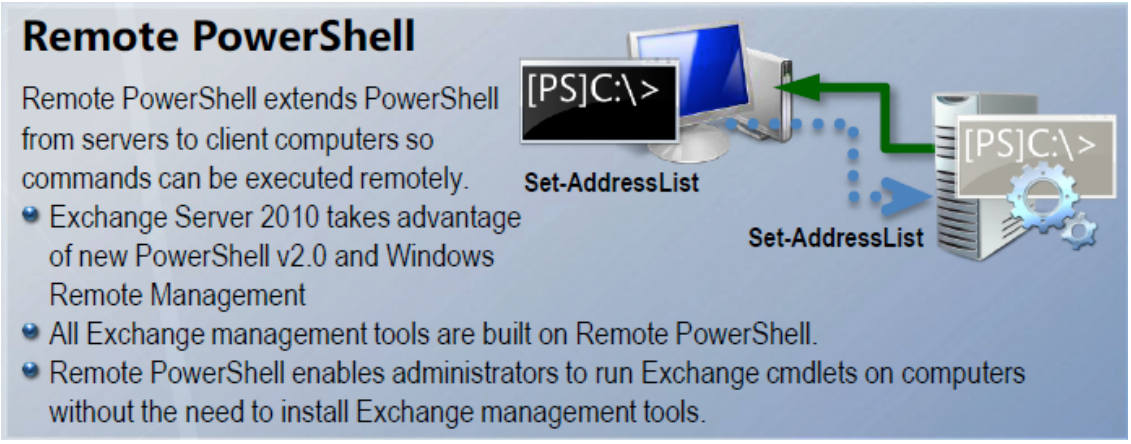
Argent for Exchange Monitoring Secrets For Exchange 2010

ARGENT

Contents

Exchange 2010 Overview

The primary role of this document is to showcase the Argent for Exchange rules that can be used for monitoring Exchange 2010 environments. The primary method of managing and monitoring an Exchange 2010 environment is the use of the Powershell Exchange Management Shell and Argent for Exchange will integrate with these management tools.



Before looking at the Argent for Exchange rules this document will cover an overview of the Exchange 2010 components and their functionality covering the following roles

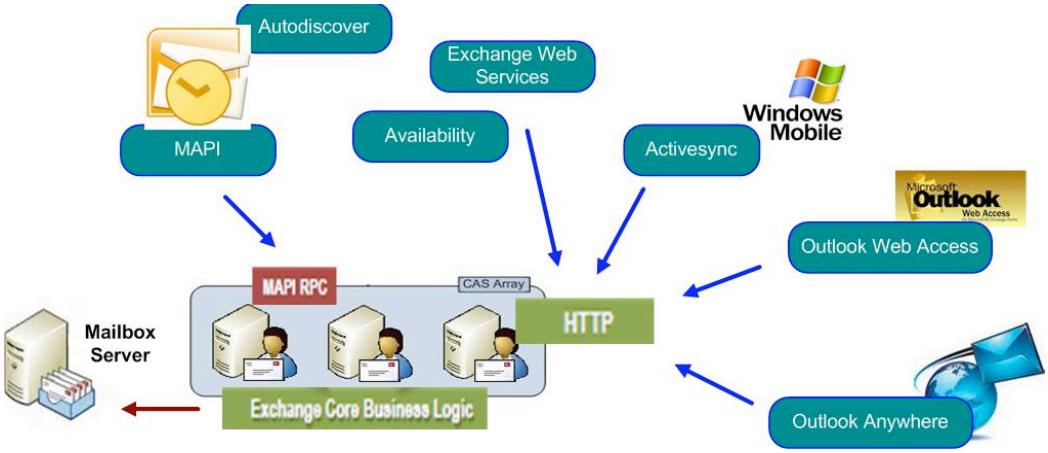
- Client Access Server
- Hub Transport Server
- Mailbox Server
- Edge Transport Server
- Unified Messaging Server

It will also give an overview of the data flow within a typical Exchange 2010 environment.

Client Access Server Role Overview

Client access server (CAS)—The CAS role allows for client connections via nonstandard methods such as Outlook Web App (OWA), Exchange ActiveSync, Post Office Protocol 3 (POP3), and Internet Message Access Protocol (IMAP). The Client Access server role also provides access to free/busy data by using the Availability service and enables certain clients to download automatic configuration settings from the Autodiscover service.

Exchange Server 2010 also forces MAPI traffic and effectively all client traffic through the CAS layer. CAS servers are the replacement for Exchange 2000/2003 front-end servers and can be load balanced for redundancy purposes.



Outlook Web Access

Outlook Web App lets you access your e-mail from any Web browser. Outlook Web App (known as Outlook Web Access in earlier versions of Microsoft Exchange) has been redesigned in Exchange 2010. Features such as Chat, Text Messaging, mobile phone integration, and Conversation View provide an enhanced user experience from a Web browser.

Exchange ActiveSync

Exchange ActiveSync lets you synchronize data between your mobile phone and Exchange 2010. You can synchronize e-mail, contacts, calendar information, and tasks.

If you use a phone that has Windows Mobile 5.0 with the Messaging Security and Feature Pack (MSFP) installed or a later version, your mobile phone will support Direct Push. Direct Push technology is built into Exchange ActiveSync and keeps a mobile phone continuously synchronized.

POP3 and IMAP

In addition to supporting MAPI and HTTP clients, Exchange 2010 also supports POP3 and IMAP4 clients.

The Autodiscover Service

The Autodiscover service enables Outlook clients and some mobile phones to receive their necessary profile settings directly from the Exchange server by using the client's domain credentials. These settings automatically update the client with the information that's needed to create the user's profile.

The Availability Service

The Exchange 2010 Availability service provides secure, consistent and up-to-date free/busy data to computers that are running Microsoft Office Outlook 2007 and later versions of Outlook. These versions of Outlook use the Autodiscover service to obtain the URL of the Availability service. Essentially, the Autodiscover service helps capable Outlook clients locate different Web services, such as the Microsoft Exchange Unified Messaging service, the Offline Address Book, and Availability services.

Outlook Anywhere

For remote connections, Outlook offers Outlook Anywhere, an alternative to VPN connections that allows you to use Outlook just as you normally do at your organization, without the need for any special connections or hardware, such as smart cards and security tokens. Outlook can connect to Exchange through the Internet by using remote procedure call (RPC) over HTTP. The Outlook Anywhere feature allows you to access your Exchange account remotely from the Internet when you are working outside your organization's firewall.

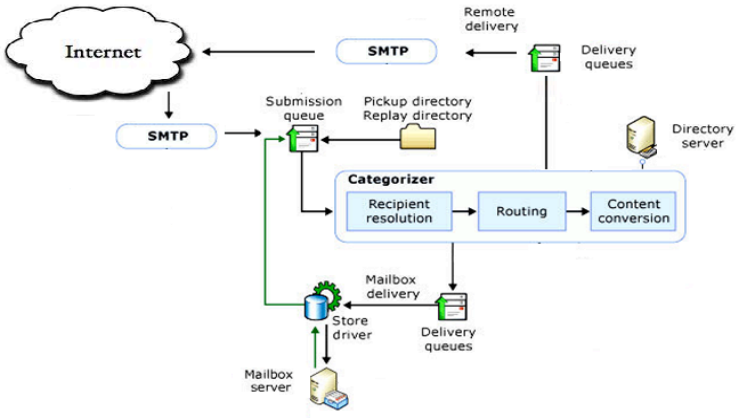
Exchange Web Services

Exchange Web Services gives you programmatic access to the information and business logic in Exchange 2010. If you think about all the stuff you do in Outlook 2007 to manage your daily life (mail to communicate with friends and coworkers, calendar items to manage your day and tasks to track the things that you need to get done), all that information and business logic (free/busy when scheduling a meeting) is provided by Exchange and accessible via the Exchange Web Services Managed API.

Hub Transport Server Role Overview

Hub Transport server—The Hub Transport server role acts as a mail bridgehead for mail sent between servers in one AD site and mail sent to other AD sites. There needs to be at least one Hub Transport server within an AD site that contains a server with the mailbox role, but there can also be multiple Hub Transport servers to provide for redundancy and load balancing. HT roles are also responsible for message compliance and rules. The HT role can be combined with other roles on a server, and is often combined with the CAS role.

Deployed inside your Active Directory forest, the Hub Transport server role handles all mail flow inside the organization, applies transport rules, applies journaling policies, and delivers messages to a recipient's mailbox. Messages that are sent to the Internet are relayed by the Hub Transport server to the Edge Transport server role that's deployed in the perimeter network. Messages that are received from the Internet are processed by the Edge Transport server before they're relayed to the Hub Transport server. If you don't have an Edge Transport server, you can configure the Hub Transport server to relay Internet messages directly or utilize a third-party smart host.



The message-processing scenarios that you can manage on the Hub Transport server role are described in the following sections.

Internal Mail Flow

The Hub Transport server role processes all messages that are sent inside the organization before the messages are delivered to a recipient's Inbox or are routed to users outside the organization. There are no exceptions to this behavior; messages are always passed through a server that runs the Hub Transport server role.

Messages are submitted to the Hub Transport server in three ways: through SMTP submission, from the Pickup directory, or when a user inside the organization sends a message, which is picked up from the user's Outbox by the *store driver*. The store *driver* is a software component of the Hub Transport server that delivers inbound messages to *Exchange stores*, the databases that contain public folder and mailbox stores.

When messages are submitted to the Hub Transport server, they're processed by the *categorizer*. The categorizer is a component of Exchange transport that processes all inbound messages and determines what to do with the messages based on information about the intended recipients. In Exchange 2010, the Hub Transport server uses the categorizer to expand distribution lists and to identify alternative recipients and forwarding addresses. After the categorizer retrieves full information about the recipients, it uses that information to apply policies, route the messages, and perform content conversion. Messages are then delivered locally by the store driver to a recipient's mailbox, or they're delivered remotely by using SMTP to send messages to another transport server. Messages that are sent by users in your organization are picked up from the sender's Outbox by the store driver and are put in the Submission queue on a server that runs the Hub Transport server role.

Messaging Policy and Compliance Features

With a collection of transport agents, you can configure rules and settings that are applied as messages enter and leave the mail flow components. You can create messaging policy and rule settings that are designed to meet different regulations and that can easily be changed to adapt to your organization's requirements. The transport-based messaging policy and compliance features include server-based rules that you configure to enforce your organization's compliance scenarios and the Journaling agent that acts to enforce message retention.

Anti-Spam and Antivirus Protection

Exchange 2010 provides anti-spam and antivirus protection for messages. Although these features are designed for use in the perimeter network on the Edge Transport server role, the Edge Transport agents can also be configured on the Hub Transport server. By default, these agents aren't enabled on the Hub Transport server role.

Mailbox server—the mailbox server role is intuitive; it acts as the storehouse for mail data in users' mailboxes and down-level public folders if required. All connections to the mailbox servers are proxy through the CAS servers.

In Microsoft Exchange Server 2010, the Mailbox server role is one of several server roles that you can install and configure on a server running Windows Server 2008. The Mailbox server role is the most common server role and is at the core of an Exchange organization. Servers on which the Mailbox server role is installed are called *Mailbox servers*.

Mailbox servers perform the following functions:

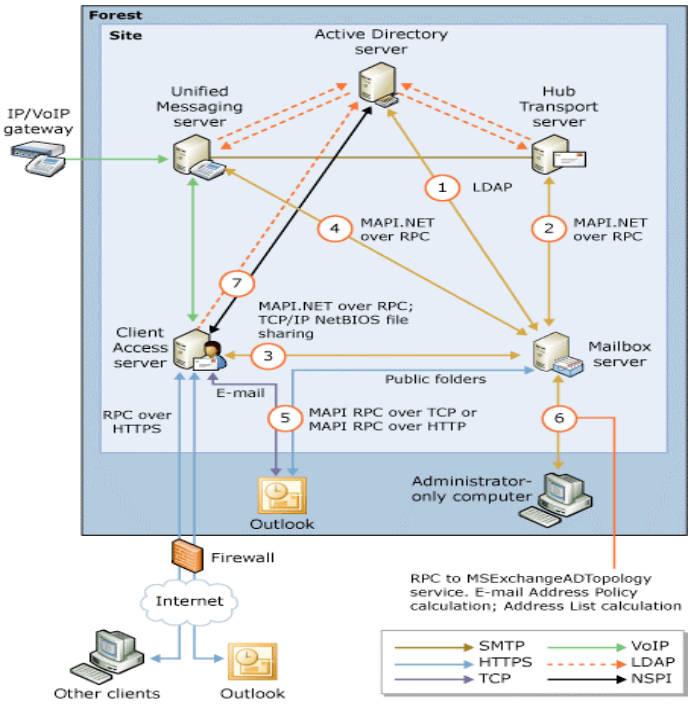
- Host mailbox databases
- Provide e-mail storage
- Host public folder databases
- Calculate e-mail address policies
- Generate address lists and offline address books (OABs)
- Conduct Multi-Mailbox Searches
- Provide high availability and site resiliency
- Provide content indexing
- Provide messaging records management (MRM) and retention policies

Unified Messaging server—The Unified Messaging server role allows a user's Inbox to be used for voice messaging and fax capabilities.

Edge Transport server—The Edge Transport server role was introduced with Exchange Server 2007, and consists of a standalone server that typically resides in the demilitarized zone (DMZ) of a firewall. This server filters inbound SMTP mail traffic from the Internet for viruses and spam, and then forwards it to internal Hub Transport servers. Edge Transport servers keep a local AD Application Mode (ADAM) instance that is synchronized with the internal AD structure via a mechanism called EdgeSync. This helps to reduce the surface attack area of Exchange Server. The Edge Transport role can only exist by itself on a server; it cannot be combined with other roles.

Exchange 2010 Simplified Data Flow

The following diagram and text explains the core data flow within an Exchange 2010 environment.



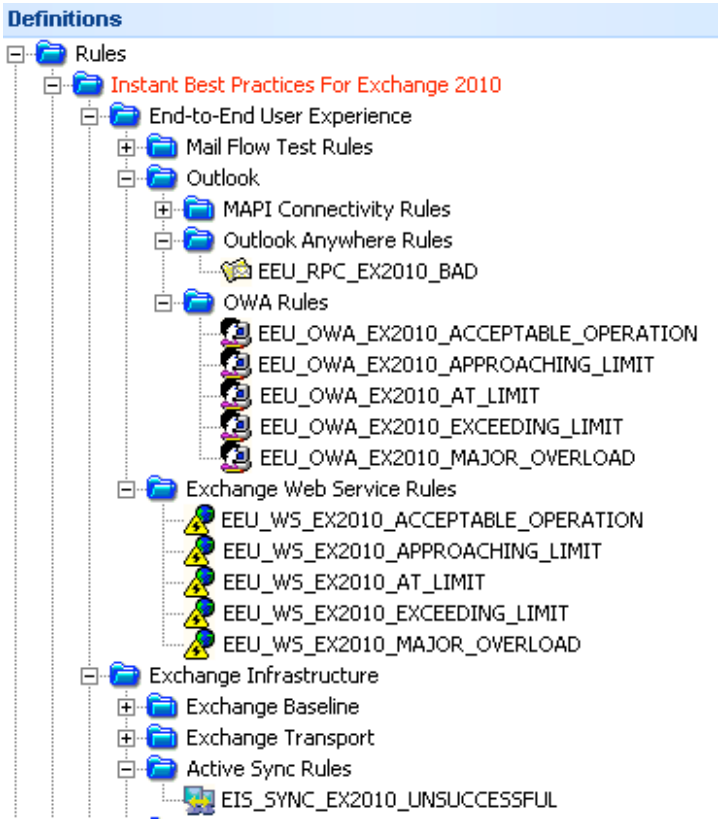
1. The Mailbox server uses LDAP to access recipient, server, and organization configuration information from Active Directory.
2. The store driver on the Hub Transport server places messages from the transport pipeline into the appropriate mailbox. The store driver on the Hub Transport server also adds messages from a sender's Outbox on the Mailbox server to the transport pipeline.
3. The Client Access server sends requests from clients to the Mailbox server, and returns data from the Mailbox server to the clients. The Client Access server also accesses OAB files on the Mailbox server through NetBIOS file sharing. The types of data that the Client Access server sends between the client and the Mailbox server include messages, free/busy data, client profile settings, and OAB data.
4. The Unified Messaging server retrieves e-mail, voice mail messages, and calendar information from the Mailbox server for Outlook Voice Access. The Unified Messaging server also retrieves storage quota information from the Mailbox server.
5. Outlook clients inside your firewall access the Client Access Server to send and retrieve messages. Outlook clients outside the firewall can access the Client Access server by using Outlook Anywhere (which uses RPC over HTTP). How ever, Outlook clients that are viewing or modifying public folders directly access the Mailbox server by using RPC over TCP.
6. The administrator-only computer retrieves Active Directory topology information from the Microsoft Exchange Active Directory Topology service. It also retrieves e-mail address policy information and address list information.
7. The Client Access server uses LDAP or Name Service Provider Interface (NSPI) to contact the Active Directory server and retrieve users' Active Directory information.

Argent for Exchange - Create CAS Test Mailbox

Create Test Mailboxes for OWA, ActiveSync, and Exchange Web Services Connectivity Monitoring.

The Argent for Exchange uses cmdlets to test Microsoft Office Outlook Web Access (OWA), Microsoft ActiveSync, and Exchange Web Services connectivity from Client Access servers to Mailbox servers.

The following expanded rules require the test CAS User accounts to be created.



These cmdlets require that a test mailbox be created on an Exchange Server 2010 Mailbox server that is to be tested. In this procedure you create test mailboxes for OWA, ActiveSync, and Exchange Web Services connectivity monitoring by using PowerShell to run the New-TestCasConnectivityUser.ps1 script.

The appropriate mailbox is created on each Mailbox server by piping in the results of get-mailboxServer.

To create a test mailbox for OWA, ActiveSync, and Exchange Web Services connectivity monitoring

1. Open Exchange PowerShell and change directory to the C:\Program Files\Microsoft\Exchange Server\v14\Scripts folder.
2. Execute the following command:
   get-mailboxServer | .\new-TestCasConnectivityUser.ps1.
3. Follow the on-screen installation instructions to complete the creation of the test mailbox.

NOTE: This process can be run from the Argent for Exchange MAIN Engine if Powershell and the Exchange Management Shell is Installed.

When trying to run new-TestCasConnectivityUser.ps1 to create some mailboxes for the test-cmdlets you may run into an error like the following error.

Mailbox could not be created. Verify that OU 'Users' exists and that password meets complexity require.

The issue may be caused by the following:

- A Users OU doesn't exist
- Or more than one OU Named Users Exists
- Password not complex enough

If you need to change the OU or testt to specify a different OU to store the Test Users in then change the following line in the new-TestCasConnectivityUser.ps1 powershell script.

Default OU entry

$OrganizationalUnit = "Users"

Example Modified OU entry

$OrganizationalUnit = "OU=Service Accounts, DC=ACME,DC=co,DC=nz"

Once the script has completed you can check where the mailbox was created.

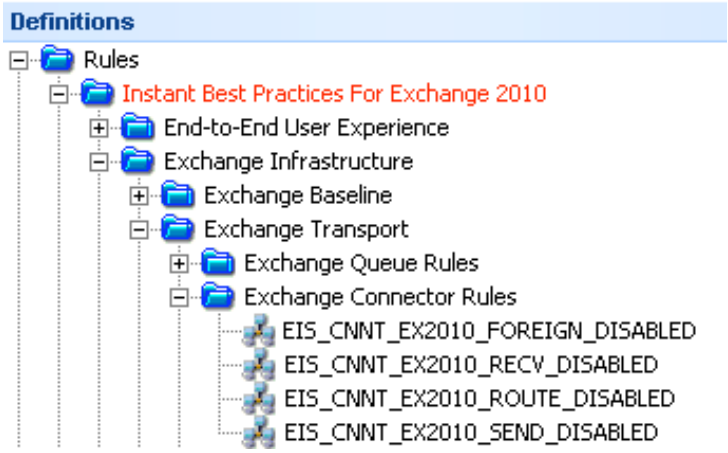
```
PS C > get-mailbox ext*

Name                 Alias                ServerName
----                 -----                ----------
extest_aa7372eb36134 extest_aa7372eb36134 mbx1
extest_dbc638da7d104 extest_dbc638da7d104 mbx2
```

NOTE: The script will only create one test mailbox per site.

Argent for Exchange – Infrastructure Rules

Exchange Connector Rule

Rule: EIS_CNNT_EX2010_SEND_DISABLED
Run On: Hub Transport



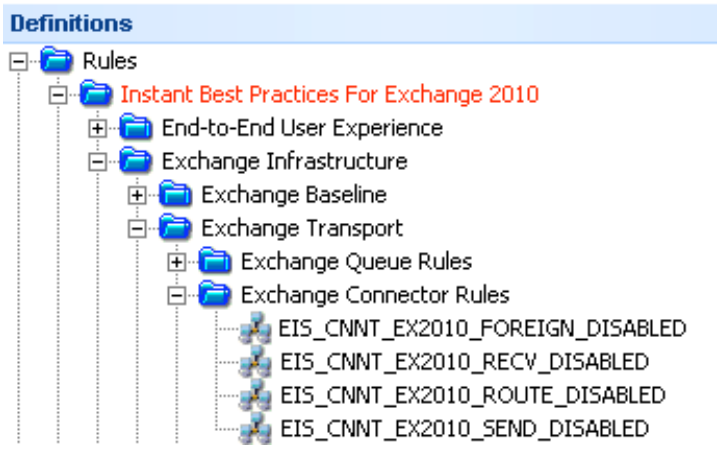
Normal Rule Results:

Connector 'pn-mspolicy' is enabled
Connector 'wn-mspolicy' is enabled

Rule Operation: This rule utilises the **Get-SendConnector** cmdlet to view the configuration information for a Send connector on a computer that has the Hub Transport server role or the Edge Transport server role installed, the rule is BROKEN if any SendConnector is Not Enabled.

```
[PS] C:\Windows\system32>Get-SendConnector

Identity                                AddressSpaces                           Enabled
--------                                -------------                           -------
wn-mspolicy                             {SMTP:*;10}                             True
pn-mspolicy                             {SMTP:*;1}                              True
```

Rule: EIS_CNNT_EX2010_RECV_DISABLED
Run On: Hub Transport

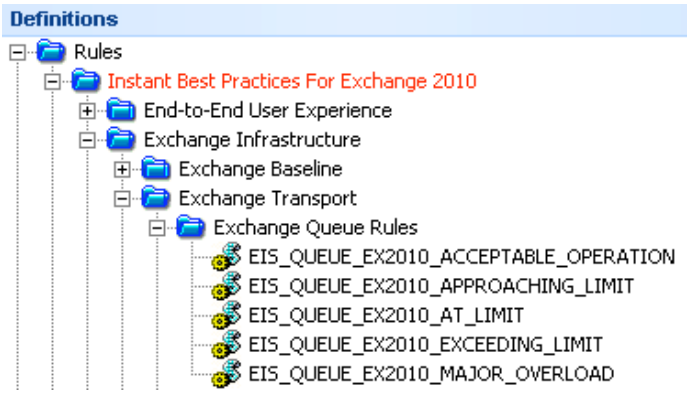


Normal Rule Results:

Connector 'HUBCAS1\Default WLGHUBCAS1' is enabled
Connector 'HUBCAS1\Client WLGHUBCAS1' is enabled
Connector 'HUBCAS1\App Connector WLGHUBCAS1' is enabled
Connector 'HUBCAS1\SMTP Relay WLGHUBCAS1' is enabled

Rule Operation: This rule utilises the **Get-ReceiveConnector** cmdlet to view the configuration information for a Receive connector on a computer that has the Hub Transport server role or the Edge Transport server role installed.

Rule: EIS_QUEUE_EX2010_*
Run On: Hub Transport



Normal Rule Results:

Message count of queue 'HUBCAS1\4085' = 0 (<= 30) - Not Broken
Status of queue 'HUBCAS1\4086' is Ready
Message count of queue 'HUBCAS1\4086' = 0 (<= 30) - Not Broken
Status of queue 'HUBCAS1\4090' is Ready
Message count of queue 'HUBCAS1\4090' = 0 (<= 30) - Not Broken
Status of queue 'HUBCAS1\Submission' is Ready
Message count of queue 'HUBCAS1\Submission' = 0 (<= 30) - Not Broken
Status of queue 'HUBCAS1\Shadow\3862' is Ready
Message count of queue 'HUBCAS1\Shadow\3862' = 1 (<= 30) - Not Broken
Status of queue 'HUBCAS1\Shadow\3863' is Ready
Message count of queue 'HUBCAS1\Shadow\3863' = 1 (<= 30) - Not Broken

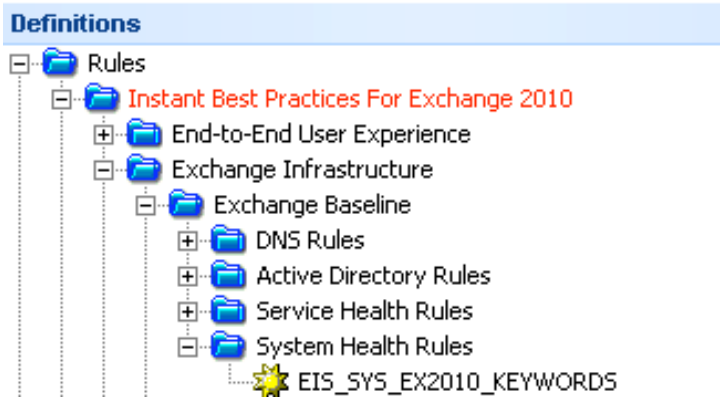
Rule Operation: This rule utilises the **Get-Queue** cmdlet to view configuration information for queues on a computer that has the Hub Transport server role or the Edge Transport server role installed.

```
[PS] C:\Windows\system32>get-queue -server wlghubcas1

Identity                                  DeliveryType Status MessageCount NextHopDomain
--------                                  ------------ ------ ------------ -------------
WLGHUBCAS1\4085                           MapiDelivery Ready  0            db11
WLGHUBCAS1\4086                           MapiDelivery Ready  0            db12
WLGHUBCAS1\Submission                     Undefined    Ready  0            Submission
WLGHUBCAS1\Shadow\3862                    ShadowRed... Ready  1            pmrhubcas1
WLGHUBCAS1\Shadow\3863                    ShadowRed... Ready  1            pmrhubcas2
```

The rule checks the queue is Ready and that the number of messages in each queue doe not exceed the threshold specified in the rule.

System Health Rule

Rule: EIS_SYS_EX2010_KEYWORDS
Run On: ExchangeServers



Normal Rule Results:

The maximum LDAP page size for the default query policy is set too high and may cause performance problems. The default of 1000 is recommended. Current value: 5000.

Rule Operation: This rule utilises the **Test-SystemHealth** cmdlet to gather data about your Microsoft Exchange system and to analyze the data according to best practices.

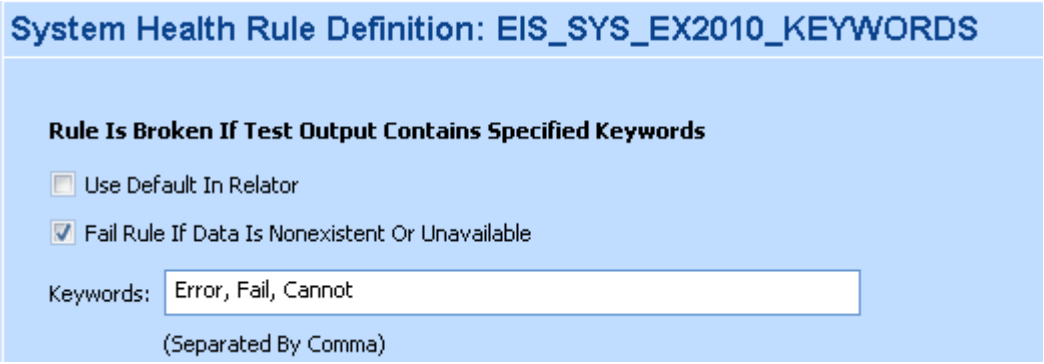
```
[PS] C:\Windows\system32>Test-SystemHealth -serverlist wlgmbx1
The maximum LDAP page size for the default query policy is set too high and may cause performance problems. The default
 of 1000 is recommended. Current value: 5000.
    + CategoryInfo          : NotSpecified: (fMaxPageSizeTooHigh:String) [Test-SystemHealth], BPAConfigurationErrorFou
   ndException
    + FullyQualifiedErrorId : D1B4AE8C,Microsoft.Exchange.Management.BestPracticesAnalyzer.TestSystemHealth
WARNING: The 'MaxQuorumLogSize' value on Exchange cluster WLGMBX1 is too small and may cause fail-over problems. The
recommended value is 4194304 (4096 KB). Current value: 1024.
WARNING: Cluster WLGMBX1 does not appear to have a dedicated heartbeat network connection. This may cause reliability
problems.
WARNING: More than 8 logical processors exist on Exchange server wlgmbx1.wan.net.nz. If this server has more than 8
physical processors installed, additional configuration tuning may be required. If this server is using hyper-threading
 technology, no action is required. Number of logical processors detected: 12.
WARNING: More than 8 logical processors exist on Exchange server WLGMBX2.wan.net.nz. If this server has more than 8
physical processors installed, additional configuration tuning may be required. If this server is using hyper-threading
 technology, no action is required. Number of logical processors detected: 12.
```

Example Powershell

This example gathers data about your Exchange server MBX1.

```
Test-SystemHealth –serverlist MBX1
```

The rule is setup by default to only check for certain keywords in the result.

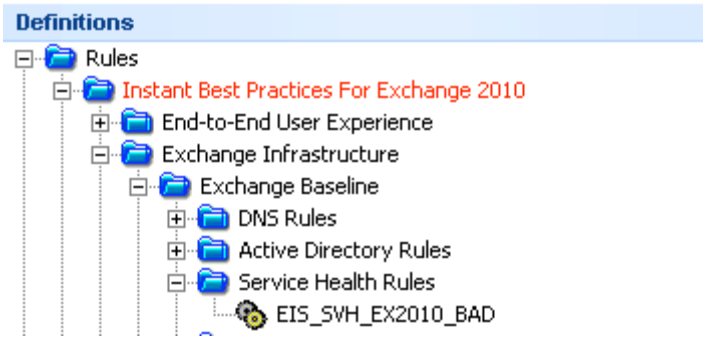


If WARNING was added then the yellow results shown in the Powershell on the previous page would appear in the rule results also.

Service Health Rule

Exchange Connector Rule

Rule: EIS_SVH_EX2010_BAD
Run On: ExchangeServers



Normal Rule Results:

Client Access Server

Following services required by server role 'Client Access Server Role' are running: IISAdmin MSExchangeAB MSExchangeADTopology MSExchangeFBA MSExchangeFDS MSExchangeMailboxReplication MSExchange ProtectedServiceHost MSExchangeRPC MSExchangeServiceHost W3Svc WinRM.

Hub Transport Server

Following services required by server role 'Hub Transport Server Role' are running: IISAdmin MSExchange ADTopology MSExchangeEdgeSync MSExchangeServiceHost MSExchangeTransport MSExchangeTrans portLogSearch W3Svc WinRM.

Mailbox Server

Following services required by server role 'Mailbox Server Role' are running: IISAdmin MSExchangeAD Topology MSExchangeIS MSExchangeMailboxAssistants MSExchangeMailSubmission MSExchangeRepl MSExchangeRPC MSExchangeSA MSExchangeSearch MSExchangeServiceHost MSExchangeThrottling MSExchangeTransportLogSearch W3Svc WinRM.

Rule Operation: This rule utilises the **Test-ServiceHealth** cmdlet to test whether all the Microsoft Windows services that Exchange requires on a server have started. The rule returns an error for any service required by a configured role when the service is set to start automatically and isn't currently running.

```
[PS] C:\Windows\system32>Test-ServiceHealth -server WLGMBX1


Role                    : Mailbox Server Role
RequiredServicesRunning : True
ServicesRunning         : {IISAdmin, MSExchangeADTopology, MSExchangeIS, MSExchangeMailboxAssistants, MSExchangeMailSub
                          mission, MSExchangeRepl, MSExchangeRPC, MSExchangeSA, MSExchangeSearch, MSExchangeServiceHost
                          , MSExchangeThrottling, MSExchangeTransportLogSearch, W3Svc, WinRM}
ServicesNotRunning      : {}
```

```
[PS] C:\Windows\system32>Test-ServiceHealth -server WLGHUBCAS1


Role                    : Client Access Server Role
RequiredServicesRunning : True
ServicesRunning         : {IISAdmin, MSExchangeAB, MSExchangeADTopology, MSExchangeFBA, MSExchangeFDS, MSExchangeMailbo
                          xReplication, MSExchangeProtectedServiceHost, MSExchangeRPC, MSExchangeServiceHost, W3Svc, Wi
                          nRM}
ServicesNotRunning      : {}

Role                    : Hub Transport Server Role
RequiredServicesRunning : True
ServicesRunning         : {IISAdmin, MSExchangeADTopology, MSExchangeEdgeSync, MSExchangeServiceHost, MSExchangeTranspo
                          rt, MSExchangeTransportLogSearch, W3Svc, WinRM}
ServicesNotRunning      : {}
```

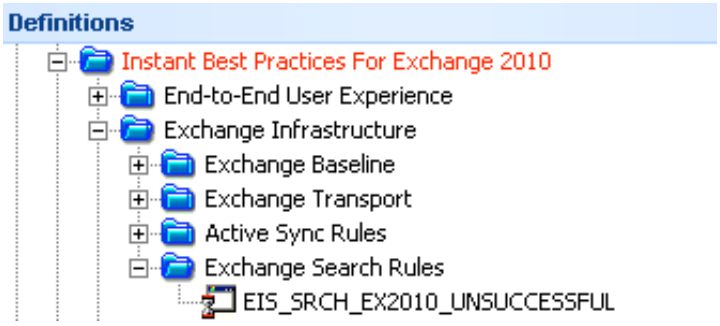
Example Powershell

Use the **Test-ServiceHealth** command without parameters to test the services on the local server.

```
Test-ServiceHealth
```

Exchange Search Rule

Rule: EIS_SRCH_EX2010_UNSUCCESSFUL
Run On: MailboxServer



Normal Rule Results:

Search test found the result - Not Broken
Search test found the result - Not Broken

Rule Operation: This rule utilises the **Test-ExchangeSearch** cmdlet to test that Exchange Search is currently enabled and is indexing new e-mail messages in a timely manner against each mailbox database (example output shown below).

```
[PS] C:\Windows\system32>Test-ExchangeSearch -server wlgmbx1

Database      Server        Mailbox       ResultFound SearchTime Error
--------      ------        -------       ----------- ---------- -----
DB11          WLGMBX1.w...  SystemMai...  True        3
DB12          WLGMBX2.w...  SystemMai...  True        3
```

The **rule** creates a hidden message and an attachment visible only to Exchange Search; the hidden message is stored in the System Attendant mailbox. The command waits for the message to be indexed and then searches for the content. It reports success or failure depending on whether the message is found.

Example Powershell

Tests Exchange Search results for the mailbox database on which the specified mailbox resides.

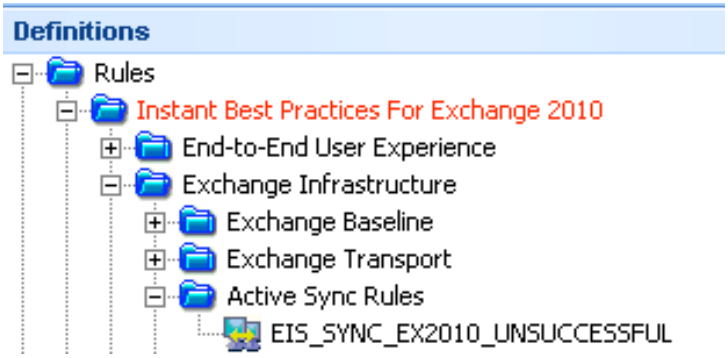
```
Test-ExchangeSearch -Identity john@contoso.com
```

Tests Exchange Search results for the mailbox database DB1 with an indexing time-out of 30 seconds.

```
Test-ExchangeSearch -MailboxDatabase "DB1"
-IndexingTimeout 30
```

Active Sync Rule

Rule: EIS_SYNC_EX2010_UNSUCCESSFUL
Run On: ClientAccessServer



Normal Rule Results:

Active Sync Test (Options) returns result 'Success'
Active Sync Test (FolderSync) returns result 'Success'
Active Sync Test (First Sync) returns result 'Success'
Active Sync Test (GetItemEstimate) returns result 'Success'
Active Sync Test (Sync Data) returns result 'Success'
Active Sync Test (Ping) returns result 'Success'
Active Sync Test (Sync Test Item) returns result 'Success'

Rule Operation: This rule utilises the **Test-ActiveSyncConnectivity** cmdlet to perform a full synchronization against a specified mailbox (extest_*) to test the configuration of Microsoft Exchange ActiveSync results shown below.

```
[PS] C:\Windows\system32>Test-ActiveSyncConnectivity -ClientAccessServer WLGHUBCAS1

CasServer   LocalSite     Scenario         Result   Latency(MS) Error
---------   ---------     --------         ------   ----------- -----
wlghubcas1  WLG           Options          Success        15.60
wlghubcas1  WLG           FolderSync       Success       218.40
wlghubcas1  WLG           First Sync       Success       218.40
wlghubcas1  WLG           GetItemEstimate  Success        62.40
wlghubcas1  WLG           Sync Data        Success       124.80
wlghubcas1  WLG           Ping             Success       853.20
wlghubcas1  WLG           Sync Test Item   Success        78.00
```

The **rule** performs a synthetic full synchronization between a mobile device and a specified mailbox to test the functionality of Exchange ActiveSync. If the synchronization fails, a failed message is displayed in the Exchange Management Shell and this will cause the rule to be BROKEN.

Example Powershell

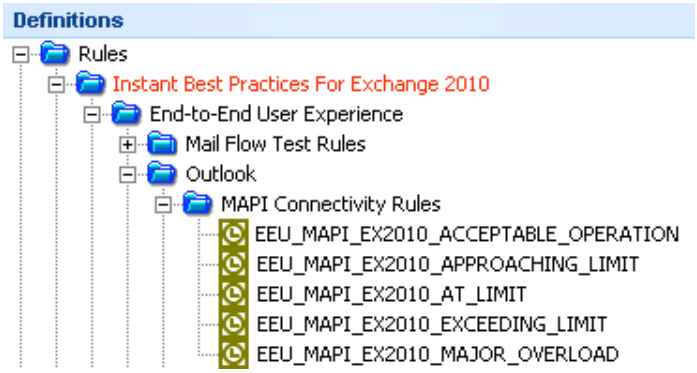
Tests Exchange ActiveSync connectivity for mailbox argent on Client Access server CAS01.

```
Test-ActiveSyncConnectivity -ClientAccessServer
CAS01 -URL “http://contoso.com/mail”
-MailboxCredential “argent”
```

Argent for Exchange – End-to-End User Experience Rules

MAPI Connectivity Rule

Rule: EEU_MAPI_EX2010_*
Runs On: MailboxServer



Normal Rule Results:

Latency of MAPI connectivity test to database 'DB11'
= 0.02 seconds (<= 15) - Not Broken.

Rule Operation: This rule utilises the **Test-MapiConnectivity** cmdlet to verify server functionality by logging on to the mailbox that you specify. If you don't specify a mailbox, the cmdlet logs on to the SystemMailbox on the database that you specify.

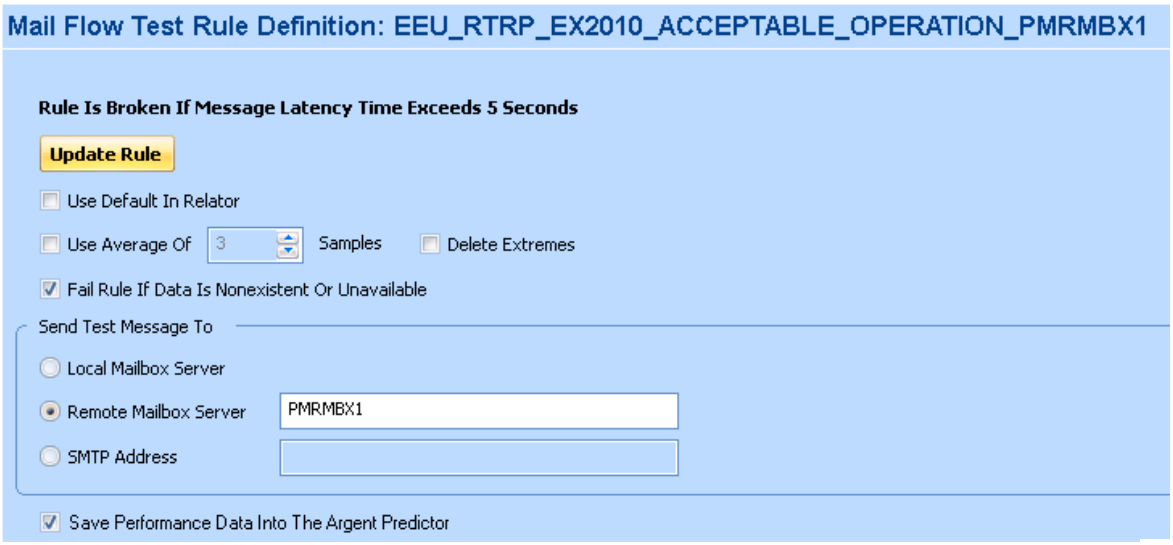
```
[PS] C:\Windows\system32>Test-MAPIConnectivity -server "WLGMBX1"

MailboxServer        Database             Result     Error
-------------        --------             ------     -----
WLGMBX1              DB11                 Success
```

The **rule** verifies server functionality. This cmdlet logs on to the mailbox that you specify (or to the SystemMailbox if you don't specify the *Identity* parameter) and retrieves a list of items in the Inbox.

The **rule** needs to access the mailbox that you specify using the credentials of the account with which you are logged on to the local computer. (Argent Service account needs to be an Exchange Administrator to have access to SystemMailboxes).

After a successful authentication, the **rule** accesses the mailbox to verify that the database is working. If a successful connection to a mailbox is made, the rule also determines the time taken for the logon attempt.

The rule can be modified to provide more granular testing using different mailboxes, databases and Domain Controllers (see Advanced Options Below).



There are three advanced Option parameters that you can use with the rule:

- The *Identity* parameter takes a mailbox identity and tests the ability to log on to a specific mailbox.
- The *Database Name* parameter tests the ability to log on to the system mailbox on the specified database.
- The *Domain Controller* parameter takes a server identity and tests the ability to log on to each system mailbox on the specified server.

Logging on to the mailbox tests two critical protocols used when a client connects to a Mailbox server: MAPI and LDAP. During authentication, the **Test-MapiConnectivity** cmdlet indirectly verifies that the MAPI server, Exchange store, and Directory Service Access (DSAccess) are working.

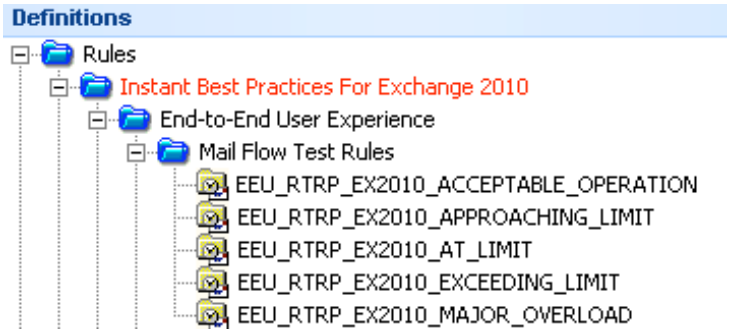
Example Powershell

This example tests connectivity to a mailbox, specified as a domain name and user name.

```
Test-MapiConnectivity -Identity "domain\argent"
```

Mail Flow Test Rule

Rule: EEU_RTRP_EX2010_*
Runs On: MailboxServer



Normal Rule Results:

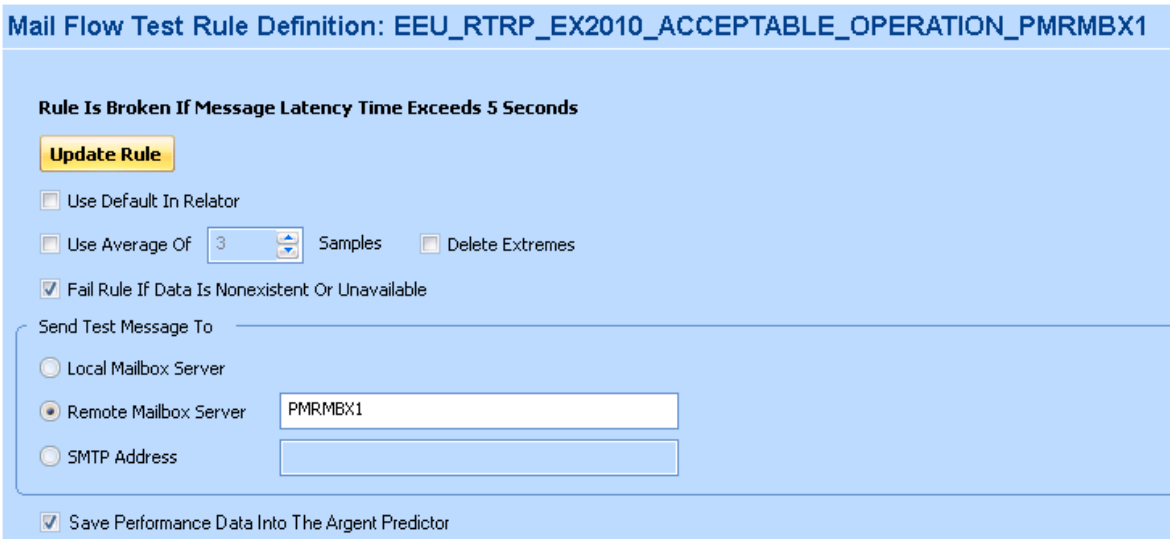
Latency of Round Trip Test = 1.70 seconds (<= 5) - Not Broken.

Rule Operation: This rule utilises the **Test-Mailflow** cmdlet to diagnose whether mail can be successfully sent from and delivered to the system mailbox on a computer that has the Mailbox server role installed. You can also use this rule to verify that e-mail is sent between Mailbox servers within a defined latency threshold.

```
[PS] C:\Windows\system32>Test-Mailflow WLGMBX1 -TargetMailboxServer PMRMBX2

RunspaceId          : 5533fe77-e14b-4884-90e4-ee3971f2ca26
TestMailflowResult  : Success
MessageLatencyTime  : 00:00:01.4933843
IsRemoteTest        : True
Identity            :
IsValid             : True
```

The **rule** tests mail submission, transport, and delivery. The rule can verify that each Mailbox server can successfully send itself a message. You can also use this rule to verify that the system mailbox on one Mailbox server can successfully send a message to the system mailbox on another Mailbox server. (A system mailbox must be present on all servers involved in the test.)

Send Test Message to options allow the test message to be sent using three options:



There are three Send Options that you can use with the rule:

- The *Local Mailbox Server* tests a mailbox server can send itself a message.
- The *Remote Mailbox Server* tests a mailbox server can send to another mailbox server.
- The *SMTP Address* tests that a mailbox server can send to an SMTP Address.

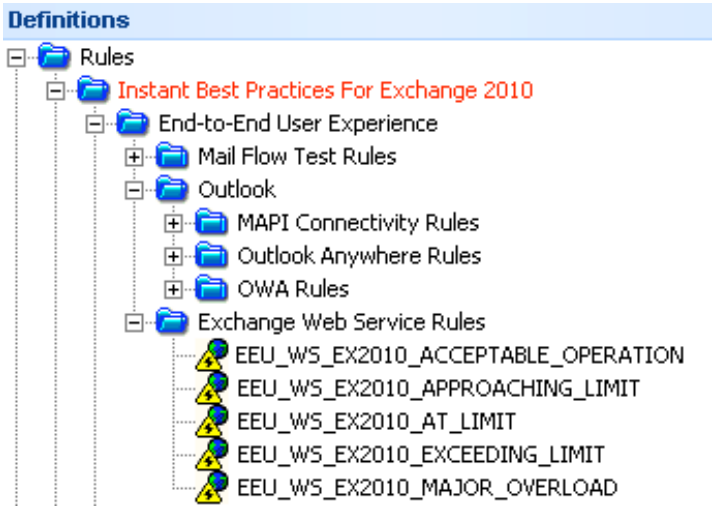
Example Powershell

This example sends test email from System Mailbox on MBX1 to SMTP email address.

```
Test-Mailflow MBX1 –TargetEmailAddress “argent@test.
com”
```

Exchange Web Services Rule

Rule: EEU_WS_EX2010_*
Runs On: ClientAccessServer



Normal Rule Results:

Latency of Web Service Connectivity Test (GetFolder) = 0.03 seconds (<= 15) - Not Broken
Latency of Web Service Connectivity Test (SyncFolderItems) = 0.08 seconds (<= 15) - Not Broken
Latency of Web Service Connectivity Test (CreateItem) = 0.08 seconds (<= 15) - Not Broken
Latency of Web Service Connectivity Test (SyncFolderItems) = 0.03 seconds (<= 15) - Not Broken
Latency of Web Service Connectivity Test (DeleteItem) = 0.06 seconds (<= 15) - Not Broken
Latency of Web Service Connectivity Test (SyncFolderItems) = 0.08 seconds (<= 15) - Not Broken

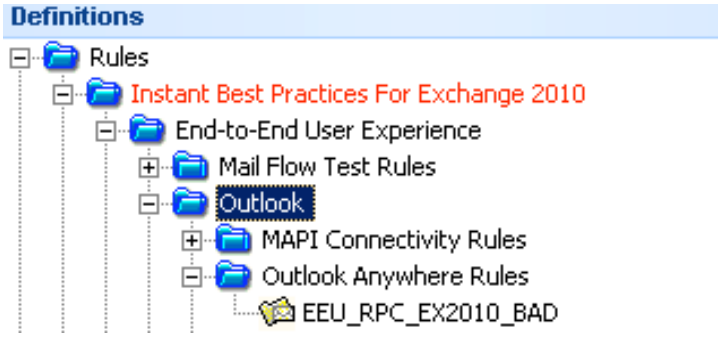
Rule Operation: This rule utilises the **Test-WebServicesConnectivity** cmdlet to perform basic operations to verify the functionality of Exchange Web Services on a server running Microsoft Exchange Server 2010 that has the Client Access server role installed.

```
[PS] C:\Windows\system32>Test-WebServicesConnectivity -clientaccessserver wlghubcas1

CasServer   LocalSite      Scenario         Result   Latency(MS) Error
---------   ---------      --------         ------   ----------- -----
wlghubcas1  WLG            GetFolder        Success        46.80
wlghubcas1  WLG            SyncFolderItems  Success       218.40
wlghubcas1  WLG            CreateItem       Success        46.80
wlghubcas1  WLG            SyncFolderItems  Success        31.20
wlghubcas1  WLG            DeleteItem       Success        46.80
wlghubcas1  WLG            SyncFolderItems  Success        46.80
```

The **rule** tests the functionality of Exchange Web Services. The rule performs basic operations to verify the functionality of Outlook Anywhere. The operations tested by default are GetFolder, CreateItem, DeleteItem, and SyncFolderItems.

Outlook Anywhere Rule

Rule: EEU_RPC_EX2010_BAD
Runs On: ClientAccessServer



Normal Rule Results:

Outlook Anywhere Test Result - Information (Event Id: 1019) - Not Broken
A valid Autodiscover service connection point was found. The Autodiscover URL on this object is https://HUBCAS1.test.net.nz/Autodiscover/Autodiscover.xml.

Outlook Anywhere Test Result - Information (Event Id: 1006) - Not Broken
Contacted Autodiscover service at https://HUBCAS1.test.net.nz/Autodiscover/Autodiscover.xml.

Outlook Anywhere Test Result - Information (Event Id: 1016) - Not Broken
[EXCH] The AS is configured for this user in the AutoDiscover response received from https://HUBCAS1.test.net.nz/Autodiscover/Autodiscover.xml.

Outlook Anywhere Test Result - Information (Event Id: 1015) - Not Broken
[EXCH] The OAB is configured for this user in the AutoDiscover response received from https://HUBCAS1.test.net.nz/Autodiscover/Autodiscover.xml.

Outlook Anywhere Test Result - Information (Event Id: 1014) - Not Broken
[EXCH] The UM is configured for this user in the AutoDiscover response received from https://HUBCAS1.test.net.nz/Autodiscover/Autodiscover.xml.

Outlook Anywhere Test Result - Information (Event Id: 1016) - Not Broken
[EXPR] The AS is configured for this user in the AutoDiscover response received from https://HUBCAS1.test.net.nz/Autodiscover/Autodiscover.xml.

Outlook Anywhere Test Result - Information (Event Id: 1015) - Not Broken
[EXPR] The OAB is configured for this user in the AutoDiscover response received from https://HUBCAS1.test.net.nz/Autodiscover/Autodiscover.xml.

Outlook Anywhere Test Result - Information (Event Id: 1014) - Not Broken
[EXPR] The UM is configured for this user in the AutoDiscover response received from https://HUBCAS1.test.net.nz/Autodiscover/Autodiscover.xml.

Outlook Anywhere Test Result - Success (Event Id: 1022) - Not Broken
Autodiscover was tested successfully.

Outlook Anywhere Test Result - Success (Event Id: 1024) - Not Broken
[EXCH] Successfully contacted the AS service at https://hubcas1.test.net.nz/EWS/Exchange.asmx. The elapsed time was 109 milliseconds.

Outlook Anywhere Test Result - Success (Event Id: 1026) - Not Broken
[EXCH] Successfully contacted the UM service at https://hubcas1.test.net.nz/EWS/Exchange.asmx. The elapsed time was 15 milliseconds.

Outlook Anywhere Test Result - Success (Event Id: 1124) - Not Broken
[Server] Successfully contacted the AS service at https://hubcas1.test.net.nz/ews/exchange.asmx. The elapsed time was 78 milliseconds.

Outlook Anywhere Test Result - Success (Event Id: 1126) - Not Broken
[Server] Successfully contacted the UM service at https://hubcas1.test.net.nz/ews/exchange.asmx. The elapsed time was 15 milliseconds.

Rule Operation: This rule utilises the **Test-OutlookWebServices** cmdlet to verify the Autodiscover service settings for Microsoft Outlook on a computer running Microsoft Exchange Server 2010 that has the Client Access server role installed.

```
[PS] C:\Windows\system32>Test-OutlookWebServices -clientaccessserver wlghubcas1


RunspaceId : 97b58bb5-e019-4fb5-b2d9-aa2137bfb8b1
Id         : 1019
Type       : Information
Message    : A valid Autodiscover service connection point was found. The Autodiscover URL on this object is https://WL
             GHUBCAS1.wan.net.nz/Autodiscover/Autodiscover.xml.
```

The **rule** uses a specified e-mail address (extest_*) to verify that the Outlook provider is configured correctly.

This rule verifies the service information that's returned to the Outlook client from the Autodiscover service for the Client Access Server Test user (extest_*). The rule verifies information for the following services:

- Availability service
- Outlook Anywhere
- Offline address book
- Unified Messaging

This example tests for a connection to each service. This example also submits a request to the Availability service for the test user to determine whether the user's free/busy information is being returned correctly from the Client Access server to the Outlook client.

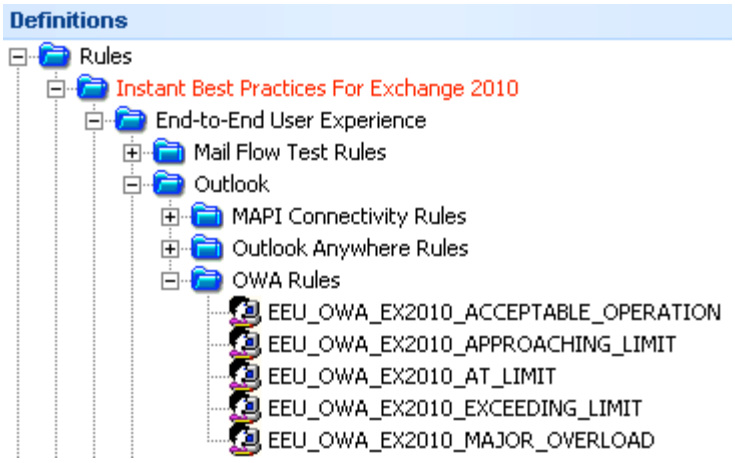
Example Powershell

This example checks Outlook Client service information for the user argent.

```
Test-OutlookWebServices -Identity:argent@test.local
```

OWA Rule

Rule: EEU_OWA_EX2010_*
Runs On: ClientAccessServer



Normal Rule Results:

Logon Latency of OWA connectivity test to virtual directory 'owa (Default Web Site) (https://webmail.test.net.nz/owa/)' = 0.14 seconds (<= 5) - Not Broken.

Rule Operation: This rule utilises the **Test-OwaConnectivity** cmdlet to verify that Microsoft Office Outlook Web App is running as expected. The rule can be used to test Outlook Web App connectivity for all Microsoft Exchange Server 2010 virtual directories on a specified Client Access server for all mailboxes on servers running Exchange that are in the same Active Directory site.

```
[PS] C:\Windows\system32>Test-OwaConnectivity -clientaccessserver wlghubcas1

ClientAccessServer MailboxServer URL                    Scenario Result  Latency Error
                                                                          (ms)
------------------ ------------ ---                     -------- ------  ------- -----
WLGHUBCAS1.wan.... WLGMBX2.wa... https://wlghubcas1.w... Logon    Success 514.80
```

To test all Outlook Web App virtual directories on a Client Access server, there must be a test Active Directory account. There must also be a test mailbox in each Active Directory site that hosts mailboxes that can be accessed through the virtual directories being tested.

If the server hosting the test mailbox isn't available, the **Test-OwaConnectivity** cmdlet returns an error that might not clearly identify the problem. To avoid this, check that the server that hosts the test mailbox is running and that the mailbox is available before you run the **Test-OwaConnectivity** cmdlet. You can use the **Test-MapiConnectivity** cmdlet to do this.

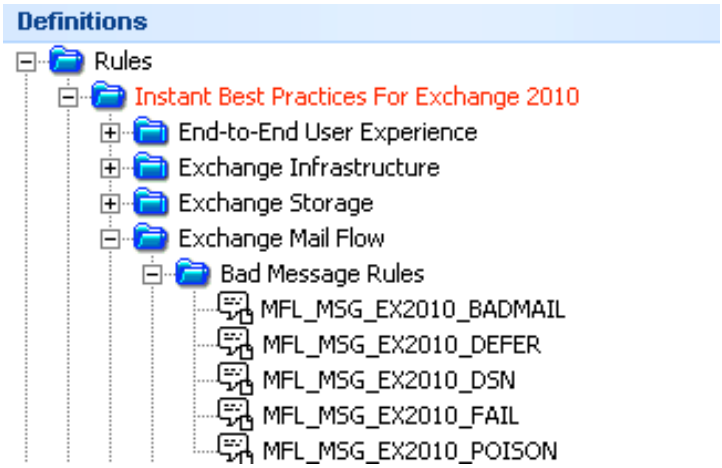
To test a single URL, run the **Test-OwaConnectivity** cmdlet with the URL parameter and credentials for an existing Exchange mailbox. If the URL is behind a load balancer, you can't predict which Client Access server the command will test. Because credentials are required as part of the parameters when you use the URL parameter, you can use any account to run the **Test-OwaConnectivity** cmdlet when you use the URL parameter.

If the command encounters a virtual directory that doesn't require Secure Sockets Layer (SSL), the command skips that directory unless the *AllowUnsecureAccess* parameter is used. If the *AllowUnsecureAccess* parameter is used, communications between servers are sent in clear text for purposes of the test.

Argent for Exchange – Exchange Mail Flow Rules

Bad Message Rules

Rule: MFL_MSG_EX2010_*
Runs On: Mailbox Server



Normal Rule Results:

Typically NONE this shows an example BAD Message.

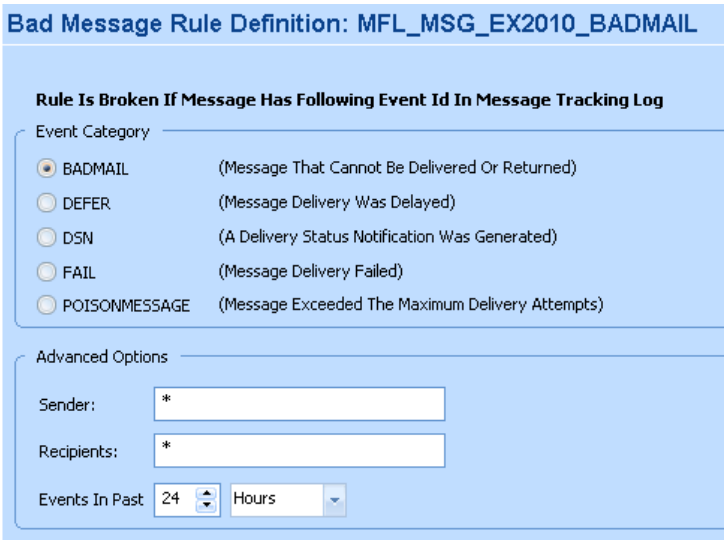
Bad Message (FAIL)

Message Id: <45922943-22010841983458131@adv.com>
Timestamp: 19 Aug 2010 20:35:22
Sender: newinfon@adv.com
Recipients: ray@tsit.co.nz
Subject: Nobody does

Rule Operation: This rule uses the **Get-MessageTrackingLog** cmdlet to search message information stored in the message tracking log. A unique message tracking log exists on each computer that has the Hub Transport server role, the Mailbox server role, or the Edge Transport server role installed. The message tracking log is a comma-separated value (CSV) file that contains detailed information about the history of each e-mail message as it travels through an individual server.

```
[PS] C:\Windows\system32>Get-MessageTrackingLog -start "08/19/2010 8:00PM" -Server WLGHUBCAS1 -EventId FAIL

EventId  Source   Sender                    Recipients                MessageSubject
-------  ------   ------                    ----------                --------------
FAIL     ROUTING  newinfon@adv.com          {ray@tsit.co.nz}          Nobody Does
```

There are 5 Event Category that can be check against for messages in the Tracking Log.



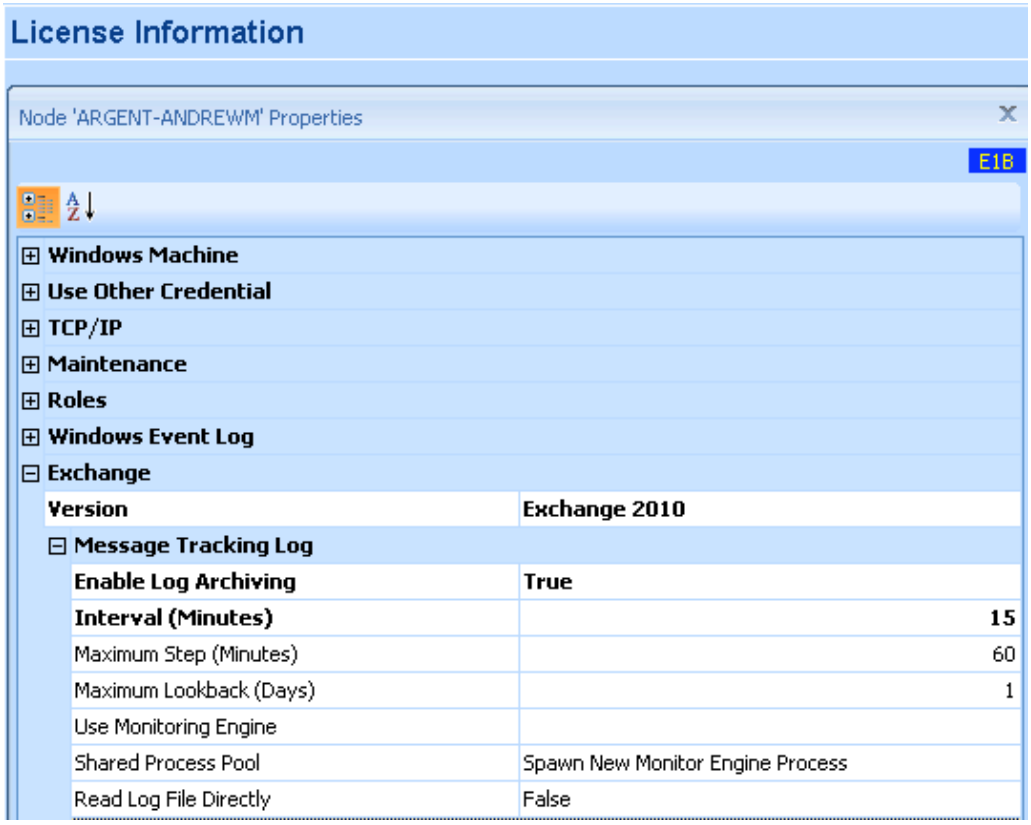
Further filtering can be achieved by specifying the sender and/or recipients and also how far back to scan the Exchange Message Tracking Log.

Example Powershell

This example uses the **Get-MessageTrackingLog** command to retrieve message tracking log entries that were created after Aug 19, 2010 at 08:00 from HUBCAS with an *EventID* parameter value of FAIL.

```
Get-MessageTrackingLog -Start “08/19/2010 8:00PM”
–Server HUBCAS –EventID FAIL
```

To utilise archiving of the exchange Message Tracking Log, configure the following within the Node Properties of the License Information screen (Control Information – Administration – License Manager).

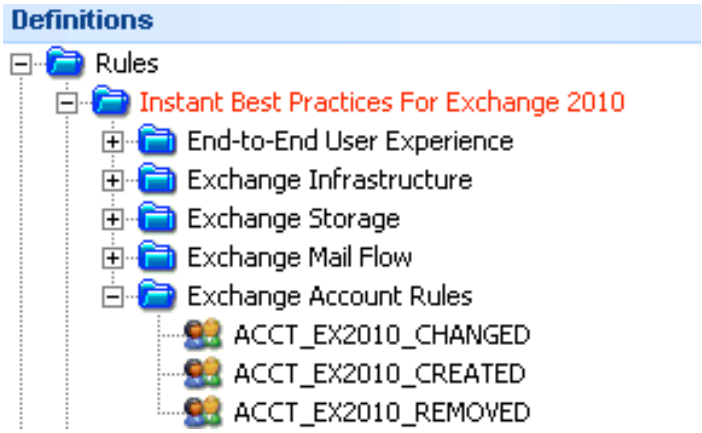


The Message Tracking Log data is stored in the ARGSOFT_EX_TRACKLOG table of the Argent AT database.

Argent for Exchange – Exchange Account Rules

Exchange Account Rules

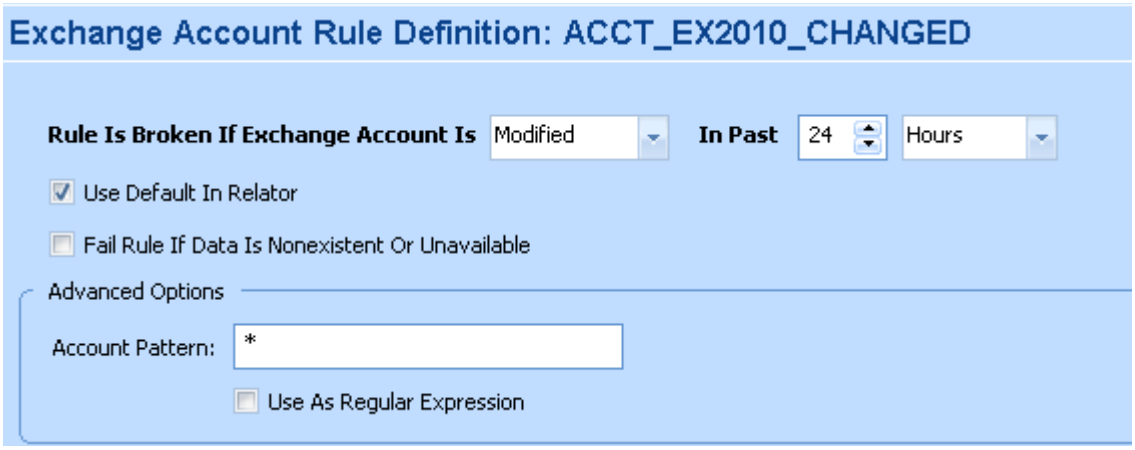
Rule: ACCT_EX2010_*
Runs On: Mailbox Server



Normal Rule Results:

Typically NONE.

Rule Operation: This rule allows an alert to be generated when a user is Created, removed or changed within a certain time windows as shown below.



Also the rule can filter the account names based on a pattern – only show accounts changed with the following pattern MEETING*

However to track accounts requires the **Enable Account Tracking** to be enabled on each exchange mailbox server.

This is done from Control Information – Administration – License manager.