Introduction

Koha versions 3.0 and later include a functional implementation of the SIP2 protocol. It was designed against the 3M SIP2 specifications, document version 2.12, April 11, 2006. It supports ALL of the SIP (version 1) commands and almost all of the SIP2 commands.

Compatible Devices and Software include:

- 3M Self-Check (SC) Terminals
- Envisionware
- CASSIE PC reservation system by Librarica
- Integratedtek

SIP2 messages supported by Koha

If the Koha Version column is blank, the SIP2 message is not yet supported.

#	Message Name	Koha Version	Note
23	Patron Status Request	3.0	
11	Checkout	3.0	
09	Checkin	3.0	
01	Block Patron	3.0	
99	SC Status	3.0	
97	Request ACS Resend	3.0	
93	Login	3.0	
63	Patron Information	3.0	
35	End Patron Session	3.0	
37	Fee Paid		
17	Item Information	3.0	
19	Item Status Update		
25	Patron Enable	3.0	
15	Hold	3.0	item-level holds not supported
29	Renew	3.0	title-level renew not supported, nor is “third party allowed”
65	Renew All	3.0	

Configuration

1. Create SIP user(s) in STAFF interface as normal for staff users. Set userid and password.

Consider setting one of the notes fields to "SYSTEM USER -- DO NOT DELETE" to warn against deletion. Note the expiration date will affect the SIP users' connection so consider setting it far in the future.

   - TIP: Learn more in this manual on the Adding Staff Patrons page.

2. Set Permissions > grant "Circulate" permissions.
   - TIP: Learn more in this manual on the Patron Permissions page.
3. Koha uses syslogd facility "local6" to catch log messages. Edit your /etc/syslog.conf file to handle them. For example, adding a line like the following to write all levels of messages to /var/log/sipserver.log:

```
sudo bash
echo "local6.*   /var/log/sipserver.log" >> /etc/syslog.conf
exit
```

4. Restart syslogd.
5. On command line, export KOHA_CONF and PERL5LIB as normal. Then copy the example SIPconfig.xml file and edit the copy, like:

```
cd $PERL5LIB/C4/SIP
cp ./SIPconfig.xml ./SIPServer.xml
vi ./SIPServer.xml    # or use whatever text editor you like
```

   1. Edit the <accounts> section to contain a <login> element for each of your SIP user accounts, including correct userid, password and institution (library). These are the values you set in the STAFF interface in Configuration Step #1.
   2. Edit the <institutions> section to contain an <institution> element for each of your SIP users' libraries. Note: it is not necessary to include all institutions here, only those to which SIP users belong.
   3. Note the port number for TCP connections is controlled in this file also.

Exapmle SIPServer.xml file with important points highlighted:

```
<acsconfig xmlns="http://openncip.org/acs-config/1.0/">
  <error-detect enabled="true" />

  <listeners>
    <service
      port="8023/tcp"
      transport="telnet"
      protocol="SIP/2.00"
      timeout="60" />

    <service
      port="127.0.0.1:6001/tcp"
      transport="RAW"
      protocol="SIP/2.00"
      timeout="60" />
  </listeners>

  <accounts>
      <login id="sipterm1" password="term1"  delimiter="|" error-detect="enabled"
institution="MAIN" />
      <login id="envision"  password="koha" delimiter="|" error-detect="enabled"
institution="CPL" />
  </accounts>

<institutions>
    <institution id="MAIN" implementation="ILS" parms="">
          <policy checkin="true" checkout="true" renewal="true"
                  status_update="false" offline="false"
                  timeout="100"
                  retries="5" />
    </institution>
    <institution id="CPL" implementation="ILS" parms="">
          <policy checkin="true" checkout="true" renewal="true"
                  status_update="false" offline="false"
                  timeout="25"
                  retries="5" />
    </institution>
</institutions>
</acsconfig>
```

Run command

From the SIP directory, start the SIP2 server as a background process like:

```
perl -I./ ./SIPServer.pm ./SIPServer.xml >~/sip_log.out 2>~/sip_log.err &
```

Note the same KOHA_CONF and PERL5LIB variables must be exported as always.
Note the redirected STDOUT and STDERR (logs) can be to arbitrary locations.

Stopping

Find the head SIP process ID that is running with "ps". Example:

```
$ ps -ef | grep SIP
atz4sip   5869     1  0 Oct28 ?        00:00:05 perl -I./ /home/atz4sip/koha/C4
/SIP/SIPServer.pm ./SIPconfig.xml
atz4sip   5870  5869  0 Oct28 ?        00:00:01 perl -I./ /home/atz4sip/koha/C4
/SIP/SIPServer.pm ./SIPconfig.xml
atz4sip   5872  5869  0 Oct28 ?        00:00:00 perl -I./ /home/atz4sip/koha/C4
/SIP/SIPServer.pm ./SIPconfig.xml
```

```
atz4sip   5873  5869  0 Oct28 ?        00:00:00 perl -I./ /home/atz4sip/koha/C4
/SIP/SIPServer.pm ./SIPconfig.xml
atz4sip   6382  5869  0 Oct28 ?        00:00:00 perl -I./ /home/atz4sip/koha/C4
/SIP/SIPServer.pm ./SIPconfig.xml
atz4sip  10045  5869  0 Nov11 ?        00:00:01 perl -I./ /home/atz4sip/koha/C4
/SIP/SIPServer.pm ./SIPconfig.xml
atz4sip  28105 32533  0 15:20 pts/23   00:00:00 grep SIP
```

In this case, the head process is 5869, serving as the parent process for all the other "worker" processes. To stop SIP server, kill the head process:

```
kill 5869
```

You can verify that it succeeded by repeating the "ps" command above. Take care not to target another user's SIP processes by mistake if you share the same system.

Testing

Manual testing

Telnet to the TCP port and login as a SIP user. Example session:

```
$ telnet localhost 8023
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
login: sipterm1
password: term1
Login OK.  Initiating SIP
9300CNsipterm1|COterm1|CPMAIN
941
2300120080623    172148AOMAIN|AA23820002060444|ACsipterm1|ADbadpassword
24              00120081216    111849AEWava, W.
Jamal|AA23820002060444|BLY|CQN|AFGreetings from Koha. |AOMAIN|
^]
telnet> quit
```

Here we check the info for a patron with barcode `23820002060444` and a "badpassword". The response declares him a valid patron (BLY) with an invalid password (CQN).

So OK, it's up, but what if you don't speak SIP... how do you test the beast then?

Automated testing

Automated testing of the SIP interface can be tedious because it is necessary to map out all the data to be used by the tests. However, the structure allows C4/SIP/t/SIPtest.pm to consolidate the data used by all the various tests. Edit the section at the top labeled "#Configuration parameters to run the test suite" to match data in your database and your other SIP configuration info. Then run "make test" from the SIP directory.

Advanced Variations

- Export environmental variable KOHA_SIP_LOG_IDENT before starting SIP server to change the name sent to syslogd (default is "acs-server"). This is useful if running multiple SIPs or multiple Koha's on the same interface. Note that the port numbers would have to be different or the 2nd and subsequent SIPs will fail to run.
- Like zebra and mysql and the apache instances, SIP2 can be run on a totally different server than your other Koha interfaces, for security or load balancing. It still needs a fully functioning Koha with DB access, of course.
- Consider using iptables or network level security to prevent unsolicited connections to your SIP2 server. In general you should know the IPs of the systems connecting your your SIP2 server.