Ludovic Rousseau's blog

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

Tuesday, June 8, 2010

PCSC sample in Ruby

Here is the PCSC sample in Ruby language I promised in PC/SC sample in different languages.

A PC/SC Wrapper for Ruby also exists at Smart-card communication, straight from ruby.. The latest version is 0.4.11 from 2009. I used an older version 0.3.1 when I worked with this wrapper.

Installation

You first need to install ruby and the other components needed for the compilation of smartcard for Ruby.

```
apt-get install ruby rake rubygems libopenssl-ruby ruby1.8-dev
```

Then fetch the archive smartcard-0.4.11.gem

```
$ sudo gem install echoe
[...]
$ unzip smartcard.zip
$ cd smartcard
$ rake manifest
[...]
$ rake package
(in /home/rousseau/HSLM/smartcard)
  Successfully built RubyGem
  Name: smartcard
  Version: 0.3.1
  File: smartcard-0.3.1.gem
Private key not found; gem will not be signed.
Targeting "ruby" platform.
$ rake test
(in /home/rousseau/HSLM/smartcard)
/usr/bin/ruby1.8 extconf.rb
checking for main() in -lpcsclite... yes
checking for wintypes.h... yes
checking for reader.h... yes
checking for winscard.h... yes
checking for pcsclite.h... yes
creating Makefile
[...]
/usr/bin/ruby1.8 -Ilib:ext:bin:test "/var/lib/gems/1.8/gems/rake-0.8.3/lib/rake/rake_test_loa
der.rb" "test/test_containers.rb" "test/test_smoke.rb"
Loaded suite /var/lib/gems/1.8/gems/rake-0.8.3/lib/rake/rake_test_loader
Started
...
Finished in 0.046223 seconds.

3 tests, 14 assertions, 0 failures, 0 errors
$ rake docs
[...]
```

Source code

Google+ Badge

Ludovic Rousseau blog

[G+] Follow

Blog Archive

► 2017 (33)
► 2016 (49)
► 2015 (51)
► 2014 (61)
► 2013 (38)
► 2012 (27)
► 2011 (46)
▼ 2010 (55)
  ► December (5)
  ► November (5)
  ► October (9)
  ► September (1)
  ► August (8)
  ► July (1)
  ▼ June (10)
    New version of pcsc-perl: 1.4.9
    PyKCS11 history
    PyKCS11 introduction
    Update on pcsc-lite security advisory CVE-2010-040...
    PCSC sample in Java
    pcsc-lite security advisory CVE-2010-0407
    PCSC sample in Ruby
    PCSC sample in Prolog
    Comments on my blog
    PCSC sample in OCaml
  ► May (6)
  ► April (10)

Search This Blog

[Search]

Subscribe To

🔲 Posts   ⌄
🔲 Comments   ⌄

Google+ Followers

```ruby
require 'smartcard'

context = Smartcard::PCSC::Context.new(Smartcard::PCSC::SCOPE_SYSTEM)
readers = context.list_readers nil
context.cancel

# Use the first reader
reader = readers.first

# Connect to the card
card = Smartcard::PCSC::Card.new(context, reader, Smartcard::PCSC::SHARE_SHARED, Smart
card::PCSC::PROTOCOL_ANY)

# Get the protocol to use
card_status = card.status

# Select applet
aid = [0xA0, 0x00, 0x00, 0x00, 0x62, 0x03, 0x01, 0x0C, 0x06, 0x01]
select_apdu = [0x00, 0xA4, 0x04, 0x00, aid.length, aid].flatten
send_ioreq = {Smartcard::PCSC::PROTOCOL_T0 => Smartcard::PCSC::IOREQUEST_T0,
              Smartcard::PCSC::PROTOCOL_T1 => Smartcard::PCSC::IOREQUEST_T1}[card_stat
us[:protocol]]
recv_ioreq = Smartcard::PCSC::IoRequest.new
response = card.transmit(select_apdu.map {|byte| byte.chr}.join(''), send_ioreq, recv_
ioreq)
response_str = (0...response.length).map { |i| ' %02x' % response[i].to_i }.join('')
puts "Answer: #{response_str}\n"

# test APDU
test_apdu = [0, 0, 0, 0]
response = card.transmit(test_apdu.map {|byte| byte.chr}.join(''), send_ioreq, recv_io
req)
response_str = (0...response.length).map { |i| ' %02x' % response[i].to_i }.join('')
puts "Answer: #{response_str}\n"
response_str = (0...response.length-2).map { |i| '%c' % response[i].to_i }.join('')
puts "Answer: #{response_str}\n"

# Deconnect
card.disconnect Smartcard::PCSC::DISPOSITION_LEAVE
context.release
```

Output

```
Answer: 90 00
Answer: 48 65 6c 6c 6f 20 77 6f 72 6c 64 21 90 00
Answer: Hello world!
```

Conclusion

Nothing more to add. If you are a Ruby user you may be interested by this wrapper.
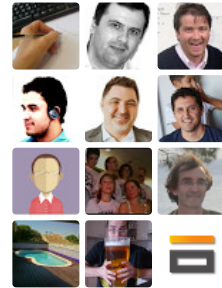
Flattr

G+

Labels: code

Bitcoin

License: by-nc-sa

This blog by Ludovic Rousseau is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License.

Simple theme. Powered by Blogger.