NetApp™

Go further, faster™

NetApp Manageability SDK Training

PERL Programming

SDK PERL Interface - Core Classes

- NaElement
  - Encapsulates one level of an XML element.
  - Elements can be arbitrarily nested.
  - Elements have names, attributes (only used for results), values (always strings) and possibly children

- NaServer
  - Encapsulates an administrative connection to a NetApp filer

SDK PERL Interface – Core Methods

- NaElement
  - set_content()
    - Sets the value of the element
  - child_*()
    - Provide operations on child elements – add, get etc
  - results_ *()
    - Provide operations on output element – status, reason, errno etc
- NaServer
  - set_*()
    - Provide operations for setting connection parameters – port, login style, server type, transport type, admin user etc
  - get_*()
    - Provide operations for getting connection parameters – port, login style, server type, transport type, admin user etc
  - invoke_*()
    - Provide operations for sending the command to the server

Configuring Session Parameters

- Server type
  - Indicates the target for API calls
  - It can be set using the method *set_server_type()*
  - parameter to *set_server_type()* method can take the following values:
    - *Filer*
      - To send API calls to NetApp Storage system
    - *DFM*
      - To send API calls to DataFabric Manager server

Configuring Session Parameters

- Server style
  - Indicates the authentication mechanism to be used for communicating with the Server
  - It can be set using the method *set_style()*
  - parameter to *set_style()* method can take the following values:
    - *LOGIN*
      - Causes the Server to use HTTP simple authentication with a username and password.
      - User needs to provide login/password details with *set_admin_user()* method
    - *HOSTS*
      - Causes the Server to authenticate against the /etc/hosts.equiv file on the filer

Configuring Session Parameters

- Transport type
  - Indicates the transport mechanism to be used for sending API commands
  - It can be set using the method *set_transport_type()*
  - Parameter to *set_transport_type()* method can take the following values:
    - HTTP
      - Default transport type
    - HTTPS
      - Provides secure communication by encrypting the API requests/responses

Configuring Session Parameters

- Server Port
  - Indicates the port on/to which API requests to be sent
  - Depends on ‘Transport type’ and ‘Server type’ settings
  - It can be set using the method *set_port()*
  - This setting should be done only after setting ‘Transport type’ and ‘Server type’ values

PERL Programming with SDK – Basic Steps

- STEP 1: Create Server Context on Client
  - Create Server Object with Server Name / IP, Product API version to be used as inputs
    - $s = NaServer->new($server-name, $majorversion, $minorversion)

- STEP 2: Set Session parameters on Client
  - Set server type : *$s->set_server_type()*
  - Set Authentication style : *$s->set_style()*
  - Set Transport type : *$s->set_transport_type()*
  - Set Server port : *$s->set_port()*

PERL Programming with SDK – Basic Steps

- STEP 3: Send Commands to the Server
  - Create a Command element and use the Core API *NaServer::invoke_elem()*

    E.g.

    ```
    $elem = NaElement->new("system-get-version");
    $s->invoke_elem(elem);
    ```

    OR

  - Send the Command as a parameter & its arguments as name/value pairs to the Core API *NaServer::invoke()*

    E.g.
    ```
    $s->invoke("snapshot-create",
      "snapshot", $ssname,
     "volume", $vol );
    ```

PERL Programming with SDK – Basic Steps

- STEP 4: Check for API Failures
  - Core API failure

    - *NaServer* class methods for setting session parameters (*set_*()*) return *NaElement* object in case of an error

      E.g.

```
$out = $s->set_transport_type();
if (ref ($out) eq “NaElement”) {
        if ($out->results_errno != 0) {
                my $r = $out->results_reason();
                print "Failed: $r\n";
        }
}
```

    - *NaElement* class methods for extracting the child element details return ‘undef’ in case of an error

      E.g.

```
$out = $elem->child_get_string();
if (! defined($out)) {
        print "Failed to get the element field\n";
}
```

PERL Programming with SDK – Basic Steps

- Product API failure
  - *results_*()* APIs of NaElement class should be used to determine the Product API error

    E.g.;

```
$out = $s->invoke("system-get-version");
if ($out->results_errno != 0) {
        my $r = $out->results_reason();
        print "Failed: $r\n";
}
```

- STEP 5: Parse the command results
  - The invoke*() Core APIs always return NaElement object
  - The result of Product API is embedded as a child NaElement object in the NaElement object returned from invoke*() Core APIs

    E.g.:

```
$out = $s->invoke("quota-list-entries");
$quota_info = $out->child_get("quota-entries");
@result = $quota_info->children_get();
```

Compiling SDK PERL Program

- The PERL libraries for the SDK are located at %sdk-root%/lib/perl/NetApp

- Make sure that *'use lib %sdk-root%/lib/perl/NetApp'* is in the program's @INC path.

SDK PERL Interface – Differences from other interfaces

- Provides automatic clean up of the resources on client at the end of the program
- Windows RPC style authentication mechanism is not supported
- No option to set timeout for the client connection in cases of delayed response from the server

Tutorial Exercises

Tutorial Exercises Contd..

1. Monitor volumes on a NetApp storage system and print message when space usage crosses a threshold

- Hint:
  - Get list of volumes with the command '*volume-list-info*'
  - Extract values of '*size_total*' and '*size_used*' fields from '*volume_info*' return element

Tutorial Exercises Contd..

2. Retrieve the quota information of a NetApp storage system in a secure way and print quota information : quota-target,vol name,quota-type

- Hint:
  - Set the transport type as ‘TRANSPORT_TYPE_HTTPS’
  - Get the quota list through the command ‘*quota-list-entries*’

Tutorial Exercises Contd..

3. Enable NFS server on NetApp storage system and check the status of the NFS server. Get the list of the export rules on the storage system.

- Hint:
  - Use the command '*nfs-enable*' for enabling NFS server
  - Use the command '*nfs-status*' to check the status
  - Use the command '*nfs-exportfs-list-rules*' to get the list of the NFS export rules

Thank You