

RAMS: A Web-Based Resource Management System

*Mike Kienenberger, Ming Jiang, and Jayashree Harikumar,
Arctic Region Supercomputing Center, Fairbanks, Alaska*

ABSTRACT: *Resource Administration and Management System (RAMS) is an integrated, demand-driven hierarchical system with a web-based graphic user interface. The integrated system is powerful yet easy-to-use, easy-to-manage, and allows users to obtain up-to-the-minute information for any resource with minimal system overhead. RAMS encapsulates existing system resources (such as pacct files, quota, UDB) into a central data server. Users are organized into a hierarchy by login ids and account ids. A regular Web client can browse the user hierarchy, and retrieve/modify resource information. The server dynamically generates all web pages according to a user's system privilege levels. Kernel table data combined with process termination records provide a comprehensive snapshot of resource usage.*

Introduction

UNIX operating systems generally track interactive session times for each user account and resource usage of running jobs. Some UNIX systems allow disk quotas to be set and enforced for user accounts to control disk usage and perform command logging to generate process termination records. However, accounting in traditional UNIX systems exists to prevent resource abuse rather than to actively track cumulative resource usage. The few systems that charge money for access do so by charging for user interactive connection time and disk usage.

UNICOS (Cray Research, Inc., (CRI)'s version of a UNIX-compatible system) works on a per-process basis and charges the process owner for memory and cpu usage when the process completes. System managers can also use the UNICOS project identifier feature to perform project accounting instead of job or user accounting. System administrators can configure a separate system to check disk usage. However the Cray System Accounting (CSA) is not a real-time system.

The Arctic Region Supercomputing Center (ARSC) uses an accounting system called "resman" from the San Diego Super-computer Center (SDSC). SDSC's cpu quota system ("resman") tracks resource usage by projects and also enforces usage quotas. However, "resman" uses 16 minutes of cpu time each day. Additionally, "resman" charges for jobs lost in a system crash. "resman" has no graphic user interface and requires users to learn a complicated and limited set of command-line programs. These command-line programs do not generate enough useful report information and require extra scripts to re-process the output.

Resource Administration Management System (RAMS) is an integrated, demand-driven hierarchical system with a Web-based graphic user interface. A regular Web client (Lynx, Mosaic, or Netscape) can connect to the server, browse the user hierarchy, and retrieve and/or modify resource information. The server generates all web pages dynamically according to the user's system privileges. The demand-driven nature of RAMS reduces the cost of real-time resource tracking by computing resource usage only when a user requests information. RAMS avoids charging users for lost jobs by using process termination records to charge for jobs. Kernel table data used in conjunction with process termination records provide a comprehensive snapshot of resource usage.

Features

We considered the following system requirements when we designed RAMS:

Accuracy

Operating system and program crashes are inevitable. RAMS restarts without intervention after a crash. Resource charges are neither lost due to interrupted calculations nor are resource charges counted twice due to restarting interrupted calculations. Service units are the billable measure of resource usage. RAMS converts each resource used into service units and charges the service units to the respective project and user. Each resource is multiplied by a weight factor, to normalize the costs of using various types of resources.

Comprehensiveness

RAMS tracks Cray Y-MP cpu time, Cray Y-MP memory, Cray T3D cpu time, disk space, and tape space used by projects. For resource management flexibility the account structure at ARSC is of a hierarchical nature. We organizationally group each user and project under another account with the exception of the root hierarchical account. We group all accounts under the root hierarchical account and all user accounts under multiple project accounts or one project account. However, each project and hierarchy account appears only under one account. Each project and user account pair form a unique leaf node in the accounting hierarchy tree. RAMS tracks resource usage separately for each project/user pair.

Management

RAMS categorizes users as general users, project managers and administrative users. General users can perform queries on their own accounts and can view past and current resource usage information for their own accounts. Project managers of a project can view past and current resource usage of their account(s) as well as set current resource distributions for the users in the project. Administrative users can view past and current resource usage for all accounts, and modify the resource allocation for any project or user account. Administrative users can also create and delete all types of accounts and grant and revoke project manager status to a user for a particular project account. Additionally, administrative users can change the weights for each resource available to users while the resource management system is running.

Real-time Information

RAMS provides real-time information regarding the use of resources. Users can view current resource usage so as to request additional resources when required.

Integration with Existing Systems

RAMS takes advantage of the existing management systems. RAMS uses the UDB system to track account information, use, the disk quota and "diskusg" programs to track disk usage, and process termination records to track cpu usage and memory usage. RAMS provides a single unified tool for adding, removing, and modifying accounts and a single interface to perform all management actions where applicable.

Security

RAMS provides access to the accounting data based on the user definition. RAMS logs all transactions made to change or view resource management data. The log entry indicates the attempted operation, whether the operation succeeded or failed, the reason for failure if the operation failed, and the uid or username of the user making the request if user authentication failed. RAMS performs user authentication with the same username/password scheme required to access the user account.

Graphic User Interface

Access to RAMS is through a graphic user interface (GUI). With a GUI, graphs that show resource usage and cost over

continuous periods of time and reports that show resource usage and cost for specific time intervals are easily generated.

Maintainability

We designed RAMS in a modular fashion to incorporate future additions and changes. RAMS is well documented so that the maintainer can follow the logic of the program. RAMS can support new accounting functions without requiring a redesign of the entire system.

Benefits

Hierarchical Account Organization

A hierarchical account is an additional grouping of project accounts. One cannot directly perform accounting operations on a hierarchical account. Instead, the accounting system must translate operations on a hierarchical account into a series of operations for each of the project accounts under the hierarchical account and combine the results of each operation to produce a single result, giving the illusion of a hierarchical account consisting of the member project and user accounts. A hierarchical organization allows the programmer to define an extended behavior for UNICOS tools. For example, the quota system works only for project and user accounts in UNICOS. RAMS automatically computes the quota for a hierarchy project based on the quota of every project account under the hierarchy account and thus extends an UNICOS feature. With RAMS, the user sees no difference in behavior for operations on hierarchy accounts and project accounts.

UNICOS can represent a hierarchical organization of accounts by using the User Database (UDB) fields to specify a list of parent accounts for each account. UNICOS associates a parent account with each resource record, whether the record is a running process record, a terminated process record, or a file. Parents of user accounts automatically become project accounts while parents of project accounts automatically become hierarchy accounts. However, the UDB enforces no limits on the values that can be contained in these fields. RAMS incorporates the above features of UNICOS while maintaining a consistent hierarchy structure. CRI does not currently distribute software which utilizes hierarchical account organization as described above. The only CRI-supplied tool that works on a hierarchical account organization is the fair-share scheduler, and this tool uses a specialized field named 'resgrp' to represent the hierarchy. However, the 'resgrp' field is not suitable for the purposes of project accounting as an user account can have more than one parent and an administrator may only specify one parent account for each account using the 'resgrp' field. Some CRI-supplied UNICOS tools separate user information by project value, such as CSA and the "diskusg" program, but no CRI-supplied tools provide support to view data by hierarchy. The only known tool that provides hierarchy data is the fair-share scheduler.

Integration with Existing Accounting Systems

RAMS uses existing accounting tools whenever possible. The UNICOS kernel tracks cpu and memory information on

process termination. CRI supplies applications for tracking disk quotas and disk usage. Instead of recreating these systems, RAMS uses these tools. This strategy keeps RAMS simple, consistent, and maintainable. Improvements to the underlying tools automatically result in improvements to RAMS. Just as RAMS integrates accounting systems from a server perspective, clients of RAMS integrate accounting systems from a client's (user's) perspective. Users instruct the RAMS client to perform all of the standard necessary operations for account monitoring and management. A single client eliminates the need for users to learn many different command-line programs which often require cryptic syntax. The client program can transparently translate user operations into a series of commands to execute existing command-line programs and return the combined results back to the user. A RAMS client allows simultaneous access to relevant information from the RAM server, the UDB, the disk quota system, and other systems.

Demand-driven, Real-time Data Access

RAMS is a demand-driven, real-time data access system. Users and administrators alike can get an up-to-date report of resource usage and allocation from RAMS. Users can request additional resources as soon as resources run short, and administrators and project managers can modify the resources allocated to an account as soon as such a change is necessary. "resman" provides real-time information about resource usage by checking resource usage every second. However, the average cost of checking UNICOS kernel tables each second is sixteen minutes of cpu time per day on the ARSC Cray Y-MP. Such checking is also unnecessary since accounting information is not needed every second. In addition, "resman" is highly susceptible to system crashes, both of the operating system and of the accounting system. Accounting system crashes cause discrepancies between actual resource usage and recorded resource usage since the accounting system cannot track resource usage when the system is not running. Also, since jobs are tracked by the second rather than at process termination, handling user refunds for jobs that cannot complete due to operating system crashes is practically impossible.

On the other extreme, CSA runs only once each night. Users and administrators have little control over resource management until CSA computes the day's resource charges. An advantage of this method is that CSA applies charges to a project only when the user job completes. However, users cannot view current resource usage, nor can the accounting system catch jobs that exceed a project's resource allocation.

RAMS updates accounting information by using a hybrid of each of these extreme methods. RAMS is demand-driven and executes when an user requests information. RAMS has the real-time access to data that real-time systems have, the crash-resistance that periodic accounting systems have, and an operating cost somewhere between the two systems. A disadvantage of RAMS is that it cannot immediately notice accounts that exceed allocation as RAMS is demand driven. At ARSC, automated queries to RAMS cause updates every six hours to

force resource charging and to check for exceeded allocations. For sites more concerned with exceeded allocation, automated queries can be sent at much smaller intervals. At small enough intervals, the demand-driven system acts like a real-time system. In a similar fashion, if only a daily automated query is sent to the resource tracking subsystem, the system acts like a periodic accounting system.

RAMS achieves hybrid accounting functionality by using the periodic data source for past user charges and using the real-time data source for estimating charges of running jobs. The periodic data source is the process accounting records which the kernel creates each time a process terminates. Using these records as the basis for user charges prevents data inconsistencies due to crashes. The real-time data is the process and session tables of the UNICOS kernel which contain information about currently-running jobs. RAMS adds these two sources of information and gets an up-to-date view of total resource usage.

Web-based User Interface

The two environments that span the platforms of ARSC users are the X Window System (X) and the world-wide web (WWW). Each environment is available for Macintosh computers and all local UNIX workstations. However, the world-wide web has the advantage of also being available on text-based terminals and is easier to program. Programming in X requires building entire applications from scratch while programming for WWW consists entirely of creating formatted web pages since Web clients already exist to interpret and display created pages. Graphical Web clients also have the ability to present information in formatted output which makes information easier to read as well as display image files such as postscript graphs of past resource usage. However, the only control a programmer has over a web client is what pages the client displays and web clients are quite limited in what web clients can do as a user interface. Fortunately, a customized WWW server used in conjunction with web clients meets the needs of the RAMS project. A custom-designed web server acts as an agent for each user's web client and transfers information between RAMS and users.

Implementation

RAMS chiefly consists of the RAM server, the RAM server daemon, the RAM Infoserver daemon, and RAMS clients. These four pieces interact with CSA, the kernel session table, the "diskusg" system, "cron", and the disk quota system.

The RAM server is responsible for recording, reporting, controlling, and enforcing resource usage and allocation. The server uses data shared with CSA, data available in the kernel tables, and "diskusg"-generated data to track resource usage. It reads the account organizational structure for accounts from the public UDB.

The RAM server daemon's primary purpose is to ensure that RAM server runs only when necessary.

The RAM server daemon transfers requests between clients of the RAM server system and RAM server. The server daemon

continually runs and constantly listens for new queries from the client software like the RAM Infoserver daemon and forwards queries to the RAM server. The RAM server daemon will restart the RAM server if the RAM server is not running when a request arrives. After sending a query, the daemon waits for a reply and sends the reply to the client which sent the query. Together with the RAM server, the RAM server daemon constitutes the RAMserver resource tracking subsystem.

The RAM Infoserver daemon takes requests from RAMS clients and queries the appropriate system to handle the request. A RAMS client can acquire and modify data for a variety of systems without needing to understand what those systems may be. A RAMS client can also submit a single query, such as creating a new account, and the Infoserver daemon sends the appropriate actions out to each subsystem, such as the UDB, the disk quota system, and the RAM server required to create the new account. The Infoserver daemon generally directly handles queries dealing with the UDB as the daemon needs to load UDB information for each query to verify username/password pairs as well as to generate an up-to-date hierarchy consisting of only those accounts to which the user making the query should have access. If the query is a hierarchy account, the Infoserver daemon breaks the hierarchy account query into a series of project queries, one for each project account under the hierarchy account. The Infoserver daemon sends the list of projects as arguments to an external script which performs a quota command on each argument and returns an appropriately-formatted result for each project. The Infoserver daemon combines the returned results appropriately and returns the results to the RAMS client which made the query.

RAMS clients provide the simple, integrated user interface to accounting systems. Currently, the only available production RAMS client is a specialized world-wide web server called RAM web. Unlike web servers that store pages as local files, the RAM web dynamically generates all page requests internally. By doing so RAM web quickly builds menus and forms based on the accounting information of the connecting user. Each page request creates a separate web client record. The web server builds the request from continuously-arriving pieces of data and processes the request after all of the data necessary to build a request arrives. Each time a web client asks for a page, the web server attempts to find the user's registration record. Registration is necessary since RAM web is not a traditional stateless web client. RAM web associates report generation preferences and the partial account hierarchy consisting of accounts to which the user has access with each user connecting to the web server. If the client is not registered, the server attempts to register the client from the hostname and password. If the username/password pair is not acceptable, the web server will not register the user, and RAM web will not distribute pages to the user. RAM web can immediately answer some page requests, but for others it forwards the query to the RAM Infoserver daemon for processing. RAM web then formats the results and presents the results to the user through the web client.

Conclusion

RAMS uses a hierarchical accounting system and integrates several UNICOS accounting systems under one graphic user interface. RAMS produces accurate resource tracking and is a secure and maintainable system. We have designed RAMS in a modular fashion to incorporate future additions and changes.

