Software's Little Helpers: Managing Your Lab Areas

Doug Simpson
University of Oregon
1225 Kincaid St.
Eugene, OR 97403-1212
541-346-1736
dsimpson@darkwing.uoregon.edu

ABSTRACT

There are always more labs and other things to attend to than available bodies to watch over said pesky details. How can we keep an eye on the ever-present large and small events in our labs while dealing with Yet More Interruptions elsewhere?

At the University of Oregon, we have found three utilities and/or technologies particularly useful:

- KeyServer, to monitor unauthorized and unusual software events
- Remote web cameras
- Web pages displaying current and archived camera output for visual backup of unadvertised hardware events (PCs as cup-holders, suddenly-mobile monitors, etc.)

Front-line staff and supervisors (whether technical or not) can use these tools to manage the details and monitor for problems.

This paper discusses the tools described above, with an emphasis on their use in a university environment where security and privacy issues are key. Other hardware and software solutions to these problems are also briefly contrasted and compared.

Categories and Subject Descriptors

D.4.6 [**Operating Systems**]: Security and Protection – *Access Controls.* K.6.5 [**Management Of Computing And Information Systems**]: Security and Protection – *Authentication*, *Physical security.*

General Terms: Management, Measurement, Security, Legal Aspects.

Keywords: KeyServer, banned or prohibited applications, report generation, web cameras.

1. INTRODUCTION

Whenever there are more rooms to supervise than available supervisors, we need to come up with strategies to watch over our facilities when we're not there. Padlocks on hardware remove some temptations, but there are other pieces of equipment difficult to secure in this way. We ask our student assistants to keep an eye on things, but they too are limited in their OA (Omnipresence Ability). We have also seen a rise in undesirable patron-installed software in the last year. Browser 'enhancements' like Bargain Buddy, casino gambling plug-ins, and cookie-laden search tools cause aggravation for the next user and eventual slowdowns for the machines. Yet we do wish to allow our lab users some room for legitimate exploration of the computer's capabilities. How can we allow a degree of freedom without sacrificing usability?

We have used KeyServer in our labs for over ten years to control licensed applications, but had never made time to explore all of its options. When we upgraded to the latest version some of the benefits touted (automated software audits, ability to track usage and last use of any program) caught our eye. KeyServer also offers the ability to prevent any program from running, once the program has been identified.

To help watch over our disappearing peripherals we installed web cameras some time ago and saw an immediate reduction in vandalism as a result, but we didn't have a system for storing images from the cameras. We wanted something that would be easy for us to quickly flip through recent saved images.

This paper details our on-going exploration of these technologies to fine-tune the control of our labs' software and hardware. Alternative software and hardware that we've considered for these roles will also be briefly mentioned.

2. INSTITUTION AND LAB BACKGROUND

The University of Oregon has approximately 20,000 students. The Library and Computing Center support the majority of the labs available for general student use. The Computing Center controls four labs: three combine instructional and open labs; one is open lab only. The 300 Wintel and Macintosh computers are roughly 2/3 Windows XP, 1/3 OS X. Total staff for all labs is 30+ student lab assistants, 2 student technical workers, 2 managers, and a lab coordinator.

3. KEYSERVER

KeyServer is a software licensing application sold by Sassafras Software [1] that 'keys' applications so they will only run after a license has been checked out to a workstation; if all available licenses are already in use, the application will not run. It is highly configurable, but it does this basic job so well that we were able to set it up quickly and move on to more pressing tasks. Since it worked quietly in the background we were rarely forced to consult the manual and left its more advanced features to be discovered later.

3.1 Identifying Applications to be Banned

Any version of KeyServer can be used to select applications that we want to ban (that is, to refuse permission to run).

To control a program the only thing we need to know is the program's name. If we know the name of the program's installer we can control that as well and prevent the application from being installed in the first place. KeyServer can discriminate between minor version and name differences.

The latest version (K2) supports remote audits from KeyConfig, the administrative component of KeyServer. It can scan any client computer currently connected to the network and produce a list of every application program on that machine's hard drive. Earlier versions use KeyAudit, which runs on the client computer.

3.2 Controlling Banned Applications

Using KeyConfig, find the program you wish to control within the 'Programs' windows and select the 'action' icon – it will probably currently be a green circle (for 'ignore'). Change it to 'controlled.' A 'Create License' dialog box will appear. Un-check the selection named 'Allow launch when KeyServer not available' so this program cannot run without our permission (which we will not give). When the 'License Details' dialog box appears, select 'Concurrent Use Limit (Floating License)' and set the User Limit to zero. Save the record. The program will now be banned on any client computer connected to this KeyServer.

3.3 Reporting Banned Activities

Using KeyConfig, select 'Denials (PROG x comp)' or 'Denials (PROG x user)' from the 'Reports' menu for a list of all programs that were unable to run. Any date range may be selected.

All applications controlled as described above will appear here if a user tried to run them. The report also shows the number of times the program attempted to run.

3.3.1 Other Reports

KeyServer can report on many other activities, too. Summaries of how often a program is used (or when it was last used) are useful in determining whether to continue supporting that program. Reports of weekly software use can show how busy a lab is if there is no other tracking of users.

KeyServer responds to SQL queries and report data can be exported to an external SQL server or Microsoft Access.

3.4 Alternative Ways to Control Applications

If you don't have KeyServer now or are looking for other ways to control programs, here are some other ways this can be achieved:

- Group Policy settings (Microsoft Active Directory)
- LANDesk, ZENworks (Novell), or SMS (Microsoft) to alert staff to new installs
- For some programs, installation can be defeated by placing hidden, read-only dummy files where that program expects to install itself
- DeepFreeze/DriveShield (and others) – these programs don't prevent installation of new programs or control applications, but do reset all user files back to a known default state between user logins

Note that none of these alternatives works transparently across the PC and Macintosh platforms as KeyServer does.

4. WEB CAMERAS

Before we installed our web cameras in our labs we experienced a steady trickle of small peripherals leaving the lab (mice, mouse pads, an occasional keyboard) and a few more pricey items (a computer and two overhead projectors). In rooms where the cameras were placed there has been almost no theft since that time.

Before we can peruse the archived logs of our cameras' output we need to generate some raw footage (or JPEG-age, in our case). Here are detailed some basics of selecting a camera, setting it up, and producing images for us to view.

4.1 PC-Dependent Cameras vs. Web Cameras

The terminology may be confusing when looking at all the cameras available for use with computers. By 'PC-dependent camera' I mean a digital camera that is connected to a computer by a USB or serial cable. These are the most inexpensive cameras (starting at less than $100), but contain no built-in programming or control capability. They require a directly-connected PC of some kind to produce an image.

A 'web camera' (as I call it here) combines in one case a digital camera and an embedded operating system – often a version of Linux. Acting as its own web server, it has an IP address and attaches directly to the network via an Ethernet port. The camera can also be used with a modem and a phone line, if you need to use it where there is no network connection handy. Their cost begins somewhere around $250 for a basic unit.

Surveillance programs that use PC-dependent cameras exist but the low cost of the camera was offset for us by the extra cost of the dedicated PC that would be required for each camera. We also needed the flexibility the web camera allowed. To complete the installation we needed a visit from our Network Services folks for an Ethernet port high in one strategically-chosen corner, another visit from the Physical Plant to add an electrical outlet, and we were ready to plug in the web camera.

4.2 Axis 2100 Web Camera

We chose the Axis 2100 [2] because it had the desired embedded OS, built-in shell scripting, a serviceable lens, and an affordable price. JPEG images of 320 x 240 or 640 x 480 can be transmitted at up to 10 frames per second. It was helpful to know many other institutions had made the same camera choice.

More full-featured models allowed camera pan and tilt, built-in motion detection, PHP programming, and more varied lenses but the cost was significantly higher. We were told motion detection (a feature we wanted) could be implemented in the Axis 2100 through software.

4.3 Camera Setup and Operation

The serial number for the camera is also the MAC address (the network card identification number) and we used this to register each camera with Network Services so each could be assigned a static IP. This allows us to refer to the camera by name from a network browser or in other scripts.

Once this is done, to see the camera's output it is only necessary to enter a URL like the following to see live output from the camera:

http://klamath-cam3.uoregon.edu

As shipped, the camera will allow anyone to view images without a password.

The default web page presented includes an 'Admin' button which allows you to change the size of the image, color, headings, etc. in addition to other settings.

Other setup tasks:

- Be sure to change the default 'root' password so no one else can alter the camera's settings.
- You may decide to password-protect the camera's output so only your staff or other people you pick can view the images.
- Make sure the latest firmware is installed. Firmware updates and instructions for this are available for download from the Axis website. It's not difficult to do and ensures your camera has the latest operating system patches.
- The cameras have a built-in clock (very useful for time-stamping images) but the time can wander, to the point where it may show a time and date hours or days removed from the actual time. Setting the time source to "Synchronize with NTP server" will ensure the time is accurate.

If the unthinkable happens and you make a choice that leaves the camera in a strange place, there's a small reset button that returns it to the factory defaults. You'll have to re-enter any customizations you made earlier, but the camera is difficult to damage in a permanent way. Your very own server – experiment!

4.4 Viewing Multiple Cameras on One Page

If you have four cameras and wish to view each of them in turn, you'll need to save four bookmarks and switch between them continually with the out-of-the-box setup. To view more than one camera on a single web page you need to resort to some web page scripting or a viewing program that will display the images on one page for you.

Commercial, shareware, and freeware programs at all levels of complexity and cost exist for viewing images from web cameras. The majority of these programs are written for the Wintel platform, but several good ones exist for Macintosh [3] and Linux [4] users as well.

Advantages of pre-written programs include the fact that they are ready for use and may come with support and printed manuals (at least, if you pay for this privilege). They may include handy buttons to capture a particular image you want to save, or include motion-detection software to alert you to changes in a certain part of the screen you wish to focus on. They may incorporate archives of the camera images for later viewing (see next section for more about this).

If all you need is a simple viewer of up to three cameras at one time the free versions are adequate for this. We are currently using one of these while we work on our own customized viewer.

5. WEB PAGES WITH ARCHIVED OUTPUT

After we had our cameras set up and in use in our labs, we left them at that stage for more than a year. We derived many benefits from them as they were (Are the labs open at 8am? How busy is it in a lab right now?) but wanted to be able to look back in time to answer other questions (Are our patrons making adjustments to the printers we'd prefer they did not? Did a lab close early last night?).

There is another advantage to sending a camera's output to a remote storage point – a camera can only support a maximum of 10 direct connections at one time. As more people connect to the camera, the refresh rate (the number of frames per second it can display) will drop. Sending the output to a more robust file server allows the camera to capture images at its highest rate of speed.

5.1 Commercial Programs

As with their simple viewer program brethren, commercial and shareware programs exist that can archive old images, string them into movie files, alert you when motion is detected during certain hours, and more. The cost for these programs ranges from just $30 to over $2000. As we are building our own system from scratch, we have (so far) only viewed product information for these programs for their inspirational value.

5.2 Our Homegrown Solution

5.2.1 Archived Output

We have implemented basic capture of images, via ftp, to a convenient file server. The capability to do this is built into the cameras and requires only these steps (select the Admin button on the camera's web page):

- Network – TCP/IP: enter your network's domain and DNS server information (necessary for the camera to find the ftp server it will send images to)
- Operation – Selection: select Sequential Mode
- Operation – Scheduler: select how often you wish to save a picture and (optional) hours or days to suspend picture taking
- Operation – Upload: enter the remote server, name and password, and directory to save the images in. You also choose the size of the image here and whether or not to save a fixed number of images (one way to limit the total images saved)
- Operation – Enable: click 'Enable' to begin image capture

5.2.2 Storage Requirements

You will have to calculate how much space you need to reserve for your archived images. An image can be as small as 3Kb (320 x 240, high compression) or as large as 250Kb (640 x 480, lowest compression). A camera sending a 320 x 240 image at lowest compression (about 80K) once a minute would consume about 5 megabytes of storage each hour. Using the camera feature that suspends image capture at certain hours will allow more time before recycling the archive becomes necessary.

6. PLANS FOR IMPROVEMENT

6.1.1 Customized Camera Viewer

We are developing a better viewing program for quick and easy monitoring current conditions in our labs. We want to be able to choose which of our cameras to have on-screen and to have the ability to capture an image to a named file. We'd like the viewer located on a central fileserver and controlled by Java (or some other cross-platform technology) so we can use it from any of our workstations.

6.1.2 Better Archived Output

We would like to implement some of these options for archived output:

- Automated archiving and deletion of images (perhaps rotated to CD-RW disks if we think long-term storage would be useful)
- More convenient image viewing on server
- Converting a string of images to an MPEG or AVI movie stream [5]

6.1.3 Intruder Alert!

Not just for intruders, but anytime we're looking for the unusual:

- Triggered e-mail of images when certain events occur (movement in the lab when there should be none, etc.)
- Software motion detection

6.1.4 Exploration of Scripting Commands

The camera's built-in scripting commands make some of the desired actions above easier to achieve. Axis provides a guide on their website that details the built-in scriptable commands and offers some hints to how they can be used [6].

Once the image has been moved to a remote location, further processing of the output would take place on that host. The shell scripting commands on our Unix-based file servers and the Axis cameras are quite similar, which we hope will make joint development easier.

We would be glad to share any of our solutions which might be useful for other institutions. Please feel free to contact us, or refer to a web page [7] we'll keep current with our experiments.

7. SOME LEGAL ISSUES

(Disclaimer: I'm not a lawyer, nor do I play one on campus.) I've been watching the SIGUCCS [8] and LABMGR [9] listservs for the past year or so to see how other campuses deal with the privacy issues web cameras raise. It appears some institutions have a clear, top-down policy that everyone abides by; others (by design or not) have policy made on a departmental or lab-by-lab basis. On our campus we have no explicit policy yet; there are departments that forbid the use of cameras in their labs and others that allow them.

8. CONCLUSIONS

We found that exploring capabilities of products we already owned resulted in new functionality without extra cost to us. Also, technologies sometimes complemented each other in ways we hadn't anticipated. With KeyServer and cameras in operation in the same room it might now be possible to observe repeated attempts to install a notorious application on a particular workstation (recorded in KeyServer's reports) and use the archived camera images to see who was responsible for the attempted installation.

9. ACKNOWLEDGMENTS

I wish to extend my sincere appreciation and thanks to our public labs coordinator, Mary Bradley, who supports explorations of new technology like this even when our daily 'To-Do' card is already full – and we've turned it over and are using the back of it.

10. REFERENCES

[1] Sassafras Software, http://www.sassafras.com

[2] Axis Communications, http://www.axis.com/products/cam_2100/index.htm

[3] SecuritySpy, http://www.bensoftware.com/ss/index.html

[4] Linux Video Surveillance AKA eLViS, http://www.silicontao.com/software/lvs/doc/information.html

[5] "Making MPEG Movies with Axis Network Cameras," http://www.linuxjournal.com/article.php?sid=4535

[6] Axis Scripting Guide, http://www.axis.com/techsup/cam_servers/dev/files/script_guide.pdf

[7] http://darkwing.uoregon.edu/~dsimpson/cameras/

[8] SIGUCCS listserv info, http://www.acm.org/sigs/siguccs/lists.htm

[9] LABMGR listserv info, http://www.lsoft.com/scripts/wl.exe?SL1=LABMGR&H=LISTSERV.UARK.EDU