Fred King, MSLS, AHIP

Medical Librarian
MedStar Washington Hospital Center
Washington, DC
and
Member-at-Large (Unsupported) 2016-2019
koha-US.org
E-mail: koha@avengingchicken.online

This presentation was originally given at the koha-US conference in Pueblo, Colorado, September, 2019. Released under Creative Commons: you are welcome to use and modify this presentation as you wish. Please give credit to Fred King if possible. All non-credited photos are by Fred King, also released under Creative Commons.
Got a hundred dollars?
Get an ILS!
How to use Koha, MarcEdit, a Raspberry Pi, and a Chicken (optional) to create an ILS for under $100. Koha, MarcEdit, and Raspberry Pi photos from their respective web sites. If you’re wondering about the Avenging Chicken, go to http://www.avengingchicken.online.
Every presentation by a librarian needs to have a picture of a cat.
Thank you to the people who helped me get past an especially troublesome problem. My wife declined to be mentioned by name here, but I’d like to thank her for listening to me talk about Koha day after day. When I thanked her at the 2013 conference in Reno, pretty much every developer in the room nodded understandingly.
The most recent Raspberry Pi (as of September, 2019) can run up to 4GB RAM. Add a 250GB SD card and it rivals some desktop systems. One of the technical manuals is shown in the top right.

**So why a Raspberry Pi?**

- Incredibly complex computing platform, needs extensive training to utilize.
- Latest version is approaching a desktop in computing power.
- Readily available.
- Basic model costs $35.00.
- There are lots of Raspberry Pi projects around. Why not an ILS?
- I wanted to see if it was possible.
The Pi I’m running has 2GB RAM. You can run it without a case, but I feel more secure with one. You can actually use a 4GB card, but I couldn’t find a 4GB Class 10 card. That’s a US quarter in the picture.
I’m assuming that if you’re going to take this on, you have a Drawer of Things with a keyboard, and you have an extra monitor hanging around. If you’re going to use Z39.50 you’ll need a network connection. The only thing you absolutely need another computer for is to flash the SD card, though I used it for other tasks as well.
If you drop the card on the floor, it can be kind of hard to find. If you have a dog that thinks everything that falls on the floor is hers... Well, SD cards are inexpensive enough that you won’t have to try to find it. A replacement card is not included in the $100 budget.
Now that the Pi is assembled and the card is installed, we’re ready to go. Well, almost. It’s pretty easy to miss the SD slot, and the card gets stuck between the Pi and the case. Shake it until it comes out again. Yes, that’s a 64GB card whereas I’m using a 32GB card for the project. There’s a lot of bad continuity in this presentation. Spot them all and win a prize! (Not really.)
Steps

- Install Operating System
- Install and Configure Koha
- Obtain list of ISBNs
- Use MARCEdit and Z39.50 to harvest MARC records
- Edit the records, upload to Koha
- Add items (not shown)

Not much to add in the notes here.
You can find a lot of information about the Raspberry Pi, including this image, at, not surprisingly, www.raspberrypi.org.
There’s a version of Ubuntu available for the Raspberry Pi 2 and 3, but not yet for the 4. There will be one available eventually. For now, I’m using Raspbian, yet another version of Debian.
For this experiment, I’m using the bare-bones, command-line-only version. You can actually install a desktop, and it won’t have a huge impact on performance.
I used BalenaEtcher, but you can use other software to flash a card. BalenaEtcher can read a number of archive and image formats, which makes it especially convenient.
Select your file
It flashes and then verifies the image. This can take a long time.
Connect the cables, attach a monitor and keyboard, and turn it on. It works! The default username is pi, the default password is raspberry. I’m connecting via SSH for this screen shot, but you can log in from a keyboard attached to the Raspberry Pi.
You need to run raspi-config to tell it which keyboard layout to use, etc. The default is Great Britain, which is slightly different from the US layout. In the top right I’m enabling SSH so I can connect from my other computer. This isn’t absolutely necessary, but it does make screenshots a lot easier. These were taken of the actual monitor with a camera.
I used a combination of pages from the Koha Community Wiki to help me install Koha.

There are two important things to remember: when you’re adding the Koha repository, include [arch=i386] in the command line. If you don’t, you’ll get an error message and nothing will work. Second, and this is true for all installations, the letter after wget is a capital letter O. Everybody else probably figured that out the first time.
Prepare to install the software

- Update the software packages
  - `sudo apt-get update`
- Then upgrade the software
  - `sudo apt-get upgrade`

Then update the software packages and upgrade the software. This is very important; if you don’t, it won’t work. Don’t ask me why I know this.
This actually happens a little further down the road, but finding a solution (with the help of the people I thanked at the beginning), took a couple of days. I hit a brick wall and almost gave up.

If you’re worried about the chicken sitting on dirty concrete, please note that it is sitting on a doily.
I ran all of these, and eventually worked. Someone with more knowledge can probably figure out which one. I’m more interested that it worked.
Next, install the main Koha software, koha-common. Then install mariadb (or mysql).

And next, create the library! I’m calling this one “libcat”.
Make a few changes to the web configuration files

You have to tell Apache where the staff client page is (I’m using port 8080), and make sure that it’s listening at that port. I also added port 81, which isn’t necessary but I often use for other purposes.

The file on the left is /etc/apache2/sites-enabled/libcat.conf; the one on the right is /etc/apache2/ports.conf. You’ll need to disable or rename /etc/apache2/sites-enabled/000-default.conf.
Now you can start to set it up

- You’ll need to know your Pi’s IP address
  - `ifconfig`
  - This one is 192.168.5.137
    - (though it changes later)

- You’ll need a password to get in
  - `sudo koha-passwd libcat`
    - gJeqtYxEqO8BIq0@

You’ll need to run `ifconfig` to find out the Pi’s IP address is before you can log into the web-based staff client. (There are many other ways to find the IP as well.) The command “`sudo koha-password [library name]`” will give you the password to log in. The IP changes on the next page. Bad continuity again…
Point your web browser to the Pi’s IP address, then log in with “koha_[library’s name]” and that string of gibberish that’s the password.

Now you can finally use your web browser!
Now go through the Web installer. This will take several minutes and several screens. Don’t worry, you’ll get to the finish eventually.
This is the screen that stopped me before. I was very happy when I no longer got the error message.
Koha will configure the database setting for you...
...then set up the database...
...and when it says Success, you’ve succeeded!
Now install the basic configuration settings.
I’m using MARC 21 here.
You’ll have to make some choices here. I generally install only the mandatory settings.
Linux has a command called “top” that shows what is going on, similar to the Windows Task Manager. The third line down shows the percentage of the CPU’s capacity being used. For most of the installation, it was well below 50%. Right now, 69.5% of the CPU is idle. The fourth line shows how much RAM is being used, in this case, about one quarter. This was a lot less than I expected.
## Web Installer - Default data loaded

### Mandatory data added
- `koha_web_report.pl`
- `webset.sql`
- `message_transport_types.sql`
- `sample_inhouse_message_transport.sql`
- `sample_inhouse_message_transport_report.sql`
- `sample_bibliotec.sql`
- `db_migration.sql`
- `add_migration_data_set.sql`
- `index.blank">
- `marc21_table.marcxml`
- `sample_bibliotec.sql`
- `sample_inhouse.sql`
- `sample_oa_compliance.sql`

### MySQL data added
- `core.sql`
- `coreflags.sql`
- `corepm.sql`
- `corepm_emails.sql`
- `corepm_oa.sql`
- `corepm_oa_compliance.sql`
You’re ready to start adding information!
Create a library code and name. This library is called Raspberry Pi and the code is pi.
Create a Patron Category

You need to add a patron category.
And an administrator so you can log in and run the system. For this library, the Avenging Chicken is the administrator, the username is ac, and the password is Koha1.
And an item type. You can delete this one and add others later.
You can add more circulation rules, too.
Arewethereyet? Arewethereyet? Arewethereyet?

Finished and ready to log in.
In this instance, log in with username ac and password Koha1.
Finally!

And this screen should look familiar.
Yes, but now what?
You have the ILS, but what about the bib records?
I’m cheating again

- But I’ll get to that momentarily
- It’s not just because I’m using bits of a presentation I gave at MLA in 2019
Meet MARCEdit!

- Who uses it?
  - Anybody who does anything with MARC records
Available for Windows, MacOS, and Linux. Yes, you can run it on a Raspberry Pi, though I didn’t for this presentation. See https://marcedit.reeset.net.
What does it do?
It’s kind of like a Swiss Army Knife. When I first used it, I only used the knife and corkscrew. (Yes, there’s no corkscrew here. Deal with it.)
Then I found out that it did a few more things.
Make that a whole lot of other things. I still don’t know how to use all its functions.
This is the main screen. Even from here you can see that it does interesting things. MarcEdit will work directly with Koha, but I haven’t figured that bit out yet.
This is where you need the network connection.
I was given my first book during the Eisenhower administration, when I was a few weeks old. My collection grew from there. Then when I was in my 50s I got married to another librarian. She, too, has a large book collection. We have more books than a lot of small library branches.
And fortunately, a lot of books have their ISBNs in barcodes on their covers. Scan the codes and you have a list of ISBNs. I didn’t include a barcode scanner in my budget because you can also type them into a text file, but if you have an extra $35 for a low-end scanner, it’s well worth it. I think this wireless scanner was around $50.
Ran out of time, and LC Z39.50 server wasn’t working very well

- So this is where I’m cheating

So instead of using my own collection…
I used the library where I worked. I exported the five different itypes shown above. Popular Health Reading is our “what did you read as a child that made you want to become a nurse/doctor?” collection.
I probably could have written a report to do this, but I just dumped the text into a word processor, sorted it, and deleted everything but the 020 fields. Not fancy, but it worked.
Now we’re ready to use the Z39.50 client.
I'm using our Popular Health Reading collection for this example.
Wait a minute—now I’m using the reference collection. I told you the continuity wasn’t great.
And you have MARC records!

- Circulating books
  - 3721 books
    - Found: 2037
    - Not Found: 516

- Reference
  - 840 Books
    - Found: 458
    - Not found: 175

- Reserve
  - 446 Books
    - Found: 424
    - Not found: 356

- Lynch
  - 165 Books
    - Found: 165
    - Not found: 92

- Popular Health Reading
  - 165 Books
    - Found: 99
    - Not found: 194

This was just from the ISBNs. MarcEdit gives you a list of numbers not found, so you can run those in another Z39.50 server, use their LCCNs, etc.
These numbers don’t add up. Partly it’s because some books have ISBN-10s and ISBN-13s, and partly for reasons I don’t know.
MARC Breaker

- Changes machine-readable .mrc file to text .mrk file

Another great feature of MarcEdit is Marc Breaker.
Yep, that’s a MARC record. Since this is a medical library, we’re using 060 call numbers from the NLM system.
Koha really wants a 942$c field, the default item type. If you go to Add/Delete Field, you can add it. I could also have added the 942$2, default classification system, at the same time. Oh well.
Once you have the field added, recompile to MRC format

There it is!
Now I’m uploading the Lynch collection.
Records Staged

Stage MARC records for import
- Processing bibliographic records
- 165 records in file
- 0 records not staged because of MARC error
- 165 records staged
- Did not check for matches with existing records in catalog
- 0 item records found and staged

Everything uploaded correctly.
And they’re imported!

### Manage Staged MARC Records > Batch 1

<table>
<thead>
<tr>
<th>File name:</th>
<th>local_data_records.xlsx</th>
</tr>
</thead>
<tbody>
<tr>
<td>Records</td>
<td>100</td>
</tr>
<tr>
<td>Type:</td>
<td>Biographic records</td>
</tr>
<tr>
<td>Staged:</td>
<td>80</td>
</tr>
<tr>
<td>Status:</td>
<td>Imported</td>
</tr>
<tr>
<td>Missing:</td>
<td>No missing</td>
</tr>
<tr>
<td>Number:</td>
<td>0</td>
</tr>
<tr>
<td>Added:</td>
<td>0</td>
</tr>
<tr>
<td>Searched:</td>
<td>0</td>
</tr>
<tr>
<td>Done:</td>
<td>0</td>
</tr>
<tr>
<td>Complete:</td>
<td>Yes</td>
</tr>
<tr>
<td>Error:</td>
<td>No</td>
</tr>
<tr>
<td>Error msg:</td>
<td>Null</td>
</tr>
<tr>
<td>Error line:</td>
<td>Null</td>
</tr>
<tr>
<td>Error details:</td>
<td>Null</td>
</tr>
</tbody>
</table>

#### Completed report of results

- Number of records added: 100
  - Number of records created: 0
  - Number of records imported: 100
  - Number of records checked: 0
  - Number of records ignored: 0
  - Number of records updated: 0

#### Records imported into MARC

<table>
<thead>
<tr>
<th>#</th>
<th>Citation</th>
<th>Status</th>
<th>Match type</th>
<th>Match details</th>
<th>Field</th>
<th>Record</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Clinical trials that provide a detailed overview of critical outcomes...</td>
<td>Imported</td>
<td>1 record</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Clinical Trials: a detailed overview of critical outcomes...</td>
<td>Imported</td>
<td>1 record</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Clinical Trials: a detailed overview of critical outcomes...</td>
<td>Imported</td>
<td>1 record</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Clinical Trials: a detailed overview of critical outcomes...</td>
<td>Imported</td>
<td>1 record</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Clinical Trials: a detailed overview of critical outcomes...</td>
<td>Imported</td>
<td>1 record</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Clinical Trials: a detailed overview of critical outcomes...</td>
<td>Imported</td>
<td>1 record</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Clinical Trials: a detailed overview of critical outcomes...</td>
<td>Imported</td>
<td>1 record</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Clinical Trials: a detailed overview of critical outcomes...</td>
<td>Imported</td>
<td>1 record</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Clinical Trials: a detailed overview of critical outcomes...</td>
<td>Imported</td>
<td>1 record</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
That’s a fairly bare home OPAC screen, but you can change it. At this point, I had loaded all the MARC records into the system.
Not surprisingly for a medical library, the keyword “surgery” brings up a lot of titles.
That’s all for now...