Name create-message-security-provider – enables administrators to create a message security provider, which specifies how SOAP messages will be secured.

Synopsis

```
create-message-security-provider
[--help]

  --classname    provider_class
 [--layer    message_layer ] [--providertype provider_type ]
 [--requestauthsource   request_auth_source  ]
 [--requestauthrecipient   request_auth_recipient ]
 [--responseauthsource   response_auth_source  ]
 [--responseauthrecipient   response_auth_recipient ]
 [--isdefaultprovider] [ --property   name=value[:name=value]*  ]
 provider_name
```

Description The `create-message-security-provider` subcommand enables the administrator to create a message security provider for the security service which specifies how SOAP messages will be secured.

This command is supported in remote mode only.

Options If an option has a short option name, then the short option precedes the long option name. Short options have one dash whereas long options have two dashes.

`--help`
Displays the help text for the subcommand.

`--target`
Do not specify this option. This option is retained for compatibility with other releases. If you specify this option, a syntax error does not occur. Instead, the subcommand runs successfully and the option is silently ignored.

`--classname`
Defines the Java implementation class of the provider. Client authentication providers must implement the `com.sun.enterprise. security.jauth.ClientAuthModule` interface. Server-side providers must implement the `com.sun.enterprise.security jauth.ServerAuthModule` interface. A provider may implement both interfaces, but it must implement the interface corresponding to its provider type.

`--layer`
The message-layer entity used to define the value of the `auth-layer` attribute of `message-security-config` elements. The default is `HttpServlet`. Another option is `SOAP`.

`--providertype`
Establishes whether the provider is to be used as client authentication provider, server authentication provider, or both. Valid options for this property include `client`, `server`, or `client-server`.

`--requestauthsource`
The `auth-source` attribute defines a requirement for message-layer sender authentication (e.g. username password) or content authentication (e.g. digital signature) to be applied to request messages. Possible values are `sender` or `content`. When this argument is not specified, source authentication of the request is not required.

`--requestauthrecipient`
The `auth-recipient` attribute defines a requirement for message-layer authentication of the receiver of a message to its sender (e.g. by XML encryption). Possible values are `before-content` or `after-content`. The default value is `after-content`.

`--responseauthsource`
The `auth-source` attribute defines a requirement for message-layer sender authentication (e.g. username password) or content authentication (e.g. digital signature) to be applied to response messages. Possible values are `sender` or `content`. When this option is not specified, source authentication of the response is not required.

`--responseauthrecipient`
The `auth-recipient` attribute defines a requirement for message-layer authentication of the receiver of the response message to its sender (e.g. by XML encryption). Possible values are `before-content` or `after-content`. The default value is `after-content`.

`--isdefaultprovider`
The `default-provider` attribute is used to designate the provider as the default provider (at the layer) of the type or types identified by the `providertype` argument. There is no default associated with this option.

`--property`
Use this property to pass provider-specific property values to the provider when it is initialized. Properties passed in this way might include key aliases to be used by the provider to get keys from keystores, signing, canonicalization, encryption algorithms, etc.

The following properties may be set:

`security.config`	Specifies the location of the message security configuration file. To point to a configuration file in the `domain-dir/config` directory, use the system property `${com.sun.aas.instanceRoot}/config/`, for example: `${com.sun.aas.instanceRoot}/config/wss-server-config` The default is *domain-dir*`/config/wss-serverconfig-1.0.xml`.
`debug`	If `true`, enables dumping of server provider debug messages to the server log. The default is `false`.
`dynamic.username. password`	If `true`, signals the provider runtime to collect the user name and password from the `CallbackHandler` for

	each request. If `false`, the user name and password for `wsse:UsernameToken(s)` is collected once, during module initialization. This property is only applicable for a `ClientAuthModule`. The default is `false`.
`encryption.key.alias`	Specifies the encryption key used by the provider. The key is identified by its keystore alias. The default value is `s1as`.
`signature.key.alias`	Specifies the signature key used by the provider. The key is identified by its keystore alias. The default value is `s1as`.

Operands

provider_name
The name of the provider used to reference the `provider-config` element.

Examples

EXAMPLE 1 Creating a Message Security Provider

The following example shows how to create a message security provider for a client.

```
asadmin> create-message-security-provider
--classname com.sun.enterprise.security.jauth.ClientAuthModule
--providertype client mySecurityProvider
```

Exit Status

0	command executed successfully
1	error in executing the command

See Also

`delete-message-security-provider`(1), `list-message-security-providers`(1)

`asadmin`(1M)