Ludovic Rousseau's blog

My activities related to smart card and Free Software (as in free speech).

Wednesday, September 24, 2014

PCSC sample in JavaScript (Node.js)

To continue the list of PC/SC wrappers initiated more than four years ago with "PC/SC sample in different languages" I now present a PC/SC sample written in JavaScript using Node.js.

node-pcsclite project

The node-pcsclite project is hosted at github and is quiet active. Support of Windows is not yet available. Support of Mac OS X is now correct after I proposed some patches.

The installation on a Debian unstable or testing (Jessie) system is easy. Just follow the project documentation. Debian stable (Whezzy) do not have the nodejs packages but these packages are available in wheezy-backports.

One potential problem is that the Node.js binary is called `nodejs` on Debian to avoid a conflict with another `node` binary. To have a `node` binary corresponding to Node.js you need to install the Debian package `nodejs-legacy`. It is not difficult but may be the source of some difficulties at the beginning.

PC/SC accesses from node-pcsclite

The wrapper provides access the following PC/SC functions:

- connect
- disconnect
- transmit
- control

A reader event (reader removed) is reported as an event.

The card status change is reported as an event.

The reconnect function is missing. A bug #10 is open requesting its addition.

Sample source code

```
#!/usr/bin/env node

var pcsc = require('./lib/pcsclite');

var pcsc = pcsc();

pcsc.on('reader', function(reader) {

    function exit() {
        reader.close();
        pcsc.close();
    }

    cmd_select = new Buffer([0x00, 0xA4, 0x04, 0x00, 0x0A, 0xA0, 0x00, 0x00, 0x00, 0x6
2, 0x03, 0x01, 0x0C, 0x06, 0x01]);
    cmd_command = new Buffer([0x00, 0x00, 0x00, 0x00]);

    console.log('Using:', reader.name);

    reader.connect(function(err, protocol) {
        if (err) {
            console.log(err);
            return exit();
        }
        reader.transmit(cmd_select, 255, protocol, function(err, data) {
```

Google+ Badge

Ludovic Rousseau blog

G+ Follow

Blog Archive

Search This Blog

Search

Subscribe To

Posts

Comments

Google+ Followers

```
            if (err) {
                console.log(err);
                return exit();
            }
            console.log('Data received', data);
            reader.transmit(cmd_command, 255, protocol, function(err, data) {
                if (err) {
                    console.log(err);
                } else {
                    console.log('Data received', data);
                    console.log('Data received', data.toString());
                }
                return exit();
            });
        });
    });
});

pcsc.on('error', function(err) {
    console.log('PCSC error', err.message);
});
```

Remarks

Node.js is an asynchronous framework. So a typical Node.js design pattern is to use a call-back instead of blocking the execution of a function.

The code can be complex to follow since you have a cascade of call-backs if you need to send a sequence of APDU. In the sample we only need to send 2 consecutive APDU.

The program is not sequential but event based. So without the explicit exit after 1 second the program never terminates and you need to stop it using Control-C. It is strange for me.

Output

```
Using: Gemalto PC Twin Reader 00 00
Data received <SlowBuffer 90 00>
Data received <SlowBuffer 48 65 6c 6c 6f 20 77 6f 72 6c 64 21 90 00>
Data received Hello world!�
```

Similar projects

Two other similar projects are also found at github. They have both the same name node-pcsc but are not the same project:

- node-pcsc from jokesterfr
- node-pcsc from coolbong

coolbong node-pcsc

The node-pcsc interface from coolbong uses a synchronous API so no call-back are involved for PC/SC calls. You can send a sequence of APDU as you would do in C.

This wrapper is for Windows only and need some work to port it to Unix. I opened a bug #1 requesting Unix support.

jokesterfr node-pcsc

This wrapper is not yet able to send arbitrary APDU to a card. It looks like a work in progress that stopped in November 2013.

Conclusion

If you want to use a smart card from a JavaScript program using Node.js the best choice may be the node-pcsclite project. The project maintainer is nice and reactive.

If you know a PC/SC wrapper that is not yet in my list then please contact me.

Edit, October 3rd 2014

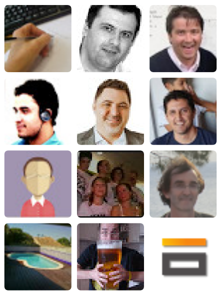
After discussing with Santiago Gimeno (node-pcsclite author) and fixing Mac OS X bugs in node-pcsclite I modified the sample source code to add the clean up `exit()` function and exit properly from the program when no more callbacks are waiting.

G+

Labels: code

Ludovic Rousseau b...

Follow



336 have us in circles

View all

Newer Post Home Older Post

Bitcoin

License: by-nc-sa