HTTP-based SNMP and CMIP Network Management

Luca Deri
IBM Zurich Research Laboratory
lde@zurich.ibm.com

November 19, 1996

Status of this Memo

1. Abstract

This document describes the application of the HyperText Transfer Protocol (HTTP) [HTTP] for the purpose of SNMP [SNMP] and CMIP [CMIP] management. It shows how SNMP and CMIP resources can be managed by using the standard HTTP protocol by defining a mapping between SNMP/CMIP protocols and HTTP. The mapping is very simple and based on strings which can easily be handled by any programming and scripting language. This will allow light and simple HTTP-based applications to be created, since they have not to include any management service like encoding/decoding nor to handle complex data types.

This document does not cover management of HTTP [Hazewinkel].

2. World Wide Web and Network Management

The World Wide Webarchitecture is depicted below:

------>

```
                   HTTP client    HTTP      HTTP Server
                                 <------
                     Figure 1: World Wide Web architecture
```

HTTP servers provide information which can be retrieved by WWW browsers using the HTTP protocol. The HTTP protocol is a simple, stateless information retrieval protocol based on TCP/IP. The retrieved information can be specified in several formats including graphics, text, binary and HTML (Hyper Text Markup Language) [HTML].

In order to manage network resources using HTTP it is necessary to have an application which speaks both HTTP and SNMP/CMIP. This can be achieved in two ways by:

1. extending standard HTTP servers
2. creating a proxy application which allows to issue SNMP/CMIP protocol requests using HTTP.

The first solution has the following advantages:

1. it is relatively easy to implement since the HTTP server transparently handles the HTTP protocol
2. existing character-based network management applications can easily become Web-aware since it is straightforward to enrich the textual output with HTML tags.

The second solution requires the implementation of the HTTP protocol but it offers better performance. In fact, HTTP servers usually offer standard interfaces such as CGI (Common Gateway Interface) which allows external applications to be executed when a certain URL [URL] (Uniform Resource Locator: a physical address of an object which is retrievable using network protocols such as HTTP) is requested. Since launching applications requires some system resources, the performance is degraded proportionally to the complexity of the CGI application that has to be executed. Apart from offering a better performance, this second solution allows network events (SNMP traps and CMIP notifications) to be handled without the need to rely on another application. This is because the proxy can receive network events and store them. In the case of a CGI-based solution, an external application has to receive the network events which can then be retrieved through a CGI application.

In both cases, users communicate with an HTTP application regardless of the nature of this application, either HTTP server extended with CGI applications or proxy.

3. URL Convention for SNMP/CMIP Management

In order to manage network resources using HTTP, a mapping between management operations and URLs has to be defined. The proposed mapping is compliant with the standard URL syntax and therefore it can be handled by conventional HTTP servers and WWW browsers. The URL is composed of 5 elements,
http://<host>/<protocol>/<operation>/<context>?<parameters>, where:

1. <host> identifies the host where the HTTP server runs;
2. <protocol> specifies the protocol used;
3. <operation> specifies the protocol operation;
4. <context> specifies the context to use, if any;
5. <parameters> contain the operation parameters, if any.

Although this mapping is quite general and can be used for instance with protocols like SQL, our interest focuses on SNMP and CMIP:

1. SNMPv1

```
 <protocol>:    SNMP
 <operation>:   GET, GETNEXT, SET, WALK, TRAP(used to retrieve traps)
 <context>:     object identifier which identifies the MIB attribute
 <parameters>:  additional operation-dependent values (for instance the value
                to be used for the SET)
```

2. CMIP

```
 <protocol>:    CMIP
 <operation>:   GET, SET, CREATE, DELETE, ACTION, EVR (used to retrieve
                event reports)
 <context>:     Agent AE-Title/managed object instance
 <parameters>:  additional operation-dependent values (for instance the value
                to be used for the SET)
```

Since this mapping is supposed to be used not only by software applications (for instance CGI scripts) but also by human operators from within their WWW browsers, it is very important to have a clean and simple syntax. All the ASN.1 values have to be expressed in string format and binary values (BER) are not allowed since they cannot be handled by humans. The object identifiers contained inside the URL can be both in numeric and symbolic form.

ASN.1 values for SNMP are fairly simple hence their conversion to string is straightforward:

INTEGER	'1996'
OCTET STRING	'Luca'
OBJECT IDENTIFIER	'1.3.6.1.2' or 'mgmt'
IpAddress	'9.4.33.33' or 'kis.zurich.ibm.com'
Counter32	'4294967294'
Gauge32	'3265532231'
TimeTicks	'2645533545'
Opaque	'<BER-encoded data>'

In the case of CMIP, ASN.1 values can be very complex hence it is necessary to identify a mapping from ASN.1 to strings and vice-versa. For this purpose, a mapping such as the one defined in [CMIPRun] can be used.

The following examples show how to use the mapping between SNMP/CMIP and URL. Supposing we have an HTTP server (or the proxy) running on host kis.zurich.ibm.com:

1. SNMP
   get the value of sysDescr.0 contained in the SNMP agent running on host bal.zurich.ibm.com:

   http://kis.zurich.ibm.com/SNMP/GET/sysDescr.0?Host=bal.zurich.ibm.
   com&Community=public

Example 1: SNMP Request

2. CMIP
   set the administrativeState attribute to 'enabled' of the managed object instance systemId=(name IBM) contained on the agent whose AE-title is abc:

```
http://kis.zurich.ibm.com/CMIP/SET/abc/systemId=(name+IBM)?administrative
State=enabled&timeout=30
```

Example 2: CMIP Request

In the case CGI is used, the HTTP server has to be configured in such a way that it recognises the protocol name and then starts CGI applications whose names correspond to the value of <operation>. For the previous examples, a possible configuration file for an HTTP server is:

```
ScriptAlias /CMIP/ /usr/local/httpd/CMIP-bin/
ScriptAlias /SNMP/ /usr/local/httpd/SNMP-bin/
```

Figure 2: Sample HTTP configuration for CGI applications

where on every -bin directory the CGI applications are contained. For instance, in the Example 1., inside the directory /usr/local/httpd/SNMP-bin/ there is a CGI application called GET.

4. SNMP/CMIP requests over HTTP

Once the mapping between SNMP/CMIP and URLs is defined, it is now necessary to specify what the format of the information returned by the HTTP protocol is. Supposing the URL shown in the Example 1 is selected, an HTTP client will send the following data to the HTTP server running on kis.zurich.ibm.com:

```
GET SNMP/GET/sysDescr.0?Host=bal.zurich.ibm.com&Community=public HTTP/1.0
[empty line]
```

Figure 3: Data sent by an HTTP client in order to issue the request shown in Example 1

The HTTP response, returned by the HTTP server, is always positive (HHTP wise) unless the requested URL cannot be found or if some other problem arose (for instance authentication problems). In case the HTTP response is positive, it will contain the SNMP response which can be either positive or negative. The HTTP response contains a set of pairs (<identifier>, <value>) separated with carriage return. In case the SNMP response is negative, the last pair is (<empty line>, <error code>) where <error code> contains the error code corresponding to the SNMP request in numeric or string format (for instance "noSuchName" or "2" as defined in the SNMP RFC). Identifiers are object identifiers, usually in symbolic form, whereas values are strings encoded using the encode scheme used by the HTTP protocol. Each line part of the HTTP response is encoded using the URL encoding. This is necessary to avoid that characters like '\n' or '\r', which may be part of the response, interfere with the carriage return used to separate the lines. Multipart MIME encoding can be used as well but it is much more complicated than URL encoding.

A positive response for the previous requests is the following:

```
HTTP/1.0 200 OK
Server: IBM ZRL Proxy Server
Date: Fri, 28 Jun 1996 12:30:16 GMT
Content-type: text/x-www-form-urlencoded
Content-length: 35

sysDescr.0
IBM+RISC+System%2F6000
```

The proposed solution allows to return both a single response or to return multiple responses encapsulated in a single HTTP response. In the case of SNMP Walk for instance, the response contains multiple pairs, one for each element of the MIB. HTTP responses can contain additional fields, like the value type (for instance 'TimeTicks'), which can be used by the client application (for instance the WWW browser) to display the returned value properly.

Similar considerations can be done for CMIP. The only difference with respect to SNMP is that CMIP scoped requests can return multiple CMIP responses where each CMIP response contains multiple attributes relative to a specific object instance. In this case CMIP responses are separated with a (<empty line>, <empty line>). Please note that if the first line of the HTTP response is an <empty line> when the response is negative otherwise it is positive. Due to this there is not ambiguity between (<empty line>, <empty line>) and (<empty line>, <error code>) in case <error code> is empty.

5. Conclusion

The network management world can significantly benefit from the use of the WWW. This document proposes a way to issue network management requests using the HTTP protocol. Major characteristics of the proposed solutions are:

* use of the standard HTTP protocol;
* use of symbolic strings which are handled efficiently by common tools such as awk, sed, and perl and (almost) any programming language;
* ability to manage SNMP/CMIP resources located across firewalls using the HTTP protocol.

This work puts the emphasis on the communication aspect rather the programming aspect. In order to create systems manageable using HTTP, it is necessary to specify the format of the request/response leaving freedom to the developers to select the most appropriate way to issue requests and to handle responses.

6. Acknowledgements

The author gratefully acknowledges Robert Akolk, Bela Ban, Benjamin
Reed and Bert Wijnen for all their comments and suggestions.

7. References

[CMIP] ISO/IEC, CCITT, "Information Technology-OSI, Common Management
Information Protocol (CMIP)-Part 1": Specification ISO/IEC 9596-1,
CCITT Recommendation X.711, 1991.

[CMIPRun] Wade Allen, "An Alternate API for Representing ASN.1
Values", CMIP Run! Magazine, Volume 2, Number 4, 4Q93.

[Hazewinkel] H. Hazewinkel, E. van Hengstum and A. Pras, "Definitions
of Managed Objects for HTTP", draft-hazewinkel-httpmib-00.txt,
University of Twente, April 1996.

[HTML] D. Raggett, "HyperText Markup Language Specification Version
3.0 (HTML)", Internet Draft, April 1995.

[HTTP] T. Berners-Lee, R. Fielding and H. Frystyk, "Hypertext Transfer
Protocol HTTP/1.0", RFC 1945, May 1996.

[SNMP] J. Case, M. Fedor, M. Schoffstall and C. Davin, "The Simple
Network Management Protocol (SNMP)", RFC 1157, May 1990.

[URL] T. Berners-Lee, "Uniform Resource Locators (URL)", Internet
Draft, March 1994.