The following paper was originally published in the
Proceedings of the Large Installation System Administration of Windows NT Conference
Seattle, Washington, August 5–8, 1998

Monitoring Utilization in an NT Workstation Lab

Paul Kranenburg
Erasmus University Rotterdam

For more information about USENIX Association contact:

1. Phone: 510 528-8649
2. FAX: 510 548-5738
3. Email: office@usenix.org
4. WWW URL: http://www.usenix.org/

Monitoring Utilization in an NT Workstation Lab

Paul Kranenburg
Erasmus University Rotterdam
e-mail: kranenburg@few.eur.nl

Abstract

This paper describes a set of tools used to monitor and maintain NT workstations at the faculty of Economics at the Erasmus University Rotterdam. These tools have been employed in different shapes and on a variety of systems since 1992. In particular, the data gathered on usage patterns of workstations in the faculty's public student labs has proved to be valuable for long-term planning of lab capacity and resource allocation. The collected data is also made available for live monitoring of workstation status and utilization. This is helpful both to tutors giving classes and to lab assistants while troubleshooting. Since mid-1997, all lab machines run Windows NT and the monitoring and maintenance system has been updated to take advantage of features that NT offers.

0. Introduction

As heirs from a predominantly do-it-your-self style of operation, contemporary PC desktop systems still present some challenges to any system administration group setting out to efficiently manage them as a uniform collection of computing resources. Only recently, with the arrival of Windows NT as a mainstream desktop operating system, a reasonably balanced platform has emerged that allows a decent form of system administration to take shape in this area. This paper addresses two issues which we consider to be of prime importance to sustaining smooth operation of our installed base of desktop workstations: (1) maintaining information on utilization and operational status of individual systems at a central location, and (2) the ability of making any necessary adjustments to the system configuration swiftly and with minimal disruption.

The tools presented here share a common design strategy, which in general takes the form of a small policy-less stub on each workstation that communicates with one or more counterparts running on a server. Such a scheme facilitates the concentration of all system configuration policies at a central location and also allows for easy processing and consultation of system status data collected on all desktop workstations.

1. The early stages

The control and monitoring features described here have their roots in the early 1990's, at which time we prepared to integrate a lab of DOS-based machines into an existing network environment consisting of Unix servers and workstations. Noteworthy characteristics of the lab environment include: (1) support for a large number (˜10000) of users, demanding a docile authentication system; (2) all machines are directly connected to the Internet, also requiring proper authentication; (3) major changes can be made only once a year (during the summer break); (4) class schedules require a high degree of availability and robustness.

To compensate for the lack of any access control mechanisms both in the PC firmware and in DOS, we developed a set of utilities that ran as a front-end to DOS and which implemented authentication and access control by accessing (through a simple network protocol) a server-resident account database. Once in place this framework was easily extended to include the information to accurately track logon and logoff activity on the lab stations.

Note that the goal of this effort was to offer a reasonably stable environment where helpdesk operators and tutors can turn their attention to assisting the users, instead of needlessly worrying about basic system integrity. It was not designed to cover any deliberate attempt to circumvent the access control features.

A cost-effective method of low-level access control is achieved by plugging a PROM on the PC's network controller board. The PROM hooks into the system boot-strap process, allowing the default boot sequence to be replaced by one that effects an authentication transaction with a central accounts database implemented on a Unix server.

The PROM itself merely contains the necessary driver and networking code to retrieve a second-stage boot program (referred to here as the *PC monitor program*, analogous to what is commonly found built-in on professional workstations) using the TFTP protocol. This PC monitor program then implements the policies for authentication and access control by consulting a service running on a Unix host.

A simple UDP-based protocol is used to communicate with the server. All necessary network parameters that the PC monitor program needs are retrieved using the BOOTP protocol, including the address of the server running the authentication service which is implemented as a BOOTP 'vendor extension'.

The PC monitor program collects the user credentials and presents these to the server for authentication. If authentication is successful, the monitor program receives a set of capabilities based on the user-id, which are used to determine how the boot process can continue.

Normally, the only way to progress is to boot from the local disk holding the regular operating system. However, administrative accounts may receive the capability to boot the machine from, for example, a diskette station or a maintenance partition on the local disk. In any case, the monitor program reports its actions to another service process for the purpose of maintaining login session statistics. As soon as the monitor program is loaded a 'boot' event is generated that marks the time of a machine start. A 'login' event is generated when a user is successfully authenticated and has selected a valid device or partition to boot from. These events allow the server to maintain a history of login activity on all lab stations, similar to the UTMP records that can be found on Unix systems. This history of login sessions can be effectively used to generate both instantaneous and long-term usage statistics.

2. Transition to NT workstation

Early 1997 we began to prepare for the installation of Windows NT Workstation on all lab stations.

It was deemed desirable to continue to operate in some form the UTMP system that had already proved to be a useful tool in the pre-NT era. Indeed, the system's multi-tasking features open the prospect of extending its abilities by continually monitoring and reporting on the system status instead of being active only as a front-end to the operating system.

Though before undertaking the porting job, several other constraints were to be taken into consideration. Whereas the time of introduction of NT (driven by class timetables) in the student labs was set for the summer of 1997, our network server configuration remained locked into supporting existing PCs at other departments for some time to come, precluding any radical changes in our network server setup. The network servers all run Novell's NetWare 4 operating system and account management is based on the NetWare Directory Services (NDS).

Thus, the Windows NT installation must co-operate with the existing server network installation using Novell's network client software for Windows NT, including automatic local user account management based on the NDS database.

It follows that the envisioned port of the UTMP session logging mechanism had to fit into this hybrid environment. Fortunately, implementing UTMP on top of NT's audit-event capabilities, as explained in the next section, nicely de-couples it from the lower-level authentication packages employed by NT.

Another important consideration is the organization of workstation software maintenance. Our goal is to avoid any manual intervention to install or upgrade software components. In particular, there should be no need for anyone to be logged on to the console for these installations or upgrades to be accomplished. In addition, any mechanism deployed for this purpose must be able to strictly separate machine-specific and user-specific parts of the installation process.

Since none of the commercial solutions we explored in early 1997 (e.g. McAfee Brightworks and Microsoft SMS v1.1) met the demands posed by the hybrid environment as sketched above, we decided to develop our own. The results of this project are discussed in section 5.

3. The NT utmp service

Since NT offers a mature operating system environment on the lab stations, many functions of the original PC monitor program became obsolete. For instance, access to the local machine is now controlled by NT's security subsystem, assisted by a NetWare component that is responsible for authentication using the NetWare directory services; obviating the home-grown authentication mechanism developed for DOS-based installations.

The monitor program is still retained however to control the boot process. As before, it only allows booting from the device holding the default operating system unless proper credentials are presented that enable other bootable media for maintenance purposes.

Since the PC monitor program is no longer in a position to obtain the information to generate UTMP records, this functionality has been moved into an NT service program. The operation of this service relies on the system's built-in auditing features. Several groups of security-related facilities can be independently enabled to generate *security audit events*. These events are collected by the system in the Security Event Log, and can be examined by a properly privileged process. A description of the general framework for event logging can be found in the NT Resource kit[1]; programmatic details on accessing the system event logs are given in the MSDN documentation[2]. The UTMP service uses the LOGON/LOGOFF category of auditing events that are generated by the so called Authentication Packages, such as the Graphical Identification and Authentication modules (GINAs) and SMB network connection authenticator (showing up in the event logs as the KsecDD module).

Within a given category each event is uniquely identified by its event-id. The UTMP service looks for just two types of events in the Logon/Logoff category: SUCCESSFUL LOGON and SUCCESSFUL LOGOFF. Associated with each event type is a collection of additional data, the interpretation of which is specific to the event-id.

The LOGON event carries the following event-specific data:[1]

- the user name,
- the domain name to which the user belongs,
- a logon id, which is a unique identifier for each logon session,
- logon type (e.g. *interactive*, *batch*, *service*, etc.),
- a logon process,
- the name of the authentication package,
- the name of the remote machine (if any) where the session originates.

[1] This additional data can be easily viewed using the EventViewer utility that comes with Windows NT.

The LOGOFF event carries these additional data:

- user name,
- the domain name to which the user belongs,
- the logon id, the unique identifier for the logon session,
- the logon type.

The UTMP service program utilizes this data to keep track of logon sessions on the machine. In particular, it uses the <logon id> that is passed along with each LOGON and LOGOFF audit event to pair the events that mark the start and end of logon sessions.

In addition to the audit events in the LOGON/LOGOFF category, the NT system events CTRL_LOGOFF_EVENT and CTRL_SHUTDOWN_EVENT, which are generated when a logon session on the console is terminated, are also monitored. This is done to safeguard against latencies in the generation of the LOGOFF audit events

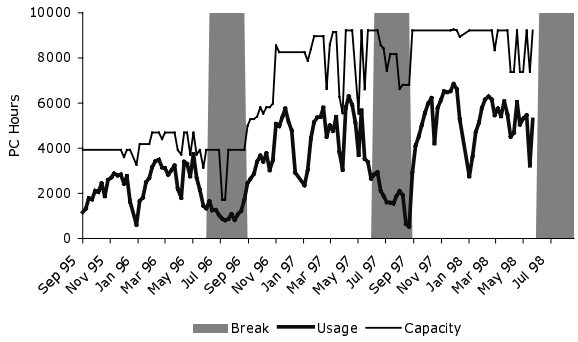

Figure 1: available capacity vs. usage spanning three academic seasons

as a result of inadequate behaviour of some NT authentication packages[2].

A useful bonus of the fact that the UTMP service tracks logon sessions, is the ability to run an arbitrary program in the context of the local system account at the start or end of a user logon session. This feature allows us to apply and cleanup user-specific modifications to the local machine configuration in support

[2] Apparently, the system generates a LOGOFF audit event only when the last reference to a process in a logon session goes away. Hence, neglecting to close, say, a process or thread handle to any of the user processes postpones the LOGOFF event even though the console may long be vacated. This behaviour was exhibited by an early version of the NetWare logon module (NWGINA). It was also observed in the telnet service that comes with the Windows NT 4.0 resource kit.

of a number of ignorant applications without having to relax standard security levels.

4. Displaying session data

The data collected by the UTMP daemon enables us to produce both long-term and short-term overviews of the usage of all stations in the labs. Long-term overviews provide valuable information to spot growing capacity problems well in advance. The figures illustrate aggregated usage data collected over a three-year period.

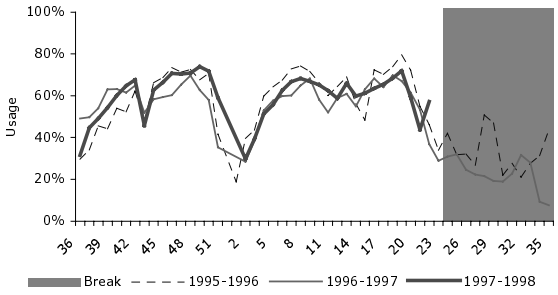

Figure 2: relative utilization in three successive seasons.

Figure 1 shows lab capacity utilization during three academic seasons based on weekly averages. Lab capacity is expressed in *PC hours* which is the combined login session time available on all machines - i.e. the product of the number of lab stations and lab opening hours. The actual capacity in a given week may vary due to holidays and scheduled maintenance. The bold line shows the measured utilization from the UTMP data. The shaded areas represented the midsummer holiday breaks.

Figure 2 illustrates the relative utilization for three academic seasons. It shows that the degree of utilization during each period remains the same despite a marked increase in capacity realized in November 1996. From this observation one may conclude that "demand" for lab capacity still exceeds the "supply". It should be noted that - in our experience - a weekly average utilization degree exceeding 70% actually means "a very busy period".

Lab station usage is also converted in real-time to a set of HTML pages that can then be viewed on any workstation running an HTML browser. The information plotted in these pages includes details about all lab stations known to the UTMP daemon, including the currently logged-on username and the duration of the logon session.

An example of a real-time view can be seen in figure 3. The map layout reflects the actual locations of the workstations in the lab rooms. Color codes are used to give an indication of each station's disposition, for example `idle', `in use' or `not responding', making it easy for support personnel to spot signs of trouble at the earliest opportunity. The data is also displayed on a *kiosk* monitor which arriving students can use to quickly locate an available workstation in one of the lab rooms.

5. The Autoadm service

The development of the mechanism to automatically distribute software and system patches quickly turned into an exercise in glueing together various existing and readily available tools. Things are set in motion at the workstation by an NT service program called *autoadm*[3].

Its sole purpose is to periodically set up a network connection to a central repository containing the full set of tools and data that define the distribution of software in units called *packages*. A package can be anything, ranging from the complete installation of MS Office to adjusting the size of the system's swap file. Note that this service neither places any restrictions on nor offers assistance with the construction of the package contents. However, a simple interface for passing status information and error messages is defined, which all package installation scripts must adhere to. A frequently used tool at our site for actually constructing package installation procedures suitable for unattended installation is McAfee Wincompare.

At the heart of the distribution mechanism is a Perl[3] script, that is started once the network connection to the repository has been set up successfully. This script consults a package configuration file, that defines the packages that are to be installed on the workstation. The package configuration file allows various control parameters to be specified that determine where and when a particular package should be installed. For example, an arbitrary collection of machines can be named as a group that can be used as optional per-package target parameter in the configuration file.

[3] The installation of this service is integrated with the initial installation of NT, so its services are available right away without further operator intervention.

The results of an attempt to install a package are recorded in two locations: locally in the machine's system registry and remotely by opening a network connection to a service process designed to assist the NT package installation procedure.

This service process controls the package installation in several ways:

(1) Establishing the workstation's identity by requesting its unique installation key and verifying it by engaging in a simple challenge/reply using a pair of cryptographic keys assigned to each workstation[4].

(2) Offering a load-balancing option to control the pace when installing large numbers of machines concurrently.

(3) Maintaining a centralized logging facility through which it is easy to monitor the progression of package installations on all workstations.

(4) Providing a secure channel on which auxiliary (presumably sensitive) package installation parameters can be transported, deploying the same cryptographic keys that are used in the authentication step in (1). For instance, we use this facility to periodically change the local `Administrator` password on all NT machines.

The central repository is located on a Unix host running Samba[4], which makes it easy to connect to using NT's native SMB protocol. Thus, the service is functional immediately after the initial NT installation is complete and will automatically extend the operating system installation by processing the packages prepared on the distribution repository.

So far, the description of the automatic package distribution mechanism pertains to packages targeted at machines. While most packages are machine-specific and indeed are scheduled to run at a point of time when no one is logged on to the machine, it is sometimes necessary to effect accompanying changes in the user's context. A completely analogous method is used to distribute and administer such user-specific packages. In fact, the same package configuration file format and perl script is used to accomplish the installation of user-specific packages. The distinction between user and machine packages is entirely the result of the different environment in which the package distribution tools operate, which can be summarized thus: (1) the package update script runs as part of the user's login script; (2) a different - user accessible - package repository is used to hold the package data; (3) package installations are registered in the user portion of the NT registry; (4) the results are logged in a separate log file on the server.

[4] Assignment of the unique identifier and the cryptographic keys are also part of the initial installation procedure.

6. The future

The package installation mechanism was born out of necessity to fit the parameters of our environment. It shows that combining the strengths of several proven and readily available tools can achieve a marked improvement in tractability of a large desktop network. A weakness of the current system can be found in the handling of the ever-increasing amount of logged data. Obviously, employing a mature database system would help out in that area. Hence, we'll continue to look at off-the-shelf solutions as they arrive at the marketplace.

Likewise, the data generated by the UTMP service is also in need of a more sophisticated data management system. The accumulation of login session data spanning several years is starting to stretch the capability of the currently used tools to process the collected information efficiently. This topic will be addressed in a future revision.

7. References

1. Windows NT workstation 4.0 Resource kit; Microsoft Press.

2. Microsoft Developer Network Library; Microsoft Corporation

3. Larry Wall, Tom Christiansen & Randal L. Schwartz, Programming Perl; O'Reilly 1996

4. SAMBA; http://samba.anu.edu.au/samba/samba.html

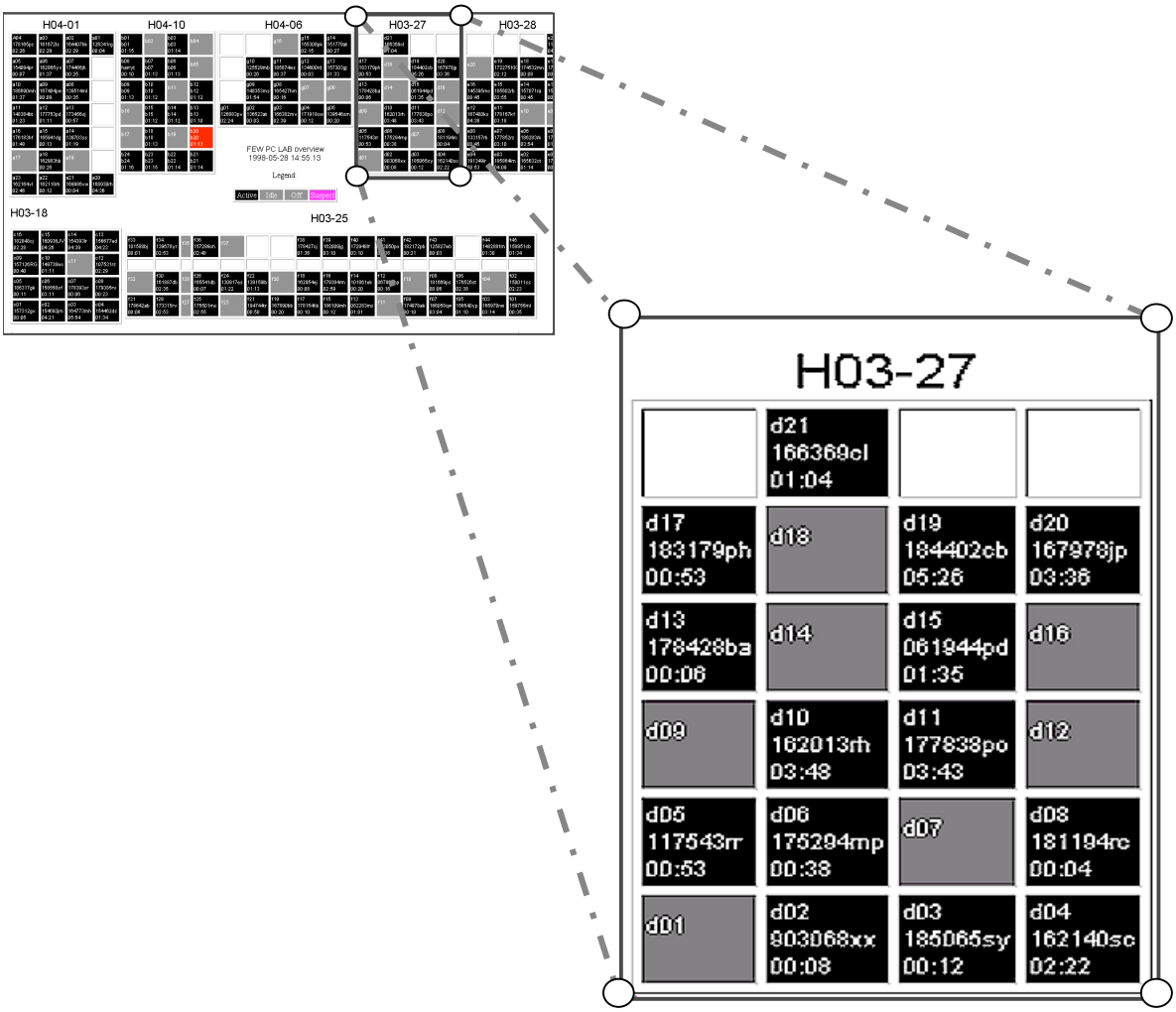

Figure 3: browser view snapshot of real-time utilization