

Working with SAS on a Remote UNIX Server from a Local Windows Client.

By Alex Zolot (Zolotovitski), San Diego, CA

ABSTRACT

This paper summarizes experience of work with SAS located on a remote UNIX server from a local Win client in situations

- when we have SAS or SAS data provider installed on local machine and when we do not have it;
- when we have good connection to UNIX server that allows transfer of full graphic display and
- when the connection through VPN – SSH is too slow for transferring graphic display;
- with and without utilizing SAS/IntrNet server.

The presentation contains some useful SAS, UNIX shell, python, WinSCP scripts.

INTRODUCTION.

In this paper we will consider different ways of working with SAS on a Remote UNIX Server from a Local Windows Client:

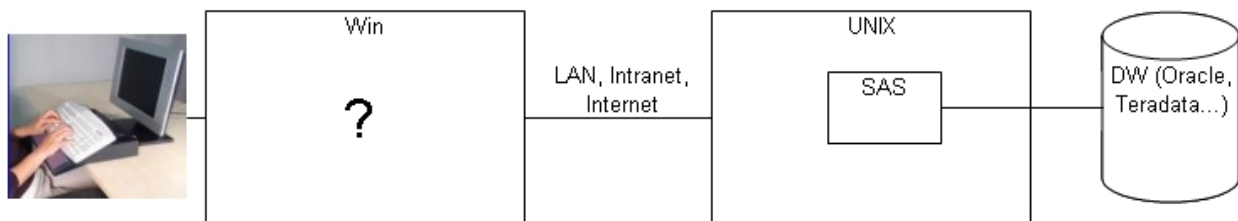
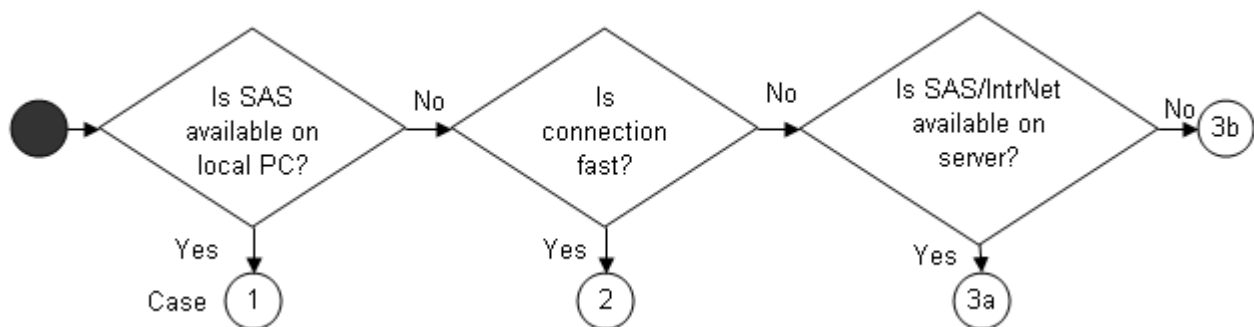
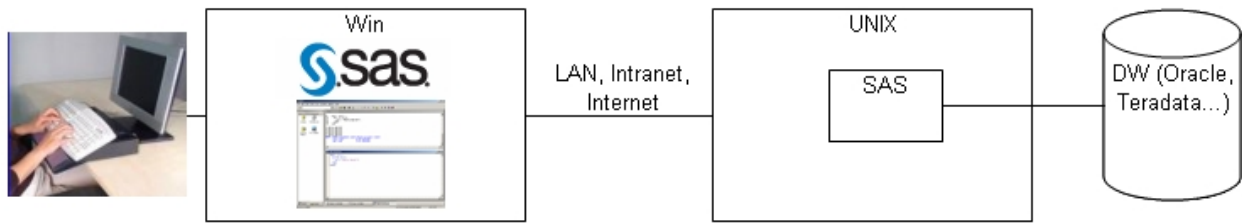


Figure 1. General Configuration.

Situation when we have to work with SAS located on remote UNIX server from local Win client happens very often. This combination has many advantages. Even if we have SAS installed on local PC, remote UNIX server often has higher performance and better connection with Data Warehouse. We will discuss what s/w and what code should use user - instead of the question mark on the Figure 1. We will consider three different cases depending on answers for three following questions:



CASE 1. SAS IS AVAILABLE ON LOCAL PC.



SAS – win is definitely the best possible client for SAS-UNIX that has only one disadvantage – it's expensive. In this case we can set connection between SAS-win and SAS-UNIX, edit code in SAS-win program editor, execute it on SAS-UNIX, view SAS-UNIX libraries and datasets in usual SAS-win way.

SAS code could look as the following one:

```
libname w "/Windir"; *Win;

data pgm=sasuser.uidpass; run;
options remote=sashost1.5019;
signon '\\abc.com\...\Server_script\sasstartbatch.scr' Username=az
Password='pw';
rsubmit;
libname r "/UNIXdir"; * Executed in UNIX;

proc sql;
  connect to oracle(user=az orapw=pw path="@DW");
  create table r.d as
  select * from connection to oracle
  (SELECT * FROM ... ); * Executed in Oracle DW;
  disconnect from oracle;
quit;

proc factor data= r.d nfact=6 out=r.factorOut; * Executed in UNIX;
var x1-x100;
run;
...
proc download data= r.factorOut out= w.factorOut; run;
proc download infile="/UNIXdir/.../&fName" outfile="C:\...\&fName";run;

endrsubmit;

proc export data= w.factorOut outfile='c:\...\factorOut.xls'; * Executed in Win;
run;

proc upload infile= 'c:\...\factorOut.csv'
  outfile="/UNIXdir/.../&fName" ";
run;

rsubmit;
...* Executed in UNIX;
%SYSRPUT l_x=&r_x;
endrsubmit;

%SYSLPUT r_x=&l_x;
```

```

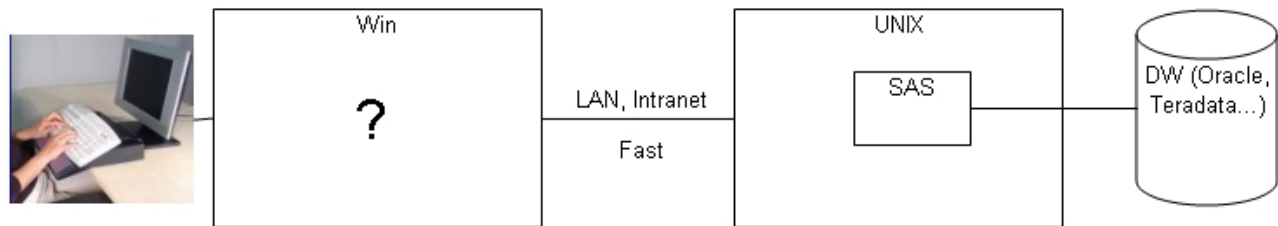
proc gplot data= w.factorOut; * Executed in Win;
plot ...;
run; quit;

signoff;

```

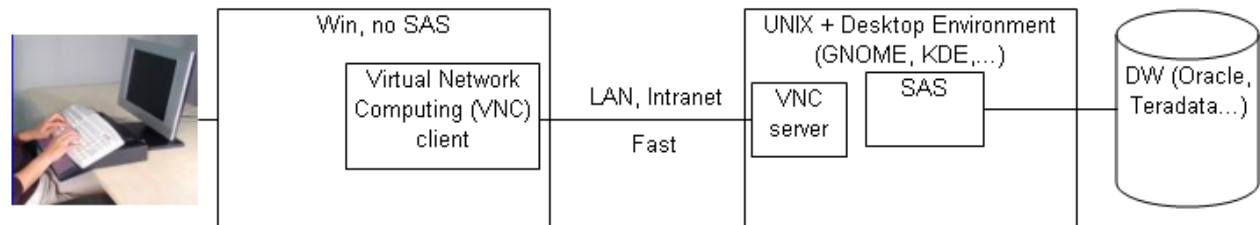
Details are in SAS/CONNECT(R) User's Guide By Sas Institute [10].

CASE 2. SAS IS NOT AVAILABLE ON LOCAL PC, CONNECTION IS FAST.



If we do not have SAS in win client, we can use s/w to get remote access to UNIX display. There are a lot of s/w products to do it [1].

E.g. Hummingbird's Exceed® is one of the most popular proprietary s/w and RealVNC [2] has one of the most popular free version. In the last case Fig.1 transforms to



Detail instruction is available at [3]

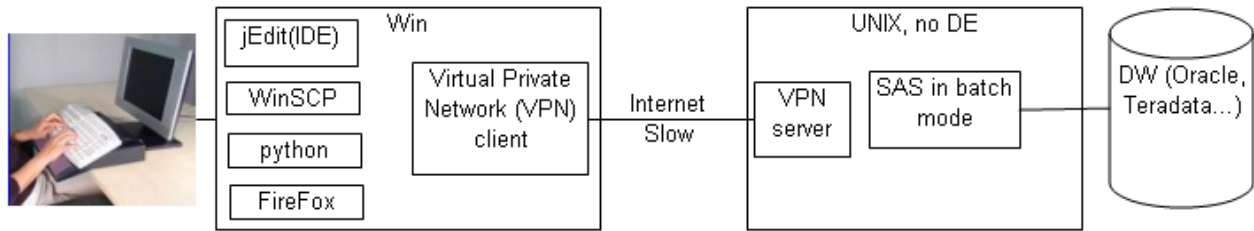
Another free option is to use Xming [4] - an [implementation](#) of the [X Window System](#) for [MS Windows](#) combined with SSH client, e.g. PuTTY [5]. A good demo how to do it [6] is on youtube.com.

All these options work good if connection between server and client is fast enough so transfer of graphic image of UNIX display does not result inconvenience in slowing down work with mouse and keyboard. This situation is typical if client and server are in the same office. In opposite situation we have Case 3.

CASE 3. SAS IS NOT AVAILABLE ON LOCAL PC, CONNECTION IS SLOW.

This situation is typical if client and server are in different locations – maybe in different cities and connected by Internet. By security reasons connection to server uses Virtual Private Network (VPN) [7]. As a result, if we use any of methods, described in case 2, work becomes inconvenient because of slowing down reaction of

display on mouse and keyboard input.



We included in this combination free s/w only. Using proprietary s/w, we can simplify some tasks.

We will use jEdit [8] as SAS editor on local win PC. The editor is free, very flexible, supports SAS syntax highlighting and has a lot of other features. To execute SAS program, user press key F5, then Run_Script.bsh java script is executed. We include in the script the following code:

```

jEdit - ( Session: default ) W:\Run_Script.bsh, W:\SSH_SAS.py
File Edit Search Markers Folding View Utilities Macros Plugins Help
Search for:
Run_Script.bsh (W:\)
1 /* Run_Script.bsh - a BeanShell macro script for the jEdit text editor
2 */
3 buffer.save(view, buffer.getPath());
4 path = buffer.getPath() + " ";
5 mode = buffer.getMode().getName();
6
7 if(mode.equals("python"))
8     runInSystemShell(view, "C:/Python25/python.exe " + path);
9 else if(mode.equals("sas"))
10    runInSystemShell(view, "C:/Python25/python.exe W:/SSH_SAS.py " + path);
  
```

This script call the following python program SSH_SAS.py with one parameter *path* that is full name of SAS file. This program prepares script for WinSCP

```

# Program SSH_SAS.py uploads .sas file to server,
# executes it and downloads results.
# Created April 26, 2008 by Alex Zolot
# Usage: SSH_SAS.py myL_Path/mySAS_prog.sas
# Prefix L means "Local", R means "Remote"

import os,sys,time

os.chdir(os.path.dirname(sys.argv[0]))# necessary in jEdit

if not sys.argv[1:] :
    print 'Error: need an argument - SAS prog name. Got arguments:',sys.argv
    sys.exit()
  
```

```

fSAS=sys.argv[1]                                # second command line parameter

(LPath, filename) = os.path.split(fSAS.strip())
filename_core=filename.replace('.sas', '')
full_log=fSAS.replace('.sas', '.log')
try: os.remove(full_log)
except : pass

SASIntrNet= ( 'myR_Filepath'.find('dispatcher') > 0 )      #boolean

script= """  option batch on
  option confirm off
  open myR_UserID:myR_Password@sasserver_host.com      # hard coded
  cd myR_Filepath                                     # hard coded
  lcd myL_Path
  put mySAS_prog.sas
  call /R_SAS_Path/SAS9 -noterminal myR_Filepath/mySAS_prog.sas > /dev/null 2>&1 &
  close
  exit
  """.replace('mySAS_prog', filename_core) \
  .replace('myL_Path', LPath)

if SASIntrNet:  script=script.replace('call /R_SAS', '#call /R_SAS')

```

Then the program calls WinSCP to execute the script, then downloads .log and .lst files and opens it in jEdit. In Case 3a (SAS/IntrNet) the sas code execution called not by WinSCP script, but through a web browser (in the following code it is Mozilla FireFox) and results of execution also appears in the web browser:

```

print >>open('script.txt','w'), script
os.system('C:\\Progra~1\\WinSCP\\winscp.com /console /script=script.txt')

if SASIntrNet:
url='http://sasserver_host.com:8080/cgi-bin/broker?_service=default' \
  + '&_DEBUG=131&_program=filedisp.'+filename
os.system('C:\\Progra~1\\Mozill~1\\firefox.exe -new-tab '"+url)
sys.exit()

t0 = time.time()
while not os.path.exists(full_log):
  if time.time() > t0+300: break                                # hard-coded timeout
  time.sleep(5)
  print 'Passed %d' % ( time.time()-t0) + ' sec'
  print >>open('script.txt','w'),
    """  option batch on
    option confirm off
    open myR_UserID@sasserver.central.sun.com
    cd myR_Filepath
    lcd myL_Path
    get mySAS_prog.log
    get mySAS_prog.lst
    close
    exit
    """.replace('mySAS_prog', filename_core) \
    .replace('myL_Path', LPath)
  os.system('C:\\Progra~1\\WinSCP\\winscp.com /console /script=script.txt')

os.system('jedit '+full_log)

```

With small efforts the UNIX user password could be hidden [9] and hard coded host name, ID and directory replaced to parameters. Instead of WinSCP in this script we could use simpler program plink (a command-line interface to the PuTTY back ends) that is included in PuTTY [5].

CONCLUSION

We see that usage of remote SAS UNIX from win PC is possible even if we use only free s/w, but it takes some effort, especially in case when connection is poor.

REFERENCES

1. http://en.wikipedia.org/wiki/Comparison_of_remote_desktop_software Remote Desktop s/w
2. <http://www.realvnc.com/products/download.html> RealVNC
3. <http://www.realvnc.com/support/getting-started.html> RealVNC – details
4. <http://sourceforge.net/projects/xming> Xming
5. <http://www.chiark.greenend.org.uk/~sgtatham/putty/> PuTTY
6. <http://www.youtube.com/watch?v=EsHuZJ5gORE> PuTTY and Xming Demo
7. <http://en.wikipedia.org/wiki/Vpn> VPN
8. <http://www.jedit.org/> jEdit
9. <http://winscp.net/eng/docs/scripting> WinSCP Scripting

SAS

10. <http://support.sas.com/documentation/cdl/en/connref/61908/HTML/default/titlepage.htm> SAS/CONNECT
11. <http://support.sas.com/documentation/onlinedoc/intrnet> SAS IntrNet
12. http://support.sas.com/documentation/cdl/en/dispatch/59547/HTML/default/main_contents.htm SAS IntrNet Dispatch

Trademark Citation

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SASInstitute Inc. in the USA and other countries. ® indicates USA registration.

Contact Information

The author welcomes feedback via email at alex.zolot@statvis.com or alex1@zolot.us . Web: www.statvis.com, www.zolot.us .