Automating Printing Configuration

Jon Finke – Rensselaer Polytechnic Institute

ABSTRACT

Maintaining the printing configuration files for a large site (400 machines, 60 public printers, 40 private printers, 30 print spoolers) can be a major job. At RPI, we developed a system that will automatically generate the printer configuration file for any machine, depending on what printers are driven by that machine. It also allows us to only have a printer appear only on a subset of machines, rather than on all machines.

This paper describes the design and deployment of the system. We use a relational database to manage the printer information, printer type attributes, host grouping and to track hosts using the system. All sources and related information are available for anonymous FTP.

Motivation

One of the first parts of the Simon project[1], was a program called printmaster, which was used to generate the /etc/printcap file for any of our workstations or print servers.[2] While it did have attribute matching for printer types, it did not support host groups, and in fact did not even have a UI to make changes. All configuration info was entered using SQL[3] scripts via the Oracle *SQL*Plus* program. An even bigger problem, was that the configuration generation program ran on a central machine and "pushed" the configuration out to the client host. What finally killed this version, was that we had lost source to one of the critical libraries, so we were unable to make changes in the program.

Despite some of the operational problems with the original program, the basic design was solid, and a few years of operational experience made it very easy to update the design and re-implement it (with a few improvements) on our production systems. We considered going to a double entry approach suggested by David Wood[1], but decided that the ability to restrict printers to just a subset of hosts, as well having it fit with our general system administration model ruled the day and we did the rewrite.

This resulted in three new programs,

etcprintcap which generates the actual printcap file and installs it if needed,

printmaster which provides a user interface to edit printers and printer attributes, and finally,

printcap_status which reports on what version printcap file each host has, and when that host last checked in.

We now have the *etcprintcap* program running on all of our spooler machines via cron every night. We also have another cron job on the database machine that refreshes the common printcap[4] file that the rest of the generic machines use. This has made it quite easy to add new printers to the system. Just run printmaster, answer a few questions (a few more if it is a new printer type), and by the next morning, the correct definitions are installed on all of the machines.

Printers and Printer attributes

The printmaster program is used to maintain the two primary tables, **PRINTERS** and **PRINTER_ATTR.** In addition, a third table **PRINTCAP_LOG** is used to record when a host gets a new printcap file, and the last time it has checked the database. These tables are described below. We also make use of some of the Hostmaster tables and some host grouping tables that have been described in other papers.[2]

Each printer has certain characteristics that are unique to that specific printer, such as the name, the spooler machine, the type of printer, the type of connection to the spooler, the host group and so on. This information is stored in the **PRINTERS** table.

PRINTERS Table Definition

id *number* A Simon.Peoplecount.Nextval[5] that uniquely identifies this printer. It is used when other tables need to join to a printer name.

---

[1] Simon is a project we are developing to automate many aspects of computer system administration. It relies heavily on the use of a relational database to manage user, host, and other types of data, and allow later modules to build effectively on earlier work. Simon has been described in a number of papers and presentations, see the **Availability** section for details.

[2] We use a somewhat customized version of the Berkeley lpr print spooling programs for both SunOS and AIX.

[3] SQL is the language used by Oracle and other relational databases for specifying queries and other functions.

[4] Many local files on our machines are maintained out of a common file system (AFS) via package.

longname *char(36)* The long form of the printer name. This is the name that is generally used in displays and usually defines for people, what and where the printer is.

shortname *char(8)* A short form (8 characters or less) of the printer name. This is generally used when a person has to enter the printer name by hand. This name is also used for the spool and log directory trees (as defined in the attributes).

spooler *number* The Simon.Dns_Domains.Id[6] of the host that drives or spools for the printer. This is used to find the hostname for "rm=" entries and when generating printcap files on that host.

printer_type *char(8)* The type of printer being driven. This is matched with the Print_Attr.Printer_Type field when attributes are generated. In practice, we use generic printer types to avoid duplicating too many attributes.

connection_type *char(8)* The type of connection used to connect the printer to the spooler. This is only relevant on the spooler machine. For attribute matching, all remote printers are assumed to have a connection type of "LPD". This is matched to Print_Attr.Connection_Type.

per_page_charge *number* The per page charge for the printer in centi-cents. This can be included in the printcap file, and should also be used by the printer accounting software.

per_job_charge *number* The per job charge for the printer in centi-cents. This can be included in the printcap file and should also be used by the printer accounting software.

budget *number* The Index to the budget that is credited with all charges from this printer. This is used by the printer accounting software. Values here are defined by the accounting system.

spool_from *number* Is the Simon.Dns_Domains.Id of the single host, or Simon.Host _Groups.Group_Id of the host group that contains all the hosts that should have this printer in their printcap file. This allows a printer to only appear on a subset of all hosts.

when_inserted *number* The Simon.Transcount. Nextval of when this record was inserted into the table. This is used to help set the database version.

[5]A *object*.Nextval refers to the next value of a sequence named *object*. Each time *object.Nextval* is referenced, it returns the next value in the sequence (generally, increment by 1.) These are maintained automatically by Oracle.

[6]The format *Object.Member* is generally used to indicate that we are referring to column *Member* of table *Object*. In this case, we are referring to the **Id** column of the **Dns_Domains** table.

when_updated *number* The Simon.Transcount. Nextval of when this record was last changed. This is used to help determine the database version number.

when_marked_for_delete *number* The Simon. Transcount.Nextval if when this record is considered obsolete and should be ignored. This is how we can delete a printer.

comments *char(240)* A space to include a single line of comments on a printer. These comments, if any, will be included as comments in the printcap file.

system_id *number* A Simon.Systems. Id value that can be used to assign location, owner, sys admin and other systems and related attributes to this printer.

when_attributes_updated *number* A Simon.Transcount. Nextval of when an attribute that matches the id, printer type and connection_type of this printer is updated. This is used to trigger a printcap regeneration when just an attribute of a printer changes.

Most of the text that goes into the actual printcap file, is extracted from the **PRINTER_ATTR** table, which allows matching by specific printer, and by combinations of printer type and connection type.

PRINTER_ATTR Table definition

printer *number* The Simon.Printers.Id of the specific printer that this printcap entry is for. If null, connection type and printer type are used for matching.

printer_type *char(8)* An optional printer type used to match with Simon.Printers.Printer_Type.

connection_type *char(8)* An optional connection type use to match with Simon. Printers.Connection_type. During printcap generation, all remote printers are assumed to have the type "LPD".

rank *number* A number used to rank order lines in the printcap file, and to allow more specific attribute matches to "override" more general attribute matches. Each different printcap keyword has a specific rank.

entry *char(120)* The actual entry to be included in the printcap file. A number of macro strings are defined by the etcprintcap program that will be expanded at printcap generation time.

when_inserted *number* The Simon.Transcount. Nextval of when this entry was inserted into the table.

when_updated *number* The Simon.Transcount. Nextval of when this entry was changed in some way.

when_marked_for_delete *number* The Simon. Transcount.Nextval of when this entry is considered obsolete and no longer included in the printcap file.

Consider the following selection from the **PRINTERS** table in Figure 1 (with spooler_id replaced by the actual spooler name).

Short Name	Printer Type	Link Type	Spooler Name
AE325LW	LW	SER	beale.math.rpi.edu
AE217LW	LW	SER	buster.math.rpi.edu
VC215LW	LW	CAP	netserv2.its.rpi.edu

Figure 1: Printers table excerpt

And the following selection from the **PRINTER_ATTR** table in Figure 2. (note: A (null) indicates that no value is set for that particular data element. In this case, it means "don't care".):

We order the attribute lines by the rank value, and by the "closeness" of the match. This will let an attribute for that specific printer override an attribute that only matched for the printer type. This allows us to override one particular line for a given printer, and still have the printer get the standard entries for that particular type. The rank ordering also allows us to maintain a consistent order in the printcap entries. There is also a special link type, "LPD". This is used when matching for printers that are NOT driven by the current host. Assuming we were on beale.math, we would get a printcap file like the one in Figure 3 and for the printcap on buster.math, one like Figure 4.

We can see a couple of special cases here. The VC215LW printer is located on a host *outside* of the systems we administer, and that printer name does not follow the pattern we use. We can override the default remote printer name ("rp") with a specific entry for just this printer. We did a similar thing for the "lp" name for AE217LW.

Printcap generation

With at least a rough idea on how the attribute matching works, lets take a step back and look at the whole process of generating and installing a printcap file. All of our hosts periodically (usually once a day), run a program to update the system configuration and software. On a subset of these, the *etcprintcap* program is run.

Short Name	Printer Type	Link Type	Rank	Entry
(null)	LW	SER	110	lp=/dev/{printer_name}
AE217LW	(null)	SER	110	lp=/dev/ttya
(null)	(null)	(null)	120	sd=/usr/spool/{printer_name}
(null)	(null)	LPD	150	rm={spooler_name}
(null)	LW	LPD	160	rp={printer_name}
VC215LW	LW	LPD	160	rp=sparclw

Figure 2: Printer_Attr table excerpt

```
#
# System printcap file for beale.math.rpi.edu.
# Generated by SIMON on 17-MAY-94 from
# printer DB version 6358435
#
AE325LW|Amos_Eaton_325_Private_LaserWriter:\
       :lp=/dev/AE325LW:\
       :sd=/usr/spool/AE325LW:...

AE217LW|Amos_Eaton_217_Private_LaserWriter:\
       :sd=/usr/spool/AE217LW:\
       :rm=buster.math.its.rpi.edu:\
       :rp=AE217LW:....

VC215LW|VCC_215_Private_LaserWriter:\
       :sd=/usr/spool/VC215LW:\
       :rm=netserv2.its.rpi.edu:\
       :rp=sparclw:...
```

Figure 3: Beale printcap file Excerpt

On startup *etcprintcap* determines the current version of the printer database, and then compares it to the current version of the */etc/printcap* file. If things are up to date, it logs the fact that it checked in the **PRINTCAP_LOG** table and exits. If, on the other hand, the printcap file is out of date, it selects the list of printers for that host (based on host groups), and then for each printer, selects the appropriate attributes, does any macro substitution needed, and writes the file. These steps are explained in more detail in the following sections.

Version numbers

Each printcap file has a version number. This is essentially the maximum value of all the version numbers for each of the printers being included on the machine. Each printer entry in turn, has two version numbers, the maximum value of **Printers.When_Inserted**, **Printers.When_Updated** and **Printers.When_Marked_For_Delete** which gives the printer version number, and the **Printers.When_Attributes_Updated** value which is updated by the *printmaster* program when a matching attribute is changed.

When the printcap file is generated, the newly determined version number is written in the header of the */etc/printcap* file (like in figures 3 and 4). The file version number is determined by reading the first few lines of the file and searching for the string **"database version"** and taking the number after it as the file version number.

Host Groups

One of the requirements of the printcap configuration system, is the ability to have some printers only appear in the printcap files of particular host, or group of hosts. These special case printers ranged from test printers that we were evaluating, to private dot matrix printers attached to someone's PC, to the printer that has the payroll check forms mounted.

```
General_Printcap
   All_Rcs_Hosts
       Engineering_Lab_1
          ELab1-01.rpi.edu
          ELab1-02.rpi.edu
          ...
       Engineering_Lab_2
          ...
   Faculty_Hosts
       Math_Faculty_WS
          beale.math.rpi.edu
          buster.math.rpi.edu
   ...
```

Figure 5: Host Group Tree

We had already done some work with host groupings for generating usage statistics[3], so we decided to build on this existing work. The key to making this work effectively was to allow a group to be a member of another group. In this way, we could define a group, *General_Printcap*, which most of the printers have set as the **Printers.Spool_From** entry. This group formed the top of the host tree as shown in Figure 5. We can see that the machine *ELab1-01.rpi.edu* is a member of host group *Engineering_Lab_1* which in turn is a member of *All_Rcs_Hosts* which is a member of *General_Printcap*. This way, when a new machine comes online, we just add it to the appropriate host group, and gets membership in the other appropriate groups.

```
#
# System printcap file for buster.math.rpi.edu.
# Generated by SIMON on 17-MAY-94 from
# printer database version 6358435
#
AE325LW|Amos_Eaton_325_Private_LaserWriter:\
  :sd=/usr/spool/AE325LW:\
  :rm=beale.math.rpi.edu:\
  :rp=AE325LW:....

AE217LW|Amos_Eaton_217_Private_LaserWriter:\
  :lp=/dev/ttya:\
  :sd=/usr/spool/AE217LW:....

VC215LW|VCC_215_Private_LaserWriter\
  :sd=/usr/spool/VC215LW:\
  :rm=netserv2.its.rpi.edu:\
  :rp=sparclw:...
```

Figure 4: Buster printcap file excerpt

Printers can of course, have a different group set in the **Printers.Spool_From** column. If, for example, the printer **AE217LW** had **Spool_From** set to *Math_Faculty_WS*, the entry for it would only appear on the machines in the Math department. A more detailed explanation of how to do nested groups in included at the end of the paper.

String Substitution

Another key part of the printcap file generation, is being able to specify generic attributes, that then are filled in with the specific information for the current printer and host. As we can see going from the attributes in Figure 2, to the printcap file in Figure 3, the string **"{printer_name}"**is replaced with the appropriate **Printers.Shortname** value.

We have a simple set of macro routines[7] that allow you to load values for a number of keywords, and then give it a line of text, and it will replace any **{keyword}** pattern, for the corresponding value. These keywords are updated for each printer as it is processed. We currently define the following keywords:

Printer_Name which is the lowercase version of **Printers.Shortname**.

printer_type which is the **Printers.Printer_Type** value.

connection_type which is the **Printers.Connection _Type** value.

spooler_name which is the actual hostname that corresponds to **Printers.Spooler**.

Since these macro routines are frequently used with Oracle database, we have added a *load_row* subroutine that loads the current row from the database, using the column names as the keywords. In the printcap example, all of the keywords except **Spooler_Name** are loaded via the *load_row* routine.

Installing a new file

Since the *etcprintcap* program is intended to be run automatically in the early hours of the morning, we need to be very careful to fail in a safe way. To this end, we actually build the new printcap file in */etc/printcap.new*, and check the return codes from *fopen* and *fclose*, and only if everything checks out ok, we then rename */etc/printcap* to */etc/printcap.old* and then */etc/printcap.new* to */etc/printcap*. This does leave a very small window between the two renames where an ill timed system crash could leave a machine without a printcap file, but so far, we have not had any problems. By working to the side, we can also trap things like file system full and other errors.

[7]These are part of the Sandbox routines. See **Availability** for details.

The other part of installing the printcap file, is the creation of the spool and log directories. During the generation of attributes, each attribute line is passed to a routine called *check_for_files* that checks to see if the attribute is specifying a filename. If it is, it checks to see if the file (and parent directory) exist, and if not, create them with appropriate owner and permissions. This is not the prettiest solution, but it seems to work quite well. This would need to be expanded to handle a SysV printing system of course.

PRINTCAP_STATUS

Since the *etcprintcap* program records whenever it checks or updates a printcap file in the **Printcap_Log** table, we also wrote a program, *printcap_status*, the reads all the entries in the table and prints out any that have not reported in in the past 24 hours. It also flags hosts that have recorded a database version of 0. This has been useful in finding hosts that have had problems with the nightly updates and need intervention on the part of a system administrator.

PRINTCAP_LOG Table definition

domain_id *number* The Simon.Dns_Domains. Domain_Id of the host that this record is for.

pcap_version *number* The version number of the printcap file from this host.

last_check *date(7)* The date on when this host last checked the version of the printcap file.

last_update *date(7)* The date on when this host last updated the printcap file.

PRINTMASTER

The key to making all of this usable, is the *printmaster* program. This is an interactive program that assists you in manipulating printer definitions and printer attribute definitions. For the most part, the commands give you a way to display and update different columns in the **Printers** and **Printer_Attr** tables, automatically converting from the internal formats[8] to a more human version as needed. It also provides some simple searching and selection functions to simplify locating a particular printer or attribute. The following top level commands are defined, most of which are for manipulating entries in **Printers**.

list Lists all the printers in the database. This isn't actually used very much anymore, as we have around 150 printers currently defined.

destroy This destroys a printer entry. It prompts you for a printer name. The actual destruction is

[8]For example, **Printers.Spooler** is simply a number, which then has to be joined with the **Dns_Domains.Id** to actually produce a host name.

done by setting the **Printers. When _Marked_For_Delete** field to **Transcount. Nextval**. If needed, we could implement an undelete command.

detail This lists relevant information about a specific printer, basically columns selected from the **Printers** table.

entry This will generate the entry as would appear in the printcap file on both the spooler machine, and some other remote machine. This is very handy for checking that you have all the desired attributes set, and is much quicker than generating a full copy of */etc/printcap*.

comments This allows you to set or clear a few lines of comments that will be included in the start of the printcap file.

connection Allows you to change the **Printers.Connection_Type** of the printer.

printer Allows you to change the **Printers.Printer_Type** of the printer.

group Allows you to change the host group that gets this printer in it's */etc/printcap* file. This can also be a single host if desired. While *printmaster* doesn't currently support this option, setting this to (Null) would have the effect of removing the printer from all printcaps, yet keep the name reserved.

shortname Lets you change the shortname of a printer. I have no idea why I never implemented a "longname" command.

spooler Lets you change the machine that drives the printer. This machine must be in the Hostmaster tables.

alias Lets you assign another alias for the printer. This information is stored in the **Printer_Alias** table. This is a small table with basically just the **Printer.Id** value and the text of the alias.

create Allows you to define a new printer. If you are adding a new type of printer, you will need to add attributes as well.

attributes Drops you into the attributes subcommand mode, described below.

touch Updates the **Printers.When_Updated** field. This will cause new printcap files to be generated. This is not normally needed, as any of the commands to change something, will also update these field automatically. It is handy for development and testing.

types Lists all the **Printer_Types** and **Connection_Types** currently defined in both **Printers** and **Printer_Attr**, and the number of occurrences of each. This is often useful to remind you what types you have already defined.

hosts This lists all the hosts that have printers attached to them. The first part of the list are hosts with more than one printer, and this includes the number of printers. The second part of the list, are all the hosts with just one printer.

stop Lets you out of the program.

To work on attributes, you select the **attributes** command and you then get the attributes menu. While in the attribute mode, the program defaults to the last connection type, printer type and rank that you specified. This can save a lot of typing when working on attributes.

clear Clears the default attributes from the previous command.

clone Copies a set of attributes from one printer type to a new printer type. We often get different printers that are all driven the same way, and the quickest way to load a new set of attributes, is to copy them from a similar printer and change the printer type on the fly.

connection Sets, clears or changes the **Printer_Attr.Connection_Type** on a particular attribute.

create Creates a new attribute. You are prompted for the different fields needed.

delete Deletes an attribute. As with printers, this is done by setting the **Printer_Attr.When _Marked_For_Delete** field. If needed, we could add an undelete command.

entry Changes the text of particular attribute.

list Lets you list attributes. You can specify any combination of printer name, printer type, connection type and rank to reduce the size of the list.

printer Sets, clears or changes the printer for this attribute. (Many attributes are not assigned to a specific printer.)

ranks Prints out each rank, the first 3 characters of one of the entries, and the number of attributes for that rank. This is very important to help keep the ranks and attributes matched up.

stop Returns you to the printer command menu.

The *printmaster* program automatically updates the **When_Updated** column when a printer entry is changed, and updates the **When_Attributes_Updated** column for all matching printers when an attribute is updated. This helps *etcprintcap* detect when just a printer attribute has changed.

Operational Status

The current version of the printmaster system has been in production use for about 2 months now (we were very glad to retire the old mainframe based system) and we have been very pleased with the results. Before we had automated the */etc/printcap* generation, adding a new printer was a hassle, often taking several hours to coax the old program to run, and we always had a number of machines with printcap files that were months out of date. This was such a problem, that we really couldn't afford to offer a private printer service on faculty workstations. Even when we would install one, it generally took several hours of staff time to coordinate it, and often the change would be made by hand, and sometime later on, the faculty member would want access to some new printer, and the definition for their private printer would get overwritten. With the current system, I get a request with the required info, I spend 2 minutes with the *printmaster* program, and by the next morning, not only does the faculty workstation have the printer definition, all the rest of the campus hosts have it as well.

Many of our printers are connected via AppleTalk, and we drive them via CAP. Given our network layout, the choice of what print spooler machine to use to drive a given printer was often pretty arbitrary, and in the past, often depended on which spooler had a printcap file we could regenerate at the time. Now it is trivial to put the printer on any spooler, and moving a printer between spoolers is a single command, and waiting to the next day. **Note:** This can be a problem if the spoolers get updated in the wrong order. For moving active printers, you should be sure the update the new spooler before the old spooler, otherwise you may get jobs bouncing between machines.

Future Directions

The printcap project has laid some groundwork for some other sys admin automation projects. Given the host group support, and some modifications of the techniques used to extract the lines for the printcap file itself, we plan on implementing a system to generate other system configuration files, such as crontab, syslog.conf, etc. This will be very useful as these files are generally the same for all of the systems, except for a few systems that need one line added or changed.

The *printmaster* program itself can use some more safety checks. Right now, it is possible for someone to mess things up a bit. An example of this are the ranks that are used in the **Printer_Attr** table. Each of the two character capability codes as described in *printcap*(5) should each have it's own, consistent rank. This is only enforced by convention now, but should be enforced by the software. We don't have a lot of policy built into the program. While this may be a good thing in some respects, it does require a higher level of understanding and training on the part of folks who use it.

There are also some things that are not currently being used, such as the per page charge. I expect that as we rewrite the printer accounting system, these will start being used. Right now we have two lists of printers, the one in **Printers** and the one used by the current printer account system. That is one list too many in my opinion. We also will eventually plug into the **Systems** table, but that is mostly for our own record keeping and not really related to printcap generation. Once thing that will give us however, is a standard Simon format location for each printer, which would be a prerequisite to providing a printer location service (*Where is the closest printer to this workstation?*).

System V support

While it is likely that RPI will continue to use the Berkeley lpd/lpr/lpq suite of programs for some time to come (the changes we have made to it could fill another paper....) It would be close to trivial to modify Printmaster to generate /etc/qconfig and other SysV files. This would require changes to the code that generates the headers for each printer stanza (and in fact, it would make sense to move most of this to **Printer_Attr** anyway), and the *check_for_files* would need to be updated.

Supporting both Berkeley and SysV at the same time would be a bit more work. You would want to add another field to **Printer_Attr** called something like **SpoolSys_Type** and then set that to **BSD** or **SYSV** for each attribute. What might be even simpler, is to define that BSD systems use ranks 1-9999, SysV uses 10000-19999, and so on. This would limit the changes to just the *etcprintcap* program. Not counting time required to double enter all of the attributes, the program changes would be on the order of a few hours.

POSIX – Palladium

I have no direct experience with either Palladium[4][5] or POSIX P1003.7.1/D6[6], although I have read some papers on Palladium and one version of the POSIX draft. All sources concentrated on the actual printing process, and did not say much about managing the configuration. It appears that print servers still have printcap files, which could certainly benefit from this project. The clients get information from the Hesiod server, and it may be possible to provide the input data to Hesiod from this as well. However, this is simply speculation on my part.

Installation Hints

As with any of the Simon modules, you need to obtain the Sandbox and Rsql modules. We (RPI) developed an API that sits on top of the oracle call interface (OCI). This also includes a network portion so that the program can run on a different machine than the database server. These have also been ported to run on top of Sybase, I can put you in touch with those folks if needed.

If you are willing to give up the host grouping functionality, you could remote the host group check, and replace the **Printers.Spooler** with the actual spooler name. This would allow you to avoid installing the Simon Hostmaster[9] module, although you are exposed to problems with host name changes. A second option would be to install just the **Dns_Domains** table and populate it with hostnames of hosts that will drive printers or will actually be running *etcprintcap*. Basically, this would be like installing Simon Hostmaster, but not actually populating most of the tables. We actually ran this way with the earlier version of printmaster.

Given at least the name part of Simon Hostmaster, the host group support would be trivial to install. It consists of two tables and a program to maintain the group memberships. (We also use host group memberships to generate host access control lists */etc/hosts.lpd* and */etc/netgroup*.) If you find that you also need to manage some part of your host name space, generate RR files for bind, etc, you may want to consider using more of Simon Hostmaster.

Grouping in Oracle

As discussed earlier, one very useful module for printing support, is the host group work. A lot of the system management problems become easier to handle when you can break up machines into logical groups, and then join those groups together into other groups and so on (see **Figure 5**). While in this example we are doing groups of hosts, we use these same techniques for managing groups of unix ids (*/etc/group*) and groups of mail aliases (*/etc/aliases*).

We used two new tables for host groups. The first, **Host_Groups**, is used to hold actual group name and has the following columns of interest:

group_id *number* A Simon.People.Nextval that uniquely identifies this host group. This is a join column, and is based in this table.

group_name *char(64)* The name of the host group. The host group name space is a flat space, with no hierarchy provided by Simon. In use, system admins may wish to follow a naming convention.

The second table is **Host_Group_Members** and is used to hold the membership of groups. The fields that we are interest in here are:

member_id *number* The Dns_Domains.Domain_id or the host_groups.group_id of this entry. Since host ids and group ids are both drawn from the same sequence, they are globally unique.

group_id *number* The host_groups.group_id of this entry.

When a group is created, it is assigned a **Group_Id**. For each member in the group, a row is inserted into the **Host_Group_Members** table with the **Group_Id** and appropriate **Member_Id**. For a normal host entry, this is the **Dns _Domains.Domain_id**, but for a group member, this is the **Group_Id** of the group that is to be a member of the current group. Since both the **Group_Id**s and the **Domain_Id**s are drawn from the same oracle sequence, there are no collisions between the two sets of ids, and a host and a group can never have the same ID number. This makes things very nice when working with groups of groups.

To select the list of printers that we can should include on a given host, we want to find all printers where the **Spool_From** is either the **Domain_Id** of the current host, or the **Group_Id** of a group that contains the current host. To do this, we execute the following bit of SQL code.

```
Select id,shortname,spooler....
  from Simon.Printers
 where spool_from in
   (Select distinct group_id
      from simon.host_group_members
   connect by prior
           group_id = member_id
     start with member_id=$HOSTID
   union select $HOSTID from dual)
```

The clause *where {ITEM} in {LIST}* matches all rows where the value in column {ITEM} is found in the list specified by {LIST}. In this case, we are generating a {LIST} of all the **Group_Id**s that our host is a member of, along with the **Host_Id** of the current host. This combined list is used to select our set of printers. Since we are working entirely with **Host_Id**s and **Group_Id**s here, we don't need the **Host_Groups** at generation time. However, it is used by the *printmaster* program to set **Printers.Spool_From**.

Availability

All the source code for the Printmaster suite of programs, as well as table definitions, source code and additional reference material for the Hostmaster and host group modules are available for anonymous FTP from ftp.rpi.edu. See the file /pub/its-release/Simon.Info for details on where to find everything. Some papers and presentations that discuss other parts of the Simon project are available in /pub/its-papers.

If you have AFS available, you can browse many parts of the Simon tree. Look in */afs/rpi.edu/campus/rpi/simon/printmaster* for this project, and */afs/rpi.edu/campus/rpi/{sql,sandbox, netjack}* for other related parts. The anonymous ftp tree is also available in */afs/rpi.edu/campus /rpi/anon-ftp/1.0/common*.

If you just want to poke around, some information is available via http://www.rpi.edu/~finkej /Simon.html .

[9]Simon Hostmaster is another Simon module that manages all the data that goes into the RR files used for Bind and the */etc/hosts* file. It is a prerequisite for all of the Simon host configuration management modules. It is also freely available from ftp.rpi.edu.

Author Information

Jon Finke graduated from Rensselaer in 1983, where he had provided microcomputer support and communications programming with a BS-ECSE. He continued as a full time staff member in the computer center. From PC communications, he moved into mainframe communications and networking, and then on to Unix support, including a stint in the Nysernet Network Information Center. A charter member of the Workstation Support Group he took over printing development and support and later inherited the Simon project, which has been his primary focus for the past 3 years. Reach him via USMail at RPI; VCC 315; 1108th St; Troy, NY 12180-3590. Reach him electronically at finkej@rpi.edu.

References

[1] Hein, T., & Nemeth, E. "Topics in System Administration I." *Tutorial Notes,* USENIX LISA VII 1993.

[2] Finke, J. "Simon System Management: Hostmaster and Beyond" *Proc. Community Workshop '93,* Simon Fraser University, June 1993.

[3] Finke, J. "Monitoring Usage of Workstations with a Relational Database" *Proc.,* USENIX LISA VIII 1994.

[4] Handspicker, B., Hart, R., & Roman, M. "The Athena Palladium Print System" *Publisher Unknown* 9 Dec 1988.

[5] Peisach, E. "Distributed Printing Within the Athena Environment" *Athena Technical Conference* Copy of presentation slides, date unknown, estimate 1990.

[6] "IEEE std 1003.7.1 Draft System Administration Interface/Printing" *IEEE* October 1992.