리눅스 하드웨어관리 솔루션 개발지원

한국소프트웨어진흥원
공개SW기술지원센터

<Revision 정보>

일자	VERSION	변경내역	작성자
2007.11.12	0.1	초기 작성	김상운

목 차

1. 문서 개요

본 문서는 (주)티씨오솔루션 사의 요청으로 인해 자사에서 출시하고자 하는 리눅스 클라이언트용 관리 솔루션의 기능중 리눅스 기반 데스크탑 하드웨어의 시스템정보를 추출하는 기능구현개발을 위한 참고 문서이다.

가. 문서의 목적

다음과 같은 세부적인 목적을 달성하기 위하여 작성되었다.

0 리눅스 상에서 하드웨어 정보 추출 구현 방법 소개

0 관련 개발 환경 정의 및 구현 소스 개발/제공

0 관련 개발자들에게 관련 문서 배포

나. 본 문서의 사용방법

다음과 같은 방법으로 사용할 수 있다.

0 본 문서는 일반 리눅스 클라이언트에 적용될 에이전트(Agent)에 한정하고 있음

0 본 문서에 포함된 소스는 GPL기반으로 공개 예정이므로 향후 자사 제품을 비공개하기 위해서는 별도의 개발 필요

다. 참고사항

0 본 소스로 개발된 프로그램은 perl 기반에서 적용가능하며 본 문서는 한글과컴퓨터 아시아눅스 데스크탑 3.0 기반으로 개발 및 테스트되었음

0 일부 배포판중에서 본 문서에서 설명하는 개발도구 또는 라이브러리가 포함되어 있지 않을 경우 본 문서의 내용이 지원되지 않을 수 있으므로 참고바람

2. 개발지원 요청내역

가. 요청 사항

항목	요청내용	지원 내역	비 고
	리눅스 클라이언트용 하드웨어 시스템 정보 추출 기능 개발 지원	- perl 환경에서 리눅스 클라이언트에 적용될 배포 에이전트 개발 지원 - 관련 예제 소스 지원	

나. 지원인력구성

담당	직급	성명	소속사	인력 구분	기간	지원내용	비고
솔루션 개발팀	과장	김상운	㈜한글과 컴퓨터	상주	2007.10.10~ 2007.11.02	리눅스 클라이언트용 에이전트 개발 지원	

3. 개발내역

가. 개발 소프트웨어 기본정보

-제품 이름 : 미정
-버전 : 1.0
-소스코드 언어 및 환경 : perl

나. 개발 및 적용환경

-소프트웨어 환경 (한글과컴퓨터 리눅스 데스크탑 3.0)

커널	2.6.14 이상
X-Windows	Xorg 7.1 이상
Glibc	2.3 이상
GCC	3.0 이상
waxlib	1.2 이상

다. 개발내역

1) 개요

리눅스 상에서 하드웨어 정보를 보기 위해서는 관련 디렉토리(/var, /proc 등)과 실시간 정보를 가져오기 위한 /dev 디렉토리안의 디바이스들을 모니터링 해야 한다.

또한 수집된 정보에 대해서 서버로 전송하기 위해서는 별도의 포맷과 알고리즘을 이용하여 외부에 유출되지 않도록 암호화 또는 캡슐화를 하여 진행하여야 한다.

본 프로그그램은 다음과 같은 구조로 동작하게 된다.

* /dev, /proc, /var 디렉토리를 참조하여 하드웨어 정보 취득 -> 전체 정보를 데스크탑에 특정 포맷으로 암호화 하여 통합 저장 -> 서버로 TCP를 이용해서 전송 -> 서버에서 정보를 전달받아 서버에 있는 DB에 데스크탑 정보를 업데이트 함.

현재 클라이언트상에서 이들 이벤트에 맞추어 각 배포판별로 커널에 상관없이 동작할수 있는 에이전트 개발이 필요로 하고 있다. 따라서 배포판과 문제없이 적용될수 있는 파이썬과 RPM을 이용하여 구현될 수 있도록 개발 중심을 맞추었으며 다음과 같은 개발 소스가 제공되었다.

참고로 아래의 소스는 sourceforge.net 의 Linux Hardware Inventory 1.1 소스를 참고하여 수정 및 개발되었다.

2) 관련 제공 소스

```
use strict;
use Getopt::Std;
use Sys::Hostname;
use vars qw/$opt_v $opt_c $opt_t $opt_s $opt_e/;

getopts('vsc:t:e:');

my $PROCFS    = '/proc';
my $MAILER    = '/usr/lib/sendmail';
my $hinv_out = '';

if (-e $PROCFS) {
        my $KERNEL  = do { open F, "$PROCFS/sys/kernel/osrelease" and <F> } or die $!;

          if ($KERNEL =~ m/^2.1.*₩|2.0.*₩|1.*/) {
                  print STDERR "Fatal: Your system is running a pre-2.2.x kernel.₩n";
                  exit 1;
        }
```

```
} else {
        print STDERR "Your kernel does not appear to support proc/:₩n";
        print STDERR "To use this tool, first rebuild the kernel with proc/ support.₩n";
      exit 1;
}

my $VERBOSE = 1 if $opt_v;


my %class = (
      processor => [ 'ncpu' ],
      disk       => [ 'scsi', 'ide' ],
      memory     => [ 'mem' ],
      serial     => [ 'ser' ],
      parallel   => [ 'par' ],
      graphics   => [ 'vid' ],
      network    => [ 'net' ],
      scsi       => [ 'scsi' ],
      ide        => [ 'ide' ],
      all        => [ 'ncpu', 'mem', 'scsi', 'ide', 'ser', 'par', 'vid', 'net']
);

$opt_c = 'all' unless defined $opt_c;
usage() unless defined $class{$opt_c}[0];

if (defined $opt_t) {
      my $return = ncpu($opt_t);
      exit $return;
}

my $return = '';

foreach my $function (@{ $class{$opt_c} }) {

      no strict 'refs';
      $return = &$function();
}

if (defined $opt_e && $hinv_out) {

      my $HOST = hostname();
      my $USER = "<$ENV{'USER'}₩@$ENV{'HOSTNAME'}>";
      my $RECV = "<$opt_e>";

      open SENDMAIL, "|$MAILER -oi -t " or die $!;
      print SENDMAIL <<"EOF";
From: $USER
To: $RECV
```

```
Subject: Hardware Inventory for $HOST

$HOST Hardware Inventory:

------------------------------------------------------

$hinv_out
------------------------------------------------------
Generated by: $USER
EOF
        close SENDMAIL;

        exit 0;
}

exit $return;

# Functions. Here there be tygers.

##############################################################################

sub out {
        chomp(my $msg = shift);
        return 0 if defined $opt_s;
        if (defined $opt_e) {
                $hinv_out .= "$msg₩n";
                return 0;
        }
        print STDOUT "$msg₩n";
        return 0;
}

sub usage {
        print STDERR "$0 Usage: $0 [-v] [-c class] [-t type] [-s] [-e email address]₩n";
        exit 1;
}

sub ncpu {
        my $query = shift;
        my %type = ();
        my $suffix = '';
        my $cpu_info = "$PROCFS/cpuinfo";
        my ($nproc, %iproc) = get_listinfo($cpu_info, '^processor₩s+');
        $suffix = 's' if $nproc > 1;
        chomp(my $arch = `/bin/uname -m`);
        my $clock = int($iproc{'cpu MHz'});
        $type{'arch'} = "$nproc $clock MHz $arch Processor$suffix";
```

```
        $type{'cpu'} = "CPU: $iproc{'vendor_id'} $iproc{'model name'}";
        if ( $iproc{'fpu'} eq 'yes') {
                $type{'fpu'} = "FPU: $iproc{'vendor_id'} $iproc{'model name'} Floating Point";
        }
        if (exists($iproc{'cache size'})){
                $type{'scache'} = "  Secondary cache: $iproc{'cache size'}";
        }

        unless (defined $query) {

                map { out($type{$_}) if defined $type{$_} } keys %type;
                return 0;
        }

        if (exists($type{$query})) {

                out("$type{$query}");
                return 0;

        } else {

                $query =~ tr/a-z/A-Z/;
                out("Type $query nonexistent");
                usage();
                return 1;
        }
}

sub mem {
        unless (-e "$PROCFS/kcore") {
                out('Memory information Unavailable');
                return 1;
        }

        my $mem_mb = int(((( stat "$PROCFS/kcore")[7] / 1048576) +.5 ));
        out("Main memory size: $mem_mb Mbytes");
        return 0;
}

sub par {
        my $par_info = "$PROCFS/parport";

        unless (-e $par_info) {
                out('No parallel port configured') if ($VERBOSE);
                return 1;
        }

        foreach my $dev (@{get_dirinfo($par_info)}) {
```

```
                next unless $dev =~ /₩d+/;
                my ($nelems, %parinfo) = get_listinfo("$par_info/$dev/hardware", 'modes');
                out("Configured $parinfo{'modes'} mode parallel port at $parinfo{'base'}");
        }
}

sub ser {
        my $ser_info = "$PROCFS/tty/driver/serial";

        unless (-e $ser_info) {
                out("No serial ports configured") if $VERBOSE;
                return 1;
        }
        my $uart    = '';
        my $nser    = 0;
        my $suffix = '';

        open SERIAL, "$ser_info" or die $!;
        while (<SERIAL>){
                next unless /tx|rx/;
                $nser++;
                ($uart = $_) = m/^₩d+:₩s+uart:(₩d+₩w*)/;
        }
        close SERIAL;

        $suffix = 's' if $nser > 1;
        out("$nser $uart UART serial port$suffix");
        return 0;
}

sub vid {
        my $vid_info = "$PROCFS/ioports";
        my ($vid_n, %ioports) = get_listinfo($vid_info, 'vga');
        my %vid_data = reverse %ioports;

        if (!defined $vid_n){
                out('No Graphics boards found') if ($VERBOSE);
                return 1;
        }

        out("$vid_n Graphics board: VGA detected at $vid_data{'vga+'}") if $vid_n;
        my $vid_pci = get_pciinfo('VGA compatible controller');
        out("   $vid_pci") if $vid_pci;
        return 0;
}

sub scsi {
```

```
        my $scsi_info = "$PROCFS/scsi";

        unless (-e "$scsi_info/scsi") {
                out('No SCSI subsystem configured') if $VERBOSE;
                return 1;
        }
        foreach my $dev (@{get_dirinfo($scsi_info)}) {

                next if $dev =~ /scsi/;

                foreach my $bus (@{get_dirinfo("$scsi_info/$dev")}) {
                        out("SCSI controller $bus: Version $dev");
                        next unless -e "$scsi_info/scsi";
                        open SCSI, "$scsi_info/scsi" or die $!;
                        while (<SCSI>) {
                                if (/scsi$bus/) {
                                        my @target = m/Id:₩s+(₩d+)₩s*Lun:₩s+(₩d+)/g;
                                        out("  Device $target[0] on SCSI controller $bus, Lun $target[1]");
                                }
                        }
                        close SCSI;
                }
        }
        return 0;
}

sub ide {
        my $ide_info = "$PROCFS/ide";

        foreach my $dev (@{get_dirinfo($ide_info)}) {

                next unless ($dev =~ /^ide₩d+/);
                my $bus_id = substr($dev, -1, 1);
                my $model  = do { open F, "$ide_info/$dev/model" and <F> } or die $!;

                $model =~ tr/a-z/A-Z/;
                out("IDE controller $bus_id: $model");

                foreach my $unit (@{get_dirinfo("$ide_info/$dev")}) {

                        next unless $unit =~ /^hd₩w+/;
                        open MEDIA, "$ide_info/$dev/$unit/media" or die $!;
                        chomp (my $media = <MEDIA>);
                        close MEDIA;

                        $media =~ tr/a-z/A-Z/;
                        my $ide_id = substr($unit, -1, 1);
                        out("  $media Device: unit $ide_id on IDE controller $bus_id");
```

```
                }
        }
        return 0;
}

sub net {
          my $if_list = "$PROCFS/net/dev";
          my %if_names = (
                'eth' => 'Ethernet',
                'ppp' => 'Point-to-Point',
                'slp' => 'Serial Line IP',
        );
          my @if_dev = ();
          unless (-e $if_list) {
                out('No network interfaces configured') if $VERBOSE;
                return 1;
        }

        open NET, $if_list or die $!;
        while (<NET>) {
                push @if_dev, m/^₩s*(₩w+₩d+):/;
        }
        close NET;

        foreach my $dev (@if_dev){
                chop(my $key = $dev);
                out("$if_names{$key} interface configured: $dev");
        }
        return 0;
}

sub get_pciinfo {
        my $query = shift;
        my $line  = '';

        if (-e "$PROCFS/pci") {

                open PCI, "$PROCFS/pci" or die $!;
                while (<PCI>){
                        s/^₩s*//;
                        $line = $_ if /$query/;
                }
                close PCI;
        }
        chomp  $line;
        return $line;
}
```

```
sub get_listinfo {
        my ($list, $count) = @_;
        my $nelems = 0;
        my %ldata  = ();

        if (-e $list) {

                open LIST, $list or die $!;
                %ldata = map { $nelems++ if (/$count/); chomp; split /₩s*:₩s*/; } <LIST>;
                close LIST;
        }

        return $nelems, %ldata;
}

sub get_dirinfo {
        my $query = shift;
        my @elems = ();

        if (-e $query) {

                opendir DIR, $query or die $!;
                @elems = grep !/^₩.₩.?$/, readdir DIR;
                closedir DIR;
        }

        return ₩@elems;
}

__END__
```

라. 향후개선(지원요청)내역

- 본 소스는 티씨오솔루션의 개발에 참조하기 위해 만들어진 소스이며 향후 요청에 따라 확대 개발 예정