384	256-bit AES encryption in Galois Counter Mode, SHA-2 384-bit message digest
TLS_CHACHA20_POLY1305_SHA256	256-bit CHACHA encryption with POLY1305 message authentication, SHA-2 256-bit message digest
TLS_AES_128_GCM_SHA256	128-bit AES encryption in Galois Counter Mode, SHA-2 256-bit message digest

13.4 UACL Generics - UBROKER

13.4.1 UACL Generics

```
Generics
--------

Generics allow you to specify a string pattern to match a value.
The string pattern is a convenient way of specifying one or more
values.  The following pattern control characters may be used:

  *       Match 0 or more characters.
  ?       Match one character.
  /       Escape character to escape matching control characters
          so they are used as literal characters.

In addition to the pattern control characters, pattern control
codes may be specified to control how the pattern matching is
performed.  The following pattern control characters may be used:

  c       Perform a case insentitive compare.
  C       Perform a case sensitive compare (the default).
  s       Normalize spaces by reducing multiple spaces to one.
  S       Don't normalize spaces (the default).

Examples:

  *le     matches "apple", "le", and "red apple".
  /*le    matches "*le" only.
  ap?le   matches "apple", "ap le", but not "aple".
  /c*le   matches "apple", "APPLE", and "appLe".
  a/c*le  matches "apple", "aPpLe", but not "APPLE".
  /s*le   matches "apple", "red apple", and "red     apple".
```

13.5 UTT Files - UBROKER

The following table identifies the Universal Translate Table (UTT) files that are used to translate between Unicode and the local single-byte code page.

Operating System	UTT File Location*
IBM i	UTT files are located in the **UNVPRD520/UNVNLS** file. *codepage* is the member name of the UTT file.
z/OS	UTT files are members of the PDS allocated to the Broker ddname **UNVNLS**. *codepage* specifies the member name.
UNIX	UTT files are located in the directory specified by the NLS_DIRECTORY option, which defaults to **/opt/universal/nls**. *codepage* is the base file name of the UTT file.
Windows	UTT files are located in the **NLS** subdirectory of the installation directory. *codepage* is the base file name of the UTT file.
HP NonStop	UTT files are located in the **$SYSTEM.UNVNLS** subvolume. *codepage* is the base file name of the UTT file.