SENDS: a Tool for Managing Domain Naming and Electronic Mail in a Large Organization

Jerry Scharf – Sony Electronics
Paul Vixie – Vixie Enterprises

ABSTRACT

Managing the Domain Naming Service and electronic mail routing in a large organization has always been a difficult problem. Systems designed to automate these tasks encounter the basic difficulty that the information needed to maintain an accurate picture of the organization is distributed far more widely than the expertise needed to operate such a system.

This paper describes a simple set of tools that provides automatic central management of host and electronic mail information. It attempts to find a balance between centralized management and distributed autonomy by centralizing the tools and accumulated data and distributing the source of the data. It also centralizes the mail delivery technology. With this, the users and local groups are provided a higher level of service without the loss of control and local administration.

Background

Sony Corporation of America's TCP/IP network evolved the way many other organizations did. Many groups purchased islands of workstations to meet specific needs. At first, the central networking organization ignored these machines as not central to the business, and everyone was happy. The use of Domain Name Service was spotty in deployment, integration, accuracy and quality.

Groups then discovered that they needed to share information, and the corporate networking organization was asked to make the systems work together. After a short period of time, the management of routers and subnets was well controlled, but the naming and electronic mail systems were still variable in coverage and quality. There was some amount of friction between the networking group and some of the more independent groups over control and ownership of information. It was also discovered that very few groups had any technical knowledge of DNS. Concurrently, Sony was starting to migrate applications off the IBM mainframes onto client/server platforms. These are Unix platforms communicating with TCP/IP. This has forced a whole new group of people to deal with the administration of machines and routed networking.

At the same time that this effort started, the central networks group needed to develop a common structure for the name space for the US network. When we discussed this with some of the key groups, all the discussions centered around the inextricable link of domain names and electronic mail addresses. At the same time, certain issues about the e-mail delivery systems came up repeatedly. The most important issue was that it was very difficult to find the e-mail address of a given user. We realized that the management of e-mail should be folded into the DNS management system, providing the users with the increased functions that made the changes worthwhile.

Problems to Solve

Domain Naming Service Administration Issues

It is beyond the scope of this paper to describe DNS, but a brief introduction is needed to highlight some problems to be solved. DNS [1,6,7] was designed to be a hierarchical naming system to allow the ARPAnet to introduce tiered management of hostname to IP address translation. The tiers are produced by the ''delegation'' of a section of the naming tree to a subordinate administration and nameserver.

The delegation of administration by naming hierarchy worked well in the basic designs envisioned when the system was designed, but has since become problematic. Most notably the IP address to name lookup is based on the lexical structure of the dotted quad address, e.g. 10.0.0.1. With the significant use of subnetted networks and the development of Classless Inter Domain Routing (CIDR) [4] this mapping of administration to name often fails to align with the administrative boundaries of the organization.

At the administration level, electronic mail delivery is often considered a separate issue from providing DNS services. When the electronic mail is SMTP based, this is no longer true. DNS provides the mapping of addresses to hosts by the use of Mail eXchanger (MX) records and Address (A) records. There is a further link in that it is often desirable for the mail delivery system to modify addresses in the mail header based on the address.

Mail Delivery Issues

Every organization has a slightly different opinion of how SMTP [2,3,8] based mail delivery should be handled. In particular, the processing of headers is highly variable. At Sony, we added two features to the basic processing of mail headers.

The first feature was the ability to selectively remove the ''host'' from the ''domain'' in the mail headers. The reason for this to be selective is that it is often the case that groups have a common ''aliases'' file, in which case the host can be safely stripped. In the case where the local machine keeps a separate aliases file it is unsafe to strip the host, because user@grouphost.domain may map differently than user@host.domain, and replies to the message could go to the wrong address. We added a flag to the hosts file to indicate this condition, which is converted into a mapping database. It should be noted name mappings that occur in the user's mail reader are not a problem, since those are applied as the message is being built for delivery.

The second feature was the desire to transform the username on the left side of the @ to a firstname_lastname. This needed to be implemented on a group by group basis as groups often have duplicated usernames.

A final necessity is the ability to manage and forward deprecated addresses. One centrally maintained file keeps all this information, which facilitates maintenance, especially group name changes.

Sony's Solution

The design of any system must reflect the basic structure of the organization. In Sony's case, there has been a strong push towards decentralization of control of the MIS functions as it related to systems and applications. These groups can range in size from five people to hundreds. The emerging client/server applications are moving away from the centralization of the mainframe. The networks, on the other hand, are centrally planned and managed. This means that any tool must account for the needs of the groups and network managers alike.

The lack of technical expertise in the individual groups was a key concern that led us to the set of tools we have now. In our survey of the other tools available for managing DNS naming, we found that they implicitly linked the loading of the data into the tool and the operation of the name server. Our goal became to find a way to allow the central networks group to operate a cluster of name servers based on information fed in by the distributed, autonomous groups. These groups, with their individual administrators, are the clients of SENDS.

At the same time, the staff of the Corporate Networks group was extremely overcommitted with

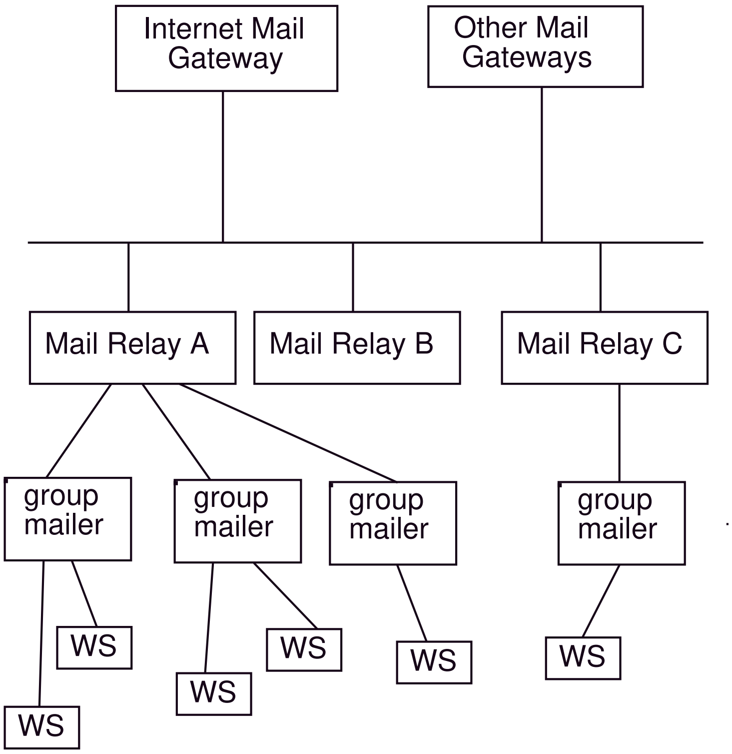

Figure 1: Mail Routing Diagram

the installation of routers and circuits that were being requested. There were no clerical resources to set up a call in line to get IP addresses registered. The groups were concerned that the speed of a clerical solution would hinder their local administration. We needed a tool that automated all the common actions in both normal and abnormal situations.

In looking at the problem of host registration, we realized that the same lack of expertise existed for the SMTP mail delivery. Some groups had very complex mail systems, some of which broke regularly, while others had no mail outside their group at all. Once again we needed to coalesce the mail delivery system into a task that a small number of people could maintain. The goal was to make minimal changes to the mail configuration on a group's mail server to incorporate it into the larger mail delivery system.

We settled on the idea of customizing a small number of machines and distributing them throughout the country as mail relays. These machines are owned and operated by Corporate Networks. The mail to and from the various groups flows through these machines. Each machine is identical in function and acts as a backup for other mail relays. They also provide DNS and NTP [5] time services to their area. We currently have 4 systems, each with the same software and databases, utilizing BSDI/Intel and HP platforms. See Figure 1.

Another direct benefit of centralized mail management is the ability to implement directory services. The form and content of information to be offered in a directory service will change over time, so the input format for the information must be extremely flexible. This is in sharp contrast to the exact information to be loaded into DNS. This led us to have two files, one for host information and the other for mail and directory information.

The lack of global directory services has always been one of the detractions of SMTP mail when compared to integrated mail systems, especially on the PC. Our directory service is bound to the mail delivery mechanism, which gives the groups a strong desire to keep it accurate.

To make the system as flexible as possible, the mechanism for processing the files must be separated from the structure that the namespace takes. The scripts are more complicated because of the fact that namespace overlap is allowed.

For each group that is managed under SENDS, the domains for the mail for the hosts are specified or inferred. The tools then generate all the DNS zone files and email mapping files based on these values. This allows groups to share host domains or email domains with other groups, while preventing naming conflicts. We have used this to allow the different corporate entities within Sony to structure their mail names to reflect their business structure.

In the last 10 years, many people put GUIs on management tools under the flag of ease of use. In our environment, with users spread from IBM mainframes to all flavor of Unix boxes to PCs and Macs, there is no common form of GUI. The only thing that can cover this range of platforms is a text based tool with character conversion. FTP does the conversion, so we use it as the transport with simple text files as the input format.

It also allows local groups to develop tools specific to their needs. They can generate the files with any editor on their local system. We also have groups that generate their files automatically from other data within the group.

SENDS Processing

The heart of the system is the processing of the input files from the users. The tools are a group of *Perl* scripts run from cron. These scripts process the two files that each group maintains, the *hosts* and *users* files. The results of the tools are zone files for the primary DNS server, boot files for the primary and secondary DNS server, and database files distributed to the mail relay machines.

Each group administrator is given an FTP [9] only account on the registry machine. When the group administrator wishes to change the hosts or users files, they use FTP to drop a new version of the file into their FTP drop directory. SENDS processes each new file sequentially, sending mail with the results back to the group. We are using DECWRL's FTP daemon, and have it set up to chroot the admin directly to the ftp drop directory for the group. There is a restricted shell that only allows the group administrator to change the password of the ftp account.

The *hosts* file is the standard */etc/hosts* file used on Unix file systems. The *users* file is a freeform file with key:token pairs on separate lines which combine to make up records. The records are separated by blank lines, similar to uumap entries. A complete set of example input files and generated files are included in Appendix B.

Directory Structure and Control Files

The top level of the SENDS directory contains the directories *data*, *bin* and *lib*. The *bin* and *lib* directories are for the tools, *data* is where the action is; see Figure 2. Under the *data* directory are two fixed directories, *Zones* and *Users*. *Zones* is where the final DNS information lives, and *Users* is where the final mail processing data resides. All subdirectories of *data* that start with a lower case character are treated as domains, and are assumed to have group files. All other directories are ignored.

The domains contain zero or more group directories and only those that start with lower case characters are searched. This allows the SENDS administrator to create a new group, beginning with a starting upper case character so that it will not be processed while being set up. Within each group are the *control* file and four directories: *ftp*, *work*, *output* and *bad*.

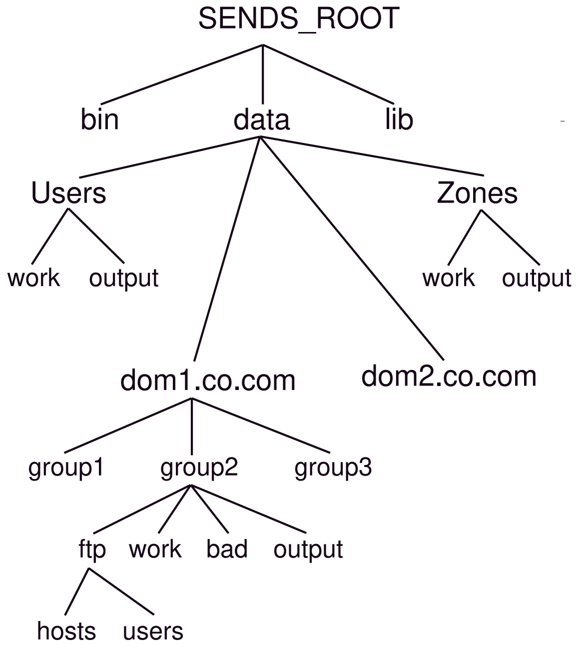

Figure 2: Directory Structure for SENDS

The common control features of the *Perl* scripts are put in a global file, *sends.conf*. The base directory for the SENDS tree is controlled by an environment variable, because it is needed before the config file can be located. The majority of the entries in the config file are template information used in the files generated by SENDS. It also includes the user record parsing requirements. The sample file is included in Appendix A.

The more important parameters are in the per group control file. This controls how the group's data is merged into the final output. The file is named *control*, and lives in the top level of the group directory tree. It has 6 keywords supported as described below.

The *network* lines are the most important in processing the group's host file. They take the form of Access Control List commands *accept* and *deny*, and there is an implicit deny all at the end of the list. We have adopted an unusual form to specify ranges, allowing the form (a-b) in each of the dotted quad locations. This allows ranges to be specified on any boundary, providing more flexibility than bitmasks. It allows the SENDS administrator to give a group a block of subnets with identical ranges inside each subnet. If, for example, addresses above 240 are reserved for routers, and the users aren't allowed to set the network name, this could be described by the one ACL line:

```
network: 10.0.(10-19).(1-239) accept
```

The *dri* is the mail address of a designated responsible individual. There can be multiple *dri* lines, and the results of processing any new group files are sent to all *dris* as well as the SENDS notify address specified in the sends.conf file.

The *relay* keyword is used to specify mail relays for the group and backup mailers for all the hosts in the group. The value must be a fully qualified hostname including a trailing dot. The order of these statements is important, in that the first *relay* line gets MX preference 1, the second 2, and on. There can be more than one host on a *relay* line, in which case all the hosts on that line get MX records with the same value, allowing load sharing.

The *fqdn* specifies the place in the naming tree that the host entries will be placed, ie. host.gr.mycorp.com has the *fqdn* gr.mycorp.com. If this is not specified, it is generated from the directory structure below the data directory for SENDS, as group.domain. The *maildom* specifies the location of the group MX, the base address of the users in the users file, and the address the hosts are mapped to if they do not have the unique-alias flag set. The MX records for the host still go in the *fqdn* location. If a *maildom* is not set, it is assumed to be the *fqdn*.

The *dots-allowed* keyword accepts yes or no, and determines if hostname entries in the host file are allowed to have dots. Hostnames that tail match the current domain with just the host remaining are not considered to have dots. This allows the SENDS administrator to control whether groups are allowed to introduce new levels of hierarchy.

Host File Processing

As was mentioned above, the hosts file is in standard Unix format. We have modified it slightly by permitting magic cookie comment that allows us to have flags for a host. Most important of these is the unique-alias flag, which prevents hostname to groupname mapping on the specific host. We have tightened some of the rules that we felt were ambiguous when converting to DNS. One goal was to allow users to use their existing host files as is, so the network ACLs in the control file are essential.

When a new host file is processed, we first process the hosts files of all the other groups, and construct a series of tables with names and IP addresses used. Then we read the current file. This enforces the first come, first serve allocation of IP addresses and names. Any syntactic errors in the file cause the update to be aborted, and the input file is moved to the bad directory. Other errors include: duplicate IP address, aliases (fully qualified) that match a basename or aliases of another host and a hostname or alias that is qualified but does not tail match the *fqdn* being processed. The hostname is the first name in the list after the IP address, and all the other names are aliases.

The IP address is checked against the network ACLs in the control file. If they do not end with an accept, a warning is produced, the line is ignored, and the processing of the file continues. This allows the group admins to have local additions to their hostfile without causing problems for the DNS namespace. Since almost every Unix hostfile has a 127.0.0.1 address in it, it would be a race to see which domain gets the reverse pointer, or every admin would have to have a separate file with these lines stripped.

Once the processing of a new file has completed successfully, a complete set of forward and reverse zone files for all groups is written into the work subdirectory of *data/Zones*, each with the SOA record removed. Each one is compared to the existing zone file, and if they have changes, the file is moved over and a new SOA file is created. We use the serial number format *yyyymmddnn* for all our SOAs. A new pair of boot files are produced, one for the primary server and one for a secondary. These are set up to be included from the *named.boot* file for the nameservers.

Finally for all the hosts that did not have the unique-alias flag, a file is produced that has the fully qualified hostname and the groupname to which it is to be translated. This is especially important if you have set up a namespace where hosts in one domain may be split among several mail domains.

Mail File Processing

Mail delivery for a group is controlled by the users file in the group. As mentioned above, we use a uumap style, in which a record is built up of key: value pairs, one per line, and a blank line separates records. This is a very general format that allows us to have many types of optional data and allows for future format changes.

There are currently three required keywords for each user: *username*, *mailname* and *mailhome*. The *username* is the name of the account to which mail is sent and received within the group, and it must be unique within the group. The *mailname* is the name to which the *username* is transformed when it passes through the mail relay, and also must be unique within the maildom. It is fine to have the *username* and *mailname* be the same, but both must be specified. The *mailhome* is the host to which mail for this user is to be delivered, and must be a fully qualified hostname. This allows different machines to receive mail for subsets of the users within a group.

A special record is supported with the *mailname* and *username* both set to ''*''. This is the place where all mail to a group that fails to match either a *username* or a *mailname* is sent. This allows mail aliases to work without explicit listing in the users file.

In addition to user records we support a record with a *default* keyword. The idea is that the default record is appended to each record that follows, with any duplicate keys being set to the user record's value. This allows much of the non-unique data in the file to be specified only once. There are two values for the default key, *replace* and *merge*. *Replace* clears the current default record and adds the contents of this record. *Merge* adds or replaces the keywords in the current default record.

Here is an input users file:

```
default:       replace
organization:  usenix
scope:         US
mailhome:      mail1.us.lisa.usenix.org

username:      scharf
mailname:      Jerry_Scharf

username:      vixie
mailname:      Paul_Vixie
mailhome:      ws1.us.lisa.usenix.org

default:       merge
scope:         everywhere

username:      nile
mailname:      Martin_Nile

default:       replace
mailhome:      prim.us.lisa.usenix.org

username:      brister
mailname:      James_Brister
```

and here is the expanded output after processing:

```
mailname: Jerry_Scharf
username: scharf
mailhome: mail1.us.lisa.usenix.org
scope: US
organization: usenix

mailname: Paul_Vixie
mailhome: ws1.us.lisa.usenix.org
username: vixie
scope: US
organization: usenix

mailname: Martin_Nile
username: nile
mailhome: mail1.us.lisa.usenix.org
scope: everywhere
organization: usenix

mailname: James_Brister
username: brister
mailhome: prim.us.lisa.usenix.org
```

It is important to notice that all groups that have the same maildom must be processed at once and all uniqueness tests are done across a mail domain, not a single group.

Once all the files for all the mail domains are loaded, a file is generated for each maildom with each user having a fully expanded record. A three columned file is produced specific to the mapping and sendmail configuration we support at Sony. The first column is username<@groupaddr>, the second mailname<@groupaddr> and the third one username<@mailhome>. The mapping from column one to column two is used on mail leaving the group, and the mapping from column two to column three is for mail going to the group. The reason for the difference between column one and column three is that we want to remap the username from any machine in the group, so we first map the user@host.group.dom to user@group.dom, them map that to mailname@group.dom.

The Mail Delivery System

For each user in each group, a mail target machine is defined. The mail targets for the groups are a list of mail relays. Mail destined for a specific machine is delivered directly to the machine. Mail destined for the group is delivered to a mail relay, which looks up the address, determines the machine and account name it should be sent to, and sends it on.

The current system uses IDA sendmail with *dbm* support and custom *sendmail.cf* files to do the work. The *dbm* databases provide the following mappings:

```
hostname->groupname
       for header rewriting
user@groupname->fullname@groupname
       for header rewriting
user@groupname->user@mailhost.groupname
       for envelope rewriting
fullname@groupname->user@mailhost.groupname
       for envelope rewriting
obsolete_address->new_address
       for header and envelope rewriting
```

Directory Services

Our first directory service is a *whois* server based on a simple *Perl* script. Each night, the expanded users database is processed, merged with data from other mailers that have SMTP attachment, and written to a flat file with artificial handles. The *Perl whois* server does something similar to a grep of the file for the string, and returns matches.

Deployment

SENDS was put into production October of 1993. The current revision has been running since April of 1994. We have in that time been supporting two separate naming structures simultaneously.

SENDS is managing about 50 distinct groups inside Sony in the US. There are 3 separate corporate entities using it, each with their own substructure. There are approximately 250 DNS zones generated, 500 users and the obsolete address forwarding list is about 1300 entries. These all seem modest from our past experience, and we have had no serious scaling problems to this point.

The regional mail relay systems are too lightly loaded to show any penalties of the database lookups, passing about 2000 messages a day. We will need to get in the range of 2000 messages per hour per machine before we can study those effects.

We have set up the cron job to run the file scanner every 10 minutes. Our DNS records have a Time To Live set at 30 minutes for both zones and cache entries. This guarantees convergence in 70 minutes from when a file is submitted, with the average in the 20-30 minute range. Our mail databases are loaded hourly, so we have a similar convergence time there.

Our experience in setting up novice groups to run SENDS and mail have been very good. If we have seen the O/S and version before, it is usually one to two hours to create the groups in SENDS, load the hosts and users files, set up one group mail server for sending and receiving and let the data propagate through the network.

Conclusions

SENDS is a simple and powerful tool for maintaining the DNS namespace and e-mail administration. SENDS has proven ample for Sony's current and near term needs. The groups are pleased with the response time to changes, and Corporate Networks staff are pleased with a reduction of workload in managing and tracking of DNS names. The integration and ease of use has also caused the groups to be more diligent at maintaining their data.

We would recommend SENDS to any organization who had more than a few separate administrative groups with IP machines. The key assumption of this design is that the individual groups maintain control of allocation of their section of the IP and DNS spaces. If that doesn't follow the model your organization wants, SENDS is not the tool for you.

The directory services are still new at Sony and not fully developed. We will need to develop a better idea of which information is needed in the directory, and which access methods work best. There will need to be a tool that will allow the group systems administrator to delegate the directory maintenance to the individual users. We have tried to make the system independent of naming structure, but this will only be tested when others attempt to implement their designs with these tools.

We have been pleasantly surprised a couple times by what we have been able to persuade SENDS to do. Though we would not recommend using SENDS as a mailing list administration tool, we set up a pager maildom with a list of users and their

Skyword pins, so people can mail to scharf@pager... and get a message to me.

Because Sony's use of TCP/IP based machines is modest, we have kept the tools simple at the cost of speed. We can process all of our data in under 1 minute, so things like caching *Perl* internal tables have not been done. If someone needs to manage a set of machines many times the size, this should be considered.

Acknowledgements

We would like to thank several people. First and foremost we would like to thank Sandi Resnikoff, whose skills and commitment to quality have allowed us to accomplish what we have. James Brister did major work on the code. Tim Guarnieri, Alison Nicklin and Cindy Goral did proofreading, blame us for the mistakes, thank them for the quality. Thanks also to Martin Nile, PJ DeFrance and the San Jose networks group for making this all work.

Availability

SENDS is available with a license that says that changes must be provided back to the maintainers and may not be resold. This is to keep the number of forks to a minimum. The files are available for anonymous FTP from ftp.vix.com in /pub/vixie/SENDS/sends.tar.gz

Author Information

Jerry Scharf is currently Strategic Planner for Corporate Networks at Sony Electronics Corp, and also manages Sony's Internet gateway. He has also worked as a system administrator and planner at Digital Equipment Corp., Nasa Ames Research Center and Cornell University. He did his undergraduate studies in physics at Pennsylvania State University. Reach him via US Mail at 3300 Zanker Rd, San Jose, CA 95134. His electronic mail address is <scharf@sony.com>

Paul A. Vixie is a system, software and network consultant in the San Francisco Bay Area. From 1988 to 1993 he worked for Digital Equipment Corporation's Western Research Laboratory ("DECWRL") where among other things he catapulted the largely unknowing and unwilling corporation into the Internet limelight by creating gatekeeper.dec.com, now one of the largest and busiest Anonymous FTP hosts in the world. He is the author of BSD's cron daemon and of King James Sendmail; he is also the current maintainer of the BIND domain name server. Reach him via US Mail at 116 Stanley St., Redwood City, CA 94062, USA. His electronic mail address is <paul@vix.com>

References

[1] Paul Albitz and Cricket Liu, ''DNS and BIND,'' O'Reilly & Assoc.

[2] Bryan Costales, Eric Allman and Niel Rickert, ''Sendmail,'' O'Reilly & Assoc.

[3] D. Crocker, ''RFC822: Standard for the format of ARPA Internet text messages,''.

[4] V. Fuller, T. Li, J. Yu, K. Varadhan, ''RFC1519: Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy''.

[5] D. Mills, ''RFC1305: Network Time Protocol (v3)''.

[6] P. Mockapetris, ''RFC1101: DNS encoding of network names and other types''.

[7] P. Mockapetris, ''RFC1035: Domain names - implementation and specification''.

[8] J. Postel, ''RFC821: Simple Mail Transfer Protocol''.

[9] J. Postel and J. Reynolds, ''RFC959: File Transfer Protocol''.

Appendix A: SENDS config files

An example global config file:

```
# Config file for SENDS.
#

# address of primary nameserver for the zones we control. A dotted-quad is
# REQUIRED
primary         10.0.1.10

zonemask        10.0.0.0 0xffffff00
zonesize        10.0.0.0 3       # in octets

# fields in a user entry that may not have default values REQUIRED
user_key_no_defaults    username mailname

# fields in a user entry that may not have duplicates REQUIRED
user_key_no_dups        username mailname

# fields in a user entry that have to exist REQUIRED
user_key_min_fields     username mailname mailhome

# names of the name servers (primary and secondary) for our zone.
# These entries go in the zone boot file for named.
name_servers    prim.us.lisa_test.usenix.org. sec.us.lisa_test.usenix.org. sec.t
hem.lisa_test.usenix.org.

# hostmaster entry for the SOA record. REQUIRED
hostmaster      hostmaster.lisa_test.usenix.org.

# master host for the zones SOA record. REQUIRED.
zone_master     prim.us.lisa_test.usenix.org.

# zone expiration value. OPTIONAL--default is 604800
zone_expire     604800

# zone retry value. OPTIONAL--default is 600
zone_retry      600

# zone refresh value for the SOA record. OPTIONAL--default is 1800
zone_refresh    1800

# zone minimum value for SOA record. OPTIONAL--default is 1800
zone_minimum    1800

# directory for putting temp files. OPTIONAL--/var/tmp/is default.
#tmpdir         /var/tmp

# address for mailing copies of processing log and data file
# notifications. REQUIRED
notify_addr     scharf

# location of sendmail binary. OPTIONAL--default is first value
# found on PATH.
sendmail        /usr/sbin/sendmail

# predefined domains that hostnames cannot match against. OPTIONAL
bad_domains     foo.gov

# debug         1
#dont_mail      1

whois_file      data/whois
```

An example group control file:

```
dri:            scharf@us.lisa.usenix.org
dri:            vixie@us.lisa.usenix.org
relay:          mail1.us.lisa.usenix.org.
relay:          mailx.them.lisa.usenix.org.
```

```
network:        10.0.(0-30).(0-254)    accept
dots-allowed:   no
```

Appendix B

These are sample input and output files for the group ‘‘us’’ in domain ‘‘lisa.usenix.org’’. This was run with the control files in Appendix A.

Host file for the group ‘‘us’’:

```
127.0.0.1       localhost loghost
#
10.0.1.0        service-net     # this is a comment
10.0.1.10       prim dns_root ns
10.0.1.11       sec dns_sec
10.0.1.12       mail1
10.0.1.20       big_serv nfsserv nfsserv_ef0
#
10.0.10.20      big_serv nfsserv nfsserv_ef1
10.0.10.30      ws1
```

The users file for the group ‘‘us’’, and was used as the example of the default record expansion, and can be found in the Mail File Processing section of the paper.

This is the mail file sent for processing the hosts file in ‘‘us’’:

```
To: scharf@us.lisa.usenix.org, vixie@us.lisa.usenix.org
Cc: scharf@morality.sjc.hw.Sony.COM
Subject: new "hosts" file for [lisa.usenix.org, us] received by SENDS

-------- installation
/home/scharf/net/LISA/testrun/data/lisa.usenix.org/us/work/hosts:1:
    127.0.0.1 is not in your network ACL (ignored)
      127.0.0.1   localhost loghost
-------- end of installation

succeeded - accepting input file
```

Here is the DNS zone file for us.lisa.usenix.org:

```
; this file was automatically generated - do not edit
$ORIGIN us.lisa.usenix.org.
@                                IN MX   1 mail1.us.lisa.usenix.org.
                                 IN MX   2 mailx.them.lisa.usenix.org.
localhost                        IN A    127.0.0.1
service-net                      IN PTR  0.1.0.10.in-addr.arpa.
prim                             IN A    10.0.1.10
                                 IN MX   0 prim
                                 IN MX   1 mail1.us.lisa.usenix.org.
                                 IN MX   2 mailx.them.lisa.usenix.org.
dns_root                         IN CNAME prim
ns                               IN CNAME prim
sec                              IN A    10.0.1.11
                                 IN MX   0 sec
                                 IN MX   1 mail1.us.lisa.usenix.org.
                                 IN MX   2 mailx.them.lisa.usenix.org.
dns_sec                          IN CNAME sec
mail1                            IN A    10.0.1.12
                                 IN MX   0 mail1
                                 IN MX   1 mail1.us.lisa.usenix.org.
                                 IN MX   2 mailx.them.lisa.usenix.org.
big_serv                         IN A    10.0.1.20
                                 IN MX   0 big_serv
                                 IN MX   1 mail1.us.lisa.usenix.org.
                                 IN MX   2 mailx.them.lisa.usenix.org.
```

```
nfsserv                         IN A    10.0.1.20
                                IN MX   0 nfsserv
                                IN MX   1 mail1.us.lisa.usenix.org.
                                IN MX   2 mailx.them.lisa.usenix.org.
nfsserv_ef0                     IN A    10.0.1.20
                                IN MX   0 nfsserv_ef0
                                IN MX   1 mail1.us.lisa.usenix.org.
                                IN MX   2 mailx.them.lisa.usenix.org.
big_serv                        IN A    10.0.10.20
nfsserv                         IN A    10.0.10.20
nfsserv_ef1                     IN A    10.0.10.20
                                IN MX   0 nfsserv_ef1
                                IN MX   1 mail1.us.lisa.usenix.org.
                                IN MX   2 mailx.them.lisa.usenix.org.
ws1                             IN A    10.0.10.30
                                IN MX   0 ws1
                                IN MX   1 mail1.us.lisa.usenix.org.
                                IN MX   2 mailx.them.lisa.usenix.org.
```

Here is the header file for the zone us.lisa.usenix.org:

```
; this file was automatically generated - do not edit
$ORIGIN us.lisa.usenix.org.
@        IN SOA  prim.us.lisa_test.usenix.org. hostmaster.lisa_test.usenix.org. (
                               1994072900      ; Serial
                               1800            ; refresh
                               600             ; retry
                               604800          ; expire
                               1800            ; minimum
                        )
         IN NS   prim.us.lisa_test.usenix.org.
         IN NS   sec.us.lisa_test.usenix.org.
         IN NS   sec.them.lisa_test.usenix.org.

$INCLUDE sends/us.lisa.usenix.org.data
```

This is the reverse lookup zone for subnet 10.0.1.x:

```
; this file was automatically generated - do not edit
$ORIGIN 1.0.10.in-addr.arpa.
0.1.0.10.in-addr.arpa.          IN PTR  service-net.us.lisa.usenix.org.
                                IN A    255.255.255.0
10.1.0.10.in-addr.arpa.         IN PTR  prim.us.lisa.usenix.org.
11.1.0.10.in-addr.arpa.         IN PTR  sec.us.lisa.usenix.org.
12.1.0.10.in-addr.arpa.         IN PTR  mail1.us.lisa.usenix.org.
20.1.0.10.in-addr.arpa.         IN PTR  big_serv.us.lisa.usenix.org.
                                IN PTR  nfsserv.us.lisa.usenix.org.
                                IN PTR  nfsserv_ef0.us.lisa.usenix.org.
```

Here are the primary and secondary boot files to be included from named.boot.

```
; this file was automatically generated - do not edit
; it is intended to be included from a named.boot file
primary 1.0.10.in-addr.arpa             sends/10.0.1
primary 10.0.10.in-addr.arpa            sends/10.0.10
primary us.lisa.usenix.org              sends/us.lisa.usenix.org

; this file was automatically generated - do not edit
; it is intended to be included from a named.boot file
secondary 1.0.10.in-addr.arpa             10.0.1.10 sends/10.0.1
secondary 10.0.10.in-addr.arpa            10.0.1.10 sends/10.0.10
secondary us.lisa.usenix.org              10.0.1.10 sends/us.lisa.usenix.org
```

Finally, here is the three columned file used for mail address transformations. When mail is sent out from a group, sending addresses that match column 1 are replaced with column 2. Inbound mail that matches either column 1 or column 2 are sent to the address in column 3.

```
scharf<@us.lisa.usenix.org> Jerry_Scharf<@us.lisa.usenix.org> scharf<@mail1.us.lisa.usenix.org>
vixie<@us.lisa.usenix.org> Paul_Vixie<@us.lisa.usenix.org> vixie<@ws1.us.lisa.usenix.org>
nile<@us.lisa.usenix.org> Martin_Nile<@us.lisa.usenix.org> nile<@mail1.us.lisa.usenix.org>
brister<@us.lisa.usenix.org> James_Brister<@us.lisa.usenix.org> brister<@prim.us.lisa.usenix.org>
```