Imsd windows Test  
ljj版

一、背景:

1. 本篇文档主要是介绍透过windows pc 环境如何测试imsd，包括环境搭建，server 运行，imsd 测试等等内容；
2. 其中server 端主要模拟服务： P-CSCF、AT CMD Server 、Xcap、GBA等等，并base 具体测试的需求回应对应的消息给imsd 这边，以达到模拟测试各test case的目的；
3. 测试server 及imsd 会run在同一台pc中；
4. 使用到的工具环境信息如下：
5. For Imsd：
6. Cygwin
7. Mingw
8. Make工具
9. windows git（git bash）
10. syslog-ng
11. For server：
12. Eclipse
13. Jdk
14. windows git（git bash）

二、环境搭建：

【For imsd】

1. Cygwin 安装:
2. 对Cygwin 获取可以从网上下载，也可以使用如下链接中的：

[\\10.26.128.102\imsshare\ims\_windows\cygwin\_download](file:///\\10.26.128.102\imsshare\ims_windows\cygwin_download)

1. 安装各工具选项时为了避免工具的遗漏，勾选所有的安装即可(默认不是全装， 要改一下)
2. Mingw 安装：

Mingw 可以从如下路径获取：

\\10.26.128.102\imsshare\ims\_windows\ MinGW.7z

1. Make工具构建：

由于ismd这边在cygwin 环境下面build， Cygwin 在安装的时候并没有make 工具，需要自己构建出make 工具来用于imsd 后续的build;

【make generate 方法】

1. download make source code 包

进入如下网站download make source code 包（如：make-3.81.tar.gz）：

<http://ftp.gnu.org/gnu/make/>

如下路径是我这边使的，可以参考使用

[\\10.26.128.102\imsshare\ims\_windows\make-3.81.tar.gz](file:///\\10.26.128.102\imsshare\ims_windows\make-3.81.tar.gz)

1. 透过cygwin的终端环境解压该压缩包（note: Cygwin 同pc 端的映射path 可以在Cygwin的终端透过*mount* cmd看到）：

*tar –zxvf make-3.81.tar.gz*

1. 经过步骤2>就可以看到解压出来的make source code 包了（文件夹：make-3.81），进入该文件夹, 然后follow 如下命令就会生成对应的make.exe 工具了:

*./configure*

1. copy 该make.exe 至 Cygwin安装对应bin 目录中，如我的Cygwin 是安装在D:\sw\_install\cygwin，这样只需要将make.exe copy 到

D:\sw\_install\cygwin\bin下面即可；

1. windows git（git bash）安装及配置:
2. 从网上下载windows git 并安装；

可以从如路径下载windows git 安装软件

<https://gitforwindows.org/>

1. 安装完成后启动git bash（run的是mingw64终端）

开始---》所有程序 ---》git –》git bash

1. 在git bash终端配置git项：
2. Ssh public key(for gerrit), 并将生成的key add 进gerrit的ssh public key setting中:

*ssh-keygen -t rsa*

不用输入密码，一直回车，会在$HOME/.ssh下生成一对密钥

将id\_rsa.pub注册到gerrit

将id\_rsa备份到本地

1. git config:

*git config --global http.sslverify false  
git config --global credential.helper store  
git config --global user.name username  
git config --global user.email*[*username@asrmicro.com*](mailto:username@asrmicro.com) *git config --global color.ui auto   
rm -rf ~/.repoconfig*

1. 问题解决：

“在Windows上更新了git 版本后，clone/pull时出现错误:

unable to negotiate with \*.\*.\*.\*: no matching key exchange methodfound.

Their offer: diffie-hellman-group1-sha1”

解决方法：

在执行git pull/clone之前，输入：

*export GIT\_SSH\_COMMAND='ssh -o KexAlgorithms=+diffie-hellman-group1-sha1'*

这种方法每次打开git窗口，都要重新输入一次

永久生效的方法是在~./ssh/目录下新建一个config文件，将如下文件内容复制进去并保存：

host \*

KexAlgorithms +diffie-hellman-group1-sha1

KexAlgorithms +diffie-hellman-group-exchange-sha256

1. syslog-ng 安装及配置：

参考:

<https://techtorials.me/cygwin/logging/#syslogd>

<https://blog.csdn.net/wentianyao/article/details/50505581>

方法如下（在cygwin 终端中执行）：

1. 执行如下命令并选择yes

*/bin/syslog-ng-config*

*yes*

1. 配置 */etc/sshd\_config*

[老实说我本地没该配置文件，直接从windown git 里面的配置中copy了一份]

主要修改如下两项:

SyslogFacility LOCALS

LogLevel INFO

1. 启动 syslog-ng

*net start syslog-ng*

*cygrunsrv –S syslog-ng*

1. Imsd的运行log 会保存在如下路径：

Cygwin\_install\_path/var/log/messages

Cygwin\_install\_path即Cygwin的安装路径，如我本地cygwin 安装在：

D:\sw\_install\cygwin

那么对应的imsd log 就会在下面文件中：

D:\sw\_install\cygwin\var\log\messages

1. /var/log/messages 重新记录:

从测试看/var/log/messages的内容是累加的，如果要重新记录日志，

请follow如下步骤:

1. 根据需要备份之前的/var/log/messages

*mv /var/log/messages /var/log/messages\_bak*

1. 重新创建/var/log/messages

*touch /var/log/messages*

1. 重启syslog-ng服务(从测试看需要这样)

*net stop syslog-ng*

*net start syslog-ng*

1. 执行程序，就会在/var/log/messages看到新的log内容了
2. 其他相关配置文件及参考:

/etc/syslog-ng/syslog-ng.conf

<https://blog.csdn.net/codingcs/article/details/78001796>

<https://blog.csdn.net/junli_chen/article/details/51566840>

注：我本地有关/etc/syslog-ng/syslog-ng.conf 及 */etc/sshd\_config的配置*

*有放在如下路径中了，供参考：*

[*\\10.26.128.102\imsshare\ims\_windows\config\_file*](file:///\\10.26.128.123\imsshare\ims_windows\config_file)

【For Ims Simulation Server】

1. Eclipse 安装:

可以从网上下载，也可以从如下路径获取Eclipse安装包，安装即可(选择for java)：

[\\10.26.128.102\imsshare\ims\_windows](file:///\\10.26.128.102\imsshare\ims_windows)

1. JDK安装：

可以从网上下载，也可以从如下路径获取Eclipse安装包，安装即可：

[\\10.26.128.102\imsshare\ims\_windows](file:///\\10.26.128.102\imsshare\ims_windows)

1. windows git（git bash）：

在for ismd环境准备中如果已经安装，该步骤可以跳过了，如果没有安装请

参考for imsd 环境准备的 “windows git（git bash）安装及配置:”章节的描述处理

【For All】

1. JDK 环境变量配置：

该部分网上资料比较多，也可以参考下面这个

<https://baijiahao.baidu.com/s?id=1614936534308929731&wfr=spider&for=pc>

1. 各种lib 及bin path 配置：

包括cygwin/mingw等等

三、imsd code 下载及编译：

1. imsd code 下载:  
 a. 登录win git bash 终端；

b．执行如下命令:

*export GIT\_SSH\_COMMAND='ssh -o KexAlgorithms=+diffie-hellman-group1-sha1'*

c.下载code（使用ssh方式）

” <https://source.asrmicro.com/#/admin/projects/platform/vendor/asr/ims/IMS_client>”

*如：git clone ssh://yufengzhu@source.asrmicro.com:29418/platform/vendor/asr/ims/IMS\_client*

d．切换至branch：pxa\_1826

*git checkout pxa\_1826*

1. Code build:

【方法1】

在cygwin终端build：

*cd IMS\_client/winmake*

*make all*

*【方法2】*

*透过Eclipse(要用可以编译cpp的版本) Build（参考补充部分）*

1. Build 成功之后imsd.exe 就产生在如下path:

\IMS\_client\exec

1. Imsd运行：

在cygwin环境下面（如我本地的IMS\_client为：/cygdrive/e/ws\_c\_ims\_ex2/IMS\_client）：

1. 切换至path：/cygdrive/e/ws\_c\_ims\_ex2/IMS\_client
2. 将Settings\_preferences\_win.xml放在

/cygdrive/e/ws\_c\_ims\_ex2/IMS\_client/下面

注：

我本地测试用的Settings\_preferences\_win.xml 有放在如下path，供参考：

[\\10.26.128.102\imsshare\ims\_windows\config\_file](file:///\\10.26.128.123\imsshare\ims_windows\config_file)

1. 透过如下命令运行：

*./exec/imsd.exe -c Settings\_preferences\_win.xml*

四、ImsSimulation Server code下载及运行：

1. code 下载：

a. 登录win git bash 终端；

b．执行如下命令:

*export GIT\_SSH\_COMMAND='ssh -o KexAlgorithms=+diffie-hellman-group1-sha1'*

c.下载code（使用ssh方式）

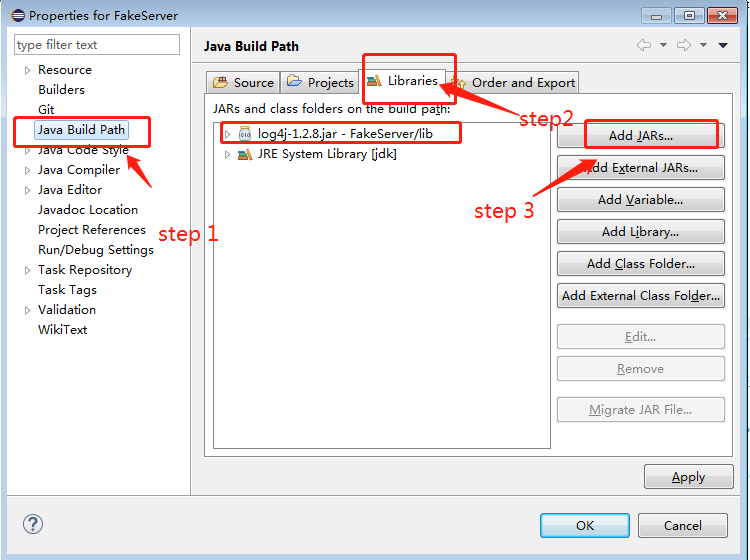
” https://source.asrmicro.com/#/admin/projects/asr/ImsSimulation”

*如： git clone ssh://yufengzhu@source.asrmicro.com:29418/asr/ImsSimulation*

1. code eclipse导入：
2. add 工具lib

右键工程文件夹---》Build Path ---》Config Build Path –》（Properties …）

然后follow如下截图的提示选择add lib 下面的对应的jar



1. 运行：

右键工程文件夹（FackServer）---》Run As ---》java application

五、测试步骤：

Step1：

运行ImsSimulation Server

Step2:

运行 imsd

Step3:

在ImsSimulation Server的终端输入对应的test case 请求

Step4：

ImsSimulation Server跑对应的test case