通过Hyperic-hq产品的基础包sigar.jar来实现服务器状态数据的获取 收藏

通过Hyperic-hq产品的基础包sigar.jar来实现服务器状态数据的获取。Sigar.jar包是通过本地方法来调用操作系统API来获取系统相关数据。Windows操作系统下Sigar.jar依赖sigar-amd64-winnt.dll或sigar-x86-winnt.dll，linux 操作系统下则依赖libsigar-amd64-linux.so或libsigar-x86-linux.so……而Sigar.jar还依赖于jug-asl-2.0.0.jar、log4j-1.2.14.jar、Junit.jar，

Hyperic-hq官方网站：http://www.hyperic.com

Sigar.jar下载地址：http://sigar.hyperic.com

Sigar.jar包的使用方法：

1、 CPU资源信息

a) CPU数量（单位：个）

privatestaticint getCpuCount() throws SigarException {

Sigar sigar = new Sigar();

try {

return sigar.getCpuInfoList().length;

} finally {

sigar.close();

}

}

b) CPU的总量（单位：HZ）及CPU的相关信息

Sigar sigar = getSigar();

CpuInfo infos[] = sigar.getCpuInfoList();

for (int i = 0; i < infos.length; i++) {//不管是单块CPU还是多CPU都适用

CpuInfo info = infos[i];

traceln("mhz=" + info.getMhz());//CPU的总量MHz

traceln("vendor=" + info.getVendor());//获得CPU的卖主，如：Intel

traceln("model=" + info.getModel());//获得CPU的类别，如：Celeron

traceln("cache size=" + info.getCacheSize());//缓冲存储器数量

}

c) CPU的用户使用量、系统使用剩余量、总的剩余量、总的使用占用量等（单位：100%）

publicvoid testCpuPerc() {

Sigar sigar = new Sigar();

// 方式一，主要是针对一块CPU的情况

CpuPerc cpu;

try {

cpu = sigar.getCpuPerc();

printCpuPerc(cpu);

} catch (SigarException e) {

e.printStackTrace();

}

// 方式二，不管是单块CPU还是多CPU都适用

CpuPerc cpuList[] = null;

try {

cpuList = sigar.getCpuPercList();

} catch (SigarException e) {

e.printStackTrace();

return;

}

for (int i = 0; i < cpuList.length; i++) {

printCpuPerc(cpuList[i]);

}

}

privatevoid printCpuPerc(CpuPerc cpu) {

println("User :" + CpuPerc.format(cpu.getUser()));// 用户使用率

println("Sys :" + CpuPerc.format(cpu.getSys()));// 系统使用率

println("Wait :" + CpuPerc.format(cpu.getWait()));// 当前等待率

println("Nice :" + CpuPerc.format(cpu.getNice()));//

println("Idle :" + CpuPerc.format(cpu.getIdle()));// 当前空闲率

println("Total :" + CpuPerc.format(cpu.getCombined()));// 总的使用率

}

d) ……

2、 内存资源信息

a) 物理内存信息

Sigar sigar = new Sigar();

Mem mem = sigar.getMem();

// 内存总量

System.out.println("Total = " + mem.getTotal() / 1024L + "K av");

// 当前内存使用量

System.out.println("Used = " + mem.getUsed() / 1024L + "K used");

// 当前内存剩余量

System.out.println("Free = " + mem.getFree() / 1024L + "K free");

b) 系统页面文件交换区信息

Sigar sigar = new Sigar();

Swap swap = sigar.getSwap();

// 交换区总量

System.out.println("Total = " + swap.getTotal() / 1024L + "K av");

// 当前交换区使用量

System.out.println("Used = " + swap.getUsed() / 1024L + "K used");

// 当前交换区剩余量

System.out.println("Free = " + swap.getFree() / 1024L + "K free");

c) ……

3、 操作系统信息

a) 取到当前操作系统的名称：

private String getPlatformName() {

String hostname = "";

try {

hostname = InetAddress.getLocalHost().getHostName();

} catch (Exception exc) {

Sigar sigar = new Sigar();

try {

hostname = sigar.getNetInfo().getHostName();

} catch (SigarException e) {

hostname = "localhost.unknown";

} finally {

sigar.close();

}

}

return hostname;

}

b) 取当前操作系统的信息

publicvoid testGetOSInfo() {

OperatingSystem OS = OperatingSystem.getInstance();

// 操作系统内核类型如： 386、486、586等x86

System.out.println("OS.getArch() = " + OS.getArch());

System.out.println("OS.getCpuEndian() = " + OS.getCpuEndian());//

System.out.println("OS.getDataModel() = " + OS.getDataModel());//

// 系统描述

System.out.println("OS.getDescription() = " + OS.getDescription());

System.out.println("OS.getMachine() = " + OS.getMachine());//

// 操作系统类型

System.out.println("OS.getName() = " + OS.getName());

System.out.println("OS.getPatchLevel() = " + OS.getPatchLevel());//

// 操作系统的卖主

System.out.println("OS.getVendor() = " + OS.getVendor());

// 卖主名称

System.out.println("OS.getVendorCodeName() = " + OS.getVendorCodeName());

// 操作系统名称

System.out.println("OS.getVendorName() = " + OS.getVendorName());

// 操作系统卖主类型

System.out.println("OS.getVendorVersion() = " + OS.getVendorVersion());

// 操作系统的版本号

System.out.println("OS.getVersion() = " + OS.getVersion());

}

c) 取当前系统进程表中的用户信息

publicvoid testWho() {

try {

Sigar sigar = new Sigar();

Who who[] = sigar.getWhoList();

if (who != null && who.length > 0) {

for (int i = 0; i < who.length; i++) {

Sysout.out.println("\n~~~~~~~~~" + String.valueOf(i) + "~~~~~~~~~");

Who \_who = who[i];

Sysout.out.println ("getDevice() = " + \_who.getDevice());

Sysout.out.println ("getHost() = " + \_who.getHost());

Sysout.out.println ("getTime() = " + \_who.getTime());

//当前系统进程表中的用户名

Sysout.out.println ("getUser() = " + \_who.getUser());

}

}

} catch (SigarException e) {

e.printStackTrace();

}

}

d) ……

4、 资源信息（主要是硬盘）

a) 取硬盘已有的分区及其详细信息（通过sigar.getFileSystemList()来获得FileSystem列表对象，然后对其进行编历）：

publicvoid testFileSystemInfo() throws Exception {

Sigar sigar = getSigar();

FileSystem fslist[] = sigar.getFileSystemList();

String dir = System.getProperty("user.home");//当前用户文件夹路径

for (int i = 0; i < fslist.length; i++) {

System.out.println("\n~~~~~~~~~~" + i + "~~~~~~~~~~");

FileSystem fs = fslist[i];

// 分区的盘符名称

System.out.println("fs.getDevName() = " + fs.getDevName());

// 分区的盘符名称

System.out.println("fs.getDirName() = " + fs.getDirName());

System.out.println("fs.getFlags() = " + fs.getFlags());//

// 文件系统类型，比如 FAT32、NTFS

System.out.println("fs.getSysTypeName() = " + fs.getSysTypeName());

// 文件系统类型名，比如本地硬盘、光驱、网络文件系统等

System.out.println("fs.getTypeName() = " + fs.getTypeName());

// 文件系统类型

System.out.println("fs.getType() = " + fs.getType());

FileSystemUsage usage = null;

try {

usage = sigar.getFileSystemUsage(fs.getDirName());

} catch (SigarException e) {

if (fs.getType() == 2)

throw e;

continue;

}

switch (fs.getType()) {

case 0: // TYPE\_UNKNOWN ：未知

break;

case 1: // TYPE\_NONE

break;

case 2: // TYPE\_LOCAL\_DISK : 本地硬盘

// 文件系统总大小

System.out.println(" Total = " + usage.getTotal() + "KB");

// 文件系统剩余大小

System.out.println(" Free = " + usage.getFree() + "KB");

// 文件系统可用大小

System.out.println(" Avail = " + usage.getAvail() + "KB");

// 文件系统已经使用量

System.out.println(" Used = " + usage.getUsed() + "KB");

double usePercent = usage.getUsePercent() \* 100D;

// 文件系统资源的利用率

System.out.println(" Usage = " + usePercent + "%");

break;

case 3:// TYPE\_NETWORK ：网络

break;

case 4:// TYPE\_RAM\_DISK ：闪存

break;

case 5:// TYPE\_CDROM ：光驱

break;

case 6:// TYPE\_SWAP ：页面交换

break;

}

System.out.println(" DiskReads = " + usage.getDiskReads());

System.out.println(" DiskWrites = " + usage.getDiskWrites());

}

return;

}

b) ……

5、 网络信息

a) 当前机器的正式域名

public String getFQDN(){

try {

return InetAddress.getLocalHost().getCanonicalHostName();

} catch (UnknownHostException e) {

try {

Sigar sigar = new Sigar();

return sigar.getFQDN();

} catch (SigarException ex) {

returnnull;

} finally {

sigar.close();

}

}

}

b) 取到当前机器的IP地址

public String getDefaultIpAddress() {

String address = null;

try {

address = InetAddress.getLocalHost().getHostAddress();

// 没有出现异常而正常当取到的IP时，如果取到的不是网卡循回地址时就返回

// 否则再通过Sigar工具包中的方法来获取

if (!NetFlags.LOOPBACK\_ADDRESS.equals(address)) {

return address;

}

} catch (UnknownHostException e) {

//hostname not in DNS or /etc/hosts

}

Sigar sigar = new Sigar();

try {

address = sigar.getNetInterfaceConfig().getAddress();

} catch (SigarException e) {

address = NetFlags.LOOPBACK\_ADDRESS;

} finally {

sigar.close();

}

return address;

}

c) 取到当前机器的MAC地址

public String getMAC() {

Sigar sigar = null;

try {

sigar = new Sigar();

String[] ifaces = sigar.getNetInterfaceList();

String hwaddr = null;

for (int i = 0; i < ifaces.length; i++) {

NetInterfaceConfig cfg = sigar.getNetInterfaceConfig(ifaces[i]);

if (NetFlags.LOOPBACK\_ADDRESS.equals(cfg.getAddress())

|| (cfg.getFlags() & NetFlags.IFF\_LOOPBACK) != 0

|| NetFlags.NULL\_HWADDR.equals(cfg.getHwaddr())) {

continue;

}

/\*如果存在多张网卡包括虚拟机的网卡，默认只取第一张网卡的MAC地址，如果要返回所有的网卡（包括物理的和虚拟的）则可以修改方法的返回类型为数组或Collection，通过在for循环里取到的多个MAC地址。\*/

hwaddr = cfg.getHwaddr();

break;

}

return hwaddr != null ? hwaddr : null;

} catch (Exception e) {

returnnull;

} finally {

if (sigar != null)

sigar.close();

}

}

d) 根据MAC地址来获得一个GUID号

public String getGUID(String mac) {

if (mac == null)

returnnull;

EthernetAddress eAddr = new EthernetAddress(mac);

return UUIDGenerator.getInstance().generateTimeBasedUUID(eAddr)

.toString();

}

e) 获取网络流量等信息

publicvoid testNetIfList() throws Exception {

Sigar sigar = new Sigar();

String ifNames[] = sigar.getNetInterfaceList();

for (int i = 0; i < ifNames.length; i++) {

String name = ifNames[i];

NetInterfaceConfig ifconfig = sigar.getNetInterfaceConfig(name);

print("\nname = " + name);//网络设备名

print("Address = "+ ifconfig.getAddress());//IP地址

print("Netmask = "+ ifconfig.getNetmask());//子网掩码

if ( (ifconfig.getFlags() & 1L) <= 0L) {

print("!IFF\_UP...skipping getNetInterfaceStat");

continue;

}

try {

NetInterfaceStat ifstat = sigar.getNetInterfaceStat(name);

print("RxPackets = " + ifstat.getRxPackets());//接收的总包裹数

print("TxPackets = " + ifstat.getTxPackets());//发送的总包裹数

print("RxBytes = " + ifstat.getRxBytes());//接收到的总字节数

print("TxBytes = " + ifstat.getTxBytes());//发送的总字节数

print("RxErrors = " + ifstat.getRxErrors());//接收到的错误包数

print("TxErrors = " + ifstat.getTxErrors());//发送数据包时的错误数

print("RxDropped = " + ifstat.getRxDropped());//接收时丢弃的包数

print("TxDropped = " + ifstat.getTxDropped());//发送时丢弃的包数

} catch (SigarNotImplementedException e) {

} catch (SigarException e) {

print(e.getMessage());

}

}

}

void print(String msg){

System.out.println(msg);

}

f) 一些其他的信息

privatevoid getEthernetInfo(){

Sigar sigar = null;

try {

sigar = new Sigar();

String[] ifaces = sigar.getNetInterfaceList();

for (int i = 0; i < ifaces.length; i++) {

NetInterfaceConfig cfg = sigar.getNetInterfaceConfig(ifaces[i]);

if (NetFlags.LOOPBACK\_ADDRESS.equals(cfg.getAddress())

|| (cfg.getFlags() & NetFlags.IFF\_LOOPBACK) != 0

|| NetFlags.NULL\_HWADDR.equals(cfg.getHwaddr())) {

continue;

}

System.out.println("cfg.getAddress() = " + cfg.getAddress());//IP地址

System.out.println("cfg.getBroadcast() = " + cfg.getBroadcast());//网关广播地址

System.out.println("cfg.getHwaddr() = " + cfg.getHwaddr());//网卡MAC地址

System.out.println("cfg.getNetmask() = " + cfg.getNetmask());//子网掩码

System.out.println("cfg.getDescription() = " + cfg.getDescription());//网卡描述信息

System.out.println("cfg.getType() = " + cfg.getType());//

System.out.println("cfg.getDestination() = " + cfg.getDestination());

System.out.println("cfg.getFlags() = " + cfg.getFlags());//

System.out.println("cfg.getMetric() = " + cfg.getMetric());

System.out.println("cfg.getMtu() = " + cfg.getMtu());

System.out.println("cfg.getName() = " + cfg.getName());

System.out.println();

}

} catch (Exception e) {

System.out.println("Error while creating GUID" + e);

} finally {

if (sigar != null)

sigar.close();

}

}

本文来自CSDN博客，转载请标明出处：http://blog.csdn.net/zhuzy2004/archive/2009/06/26/4300501.aspx