XAM Mandated Fields

Finnbarr P. Murphy

(fpm@fpmurphy.com)

In this post I look at what fields are mandated by the SNIA XAM v1.0 specification and write a small Java application to retrieve their default values using the XAM Reference VIM and EMC's Centera XAM VIM.

What is meant by a field in XAM? According to Section 3.1.5 of the XAM specification v1.0, Part 1, a field is

> a piece of uniquely identifiable data that can be attached to an XSet, an XSystem, or a XAM Library.

More concretely, a field has a name, a number of attributes that describe how to interact with the object, and a value. Any XAM primary object, i.e. an XSystem, an XSet, or a XAM Library object can contain one or more fields.

Field names are case sentitive UTF-8 encoded strings with a maximum length of 512 bytes and no embedded NULL characters. To avoid namespace clashes, the field namespace is allocated between SNIA, XAM storage system vendors and XAM application vendors. The following table shows the currently reserved namespace for field names:

NAMESPACE	DESCRIPTION
.xam.*	The XAM Library-owned portion of the namespace. Fields in this namespace shall be defined in this specification and its follow-ons and shall not be extended by XAM Storage System vendors.
.xsystem.*	The XSystem-owned portion of the namespace. Fields in this namespace shall be defined in this specification and its follow-ons and shall not be extended by XAM Storage System vendors.
.xset.*	The XSet-owned portion of the namespace. Fields in this namespace shall be defined in this specification and its follow-ons and shall not be extended by XAM Storage System vendors.
.vnd..*	The XAM System vendor-owned namespace within the XSystem namespace, where is the XAM Storage System vendor's reverse DNS name
org.snia.*	Reserved for SNIA
org.snia.xam.*	Reserved for SNIA FCWG

To avoid field namespace clashes between XAM storage system vendors in the remaining unreserved namespace and aleviate the need for a central XAM field name registry, the first portion of a vendor field name shall be the vendor's domain name in reverse order, followed by the vendor-defined field name, e.g. *com.emc.centera.xam.vim.version*.

As mentionly previously a field can have attributes. The following four attributes are mandated by the XAM specification:

ATTRIBUTE NAME	DESCRIPTION

ATTRIBUTE NAME	DESCRIPTION
Type	The MIME type of the value. The type attribute shall be US-ASCII encoded with a maximum length of 512 bytes.
Binding	A Boolean value indicating if the field is bound to the XUID of the XSet.
Readonly	A Boolean value indicating if the field is protected against modification by the standard field operations.
Length	The length of the value in bytes.

Two distinct types of fields exist: a property and an XStream. For the purposes of this post, we are only interested in fields which are properties as defined in section 3.1.12 of the XAM specification, i.e

> a field whose MIME type attribute is one of the XAM-defined simple types (stypes).

Here is a list of the specified XAM stypes:

NAME	MIME TYPE STRING	DESCRIPTION	LENGTH ATTRIBUTE VALUE
xam_boolean	application/vnd.snia.xam.boolean	either TRUE or FALSE	1 byte
xam_int	application/vnd.snia.xam.int	A signed twos complement 64-bit integer	8 bytes
xam_double	application/vnd.snia.xam.double	An IEEE754 double-precision floating point number	8 bytes
xam_string	application/vnd.snia.xam.string	A UTF-8 encoded string (maximum length of 512 bytes)	Actual length in bytes
xam_datetime	application/vnd.snia.xam.datetime	Shall be a UTF-8 encoded timestamp identifier per ISO8601 as profiled by the XAM specification	Actual length in bytes
xuid	application/vnd.snia.xam.xuid	An XUID	Actual length in bytes (9 - 80)

Turning now to the two objects that we are interested in, namely the XAM Library object and the XSystem object.

XAM Library properties are always nonbinding i.e. a change in the property value does not trigger the creation of a new XSet with a corresponding new XUID. Some may be readonly and intended to be only inspected by the application. Others such as *.xam.log.level* may be modifiable by a XAM application to effect a change in the behavior of the XAM Library object. Note however that changes are not persisted by the XAM Library.

Here is the list of the mandated properties for the XAM Library:

FIELD NAME	TYPE	BINDING	READONLY
.xam.identity	xam_string	FALSE	TRUE
.xam.log.level	xam_int	FALSE	FALSE
.xam.log.verbosity	xam_int	FALSE	FALSE
.xam.log.path	xam_string	FALSE	FALSE

FIELD NAME	TYPE	BINDING	READONLY
.xam.apiLevel	xam_string	FALSE	TRUE
.xam.vim.list.	xam_string	FALSE	TRUE

An XSystem property is strongly typed to help XAM application interoperability. The stype is checked and the actual MIME type is set based on the specific method that the XAM application uses to create the field. A number of other field consistency checks are also done by an XSystem including checking that field names do not begin with a period, are a valid *xam_string* and do not have embedded NULLs. See section 6.3.5 of the XAM specification for a complete description of the field consistency checks.

Here is the list of mandated properties for an XSystem object:

FIELD NAME	TYPE	BINDING	READONLY
.xsystem.identity	xam_string	FALSE	TRUE
.xsystem.time	xam_datetime	FALSE	TRUE
.xsystem.limits.maxFieldsPerXSet	xam_int	FALSE	TRUE
.xsystem.limits.maxSizeOfXStream	xam_int	FALSE	TRUE
.xsystem.auth.SASLmechanism.list.	xam_boolean	FALSE	TRUE
.xsystem.auth.SASLmechanism.default	xam_string	FALSE	TRUE
.xsystem.auth.granule.list.	xam_boolean	FALSE	TRUE
.xsystem.thentication	xam_string	FALSE	TRUE
.xsystem.auth.identity.authorization	xam_string	FALSE	TRUE
.xsystem.auth.expiration	xam_int	FALSE	TRUE
.xsystem.access	xam_boolean	FALSE	TRUE
.xsystem.access.policy.list.	xam_string	FALSE	TRUE
.xsystem.job.commit.supported	xam_boolean	FALSE	TRUE
.xsystem.job.xam.job.query.continuance.supported	xam_boolean	FALSE	TRUE
.xsystem.job.xam.job.query.level1.supported	xam_boolean	FALSE	TRUE
.xsystem.job.xam.job.query.level2.supported	xam_boolean	FALSE	TRUE
.xsystem.retention.enabled.policy.list.	xam_string	FALSE	TRUE
.xsystem.retention.duration.policy.list.	xam_string	FALSE	TRUE
.xsystem.deletion.autodelete	xam_boolean	FALSE	TRUE
.xsystem.deletion.autodelete.policy.list.	xam_string	FALSE	TRUE
.xsystem.deletion.shred	xam_boolean	FALSE	TRUE
.xsystem.storage.policy.list.	xam_string	FALSE	TRUE
.xsystem.management.policy.list.	xam_string	FALSE	TRUE
.xsystem.management.policy.default	xam_string	FALSE	TRUE

Here is a simple Java application which iterates over the list of available fields and output the fields which match the specified field prefix or all fields if no prefix is entered. It uses the XAM *FieldIterator* class to retrieve the requested fields from either the SNIA-XAM SDK reference VIM or a third party VIM such as the EMC Centera VIM.

```
import java.io.ByteArrayOutputStream;
import java.io.BufferedOutputStream;
import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
```



```
import java.util.Calendar;
import java.text.SimpleDateFormat;
import java.util.Properties;

import org.snia.xam.XAMLibrary;
import org.snia.xam.XAMLibraryObj;
import org.snia.xam.XAMException;
import org.snia.xam.XSystem;
import org.snia.xam.XSet;
import org.snia.xam.XIterator;
import org.snia.xam.toolkit.XAMXUID;
import org.snia.xam.util.XAMLibraryFactory;
import org.snia.xam.FieldContainer;

import org.snia.xam.XUID;
import org.snia.xam.base.XAMImplementation;
import org.snia.xam.util.SASLUtils;
import org.snia.xam.vim.reference.ReferenceAuthenticationStatus;
import org.snia.xam.vim.reference.utils.ReferenceSaslUtils;

public class XamFieldIterator {
    private static XAMLibrary xamLib;
    private static XSystem xSystem;
    protected static String xri;

    public static boolean  IS_REFERENCE_VIM = false;
    public static final String TEST_PROP_FILE = "xam.test.props";
    public static final String XRI_PROP  = "xam.test.xri";
    public static final String CONFIG_PROP = "xam.test.vims";
    public static final String USER_PROP = "xam.test.username";
    public static final String PASS_PROP = "xam.test.password";
    protected static final String DEFAULT_USER = "test";
    protected static final String DEFAULT_PASS = "test";
    protected static String s_pass;
    protected static String s_user;

    public static void initLibrary() throws Exception {
        if (xamLib == null) {
            System.out.println("\nInitializing VIM");
            Properties props = new Properties();
            String testPropFile = System.getProperty(TEST_PROP_FILE);
            if (testPropFile == null)
                testPropFile = TEST_PROP_FILE;

            System.out.println("Loading test properties from file: " + testPropFile);
            props.load(new FileInputStream(testPropFile));

            xri = props.getProperty(XRI_PROP);
            s_user = props.getProperty(USER_PROP, DEFAULT_USER);
            s_pass = props.getProperty(PASS_PROP, DEFAULT_PASS);

            // XAM Library loads the VIM name and associated Java implementation class
            System.out.println("Loading the VIM using the Java XAM Library.");
            System.out.println("VIM Configuration contained in file: " +
                    props.getProperty(CONFIG_PROP));
            xamLib = new XAMImplementation(props.getProperty(CONFIG_PROP));
        }
    }

    private static XSystem connectToVIM(String xri) throws Exception {
        XSystem xsystem = null;
        xsystem = xamLib.connect(xri);
        return xsystem;
    }

    private static void authenticate(XSystem system) throws XAMException {
        String defMech = system.getString(XSystem.XAM_XSYSTEM_AUTH_SASL_DEFAULT);
        ByteArrayOutputStream response = new ByteArrayOutputStream(200);
```



```
        byte[] inputData = null;
        int retValue = 0;

        if (defMech.equals(ReferenceSaslUtils.SASL_MECHANISM_ANONYMOUS)) {
            retValue = system.authenticate(inputData, response);
        } else if (defMech.equals(SASLUtils.SASL_PLAIN)) {
            byte[] creds = ReferenceSaslUtils.encodeSASLPlain(null,
                    ReferenceAuthenticationStatus.TEST_USERNAME,
                    ReferenceAuthenticationStatus.TEST_PASSWORD);
            retValue = system.authenticate(creds, response);
        } else {
            throw new XAMException("Unknown default SASL mechanism " + defMech);
        }

        // Validate return status value
        if (retValue != XSystem.XAM_SASL_COMPLETE) {
            throw new XAMException("Failed to authenticate.");
        }
    }

    /* a FieldContainer is a XAM superclass for the 3 primary objects (XAMLibrary, XSystem,
 XSet) */
    private static void iterateFields(FieldContainer fieldContainer, String prefix) throws
 XAMException {
        String fieldName;
        String fieldType;
        String fieldValue;
        String fieldBinding;
        String fieldReadOnly;

        XIterator xIterator = fieldContainer.openFieldIterator(prefix);
        while (xIterator.hasNext()) {
            fieldName = (String)xIterator.next();
            fieldType = fieldContainer.getFieldType(fieldName);

            fieldBinding = fieldContainer.getFieldBinding(fieldName) != true ? "NB" : "BI"
;
            fieldReadOnly =  fieldContainer.getFieldReadOnly(fieldName) != true ? "RW" : "
RO";

            if (fieldType.equals(XAMLibrary.STYPE_BOOLEAN_MIME_TYPE))
                fieldValue = fieldContainer.getBoolean(fieldName) != true ? "false" : "tru
e";
            else if (fieldType.equals(XAMLibrary.STYPE_INT_MIME_TYPE))
                fieldValue = Long.toString(fieldContainer.getLong(fieldName));
            else if (fieldType.equals(XAMLibrary.STYPE_DOUBLE_MIME_TYPE))
                fieldValue = Double.toString(fieldContainer.getDouble(fieldName));
            else if (fieldType.equals(XAMLibrary.STYPE_XUID_MIME_TYPE))
                fieldValue = fieldContainer.getXUID(fieldName).toString();
            else if (fieldType.equals(XAMLibrary.STYPE_STRING_MIME_TYPE))
                fieldValue = fieldContainer.getString(fieldName);
            else if (fieldType.equals(XAMLibrary.STYPE_DATETIME_MIME_TYPE)) {
                Calendar now = fieldContainer.getDateTime(fieldName);
                int Y = now.get(Calendar.YEAR);
                int M = now.get(Calendar.MONTH);
                int D = now.get(Calendar.DAY_OF_MONTH);
                int h = now.get(Calendar.HOUR_OF_DAY);
                int m = now.get(Calendar.MINUTE);
                int s = now.get(Calendar.SECOND);
                fieldValue = Y + "-" + M + "-" + D + " " + h + ":" + m + ":" + s;
            } else
                fieldValue = fieldContainer.getFieldLength(fieldName) + " bytes";

            System.out.println(String.format("%s   (%s) %s %s   \"%s\"",
                 fieldName, fieldType, fieldBinding, fieldReadOnly, fieldValue));
        }
        xIterator.close();
    }
```



```
    public static void main(String[] args) {
        long exitCode = 0;

        InputStreamReader inputReader = new InputStreamReader(System.in);
        BufferedReader stdin = new BufferedReader(inputReader);

        try {
            /* simple check for command line option of "-r" */
            if (args.length == 1 ) {
               if (args[0].equals("-r"))
                  IS_REFERENCE_VIM = true;
            }

            if (IS_REFERENCE_VIM) {
                initLibrary();
                System.out.println("Connecting to XSystem " + xri + "\n");
                xSystem = connectToVIM(xri);
                authenticate(xSystem);
            } else {
                xri = "snia-xam://centera_vim!128.221.200.60?/home/fpm/xam/xamconnect.pea"
;
                xamLib = XAMLibraryFactory.newXAMLibrary();
                System.out.println("Connecting to XSystem " + xri + "\n");
                xSystem = xamLib.connect(xri);
            }

            System.out.print("Enter prefix to filter results (blank for all): ");
            String prefix = stdin.readLine();

            iterateFields(xSystem, prefix);

            xSystem.close();
            System.out.println( "\nClosed connection to XSystem" );

        } catch (XAMException xe) {
            System.err.println("XAM ERROR: " + xe.getMessage());
            exitCode = 1;
        } catch (IllegalArgumentException e) {
            System.out.println(e.getMessage());
            e.printStackTrace();
            exitCode = 1;
        } catch (IOException e) {
             System.err.println("IO ERROR: " + e.getMessage());
             e.printStackTrace();
             exitCode = 1;
        } catch (Exception ex) {
             ex.printStackTrace();
             exitCode = 1;
        }

        System.exit((int) exitCode);
    }
}
```

Here is the set of fields, i.e those fields which start with *.xam.* for the XAMLibrary object when the application is connected to an EMC Centera using EMC's XAM VIM. It is followed by the set of fields, i.e. those fields starting with *.xsystem.,* for an XSystem object which we create using this XAMLibrary object.

```
$ java XamFieldIterator
Connecting to XSystem snia-xam://centera_vim!XXX.XXX.XXX.XXX?/home/fpm/xam/xamconnect.pea

Enter prefix to filter results (blank for all): .xam

.xam.log.component.filter   (application/vnd.snia.xam.string) NB RW   ""
.xam.log.message.filter   (application/vnd.snia.xam.string) NB RW   ""
```



```
.xam.log.max.rollovers   (application/vnd.snia.xam.int) NB RW   "1"
.xam.log.max.size   (application/vnd.snia.xam.int) NB RW   "1048576"
.xam.log.append   (application/vnd.snia.xam.boolean) NB RW   "false"
.xam.log.format   (application/vnd.snia.xam.int) NB RW   "1"
.xam.log.path   (application/vnd.snia.xam.string) NB RW   "xam.log"
.xam.log.verbosity   (application/vnd.snia.xam.int) NB RW   "0"
.xam.log.level   (application/vnd.snia.xam.int) NB RW   "0"

Closed connection to XSystem

$ java XamFieldIterator
Connecting to XSystem snia-xam://centera_vim!XXX.XXX.XXX.XXX?/home/fpm/xam/xamconnect.pea

Enter prefix to filter results (blank for all): .xsystem

.xsystem.auth.identity.authorization   (application/vnd.snia.xam.string) NB RO   "xam_chal
lenge"
.xsystem.auth.identity.authentication   (application/vnd.snia.xam.string) NB RO   "xam_cha
llenge"
.xsystem.limits.maxSizeOfXStream   (application/vnd.snia.xam.int) NB RO   "107374182400"
.xsystem.limits.maxFieldsPerXSet   (application/vnd.snia.xam.int) NB RO   "922337203685477
5807"
.xsystem.identity   (application/vnd.snia.xam.string) NB RO   "EMC Centera, ID# 34372862-1
dd2-11b2-aea0-f013836b5e75"
.xsystem.auth.SASLmechanism.list.ANONYMOUS   (application/vnd.snia.xam.boolean) NB RO   "t
rue"
.xsystem.auth.SASLmechanism.list.PLAIN   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.SASLmechanism.default   (application/vnd.snia.xam.string) NB RO   "ANONYMOUS"
.xsystem.time   (application/vnd.snia.xam.datetime) NB RO   "2009-5-28 15:10:42"
.xsystem.auth.granule.list.read   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.granule.list.write-application   (application/vnd.snia.xam.boolean) NB RO   "
true"
.xsystem.auth.granule.list.write-system   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.granule.list.create   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.granule.list.delete   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.granule.list.job   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.granule.list.job-commit   (application/vnd.snia.xam.boolean) NB RO   "false"
.xsystem.auth.granule.list.hold   (application/vnd.snia.xam.boolean) NB RO   "false"
.xsystem.auth.granule.list.retention-event   (application/vnd.snia.xam.boolean) NB RO   "t
rue"
.xsystem.job.list.xam.job.query   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.expiration   (application/vnd.snia.xam.int) NB RO   "-1"
.xsystem.access   (application/vnd.snia.xam.boolean) NB RO   "false"
.xsystem.job.commit.supported   (application/vnd.snia.xam.boolean) NB RO   "false"
.xsystem.job.xam.job.query.continuance.supported   (application/vnd.snia.xam.boolean) NB R
O   "false"
.xsystem.job.xam.job.query.level2.supported   (application/vnd.snia.xam.boolean) NB RO   "
false"
.xsystem.job.xam.job.query.level1.supported   (application/vnd.snia.xam.boolean) NB RO   "
true"
.xsystem.deletion.autodelete   (application/vnd.snia.xam.boolean) NB RO   "false"
.xsystem.deletion.shred   (application/vnd.snia.xam.boolean) NB RO   "false"
.xsystem.management.policy.default   (application/vnd.snia.xam.string) NB RO   "default"
.xsystem.management.policy.list.default   (application/vnd.snia.xam.string) NB RO   "defau
lt"
.xsystem.retention.duration.policy.list.0seconds   (application/vnd.snia.xam.string) NB RO
   "0seconds"
.xsystem.retention.duration.policy.list.10min   (application/vnd.snia.xam.string) NB RO   "
10min"
.xsystem.retention.duration.policy.list.1day   (application/vnd.snia.xam.string) NB RO   "
1day"
.xsystem.retention.duration.policy.list.1hr   (application/vnd.snia.xam.string) NB RO   "1
hr"
.xsystem.retention.duration.policy.list.30days   (application/vnd.snia.xam.string) NB RO
 "30days"
.xsystem.retention.duration.policy.list.60days   (application/vnd.snia.xam.string) NB RO
 "60days"
.xsystem.retention.duration.policy.list.CBR_EMail   (application/vnd.snia.xam.string) NB R
```



```
O   "CBR_EMail"
.xsystem.retention.duration.policy.list.CBR_Email   (application/vnd.snia.xam.string) NB R
O   "CBR_Email"
.xsystem.retention.duration.policy.list.CBR_PACS   (application/vnd.snia.xam.string) NB RO
   "CBR_PACS"
.xsystem.log.level   (application/vnd.snia.xam.int) NB RW   "0"
.xsystem.log.verbosity   (application/vnd.snia.xam.int) NB RW   "0"
.xsystem.log.path   (application/vnd.snia.xam.string) NB RW   "centera_vim.log"

Closed connection to XSystem
$
```

Here are the corresponding outputs when the application instead connects to the SNIA XAM-SDK reference VIM which incidently is written completely in Java; hence the need for two different factory methods in our application.

```
$ java XamFieldIterator -r
Initializing VIM
Loading test properties from file: xam.test.props
Loading the VIM using the Java XAM Library.
VIM Configuration contained in file: ../config/ReferenceVIM.config
Connecting to XSystem snia-xam://SNIA_Reference_VIM!localhost?dir=/home/fpm/xam/xam_storag
e

Enter prefix to filter results (blank for all): .xam
.xam.apiLevel   (application/vnd.snia.xam.string) NB RW   "01.00.00"
.xam.identity   (application/vnd.snia.xam.string) NB RW   "SNIA XAM Java Library v1.0"
.xam.log.append   (application/vnd.snia.xam.boolean) NB RW   "false"
.xam.log.level   (application/vnd.snia.xam.int) NB RW   "1"
.xam.log.max.rollovers   (application/vnd.snia.xam.int) NB RW   "1"
.xam.log.max.size   (application/vnd.snia.xam.int) NB RW   "10"
.xam.log.path   (application/vnd.snia.xam.string) NB RW   "SNIA-XAM.log"
.xam.log.verbosity   (application/vnd.snia.xam.int) NB RW   "1"
.xam.vim.list.SNIA_Reference_VIM   (application/vnd.snia.xam.string) NB RW   "SNIA_Referen
ce_VIM"

Closed connection to XSystem

$ java XamFieldIterator -r

Initializing VIM
Loading test properties from file: xam.test.props
Loading the VIM using the Java XAM Library.
VIM Configuration contained in file: ../config/ReferenceVIM.config
Connecting to XSystem snia-xam://SNIA_Reference_VIM!localhost?dir=/home/fpm/xam/xam_storag
e

Enter prefix to filter results (blank for all): .xsystem
.xsystem.access   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.access.policy.list.read   (application/vnd.snia.xam.string) NB RO   "Read"
.xsystem.access.policy.list.write-application   (application/vnd.snia.xam.string) NB RO   "
Write-application"
.xsystem.access.policy.list.write-system   (application/vnd.snia.xam.string) NB RO   "Writ
e-system"
.xsystem.auth.SASLmechanism.default   (application/vnd.snia.xam.string) NB RO   "PLAIN"
.xsystem.auth.SASLmechanism.list.ANONYMOUS   (application/vnd.snia.xam.boolean) NB RO   "t
rue"
.xsystem.auth.SASLmechanism.list.PLAIN   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.expiration   (application/vnd.snia.xam.int) NB RO   "-1"
.xsystem.auth.granule.list.create   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.granule.list.delete   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.granule.list.hold   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.granule.list.job   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.granule.list.job-commit   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.granule.list.read   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.granule.list.retention-event   (application/vnd.snia.xam.boolean) NB RO   "t
```



```
rue"
.xsystem.auth.granule.list.write-application   (application/vnd.snia.xam.boolean) NB RO   "
true"
.xsystem.auth.granule.list.write-system   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.auth.identity.authentication   (application/vnd.snia.xam.string) NB RO   ""
.xsystem.auth.identity.authorization   (application/vnd.snia.xam.string) NB RO   ""
.xsystem.deletion.autodelete   (application/vnd.snia.xam.boolean) NB RO   "false"
.xsystem.deletion.shred   (application/vnd.snia.xam.boolean) NB RO   "false"
.xsystem.identity   (application/vnd.snia.xam.string) NB RO   "ultra.localdomain File Syst
em"
.xsystem.job.commit.supported   (application/vnd.snia.xam.boolean) NB RO   "false"
.xsystem.job.list.xam.job.query   (application/vnd.snia.xam.boolean) NB RO   "true"
.xsystem.job.xam.job.query.continuance.supported   (application/vnd.snia.xam.boolean) NB R
O   "false"
.xsystem.job.xam.job.query.level1.supported   (application/vnd.snia.xam.boolean) NB RO   "
true"
.xsystem.job.xam.job.query.level2.supported   (application/vnd.snia.xam.boolean) NB RO   "
false"
.xsystem.limits.maxFieldsPerXSet   (application/vnd.snia.xam.int) NB RO   "10000"
.xsystem.limits.maxSizeOfXStream   (application/vnd.snia.xam.int) NB RO   "922337203685477
5807"
.xsystem.management.policy.default   (application/vnd.snia.xam.string) NB RO   ".org.snia.
refvim.default.mgmt.policy"
.xsystem.management.policy.list..org.snia.refvim.default.mgmt.policy   (application/vnd.sn
ia.xam.string) NB RO   ".org.snia.refvim.default.mgmt.policy"
.xsystem.time   (application/vnd.snia.xam.datetime) NB RO   "2009-5-29 10:24:0"

Closed connection to XSystem
$
```

As you can see there are lots of interesting and potentially useful information available in the fields associated with a XAMLIbrary or XSystem object. Hopefully this post has improved your knowledge of the mandated fields and the field namespace.

Happy Independence Day!

