# All-In-One 打包步骤说明

Version 0.5

Author： 骈海生

## 打包环境

Server: 192.168.8.201 cdapbuilder/Zaq1\_2wsx

Windows Client: 192.168.8.102 虚拟机有问题，编译超慢，建议在自已的工作机上编译

AAaa11/AAaa11(远程桌面方式连接)

Linux Client: docker env

X64: 192.168.8.100 root/ Zaq1\_2wsx

X64: 192.168.8.237 root/ Zaq1\_2wsx

X86: 192.168.8.101 root/ Zaq1\_2wsx

## 打包步骤

**（注意192.168.8.201一定要以cdapbuilder用户名登陆, 并且拷贝文件的时候也要以**cdapbuilder用户，不能用root用户，不然权限不够，没法访问**）**

工作目录： /home/cdapbuilder/

1. 修改配置文件 shell\_config/acesure\_6.0\_package\_shell\_config.ini

*#project\_name : cdap boxsure acesure\_28lab*

*[PROJECT]*

*# rpm name prefix*

*project\_name = cdap*

*[VERSION]*

*# rpm version*

*package\_version = 6.20.0.00 RPM包的版本号*

*# useless*

*web\_version =*

*#branch : release trunk 28lab acesure6.1*

*#web\_branch : libraries6.0 BoxSure6.x 28Cloud*

*[BRANCH]*

*# branch name of svn*

*branch = release 打包的分支，目前都是从release打包，不需要修改*

*# web branch name of svn*

*web\_branch = libraries6.1 web的打包分支，不需修改*

*[TYPE]*

*type = debug*

*[ACTION]*

*#第一次编译的时候应该都设置为1，但是如果有一台客户端不能正常编译的*

*#话, client\_file\_copy 这个属性要设成0.*

*# update svn branch of C++ modules, MessageService, Storage, Compute, ...*

*fun\_svn\_update = 0 如果需要自动更新代码，将其设置为1*

*# update svn brnach of Web*

*web\_svn\_update = 0 如果需要自动更新代码，将其设置为1*

*# compile C++ modules*

*fun\_compile = 0 如果要编译后台的C++模块，将其设置为1，只要有一个模块代码有更新，都要设置为1*

*# compile client packages*

*client\_compile = 0 如果要编译客户端，将其设置为1*

*# copy c++ modules to rpmbuild folder*

*fun\_file\_copy = 0 如果要拷贝后台release分支的编译结果文件，将其设置为1*

*# copy web module to rpmbuild folder*

*web\_file\_copy = 0 如果要拷贝web分支上的文件，则设其为1*

*# copy client packages to rpmbuild folder*

*client\_file\_copy = 0 如果要拷贝客户端安装包，设其为1*

*# rpmbuild server*

*rpm\_package\_server = 1*

*# rpmbuild storage*

*rpm\_package\_storage = 1*

*# rpmbuild compute*

*rpm\_package\_compute = 1*

*[PATH]*

*log\_file\_path = /home/cdapbuilder/package\_log/package\_cdap6.0\_20201230-220643.log*

1. 执行./test\_package\_rpm.sh
2. 编译64位客户端

1) 登陆192.168.8.100

cd /docker/public

./package-release.sh

完成后，/docker/public/Client/（docker和宿主机的共享目录）下会生成以下客户端安装包

drwxr-xr-x. 7 root root 4096 12月 28 19:58 Clone\_Client\_Redhat4x86\_64

drwxr-xr-x. 7 root root 4096 12月 28 20:10 Clone\_Client\_Redhat5x86\_64

drwxr-xr-x. 6 root root 4096 12月 28 20:19 Clone\_Client\_Redhat7x86\_64

drwxr-xr-x. 6 root root 4096 12月 28 20:29 Clone\_Client\_Suse10x86\_64

drwxr-xr-x. 6 root root 4096 12月 28 20:38 Clone\_Client\_Suse12x86\_64

drwxr-xr-x. 6 root root 4096 12月 28 20:48 Clone\_Client\_Ubuntu12x86\_64

2) 登陆 192.168.8.237

cd /home/

    sh packageRelease.sh

等待编译完成  在 /home/Client/ 下

drwxr-xr-x 6 root root 123 Aug 23 22:24 Clone\_Client\_Redhat8x86\_64

drwxr-xr-x 7 root root 134 Aug 23 22:56 Clone\_Client\_Suse15x86\_64

drwxr-xr-x 6 root root 123 Aug 23 22:43 Clone\_Client\_Ubuntu18x86\_64

    成功 生成客户端包 Clone\_Client\_xxx

    失败 生成日志信息 xxx-compile.log

编译完成后自动拷贝至8.201上的：/home/cdapbuilder/Client/stuff\_release下

1. 编译32位客户端

登陆192.168.8.101

cd /docker/hal

./package-release.sh

编译完成后，/docker/hal/Client（docker和宿主机的共享目录）下会生成以下客户端安装包

drwxr-xr-x 7 root root 4096 12月 28 20:45 Clone\_Client\_Redhat4i686

drwxr-xr-x 7 root root 4096 12月 28 21:09 Clone\_Client\_Redhat5i686

drwxr-xr-x 6 root root 4096 12月 28 21:24 Clone\_Client\_Suse10i686

1. 3,4步完成后，会自动将客户端的包拷贝至8.201上的：

/home/cdapbuilder/Client/stuff\_release下

新适配的客户端Clone\_Client\_Suse15x86\_64、Clone\_Client\_Ubuntu18x86\_64、Clone\_Client\_Redhat8x86\_64也要拷贝至上面文件夹下。

待林超的docker环境ready后，可以实现3，4步类似的效果

1. 等待生成三个rpm文件

Windows下打包：

1、192.168.8.102上

执行C: Cdap6.0\_Client\release\Client\_6.0\package\windows\client\_6\_0\_package\_bak.bat。

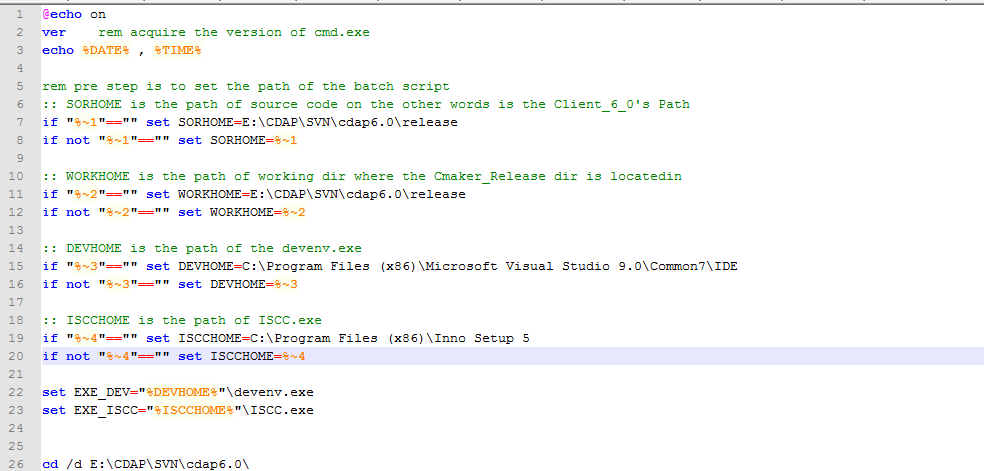
其他需要的什么环境都从这里面拷贝（比如HAL, OpenSrc）

2、自己电脑上（千万不要在物理机上装客户端）

安装vs2008， innosetup-5.5.9 ，安装cmake， 安装wdk（[\\192.168.0.111\share\zxx\安装包\win7ddk.ISO](file:///\\192.168.0.111\share\zxx\安装包\win7ddk.ISO)）虽然wdk是win7版本，但是win10是适配的。

1、修改Client\_6\_0\package\Windows的client\_6\_0\_Package\_release.bat脚本需要更改四个目录。

前两个代表svn代码的路径，第三个是vs2008的路径，第三个是innosetup的路径。



2、把opensrc的winodows开源库拷贝到代码环境中（拷贝到OpenSrc\Windows目录下）

目前是\\192.168.0.111\share\PHS\opensrc

3、Client\_6\_0\diskmanager\ hotplug\windows\MonitorDiskChange这里面的sln生成一下(生成MonitorDiskChange链接库)

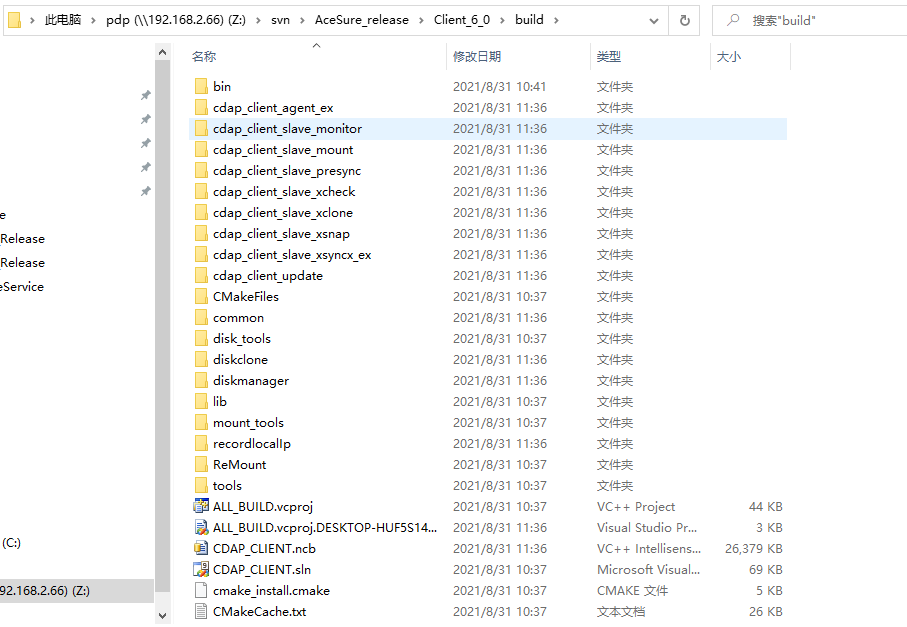
4、运行client\_6\_0\_Package\_release.bat。

5、生成的Cmaker\_Release文件夹中的CDAP\_CLIENT.sln用vs打开后，按F7就可以生成解决方案。（这一步不是必须的，这样做的目的是为了找到编译是哪里出错了，脚本里找不到哪里出错了。）

打包的结果位置是Client\_6\_0\package\Windows\clientPackage\Client\_6\_0\Output\

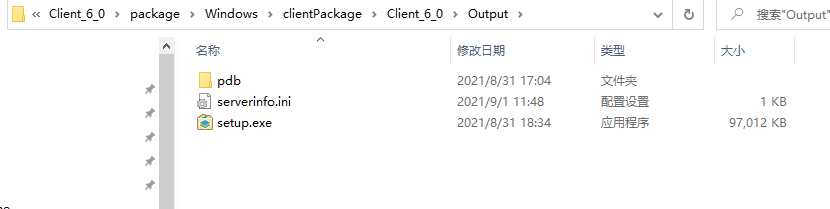
注：

如果想生成sln，可以在Client\_6\_0目录下建一个build目录，然后cmake ..，就会产生sln文件了，就可以进行代码编译了。

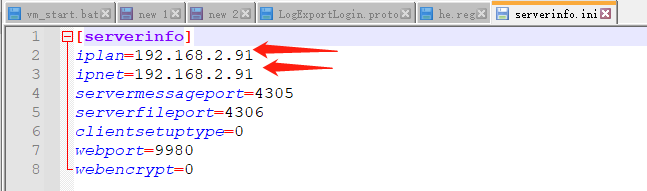


安装客户端程序：

安装windows客户端只需要这两个文件，一个是ini， 一个是setup，需要把



把这两个IP地址改成你想连接的IP，然后执行setup就可以安装好了。



## 相关文件说明

1. 后台服务端脚本：

cd Server/shell/

test\_fun\_compile.sh 编译后台所有c++模块

test\_fun\_file\_copy.sh 拷贝后台所有c++模块的相关文件至rpmbuild的工作文件夹

test\_fun\_svn\_update.sh 更新后台所以C++模块代码

test\_web\_file\_copy.sh 拷贝web模块的相关文件至rpmbuild的工作文件夹

test\_web\_svn\_update.sh 更新web模块代码

1. 客户端相关脚本

cd /home/cdapbuilder/Client/shell

compile\_client.ini 客户端列表

linux\_client\_compile.sh 客户端编译脚本

scp\_call.sh 拷贝文件至8.201

scp.sh 执行文件

windows\_client\_compile.sh windows平台编译脚本

/home/cdapbuilder/Client/test\_client\_compile.sh 编译脚本

/home/cdapbuilder/ Client/test\_client\_file\_copy.sh 拷贝脚本

1. Rpmbuild打包脚本

rpmbuild/SOURCES/test\_rpm\_package\_server.sh

rpmbuild/SOURCES/test\_rpm\_package\_storage.sh

rpmbuild/SOURCES/test\_rpm\_package\_compute.sh

1. Rpmbuild机制

SOURCE源文件位置：

/home/cdapbuilder/rpmbuild/SOURCES/SBF\_release

compute

server

storage

SPEC文件：（就是安装时执行的脚本）

/home/cdapbuilder/rpmbuild/SPECS

CDAPSERVER.spec\_release

CDAPCompute.spec\_release

CDAPStorage.spec\_release

rpmbuild命令会基于spec文件中的脚本内容，将source文件打成rpm文件，并存放在：

/home/cdapbuilder/rpmbuild/RPMS/x86\_64/下：

/home/cdapbuilder/rpmbuild/RPMS/x86\_64/cdap\_compute-6.20.0.00-1.x86\_64.rpm

/home/cdapbuilder/rpmbuild/RPMS/x86\_64/cdap\_server-6.20.0.00-1.x86\_64.rpm

/home/cdapbuilder/rpmbuild/RPMS/x86\_64/cdap\_storage-6.20.0.00-1.x86\_64.rpm